# Path-tracking velocity control for robot manipulators with actuator constraints

Olofsson, Björn; Nielsen, Lars

*Published in:*
Mechatronics

*DOI:*
10.1016/j.mechatronics.2017.05.008

2017

*Document Version:*
Peer reviewed version (aka post-print)

# Path-Tracking Velocity Control for Robot Manipulators with Actuator Constraints

Björn Olofsson[a,*], Lars Nielsen[b]

[a] *Department of Automatic Control, LTH, Lund University, SE–221 00 Lund, Sweden*
[b] *Department of Electrical Engineering, Division of Vehicular Systems, Linköping University, SE–581 83 Linköping, Sweden*

**Abstract**

An algorithm for high-performance path tracking for robot manipulators in the presence of model uncertainties and actuator constraints is presented. The path to be tracked is assumed given, and the nominal trajectories are computed using, for example, well-known algorithms for time-optimal path tracking. For online path tracking, the nominal, feedforward trajectories are combined with feedback in a control architecture with a secondary controller, such that robustness to uncertainties in model or environment is achieved. The control law is based on existing path-velocity control (PVC), or so called online time scaling, but in addition to speed adaptation along the tangent of the path, the algorithm also comprises an explicit formulation and approach, with several attractive properties, for handling the deviations along the transversal directions of the path. For achieving fast convergence along the normal and binormal directions of the path in 3D motion, the strategy proposed has inherent exponential convergence properties. The result is a complete architecture for path-tracking velocity control (PTVC). The method is evaluated in extensive simulations with manipulators of different complexity, and PTVC exhibits superior performance compared to PVC.

*Keywords:* path following, optimal path tracking, natural path coordinates, robot-control architecture.

## 1. Introduction

The task considered in this paper is for a controlled mechanical system to follow a predefined geometric path. A path is a curve in space, whereas for a

*Corresponding author (Phone: +46 46 222 87 60).
*Email addresses:* bjorn.olofsson@control.lth.se (Björn Olofsson), lars.nielsen@liu.se (Lars Nielsen)

trajectory the curve is time-parametrized, or alternatively, a corresponding velocity profile is given. The fundamental difference between path tracking and trajectory tracking is consequently that the velocity along the path can be modified in the case of path tracking. Path tracking, or equivalently path following, is a fundamental control problem with many applications, and it is well-known for robot manipulators in tasks such as machining, welding, gluing, and cutting. In intelligent and/or autonomous systems it is customary with a decoupled approach, *i.e.*, to segment motion control in the levels of path planning and path tracking [1, 2] in a hierarchical structure to reduce the complexity of the complete motion-planning problem. Therefore, path tracking is a major component in new developments in intelligent robotics, unmanned aerial vehicles (UAVs), and autonomous vehicles, and the current interest in robust algorithms for path tracking is considerable [3, 4, 5].

### *1.1. Trajectory Tracking versus Path Tracking*

One way of approaching path tracking is to consider it as the task of tracking a sequence of trajectory points for a vector $x(t)$ of position coordinates in space, given as function of time $t$. This implies that path tracking is achieved by trajectory tracking. Trajectory tracking in this context means that the time frame, including the time when reaching the final state, is fully specified, and a common method is model-based feedforward control combined with online feedback using predefined trajectories.

In contrast, there are many examples where path tracking is possible but trajectory tracking is not, and a typical situation is when an actuator reaches its saturation limit. In all practical systems, there are always limitations on the available control authority, and then the only possibility may be to adjust the speed along the path, or phrased equivalently, to scale the time frame available for completion of the task. As a concrete example, when driving a car and the road exhibits high slip, the only way to stay on the road—*i.e.*, on the desired predefined path—may be to adjust the speed. Further, it is natural, as in the car-driving example, to think in position along the path instead of time: The driver turns at a bend (path tracking) not after a certain time (trajectory tracking). In practice this means that the time frame is released, or equivalently phrased, that the speed along the path is adjusted. Nevertheless, there must still be coordination between the degrees of freedom (DoF) of the system to follow the desired path, and early research in this spirit is by means of dynamic scaling of the trajectories [6].

To utilize the freedom of velocity control along the desired path, it is clear that there are two different problems to be solved. One task is to control the traversal along the tangential direction of the path, where the objective typically is some optimal performance, *e.g.*, minimum-time or minimum-energy. The other

task is to follow the path, *i.e.*, to coordinate the different DoF of the system such that the desired path is tracked. It is therefore natural to consider both the control of tangential motion along the path and the motion toward the path (along the normal or binormal directions). To this end, we use the concepts $x_{\parallel}$ and $x_{\perp}$, defined more precisely in Section 3, where the main idea is to have largest possible control freedom along the directions of $x_{\parallel}$ together with desired convergence to the path along the directions of $x_{\perp}$.

### *1.2. Previous Research on Path Tracking*

Initial research on time-optimal trajectory generation for path tracking with robot manipulators was presented in [7, 8, 9, 10, 11], and extensions with respect to dynamic uncertainties and singular control were proposed in [12, 13]. Recently, a convex reformulation of the trajectory-generation problem for time-optimal path-tracking was suggested in [14, 15], together with efficient algorithms for computation of the optimal trajectories. Extensions with respect to convex-concave constraints were presented in [16]. Methods for online trajectory generation for time-optimal path tracking were considered in [17, 18]. Further application areas for the methods in [15] were investigated in [19].

As stated in the previous subsection, path tracking can be achieved by trajectory tracking, conditioned on that sufficient control authority is available. Within this class of approaches with a given time horizon, there are established algorithms to adjust the control inputs to obtain path tracking in the cases that there is some degree of repetitiveness of the path. Iterative learning control (ILC) is one such successful strategy [20, 21, 22, 23]. The typical application scenario of ILC is offline in a batch-oriented structure. Often, ILC methods are limited to tasks where the trajectory is of fixed length, thus not permitting time scaling. Methods for relaxing this requirement in learning were investigated in [24, 25]. ILC for optimal path tracking was considered in [26]. Path tracking for mobile platforms was investigated in [27, 28]. Feedback linearization for trajectory planning was proposed in [29] and trajectory optimization in constrained environments was considered in [30]. Control laws for path tracking with mobile robots resulting in exponential convergence were proposed in [31]. The major difference between manipulators considered in this paper and mobile robots is the non-holonomic constraints that might be necessary to consider for the latter category of systems.

The present research is a new approach to dynamic scaling of trajectories [6], where the formulation is most related to the previous research in [32]. The algorithm proposed in [32] was formulated as online time scaling of precomputed time-optimal trajectories, but could equivalently be phrased as online velocity control along the path and is here denoted path-velocity control (PVC) [33, 34]. Extensions of the PVC algorithm in [32] and alternative approaches to high-accuracy

path tracking have later been proposed in, *e.g.*, [35, 36, 37, 38, 39, 40, 41], where the extensions mainly are with respect to the constraints on the system that can be accounted for and regarding developments for practical implementations of the algorithms in industrial systems. In particular, important extensions with respect to constraints in the robot workspace were proposed. Dynamic scaling of trajectories for robots with elastic joints was considered in [42], and dynamic time scaling for generating energy-optimal trajectories for robots was proposed in [43]. Another approach to path tracking for position servos with actuator limitations was considered in [44]. Initial research on an alternative formulation of the path-tracking control problem with actuator constraints was presented in [45], where control along the orthogonal directions of the path was introduced and given priority over the control along the tangent of the path, thus not focusing on the coordination of the DoF. Another approach to orbital stabilization for nonholonomic underactuated systems based on transverse linearization, related to the control architecture in this paper, was suggested in [46]. Moreover, a predictive path parametrization for online path tracking was considered in [47], with a similar purpose as the PVC algorithm in [32].

### *1.3. Contributions*

The main contribution of this paper is a complete and integrated control architecture along the lines outlined in Section 1.1 for robust path tracking for robot manipulators with actuator constraints. In the setting of $x_\perp$ and $x_\parallel$, the control along $x_\parallel$ utilizes PVC and builds on previous algorithms for time scaling [6, 32], such that if the robot is on the path the controller is equivalent to the algorithm in [32]. In addition, an extension of the controller is introduced that handles fast convergence along the orthogonal directions of the path, $x_\perp$, should the robot be off the path. The result is a control architecture that employs coordinated feedback control both along $x_\perp$ and $x_\parallel$, which is called path-tracking velocity control (PTVC). This strategy, PTVC, achieves robust path tracking for a wide class of mechanical systems and results in a substantial performance increase in the achieved path-tracking accuracy compared to PVC, as demonstrated in Section 7.

### *1.4. Outline*

A motivating example explaining fundamental ideas of the algorithm is presented in Section 2. The natural coordinates of a curve [48, 49] is an appropriate framework for defining the tangential direction $x_\parallel$ and the orthogonal directions $x_\perp$. This is described in Section 3, where also the definition of radius of curvature that is used in the control law is provided. The mechanical systems under consideration are given their mathematical formulation in Section 4. In this section, trajectory generation for such systems is also discussed. Then, based on natural coordinates, the control architecture PTVC with separate terms for tangential

and orthogonal control is presented in Section 5. An advantageous property is exponential convergence along the normal directions of the path, and an analysis of convergence properties is presented in Section 6. The complete algorithm is evaluated in Section 7 with extensive simulations on different manipulators. The obtained results, possible extensions, and generalizations of the control architecture are discussed in Section 8, and finally conclusions are drawn in Section 9.

## 2. A Conceptual Example

The following example introduces some of the main ideas for obtaining convergence of the path tracking along the transversal directions by creating an inherent attraction to the path, path stability, without hampering the control possibilities tangential to the path, *i.e.*, the path-velocity control.

### *2.1. Tracking of a Circular Path*

Consider the path-tracking task where the path to follow is a circle with constant radius $R$ in a two-dimensional space. Let $x_1$ and $x_2$ be the positions along the respective coordinate axis. Different descriptions of the path will be valuable in the conceptual treatment of the task. The circle can be described by an equation

$$g(x_1, x_2) = x_1^2 + x_2^2 - R^2 = 0, \tag{1}$$

and it can be described in parametrized form as

$$x_1 = R\cos(\theta), \tag{2}$$
$$x_2 = R\sin(\theta), \tag{3}$$

where $\theta$ is the angle with respect to the positive $x_1$-axis. The latter relation is put in vector form by defining $f = \begin{pmatrix} x_1 & x_2 \end{pmatrix}^{\mathrm{T}}$. Note that $f$ here denotes the coordinates in Cartesian space, while in Section 4 it will denote the path for the generalized coordinates. Introduce the path parameter $s$ as the length along the path. Then, $\theta = s/R$ and

$$f(s) = \begin{pmatrix} R\cos(s/R) \\ R\sin(s/R) \end{pmatrix}. \tag{4}$$

The motion along the circle can also be expressed as a state-space dynamic system according to

$$\dot{x}_1(t) = \omega x_2(t), \tag{5}$$
$$\dot{x}_2(t) = -\omega x_1(t), \tag{6}$$

where $\omega$ is the angular velocity, defined by the relations $\omega = -\dot{\theta} = -\dot{s}/R$. Introduce $r$ as the distance from the origin according to $r^2 = x_1^2 + x_2^2$. The path-tracking task is to keep $r(t) = R$. Considering (5)–(6), any model error or disturbance leading to $r(t) \neq R$, will result in the system continuing to evolve with a new faulty radius different from $R$.

To achieve path stability—*i.e.*, the property that $r(t)$ will converge to $R$ after a disturbance—it is natural to introduce corrections orthogonal to the path. Here, this means in the direction of the gradient to (1), which is denoted $\nabla g$. Justified by the analysis following in Section 6, one choice is to introduce a path-stability term according to

$$\frac{1}{2}\left(R^2 - r^2\right)\nabla g = \left(R^2 - r^2\right)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} x_1(R^2 - r^2) \\ x_2(R^2 - r^2) \end{pmatrix}. \tag{7}$$

Introducing the compensation term (7) in (5)–(6) leads to the resulting dynamics given by

$$\dot{x}_1(t) = \omega x_2(t) + x_1(t)\left(R^2 - r(t)^2\right), \tag{8}$$

$$\dot{x}_2(t) = -\omega x_1(t) + x_2(t)\left(R^2 - r(t)^2\right). \tag{9}$$

Eliminating $x_1$ and $x_2$ from the relations (8)–(9) is made by multiplying the first and second equation with $x_1$ and $x_2$, respectively, and then add and subtract equations pairwise. Using that $r\dot{r} = x_1\dot{x}_1 + x_2\dot{x}_2$ for the first case and substituting $x_1 = r\cos(\theta)$, $x_2 = r\sin(\theta)$ for the second case, this procedure leads to the following decoupled set of differential equations

$$\dot{r}(t) = r(t)(R^2 - r(t)^2), \tag{10}$$

$$\dot{\theta}(t) = -\omega. \tag{11}$$

The differential equations above are solved independently. Starting with (10), the principle of separation of variables is used to obtain the positive solution, which is given by

$$r(t) = R\sqrt{\frac{1}{1 + ke^{-2R^2t}}}, \tag{12}$$

where $k$ is a constant determined by the initial value $r(0)$. Here, it holds that $R - r = O(e^{-2R^2t})$, so it is clear that $r(t)$ converges to $R$ exponentially as $t \to \infty$. The second differential equation, (11), gives

$$\dot{\theta}(t) = \frac{\dot{s}(t)}{R}, \tag{13}$$

which is just a rephrasing of the definition. Consequently, (8)–(9) result in exponential convergence to the path, while behavior along the path is still free to be controlled by a separate control law. This tangential behavior is described by the path parameter $s$ and it can thus be used for tangential path-velocity control.

*2.2. Outline of Algorithm Ideas*

The key idea in this paper is to use the separation principle, indicated in the previous example by the independence of (10) and (11), between tangential control and exponential path convergence along the directions normal to the path. Then, for a given path, there needs to be a sequence of local coordinate systems, defining the local variables corresponding to $x_1$, $x_2$, and the radius $R$. A natural choice here is to use the well established mathematical concept of natural coordinates for a path [48, 49], as will be described in Section 3. Conceptually, the center of curvature will serve as the origin for the variables $x$ and the radius of curvature will serve as $R$. For a general path, a well established concept is to reduce the multi-dimensional control problem to a one-dimensional problem by using the correspondence to (4)—*i.e.*, to parametrize the path $f$ in a scalar path coordinate $s$, and this has been used in optimal trajectory planning [7, 8, 9, 10] and also in feedback architectures such as [32]. Conceptually, this means that the path is a constraint in $x$, and that $s$ and its time-derivatives are used to compute the optimal velocity profile along the path. This parametrization is presented in Section 4, and is used for tangential control, which is equivalent to control along $x_\parallel$. The control architecture in Section 5 integrates the control along $x_\parallel$ from [32] with an approach to control along $x_\perp$, similarly to the path-stability term in (7) in the example described in Section 2.1.

