Optimization-Based Path-Velocity Control for Time-Optimal Path Tracking under Uncertainties

Jia, Zheng; Karayiannidis, Yiannis; Olofsson, Björn

Published in:
2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2025)

2025

Optimization-Based Path-Velocity Control for Time-Optimal Path Tracking under Uncertainties

Zheng Jia[1], Yiannis Karayiannidis[1], and Björn Olofsson[1,2]

Abstract— **This paper addresses the path-tracking problem of time-optimal trajectories under model uncertainties, by proposing a real-time predictive scaling algorithm. The algorithm is formulated as a convex optimization problem, designed to balance the trade-off between improving feasibility and time optimality of a trajectory. The predicted trajectory is scaled based on the presence of path segments that are particularly sensitive to model uncertainties within the prediction horizon. Numerical simulations and experiments demonstrate that the proposed scaling algorithm reduces the path traversal time, while preserving similar path-tracking accuracy compared to an existing non-predictive method.**

I. Introduction

Following a prescribed path as fast as possible, while guaranteeing high path-tracking accuracy is important for maximizing the production rate with high quality in robotic applications. This may be addressed by first solving the time-optimal path-tracking problem [1]–[6], and second tracking the trajectories by available high-performance robot controllers [7]. However, model uncertainties may cause the planned trajectories become not realizable because of the physical limits of the robot; time-optimal motions may be occasionally or always at their torque limits, which leaves no control margin for feedback control [8]–[16].

One effective way to address this problem is trajectory scaling, i.e., adjusting the movement speed to comply with the possible kinematic, dynamic, and torque constraints [8]–[23]. Both offline and online methods have been explored within this framework. In [8], the adjustment is done offline by uniformly scaling the motions by a constant factor. Alternatively, parametric assumptions on model uncertainties can be made for robust trajectory generation [6]. As model uncertainties are not known a priori, optimistic or conservative assumptions may be made in offline methods. To improve time optimality and handle unexpected disturbances, online adjustment methods may be adopted, which can be either non-predictive or predictive [9]–[24]. Non-predictive methods adjust the time scaling for current constraint satisfaction, whereas predictive methods consider the evolution of states over a horizon. The advantages of non-predictive methods include fast computation and flexibility [9]–[15], [17]–[19]. In particular, Path-Velocity Control (PVC) in [11] is a component external to the robot controller, which modifies the reference signal, thus significantly simplifying integration with existing control frameworks (a detailed stability analysis and further developments in handling path deviations along the transversal directions of the path are presented in [9], [15]). On the other hand, predictive scaling in the Model Predictive Control (MPC) framework has shown its advantages over the non-predictive methods in terms of shorter achievable path traversal time and better fulfillment of state and input constraints, at the cost of possible challenges in implementation [16], [20]–[24].

In this paper, we propose an online predictive trajectory scaling algorithm, referred to as optimization-based path-velocity control (OPVC), to address the path-tracking problem of time-optimal trajectories. Unlike the MPC formulations in [16], [20]–[24], the proposed method performs the prediction by formulating an optimization problem similar to [4], with a solution inspired by [8]. The algorithm integrates information about the location of areas particularly sensitive to uncertainties, in the vicinity of the so-called critical and tangency points [1], [3], which, as discussed in the paper, are closely related to trajectory feasibility. The proposed algorithm is computationally fast and can be integrated with existing robot controllers as an external component, providing both flexibility and speed in real-time robotic applications. The key contributions of the paper are:

1) A real-time predictive trajectory scaling algorithm for the time-optimal path-tracking problem.
   - Flexibility: As an external component, OPVC only modifies the reference signal to the robot controller.
   - Constraint satisfaction & time optimality: As an optimization-based method, OPVC aims to minimize path traversal time, while fulfilling torque constraints, in a receding-horizon fashion.
   - Computational speed: OPVC is lightweight and can run on a Franka Emika Panda robot at 1 kHz.
2) A strategy to balance the trade-off between improving trajectory feasibility and time optimality.
   - Through simulations and experiments, we observe that time-optimal trajectories are more prone to infeasibility around critical and tangency points as a result of model uncertainties.
   - We demonstrate that integrating information about the locations of these points enables OPVC (and potentially other scaling methods) to determine when trading off time optimality is necessary to improve trajectory feasibility.

This work was supported by the ELLIIT Strategic Research Area, and partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

[1]Zheng Jia, Yiannis Karayiannidis, and Björn Olofsson are with the Department of Automatic Control, Lund University, Sweden, {zheng.jia, yiannis, bjorn.olofsson}@control.lth.se.

[2]Björn Olofsson is also with the Division of Vehicular Systems, Department of Electrical Engineering, Linköping University, Sweden.

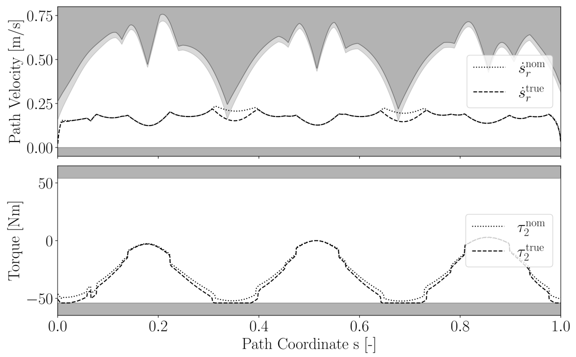


Fig. 1: Trajectory of the path tracking of a cylindrical path based on the nominal (dotted line) and true (dashed line) robot models. In the top subplot, the dark gray area indicates the region above the maximum velocity curve (9) of the nominal model. The light gray area is caused by model uncertainties.

II. Problem Formulation

Consider a robot manipulator with joint variables denoted by $\boldsymbol{q}$ and its dynamical model given by [25], [26]

$$\boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{G}(\boldsymbol{q}) = \boldsymbol{\tau}, \tag{1}$$

where $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{G}$ are the inertia matrix, Coriolis and gravity vectors, respectively. Let $\boldsymbol{q}_r = \boldsymbol{f}(s)$, where $s \in [0, 1]$, be a reference joint path where the time scaling $s(t)$ determines the motion along the path. The torque $\boldsymbol{\tau}$ that can actuate the robot to follow the reference path can be parameterized in the form [11]

$$\boldsymbol{\tau} = \boldsymbol{\beta}_1(\boldsymbol{q}, s)\ddot{s} + \boldsymbol{\beta}_2(\boldsymbol{q}, \dot{\boldsymbol{q}}, s, \dot{s}), \tag{2}$$

where the dependence on $\ddot{s}$ is a result of the existence of a reference acceleration in the controller. Note that (2) encompasses a variety of common control laws with feed-forward acceleration terms, see examples in [11]. Note also that the same parameterization is used when offline computing optimal trajectories, albeit the dependence on $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$ in $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$ is replaced by s and $\dot{s}$, by setting $\boldsymbol{q} = \boldsymbol{q}_r$ [1], [4]–[6].

