Multibody System Dynamics
https://doi.org/10.1007/s11044-025-10113-w

RESEARCH

Adjoint gradient computation for an extremal value of a system output

Philipp Zallinger[1,2] · Daniel Lichtenecker[3] · Philipp Eichmeir[1,4] · Wolfgang Steiner[1] · Karin Nachbagauer[1,5]

Received: 30 April 2025 / Accepted: 30 September 2025

Abstract
Extremal values of a system output pose major issues in various disciplines, e.g., the maximum velocity in human-robot collaboration results in high contact forces in the event of a collision, or force and stress peaks cause faster crack growth or fatigue of components. Reducing these extremal values implies a reduction in the risks to humans and an increase in the durability of the components. Therefore, the present paper focuses on minimizing an extremal value of a system output of dynamical system, whereby a gradient-based solution strategy based on the adjoint method is proposed. Since several local extremal values can occur in the time evolution of the system output, it is necessary to apply multi-objective optimization, whereby in particular the largest value is to be minimized. One promising approach in this regard is found in the goal attainment method, which is implemented in the MATLAB function `fminimax`, or alternatively, the so-called minimax problem can be investigated in a smoothed objective open for any nonlinear programming software package. In the scope of these minimax problems, the maximum reaction force of a one-mass oscillator and the maximum velocity of the tool center point of a two-axis robot during a rest-to-rest maneuver are minimized efficiently using the proposed adjoint gradient.

Keywords Adjoint gradient method · Multi-objective optimization · Minimax problems · Nonlinear programming

1 Introduction

Nowadays, guaranteeing the functionality of multibody systems is no longer sufficient, but rather its optimal design or performance is crucial. Probably one of the most popular tasks in production or robotics is to reduce the cycle time of a work step to a minimum in order to reduce costs and energy. With the increasing integration of humans into production procedures in the form of human-centered automation or human-robot collaboration (HRC), other important aspects are emerging. As stated in [17], exceeding high velocities and unpredictable motions of the robot risk adverse effects on human safety and health. Especially the maximum velocities pose a high conflict potential for collaboration, since in the event of a collision the high momentum results in high contact forces, according to the evaluation of the resulting forces for different velocities in [30]. To avoid such situations from the beginning, a common approach is speed and distance monitoring, which aims to adjust the speed

Extended author information available on the last page of the article of the robot so that it stops in time to avoid collisions. To circumvent possible collisions, but also to improve productivity during HRC, dynamic zones are arranged around the robot in [19, 31], which must not collide with the human. To be more precise, in [19], the speed of the robot is maximized and these zones are continuously adjusted to ensure collision-free stopping of the robot if necessary. In [31], the stop time of the robot is minimized by online scaling of dynamic safety zones. In [35], the trajectory of the robot manipulator is optimized to achieve a constant absolute speed, which is required in some production steps, e.g., for surface processing operations or for endurance tests. It is worth mentioning that certain waypoints can be specified for the manipulator, but collision avoidance for obstacles is also integrated there.

In contrast to the examples mentioned above, not all problems can be formulated as single-objective optimization, as they involve several objectives to fulfill simultaneously. Furthermore, objective functions in multidimensional, multi-objective optimization problems of complex systems are usually conflicting or contradictory. However, the challenge is to consider multiple optimization objectives simultaneously in order to achieve the optimization goal. To determine so-called Pareto-optimal solutions, various solution strategies have been developed in a wide range of disciplines since the very beginnings of Edgeworth [7] and Pareto [28]. Essentially, a solution is said to be Pareto-optimal for a multi-objective problem if all other solutions have a higher value for at least one of the objective functions or have the same value for all the objective functions. As described in [4, 24], Pareto-optimal solutions are often obtained by either formulating appropriate single-objective optimization problems, e.g., weighted sum method or epsilon-constraint method, or by applying evolutionary algorithms. A review article with almost 200 references of multi-objective optimization methods for mechanical engineering applications shows various approaches and discusses their advantages and disadvantages in different scenarios [29].