## 3. Natural Path Coordinates

When studying motion along a path—*i.e.*, a curve in space—the concept of natural coordinates is often used. This formulation can be found in standard textbooks both in mechanics [49] and in mathematics [48]. The main results used later in this paper are briefly recapitulated in this section, but the proofs are to a large extent omitted and the reader is referred to the mentioned references for details. The important role of tangential motion and the role of curvature along the normal direction should be noted.

*3.1. Definition of Coordinate System*

The natural coordinate system is usually derived by means of a trajectory, which from the geometrical point of view is a parametrized curve in a three-dimensional space. Thus, let the trajectory be given by the vector $\rho(t)$ according to

$$\rho(t) : [t_0, t_1] \subset \mathbb{R} \quad \text{and} \quad \rho \in \mathcal{C}^2[t_0, t_1], \tag{14}$$

which means that the trajectory is a parametric curve that is two times continuously differentiable on the parameter interval $[t_0, t_1]$. Moreover, the unit vector $\pi$ is the tangent to the trajectory at each point and is defined using the velocity vector $v(t) = \dot{\rho}(t)$. The formal definition is:

**Definition 1 (Tangent Vector).** *Given a trajectory $\rho(t)$, the unit tangent vector to the trajectory is defined as*

$$\pi = \frac{v}{||v||_2}. \tag{15}$$

This vector $\pi$ is the first basis vector of the natural coordinate system. Since $v$ is a function of the parameter $t$, which describes the trajectory in the geometric space, $\pi$ itself is a function of the same parameter, so in general $\pi$ is not constant along the trajectory. Let $s$ be the length of the curve traversed in the interval $[t_0, t]$, *i.e.*,

$$s(t) = \int_{t_0}^{t} \sqrt{v(\zeta)^{\mathrm{T}} v(\zeta)}\ \mathrm{d}\zeta,\ t \leq t_1. \tag{16}$$

If $v \geq 0$, with equality only in isolated time instants, observe that the length is a non-negative, monotonically increasing, and real function of the parameter $t$ and that

$$\dot{s} = \frac{\mathrm{d}s}{\mathrm{d}t} = \sqrt{v^{\mathrm{T}} v} = ||v||_2. \tag{17}$$

From this relation it follows that

$$\frac{\mathrm{d}\rho}{\mathrm{d}s} = \frac{\mathrm{d}\rho}{\mathrm{d}t}\frac{\mathrm{d}t}{\mathrm{d}s} = \frac{v}{||v||_2} = \pi. \tag{18}$$

This means that if the trajectory is parametrized in its length, then the velocity vector is the unit vector $\pi$, which is the tangent to the curve. The length of the curve in space is usually referred to as the natural parameter [49].

The second unit vector of the natural coordinate system is a consequence of an important property of $\pi$, which is given by the following theorem.

**Theorem 1 (Orthogonality).** *For the tangent vector $\pi$ and its derivative with respect to $t$, it holds that*

$$\pi \perp \dot{\pi}. \tag{19}$$

For the derivation and proof of this relation, see [48, 49]. From this theorem, the following definition is natural:

**Definition 2 (Normal Vector).** *Given a trajectory $\rho$, the unit vector normal to the trajectory is defined as*

$$n = \frac{\dot{\pi}}{||\dot{\pi}||_2}. \tag{20}$$

The plane spanned by $(\pi, n)$ is called the osculating plane [49], which will be used later when deriving the controller in Section 5. Introduce the curvature $\kappa$ by $\kappa = ||\dot{\pi}||_2$. Then, it holds that $\kappa \geq 0$, and it follows from (20) that

$$\dot{\pi} = \kappa n. \tag{21}$$

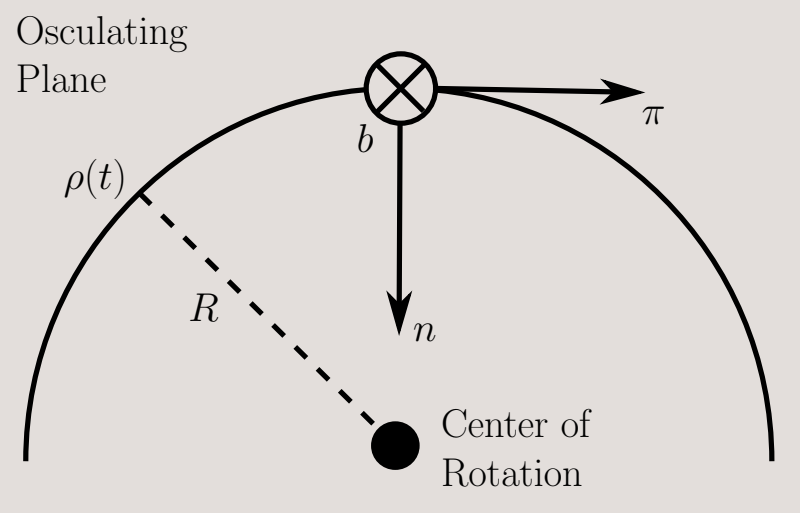


Figure 1: Axes of the natural coordinate system for an example curve $\rho$ with constant radius $R$.

When a trajectory $\rho$ is given and the curve is parametrized in the length $s$, then it follows from (17) that $||v||_2 = 1$ and that $\dot{v} = \kappa n$, as a consequence of (21). The physical dimension of $\kappa$ is the inverse of a length and it is the curvature of the trajectory along the curve. This leads to the following central definition:

**Definition 3 (Radius of Curvature).** *The radius of curvature $R$ is defined as*

$$R = \frac{1}{\kappa}, \quad \text{if} \quad \kappa > 0. \tag{22}$$

Since $\kappa \geq 0$, the radius of curvature is non-negative. To complete the natural coordinate system with its third basis vector, the following definition is adopted:

**Definition 4 (Binormal Vector).** *For a given trajectory $\rho$, where $\pi$ and $n$ have been defined, the binormal unit vector is defined as the vector product*

$$b = \pi \times n. \tag{23}$$

The unitary of the vector $b$ is a direct consequence of the fact that $\pi$ and $n$ are perpendicular unit vectors. The natural coordinate system is depicted in Figure 1.

*3.2. Definition of $x_{\parallel}$ and $x_{\perp}$*

The framework of natural coordinates, with the moving coordinate system $(\pi, n, b)$, is not dependent on the original parametrization. Consequently, the parametrization is an intrinsic property of a curve in space, and thus an intrinsic

property of a path. Relating to Section 2.2, the direction $x_{\|}$ is along the basis vector $\pi$ and the directions $x_{\perp}$ are the orthogonal subspace, here spanned by the normal and binormal basis vectors $(n, b)$. In a typical path-tracking task for a mechanical system, the centripetal forces will be along the direction of the normal vector $n$. As a consequence, control along the normal direction $n$ is in general more crucial than the corresponding control along the binormal direction $b$. The radius of curvature $R(s)$, obtained as part of the procedure of describing the path in natural coordinates and in general varying along the path, is used as the key local path descriptor as indicated in the example in Section 2.1.

## 4. Modeling Assumptions and Trajectory Optimization

This section defines the class of mechanical systems considered in this paper. Moreover, the parametrization of the geometric path and nominal trajectory optimization, or more generally trajectory planning, are discussed.

### *4.1. Dynamics*

The dynamics of the considered class of mechanical systems are assumed to be possible to write on the following form

$$\tau = M(q)\ddot{q} + C(\dot{q}, q)\dot{q} + D(q)\dot{q} + G(q), \tag{24}$$

where $\tau \in \mathbb{R}^{n_d}$ are the input torques, $q \in \mathbb{R}^{n_d}$ are the generalized coordinates, $M(q)$ is the inertia matrix, $C(\dot{q}, q)$ is the Coriolis/Centripetal matrix, $D(q)$ is the viscous friction matrix, and $G(q)$ is the gravity vector [50, 51]. The form of the dynamic equations in (24) is applicable for a large number of systems; important examples include rigid-body robot manipulators (with a fixed rigid base) and acceleration or torque-controlled position servos. It is assumed that the control objective is the position $x \in \mathbb{R}^3$ of the mechanical system in a Cartesian space[1].

**Remark 1.** *The theory presented in this section can be extended to include flexible-joint models with nonactuated DoF, see Chapter 3 in [33]. In practice, this means that the model will require an increased number of states, and that the system dynamics need to be represented using time-derivatives of the path parameter of higher order than two. The focus of the presentation in this paper is on the rigid-body case.*

[1]For a discussion on the case with both specified position and orientation, see Section 8.4.

*4.2. Path Parametrization*

A path $f \in \mathbb{R}^{n_d}$ is assumed to be available, defining the desired motion of the $n_d$ generalized coordinates $q$ of the system. Note that this is the path for the generalized coordinates of the system, in contrast to the path in Cartesian space discussed in Section 2.1. The reason is that constraints on the control inputs appear in the actuators, *i.e.*, in the joint space in the examples in Section 7, and then it is important for the time-scaling approach to define the path in this space both in the motion planning phase and in the subsequent path-tracking control. As in the definition of a curve in Section 3, the path is assumed to be two times continuously differentiable. In general, the path is determined by a high-level path planner in Cartesian space and subsequently described in natural coordinates as outlined in Section 3. Hence, the inverse-kinematics relations are necessary for transformation to corresponding generalized coordinates. This procedure is in general not a problem, since it is performed offline. Note also that specification of the generalized coordinates implies a full specification of the end-effector position and orientation. The end-time $t_f$ of the time frame is in many cases unknown *a priori* and should thus be computed as part of the trajectory-planning procedure. In order to establish a time-invariant description of the path, the path coordinate $s \in [s_0, s_f]$ from Section 3 is adopted, see, *e.g.*,[32] or Chapter 3 in [33]. This is formalized as

$$f(s), \ s \in [s_0, s_f]. \tag{25}$$

The interval for the path coordinate can be scaled to a nominal interval between zero and one, but the default choice here is the Euclidean length of the path. The corresponding time-derivatives of the path coordinate, $\dot{s}$ and $\ddot{s}$, are referred to as the path velocity and the path acceleration, respectively.[2]

*4.3. Nominal Trajectory Optimization*

In the control architecture proposed in Section 5, it is assumed that a nominal trajectory is available. For task effectiveness, this nominal trajectory is natural to compute using a time-optimal criterion, with the maximum available inputs to the actuators as constraints. However, also other approaches to determine the nominal trajectory are possible to use together with the considered control architecture.

It is well-known in the literature [7, 8, 9, 10] that the time-optimal path-tracking problem for systems of the format (24) can be efficiently solved by transforming the original problem in the time-domain with an open interval $[0, t_f]$ to a

[2]In the case of dynamic models with an increased number of states (such as the flexible-joint model discussed in Remark 1), also time-derivatives of higher order of the path coordinate are required.

corresponding problem over the fixed horizon $[s_0, s_f]$ with a significantly reduced number of states. If the optimization criterion is chosen as the time for traversing the path, the cost function can be reformulated as function of the path velocity instead [52] as follows

$$t_f = \int_0^{t_f} \mathrm{d}t = \int_{s_0}^{s_f} \frac{\mathrm{d}t}{\mathrm{d}s}\mathrm{d}s = \int_{s_0}^{s_f} \frac{1}{\dot{s}}\mathrm{d}s. \tag{26}$$

Employing the requirement that $q = f(s)$ along the desired path, it follows by using the chain rule that

$$\dot{q} = f'(s)\dot{s}, \tag{27}$$
$$\ddot{q} = f''(s)\dot{s}^2 + f'(s)\ddot{s}, \tag{28}$$

where $(\cdot)'$ is the derivative with respect to $s$. Further, it is straightforward to reformulate the system dynamics (24) as a function of the path coordinate and its time-derivatives using (27)–(28), see, *e.g.*, [8, 10, 52], according to

$$\tau = \Gamma_1(s)\ddot{s} + \Gamma_2(\dot{s}, s), \tag{29}$$

where

$$\Gamma_1(s) = M(f(s))f'(s), \tag{30}$$
$$\Gamma_2(\dot{s}, s) = [M(f(s))f''(s) + C(f(s), f'(s))]\dot{s}^2 + D(f(s))f'(s)\dot{s} + G(f(s)). \tag{31}$$

Introducing the state variable as $\beta(s) = \dot{s}^2$ and the input $\alpha(s) = \ddot{s}$, see Chapter 3 in [33] and [15], the optimization problem for time-optimal path tracking is

$$\underset{\alpha(s),\beta(s)}{\text{minimize}} \quad \int_{s_0}^{s_f} \frac{1}{\sqrt{\beta(s)}}\mathrm{d}s \tag{32}$$
$$\text{subject to} \quad \tau(s) = \Gamma_1(s)\alpha(s) + \Gamma_2(\sqrt{\beta(s)}, s),\ \beta(s_0) = \beta(s_f) = 0, \tag{33}$$
$$\beta'(s) = 2\alpha(s),\ \beta(s) \geq 0,\ \tau_{\min} \leq \tau(s) \leq \tau_{\max}. \tag{34}$$

Finding a solution of the optimal-control problem corresponding to (32)–(34) was originally performed by constructing the solution in the phase plane $(s, \dot{s})$ by utilizing the constraints on the path acceleration, see [8, 10]. For numerical efficiency, the solution to (32)–(34) can directly be computed using parametric optimization methods such as collocation [53, 54]—*i.e.*, discretization of the complete optimization problem and subsequent solution using general-purpose solvers for large nonlinear programs (NLPs). Given certain constraints on the dynamic model (24), the problem is convex and can thus be solved efficiently with guaranteed convergence to a global minimum [55]. More specifically, it is shown in

[14, 15, 16] that the optimization problem (32)–(34) is convex when $D(q) = 0$, because in this case the system dynamics (29) are affine in the state $\beta$ and the input variable $\alpha$, as can be seen from (30)–(31). Irrespective of the solution method, the result of the optimization procedure is a discretized time-optimal trajectory for tracking of the specified path $f$. The relation between the time $t$ and the path coordinate $s$ is given by

$$t(s) = \int_{s_0}^{s} \frac{1}{\sqrt{\beta(\zeta)}} \, \mathrm{d}\zeta, \;\; s_0 \leq s \leq s_f, \tag{35}$$

from which it is clear that there is a bijective relation between $s$ and $t$.