In robot control and trajectory optimization problems, the torque required to follow a path or trajectory is constrained as follows

$$\boldsymbol{\tau}_{\min} \le \boldsymbol{\tau} \le \boldsymbol{\tau}_{\max}. \tag{3}$$

Additional kinematic constraints are introduced as

$$\dot{\boldsymbol{q}}_{\min} \le \dot{\boldsymbol{q}}_r \le \dot{\boldsymbol{q}}_{\max}, \tag{4}$$
$$\ddot{\boldsymbol{q}}_{\min} \le \ddot{\boldsymbol{q}}_r \le \ddot{\boldsymbol{q}}_{\max}. \tag{5}$$

Using (2) in (3) and the chain rule $\dot{\boldsymbol{q}}_r = \boldsymbol{f}'(s)\dot{s}$, $\ddot{\boldsymbol{q}}_r = \boldsymbol{f}'(s)\ddot{s} + \boldsymbol{f}''(s)\dot{s}^2$ in (4)–(5), where $(\cdot)'$ denotes $\frac{\mathrm{d}}{\mathrm{d}s}(\cdot)$, the constraints can be transformed to constraints on $\ddot{s}$ and $\dot{s}$

$$\ddot{s}_{\min}(\boldsymbol{\beta}_1, \boldsymbol{\beta}_2) \le \ddot{s} \le \ddot{s}_{\max}(\boldsymbol{\beta}_1, \boldsymbol{\beta}_2), \tag{6}$$
$$0 \le \dot{s} \le v_{\max}(\dot{\boldsymbol{q}}_{\min}, \dot{\boldsymbol{q}}_{\max}), \tag{7}$$
$$a_{\min}(\ddot{\boldsymbol{q}}_{\min}, \ddot{\boldsymbol{q}}_{\max}) \le \ddot{s} \le a_{\max}(\ddot{\boldsymbol{q}}_{\min}, \ddot{\boldsymbol{q}}_{\max}), \tag{8}$$

where $\ddot{s}_{\min}$ and $\ddot{s}_{\max}$ are dynamic path-acceleration limits, $v_{\max}$, $a_{\min}$ and $a_{\max}$ are the kinematic bounds, and $\dot{s}$ is constrained to be non-negative for a forward motion. Note that (3) not only defines an admissible control interval of $\ddot{s}$ (6), but also implies an admissible state region of $\dot{s}$

$$\dot{s} \le \dot{s}_{\max} = \underset{\boldsymbol{\tau}_{\min} \le \boldsymbol{\tau} \le \boldsymbol{\tau}_{\max}}{\operatorname{argmax}} \quad \dot{s} \tag{9}$$

where $\dot{s}_{\max}$ is the maximum-velocity curve on which $\ddot{s}_{\min} = \ddot{s}_{\max}$ [1], [3]. In trajectory optimization, the limits in (6)–(9) and the nominal solutions, here denoted as $\dot{s}_r$ and $\ddot{s}_r$, are computed offline assuming accurate robot models [1], [4]–[6]. However, when model uncertainties cause the limit in (6) and (9) to deviate from their nominal values, it is possible that $\dot{s}_r$ and $\ddot{s}_r$ become infeasible. An example is shown in Fig. 1, where the superscripts $(\cdot)^{\text{nom}}$ and $(\cdot)^{\text{true}}$ are used to distinguish the trajectory computed from the nominal and true robot models. The nominal reference path velocity $\dot{s}_r^{\text{nom}}$ is not feasible at $s \approx 0.33$ and $s \approx 0.66$ because

$$\dot{s}_r^{\text{nom}} > \dot{s}_{\max}^{\text{true}}, \tag{10}$$

and tracking $\dot{s}_r^{\text{nom}}$ causes bound-inversion problems [11]

$$\ddot{s}_{\min}(\boldsymbol{\beta}_1, \boldsymbol{\beta}_2) > \ddot{s}_{\max}(\boldsymbol{\beta}_1, \boldsymbol{\beta}_2). \tag{11}$$

Note that (10) appears in path segments particularly sensitive to model uncertainties, where $\dot{s}_r^{\text{nom}}$ is near $\dot{s}_{\max}^{\text{nom}}$, i.e., so-called critical and tangency points [1], [3]. Consequently, $\boldsymbol{\tau}$ saturates the torque constraint (3), which also occurs when $\ddot{s}_r^{\text{nom}}$ is not feasible, leading to path deviation. Thus, the problem considered in this paper is addressing the path-tracking problem of time-optimal trajectories, when the actual limits in (6) and (9) differ from their nominal values.

A. Path-Velocity Control

Computing a feasible $\ddot{s}$ within the limits in (6) that do not coincide with the nominal values can be solved by PVC, where the limits are re-computed online and used as the lower and upper bounds in a saturation function $\operatorname{sat}(\cdot; \ddot{s}_{\min}, \ddot{s}_{\max})$

$$\ddot{s} = \operatorname{sat}(u; \ddot{s}_{\min}, \ddot{s}_{\max}), \tag{12}$$
$$u = \ddot{s}_r(s) + \frac{\alpha}{2}\left(\dot{s}_r^2(s) - \dot{s}^2\right), \tag{13}$$

which asymptotically tracks $\dot{s}_r$ with α as a control parameter, whenever $\operatorname{sat}(\cdot; \ddot{s}_{\min}, \ddot{s}_{\max})$ is not active [9]–[12].

B. Adaptive Trajectory Scaling

The state constraint (9), however, is not considered in (12)–(13). To reduce the risk of the problem (10)–(11) occuring, a velocity margin to $\dot{s}_{\max}$ can be created by scaling $(\ddot{s}_r, \dot{s}_r)$ down by a factor γ that is appropriately adapted [9]–[12]

$$u = \gamma^2\ddot{s}_r(s) + \frac{\alpha}{2}\left(\gamma^2\dot{s}_r^2(s) - \dot{s}^2\right). \tag{14}$$

By constraining γ to $(0, 1]$ and assuming $\dot{\gamma}$ is sufficiently small, (7)–(8) can also be satisfied

$$0 \le \gamma\dot{s}_r \le v_{\max}, \tag{15}$$
$$a_{\min} \le \gamma^2\ddot{s}_r \le a_{\max}. \tag{16}$$

In [11], a feedback law is designed to adapt γ

$$\dot{\gamma} = \begin{cases} k_\gamma\left(\dot{s} - \gamma\dot{s}_r\right)\hat{\gamma}, & \gamma\dot{s}_r \ge \dot{s} \\ 0, & \gamma\dot{s}_r < \dot{s} \end{cases} \tag{17}$$

where k_γ controls the adaptation rate and $\hat{\gamma} = \frac{\dot{s}}{\dot{s}_r}$. When $\gamma \dot{s}_r \geq \dot{s}$, it indicates that the path-acceleration capability, limited by the torque constraint and model uncertainties, is insufficient to track the scaled solution $\gamma \dot{s}_r$, and γ is adapted to decrease and create a larger margin. However, since a trade-off exists between increasing the margin and maximizing path traversal speed, the adaptive law (17) with a focus on creating the margin may lead to overly conservative scaling even if it is unnecessary. Additionally, as a non-predictive method, the future path information is not used in (17). Furthermore, while the locations of critical and tangency points are known, this information is not considered in (17). The predictive method OPVC proposed in this paper aims at improving the balance between minimizing the risks of infeasibility and path traversal time of trajectories, based on the presence of the critical and tangency points within the prediction horizon.

