Predictive Trajectory Tracking Algorithm of Underactuated Systems Based on the Calculus of Variations

Bálint Bodor[1]
Department of Applied Mechanics,
Budapest University of Technology and Economics,
Budapest 1111, Hungary
e-mail: bodor@mm.bme.hu

Ambrus Zelei
MTA-BME Research Group on Dynamics of Machines and Vehicles,
Budapest 1111, Hungary
e-mail: zelei@mm.bme.hu

László Bencsik
MTA-BME Research Group on Dynamics of Machines and Vehicles,
Budapest 1111, Hungary
e-mail: bencsik@mm.bme.hu

The tracking control of underactuated systems is a challenging problem due to the structural differences compared to fully actuated systems. Contrarily to fully actuated systems, resolving the inverse kinematics problem of underactuated systems is not possible independently from the dynamic equations. Instead, the inverse dynamics must be addressed. It is common to extend the computed torque control (CTC) technique with servoconstraints. Besides the CTC's clearness, the stability of the system cannot be always guaranteed. A novel predictive controller (PC) is presented in this paper. Our PC applies the variational principle to design the motion of the system in order to achieve a stable motion with the lowest possible tracking error. To demonstrate the applicability and the performance of the PC method, a numerical study is presented for a planar manipulator resulting in about 20% RMS error compared to the CTC method from the literature. [DOI: 10.1115/1.4051168]

1 Introduction

There are many definitions in the literature for underactuated systems. The most classic is the following: the system which has less independent control inputs than degrees-of-freedom (DoF) is called *underactuated* [1]. Manipulators with passive joints or flexible elements [1], cranes [2], and unmanned flying vehicles [3] all satisfy the above definition. Although underactuated systems necessitate intricate control, their typical benefit is energy efficiency and agility [4]. Nature provides many examples where these advantages are utilized, e.g., the elaborated balancing technique is the price of the gains achieved by bipedal walking, running, or hopping [5,6].

The trajectory tracking of underactuated systems is investigated in this paper. Control strategies of these systems may be derived with the generalizations of classic computed torque control methods [7–11] or the feedback linearization [12] which is called partial feedback linearization in the case of underactuated systems [13]. These techniques result in the required control inputs that realize the desired motion. This required input is considered as a feedforward term, and in practice, a feedback term is added too. We focus on the computation of the feedforward term in this paper.

The above-mentioned algorithms [7–13] can guarantee the stability of the so-called *controlled dynamics* only while the stabilization of the *internal dynamics* is not addressed. The states which are directly related to the desired task belong to the controlled dynamics. In contrast to underactuated ones, for fully actuated systems all of the states belong to the controlled dynamics normally. However, in the case of many underactuated systems, a particular part of the system cannot be controlled directly and it is called internal dynamics [14]. If the internal dynamics are unstable, then it leads to the stability loss of the whole system. Consequently, the above control algorithms can guarantee stability for systems possessing stable internal dynamics only.

One approach to stabilize a system with unstable internal dynamics can be done through the slight modification of the desired task (servoconstraints) [9]. Although stability can be achieved, the task modification is an intuitive process and, furthermore, the tracking accuracy is decreased. The authors presented a predictive approach in Ref. [15]; however, this also requires the modification of the original desired task. Another solution is to periodically switch between two desired tasks: (1) a modified one that results in a stable motion but not guarantees trajectory tracking; (2) and the original one that maintains the desired task but is unstable alone [16]. As a drawback, the periodic switching of the controller results in noncontinuous inputs that are hard to realize. It is also possible to optimize the internal dynamics of robots by connecting additional masses to the unactuated parts of the robot [17], however, this solution modifies the controlled system which may not always be possible.

Another approach is to formulate the task as an optimal control problem. The most widely known controller of this type is the linear quadratic regulator which is applicable to stabilize an unstable equilibrium state of a linear system. Linear quadratic tracking problems are also investigated in the literature [18,19]. As the objective of a continuous-time optimal control problem is functional depending on the states and the inputs, these controllers are closely related to the calculus of variations that is applied to derive the solutions to these problems. This is an analytical approach therefore these methods belong to the indirect optimal control solutions. Both linear quadratic regulator and linear-quadratic tracking controllers are designed for linear systems thus the application of these techniques requires further generalization. Although it is possible to achieve an accurate feedforward optimal controller for the nonlinear system by solving the global optimization problem numerically, the computational demand for solving such problems, e.g., based on the direct multiple-shooting method is very high [20].

The main difference compared to the earlier cited works is that instead of solving a global optimization problem numerically, consecutive linear optimization problems are tackled. The use of local optimal controls is combined with the receding horizon control [21,22] approach which finally results in a stable controller that can be computed online. This technique is applicable to underactuated systems with general structures. The design is briefly introduced in Refs. [23] and [24] where numerical tools have been used to synthesize the control algorithm. Contrarily to these, the work done in this paper presents a variational principle-based predictive control approach. In our solution, the motion of the system is examined on a finite time horizon and the goal is to find the motion which provides minimum tracking error and

[1]Corresponding author.

Manuscript received April 8, 2020; final manuscript received May 3, 2021; published online June 16, 2021. Assoc. Editor: Andreas Mueller.

stability for both the controlled and internal dynamics at the same time. These problems are dealt with semi-analytic tools, which, therefore, concludes in lower computational demand. This results in the possibility of real-time execution contrarily to the previously mentioned approach [20].

Summing up, the novelties of the proposed control algorithm, which utilizes the properties of mechanical systems, are: (1) the internal dynamics of the system is stabilized; (2) there is no need for the modification of either the task or the controlled system; (3) due to the receding horizon approach the optimization problem is rearranged in a linear form resulting in low computational demand.

The paper is organized as follows: Sec. 2 presents the dynamic modeling approach and the control algorithm. Section 3 demonstrates the feasibility and the effectiveness of the proposed method by means of numerical simulations carried out for the benchmark problem in Ref. [9]. Additionally, a second benchmark problem is presented with a higher degree-of-freedom and complexity to prove the wide range of applicability. Section 3.5 summarizes the results.

2 Problem Formulation and the Predictive Control Algorithm

The basic idea of the approach presented in this paper is to calculate a *dynamically possible* motion of the system (i.e., a motion that satisfies the equation of motion) which renders an appropriately defined cost function to a minimal value. First, we introduce the equation of motion, then write about the chosen cost function. Finally, we present the procedure of the optimization process.

2.1 Equation of Motion. We assume the equation of motion written in the following form, which is applicable for any type of underactuated rigid body system without the loss of generality

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}(\mathbf{q},\dot{\mathbf{q}}) = \mathbf{H}(\mathbf{q})\boldsymbol{\tau} \tag{1}$$

The motion of the f DoF system is described by the set of independent generalized coordinates $\mathbf{q}(t) \in \mathbb{R}^f$. The mass matrix is $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{f\times f}$ and the vector of the inertial and external forces is $\mathbf{h}(\mathbf{q},\dot{\mathbf{q}}) \in \mathbb{R}^f$. We denote the vector of the independent control forces/torques by $\boldsymbol{\tau} \in \mathbb{R}^m$, hence that the system has m number of independent inputs realized by the actuators. The distribution matrix of the actuator forces is $\mathbf{H}(\mathbf{q}) \in \mathbb{R}^{f\times m}$, with $m<f$. Note that in the case of $m=f$, when the system is fully actuated, the present algorithm may be simplified.

The equation of motion (1) is partitioned and transformed to the form $\mathbf{f} = \mathbf{0}$ and $\mathbf{g} = \mathbf{0}$ [1,25,26] which leads to

$$\mathbf{f}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}}) = \mathbf{M}_{\mathrm{a}}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}_{\mathrm{a}}(\mathbf{q},\dot{\mathbf{q}}) - \mathbf{H}_{\mathrm{a}}(\mathbf{q})\boldsymbol{\tau} \tag{2}$$

$$\mathbf{g}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}}) = \mathbf{M}_{\mathrm{u}}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}_{\mathrm{u}}(\mathbf{q},\dot{\mathbf{q}}) \tag{3}$$

with $\mathbf{f}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}}) \in \mathbb{R}^m$ and $\mathbf{g}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}}) \in \mathbb{R}^{f-m}$. Furthermore, on the right-hand side of the equation, we have $\mathbf{M}_{\mathrm{a}}(\mathbf{q}) \in \mathbb{R}^{m\times f}$, $\mathbf{M}_{\mathrm{u}}(\mathbf{q}) \in \mathbb{R}^{(f-m)\times f}$, $\mathbf{h}_{\mathrm{a}}(\mathbf{q},\dot{\mathbf{q}}) \in \mathbb{R}^m$, $\mathbf{h}_{\mathrm{u}}(\mathbf{q},\dot{\mathbf{q}}) \in \mathbb{R}^{f-m}$, and $\mathbf{H}_{\mathrm{a}}(\mathbf{q}) \in \mathbb{R}^{m\times m}$.

The first equation $\mathbf{f} = \mathbf{0}$ contains the generalized coordinates $\mathbf{q}$ and their derivatives, and the actuator forces $\boldsymbol{\tau}$. Since the control inputs are independent, the distribution matrix $\mathbf{H}$ in Eq. (1) has maximal column rank. Therefore it is possible to carry out the above transformation by multiplying Eq. (1) from left by a proper nonsingular transformation matrix $\mathbf{T}$ that satisfies

$$\begin{bmatrix} \mathbf{H}_{\mathrm{a}} \\ \mathbf{0} \end{bmatrix} = \mathbf{T}\mathbf{H} \tag{4}$$

with $\mathbf{H}_{\mathrm{a}}$ being invertible. This enables us to calculate the control torques/forces knowing a realizable $\mathbf{q}(t)$ motion of the system. Expressing $\boldsymbol{\tau}$ from Eq. (2), we obtain

$$\boldsymbol{\tau} = \mathbf{H}_{\mathrm{a}}^{-1}(\mathbf{M}_{\mathrm{a}}\ddot{\mathbf{q}} + \mathbf{h}_{\mathrm{a}}) \tag{5}$$

Equation (3) only depends on the $\mathbf{q}$ generalized coordinates and their time derivatives. From the viewpoint of the present control algorithm, $\mathbf{g} = \mathbf{0}$ is a constraint in the optimization problem introduced in the upcoming Secs. 2.2–2.5. We look for a motion $\mathbf{q}(t)$ that satisfies $\mathbf{g} = \mathbf{0}$, in which case the motion given by $\mathbf{q}(t)$ is realizable. After the optimization, we obtain the control inputs by applying Eq. (5).