**Remark 2.** *The relation* (35) *together with* (27)–(28) *can be used to transform the resulting optimal joint trajectories from functions of the path coordinate to functions of time.*

## 5. Control Architecture

To execute a high-performance trajectory, such as the time-optimal resulting from the procedure outlined in the previous section, a direct application of the nominal trajectories as described in Remark 2 does not work satisfactory, as has been observed in [32, 33, 35, 40] and also pointed out in the discussion in Section 1.1. A major reason is that time-optimal path tracking implies that at least one of the control inputs is at its limit at each time point [56], which means that there is no control authority left to keep the system on the desired path in case of modeling uncertainties or disturbances. Thus, there is a need for a feedback scheme able to accommodate for this by adapting the velocity while maintaining high-accuracy path tracking. The following sections describe the components of the PTVC algorithm.

### *5.1. Basic Control*

Besides a given path $f$, it is assumed that a controller for set-point control and trajectory tracking is available. The controller acts on reference values $q_r$, $\dot{q}_r$, and $\ddot{q}_r$ for the generalized coordinates and their time-derivatives, and there are several possibilities for this basic controller. The particular choice is not critical here, but for illustration a common choice for mechanical systems, namely the computed-torque controller [50, 51], is employed. This control law is stated as

$$\tau = \hat{M}(q) \left( \ddot{q}_r + K_v(\dot{q}_r - \dot{q}) + K_p(q_r - q) \right) + \hat{C}(\dot{q}, q)\dot{q} + \hat{D}(q)\dot{q} + \hat{G}(q), \tag{36}$$

where $K_v$ and $K_p$ are controller parameters. Note that the control is based on estimates $\hat{M}$, $\hat{C}$, $\hat{D}$, and $\hat{G}$ of the system parameters.

*5.2. Tangential Control*

Introduce the *path parameter* $\sigma$ as the actual length traveled in the online path traversal [32], *i.e.*, analogously to (16) and the path coordinate in the parametrization of the path (25). When the robot is on the path, the control is performed along the tangent, and thus the reference values for the controller are computed based on the geometric path $f$ and the current value of the path parameter and its time-derivatives. Hence, the reference values for motion along the path are computed as follows

$$q_{r,\parallel}(\sigma(t)) = f(\sigma(t)), \tag{37}$$

$$\dot{q}_{r,\parallel}(\sigma(t)) = f'(\sigma(t))\dot{\sigma}(t), \tag{38}$$

$$\ddot{q}_{r,\parallel}(\sigma(t)) = f''(\sigma(t))\dot{\sigma}(t)^2 + f'(\sigma(t))\ddot{\sigma}(t), \tag{39}$$

where the notation emphasizes that the reference values correspond to motion along the tangent.

*5.3. Transversal Control*

When deviations from the nominal path occur during the motion, corrective actions transversal to the path are required. Since the controller is based on velocity control, the transversal control modifies the velocity references for the generalized coordinates. The fundamental idea is to use a correction similar to the control law (7) discussed in the example in Section 2. However, also the cases of straight paths or paths with low curvature and motion in three dimensions need to be handled. This is treated in the following subsections.

*5.3.1. Nominal Transversal Control*

The control law for the transversal control relies on the parametrization of the path in its natural coordinates, see Section 3. This means that the moving trihedral $(\pi, n, b)$ has been defined along the path (compare with the definitions in Figure 1), and the basis vectors are consequently considered as functions of the path parameter in the controller. In addition to the Cartesian position $x$ of the system, introduce the instantaneous center of curvature $x_o$ in the Cartesian space. Let $r$ be the Euclidean length of the projection of $x - x_o$ onto the subspace defined by $\pi$ and $n$ (*i.e.*, the osculating plane defined in Section 3), where $x_o$ by definition is located in this plane. Thus, the following expression holds for $r$

$$r = ||x^{\pi,n} - x_o||_2 \,, \tag{40}$$

where $x^{\pi,n}$ is the projection of $x$ on the osculating plane. Further, define the vector $n_{\text{op}} = x^{\pi,n} - x_o$ in the osculating plane (*cf.* the vector used in the stabilization (7) in the example in Section 2.1). Using these variables, the following

term for the velocity reference along the orthogonal directions is introduced when deviating from the nominal path

$$\dot{q}_{r,\perp}(\sigma(t)) = K_\perp J(q)^{-1} \left( (R^2 - r^2) n_{\text{op}} - ((x - x_o) \cdot b)\, b \right), \tag{41}$$

where $K_\perp$ is a matrix with parameters to be chosen, $J(q)$ is the Jacobian of the considered manipulator—*i.e.*, the matrix describing the differential kinematics of the robot—and $\cdot$ denotes the scalar product of two vectors. The Jacobian is necessary for transforming the corrections required in Cartesian space to the corresponding joint-velocity reference values. The control along the binormal direction is computed as the projection of $x - x_o$ on the vector $b$ and contributes to the velocity vector along the direction defined by the binormal. Note that $R$ and the vector $b$ in (41) depend on the current value of the path parameter $\sigma$ and they are thus updated continuously along the path.

*5.3.2. Path Segments with Low Curvature*

If the path is straight or the curvature is close to zero for certain parts of the path, the radius $R$ is large or even infinite in the limit. Thus, the path correction in these cases needs to be treated differently than what is defined in (41). It is noted that the tangent $\pi$ is a good local approximation of the path under these circumstances. Therefore, the following velocity reference is used

$$\dot{q}_{r,\perp}(\sigma(t)) = K_\perp J(q)^{-1} \left( - ((x - x_r) \cdot n)\, n - ((x - x_r) \cdot b)\, b \right), \tag{42}$$

in the case that the curvature $\kappa < \kappa_l$, where $\kappa_l$ is a user-specified parameter for the threshold value. In (42), $x_r$ is the current desired position along the path in Cartesian space (corresponding to $q_{r,\|}$). Consequently, in this case the control along the normal direction $n$ is handled as the control along the binormal $b$.

**Remark 3.** *The transversal control laws* (41)–(42) *assume, for notational convenience, a manipulator with three DoF and motion in 3D. The adaptation of the control laws to 2D motion is obvious, since the control is performed only in the osculating plane in that case. In the case of more DoF of the manipulator than three, the pseudo-inverse of appropriate parts of the Jacobian for the manipulator can be considered, see also Section 8.4.*

*5.4. Reference Values and Their Interpretation*

Combining the control along the tangential and the transversal directions leads to the following reference values for the generalized coordinates

$$q_r(\sigma(t)) = q_{r,||}(\sigma(t)), \tag{43}$$

$$\dot{q}_r(\sigma(t)) = \dot{q}_{r,||}(\sigma(t)) + \dot{q}_{r,\perp}(\sigma(t)), \tag{44}$$

$$\ddot{q}_r(\sigma(t)) = \ddot{q}_{r,||}(\sigma(t)). \tag{45}$$

The interpretation of the choice of reference values is as follows. For the generalized coordinates $q$, the reference values should always correspond to a point along the geometric path and consequently no modification is introduced in the case of path deviations. For the velocities $\dot{q}$, compensation is introduced according to (41)–(42) such that motion toward the desired path is obtained when deviations arise. Regarding the corresponding accelerations $\ddot{q}$, the reference values are chosen for motion along the nominal path. The reason for this being that the path-velocity control, see Section 5.6, limits the path acceleration and scales the nominal path velocity and acceleration trajectories based on the desired and available input torques. Thus, a fundamental mechanism of the algorithm is that the acceleration is adjusted automatically and therefore no modification of $\ddot{q}_r$ is required.

### *5.5. Parametrization of the Basic Controller*

By using the expressions (43)–(45) for the reference values, all variables in the basic controller (36) can be parametrized in the path parameter $\sigma$ and its time-derivatives. This leads to the expression

$$\tau = \beta_1(\sigma)\ddot{\sigma} + \beta_2(\dot{\sigma}, \sigma, \dot{q}, q), \tag{46}$$

where it is straightforward to verify that (36) can be written on the format (46) and to derive the corresponding expressions for $\beta_1$ and $\beta_2$. Also other control laws than computed torque have been parametrized according to this procedure as discussed in Chapter 4 in [33], where examples include feedforward control combined with feedback using PD control and controllers with integral action.

### *5.6. Path-Tracking Velocity Control*

With the complete control law parametrized in the path parameter $\sigma$, the final step is to determine a scheme for online modification of $\sigma$ utilizing (46). The result is the algorithm called PTVC, and a graphical representation of the algorithm is shown in Figure 2. The lower blocks in the figure have already been defined, and the following sections describe the upper blocks for online control of $\sigma$, $\dot{\sigma}$, and $\ddot{\sigma}$. The trajectory scaling performed online using feedback from the robot is based on the PVC strategy proposed in [32, 33, 57]. This algorithm determines the rate at which the path coordinate, and consequently the reference values $q_r$ and their time-derivatives, is updated to avoid actuator saturation.

#### *5.6.1. Bounds on Path Acceleration*

In order to ensure that each control input is in the available interval $[\tau_{\min}, \tau_{\max}]$, bounds on $\ddot{\sigma}$ can be computed according to [32]. Using the parametrization (46)

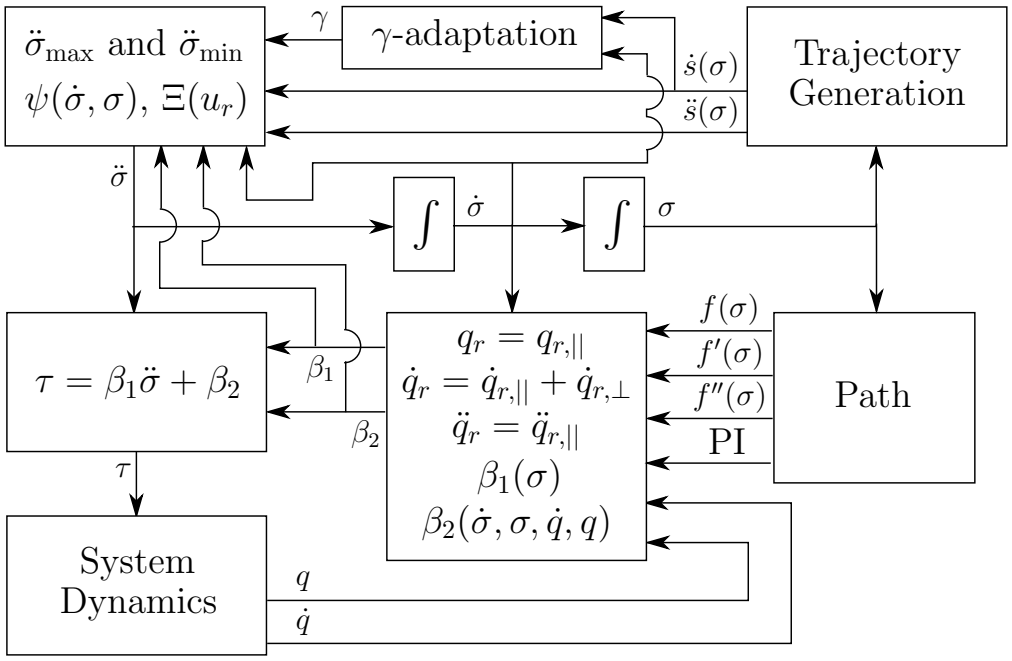


Figure 2: Structure of the PTVC algorithm. In addition to the information about the path $f$ and its derivatives, the generation of the local coordinate systems and the time-varying radius $R$ along the path is required (this is denoted path information (PI) in the block diagram). The computations performed in the upper left block are defined by (47), (49), and (50). The scaling of the trajectories with the parameter $\gamma$ in the upper middle block is governed by (48). Moreover, the functions $\beta_1$ and $\beta_2$ in the middle block are given by the parametrization of the controller defined in (46) and the reference values in the middle block are specified by (43)–(45).

of the control law for the basic controller, the bounds on $\ddot{\sigma}$ are for each input $i,\ i = 1, \ldots, n_d$, as follows

$$\ddot{\sigma}^i_{\max} = \begin{cases} \dfrac{\tau_{\min} - \beta_2^i}{\beta_1^i}, & \beta_1^i < 0 \\ \dfrac{\tau_{\max} - \beta_2^i}{\beta_1^i}, & \beta_1^i > 0 \\ \infty, & \beta_1^i = 0 \end{cases} \qquad \ddot{\sigma}^i_{\min} = \begin{cases} \dfrac{\tau_{\max} - \beta_2^i}{\beta_1^i}, & \beta_1^i < 0 \\ \dfrac{\tau_{\min} - \beta_2^i}{\beta_1^i}, & \beta_1^i > 0 \\ -\infty, & \beta_1^i = 0 \end{cases} \tag{47}$$

where, for notational convenience, it was assumed that all generalized coordinates have the same limitations on their corresponding control inputs. The computations can straightforwardly be modified to the case with different constraints on the input torques $\tau$. To satisfy the constraints on all inputs, the bounds on the path acceleration are chosen as $\ddot{\sigma}_{\min} = \max_i \ddot{\sigma}^i_{\min}$ and $\ddot{\sigma}_{\max} = \min_i \ddot{\sigma}^i_{\max}$. Further, non-constant torque limits $\tau_{\max}$ and $\tau_{\min}$ in the algorithm can be used as it is. This is illustrated in a simulation example in Section 7.2 (see Figure 9), where

state-dependent constraints are used to capture a decrease in maximum available torque because of a velocity-dependent electromotive force in the joint motor.