III. Optimization-based Trajectory Scaling

This section describes the motivation and formulation of the proposed method. The idea is based on the observation that the output of robot controllers can be expressed as an approximation of the sum of the feedforward term $\boldsymbol{\tau}_{\text{ff}}$ and the feedback term $\boldsymbol{\tau}_{\text{fb}}$, when the robot stays sufficiently close to the reference trajectories. The torque $\boldsymbol{\tau}_{\text{ff}}$ can be evaluated over a horizon and separated into a scalable part parameterized by the scaling factor γ and a non-scalable part that is not affected by γ. Information about the locations of the critical and tangency points over the horizon will be included in the proposed scaling method.

A. Controller Approximation

An example of a controller that can be approximated as the sum of $\boldsymbol{\tau}_{\text{ff}}$ and $\boldsymbol{\tau}_{\text{fb}}$ is the computed-torque control [25], [26]

$$\boldsymbol{\tau}_{\text{ct}} = \hat{\boldsymbol{B}}(\boldsymbol{q})\left(\ddot{\boldsymbol{q}}_r + \Delta\boldsymbol{v}\right) + \hat{\boldsymbol{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \hat{\boldsymbol{G}}(\boldsymbol{q}), \tag{18}$$

where the notation $(\hat{\cdot})$ denotes the estimate of robot parameters, $\Delta\boldsymbol{v} = \boldsymbol{K}_q \Delta\boldsymbol{q} + \boldsymbol{D}_q \Delta\dot{\boldsymbol{q}}$ is the feedback control input, $\Delta\boldsymbol{q} = \boldsymbol{q}_r - \boldsymbol{q}$ is the joint position error, $\boldsymbol{K}_q$ and $\boldsymbol{D}_q$ are the gain parameters. Assume $\boldsymbol{q} \approx \boldsymbol{q}_r, \dot{\boldsymbol{q}} \approx \dot{\boldsymbol{q}}_r$ can be achieved by the controller (18), then (18) may be approximated as

$$\begin{aligned}\tilde{\boldsymbol{\tau}}_{\text{ct}} &= \hat{\boldsymbol{B}}(\boldsymbol{q}_r)\left(\ddot{\boldsymbol{q}}_r + \Delta\boldsymbol{v}\right) + \hat{\boldsymbol{C}}(\boldsymbol{q}_r, \dot{\boldsymbol{q}}_r) + \hat{\boldsymbol{G}}(\boldsymbol{q}_r) &(19)\\ &= \underbrace{\hat{\boldsymbol{B}}(\boldsymbol{q}_r)\ddot{\boldsymbol{q}}_r + \hat{\boldsymbol{C}}(\boldsymbol{q}_r, \dot{\boldsymbol{q}}_r) + \hat{\boldsymbol{G}}(\boldsymbol{q}_r)}_{\boldsymbol{\tau}_{\text{ff}}(\boldsymbol{q}_r, \dot{\boldsymbol{q}}_r, \ddot{\boldsymbol{q}}_r)} + \underbrace{\hat{\boldsymbol{B}}(\boldsymbol{q}_r)\Delta\boldsymbol{v}}_{\boldsymbol{\tau}_{\text{fb}}(\boldsymbol{q}_r, \Delta\boldsymbol{v})}. &(20)\end{aligned}$$

Using $\dot{\boldsymbol{q}}_r = \boldsymbol{f}'(s)\dot{s}$ and $\ddot{\boldsymbol{q}}_r = \boldsymbol{f}'(s)\ddot{s} + \boldsymbol{f}''(s)\dot{s}^2$ in (20), $\boldsymbol{\tau}_{\text{ff}}$ can be further expressed as a function of path variables

$$\boldsymbol{\tau}_{\text{ff}} = \boldsymbol{\Gamma}_1(s)\ddot{s} + \boldsymbol{\Gamma}_2(s)\dot{s}^2 + \hat{\boldsymbol{G}}(s), \tag{21}$$

where $\boldsymbol{\Gamma}_1(s) = \hat{\boldsymbol{B}}(s)\boldsymbol{f}'(s)$ and $\boldsymbol{\Gamma}_2(s) = \hat{\boldsymbol{B}}(s)\boldsymbol{f}''(s) + \hat{\boldsymbol{C}}\left(\boldsymbol{f}(s), \boldsymbol{f}'(s)\right)$. Then (20) can be written as

$$\tilde{\boldsymbol{\tau}}_{\text{ct}} = \boldsymbol{\Gamma}_1(s)\ddot{s} + \boldsymbol{\Gamma}_2(s)\dot{s}^2 + \hat{\boldsymbol{G}}(s) + \hat{\boldsymbol{B}}(s)\Delta\boldsymbol{v}, \tag{22}$$

where $\boldsymbol{\Gamma}_1\ddot{s} + \boldsymbol{\Gamma}_2\dot{s}^2$ is the scalable part, since $(\ddot{s}, \dot{s})$ can track the scaled solution $(\gamma^2\ddot{s}_r, \gamma\dot{s}_r)$ by (12) and (14). Then (22) can be used for verifying the feasibility of nominal solutions $(\ddot{s}_r, \dot{s}_r)$ with respect to (3) and $\Delta\boldsymbol{v}$. Conversely, a feasible pair $(\ddot{s}, \dot{s})$ may be determined by solving a convex optimal control problem similar to that in [4], which minimizes the path traversal time, $T = \int_0^1 \frac{1}{\dot{s}}\mathrm{d}s$, subject to kinematic constraints (7)–(8), dynamic constraint (22), torque limits (3), boundary conditions $\dot{s}^2(0) = \dot{s}^2(1) = 0$, and the differential constraint $\frac{\mathrm{d}}{\mathrm{d}s}\dot{s}^2 = 2\ddot{s}$

$$\begin{aligned}\underset{\ddot{s},\, \dot{s},\, \boldsymbol{\tau}}{\text{minimize}} \quad & \int_0^1 \frac{1}{\dot{s}}\mathrm{d}s &(23)\\ \text{subject to} \quad & \boldsymbol{\tau} = \boldsymbol{\Gamma}_1\ddot{s} + \boldsymbol{\Gamma}_2\dot{s}^2 + \hat{\boldsymbol{G}} + \hat{\boldsymbol{B}}\Delta\boldsymbol{v}, &(24)\\ & \dot{s}^2(0) = \dot{s}^2(1) = 0,\ \left(\dot{s}^2\right)' = 2\ddot{s}, &(25)\\ & \boldsymbol{\tau}_{\min} \leq \boldsymbol{\tau} \leq \boldsymbol{\tau}_{\max}, &(26)\\ & 0 \leq \dot{s}^2 \leq v_{\max}^2,\ a_{\min} \leq \ddot{s} \leq a_{\max}. &(27)\end{aligned}$$