2.2 Task Definition. One and widely used approach to formulate the task of a trajectory tracking problem is to use the so-called *servoconstraints* [27,28]. The violation of the servoconstraint is the tracking error formalized in the following form:

$$\mathbf{E}_{\mathbf{r}}(\mathbf{q},t) = \mathbf{r}_{\mathrm{d}}(t) - \mathbf{r}(\mathbf{q}) \tag{6}$$

where $\mathbf{r}(\mathbf{q}) \in \mathbb{R}^l$ is an arbitrary function expressed with the coordinates $\mathbf{q}$, and the desired value of $\mathbf{r}_{\mathrm{d}}(t) \in \mathbb{R}^l$ defined as an explicit function of the time t. The function $\mathbf{r}(\mathbf{q})$ may contain the *tool center point* (TCP) which corresponds to the position and/or the orientation of the end effector. The consequence of using the form (6) is that $\partial\mathbf{E}_{\mathbf{r}}/\partial t$ is purely time-dependent, while $\partial\mathbf{E}_{\mathbf{r}}/\partial\mathbf{q}$ merely depends on the coordinates.

The output is defined by the equation $\mathbf{E}_{\mathbf{r}}(\mathbf{q},t) = \mathbf{0}$. Often, the cause of the failure of the servoconstraint based methods is that $\mathbf{E}_{\mathbf{r}}(\mathbf{q},t) = \mathbf{0}$ is not possible to be fulfilled without the violation of dynamic Eq. (3) of the system. The problem here is that there is no well-known algorithm with which we could find out whether the desired motion is dynamically possible. Therefore it is not possible to prescribe arbitrary trajectories to inverse dynamics-based control algorithms as they may result in stability issues.

This fact leads us to an alternative way of the formulation of the task. There are two cases: when it is possible to fulfill the servoconstraint (that is $\mathbf{E}_{\mathbf{r}} = \mathbf{0}$ is possible to fulfill), the task is *realizable*. However, there are cases when it is not possible to fulfill the servoconstraint and the task is *unrealizable*, e.g., the desired trajectory $\mathbf{r}_{\mathrm{d}}(t)$ in (6) is not continuous, or it is out of the workspace of the robot, or (3) cannot be fulfilled. Our goal in this paper is to minimize the norm of $\mathbf{E}_{\mathbf{r}}$ in a later specified sense. In other words, instead of trying to solve the problem error-less, we look for a possible motion of the system which is the closest in some sense to the desired one.

For this, we define a cost function in Sec. 2.3 and look for the dynamically possible $\mathbf{q}$ function which renders this cost function to a minimal value. After the optimization problem is solved, the inputs $\boldsymbol{\tau}$ can be calculated from the realizable motion without stability issues applying Eq. (5).

2.3 Definition of the Cost Function. We define a scalar *cost function* as a measure of the distance between the realized motion and the desired one. We do not want to examine the motion only in a single time instant (as the servoconstraint based methods do [9,16,28]). Contrarily we aim to examine the motion over a finite time interval, which is called the *receding horizon control* approach [21,22,29]. To fulfill these requirements, it looks straightforward to use the integral of the quadratic error of the output $\mathbf{E}_{\mathbf{r}}$ defined in Eq. (6)

$$J_{\mathbf{r}}\langle\mathbf{q}\rangle = \int_{t_{\mathrm{s}}}^{t_{\mathrm{e}}} \mathbf{E}_{\mathbf{r}}^{\mathsf{T}}\mathbf{E}_{\mathbf{r}}\,dt \tag{7}$$

The limits of the integral are defined by the time horizon of the optimization problem: the starting time instant is t_{s}, while the ending time instant is t_{e}. The cost function (7) is mathematically functional depending on the generalized coordinates $\mathbf{q}$ and the integration limits t_{s} and t_{e}. It is trivial that this cost function is always non-negative, and it is nought if and only if the servoconstraint (6) is exactly fulfilled over the whole time horizon (that is $\mathbf{E_r} \equiv \mathbf{0}$).

The definition of the cost function $J_\mathbf{r}$ in Eq. (7) is not appropriate to setup the optimization task. When the servoconstraint is unrealizable (see Sec. 2.2), then it is possible that there is no motion of the system that gives an extrema for $J_\mathbf{r}$, e.g., the control inputs would need to be infinity. To overcome this problem, we include the input $\boldsymbol{\tau}$ defined in Eq. (5) in the cost function. The appropriate choice of the error vector is as follows:

$$\mathbf{E} = \mathbf{W}\begin{bmatrix} \mathbf{E_r} \\ \mathbf{H}_\mathrm{a}^{-1}(\mathbf{M}_\mathrm{a}\ddot{\mathbf{q}} + \mathbf{h}_\mathrm{a}) \end{bmatrix} \tag{8}$$

with $\mathbf{E} \in \mathbb{R}^{l+m}$. Here we use a weighing matrix $\mathbf{W}$, since the numerical values of the servoconstraints and the inputs may be in different dimensions. Furthermore, it allows us to tune the balance between accuracy and control action. Although other selections may be justifiable, in this paper the weighing matrix is constructed as the following diagonal matrix:

$$\mathbf{W} = \begin{bmatrix} w_\mathbf{r}\,\mathbf{I_r} & \mathbf{0} \\ \mathbf{0} & w_\tau\,\mathbf{I}_\tau \end{bmatrix} \tag{9}$$

Here $\mathbf{I_r} \in \mathbb{R}^{l\times l}$ and $\mathbf{I}_\tau \in \mathbb{R}^{m\times m}$ denotes the identity matrices. The scalar constants $w_\mathbf{r} > 0$ and $w_\tau > 0$ are the weighing factors of the error of the outputs $\mathbf{E_r}$ and the inputs $\boldsymbol{\tau}$, respectively. Then the cost function is defined in a similar way as previously in Eq. (7)

$$J\langle\mathbf{q}\rangle = \int_{t_\mathrm{s}}^{t_\mathrm{e}} \mathbf{E}^\mathsf{T}\mathbf{E}\,dt \tag{10}$$

The goal of present the predictive optimization method is to find the $\mathbf{q}$ time histories that minimize this functional.

2.4 Illustration of the Optimization Problem. To visualize the variety of solutions, Fig. 1 is included. Figure 1 illustrates the example of an underactuated planar manipulator of which the task is the tracking of the trajectory of the TCP shown by the straight black dashed line. Starting from the initial configuration, we have infinitely many possible TCP trajectories (plotted with continuous black lines) which are drawn in each branch with a stroboscopic view. We only included three of these TCP trajectories in the figure. The pale gray arrows show the time propagation. Each TCP trajectory can be reached by infinitely many configuration evolutions since the inverse kinematic problem of the manipulator is not unique [30–32]. These configurations are shown by the red, blue, and black stroboscopic views of the manipulator. However, not all of these kinematically possible configuration evolutions are dynamically possible. Only the black one satisfies the equation of motion (1), and therefore it is the only one, which is dynamically possible. The red and the blue ones do not satisfy the equation of motion (1).

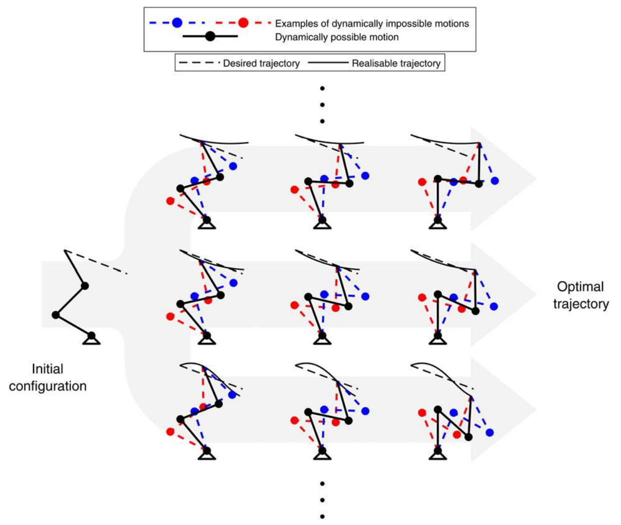

Fig. 1 Selection of the dynamically consistent optimal motion

Out of the three branches yielding different TCP trajectories, we choose the one in the second branch, since that one minimizes the cost function (10).

2.5 The Nonlinear Optimization Problem. To obtain the optimal motion of the system, we aim to find the extrema of the functional in Eq. (10) applying the constraint $\mathbf{g} = \mathbf{0}$ defined in Eq. (3). This constraint leads to a so-called isoperimetric problem [33]. To obtain the solution, we define the following Lagrangian with the vector of the multipliers $\boldsymbol{\mu}(t) \in \mathbb{R}^{f-m}$:

$$\mathcal{L} = \mathbf{E}^\mathsf{T}\mathbf{E} + \boldsymbol{\mu}^\mathsf{T}\mathbf{g} \tag{11}$$

With this Lagrangian $\mathcal{L}$, we setup the Euler–Lagrange equation [33]. The system of ordinary differential equations (ODE) that gives the optimal solution $\mathbf{q}$ is

$$\frac{\partial\mathcal{L}}{\partial\mathbf{q}} - \frac{d}{dt}\frac{\partial\mathcal{L}}{\partial\dot{\mathbf{q}}} + \frac{d^2}{dt^2}\frac{\partial\mathcal{L}}{\partial\ddot{\mathbf{q}}} = \mathbf{0} \tag{12}$$

$$\mathbf{g} = \mathbf{0} \tag{13}$$

In Eqs. (12) and (13), the unknown time-dependent functions are the generalized coordinates $\mathbf{q}$ and the Lagrange multipliers $\boldsymbol{\mu}$. Additionally, boundary conditions belong to the optimization problem

$$\dot{\mathbf{q}}\big|_{t_\mathrm{s}} = \dot{\mathbf{q}}_s \tag{14}$$

$$\mathbf{q}\big|_{t_\mathrm{s}} = \mathbf{q}_s \tag{15}$$

$$\left(\frac{\partial\mathcal{L}}{\partial\ddot{\mathbf{q}}}\right)\bigg|_{t_\mathrm{e}} = \mathbf{0} \tag{16}$$

$$\left(\frac{\partial\mathcal{L}}{\partial\dot{\mathbf{q}}} - \frac{d}{dt}\frac{\partial\mathcal{L}}{\partial\ddot{\mathbf{q}}}\right)\bigg|_{t_\mathrm{e}} = \mathbf{0} \tag{17}$$

In the initial time instant t_s the values $\mathbf{q}_s$ and $\dot{\mathbf{q}}_s$, respectively, the generalized coordinates and velocities are given, so we can prescribe that in Eqs. (14) and (15). However, in the final time instant t_e it is not possible to prescribe the configuration of the system. So we let the coordinates and the velocities be arbitrary in the time instant t_e. Therefore we specify the so-called transversality conditions [33] in the final time instant t_e, which are Eqs. (16) and (17). The linearization of the nonlinear boundary value problem (12)–(17) is presented in Secs. 2.6–2.9.