Unlike Pareto-optimal solutions, minimax problems involve finding the solution which minimizes the worst-case scenario, even though individual objectives may deteriorate as a result. Solving minimax problems and deriving necessary optimality conditions have been discussed widely. Optimal control problems with maximum functional are discussed in [23] and necessary conditions are derived, giving guidance not only for numerical procedures, but also for interpretation of important features of the problem itself. In [2], a derivation is shown for minimizing the maximum cost by an approximation technique in order to solve the optimization problem by a dynamic programming principle. Furthermore, the minimax problem of linear dynamic problems is discussed in [14] as well. There, a numerical method for convex minimax optimal control problems is proposed through the definition of a suitable discrete time approximation based on a directional derivative of the cost functional deriving optimality conditions for descent methods. A recent work [25] presented various reformulations of optimal control problems with maximum cost with state or mixed constraints suitable to direct methods as well. Moreover, considering unconstrained problems, an approximation scheme generated by a sequence of optimal control solutions has been presented therein as well.

A technique for finding minimax solutions to multi-criteria optimization problems has been presented in [6] by demonstrating a multi-criteria version of a routing problem. A further promising approach for multi-criteria optimization is the goal attainment method, which originates from Gembicki's PhD thesis [12, 13] and presents an integrated, multi-objective optimization based on a vector index approach. This multi-objective optimization strategy is implemented in MATLAB's functions `fminimax` and `fgoalattain`, following the formulation in [32]. Since this toolbox is widely used in research and industry, the present paper utilizes this workaround for the minimization of an extremal value of a system output and presents an adjoint gradient approach which can enhance the efficiency of MATLAB's `fminimax` gradient-based optimization performance. The `fminimax` function implementation is designed to minimize the maximum value of a set of objective functions while allowing for general constraints, as it is necessary, e.g., for the aforementioned problems in HRC problems. In order to address these challenges, such as the reduction of the maximum velocity in real-time HRC applications, a gradient-based optimization strategy with time-efficient gradient computation is crucial. The `fminimax` function allows the user to provide a gradient for the optimization process and, here, the present paper shows an analytical adjoint gradient for this user-prescribed option. To be more precise, the scientific contribution of the present paper is the analytical derivation of such an efficient adjoint gradient for the minimax problem, which can be utilized by any gradient-based optimization method.

The adjoint method proposed in previous works [8, 10, 21] is extended here with a particular focus on minimizing the extremal values of a system output. The approach is integrated into the multi-objective optimization provided by the `fminimax` function in MATLAB. Moreover, an alternative solution strategy for these minimax problems is presented, in which an equivalent, but smoothed objective is formulated and the according adjoint gradient approach is open for any nonlinear programming (NLP) software package. The applicability of the proposed adjoint gradient computation for extremal value minimization is demonstrated in an academic and a multibody system example with prescribed final constraints. In particular, the maximum reaction force of a one-mass oscillator and the maximum velocity of the tool center point (TCP) of a two-axis robot are minimized.

2 Adjoint gradient approach for discrete control parameterization

In this section, a nonlinear dynamical system under final constraints is considered as an optimal control problem and an adjoint gradient approach for discrete control parameterization is derived according to the results in a prework [22, Sect. 3.2]. The behavior of a nonlinear dynamical system can be described by the state equations

$$\dot{\mathbf{x}}(t) = \mathbf{f}\left(\mathbf{x}(t), \mathbf{u}(t)\right) \quad \text{with} \quad \mathbf{x}\left(t_0\right) = \mathbf{x}_0, \tag{1}$$

in which $\mathbf{x}(t) \in \mathbb{R}^n$ is the state vector in minimal coordinate formulation and $\mathbf{u}(t) \in \mathbb{R}^m$ is the vector of control inputs. Moreover, initial conditions for the states are prescribed by $\mathbf{x}_0$.