B. Optimization-Based Trajectory Scaling

Let $\ddot{s}_r(s; \Delta\boldsymbol{v})$ and $\dot{s}_r(s; \Delta\boldsymbol{v})$ denote the optimal solution of (23)–(27) with explicit dependence on $\Delta\boldsymbol{v}$. In particular, when $\Delta\boldsymbol{v} = \boldsymbol{0}$, (23)–(27) is equivalent to the optimal control problem formulated in [4], and its solution can be denoted as $\ddot{s}_r(s; \boldsymbol{0})$ and $\dot{s}_r(s; \boldsymbol{0})$. Directly solving for $\ddot{s}_r(s; \Delta\boldsymbol{v})$ and $\dot{s}_r(s; \Delta\boldsymbol{v})$ is at least as computationally expensive as solving the optimal control problem in [4], which is typically solved offline in common practice (an online variant can be found in [27]). Alternatively, we propose to scale the offline solution $\gamma^2\ddot{s}_r(s; \boldsymbol{0})$ and $\gamma\dot{s}_r(s; \boldsymbol{0})$ to solve (23)–(27) for faster computation, and let

$$\boldsymbol{\tau}_\gamma = \left(\boldsymbol{\Gamma}_1\ddot{s}_r(s; \boldsymbol{0}) + \boldsymbol{\Gamma}_2\dot{s}_r^2(s; \boldsymbol{0})\right)\gamma^2 + \hat{\boldsymbol{G}} + \hat{\boldsymbol{B}}\Delta\boldsymbol{v}. \tag{28}$$

The sub-optimal solution $\boldsymbol{\tau}_\gamma(s, \Delta\boldsymbol{v})$ is a particular form of (22), where the boundary and differential constraints (25) are automatically satisfied, and (27) can be satisfied if $\gamma \in (0, 1]$. When the integration limits are changed to $[s, s + \Delta s]$, the problem can be transformed to minimizing over a constant γ in a receding horizon

$$\begin{aligned}\underset{\gamma}{\text{minimize}} \quad & \frac{1}{\gamma}\Delta T &(29)\\ \text{subject to} \quad & \boldsymbol{\tau}_{\min} \leq \boldsymbol{\tau}_\gamma \leq \boldsymbol{\tau}_{\max},\ 0 < \gamma \leq 1, &(30)\end{aligned}$$

where $\Delta T = \int_s^{s+\Delta s} \frac{1}{\dot{s}_r(s; \boldsymbol{0})}\mathrm{d}s$ is the nominal traversal time of Δs. Nonetheless, (29) becomes singular when $\gamma \approx 0$, and the torque constraint in (30) is non-convex in γ. This is addressed by transforming the problem to

$$\begin{aligned}\underset{\gamma^2}{\text{maximize}} \quad & \gamma^2 &(31)\\ \text{subject to} \quad & \boldsymbol{\tau}_{\min} \leq \boldsymbol{\tau}_\gamma \leq \boldsymbol{\tau}_{\max},\ 0 \leq \gamma^2 \leq 1, &(32)\end{aligned}$$

which is linear in γ^2. Except for the decision variable γ^2 and the measured quantity $\Delta\boldsymbol{v}$, the remaining terms are nominal values of robot parameters and the offline solution $\ddot{s}_r(s; \boldsymbol{0})$ and $\dot{s}_r(s; \boldsymbol{0})$, readily available before solving (31)–(32).

As the bound-inversion problem (11) often occurs near the critical and tangency points, information about their locations can be used by designing an additional path-dependent term in (31) after the discretization. Let N be the number of

TABLE I: Joint torque limits (Nm) for trajectory optimization and control.

Joint	1	2	3	4	5	6	7
Cylindrical	±54	±54	±54	±54	±12	±12	±12
Zig-Zag	±54	±54	±10	±54	±12	±12	±12

discrete coordinates over $[s, s+\Delta s]$, with $x=\gamma^2$ as the optimization variable. Assume an equal distance $\delta s = \frac{\Delta s}{N-1}$ is chosen, with $s_k = s + k\delta s$ as the k-th point, and $\boldsymbol{\tau}_k = \boldsymbol{\tau}_\gamma(s_k, \Delta \boldsymbol{v})$. OPVC solves the following discretized quadratic programming problem at each discrete-time instant i. The optimal solution is denoted as x_i^*.

$$\underset{x}{\text{maximize}} \qquad x - \frac{1}{N}\sum_{k=0}^{N-1} w(s_k)\left(x_{i-1}^* - x\right)^2 \tag{33}$$

$$\text{subject to} \quad \boldsymbol{\tau}_k = \left(\boldsymbol{\Gamma}_1 \ddot{s}_r + \boldsymbol{\Gamma}_2 \dot{s}_r^2\right)_k x + \hat{\boldsymbol{G}}_k + \hat{\boldsymbol{B}}_k \Delta \boldsymbol{v}, \tag{34}$$

$$\boldsymbol{\tau}_{\min} \le \boldsymbol{\tau}_k \le \boldsymbol{\tau}_{\max},\ k = 0, \dots, N-1, \tag{35}$$

$$0 \le x \le 1. \tag{36}$$

The additional term in (33) penalizes the increase of γ with a weighting function $w(s) \in \{w_l, w_h\}$, designed to take a high value w_h when near the critical and tangency points, and a low value w_l otherwise. Let $\hat{\gamma}_{\text{opt}} = \sqrt{x_i^*}$. Since $\Delta \boldsymbol{v}$ may contain disturbances and measurement noise, a low-pass filter $\frac{\mathrm{d}}{\mathrm{d}s}\gamma = k_\gamma(\hat{\gamma}_{\text{opt}} - \gamma)$ is applied, where k_γ controls the bandwidth. The filtered signal γ is then used in (12) and (14) for tracking scaled trajectories.