2.6 Time Horizon Parameters of the Controller. The best scenario would be to solve the nonlinear optimization problem on the whole maneuver. However, in a general case, this results in a nonlinear boundary value problem. For solving this, numerical methods could be used but these require a huge computational demand, which is not admissible as we would like to establish a fast control algorithm. To overcome this problem, we apply a receding horizon control technique [21,22] illustrated in Fig. 2. Instead of solving the nonlinear optimization problem on the whole time horizon, we generate a piecewise global solution by joining the sequence of local solutions of time interval Δt_a. The local solutions are possible to obtain with less computational demand applying a linear approximation.

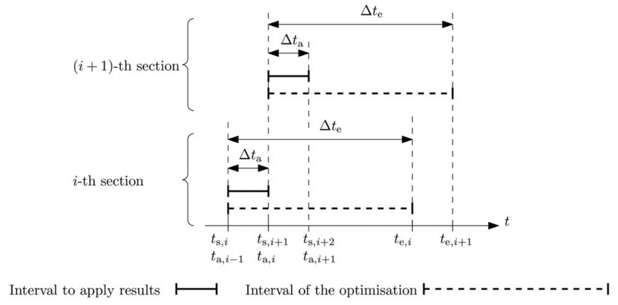
Δt_e
Δt_a
$(i+1)$-th section
Δt_e
Δt_a
i-th section
t
$t_{\mathrm{s},i}$ $t_{\mathrm{a},i-1}$
$t_{\mathrm{s},i+1}$ $t_{\mathrm{a},i}$
$t_{\mathrm{s},i+2}$ $t_{\mathrm{a},i+1}$
$t_{\mathrm{e},i}$
$t_{\mathrm{e},i+1}$
Interval to apply results
Interval of the optimisation

Fig. 2 Time intervals of the generated piecewise solution

Let us consider the time instant $t_{\mathrm{s},i}$ which is the initial time instant of the ith local optimization problem, which is preceded by the $(i-1)$th already solved local optimization problem. In the time instant $t_{\mathrm{s},i}$, we linearize the system around the current configuration and solve the ith optimization problem locally between $t_{\mathrm{s},i}$ and $t_{\mathrm{e},i}$. The linearization is detailed in Secs. 2.7–2.9. After determining the input $\boldsymbol{\tau}$, we apply it to control the system (1) on the shorter time interval between $t_{\mathrm{s},i}$ and $t_{\mathrm{a},i}$. Then a new, $(i+1)$th optimization problem is solved by setting $t_{\mathrm{s},i+1} = t_{\mathrm{a},i}$ and the process is continued again and again.

The local solutions are approximate therefore their accuracy is limited. Hence, the Δt_a application time length has to be chosen carefully. Furthermore, the selection of the optimization time horizon length Δt_e is also critical to obtain stability. The tuning of the time horizon intervals $\Delta t_\mathrm{a} < \Delta t_\mathrm{e}$ is studied in Sec. 3.5. The above-explained method belongs to the receding horizon control approaches [21,22,29].

2.7 Linearization of the Optimization Problem.

Now let us derive the local linear optimization problem for the ith section, which is introduced in Sec. 2.6 and Fig. 2. The notion of $\mathbf{q}_i$ and $\boldsymbol{\mu}_i$ is introduced for the solution of the ith linearized optimization problem. This is needed to emphasize the difference between the optimal motion $\mathbf{q}_i$ calculated from the linear optimization problem and the realized motion $\mathbf{q}$, which is the solution of the nonlinear equation of motion (1), accomplished by the inputs $\boldsymbol{\tau}$ calculated from the linear solution $\mathbf{q}_i$.

Our starting point is that the Euler–Lagrange Eqs. (12) and (13) and the boundary conditions (14)–(17) are linear, if the error vector $\mathbf{E}$ and the second-order nonholonomic constraint $\mathbf{g}$ defined in Eq. (3) and (8), respectively, are both linear in the generalized coordinates and their time derivatives. We define the following vectors for the sake of short notation:

$$\mathbf{x}^\mathsf{T} = [\mathbf{q}^\mathsf{T}, \dot{\mathbf{q}}^\mathsf{T}, \ddot{\mathbf{q}}^\mathsf{T}]^\mathsf{T} \tag{18}$$

$$\mathbf{x}_i^\mathsf{T} = [\mathbf{q}_i^\mathsf{T}, \dot{\mathbf{q}}_i^\mathsf{T}, \ddot{\mathbf{q}}_i^\mathsf{T}]^\mathsf{T} \tag{19}$$

The linearization is carried out at the time instant $t_{\mathrm{s},i} = t_{\mathrm{a},i-1}$ when the current coordinates, velocities, and accelerations computed from the preceding $(i-1)$th section are $\mathbf{x}_{\mathrm{s},i} = \mathbf{x}_{\mathrm{s},i} = \mathbf{x}|_{t=t_{\mathrm{s},i}}$. In the numerical case study of this paper, these values are computed from the solution of the nonlinear equation of motion. However, in a real application, these values are measured and therefore a closed-loop controller is obtained. With these values, we obtain the Taylor-series approximations for $\mathbf{g}(\mathbf{x})$ and $\mathbf{E}(\mathbf{x}, t)$ defined, respectively, in Eqs. (3) and (8)

$$\mathbf{E}(\mathbf{x}_i, t) \approx \mathbf{E}(\mathbf{x}_{\mathrm{s},i}, t) + \mathbf{E}'_\mathbf{x}(\mathbf{x}_{\mathrm{s},i})\,(\mathbf{x}_i - \mathbf{x}_{\mathrm{s},i}) \tag{20}$$

$$\mathbf{g}(\mathbf{x}_i) \approx \mathbf{g}(\mathbf{x}_{\mathrm{s},i}) + \mathbf{g}'_\mathbf{x}(\mathbf{x}_{\mathrm{s},i})\,(\mathbf{x}_i - \mathbf{x}_{\mathrm{s},i}) \tag{21}$$

In order to be able to derive a local analytical solution of the optimization problem (12)–(17), we approximate the time-dependent and constant terms applying the least squares approximation

$$\mathbf{E}(\mathbf{x}_{\mathrm{s},i}, t) - \mathbf{E}'_\mathbf{x}(\mathbf{x}_{\mathrm{s},i})\,\mathbf{x}_{\mathrm{s},i} \approx \mathbf{P}_{\mathbf{E},i}\,\boldsymbol{\varphi}(t) \tag{22}$$

$$\mathbf{g}(\mathbf{x}_{\mathrm{s},i}) - \mathbf{g}'_\mathbf{x}(\mathbf{x}_{\mathrm{s},i})\,\mathbf{x}_{\mathrm{s},i} \approx \mathbf{P}_{\mathbf{g},i}\,\boldsymbol{\varphi}(t) \tag{23}$$

with the vector $\boldsymbol{\varphi}(t) \in \mathbb{R}^n$ of linearly independent functions of time, e.g., polynomials or trigonometric functions and the constant matrices $\mathbf{P}_{\mathbf{E},i} \in \mathbb{R}^{(l+m)\times n}$ and $\mathbf{P}_{\mathbf{g},i} \in \mathbb{R}^{(f-m)\times n}$. The matrices $\mathbf{P}_{\mathbf{E},i}$ and $\mathbf{P}_{\mathbf{g},i}$ are numerically computed at this stage. The system of functions $\boldsymbol{\varphi}$ needs to satisfy the condition $\dot{\boldsymbol{\varphi}} = \mathbf{D}\boldsymbol{\varphi}$ with a constant $\mathbf{D} \in \mathbb{R}^{n\times n}$ differential operator matrix. The above Taylor-series approximation and least square approximation result in the form

$$\mathbf{E}(\mathbf{x}_i, t) \approx \mathbf{E}'_\mathbf{x}(\mathbf{x}_{\mathrm{s},i})\,\mathbf{x}_i + \mathbf{P}_{\mathbf{E},i}\,\boldsymbol{\varphi}(t) \tag{24}$$

$$\mathbf{g}(\mathbf{x}_i) \approx \mathbf{g}'_\mathbf{x}(\mathbf{x}_{\mathrm{s},i})\,\mathbf{x}_i + \mathbf{P}_{\mathbf{g},i}\,\boldsymbol{\varphi}(t) \tag{25}$$

Note that the derivative $\mathbf{E}'_\mathbf{x}$ does not depend on time explicitly, since its form is defined in Eq. (6).

Substituting $\mathbf{x}_i$ introduced in Eq. (19) and reordering the equation, we finally obtain the linearized form of $\mathbf{E}_i$ and $\mathbf{g}_i$ shown below:

$$\mathbf{E}_i \approx \boldsymbol{\alpha}_{\mathbf{E},i}\,\ddot{\mathbf{q}}_i + \boldsymbol{\beta}_{\mathbf{E},i}\,\dot{\mathbf{q}}_i + \boldsymbol{\gamma}_{\mathbf{E},i}\,\mathbf{q}_i + \mathbf{P}_{\mathbf{E},i}\,\boldsymbol{\varphi} \tag{26}$$

$$\mathbf{g}_i \approx \boldsymbol{\alpha}_{\mathbf{g},i}\,\ddot{\mathbf{q}}_i + \boldsymbol{\beta}_{\mathbf{g},i}\,\dot{\mathbf{q}}_i + \boldsymbol{\gamma}_{\mathbf{g},i}\,\mathbf{q}_i + \mathbf{P}_{\mathbf{g},i}\,\boldsymbol{\varphi} \tag{27}$$