In the context of a classical optimal control problem, the goal is to find a control $\mathbf{u}(t)$ that minimizes the cost functional

$$J = \int_{t_0}^{t_\mathrm{F}} h\left(\mathbf{x}, \mathbf{u}, t\right) \mathrm{d}t, \tag{2}$$

in which $h\left(\mathbf{x}(t), \mathbf{u}(t), t\right)$ is a user-defined system output, e.g., the root mean square error of the position of an intended trajectory of the TCP of a robot. Moreover, final constraints at the final time t_F can be given by

$$\boldsymbol{\phi}\left(\mathbf{x}\left(t_\mathrm{F}\right), t_\mathrm{F}\right) = \mathbf{0} \in \mathbb{R}^r. \tag{3}$$

In this paper, the optimal control problem formulated above is solved by a gradient-based optimization approach to avoid solving the underlying two-point boundary value problem, which is prone to a poor initial guess. The gradients are derived following the fundamental idea of calculus of variations by Kirk [16, Sect. 5.1]. Due to the fact that an efficient computation of these gradients is crucial for gradient-based optimizations, adjoint gradients are preferred due to computational burden. Adjoint gradients are derived here based on the fundamental work of Bryson and Ho [3] yielding an analytical gradient information in contrast to a gradient computation by numerical differentiation. Further works, e.g., [9, 21], demonstrate approaches for the simultaneous minimization of a cost functional J as well as the error regarding final constraints $\boldsymbol{\phi}$ using adjoint gradients. In contrast to the latter mentioned works, in which time-optimal control problems have been investigated, i.e., $h = 1$ and free final time t_F, here adjoint gradients are derived for problems with fixed final time t_F but general system output h.

As this paper is an extension of the latter works, a brief summary is given here to make the upcoming derivation easier to follow. In order to derive the analytical adjoint gradients for J and $\boldsymbol{\phi}$, the cost functional in Eq. (2) is extended by the state equations in Eq. (1) resulting in

$$\bar{J} = \int_{t_0}^{t_\mathrm{F}} \left[h\left(\mathbf{x}, \mathbf{u}, t\right) + \mathbf{p}^\mathsf{T}(\mathbf{f}(\mathbf{x}, \mathbf{u}) - \dot{\mathbf{x}})\right] \mathrm{d}t, \tag{4}$$

and the final constraints in Eq. (3) are extended by Eq. (1) analogously

$$\bar{\boldsymbol{\phi}} = \int_{t_0}^{t_\mathrm{F}} \mathbf{R}^\mathsf{T}\left(\mathbf{f}(\mathbf{x}, \mathbf{u}) - \dot{\mathbf{x}}\right) \mathrm{d}t + \boldsymbol{\phi}\left(\mathbf{x}\left(t_\mathrm{F}\right), t_\mathrm{F}\right). \tag{5}$$

Note that J in Eq. (2) and $\bar{J}$ in Eq. (4) are identical for the introduction of arbitrary so-called adjoint variables $\mathbf{p}(t) \in \mathbb{R}^n$ only in case the state equations are fulfilled. The same statement applies analogously to $\boldsymbol{\phi}$ in Eq. (3) and $\bar{\boldsymbol{\phi}}$ in Eq. (5) in which the adjoint variables $\mathbf{R}(t) \in \mathbb{R}^{n \times r}$ are introduced. The variation of $\bar{J}$ and $\bar{\boldsymbol{\phi}}$ resulting from an infinitesimal variation of the control $\mathbf{u}(t)$ is considered. This control variation leads to a variation of the states $\mathbf{x}(t)$ as well. The first order variation then reads

$$\delta\bar{J} = \int_{t_0}^{t_\mathrm{F}} \left[h_\mathbf{x}\delta\mathbf{x} + h_\mathbf{u}\delta\mathbf{u} + \mathbf{p}^\mathsf{T}\left(\mathbf{f}_\mathbf{x}\delta\mathbf{x} + \mathbf{f}_\mathbf{u}\delta\mathbf{u} - \delta\dot{\mathbf{x}}\right)\right] \mathrm{d}t, \tag{6}$$