*5.6.2. Trajectory Scaling and Feedback from Path Velocity*

Nominal values of the path velocity $\dot{s}$ and the path acceleration $\ddot{s}$, as functions of the path parameter, are obtained from the trajectory planning (see Section 4.3). In the case of model uncertainty, downscaling of the nominal optimal trajectories may be necessary. This is achieved by introducing a scaling parameter denoted $\gamma$, $0 < \gamma \leq 1$, depicted in the upper middle block in Figure 2. In practice, this means that the new nominal path velocity and path acceleration are $\gamma\dot{s}(\sigma)$ and $\gamma^2\ddot{s}(\sigma)$, respectively. This is equivalent to a scaling of the time frame, see Chapter 4 in [33], thus implying that the time for the path traversal is increased when $\gamma < 1$. The scaling parameter is made adaptive by the update law proposed in Chapter 4 in [33], which prescribes that

$$\dot{\gamma} = \begin{cases} \xi_{\text{sc}}\dot{\sigma}\left(\dfrac{\dot{\sigma}}{\dot{s}(\sigma)} - \gamma\right), & \gamma\dot{s}(\sigma) \geq \dot{\sigma},\ \ddot{\sigma} \text{ saturated} \\ 0, & \text{otherwise} \end{cases} \tag{48}$$

where $\xi_{\text{sc}}$ is a positive constant and the initial value of the scaling parameter is $\gamma(0) = 1$.

In addition to the trajectory scaling, feedback from the desired path velocity is introduced according to [32, 33]. This feedback $\psi(\dot{\sigma}, \sigma)$ is given as function of the path velocity and the path parameter according to

$$\psi(\dot{\sigma}, \sigma) = \frac{\xi_{\text{fb}}}{2}\left(\gamma^2\dot{s}(\sigma)^2 - \dot{\sigma}^2\right), \tag{49}$$

where $\xi_{\text{fb}}$ is the gain of the feedback and $\dot{s}(\sigma)$ is the precomputed nominal path velocity. When the path acceleration is not saturated, this feedback law results in asymptotic tracking of the desired path velocity, with the tracking time constant specified by $\xi_{\text{fb}}$, see Chapter 4 in [33]. Thus, based on the new nominal acceleration $\gamma^2\ddot{s}(\sigma)$ and the feedback in (49), the desired path acceleration $u_r$ is computed according to $u_r = \psi(\dot{\sigma}, \sigma) + \gamma^2\ddot{s}(\sigma)$. This desired path acceleration $u_r$ is finally transformed to an achievable path acceleration $\ddot{\sigma}$ using

$$\ddot{\sigma} = \Xi(u_r, \ddot{\sigma}_{\min}, \ddot{\sigma}_{\max}), \tag{50}$$

where $\Xi$ is the saturation function, with the second and third arguments as the lower and upper limit, respectively. These steps are represented by the upper left block in Figure 2.

*5.7. Summary of the Control Architecture*

To summarize the proposed path-tracking control architecture (see Figure 2), the computational algorithm for the controller is as follows. Using the expressions (43)–(45) for the reference values combining tangential and orthogonal control, the quantities $\beta_1$ and $\beta_2$ in (46) are computed. Then, the desired path acceleration $u_r$ is determined based on the (possibly scaled) nominal path acceleration and the path-velocity feedback in (49). The constrained path acceleration $\ddot{\sigma}(t)$ is finally obtained using (50), which by integration gives $\dot{\sigma}(t)$ and $\sigma(t)$. The scaling parameter $\gamma$ is simultaneously updated according to (48).

## 6. Analysis

The convergence properties of the developed control algorithm are analyzed, both with respect to the trajectory control law and with respect to convergence to the path.

*6.1. Stability in the Nominal Case*

Assuming that trajectory tracking is realized with the computed-torque control law in (36) and a correct model of the system, the resulting dynamics for the tracking error $e = q_r - q$ are

$$\ddot{q}_{r,\parallel} - \ddot{q} + K_v(\dot{q}_{r,\parallel} - \dot{q}) + K_p(q_{r,\parallel} - q) = 0, \tag{51}$$

in the cases that $\dot{q}_{r,\perp} = 0$. This means that in the nominal case of no deviations from the path, (51) coincides with the standard differential equation for the error dynamics with computed-torque control [51]. Hence, it is clear that the dynamics of the closed-loop system (*i.e.*, the dynamics for the tracking error $e$) can be arbitrarily determined by choosing the parameters $K_v$ and $K_p$. Hence, given appropriate choices of the gain matrices, this guarantees stability of the basic controller. The stability of the path-velocity control along the tangent of the path, PVC, was analyzed in [57], see also Chapter 4 in [33], and the algorithm has later been extended and verified in subsequent publications (see references in Section 1.2). The convergence properties of the PTVC algorithm are the same as those for the PVC algorithm when on the desired path. However, also an investigation of the transversal behavior for the critical case when deviating from the path is needed, which is treated next.

*6.2. The n-Transversal Behavior*

As just stated, the convergence properties of PVC have been established in previous research, so when extending to PTVC the main question is the behavior of the transversal convergence. Further, as discussed in Section 3.2, in a typical

path-tracking task for a mechanical system, the centripetal forces will be along the direction of the normal vector $n$, and as a consequence, control along the normal direction $n$ is more crucial than the corresponding control along the binormal direction $b$. Therefore, the convergence properties of the control along the normal vector $n$ are analyzed. It is implicitly assumed that the parameters of the trajectory-tracking control law result in stable behavior (*cf.* the relation (51)), and that the inputs are not exceeding their saturation limits. As a prelude to the analysis, the example considered in Section 2 is revisited, where it was assumed that the radius $R$ and the angular velocity $\omega$ were constant over time. The fundamental observation for analyzing the $n$-transversal behavior is that the transformation from (8)–(9) to (10)–(11) holds even if $\omega$ and $R$ are time-varying. This is verified by differentiating $r$ and using (8)–(9) in the obtained expression as follows:

$$\begin{aligned}\dot{r} &= \frac{\mathrm{d}}{\mathrm{d}t}\sqrt{x_1^2 + x_2^2} = \frac{x_1\dot{x}_1 + x_2\dot{x}_2}{\sqrt{x_1^2 + x_2^2}} \\ &= \frac{x_1\omega x_2 + x_1^2(R^2 - r^2) - x_2\omega x_1 + x_2^2(R^2 - r^2)}{r} \\ &= \frac{(x_1^2 + x_2^2)(R^2 - r^2)}{r} = r(R^2 - r^2), \end{aligned} \tag{52}$$

which is independent of whether $\omega$ and $R$ are constant or time-varying. The step in the example in Section 2 that does not hold in the general case is the solution of (10), which for a time-varying $R$ has a different solution than (12). It is thus interesting to investigate the behavior of the fundamental path-convergence equation (52), which states that

$$\dot{r}(t) = r(t)\left(R(t)^2 - r(t)^2\right), \tag{53}$$

for reference functions having different frequency characteristics. To illustrate the behavior of this equation for certain choices of the reference trajectory $R(t)$, (53) is simulated for

$$R(t) = 2 + \sin(\omega t), \tag{54}$$

using $\omega = 0$, 0.5, 1, and 2 rad/s. The results of the simulations are shown in Figure 3. The observation in the simulations is that if the path curvature $R(t)$ varies moderately (in the simulations interpreted as the choice of $\omega$), then the tracking is very close. In the cases where the curvature varies quickly, path deviations naturally occur (or equivalently, $R(t)$ and $r(t)$ start to differ). The interesting question following this conclusion is, if this property can be quantified and to some extent this is the case. Note that $R(s)$ (here considered as a function of the path coordinate) is given by the geometric properties of the specified path and is consequently in general precomputed. The path-velocity control of $\dot{s}$ determines

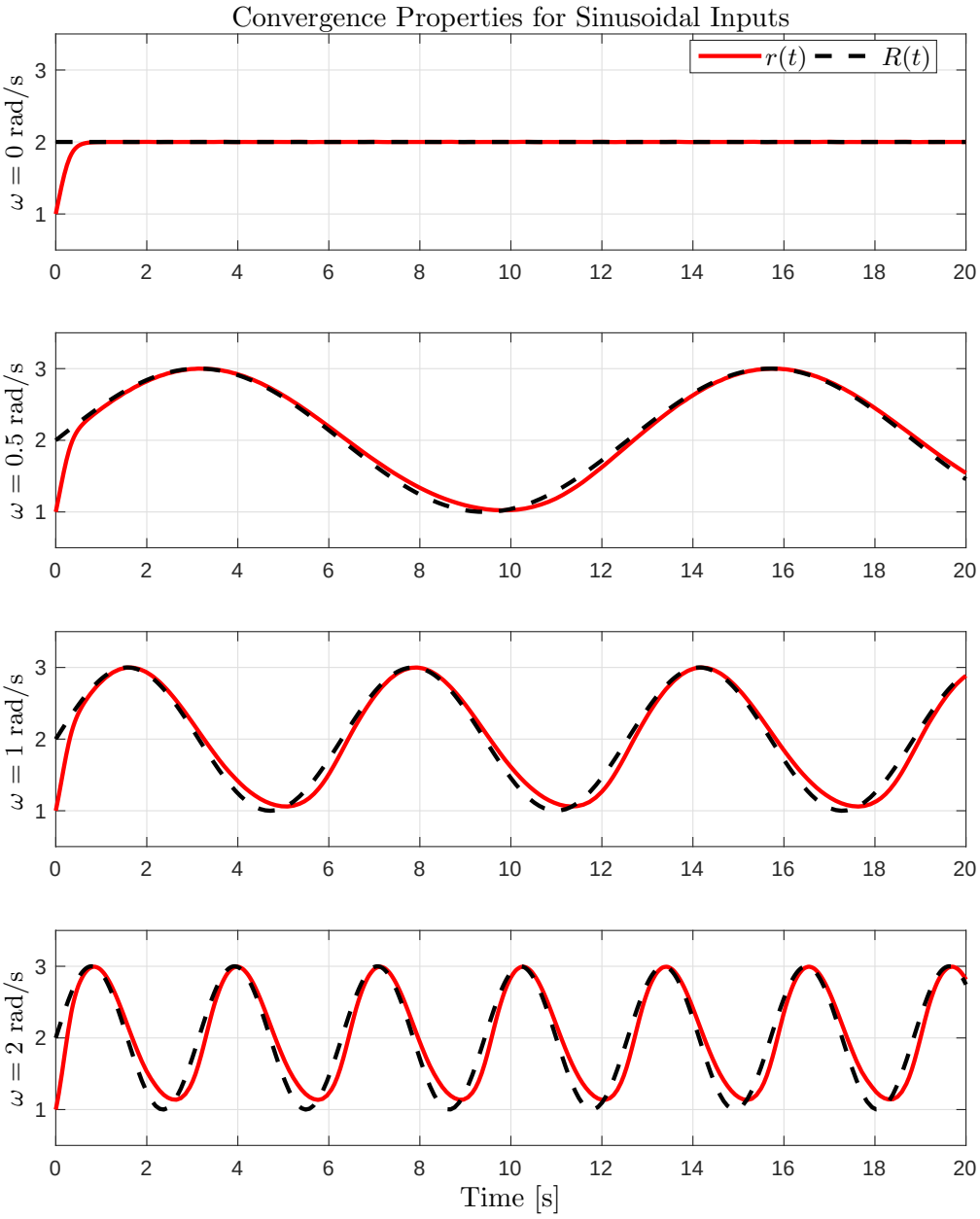


Figure 3: Convergence of $r(t)$ in the case of the reference trajectory $R(t) = 2 + \sin(\omega t)$, for different values of the angular velocity $\omega$.

how fast $R(s)$ is to be updated, so any tracking accuracy can be obtained by decreasing the speed of the system along the specified path. Consequently, the following conclusion can be made. The variable $\omega$, or more generally the speed along the tangent of the path, can always be controlled such that $r$ tracks $R$ within any specified limit.

*6.3. The n-Transversal Error Dynamics*

The employed control principle along the $n$-direction achieves convergence to a path (specified by the time-varying radius of curvature) rather than to a specific point in space. To further analyze the convergence properties of the $n$-transversal behavior, the path deviation $\delta$ is introduced as follows

$$r(t) = R(t) + \delta(t). \tag{55}$$

Differentiating (55) with respect to time, substituting (53) and (55) to eliminate $r$, and finally solving for $\delta$ gives the $n$-transversal error dynamics in form of the following differential equation

$$\dot{\delta}(t) = -\dot{R}(t) - \delta(t)\left(2R(t)^2 + 3R(t)\delta(t) + \delta(t)^2\right). \tag{56}$$

This equation can be approximated for small values of $\delta$ according to

$$\dot{\delta}(t) = -\dot{R}(t) - 2R(t)^2\delta(t). \tag{57}$$

The error dynamics—*i.e.*, the dynamics of the path deviation $\delta(t)$—are thus driven by $\dot{R}(t)$ and $R(t)$. Hence, the path tracking intuitively depends on the amount of variation in $R(t)$. If $R(t)$ is piecewise constant, then the tracking has exponential convergence because $\dot{R}(t) = 0$, which for small $\delta$ gives the path-deviation dynamics $\dot{\delta}(t) = -2R(t)^2\delta(t)$. If $R$ is time-varying, $r(t)$ will track $R(t)$ with a deviation given by (56). To illustrate this property in simulations, the reference trajectories were chosen as the functions $R(t) = 2 \pm e^{-t}$ and $R(t) = 2 \pm 1/\log(e + t)$, with the former exhibiting faster dynamics than the latter. Figure 4 shows the results from the simulations of (53) for the different choices of $R(t)$. In the simulations, $R(0)$ and $r(0)$ switch between 1 and 3 to illustrate convergence both from an initial negative error and an initial positive error. As expected from the approximate differential equation (57) for the dynamics of $\delta(t)$, a larger $R(t)$ should give faster convergence, and this is confirmed in the cases when $R(t)$ is approaching its final value from $R(0) = 3$, compared to starting in $R(0) = 1$.