Here $\boldsymbol{\alpha}_{\mathbf{E},i}$, $\boldsymbol{\beta}_{\mathbf{E},i}$, $\boldsymbol{\gamma}_{\mathbf{E},i}$ and $\boldsymbol{\alpha}_{\mathbf{g},i}$, $\boldsymbol{\beta}_{\mathbf{g},i}$, $\boldsymbol{\gamma}_{\mathbf{g},i}$ are constant matrices of dimensions $\mathbb{R}^{(l+m)\times f}$ and $\mathbb{R}^{(f-m)\times f}$, respectively. These approximations are valid only if the generalized velocities and accelerations are both at least $\mathcal{C}^1$ smooth. However, $\mathcal{C}^1$ continuity is not possible to ensure in the initial time instant $t_{\mathrm{s},i}$ of each section, since the boundary conditions in Eqs. (14)–(17) do not imply that, i.e., the acceleration cannot be prescribed in the initial time instant. However, with the appropriate selection of the time horizon parameter Δt_a, negligible errors occur (see Sec. 3.5). Equations (26) and (27) enables us to calculate $\dot{\mathbf{E}}_i$, $\ddot{\mathbf{E}}_i$, $\dot{\mathbf{g}}_i$ and $\ddot{\mathbf{g}}_i$ in a simple way

$$\dot{\mathbf{E}}_i \approx \boldsymbol{\alpha}_{\mathbf{E},i}\,\dddot{\mathbf{q}}_i + \boldsymbol{\beta}_{\mathbf{E},i}\,\ddot{\mathbf{q}}_i + \boldsymbol{\gamma}_{\mathbf{E},i}\,\dot{\mathbf{q}}_i + \mathbf{P}_{\mathbf{E},i}\,\mathbf{D}\boldsymbol{\varphi} \tag{28}$$

$$\ddot{\mathbf{E}}_i \approx \boldsymbol{\alpha}_{\mathbf{E},i}\,\ddddot{\mathbf{q}}_i + \boldsymbol{\beta}_{\mathbf{E},i}\,\dddot{\mathbf{q}}_i + \boldsymbol{\gamma}_{\mathbf{E},i}\,\ddot{\mathbf{q}}_i + \mathbf{P}_{\mathbf{E},i}\,\mathbf{D}^2\boldsymbol{\varphi} \tag{29}$$

$$\dot{\mathbf{g}}_i \approx \boldsymbol{\alpha}_{\mathbf{g},i}\,\dddot{\mathbf{q}}_i + \boldsymbol{\beta}_{\mathbf{g},i}\,\ddot{\mathbf{q}}_i + \boldsymbol{\gamma}_{\mathbf{g},i}\,\dot{\mathbf{q}}_i + \mathbf{P}_{\mathbf{g},i}\,\mathbf{D}\boldsymbol{\varphi} \tag{30}$$

$$\ddot{\mathbf{g}}_i \approx \boldsymbol{\alpha}_{\mathbf{g},i}\,\ddddot{\mathbf{q}}_i + \boldsymbol{\beta}_{\mathbf{g},i}\,\dddot{\mathbf{q}}_i + \boldsymbol{\gamma}_{\mathbf{g},i}\,\ddot{\mathbf{q}}_i + \mathbf{P}_{\mathbf{g},i}\,\mathbf{D}^2\boldsymbol{\varphi} \tag{31}$$

2.8 Linearized Euler–Lagrange Equation.

To derive the ODEs of the linear optimization problem, we apply the approximations (26)–(31) in Eqs. (12) and (13). One can see that the highest order time derivatives are $\ddddot{\mathbf{q}}$ and $\ddot{\boldsymbol{\mu}}$ in Eq. (12), and $\ddot{\mathbf{q}}$ in Eq. (13). Later, the highest order derivatives $\ddddot{\mathbf{q}}$ and $\ddot{\boldsymbol{\mu}}$ are expressed explicitly from Eqs. (12) and (13). Therefore we make $\ddddot{\mathbf{q}}$ appear in Eq. (13) too by time differentiating it twice. Then the Lagrangian $\mathcal{L}$ defined in Eq. (11) is substituted into Eq. (12) and the appropriate derivatives of $\mathbf{E}$ and $\mathbf{g}$ in $\partial\mathcal{L}/\partial\mathbf{q}$, $\partial\mathcal{L}/\partial\dot{\mathbf{q}}$ and $\partial\mathcal{L}/\partial\ddot{\mathbf{q}}$ are evaluated applying Eqs. (26)–(31). As an intermediate step, we obtain

$$(2\boldsymbol{\gamma}_{\mathbf{E},i}^\mathsf{T}\,\mathbf{E}_i + \boldsymbol{\gamma}_{\mathbf{g},i}^\mathsf{T}\,\boldsymbol{\mu}_i) - (2\boldsymbol{\beta}_{\mathbf{E},i}^\mathsf{T}\,\dot{\mathbf{E}}_i + \boldsymbol{\beta}_{\mathbf{g},i}^\mathsf{T}\,\dot{\boldsymbol{\mu}}_i) + (2\boldsymbol{\alpha}_{\mathbf{E},i}^\mathsf{T}\,\ddot{\mathbf{E}}_i + \boldsymbol{\alpha}_{\mathbf{g},i}^\mathsf{T}\,\ddot{\boldsymbol{\mu}}_i) = \mathbf{0} \tag{32}$$

$$\boldsymbol{\alpha}_{\mathbf{g},i}\,\ddddot{\mathbf{q}}_i + \boldsymbol{\beta}_{\mathbf{g},i}\,\dddot{\mathbf{q}}_i + \boldsymbol{\gamma}_{\mathbf{g},i}\,\ddot{\mathbf{q}}_i + \mathbf{P}_{\mathbf{g},i}\,\mathbf{D}^2\boldsymbol{\varphi} = \mathbf{0} \tag{33}$$

from Eqs. (12) and (13). We define the vector of the highest order derivatives $\mathbf{z}_{\mathrm{ho},i}$ and the vector of the lower order terms $\mathbf{z}_i$

$$\mathbf{z}_{\mathrm{ho},i} = [\ddddot{\mathbf{q}}_i^{\mathsf{T}}, \quad \ddot{\boldsymbol{\mu}}_i^{\mathsf{T}}]^{\mathsf{T}} \tag{34}$$

$$\mathbf{z}_i = [\dddot{\mathbf{q}}_i^{\mathsf{T}}, \quad \dot{\boldsymbol{\mu}}_i^{\mathsf{T}}, \quad \ddot{\mathbf{q}}_i^{\mathsf{T}}, \boldsymbol{\mu}_i^{\mathsf{T}}, \dot{\mathbf{q}}_i^{\mathsf{T}}, \mathbf{q}_i^{\mathsf{T}}]^{\mathsf{T}} \tag{35}$$

After substituting the approximated form of $\mathbf{E}_i$ (see Eq. (26)) and its derivatives in Eq. (32), Eqs. (32) and (33) are reordered in the following compact form:

$$\mathbf{A}_{\mathbf{z},\mathrm{ho},i}\,\mathbf{z}_{\mathrm{ho},i} + \mathbf{A}_{\mathbf{z},i}\,\mathbf{z}_i = \mathbf{P}_i\,\boldsymbol{\varphi} \tag{36}$$

Here $\mathbf{A}_{\mathbf{z},\mathrm{ho},i}$ and $\mathbf{A}_{\mathbf{z},i}$ are the coefficients of the higher and lower order terms, respectively. The matrix $\mathbf{P}_i$, which is an intricate combination of $\mathbf{P}_{\mathbf{E},i}$, $\mathbf{P}_{\mathbf{g},i}$, and $\mathbf{D}$, is the coefficient of the function system $\boldsymbol{\varphi}$, corresponding to the constant and time-dependent terms. From Eq. (36), we express the vector $\mathbf{z}_{\mathrm{ho},i}$ of the highest order derivatives

$$\mathbf{z}_{\mathrm{ho},i} = \mathbf{A}_{\mathbf{z},\mathrm{ho},i}^{-1}\,(-\mathbf{A}_{\mathbf{z},i}\,\mathbf{z}_i + \mathbf{P}_i\,\boldsymbol{\varphi}) \tag{37}$$

Then the first-order form of the linear Eq. (37) is written as

$$\dot{\mathbf{z}}_i = \mathbf{A}_{\mathrm{fof},i}\,\mathbf{z}_i + \mathbf{P}_{\mathrm{fof},i}\,\boldsymbol{\varphi} \tag{38}$$

In order to obtain a compact form, we define the constant matrices $\mathbf{A}_{\mathrm{fof},i} \in \mathbb{R}^{(6f-2m)\times(6f-2m)}$ and $\mathbf{P}_{\mathrm{fof},i} \in \mathbb{R}^{(6f-2m)\times n}$ as follows:

$$\mathbf{A}_{\mathrm{fof},i} = \left[\begin{array}{cc:cc:cc} \multicolumn{6}{c}{-\mathbf{A}_{\mathbf{z},\mathrm{ho},i}^{-1}\,\mathbf{A}_{\mathbf{z},i}} \\ \hdashline \mathbf{I}_{\mathbf{q}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{\boldsymbol{\mu}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \hdashline \mathbf{0} & \mathbf{0} & \mathbf{I}_{\mathbf{q}} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_{\mathbf{q}} & \mathbf{0} \end{array}\right] \tag{39}$$

$$\mathbf{P}_{\mathrm{fof},i} = \left[\begin{array}{c} -\mathbf{A}_{\mathbf{z},\mathrm{ho},i}^{-1}\,\mathbf{P}_i \\ \hdashline \mathbf{0} \\ \mathbf{0} \\ \hdashline \mathbf{0} \\ \mathbf{0} \end{array}\right] \tag{40}$$

where $\mathbf{I}_{\mathbf{q}} \in \mathbb{R}^{f\times f}$ and $\mathbf{I}_{\boldsymbol{\mu}} \in \mathbb{R}^{(f-m)\times(f-m)}$ denote the identity matrices.

2.9 Solution of the Linear Optimization Problem. Our goal is to obtain a closed-form sectionwise (see Sec. 2.6) solution for the linear ODE defined in Eq. (38). The solution for $\mathbf{z}_i$ is generated in two steps. First, the homogenous part of the ODE is solved using a matrix exponential

$$\mathbf{z}_{\mathrm{h},i} = e^{t\mathbf{A}_{\mathrm{fof},i}}\,\mathbf{c}_i \tag{41}$$

Here, $\mathbf{c}_i$ is a yet unknown vector that is determined below from the boundary conditions. Next, the particular solution of the inhomogeneous equation is derived assuming the form $\mathbf{z}_{\mathrm{p},i} = \mathbf{P}_{\mathrm{p},i}\,\boldsymbol{\varphi}$. Substituting this into Eq. (38), we obtain the algebraic equation for $\mathbf{P}_{\mathrm{p}}$

$$\mathbf{P}_{\mathrm{p},i}\,\mathbf{D}\boldsymbol{\varphi} = \mathbf{A}_{\mathrm{fof},i}\,\mathbf{P}_{\mathrm{p},i}\,\boldsymbol{\varphi} + \mathbf{P}_{\mathrm{fof},i}\,\boldsymbol{\varphi} \tag{42}$$

The elements of $\boldsymbol{\varphi}$ are linearly independent as we explained in Sec. 2.7, therefore the coefficients of $\boldsymbol{\varphi}$ are the same on both sides. We achieve the following linear equation:

$$-\mathbf{A}_{\mathrm{fof},i}\,\mathbf{P}_{\mathrm{p},i} + \mathbf{P}_{\mathrm{p},i}\,\mathbf{D} = \mathbf{P}_{\mathrm{fof},i} \tag{43}$$

This is a Sylvester equation [34] that is solved and hence the $\mathbf{P}_{\mathrm{p},i}$ matrix is numerically determined. Note that Eq. (42) does not necessarily have a solution, which is the case of mathematical resonance. With the selection of another $\boldsymbol{\varphi}$ function system this problem is possible to handle.