IV. Simulation

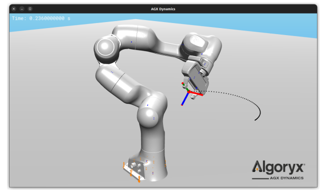
The proposed method is first evaluated on a simulated Franka Emika Panda robot in the Algoryx Dynamics simulation environment[1], based on the parameters identified in [28]. Torque limits in Table I and kinematic constraints[2] of the Panda robot are used for trajectory optimization (using the method in [4]) and control. Model uncertainties consist of differences in mass between the model and the simulated robot, where the latter has 10% higher masses for all links. Mass variations introduce deviations in the dynamic matrices $\boldsymbol{B}$, $\boldsymbol{C}$, and $\boldsymbol{G}$, e.g., when the manipulated object (treated as part of the end-effector) has uncertain mass and inertia properties. The joint path to follow corresponds to a cylindrical path in Cartesian space, as shown in Fig. 2. The path is given by $\boldsymbol{p}(s) = [0.484 + 0.2\cos(6\pi s), 0.2\sin(6\pi s), 0.25 s]$ (m), $s \in [0,1]$. The robot is controlled by (18), with gains $\boldsymbol{K}_q = \operatorname{diag}\left(\left[\omega_0^2, \dots, \omega_0^2\right]\right)$, $\boldsymbol{D}_q = \operatorname{diag}\left(\left[2\omega_0, \dots, 2\omega_0\right]\right)$, and frequency $\omega_0 = 40\pi$. Both simulation and control intervals are 1 ms. As a baseline for comparison, PVC (12)–(13) highlights which path segments the bound-inversion problem (11) may appear. The adaptive law (17) with estimate $\hat{\gamma} = \frac{\dot{s}}{\dot{s}_r}$ represents a class of non-predictive methods that adapt γ based on current path information and measurements.

A. Results

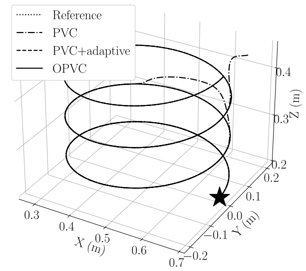
The simulation results comparing the path-tracking accuracy in Cartesian space are shown in Fig. 2. The

[1]February 25, 2025. Available: https://www.algoryx.se/agx-dynamics/

[2]February 25, 2025. Available: https://frankaemika.github.io/docs/

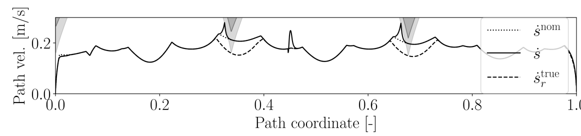


(a) Panda in Algoryx Dynamics

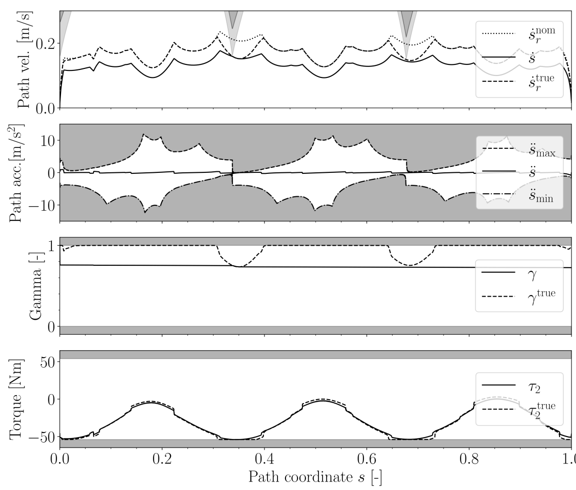


(b) End-effector position

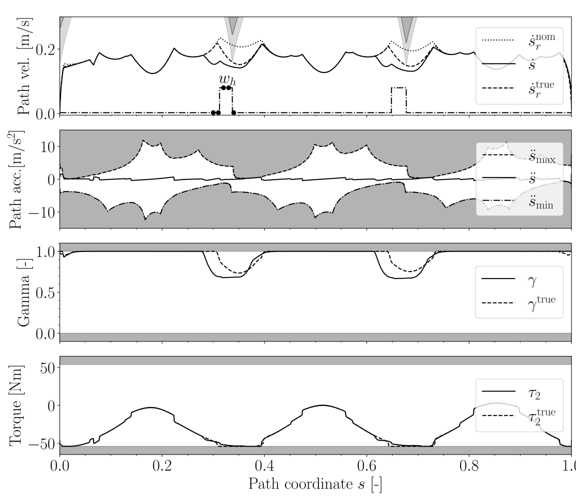
Fig. 2: Time-optimal path tracking of a cylindrical path. The starting point of the path is indicated by a black star. The lines representing Reference, PVC+adaptive, and OPVC overlap in the broader view.



(a) PVC: $\dot{s}$ enters the inadmissible region when tracking $\dot{s}_r^{\text{nom}}$.



(b) PVC+adaptive: γ decreases rapidly due to the high gain of k_γ.



(c) OPVC: γ is scaled down locally when $\dot{s}$ is near $\dot{s}_{\max}$.

Fig. 3: Simulation (cylindrical): comparison of results.

TABLE II: Simulation: RMSE (mm), Maximum Path-Tracking Error (mm), Task Execution Time (s), Percentage Changes (-), Average γ (-), and Parameters for Time-Optimal Path-Tracking of a Cylindrical Path.

Cylindrical	Metric						Parameters				
	RMSE	$e_{\max}$	T	Δp^{nom}	Δp^{true}	$\bar{\gamma}$	k_γ	w_h	w_l	Δs	N
Nominal	-	-	5.88	-	-	-	-	-	-	-	-
True	-	-	**6.09**	-	-	-	-	-	-	-	-
PVC	223	731	5.82	-1.1%	-4.5%	-	-	-	-	-	-
PVC+adapt.	1.22	2.04	**7.89**	34.3%	**29.5%**	0.74	1900	-	-	-	-
OPVC	1.22	2.09	**6.23**	6.0%	**2.2%**	0.94	200	1.5e3	50	4.0%	5

evolution of $\dot{s}$, $\ddot{s}$, γ, and τ_2 over s are shown in Fig. 3. The weighting function $w(s)$ scaled to fit the space is shown in Fig. 3c. Root mean squared errors (RMSE) and maximum path-tracking error $e_{\max} = \max_{0\leq s\leq 1} \|e(s)\|_2$ for the end-effector tip position, task execution time T, percentage changes in time $\Delta p^{\mathrm{nom}} = (T - T^{\mathrm{nom}})/T^{\mathrm{nom}}$ and $\Delta p^{\mathrm{true}} = (T - T^{\mathrm{true}})/T^{\mathrm{true}}$, the average value $\bar{\gamma} = \int_0^1 \gamma \mathrm{d}s$, and parameters are summarized in Table II.

Fig. 3a shows that the bound-inversion problem occurs at $s \approx 0.33$ and $s \approx 0.66$ because of model mismatch. As significant path deviation occurs, see the dashed-dotted line in Fig. 2b, the plots of $\ddot{s}$ and τ_2 are not shown here. The shape of $\dot{s}_r^{\mathrm{true}}$ indicates that trajectories need to be scaled down in advance, ideally only when approaching the critical points.