$$\delta\bar{\boldsymbol{\phi}} = \int_{t_0}^{t_\mathrm{F}} \mathbf{R}^\mathsf{T}\left(\mathbf{f}_\mathbf{x}\delta\mathbf{x} + \mathbf{f}_\mathbf{u}\delta\mathbf{u} - \delta\dot{\mathbf{x}}\right) \mathrm{d}t + \boldsymbol{\phi}_\mathbf{x}\Big|_{t_\mathrm{F}} \delta\mathbf{x}_\mathrm{F}, \tag{7}$$

whereby $h_\mathbf{x}$, $h_\mathbf{u}$, $\mathbf{f}_\mathbf{x}$ and $\mathbf{f}_\mathbf{u}$ are partial derivatives with respect to $\mathbf{u}$ and $\mathbf{x}$. The δ-operator denotes the variation of the respective variables. Note that in the references [9, 21] also a variation of the final time t_F due to a free final time t_F occurs, but in this paper the final time t_F is fixed. After an integration by parts, the variations in Eq. (6) and (7) can be reduced to

$$\delta\bar{J} = \int_{t_0}^{t_\mathrm{F}} \left(h_\mathbf{u} + \mathbf{p}^\mathsf{T}\mathbf{f}_\mathbf{u}\right) \delta\mathbf{u}\, \mathrm{d}t, \tag{8}$$

$$\delta\bar{\boldsymbol{\phi}} = \int_{t_0}^{t_\mathrm{F}} \mathbf{R}^\mathsf{T}\mathbf{f}_\mathbf{u}\delta\mathbf{u}\, \mathrm{d}t, \tag{9}$$

if the adjoint variables fulfill the following systems of differential equations

$$\dot{\mathbf{p}}(t) = -h_\mathbf{x}^\mathsf{T} - \mathbf{f}_\mathbf{x}^\mathsf{T}\mathbf{p}(t) \quad \text{with} \quad \mathbf{p}(t_\mathrm{F}) = \mathbf{0}, \tag{10}$$

$$\dot{\mathbf{R}}(t) = -\mathbf{f}_{\mathbf{x}}^{\mathsf{T}}\mathbf{R}(t) \qquad \text{with} \quad \mathbf{R}(t_{\mathrm{F}}) = \boldsymbol{\phi}_{\mathbf{x}}^{\mathsf{T}}(\mathbf{x}(t_{\mathrm{F}}), t_{\mathrm{F}})\,. \tag{11}$$

These equations can be solved backwards in time using the final conditions at time t_{F}. From Eq. (8) and (9), the control variations which cause the largest decrease of $\bar{J}$ and $\bar{\boldsymbol{\phi}}$, i.e., the gradients of $\bar{J}$ and $\bar{\boldsymbol{\phi}}$, can be identified independently of the type of control discretization. In this paper, the control is discretized by applying a spline parameterization consisting of a set of k grid nodes $\bar{\mathbf{u}} \in \mathbb{R}^{m\cdot k}$ and a time dependent interpolation matrix $\mathbf{C}(t) \in \mathbb{R}^{m\times m\cdot k}$, following a direct transcription method, as presented in [21]. Now, the infinite-dimensional optimization task is transformed to a finite-dimensional problem defining the discrete control and the corresponding variation of the control,

$$\mathbf{u}(t) = \mathbf{C}(t)\,\bar{\mathbf{u}} \quad \text{and} \quad \delta\mathbf{u}(t) = \mathbf{C}(t)\,\delta\bar{\mathbf{u}}. \tag{12}$$

Substituting the variation from Eq. (12) into Eq. (8) and (9), the variation of $\bar{J}$ and $\bar{\boldsymbol{\phi}}$ reads

$$\delta\bar{J} = \int_{t_0}^{t_{\mathrm{F}}} \left(h_{\mathbf{u}} + \mathbf{p}^{\mathsf{T}}\mathbf{f}_{\mathbf{u}}\right)\mathbf{C}\,\mathrm{d}t\,\delta\bar{\mathbf{u}} \tag{13}$$

$$\delta\bar{\boldsymbol{\phi}} = \int_{t_0}^{t_{\mathrm{F}}} \mathbf{R}^{\mathsf{T}}\mathbf{f}_{\mathbf{u}}\mathbf{C}\,\mathrm{d}t\,\delta\bar{\mathbf{u}}. \tag{14}$$