It is noted here that a particular solution $\mathbf{z}_{\mathrm{p},i}$ of Eq. (38) is possible to obtain by numerically integrating this first-order ODE with an arbitrary initial condition too. This way the least-squares approximations in Eqs. (22) and (23) are unnecessary steps and the accurate terms on the left-hand side of these equations could be applied. Either of the two solutions is capable of calculating the particular solution $\mathbf{z}_{\mathrm{p},i}$ and has its own advantages. The least squares approximations in Eqs. (22) and (23) are possible to calculate except for the constant terms before starting the maneuver, while the solution of the Sylvester Eq. (43) is possible to obtain with a low computing effort which is beneficial during the control of the system. On the other hand, numerically integrating Eq. (38) to get the particular solution is simpler in terms of the required algebraic manipulations.

Finally, the general solution of the linearized differential Eq. (38) is the sum of the homogeneous and the particular solutions

$$\mathbf{z}_{\mathrm{i}} = \mathbf{z}_{\mathrm{h},i} + \mathbf{z}_{\mathrm{p},i} = e^{t\mathbf{A}_{\mathrm{fof},i}}\,\mathbf{c}_i + \mathbf{P}_{\mathrm{p},i}\,\boldsymbol{\varphi} \tag{44}$$

Now we determine the specific solution that satisfies the boundary conditions as well, i.e., we calculate the numerical values in $\mathbf{c}_i$. Since Eq. (13) was time differentiated twice to obtain Eq. (33), the following conditions have to be satisfied in an arbitrary t^* time instant besides the boundary conditions defined in Eqs. (14)–(17)

$$\mathbf{g}_i|_{t=t^*} = \mathbf{0} \tag{45}$$

$$\dot{\mathbf{g}}_i|_{t=t^*} = \mathbf{0} \tag{46}$$

The value of this arbitrary time instant is chosen as $t^* = t_{\mathrm{e},i}$ shown in Fig. 2. Then the derivatives in Eqs. (16) and (17) are evaluated taking into account the form of $\mathbf{E}_i$, $\dot{\mathbf{E}}_i$, $\ddot{\mathbf{E}}_i$, $\mathbf{g}_i$, $\dot{\mathbf{g}}_i$, and $\ddot{\mathbf{g}}_i$ defined in Eqs. (26)–(31). The following system of equations is obtained as an intermediate step from the boundary conditions (14)–(17), (45), and (46):

$$\begin{bmatrix} \dot{\mathbf{q}}_i|_{t=t_{\mathrm{s},i}} \\ \mathbf{q}_i|_{t=t_{\mathrm{s},i}} \\ (2\boldsymbol{\alpha}_{\mathbf{E},i}^{\mathsf{T}}\,\mathbf{E}_i + \boldsymbol{\alpha}_{\mathbf{g},i}^{\mathsf{T}}\,\boldsymbol{\mu}_i)|_{t=t_{\mathrm{e},i}} \\ (2\boldsymbol{\beta}_{\mathbf{E},i}^{\mathsf{T}}\,\mathbf{E}_i + \boldsymbol{\beta}_{\mathbf{g},i}^{\mathsf{T}}\,\boldsymbol{\mu}_i - 2\boldsymbol{\alpha}_{\mathbf{E},i}^{\mathsf{T}}\,\dot{\mathbf{E}}_i - \boldsymbol{\alpha}_{\mathbf{g},i}^{\mathsf{T}}\,\dot{\boldsymbol{\mu}}_i)|_{t=t_{\mathrm{e},i}} \\ (\boldsymbol{\alpha}_{\mathbf{g},i}\,\ddot{\mathbf{q}}_i + \boldsymbol{\beta}_{\mathbf{g},i}\,\dot{\mathbf{q}}_i + \gamma_{\mathbf{g},i}\,\mathbf{q}_i + \mathbf{P}_{\mathbf{g},i}\,\boldsymbol{\varphi})|_{t=t_{\mathrm{e},i}} \\ (\boldsymbol{\alpha}_{\mathbf{g},i}\,\dddot{\mathbf{q}}_i + \boldsymbol{\beta}_{\mathbf{g},i}\,\ddot{\mathbf{q}}_i + \gamma_{\mathbf{g},i}\,\dot{\mathbf{q}}_i + \mathbf{P}_{\mathbf{g},i}\,\boldsymbol{\varphi})|_{t=t_{\mathrm{e},i}} \end{bmatrix} = \begin{bmatrix} \dot{\mathbf{q}}|_{t=t_{\mathrm{a},i-1}} \\ \mathbf{q}|_{t=t_{\mathrm{a},i-1}} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \tag{47}$$

As a next step, $\mathbf{E}_i$ and its derivatives are substituted into Eq. (47). Then the generalized coordinates $\mathbf{q}_i$, the multipliers $\boldsymbol{\mu}_i$ and their time derivatives are substituted from the solution $\mathbf{z}_i$ evaluated at the appropriate time instant $t = t_{\mathrm{s},i}$ or $t = t_{\mathrm{e},i}$. This way terms appear on the left side of Eq. (47) that linearly depend on the yet unknown constant $\mathbf{c}_i$. Collecting the coefficients of the terms depending on $\mathbf{c}_i$ and those that are independent of it, the matrix $\mathbf{A}_{\mathrm{bc},i}$ and the vector $\mathbf{b}_{\mathrm{bc},i}$ are formed. Finally, the boundary conditions in Eq. (47) are written in the form

$$\mathbf{A}_{\mathrm{bc},i}\,\mathbf{c}_i = \mathbf{b}_{\mathrm{bc},i} \tag{48}$$

This linear equation is solved for the constant $\mathbf{c}_i$. This way, the solution of the linear optimization problem is derived in Eq. (44).

The optimal motion of the system is determined by taking the generalized coordinates $\mathbf{q}_i$ from $\mathbf{z}_i$. The last step of the calculations is to determine the actuator forces $\boldsymbol{\tau}$ by substituting the calculated $\ddot{\mathbf{q}}_i$ values into Eq. (5). It is also possible to add

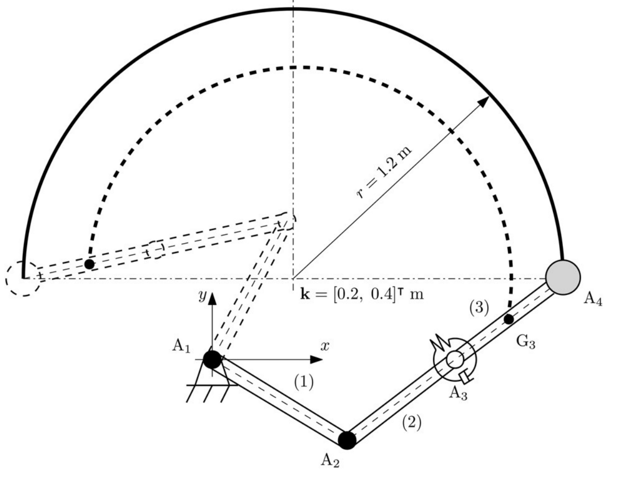

Fig. 3 Trajectory tracking task of a 3DoF planar manipulator

Table 1 Physical properties of the 3DoF planar manipulator

i	1	2	3	4
m_i (kg)	8	6	6	10
J_i (kgm^2)	0.45	0.2	0.2	—
l_i (m)	0.8	0.6	0.6	—

proportional and derivative feedback based on the computed optimal motion to overcome issues due to uncertainties of the system.

3 Numerical Case Studies

Simulations were carried out to test the optimization-based predictive control algorithm detailed in Sec. 2. We took the same 3DoF planar manipulator with the same task as the authors of Ref. [9], and another more practical passive joint manipulator which is a simple model of a flexible serial RR robot consisting of two flexible arms connected by two revolute joints.

In all of the simulations in this paper the numerical integrations of the nonlinear equations of motion (1) were carried out with the fourth-order Runge–Kutta method using a time-step of $\Delta t_{RK4} = 0.001$ s. The reason for not using any automatic step-size ODE solver is that in a real robotic application the clock of the controller has a particular frequency, which is constant. The actuator forces $\boldsymbol{\tau}$ are computed in each time-step according to Eq. (5) with the corresponding optimal motion of $\mathbf{q}_i$.

3.1 The Control Problem of the 3DoF Manipulator. As it can be seen in Fig. 3, the manipulator is modeled with three rigid beams connected to each other with frictionless revolute joints A$_1$, A$_2$, and A$_3$. The length, the mass, and the second moment of inertia of each beam is denoted by l_i, m_i, and J_i $(i = 1,\ 2,\ 3)$, respectively. These values are shown in Table 1. The end-effector is modeled by an additional mass particle of $m_4 = 10$ kg in the point A$_4$. The task of the manipulator is to move along the point A$_4$ on the prescribed trajectory in the $(x,\ y)$-plane.

The system is equipped with two actuators whose torques are collected in the vector $\boldsymbol{\tau} = [\tau_1,\ \tau_2]^\mathsf{T}$ acting in the joints A$_1$ and A$_2$. The joint A$_3$ is a passive joint where there is no actuator. Instead, there is a torsional spring with stiffness $k = 10$ Nm/rad and a torsional damping element with coefficient $c = 2$ Nms/rad.

To describe the motion, we use the independent generalized coordinates $\mathbf{q} = [\boldsymbol{\varphi}_1,\ \boldsymbol{\varphi}_2,\ \boldsymbol{\varphi}_3]^\mathsf{T}$ with $\boldsymbol{\varphi}_i$ denoting the angle of the ith beam measured from the axis x. The end-effector coordinates are $\mathbf{r}(\mathbf{q}) = [x_{\mathrm{A}_4}(\mathbf{q}),\ y_{\mathrm{A}_4}(\mathbf{q})]^\mathsf{T}$. The prescribed trajectory $\mathbf{r}_\mathrm{d}(t)$ is

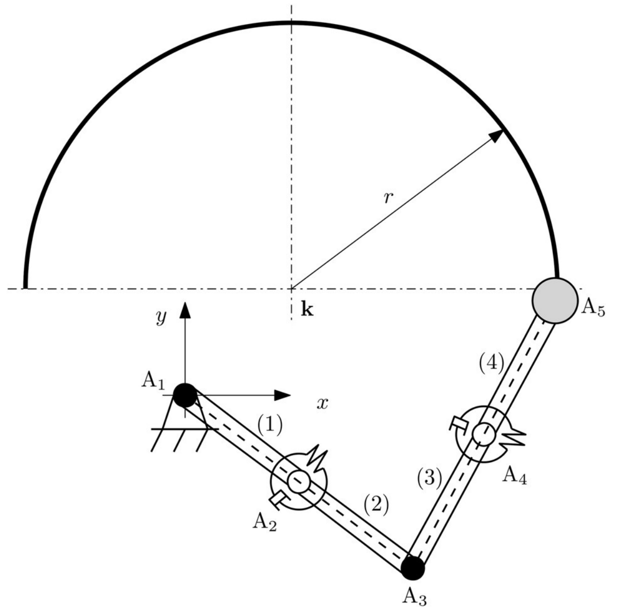

Fig. 4 Trajectory tracking task of a 4DoF planar manipulator

Table 2 Physical properties of the 4DoF planar manipulator

i	1	2	3	4	5
m_i (kg)	2	2	1	1	2
J_i (kgm^2)	0.0417	0.0417	0.0208	0.0208	—
l_i (m)	0.5	0.5	0.5	0.5	—
k_i (Nm/rad)	—	16	—	8	—
c_i (Nms/rad)	—	4	—	2	—

defined with the same ninth order polynomial functions as in Ref. [9], generating a half circle shaped trajectory with radius $r = 1.2$ m and center point $\mathbf{k} = [0.2,\ 0.4]$ m.