As seen in Fig. 3b, the adaptive law (17) scales down the nominal solution in advance to create a margin near the critical points. The parameter k_γ is incrementally increased from 0 until the bound-inversion issues are resolved (its minimum value is reported in Table II). However, as k_γ becomes larger, it results in earlier and excessive scaling down of $\dot{s}$. This is because (17) adapts γ when insufficient path acceleration has been measured, i.e., $\gamma\dot{s}_r > \dot{s}$. Such measurements at the start of the path tracking result in a substantial scaling down for the remainder of the path. It is possible to modify the adaptive law (17) to activate only near the critical points to improve time optimality. However, simulations indicate that measurements obtained near the critical points may not provide sufficient time to adapt γ; the margin may need to be established based on measurements obtained earlier, leading to overly conservative scaling.

Compared to the adaptive law (17), OPVC exploits the future path information by inspecting the nominal trajectory 4% ahead along s, see Table II, and determines a feasible $\hat{\gamma}_{\mathrm{opt}}$ compatible to $\Delta\boldsymbol{v}$. Fig. 3c shows that $\dot{s}$ circumvents the critical points, and restores back to tracking $\dot{s}_r^{\mathrm{nom}}$ afterwards. Note that the reduction of γ happens when $\ddot{s}$ is still within the path-acceleration limits, which may be difficult for non-predictive methods to achieve. It is observed that OPVC results in slightly more conservative γ than γ^{true} and (17) near the critical points. Note also that the plot of the weighting function and the locations of discrete points suggest that δs should be smaller than the segment length where w_h is applied, to ensure that the average weighting remains high when a critical point is within the horizon. The parameters w_l and w_h that achieve the shortest path traversal time are chosen. Overall, OPVC improves time optimality by reducing Δp^{true} from 29.5% to 2.2%, compared to the adaptive law (17). RMSE and $e_{\max}$ of OPVC are slightly higher than the adaptive law (17) because the effects of uncertainties increase with a higher path traversal speed, such as changes in the Coriolis effects.

TABLE III: Experiment (cylindrical): RMSE (mm), Maximum Path-Tracking Error (mm), Task Execution Time (s), Percentage Changes (-), Average γ (-), and Parameters for Time-Optimal Path-Tracking of a Cylindrical Path.

Cylindrical	Metric						Parameters				
	RMSE	$e_{\max}$	T	Δp^{nom}	γ_f	$\bar{\gamma}$	k_γ	w_h	w_l	Δs	N
Nominal	-	-	5.878	-	-	-	-	-	-	-	-
PVC	3.32	11.82	5.955	1.3%	-	-	-	-	-	-	-
PVC+adaptive	2.35	4.98	6.947	**18.2%**	0.738	0.847	200	-	-	-	-
OPVC	2.93	6.91	5.991	**1.9%**	0.999	0.976	200	5e3	50	4.0%	5

TABLE IV: Experiment (zig-zag): RMSE (mm), Maximum Path-Tracking Error (mm), Task Execution Time (s), Percentage Changes (-), Average γ (-), and Parameters for Time-Optimal Path-Tracking of a Zig-Zag Path.

Zig-Zag	Metric						Parameters				
	RMSE	$e_{\max}$	T	Δp^{nom}	γ_f	$\bar{\gamma}$	k_γ	w_h	w_l	Δs	N
Nominal	-	-	1.469	-	-	-	-	-	-	-	-
PVC	5.22	11.99	1.481	0.8%	-	-	-	-	-	-	-
PVC+adaptive	2.18	3.43	1.979	34.7%	0.682	0.730	200	-	-	-	-
OPVC	2.18	3.47	1.967	33.9%	0.999	0.751	200	5e3	100	6.0%	5

V. Experimental Results

The performance is further evaluated on the Franka Emika Panda robot in two experiments. Experiment 1 follows the same path as in the simulation. Experiment 2 follows a zig-zag path in Cartesian space, see Fig. 4. A video of two experiments is available online[3]. To evaluate the performance when more than one joint is at the torque limit, the limit of τ_3 is set to 10 Nm in Experiment 2, see Table I. The robot is controlled with a 1 ms control interval through the Franka Control Interface (FCI) and libfranka[4]. The library libfranka provides the internal numerical values of the robot model, here denoted as $\hat{\boldsymbol{B}}_{\mathrm{FE}}$, $\hat{\boldsymbol{C}}_{\mathrm{FE}}$, and $\hat{\boldsymbol{G}}_{\mathrm{FE}}$. OPVC is implemented using CVXGEN [29] and solves the optimization problem within each control interval of 1 ms. The controller used for the experiments is

$$\boldsymbol{\tau} = \hat{\boldsymbol{B}}_{\mathrm{FE}}(\boldsymbol{q})\ddot{\boldsymbol{q}}_r + \hat{\boldsymbol{G}}_{\mathrm{FE}}(\boldsymbol{q}) + \hat{\boldsymbol{F}}_{\mathrm{FE}}(\boldsymbol{q}) + \boldsymbol{K}_q\Delta\boldsymbol{q} + \boldsymbol{D}_q\Delta\dot{\boldsymbol{q}} \quad (37)$$

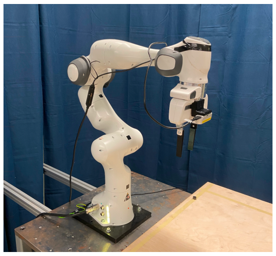
where $\boldsymbol{K}_q = \mathrm{diag}([800, 800, 800, 800, 250, 150, 50])$, $\boldsymbol{D}_q = \mathrm{diag}([60, 60, 60, 60, 30, 25, 15])$, and $\hat{\boldsymbol{F}}_{\mathrm{FE}}$ is the internal friction compensation by FCI. The controller omits the Coriolis vector and the pre-multiplication of $\hat{\boldsymbol{B}}_{\mathrm{FE}}$ to $\Delta\boldsymbol{v}$. Since FCI internally compensates $\hat{\boldsymbol{F}}_{\mathrm{FE}}$ but does not provide their numerical values, it is not used for the computation of the limits in (6).

A. Cylindrical Path

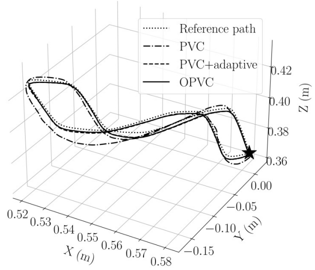
As seen in Fig. 5a, the same bound-inversion problems in simulations appear for the cylindrical path, indicating that time-optimal solutions are indeed more sensitive around these points. Fig. 5b shows that PVC+adaptive, as a non-predictive method, may scale down the path velocity more than necessary after the first critical point. Fig. 5c shows that OPVC obtains similar qualitative results to simulations; scaling down the trajectory only occurs when close to the critical points. The percentage change in time Δp^{nom} is

[3] February 25, 2025. Available: https://youtu.be/0sS3V1OyDjY

[4] February 25, 2025. Available: https://github.com/frankarobotics/libfranka



(a) Panda in experiments



(b) End-effector position

Fig. 4: Time-optimal path tracking of a zig-zag path in Cartesian space. The starting point of the path is indicated by a black star.