*6.4. Design of the n-Transversal Error Dynamics*

From a given path, $f(s)$, the curvature $R(s)$ can be precomputed. The transfer of $R(s)$ into $R(t)$ is determined from the evolution of $s(t)$, *i.e.*, from the path-velocity profile. Consequently, the analysis in the previous sections can be used to

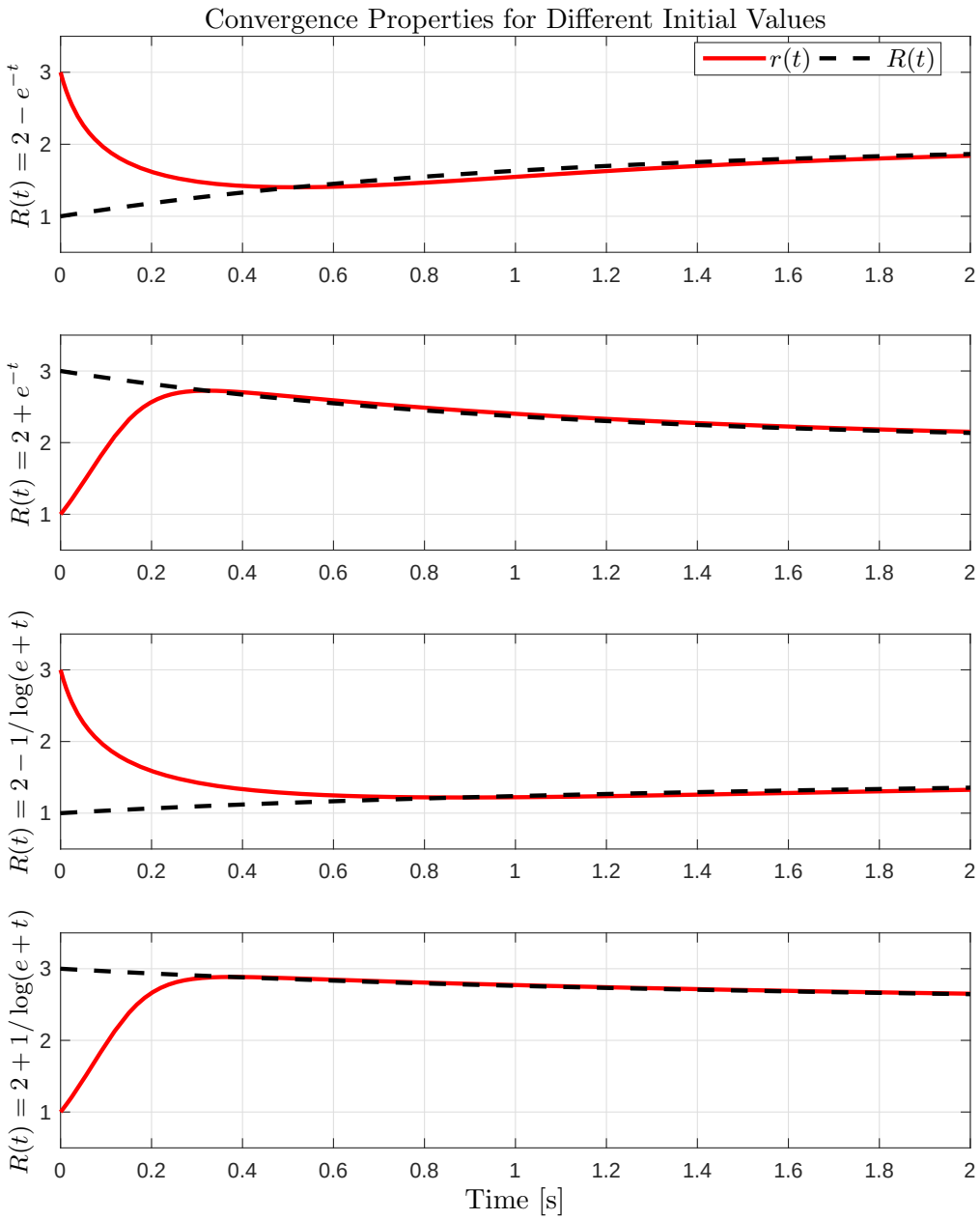


Figure 4: Convergence of $r(t)$ for different choices of the time-varying reference trajectory $R(t)$ and initial values $r(0)$.

compute restrictions on the path velocity in order to ensure desired convergence properties. It is quite intuitive that the path tracking can be excellent if the path velocity is slightly decreased, and the previous sections provide a quantitative method for this.

## 7. Simulation Results

To evaluate and quantify the performance of the PTVC architecture for path tracking, extensive simulations were performed and three of these are illustrated in this paper. The first two examples concern a manipulator with two DoF moving in a two-dimensional plane, illustrating different paths and actuator constraints. The third simulation considers a manipulator with three DoF according to Section 2.9.4 in [50], which enables arbitrary motions in a three-dimensional space for the robot end-effector. The computed-torque strategy (36) is used in all simulations as the basic controller, and the control law is parametrized in the natural frequency $\omega_0$ such that $K_v = \mathrm{diag}(2\omega_0, \ldots, 2\omega_0)$ and $K_p = \mathrm{diag}(\omega_0^2, \ldots, \omega_0^2)$, where $\mathrm{diag}(\cdot)$ is a diagonal matrix with the specified elements along the diagonal. This choice makes the convergence of the error (51) to exhibit critical damping [51]. In the simulations presented here, $\omega_0 = 8$ rad/s. In all simulation examples, the complete PTVC algorithm with both adaptive trajectory scaling (48) and path-velocity feedback (49) is used. The parameters in the algorithms are $\xi_{\mathrm{sc}} = 2$ and $\xi_{\mathrm{fb}} = 10$.

For comparison, the path-tracking accuracy is evaluated in all examples using both the previously presented existing PVC algorithm—*i.e.*, based on the tangential control only—and using the complete PTVC architecture with combined tangential and explicit transversal control. Further, the high-performance aspect is evaluated with the path-traversal time and the torque utilization.

### *7.1. Planar Robot Tracking a Circle*

The planar robot in the first example has equal mass of the links according to $m_1 = m_2 = 1$ kg and the length of each link is $l_1 = l_2 = l = 2$ m. Further, the center-of-mass is located at a distance of $l/2$ from the joint rotational center and each link has a moment of inertia about an axis through the center-of-mass according to $I = 1$ kgm$^2$. The gravity constant is $g = 9.81$ m/s$^2$. The dynamics of a planar robot (without friction in the joints) moving in a two-dimensional plane subject to gravity are given by (compare with (24) and see [50, 51] for a derivation)

$$\tau = M(q)\ddot{q} + C(\dot{q}, q)\dot{q} + G(q), \tag{58}$$

where the elements of the matrices $M$ and $C$ and the vector $G$ are as follows:

$$M_{11} = \frac{m_1 l^2}{4} + m_2 l^2 \left(\frac{5}{4} + \cos(q_2)\right) + 2I, \tag{59}$$

$$M_{12} = \frac{m_2 l^2}{2} \left(\frac{1}{2} + \cos(q_2)\right) + I, \tag{60}$$

$$M_{21} = M_{12}, \quad M_{22} = \frac{m_2 l^2}{4} + I, \tag{61}$$

and

$$c = -\frac{m_2 l^2}{2} \sin(q_2), \tag{62}$$

$$C_{11} = c\dot{q}_2, \quad C_{12} = c(\dot{q}_1 + \dot{q}_2), \tag{63}$$

$$C_{21} = -c\dot{q}_1, \quad C_{22} = 0, \tag{64}$$

$$G_1 = \left(\frac{m_1}{2} + m_2\right) gl \cos(q_1) + \frac{m_2 gl}{2} \cos(q_1 + q_2), \tag{65}$$

$$G_2 = \frac{m_2 gl}{2} \cos(q_1 + q_2). \tag{66}$$

The forward and inverse kinematics as well as the differential kinematics defining the Jacobian $J(q)$ of the manipulator are straightforward to derive, but are omitted in this presentation to save space. The robot base is located at the origin, $(0, 0)$ m, and the desired path is to move along a circle centered in $(0.5, 1.5)$ m with a radius of 1 m. The starting point is $(0.5, 0.5)$ m and the motion is performed counter-clockwise. The inverse kinematics of the robot is used for transforming the desired path in Cartesian space to joint positions $q$ defining the path $f$ to be used in the trajectory generation. The trajectory generation was performed by solving the optimal control problem (32)–(34) using the convex optimization method presented in [15], which is possible since viscous joint friction is neglected in the model. When applying the nominal optimal trajectory in the PVC and PTVC architectures, model errors were introduced in the path-tracking simulations such that the actual link masses are $m_1 = 1.1$ kg and $m_2 = 1.1$ kg, *i.e.*, each link has 10 % higher mass than expected. The assumed maximum and minimum available torques in the trajectory planning were set to $\tau_{\max} = 40$ Nm and $\tau_{\min} = -40$ Nm, respectively, for each robot joint. The allowed torque utilization for each joint is the same in the simulations of the PVC and PTVC algorithms as in the trajectory planning. The numerical implementation of the trajectory generation was made in MATLAB and the convex optimization problem was solved using the CVX toolbox [58, 59]. Moreover, the parameter $K_\perp = 10 I_{2\times 2}$ in the simulation.

As pointed out earlier in Remark 2 and the first paragraph of Section 5, a direct application of the optimal solution as reference values $q_r(t)$, $\dot{q}_r(t)$, and

$\ddot{q}_r(t)$ to the basic controller (36) does not work satisfactory, but is nevertheless presented as a reference solution in this first example. However, considering that it is common practice in trajectory tracking to plan the trajectories slightly conservative using some margin for the actuator torques, the simulation of feedback-based trajectory tracking allows, in contrast to the simulations of the PVC and PTVC algorithms, an additional 5 % torque utilization compared to the trajectory planning. The algorithms and the robot motion were simulated using MATLAB Simulink.

The resulting path-tracking accuracy in Cartesian space is shown in Figure 5. Pure trajectory tracking leads, as expected, to large deviations from the desired path even though it was given extra torque margin during the evaluations. The PVC, though without torque margin, manages to maintain control authority, and the computed-torque controller can thus to some extent reduce the path-tracking error even without explicit control along the orthogonal directions. However, it is clear that the PTVC with explicit path corrections is significantly more effective than the PVC in handling the path deviations occurring in the lower left corner of the plane as a result of the model uncertainty. The input torques $\tau$ are shown in Figure 6 and it can be seen that, at all times, one input is close to its limit demonstrating that the close tracking is not achieved by conservatively decreasing the velocity and traversing the path slower than necessary. This preservation of high performance is further visualized in Figure 7 showing the path velocity $\dot{\sigma}$ together with the nominal time-optimal solution. Observing the differences in the path velocity for the considered algorithms, it is obvious that both PVC and PTVC reduce the speed along the path in order to handle the model error. In this example, PVC and PTVC require approximately the same time to complete the path, but the path-tracking accuracy with PTVC is increased.

### *7.2. Planar Robot Tracking Circle Segments and Straight Line*

The planar robot, considered in the simulation in Section 7.1, is further used to demonstrate the behavior of the control architecture for another type of path. This path is evaluated in the case when the robot has velocity-dependent actuator limits, despite that the trajectory planning assumed constant limits. The tracking task is a path composed of two circular segments with radius 0.8 m, joined by a straight line (see Figure 8). Hence, the curvature is zero during parts of the path and the transversal control needs to be handled using the alternative control law defined in Section 5.3.2 for this path segment. The nominal model parameters and the maximum and minimum input torques are the same as in the simulation in Section 7.1. The parameter for the feedback in the transversal control is $K_\perp = 10I_{2\times 2}$ in the simulations. The time-optimal trajectory planning was performed using the same procedure as for the example in Section 7.1.

In the evaluation, model errors are introduced such that $m_1 = 0.8$ kg and

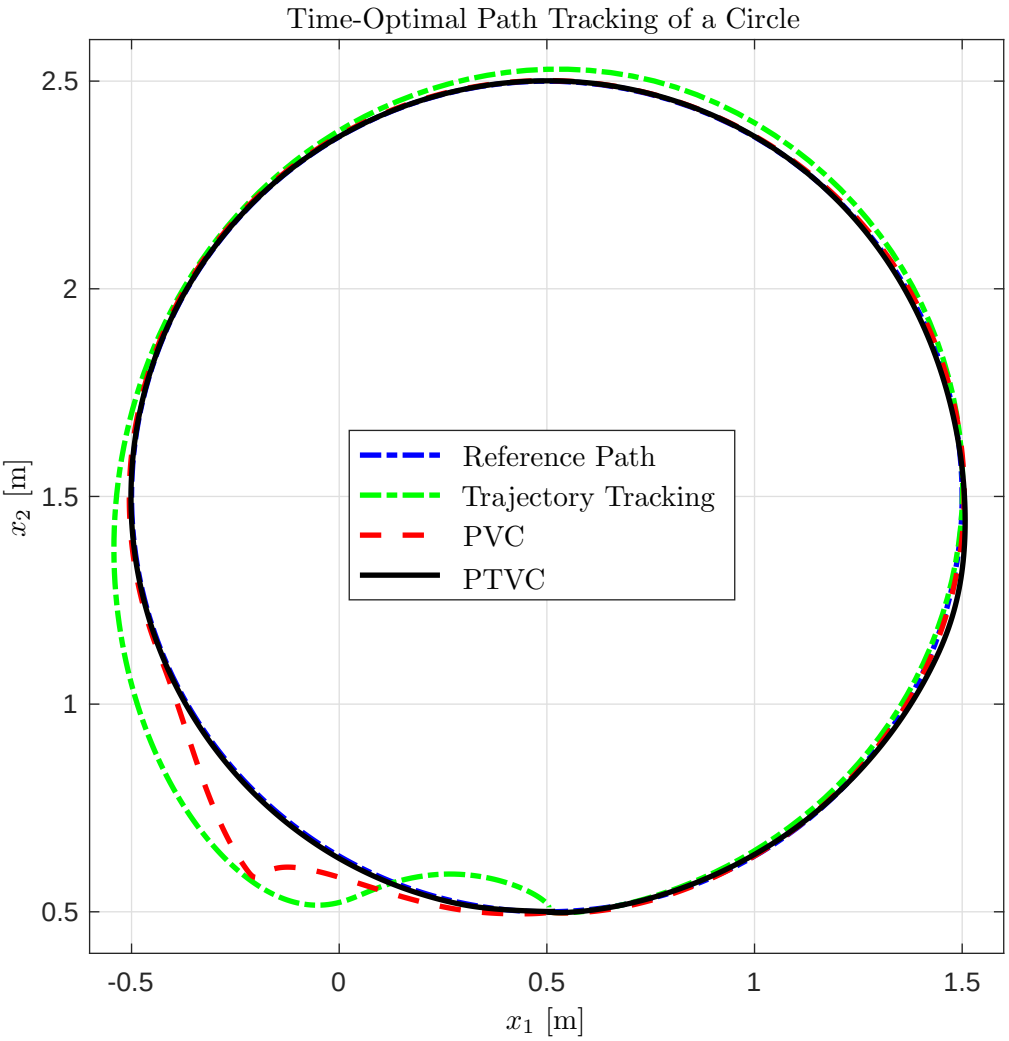


Figure 5: Robot end-effector positions $x_1$ and $x_2$ during time-optimal path tracking for a planar robot moving along a circle in a two-dimensional plane, see Section 7.1. In the simulation with trajectory tracking, the torque limits were 5 % higher compared to in the trajectory planning in order to allow some margin for the feedback control. The PVC and PTVC were not given any such extra margin, but still exhibit better path-tracking accuracy. It is clear that PTVC exhibits superior path-tracking performance.