3.2 The Control Problem of the 4DoF Manipulator. This manipulator shown in Fig. 4 is modeled similarly to the previous 3DoF one. This time four rigid beams are connected with the joints A$_1$ to A$_4$. The values of the physical parameters are shown in Table 2. This time the end-effector in the point A$_5$ has a mass m_5. The joints A$_2$ and A$_4$ are passive, the corresponding torsional spring stiffness's k_i, and the torsional damping coefficients c_i are also listed in Table 2.

Again ϕ_i is defined as the angle of the ith beam measured from the axis x, and the generalized coordinates are $\mathbf{q} = [\boldsymbol{\varphi}_1,\ \boldsymbol{\varphi}_2,\ \boldsymbol{\varphi}_3,\ \boldsymbol{\varphi}_4]^\mathsf{T}$. The system has two actuators $\boldsymbol{\tau} = [\tau_1,\ \tau_3]^\mathsf{T}$ placed in the joints A$_1$ and A$_3$. The desired trajectory of the end-effector is prescribed the same way as previously in the case of the 3DoF robot.

3.3 Simulation Results. The following results are accomplished with the intuitively chosen controller parameter set shown in Table 3. A study on the effect of these control parameters is presented later in Sec. 3.5. For the approximations in Eqs. (22) and (23), the following polynomial function system was used:

Table 3 Parameters of the controller

w_r (1/m)	w_τ (1/Nm)	Δt_a (s)	Δt_e (s)
2000	1	0.005	0.4

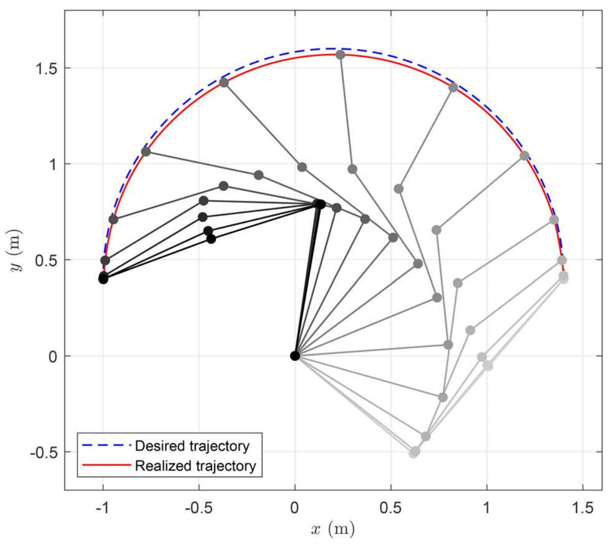

Fig. 5 Realized motion of the 3DoF robot applying the predictive control method

$$\varphi = [1, \quad t, \quad t^2]^{\mathsf{T}} \tag{49}$$

which was proven to be sufficient in our particular case study. Including higher-order term does not provide more accurate trajectory tracking.

The simulations resulted in the motion shown with a stroboscopic view in Figs. 5 and 6, where the desired path is depicted by a dashed line and the realized path is shown by a continuous line. Figures 7 and 8 show the trajectory tracking error along with the time propagation. The actuator torques are shown in Figs. 9 and 10.

3.4 Discussion of the Results. This simulation result is compared to the previous case study carried out by Blajer and Kołodziejczyk [9]. In that paper, the authors reported that the servoconstraint based controller makes the system unstable. To overcome this problem they intuitively modified the servoconstraint and controlled the G_3 center of mass of the third arm (shown in Fig. 3.) instead of its endpoint, which finally resulted in a stable

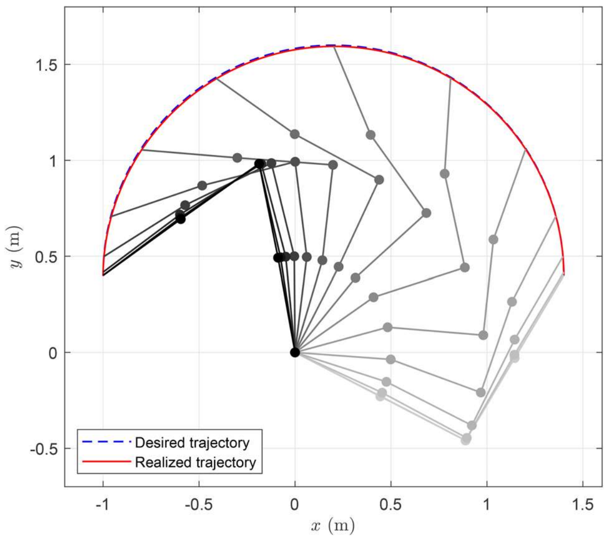

Fig. 6 Realized motion of the 4DoF robot applying the predictive control method

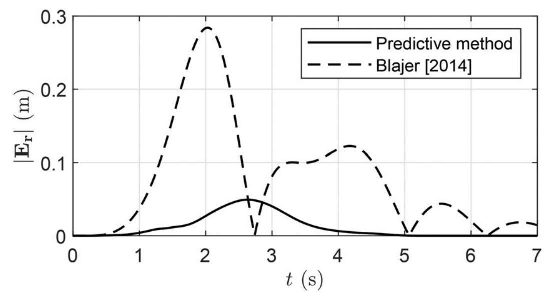

Fig. 7 Trajectory tracking errors of the 3DoF robot

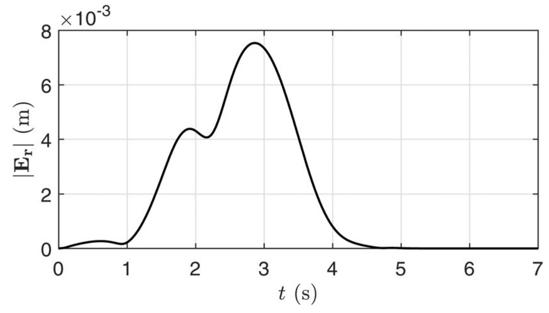

Fig. 8 Trajectory tracking errors of the 4DoF robot

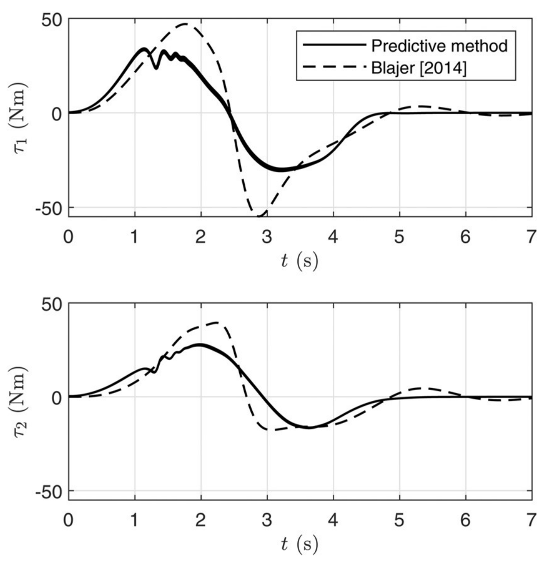

Fig. 9 Calculated actuator torques for the 3DoF robot during the maneuver

motion but a tradeoff appeared regarding the precision. In the Figs. 7 and 9, the results of both solutions are benchmarked.

The trajectory tracking errors $|\mathbf{E_r}|$ (i.e., the Euclidean-norm of $\mathbf{E_r}$ in Eq. (6)) of both maneuvers are shown in Fig. 7. The related performance measures are collected in Table 4. As it can be seen from Table 4, our present predictive method achieved better accuracy compared to the modified servoconstraint based inverse dynamics solution presented in Ref. [9]. The predictive method achieved an RMS value of 17.36% and a maximum error value of 20.84% compared to the results of the modified inverse dynamics. In our approach, the tracking error decays quickly after the final position is reached.

The actuator torques τ_1 and τ_2 are plotted in Fig. 9. Both simulations resulted in bounded torques. After the maneuver is

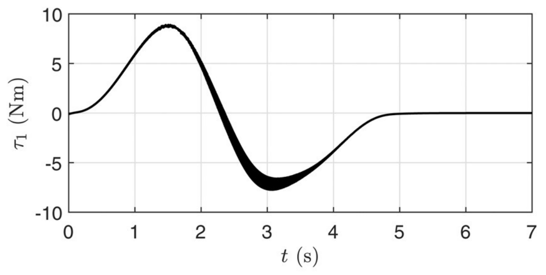
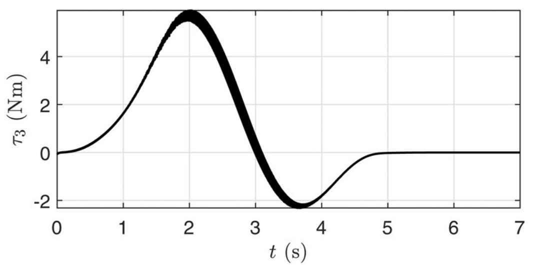

Fig. 10 Calculated actuator torques of the 4DoF robot during the maneuver

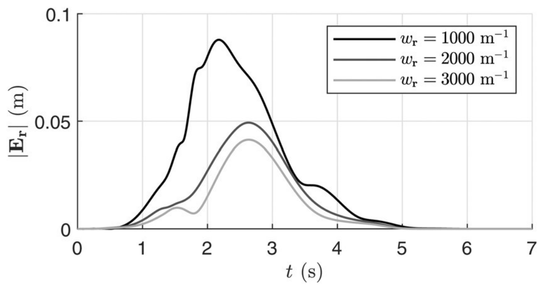

Fig. 11 Trajectory tracking errors for different w_r values (simulation set I)

Table 4 Trajectory tracking error results of the 3DoF robot

Solution	RMS (m)	Maximum error (m)
Predictive method	0.0191	0.0493
Modified inverse dynamics [9]	0.0916	0.2841

finished, the actuator torques are decaying and the final configuration is stabilized. In the case of the predictive method, smaller torque was enough, however, small high-frequency oscillations appeared. Furthermore, slow torque oscillation is present in the case of the controller in Ref. [9], while the torques settle down immediately after arriving in the final position in our approach.