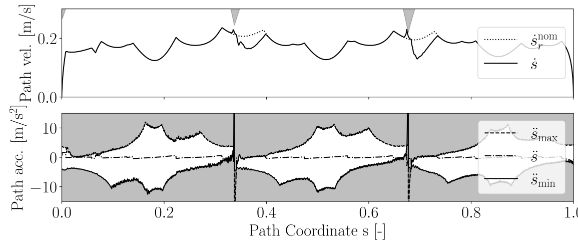
significantly lower than PVC+adaptive with similar path-tracking accuracy, see Table III.

B. Zig-Zag Path

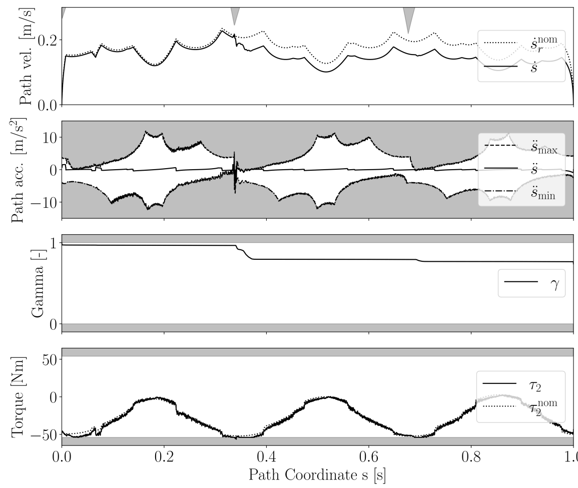
Fig. 6a shows that the time-optimal motion along the zig-zag path involves frequent acceleration and deceleration of the robot and contains six critical and tangency points. It is observed that all bound-inversion problems occur near these points. Fig. 6b shows that PVC+adaptive results in uniform scaling to circumvent all these points, without considering their precise locations. Comparing Fig. 6b and Fig. 6c, OPVC achieves higher trajectory speed $\dot{s}$ and better torque utilization than PVC+adaptive at ①④⑤, but reduces the speed more cautiously at ②③, when approaching critical and tangency points. This may be because OPVC is more sensitive than the adaptive law (17) to peak values of $\Delta \boldsymbol{v}$. In (17), these peaks are reflected in $\ddot{s}$ but are smoothed out in the velocity error $\dot{s} - \gamma \dot{s}_r$, after integrating $\ddot{s}$. While the number of critical and tangency points increases from two in the cylindrical path to six in the zig-zag path, OPVC still achieves shorter path traversal time and comparable path-tracking accuracy, see Table IV. The cautiousness of OPVC around the critical and tangency points can accumulate however, potentially affecting time optimality, as the number of critical and tangency points continues to increase. Despite this, the ability of OPVC to maximize trajectory speed elsewhere is still advantageous, and the strategy of balancing the trade-off based on the locations of critical and tangency points continues to be beneficial.

VI. Conclusion

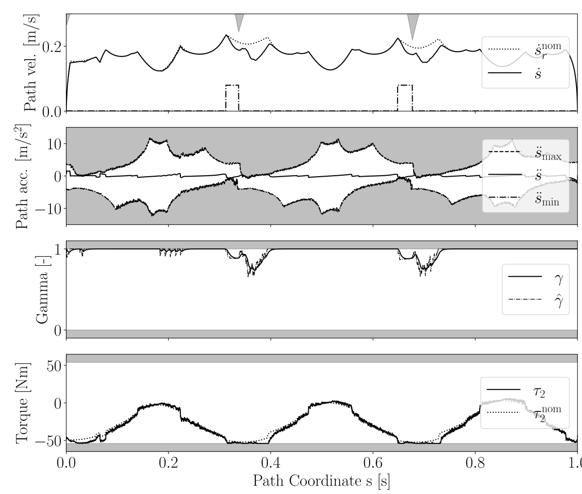
An online predictive trajectory scaling algorithm is presented, which aims to balance the trade-off between minimizing the risks of infeasibility and path traversal time in time-optimal path-tracking problems. The proposed algorithm leverages information about the locations of critical and tangency points, which are closely related to trajectory feasibility, in guiding the scaling process. Simulations and experiments show that the predictive algorithm reduces the risks of infeasibility by cautiously circumventing these points, and improves time optimality by scaling less conservatively when away from them. Additionally, the proposed algorithm is computationally fast and can be integrated with existing robot controllers, providing both flexibility and speed in real-time robotic applications. Future work may include introducing auxiliary optimization variables to reduce sensitivity to disturbance peaks and designing alternative weighting functions to further improve the performance.



(a) PVC: bound inversion occurs near the critical points.



(b) PVC+adaptive: γ decreases gradually along the path.



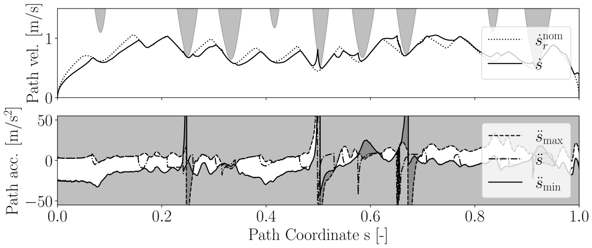
(c) OPVC: γ is scaled down locally when near the critical points.

Fig. 5: Experiment (cylindrical): comparison of results.

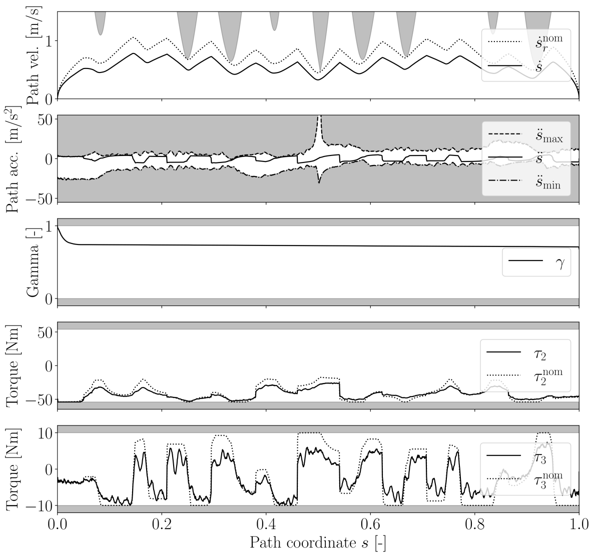
References

[1] J. E. Bobrow, S. Dubowsky, and J. S. Gibson, "Time-optimal control of robotic manipulators along specified paths," *The International Journal of Robotics Research*, vol. 4, no. 3, pp. 3–17, 1985.