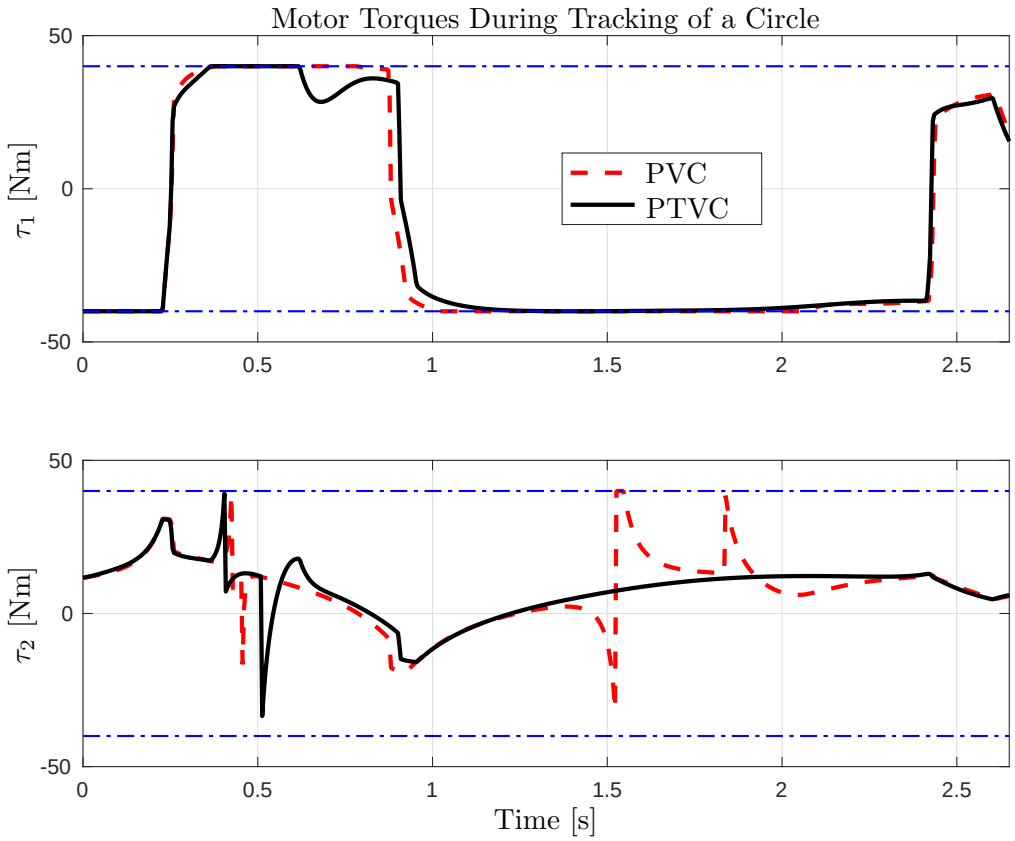


Figure 6: Input torques $\tau$ during time-optimal path tracking for a planar robot, corresponding to Figure 5. The torque limits are indicated by the horizontal blue lines.

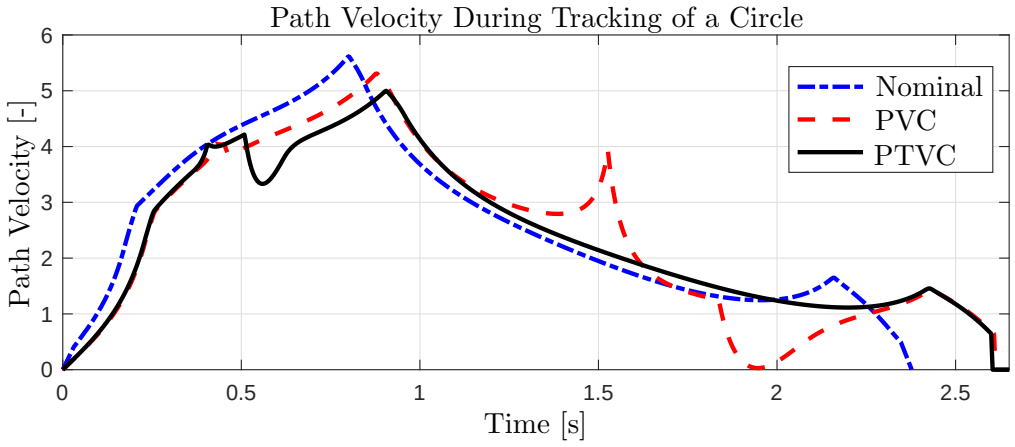


Figure 7: Path velocity $\dot{\sigma}$ during time-optimal path tracking for a planar robot, corresponding to Figure 5.

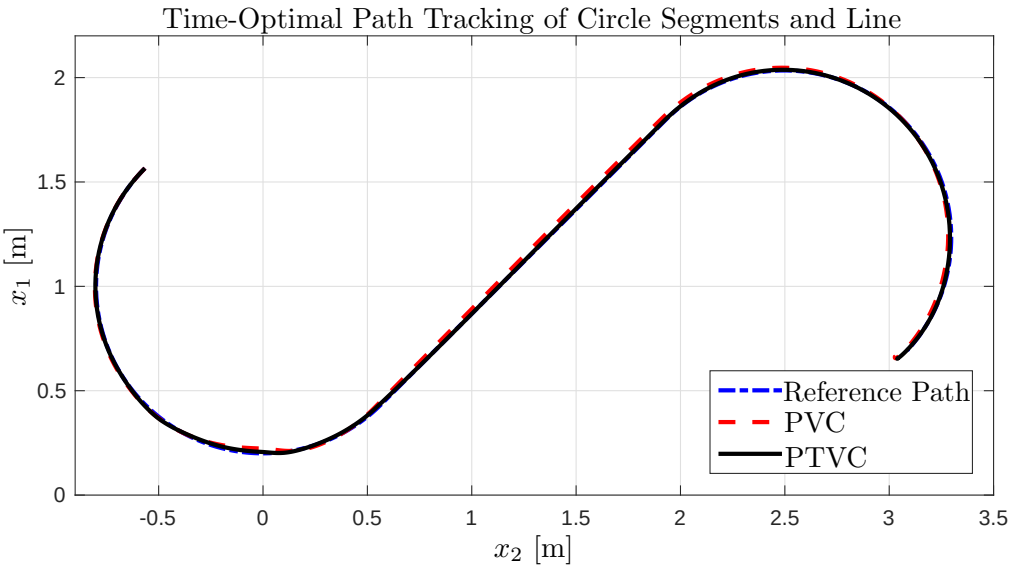


Figure 8: Robot end-effector positions $x_1$ and $x_2$ during time-optimal path tracking for a planar robot moving along a path composed of circular segments joined by a straight line, see Section 7.2.

$m_2 = 1.15$ kg, *i.e.*, one link is lighter and one link is heavier than what is assumed in the model. Further, velocity-dependent constraints are introduced to model the effect that the maximum torque decreases with increasing joint velocity because of the electromotive force. These constraints are introduced as follows

$$\tau_{i,\max} = 45 - 5|\dot{q}_i|, \;\; i = 1, 2,$$
$$\tau_{i,\min} = -45 + 5|\dot{q}_i|, \;\; i = 1, 2.$$

The resulting paths in Cartesian space from simulations of time-optimal path tracking are presented in Figure 8 for PVC and PTVC. In both cases, the path starts in the upper left corner. The results show that both PVC and PTVC achieve high accuracy, but that the transversal control in PTVC increases the accuracy of the path tracking both in the circular parts and the straight part of the path. The input torques $\tau$ are shown in Figure 9. Note in particular the state-dependent constraints on the actuators. Considering that the trajectory generation is based on constant torque limits, scaling of the timing of the time-optimal trajectories is necessary, and this is automatically performed by the control architecture to satisfy the constraints of the actual system. The path velocity in the respective simulation is visualized in Figure 10. Comparing the obtained path velocity with the nominal time-optimal, it is clear that both algorithms dilate the trajectories in time in order to handle the model uncertainty

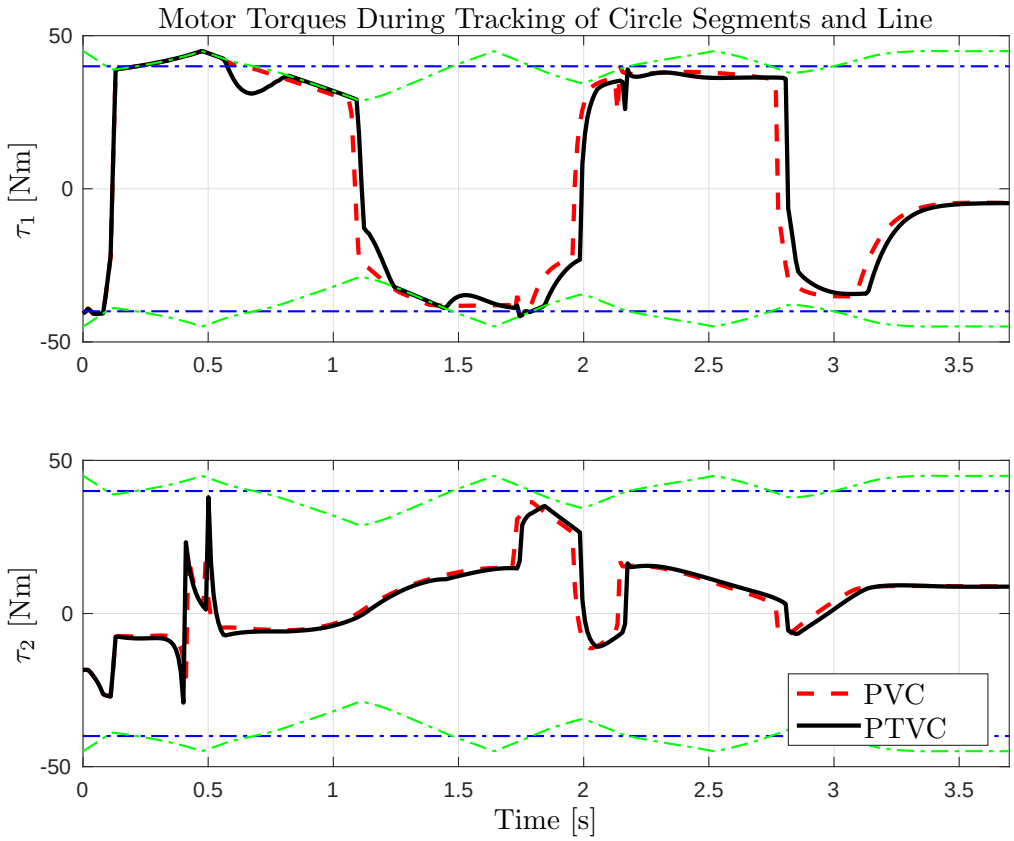


Figure 9: Input torques $\tau$ during time-optimal path tracking for a planar robot, corresponding to Figure 8. The constant torque limits used in the trajectory planning (blue) and the actual time-varying and velocity-dependent torque limits for simulation of the PTVC algorithm (green) are indicated. The corresponding torque limits for the simulation of the PVC algorithm are not shown for reasons of visibility, but are close to the limits in the PTVC simulation, though slightly translated in time.

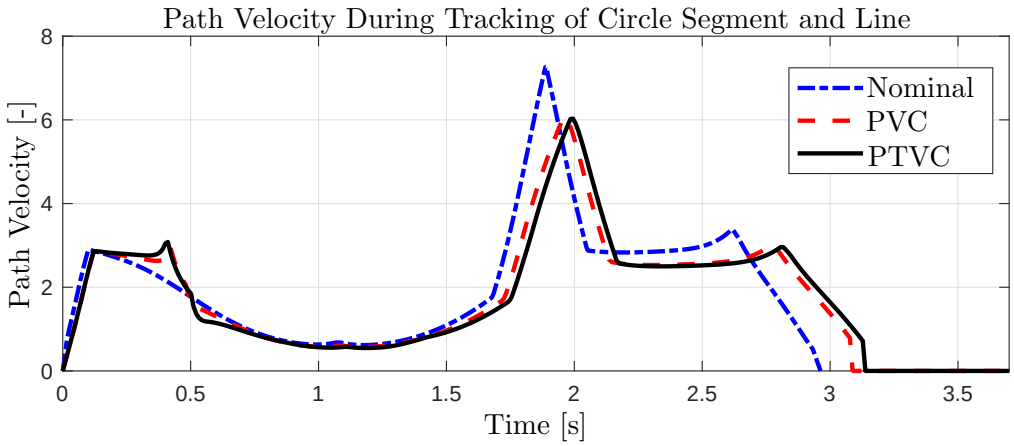


Figure 10: Path velocity $\dot{\sigma}$ during time-optimal path tracking for a planar robot, corresponding to Figure 8.

and the state-dependent constraints. It can be observed that the differences between PVC and PTVC are minor with respect to the path velocity throughout the maneuver, and that they require about the same time to finish the path.

### *7.3. Robot with Three DoF Tracking Path in 3D*

In order to investigate the performance on a robot system with more complex dynamics than the planar robot, the path-tracking algorithm is further evaluated on a robot manipulator with three DoF, see Section 2.9.4 in [50]. The masses of link two and three are $m_2 = m_3 = 1$ kg and the lengths of link two and three are $l = 2$ m. The moment-of-inertia matrices for the links are assumed diagonal with elements $I = 1$ kgm$^2$. The dynamic equations for this robot are straightforward to derive using appropriate software for symbolic manipulations of algebraic expressions, according to the strategy defined in textbooks on robot dynamics, see [50, 51]. The format is the same as for the planar robot, see (58), but the elements of the matrices are omitted to save space. Also in this simulation, no viscous friction in the joints was assumed. The robot base is located at the origin, $(0, 0, 0)$ m, and the desired path consists of segments of circles in three dimensions, starting in the point $(1, 0, 1.5)$ m. The time-optimal trajectories were generated using the same procedure as in Sections 7.1–7.2, with the same maximum and minimum torque limits $\tau_{\max}$ and $\tau_{\min}$, being $\pm 40$ Nm. In the simulations, a modeling error was introduced such that $m_3 = 1.2$ kg, *i.e.*, the mass of the third link is 20 % higher than what is assumed in the model. As in the path-tracking example considered in Section 7.1, the same limits on the available torque were used in the simulation of the PVC and PTVC algorithms

and in the trajectory planning. The parameter for the feedback in the transversal control is $K_\perp = 50 I_{3\times 3}$ in the simulations.