In the case of the 4DoF robot, Fig. 8 shows the trajectory tracking error while the actuator torques τ_1 and τ_3 are shown in Fig. 10. The error has the values of $\mathrm{RMS}(|\mathbf{E}_\mathbf{r}|) = 0.0031\,\mathrm{m}$ and $\max(|\mathbf{E}_\mathbf{r}|) = 0.0075\,\mathrm{m}$ which is acceptable compared to the dimensions of the manipulator.

In a real application, computational demand is a key property that must be addressed therefore the required calculation times have been measured and presented in Table 5 in the case of both systems. We measured the time required to solve the linear optimization problem t_opt and the time spent on the whole simulation t_sim which includes the solution of all optimization problems, determination of the actuator torques, and the integration of the equation of motion too. For each the maximum values from 10 simulation runs and the mean values $\bar{t}_\mathrm{opt}$ and $\bar{t}_\mathrm{sim}$ of these simulation runs are reported. All computations were made in MATLAB environment on the predictive controller (PC) with an Intel Core i5-8250 U CPU. As it can be seen the simulation takes less time than the length of the simulated time interval: $t_\mathrm{sim} < \Delta t_\mathrm{sim}$. The condition $t_\mathrm{opt} < \Delta t_\mathrm{a}$ holds, which is essential for real-time computing. This means that the calculations required to control the system take less time than the usage of the results. The values of t_opt show that the controller is fast enough to achieve low time delays too.

3.5 Numerical Study on the Effects of the Controller Parameters. The predictive method presented in this paper has a function system $\boldsymbol{\varphi}$ and four arbitrarily chosen parameters $w_\mathbf{r}$, w_τ, Δt_a, and Δt_e. In the simulation in Sec. 3.3, these were selected via the trial and error method. Now we aim to give a brief sensitivity analysis of these parameters via numerical experiments done on the 3DoF robot.

It is possible to choose the elements of this function system $\boldsymbol{\varphi}$ either from polynomial or trigonometrical functions. It is important to have a satisfactory number of elements in $\boldsymbol{\varphi}$. In case the number of linearly independent functions is low, then the accuracy of the controller is limited. However, with too many interpolating functions, the effect of numerical errors is raising, which finally results in undesirable oscillations and an unstable controller.

The examined parameters are collected in the second column of Table 6. In each of the following three simulation sets I, II, and III, only one of the examined parameters were swept and the other parameters were fixed according to Table 3. The parameter ranges of the simulation sets are shown in Table 6. Both edges of these ranges shown in the third and seventh columns were determined by the value where stability loss is experienced first (i.e., these parameters yield an unstable controller).

Theoretically, out of the four parameters $w_\mathbf{r}$, w_τ, Δt_a, and Δt_e only three are independent from each other. As the goal of the method is to minimize the cost function J, it is trivial that only the ratio of the weighing constants $w_\mathbf{r}$ and w_τ effects the results. An

Table 5 Measured computational times of the simulations

Simulation	$\bar{t}_\mathrm{opt}$ (s)	$\max(t_\mathrm{opt})$ (s)	Δt_a (s)	$\bar{t}_\mathrm{sim}$ (s)	$\max(t_\mathrm{sim})$ (s)	Δt_sim (s)
3DoF manipulator	0.00091	0.00385	0.005	3.554	3.615	7
4DoF manipulator	0.00111	0.00340	0.005	4.937	4.981	7

Table 6 Modified parameters of the controller during the simulation sets

Set	Examined parameter	Unstable	First stable simulation	Increment	Last stable simulation	Unstable
I.	w_r (1/m)	600	700	100	3600	3700
II.	Δt_a (s)	—	0.01	0.01	0.15	0.16
III.	Δt_e (s)	0.31	0.32	0.01	0.45	0.46

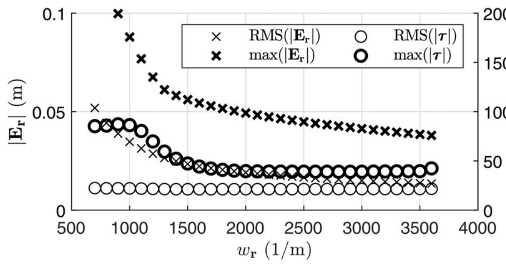

Fig. 12 Trajectory tracking errors: maximal and RMS values (simulation set I)

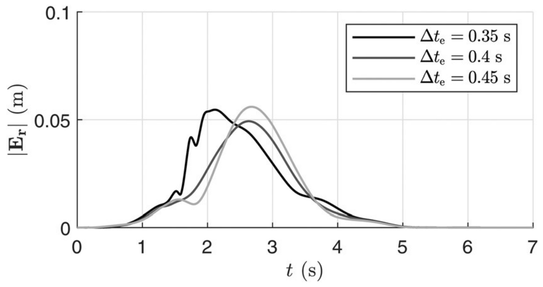

Fig. 13 Trajectory tracking errors for different Δt_e values (simulation set II)

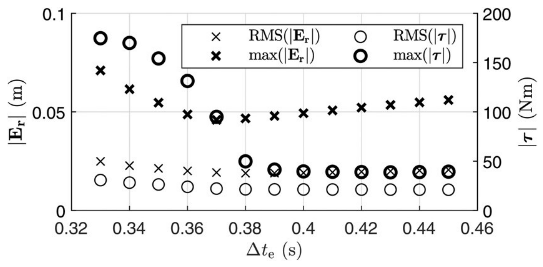

Fig. 14 Trajectory tracking errors: maximal and RMS values (simulation set II)

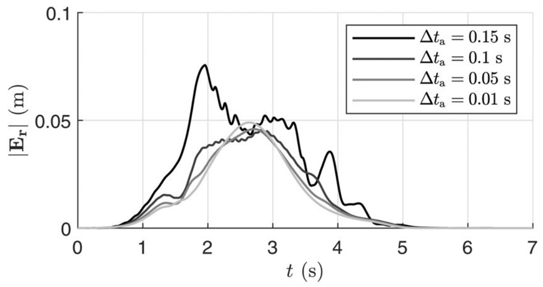

Fig. 15 Trajectory tracking errors for different Δt_a values (simulation set III)

optimization problem with the same ratio of w_r/w_τ results in the same $\mathbf{q}$ optimal motion (neglecting the effects of numerical errors). Therefore, in this paper, the value of $w_\tau = 1\,(\mathrm{Nm})^{-1}$ was chosen in every simulation, and only the sensitivity for w_r is examined.

Simulation Set I. Figure 11 shows the trajectory tracking error for different w_r values. Furthermore Figure 12 shows $\max(|\mathbf{E}_r|)$, $\mathrm{RMS}(|\mathbf{E}_r|)$, $\max(|\boldsymbol{\tau}|)$, and $\mathrm{RMS}(|\boldsymbol{\tau}|)$. It is expected

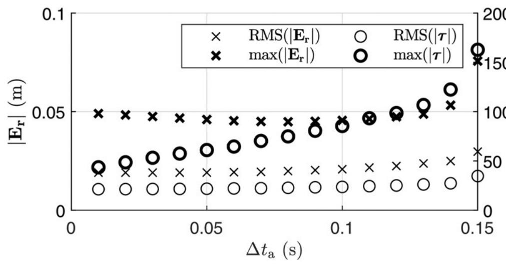

Fig. 16 Trajectory tracking errors: maximal and RMS values (simulation set III)

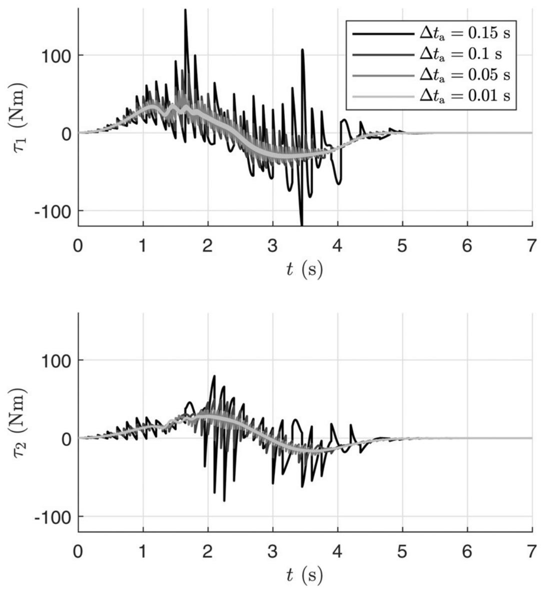

Fig. 17 Actuator torques for different Δt_a values (simulation set III)

that a higher value of w_r results in a smaller trajectory tracking error. The simulation set I. verifies this. The maximum torque is minimal for the value about $w_r = 2900\,\mathrm{m}^{-1}$. Hence, there is an optimal choice of w_r.

Simulation Set II. Figures 13 and 14 show the effect of the parameter Δt_e. Although the maximum of the error varies a bit, the RMS stays almost constant on the whole stable parameter interval. Similarly, the torques are not affected too much. The safe choice is in the middle of the stable region.

Simulation Set III. The effect of the parameter Δt_a is shown in Figs. 15–17. After an initial stagnant interval, $\mathrm{RMS}(|\mathbf{E}_r|)$ is rising slowly in Fig. 16. Therefore from the viewpoint of accuracy, lower Δt_a values are better. This is due to the fact that the controller algorithm uses the linearized model of the system around a certain configuration. This means that the results are acceptable only in a small neighborhood of this configuration which is ensured by choosing a small Δt_a value. The selection of Δt_a influences the actuator torques $\boldsymbol{\tau}$ too, which is shown in Figs. 16 and 17. As it was noted in Sec. 2.7 the validity of the linearization is not ensured by the boundary condition equations. This results in discontinuous actuator forces. Figure 17 shows that higher values of Δt_a results in greater jumps in the actuator forces. Therefore, small Δt_a values are beneficial.

On the other hand, a lower Δt_a parameter value means that the linear optimization problems are solved more frequently which results in higher computational demand. This means that the selection of the parameter Δt_a is a compromise between accuracy and computational demand.

4 Conclusion

In this paper, we proposed a predictive approach that is specially devoted to the trajectory tracking control of underactuated systems either with stable or unstable zero dynamics. First, we presented the underlying idea of the method, which is rendering an appropriately defined cost function to a minimal value. This cost function, which incorporates the linear combination of the trajectory tracking error and the actuator effort, is minimized assuming a dynamically consistent motion of the underactuated system. For determining the optimal motion, the calculus of variations is applied. We briefly introduced the equations arising from this optimization problem and proposed an approach to obtain an approximate solution. The algorithm is constructed in a sequence of steps that is possible to implement in a controller of a physical robot.