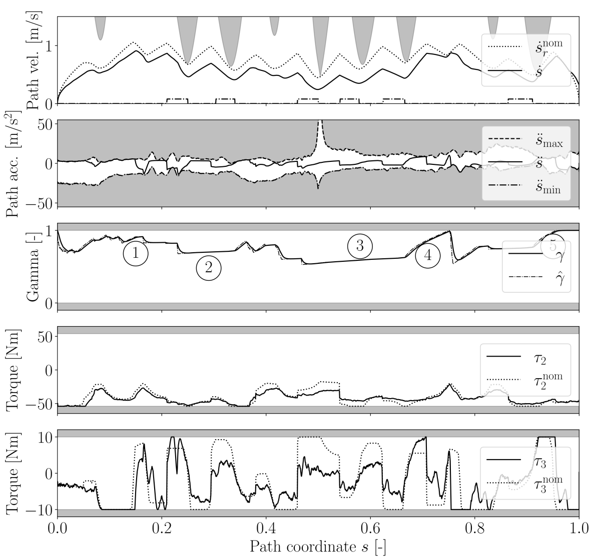
[2] K. Shin and N. McKay, "Minimum-time control of robotic manipulators with geometric path constraints," *IEEE Transactions on Automatic Control*, vol. 30, no. 6, pp. 531–541, 1985.



(a) PVC: bound inversion occurs near critical/tangency points.



(b) PVC+adaptive: γ decreases in the beginning of the path tracking.



(c) OPVC: γ increases at ①④⑤ and is conservative at ②③.

Fig. 6: Experiment (zig-zag): comparison of results.

[3] Z. Shiller and H.-H. Lu, "Computation of Path Constrained Time Optimal Motions With Dynamic Singularities," *Journal of Dynamic Systems, Measurement, and Control*, vol. 114, pp. 34–40, 03 1992.

[4] D. Verscheure, B. Demeulenaere, J. Swevers, J. De Schutter, and M. Diehl, "Time-optimal path tracking for robots: A convex optimization approach," *IEEE Transactions on Automatic Control*, vol. 54, no. 10, pp. 2318–2327, 2009.

[5] 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.

[6] 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.

[7] R. Kelly, V. Santibáñez, and A. Loría, *Control of Robot Manipulators in Joint Space*, vol. 693. Springer, 2005.

[8] J. M. Hollerbach, "Dynamic scaling of manipulator trajectories," in *1983 American Control Conference*, pp. 752–756, IEEE, 1983.

[9] O. Dahl and L. Nielsen, "Stability analysis of an online algorithm for torque limited path following," in *1990 International Conference on Robotics and Automation*, pp. 1216–1222, IEEE, 1990.

[10] O. Dahl and L. Nielsen, "Torque-limited path following by online trajectory time scaling," *IEEE Transactions on Robotics and Automation*, vol. 6, no. 5, pp. 554–561, 1990.

[11] O. Dahl, *Path Constrained Robot Control*. PhD thesis, Department of Automatic Control, Lund Institute of Technology (LTH), 1992.

[12] O. Dahl, "Path-constrained robot control with limited torques-experimental evaluation," *IEEE Transactions on Robotics and Automation*, vol. 10, no. 5, pp. 658–669, 1994.

[13] O. Gerelli and C. G. L. Bianco, "Nonlinear variable structure filter for the online trajectory scaling," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 10, pp. 3921–3930, 2009.

[14] C. G. L. Bianco and O. Gerelli, "Online trajectory scaling for manipulators subject to high-order kinematic and dynamic constraints," *IEEE Transactions on Robotics*, vol. 27, no. 6, pp. 1144–1152, 2011.

[15] B. Olofsson and L. Nielsen, "Path-tracking velocity control for robot manipulators with actuator constraints," *Mechatronics*, vol. 45, pp. 82–99, 2017.

[16] M. Faroni, M. Beschi, C. G. L. Bianco, and A. Visioli, "Predictive joint trajectory scaling for manipulators with kinodynamic constraints," *Control Engineering Practice*, vol. 95, p. 104264, 2020.

[17] G. Antonelli, S. Chiaverini, and G. Fusco, "A new on-line algorithm for inverse kinematics of robot manipulators ensuring path tracking capability under joint limits," *IEEE Transactions on Robotics and Automation*, vol. 19, no. 1, pp. 162–167, 2003.

[18] A. Dahlin and Y. Karayiannidis, "Trajectory scaling for reactive motion planning," in *2022 International Conference on Robotics and Automation*, pp. 5242–5248, IEEE, 2022.

[19] A. Bemporad, "Reference governor for constrained nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 43, no. 3, pp. 415–419, 1998.

[20] A. Bemporad, T.-J. Tarn, and N. Xi, "Predictive path parameterization for constrained robot control," *IEEE Transactions on Control Systems Technology*, vol. 7, no. 6, pp. 648–656, 1999.

[21] T. Faulwasser and R. Findeisen, "Nonlinear model predictive control for constrained output path following," *IEEE Transactions on Automatic Control*, vol. 61, no. 4, pp. 1026–1039, 2015.

[22] T. Faulwasser, T. Weber, P. Zometa, and R. Findeisen, "Implementation of nonlinear model predictive path-following control for an industrial robot," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 4, pp. 1505–1511, 2016.

[23] M. Faroni, M. Beschi, N. Pedrocchi, and A. Visioli, "Predictive inverse kinematics for redundant manipulators with task scaling and kinematic constraints," *IEEE Transactions on Robotics*, vol. 35, no. 1, pp. 278–285, 2018.

[24] A. Bemporad, A. Casavola, and E. Mosca, "A predictive reference governor for constrained control systems," *Computers in industry*, vol. 36, no. 1-2, pp. 55–64, 1998.

[25] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics*. Springer London, 2009.

[26] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*, vol. 3. John Wieley & Sons, 2020.

[27] D. Verscheure, M. Diehl, J. De Schutter, and J. Swevers, "On-line time-optimal path tracking for robots," in *2009 IEEE International Conference on Robotics and Automation*, pp. 599–605, IEEE, 2009.

[28] C. Gaz, M. Cognetti, A. Oliva, P. R. Giordano, and A. De Luca, "Dynamic identification of the Franka Emika Panda robot with retrieval of feasible parameters using penalty-based optimization," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4147–4154, 2019.

[29] J. Mattingley and S. Boyd, "CVXGEN: A code generator for embedded convex optimization," *Optimization and Engineering*, vol. 13, pp. 1–27, 2012.