The simulation results comparing the path-tracking accuracy of PVC and PTVC in Cartesian space are shown in Figure 11. It can be observed that PTVC exhibits good path-tracking performance for the considered path in three dimensions. The input torques $\tau$ during the motion are provided in Figure 12, together with the actuator limits. As previously discussed, a necessary condition for time-optimality of the path traversal is that at least one input torque is at its limit at each time point [56]. Observing the results in Figure 12, this property is indeed indicated since at least one motor torque is close to or at its limit. In addition, the path velocity obtained for the PVC and the PTVC algorithms is visualized in Figure 13. It can here be observed that the PTVC algorithm both achieves higher path-tracking accuracy and requires slightly shorter time for finishing the path traversal.

## 8. Discussion

As can be observed in the simulation results presented in Section 7, the control architecture for combined path tracking and path-velocity control, PTVC, exhibits excellent results in terms of tracking accuracy. This is not obtained by decreasing the velocity conservatively—instead high-performance character with good torque utilization is preserved. It is thus in place to give a number of remarks and observations regarding the generality of the method, as well as a discussion of limitations and possible extensions of the control architecture.

### *8.1. Definition of the Geometric Path*

The only requirement on the geometric path to be tracked is the smoothness, as defined in Section 4.2. Should there be sharp corners, where the tangent to the path is not defined, there are two options. The trajectory planning may in this case be allowed to locally modify the path such that the corners are smoothed according to predefined specifications. An example here is a spline description with limits on the path deviation. The other possibility is to be strict on the path tracking and then the strategy is to move to the corner and stop, restart, and then continue. With these strategies, any realistic geometric path can be tracked.

Both paths in the examples in Sections 7.2–7.3 consist of different segments, as seen in Figures 8 and 11. In line with the presentation in Sections 2.2 and 3, each part of the path has its own local coordinate axes in the osculating plane corresponding to that segment. The pieces of the path are connected directly, without any smoothing, and thus it is noteworthy how well the tracking is performed for interconnected segments.

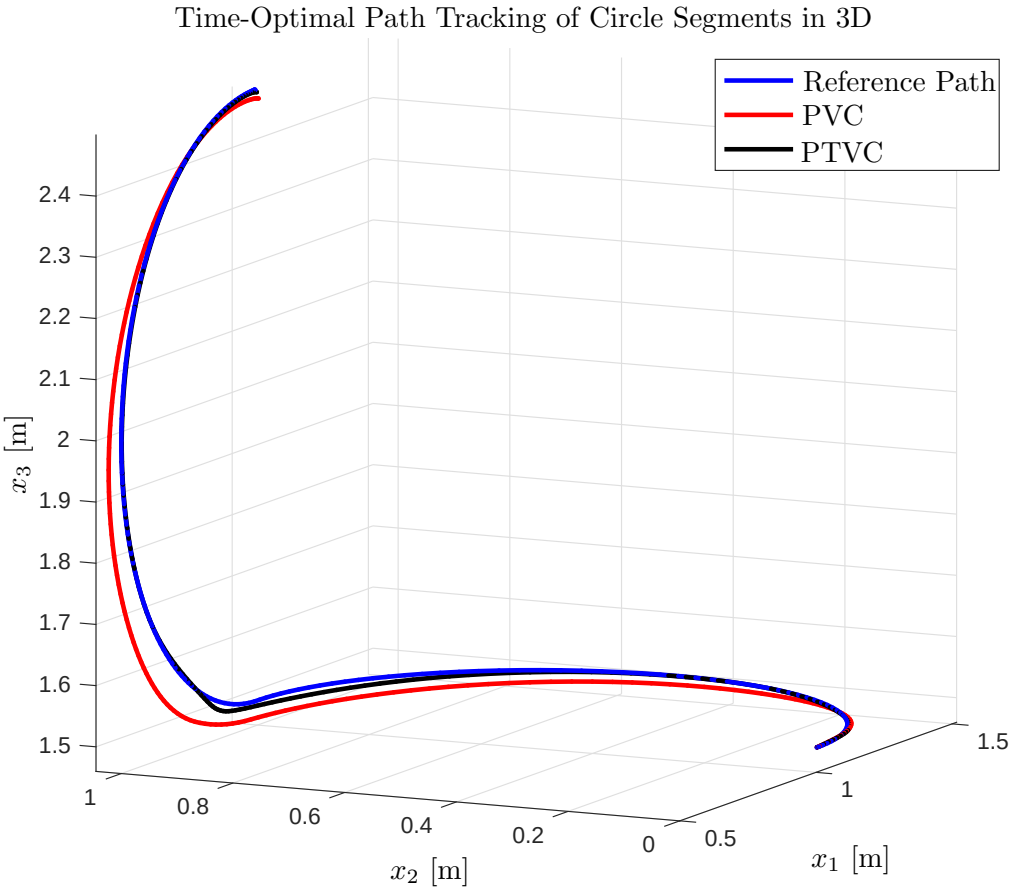


Figure 11: Robot end-effector positions $x_1$, $x_2$, and $x_3$ during time-optimal path tracking for a robot with three DoF moving along a path in a three-dimensional space, see Section 7.3.

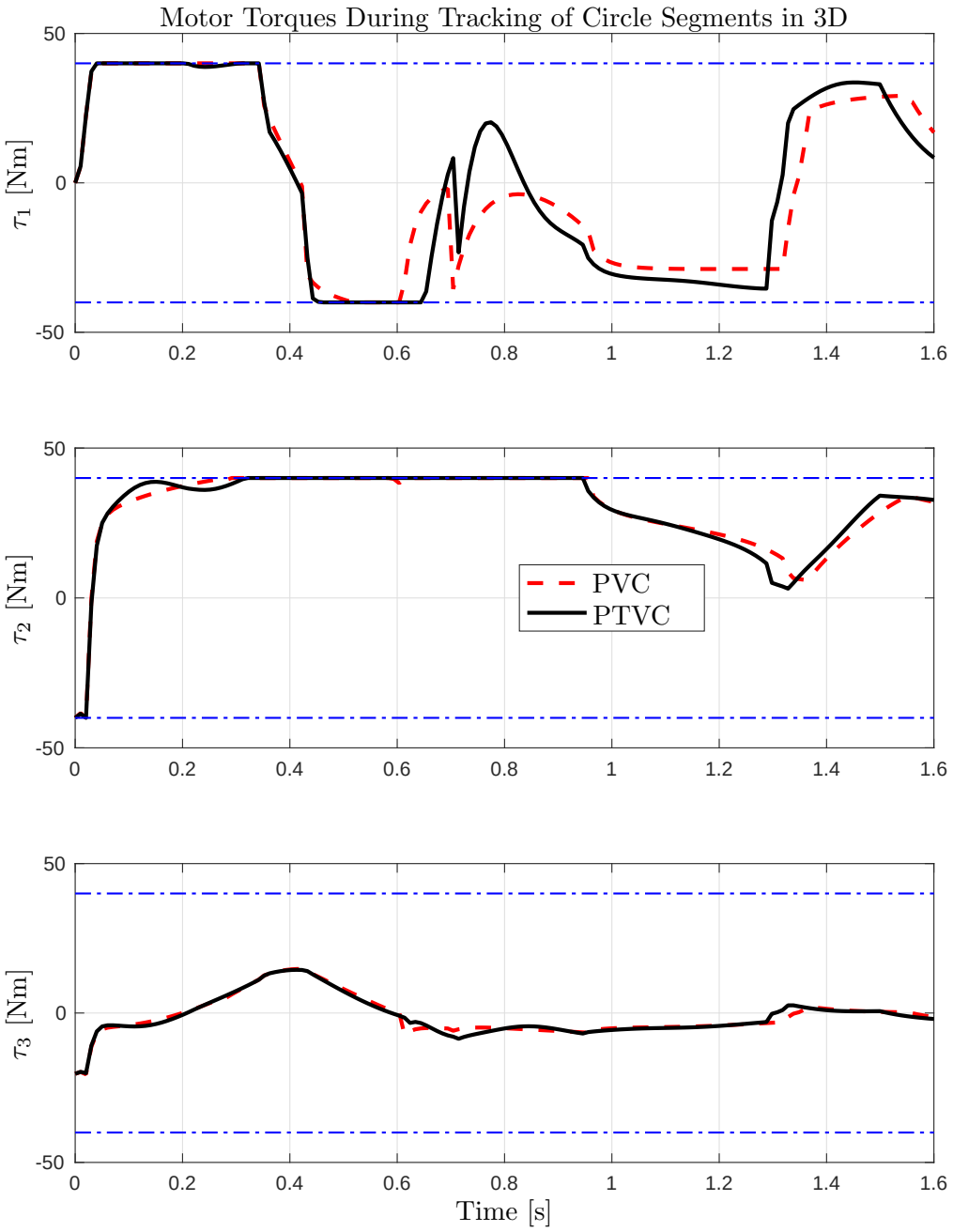


Figure 12: Input torques $\tau$ during time-optimal path tracking for a robot with three DoF, corresponding to Figure 11. The torque limits are indicated by the horizontal blue lines.

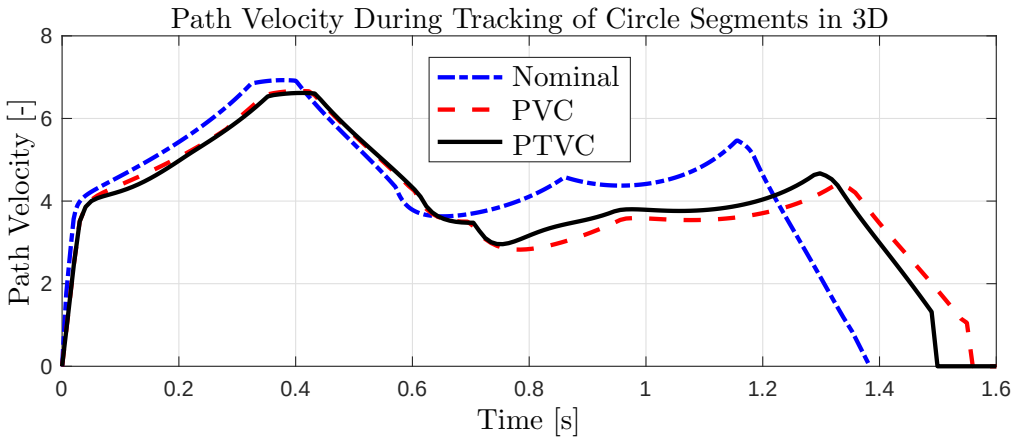


Figure 13: Path velocity $\dot{\sigma}$ during time-optimal path tracking for a robot with three DoF, corresponding to Figure 11.

Regarding the convergence to the path in the case of a deviation, there is no strict requirement that all points along the path must be visited. Rather, the approach taken here is to return to the path as soon as possible, given the limitations on the control inputs.

### *8.2. Generality Regarding the Trajectory Planning*

In the simulation examples considered in Section 7, the trajectory planning was performed using a time-optimal criterion, see Section 4.3. It should be stressed that any trajectory-planning algorithm can be used with the PTVC. The only requirement is that the computed trajectory can be specified by the path velocity as function of the path coordinate (*i.e.*, the distance along the path). This is in almost all cases possible by using the transformations (27)–(28), or if needed also time-derivatives of higher order. Naturally, time-optimal path tracking is of interest in many applications in order to maximize task effectiveness. However, due to potential wear of the actuators, modified criteria with combinations of time-optimal and energy-optimal can be used instead [14]. PTVC has no limitations with regard to this, so any trajectory planning can be used provided the monotonicity/bijectivity between the path coordinate $s$ and time $t$.

### *8.3. Controller for Trajectory Tracking*

The computed-torque control law was used for tracking of the computed reference values for the generalized coordinates when simulating the experiments presented in Section 7. However, in principle any reasonable controller can be used. The only requirement is that the chain rule can be applied as in (37)–(39),

such that the controller can be parametrized on the desired form (46). This is in general not a restrictive assumption, as already discussed in Section 5.5.

Insights have been gained in the tuning of the parameters in the trajectory-tracking controller, such as the PD-controller parameters $K_v$ and $K_p$ in the computed-torque control law (36). An advantage with the PTVC algorithm is that high gains are not necessary for path tracking, since this is obtained by the scheme itself. Instead, it is sufficient to tune the controller parameters as usual for sufficient set-point control and disturbance rejection, with consideration to noise in the measurement signals.

### *8.4. Additional Degrees of Freedom*

For robot manipulators, it is common that not only the path of the tool is specified, but also its orientation. Even though the 3D path has been the primary objective of the control in this paper, it is straightforward to extend the path-tracking algorithm to also achieve tracking of the desired orientation of a coordinate system associated with the system (important examples include the robot tool). Such an extension requires the consideration of the full 6D motion. For 3D motion, the path is a sequence of positions, whereas for 6D it is a sequence of positions and orientations. Using the inverse kinematics, the path $f(s)$ for the generalized coordinates (now 6D) is defined exactly as in (25). The parallel direction $x_{\parallel}$ along the tangent of the path defines a one-dimensional manifold where PVC is applied as before. For the orthogonal dimensions $x_{\perp}$, now typically 5D, the deviations from the desired path are handled similarly as defined in (41)–(42) in Section 5.

Even though it is outside the scope of this paper, another situation that could be considered is the case where the robot has more DoF than tracking of the specified path requires, *i.e.*, the case of redundant manipulation. In this scenario, the path in Cartesian space does not unambiguously define the joint coordinates. However, once the path for the generalized coordinates have been determined, the trajectory planning is analogous to the case of non-redundant robots in the key aspect that time-optimal path tracking would imply that at least one actuator is at the limit. Consequently, the potential benefits of online trajectory scaling are immediate. An interesting extension of the PTVC architecture in this paper, would be to investigate if the possible nullspace motion could be utilized.