A benchmark problem of a planar manipulator from a previous publication [9] was presented. In that paper, the authors showed that the inverse dynamics control of that specific mechanism results in an unstable motion. They solved the problem by modifying the task of the manipulator intuitively. Regardless, whether the prescribed task is feasible or not in the dynamical sense, the proposed method provides a stable solution resulting in the minimal cost function. Therefore it is possible to apply it to a wider range of manipulators. Another advantage of the proposed predictive approach is that it resulted in a more accurate trajectory tracking with lower control input in the case of the presented benchmark problem. We note that there are approaches, e.g., Ref. [20] capable of higher accuracy compared to our solution, but these methods are not possible to apply in real-time.

The inverse dynamical simulations in Refs. [9] and [35–37] assume that the number of actuators and outputs are the same. As a benefit, this restriction does not apply to our predictive method. This means that one is able to use the framework in the case of more general systems. For example, it is possible to solve control problems of manipulators that have redundant actuator set without defining additional secondary tasks. Over-constrained tasks are possible to solve with the same algorithm too. In case the system has a prescribed task that is impossible to realize the algorithm searches for the best solution according to the cost function. The later one opens the possibility to apply this method in force recalculation problems like [38] and [39]. Also, the algorithm could be generalized for inverse dynamics problems related to biomechanics when several muscles (actuators) are related to each DoF [40,41].

Further problems are still under investigation as the approach could be generalized to manage underactuated and fully actuated systems by the same algorithm, or systems described by redundant coordinates, such as natural coordinates [42].

Proper stability analysis should be done in future work to ensure the stable behavior of the controller since currently the parameters of the controller are chosen using a trial and error method. This stability analysis is beneficial as other stable parameter regions could be discovered.

Acknowledgment

The research reported in this paper and carried out at BME has been supported by the NRDI Fund (TKP2020 IES, Grant No. BME-IE-BIO and TKP2020 NC, Grant No. BME-NC) based on the charter of bolster issued by the NRDI Office under the auspices of the Ministry for Innovation and Technology, and by the Hungarian National Research, Development and Innovation Office (NKFI-FK18 128636).

Funding Data

- National Research, Development and Innovation Office (TKP2020 IES, Grant No. BME-IE-BIO and TKP2020 NC, Grant No. BME-NC and NKFI-FK18 128636; Funder ID: 10.13039/501100018818).

References

[1] Seifried, R., 2014, "Dynamics of Underactuated Multibody Systems: Modeling, Control and Optimal Design," Solid Mech. Appl., **205**, pp. 9–54.
[2] El-Badawy, A. A., and Shehata, M. M. G., 2015, "Anti-Sway Control of a Tower Crane Using Inverse Dynamics," ICET 2014—Second International Conference on Engineering and Technology, Institute of Electrical and Electronics Engineers, Cairo, Egypt, Apr. 19–20.
[3] Raffo, G. V., Ortega, M. G., and Rubio, F. R., 2011, "Path Tracking of a UAV Via an Underactuated H Control Strategy," Eur. J. Control, **17**(2), pp. 194–213.
[4] Tedrake, R., 2009, "Underactuated Robotics: Learning, Planning, and Control for Efficient and Agile Machines Course Notes for MIT 6.832," Massachusetts Institute of Technology, MS.
[5] Collins, S. H., Wisse, M., and Ruina, A., 2001, "A Three-Dimensional Passive-Dynamic Walking Robot With Two Legs and Knees," Int. J. Rob. Res., **20**(7), pp. 607–615.
[6] McHenry, H. M., 2009, *Evolution: The First Four Billion Years*, Harvard University Press, Human Evolution, Cambridge, MS, pp. 256–280.
[7] Kovács, L., Kövecses, J., Zelei, A., Bencsik, L., and Stépán, G., 2011, "Servo-Constraint Based Computed Torque Control of Underactuated Mechanical Systems," ASME Paper No. DETC2011-48533.
[8] Peng, W., Lin, Z., and Su, J., 2009, "Computed Torque Control-Based Composite Nonlinear Feedback Controller for Robot Manipulators With Bounded Torques," IET Control Theory Appl., **3**(6), pp. 701–711.
[9] Blajer, W., and Kołodziejczyk, K., 2014, "A Case Study of Inverse Dynamics Control of Manipulators With Passive Joints," J. Theor. Appl. Mech., **52**(3), pp. 793–801.
[10] Otto, S., and Seifried, R., 2018, "Real-Time Trajectory Control of an Overhead Crane Using Servo-Constraints," Multibody Syst. Dyn., **42**(1), pp. 1–17.
[11] Berger, T., Otto, S., Reis, T., and Seifried, R., 2019, "Combined Open-Loop and Funnel Control for Underactuated Multibody Systems," Nonlinear Dyn., **95**(3), pp. 1977–1998.
[12] Isidori, A., 1995, *Nonlinear Control Systems* (Communications and Control Engineering), Springer London, London.
[13] Spong, M., 1994, "Partial Feedback Linearization of Underactuated Mechanical Systems," Proceedings of IROS'94, Munich, Germany, Sept. 12–16, pp. 314–321.
[14] Slotine, J. J. E., and Li, W., 1995, *Applied Nonlinear Control*, Prentice Hall, Englewood Cliffs, NJ.
[15] Schnelle, F., and Eberhard, P., 2016, "Adaptive Model Predictive Control Design for Underactuated Multibody Systems With Uncertain Parameters," ROMANSY 21—Robot Design, Dynamics and Control: Proceedings of the 21st CISM-IFToMM Symposium, Udine, Italy, June 20–23, pp. 145–152.
[16] Bencsik, L., Kovács, L. L., and Zelei, A., 2017, "Stabilization of Internal Dynamics of Underactuated Systems by Periodic Servo-Constraints," Int. J. Struct. Stab. Dyn., **17**(05), p. 1740004.
[17] Seifried, R., 2009, "Optimization of the Internal Dynamics of Underactuated Robots," PAMM, **9**(1), pp. 625–626.
[18] Catlin, D. E., 1989, *The Linear Quadratic Tracking Problem*, Springer, New York, pp. 164–187.
[19] Lewis, F. L., Vrabie, D. L., and Syrmos, V. L., 2012, *Optimal Control*, Wiley, Hoboken, NJ.
[20] Bastos, G., and Brüls, O., 2020, "Analysis of Open-Loop Control Design and Parallel Computation for Underactuated Manipulators," Acta Mech., **231**(6), pp. 2439–2456.
[21] Hansen, A., Li, Y., and Hedrick, J. K., 2017, "Invariant Sliding Domains for Constrained Linear Receding Horizon Tracking Control," IFAC J. Syst. Control, **2**, pp. 12–17.
[22] Borrelli, F., Bemporad, A., and Morari, M., 2017, *Predictive Control for Linear and Hybrid Systems*, Cambridge University Press, Cambridge, UK.
[23] Bencsik, L., Bodor, B., and Kovács, L., 2016, "Predictive Trajectory Tracking of Under-Actuated Systems," Proceedings of the Fourth Joint International Conference on Multibody System Dynamics, Montreal, Canada, May 29–June 2.
[24] Bodor, B., and Bencsik, L., 2017, "Predictive Control of Robot Manipulators With Flexible Joints," Proceedings of the Ninth European Nonlinear Dynamics Conference, Budapest, Hungary, June 25–30.
[25] Zelei, A., and Stépán, G., 2012, "Case Studies for Computed Torque Control of Constrained Underactuated Systems," Period. Polytech. Mech. Eng., **56**(1), pp. 73–80.
[26] Spong, M. W., 1998, "Underactuated Mechanical Systems," Control Probl. Rob. Autom., **230**, pp. 135–150.
[27] Kirgetov, V. I., 1967, "The Motion of Controlled Mechanical Systems With Prescribed Constraints (Servoconstraints)," J. Appl. Math. Mech., **31**(3), pp. 465–477.
[28] Blajer, W., Seifried, R., and Kołodziejczyk, K., 2015, "Servo-Constraint Realization for Underactuated Mechanical Systems," Archive Appl. Mech., **85**(9–10), pp. 1191–1207.
[29] Hansen, A., and Hedrick, J. K., 2015, "Receding Horizon Sliding Control for Linear and Nonlinear Systems," Proceedings of the American Control Conference, Institute of Electrical and Electronics Engineers, Chicago, IL, July 1–3, pp. 1629–1634.

[30] Hollerbach, J. M., and Suh, K. C., 1987, "Redundancy Resolution of Manipulators Through Torque Optimization," IEEE J. Rob. Autom., **3**(4), pp. 308–316.

[31] Nakamura, Y., 1991, *Advanced Robotics: Redundancy and Optimization*, Addison-Wesley, Reading, MA.

[32] Zelei, A., Bencsik, L., Kovács, L. L., and Stépán, G., 2012, "Redundancy Resolution of the Underactuated Manipulator ACROBOTER," Romansy 19—Robot Design, Dynamics and Control: Proceedings of the 19th CISM-IFtomm Symposium, Paris, France, June 12–15, p. 17.

[33] Weinstock, R., 1952, *Calculus of Variations*, McGraw-Hill Book Company, New York.

[34] Bartels, R. H., and Stewart, G. W., 1972, "Solution of the Matrix Equation $AX + XB = C$ [F4]," Commun. ACM, **15**(9), pp. 820–826.

[35] Blajer, W., and Kołodziejczyk, K., 2007, "Control of Underactuated Mechanical Systems With Servo-Constraints," Nonlinear Dyn., **50**(4), pp. 781–791.

[36] Blajer, W., and Kołodziejczyk, K., 2008, "Modeling of Underactuated Mechanical Systems in Partly Specified Motion," J. Theor. Appl. Mech., **46**(2), pp. 383–394.

[37] Zelei, A., Kovács, L. L., and Stépán, G., 2011, "Computed Torque Control of an Under-Actuated Service Robot Platform Modeled by Natural Coordinates," Commun. Nonlinear Sci. Numer. Simul., **16**(5), pp. 2205–2217.

[38] Bencsik, L., Bodor, B., and Insperger, T., 2017, "Reconstruction of Motor Force During Stick Balancing," Proceedings of DSTA 2017: Mathematical and Numerical Aspects of Dynamical System Analysis, Lodz, Poland, Dec. 11–13, pp. 57–64.

[39] Bodor, B., Bencsik, L., and Insperger, T., 2019, "Control Force Recalculation for Balancing Problems," Int. J. Struct. Stab. Dyn., **19**(05), p. 1941010.

[40] Dziewiecki, K., Mazur, Z., and Blajer, W., 2013, "Assessment of External and Internal Loads in the Triple Jump Via Inverse Dynamics Simulation," Biol. Sport, **30**(2), pp. 103–109.

[41] Dziewiecki, K., Blajer, W., Mazur, Z., and Czaplicki, A., 2014, "Modeling and Computational Issues in the Inverse Dynamics Simulation of Triple Jump," Multibody Syst. Dyn., **32**(3), pp. 299–316.

[42] García de Jalón, J., and Bayo, E., 1994, *Kinematic and Dynamic Simulation of Multibody Systems: The Real-Time Challenge*, Springer-Verlag, New York.