### *8.5. Additional Sensors in Workspace*

Another interesting feature of PTVC is the ability to include workspace position measurements in the motion controller. It is well-known in manipulator robotics, where the joint motor positions typically are the only sensors available, that the accuracy of the tool position is limited by the accuracy of the kinematic calibration and dynamic properties such as joint and link flexibilities in

high-speed motions. Inclusion of explicit workspace measurements thus has high potential. This can easily be incorporated in the proposed architecture, since the path-tracking corrections in (41)–(42) in Section 5.3 could be based on actual workspace measurements instead of the estimates provided by the forward kinematics.

*8.6. Extensions to other Mechanical Systems*

An interesting extension of the PTVC algorithm is to consider other mechanical systems such as ground vehicles and mobile robot platforms. Since the main requirements are the bijectivity between $s$ and $t$ for the planned trajectory, and that the controller can be parametrized using the chain rule, which is almost always the case, the basic formulation is straightforward. The main difficulties remaining are those related to non-holonomic constraints and combined longitudinal and lateral wheel friction, *i.e.*, the dynamics can not be written on the form (24). For certain non-holonomic systems, this is not a problem and this was remarked in Section 4 for the flexible-joint manipulator. In other cases, questions remain to be resolved, and more specifically, what needs to be done in an extension concerning mobile platforms and vehicles is to ensure that the path-velocity control avoids exceeding the maximum possible interaction forces between the wheels and the ground.

## 9. Conclusions

A control architecture, path-tracking velocity control (PTVC), has been developed, analyzed, and evaluated in simulations. The control algorithm is based on existing path-velocity control (PVC), but in addition to velocity adaptation along the tangent of the path, the algorithm also has an explicit mechanism, with inherently attractive properties, for handling deviations transversal to the path. Consequently, the tangential direction is treated differently than its orthogonal directions, a fact that is conveniently described using the framework of natural coordinates, which also handily provides the radius of curvature of the path. The dynamics for the motion along the orthogonal directions are by feedback constructed such that the convergence to the path does not hamper the velocity control along the path. The actual velocity along the path influences the convergence to the path *via* the radius of curvature as a time function, and for iterated design of this aspect a framework of quantitative convergence analysis was presented in Section 6. Compared to trajectory tracking or existing PVC, the results in Section 7 show that the PTVC exhibits much better path-tracking behavior. Still, it maintains the high-performance character with high torque utilization, such that the traversal time is in the same order as for the PVC, or in some cases even better. In conclusion, PTVC exhibits superior performance compared to PVC.

## Acknowledgments

This research was supported by ELLIIT, the Strategic Area for ICT research, funded by the Swedish Government, and the European Commission's Seventh Framework Program under grant agreement SMErobotics (ref. #287787). B. Olofsson is member of the LCCC Linnaeus Center at Lund University, supported by the Swedish Research Council.

## References

[1] K. Kant, S. W. Zucker, Toward efficient trajectory planning: The path-velocity decomposition, Int. J. Robotics Research 5 (3) (1986) 72–89.

[2] S. M. LaValle, Planning Algorithms, Cambridge Univ. Press, Cambridge, UK, 2006.

[3] G. Conte, S. Duranti, T. Merz, Dynamic 3D path following for an autonomous helicopter, in: Proc. IFAC Symp. Intelligent Autonomous Vehicles, Lisbon, Portugal, 2004.

[4] G. M. Hoffmann, S. L. Waslander, C. J. Tomlin, Quadrotor helicopter trajectory tracking control, in: Proc. AIAA Guidance, Navigation and Control Conf. and Exhibit, Honolulu, HI, 2008, pp. 1–14.

[5] M. Hehn, R. Ritz, R. D'Andrea, Performance benchmarking of quadrotor systems using time-optimal control, Autonomous Robots 33 (1–2) (2012) 69–88.

[6] J. M. Hollerbach, Dynamic scaling of manipulator trajectories, J. Dynamic Systems, Measurement, and Control 106 (1) (1984) 102–106.

[7] J. E. Bobrow, S. Dubowsky, J. S. Gibson, On the optimal control of robotic manipulators with actuator constraints, in: Proc. American Control Conf. (ACC), San Francisco, CA, 1983, pp. 782–787.

[8] J. E. Bobrow, S. Dubowsky, J. S. Gibson, Time-optimal control of robotic manipulators along specified paths, Int. J. Robotics Research 4 (3) (1985) 3–17.

[9] F. Pfeiffer, R. Johanni, A concept for manipulator trajectory planning, IEEE J. Robot. Autom. 3 (2) (1987) 115–123.

[10] K. G. Shin, N. D. McKay, Minimum-time control of robotic manipulators with geometric path constraints, IEEE Trans. Autom. Control 30 (6) (1985) 531–541.

[11] Z. Shiller, Time-energy optimal control of articulated systems with geometric path constraints, in: Proc. IEEE Int. Conf. Robotics and Automation (ICRA), Vol. 4, 1994, pp. 2680–2685.

[12] K. G. Shin, N. D. McKay, Robust trajectory planning for robotic manipulators under payload uncertainties, IEEE Trans. Autom. Control 32 (12) (1987) 1044–1054.

[13] Z. Shiller, On singular time-optimal control along specified paths, IEEE Trans. Robot. Autom. 10 (4) (1994) 561–566.

[14] D. Verscheure, B. Demeulenaere, J. Swevers, J. De Schutter, M. Diehl, Time-energy optimal path tracking for robots: a numerically efficient optimization approach, in: Proc. 10th Int. Workshop Advanced Motion Control, Trento, Italy, 2008.

[15] D. Verscheure, B. Demeulenaere, J. Swevers, J. De Schutter, M. Diehl, Time-optimal path tracking for robots: A convex optimization approach, IEEE Trans. Autom. Control 54 (10) (2009) 2318–2327.

[16] F. Debrouwere, W. Van Loock, G. Pipeleers, Q. Tran Dinh, M. Diehl, J. De Schutter, J. Swevers, Time-optimal path following for robots with convex-concave constraints using sequential convex programming, IEEE Trans. Robot. 29 (6) (2013) 1485–1495.

[17] D. Verscheure, M. Diehl, J. De Schutter, J. Swevers, On-line time-optimal path tracking for robots, in: Proc. IEEE Int. Conf. Robotics and Automation (ICRA), Kobe, Japan, 2009, pp. 599–605.

[18] D. Verscheure, M. Diehl, J. De Schutter, J. Swevers, Recursive log-barrier method for on-line time-optimal robot path tracking, in: Proc. American Control Conf. (ACC), St. Louis, MI, 2009, pp. 4134–4140.
[19] T. Lipp, S. Boyd, Minimum-time speed optimization over a fixed path, Int. J. Control 87 (6) (2014) 1297–1311.
[20] S. Arimoto, S. Kawamura, F. Miyazaki, Bettering operation of robots by learning, J. Robotic Systems 1 (2) (1984) 123–140.
[21] F. Miyazaki, S. Kawamura, M. Matsumori, S. Arimoto, Learning control scheme for a class of robot systems with elasticity, in: Proc. Conf. Decision and Control (CDC), Vol. 25, Athens, Greece, 1986, pp. 74–79.
[22] M. Norrlöf, An adaptive iterative learning control algorithm with experiments on an industrial robot, IEEE Trans. Robot. Autom. 18 (2) (2002) 245–251.
[23] M. Norrlöf, Iterative learning control: Analysis, design, and experiments, Ph.D. thesis, Linköping University, Sweden, Thesis No. 653 (2000).
[24] X. Li, J.-X. Xu, D. Huang, An iterative learning control approach for linear systems with randomly varying trial lengths, IEEE Trans. Autom. Control 59 (7) (2014) 1954–1960.
[25] J.-X. Xu, Direct learning of control efforts for trajectories with different time scales, IEEE Trans. Autom. Control 43 (7) (1998) 1027–1030.
[26] P. Janssens, W. Van Loock, G. Pipeleers, F. Debrouwere, J. Swevers, Iterative learning control for optimal path following problems, in: Proc. Conf. Decision and Control (CDC), Florence, Italy, 2013, pp. 6670–6675.
[27] N. Sarkar, X. Yun, V. Kumar, Dynamic path following: a new control algorithm for mobile robots, in: Proc. Conf. Decision and Control (CDC), 1993, pp. 2670–2675.
[28] M. Egerstedt, X. Hu, A. Stotsky, Control of mobile platforms using a virtual vehicle approach, IEEE Trans. Autom. Control 46 (11) (2001) 1777–1782.
[29] J. Hauser, A. Banaszuk, Approximate feedback linearization around a trajectory: Application to trajectory planning, in: Proc. Conf. Decision and Control (CDC), Tampa, FL, 1997, pp. 7–11.
[30] F. Bayer, J. Hauser, Trajectory optimization for vehicles in a constrained environment, in: Proc. Conf. Decision and Control (CDC), Maui, Hawaii, 2012, pp. 5625–5630.
[31] O. Sørdalen, C. Canudas de Wit, Exponential control law for a mobile robot: extension to path following, IEEE Trans. Robot. Autom. 9 (6) (1993) 837–842.
[32] O. Dahl, L. Nielsen, Torque limited path following by on-line trajectory time scaling, IEEE Trans. Robot. Autom. 6 (5) (1990) 554–561.
[33] O. Dahl, Path constrained robot control, Ph.D. thesis, Department of Automatic Control, Lund University, Sweden, ISRN LUTFD2/TFRT--1038--SE (1992).
[34] O. Dahl, Path constrained robot control—experimental evaluation, Mechatronics 4 (2) (1994) 173–198.
[35] J. Kieffer, A. Cahill, M. James, Robust and accurate time-optimal path-tracking control for robot manipulators, IEEE Trans. Robot. Autom. 13 (6) (1997) 880–890.
[36] G. Antonelli, S. Chiaverini, G. Fusco, A new on-line algorithm for inverse kinematics of robot manipulators ensuring path tracking capability under joint limits, IEEE Trans. Robot. Autom. 19 (1) (2003) 162–167.
[37] O. Gerelli, C. Guarino Lo Bianco, Real-time path-tracking control of robotic manipulators with bounded torques and torque-derivatives, in: Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS), Nice, France, 2008, pp. 532–537.
[38] C. Guarino Lo Bianco, F. Wahl, A novel second order filter for the real-time trajectory scaling, in: Proc. IEEE Int. Conf. Robotics and Automation (ICRA), Shanghai, China, 2011, pp. 5813–5818.
[39] C. Guarino Lo Bianco, O. Gerelli, Online trajectory scaling for manipulators subject to high-order kinematic and dynamic constraints, IEEE Trans. Robot. 27 (6) (2011) 1144–

1152.
[40] G. Antonelli, C. Curatella, A. Marino, Constrained motion planning for open-chain industrial robots, Robotica 29 (2011) 403–420.
[41] C. Guarino Lo Bianco, F. Ghilardelli, Real-time planner in the operational space for the automatic handling of kinematic constraints, IEEE Trans. Autom. Sci. Eng. 11 (3) (2014) 730–739.
[42] A. De Luca, R. Farina, Dynamic scaling of trajectories for robots with elastic joints, in: Proc. IEEE Int. Conf. Robotics and Automation (ICRA), Vol. 3, Washington, DC, 2002, pp. 2436–2442.
[43] O. Wigström, B. Lennartson, A. Vergnano, C. Breitholtz, High-level scheduling of energy optimal trajectories, IEEE Trans. Autom. Sci. Eng. 10 (1) (2013) 57–64.
[44] W. Niu, M. Tomizuka, A new approach of coordinated motion control subjected to actuator saturation, J. Dynamic Systems, Measurement, and Control 123 (3) (2001) 496–504.
[45] H. Arai, K. Tanie, S. Tachi, Path tracking control of a manipulator considering torque saturation, IEEE Trans. Ind. Electron. 41 (1) (1994) 25–31.
[46] L. B. Freidovich, A. S. Shiriaev, Transverse linearization for underactuated nonholonomic mechanical systems with application to orbital stabilization, in: R. Johansson, A. Rantzer (Eds.), Distributed Decision Making and Control, Springer Verlag, London, 2012, pp. 245–258.
[47] A. Bemporad, T. Tarn, N. Xi, Predictive path parameterization for constrained robot control, IEEE Trans. Control Syst. Technol. 7 (6) (1999) 648–656.
[48] J. A. Thorpe, Elementary topics in differential geometry, Springer Verlag, Berlin Heidelberg, 1979.
[49] J. L. Meriam, L. G. Kraige, Engineering mechanics: Dynamics, Vol. 2, John Wiley & Sons, Hoboken, NJ, 2012.
[50] B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo, Robotics: Modelling, Planning and Control, Springer Verlag, London, 2009.
[51] M. W. Spong, S. Hutchinson, M. Vidyasagar, Robot Modeling and Control, John Wiley and Sons, Hoboken, NJ, 2006.
[52] O. Dahl, Path constrained motion optimization for rigid and flexible joint robots, in: Proc. IEEE Int. Conf. Robotics and Automation (ICRA), Atlanta, GA, 1993, pp. 223–229.
[53] J. Åkesson, Languages and tools for optimization of large-scale systems, Ph.D. thesis, Department of Automatic Control, Lund University, Sweden, ISRN LUTFD2/TFRT--1081--SE (2007).
[54] B. Olofsson, H. Nilsson, A. Robertsson, J. Åkesson, Optimal tracking and identification of paths for industrial robots, in: Proc. 18th World Congress of the International Federation of Automatic Control (IFAC), Milano, Italy, 2011, pp. 1126–1132.
[55] S. Boyd, L. Vandenberghe, Convex Optimization, 6th Edition, Cambridge Univ. Press, Cambridge, UK, 2004.
[56] Y. Chen, A. A. Desrochers, Structure of minimum-time control law for robotic manipulators with constrained paths, in: Proc. IEEE Int. Conf. Robotics and Automation (ICRA), Scottsdale, AZ, 1989, pp. 971–976.
[57] O. Dahl, L. Nielsen, Stability analysis of an online algorithm for torque limited path following, in: Proc. IEEE Int. Conf. Robotics and Automation (ICRA), Cincinnati, OH, 1990, pp. 1216–1222.
[58] CVX Research Inc., CVX: Matlab software for disciplined convex programming, version 2.0 beta, http://cvxr.com/cvx (2015).
[59] M. Grant, S. Boyd, Graph implementations for nonsmooth convex programs, in: V. Blondel, S. Boyd, H. Kimura (Eds.), Recent Advances in Learning and Control, Springer Verlag, Berlin Heidelberg, 2008, pp. 95–110.