In case of using a direct optimization method, the corresponding analytical adjoint gradients can be used instead of gradients computed by numerical differentiation due to computational time efficiency. The adjoint gradients can now be defined by

$$\nabla_{\bar{\mathbf{u}}}\bar{J}^{\mathsf{T}} = \int_{t_0}^{t_{\mathrm{F}}} \left(h_{\mathbf{u}} + \mathbf{p}^{\mathsf{T}}\mathbf{f}_{\mathbf{u}}\right)\mathbf{C}\,\mathrm{d}t, \tag{15}$$

$$\nabla_{\bar{\mathbf{u}}}\bar{\boldsymbol{\phi}}^{\mathsf{T}} = \int_{t_0}^{t_{\mathrm{F}}} \mathbf{R}^{\mathsf{T}}\mathbf{f}_{\mathbf{u}}\mathbf{C}\,\mathrm{d}t, \tag{16}$$

and can be directly provided in this form to a NLP software package, e.g., the MATLAB function `fmincon` [32] and Python packages `SciPy` [33] or `IPOPT` [34].

As an outlook, it is emphasized that the proposed adjoint approach is not restricted to minimal coordinate formulations. Available extensions, e.g., in [11, 26] allow its application to systems with redundant generalized coordinates and kinematic constraints, enabling the modeling of more complex systems and paving the way for automated multibody co-simulation procedures, as e.g., outlined in [27].

3 Adjoint gradient for a single extremal value

The derivation for the adjoint gradients in the previous section will now be extended for the optimal control problem minimizing a single extremal value of a considered system output $h(\mathbf{x}(t), \mathbf{u}(t), t)$. In practical applications, the limitation or reduction of extremal values is of particular interest. As an example of a multibody system, consider a rest-to-rest maneuver of a robot in which the maximum constraint forces or velocities must be reduced or minimized while still meeting the final constraints. Note that the derivation of the adjoint gradients for the final constraints as presented in Eq. (3) is not influenced by the particular optimization goal in this paper and therefore can be used later on in the optimization as presented above in Eq. (16). In this section, the cost functional in Eq. (2) will be formulated for such optimization goals and the corresponding gradient will be derived following the adjoint gradient approach from Sect. 2. For this purpose, the extremal value of a considered system output in time interval $t \in [t_0, t_\mathrm{F}]$ has to be first identified by

$$\dot{h}(\mathbf{x}(t), \mathbf{u}(t), t) \overset{!}{=} 0, \tag{17}$$

which results in

$$t_\mathrm{M} = \underset{t}{\operatorname{argmax}} \left(h(\mathbf{x}(t), \mathbf{u}(t), t) \right). \tag{18}$$

In this section, only one extremal value is assumed to exist in the given time interval $t \in [t_0, t_\mathrm{F}]$, i.e., exactly one time instance $t = t_\mathrm{M}$ fulfills Eq. (17). The cost functional J in Eq. (2) is formulated as

$$J = \max_{t} \left(h(\mathbf{x}(t), \mathbf{u}(t), t) \right) = h(\mathbf{x}(t_\mathrm{M}), \mathbf{u}(t_\mathrm{M}), t_\mathrm{M}). \tag{19}$$

Now, the goal is to find the control $\mathbf{u}(t)$ that minimizes this cost functional in order to decrease the extremal value of the system output, i.e., $h(\mathbf{x}(t_\mathrm{M}), \mathbf{u}(t_\mathrm{M}), t_\mathrm{M})$, by using an adjoint gradient approach. The cost functional in Eq. (19) is extended by the state equations analogously to Eq. (4) as follows

$$\hat{J} = h(\mathbf{x}(t_\mathrm{M}), \mathbf{u}(t_\mathrm{M}), t_\mathrm{M}) + \int_{t_0}^{t_\mathrm{M}} \mathbf{p}^\mathsf{T} (\mathbf{f}(\mathbf{x}, \mathbf{u}) - \dot{\mathbf{x}})\, \mathrm{d}t, \tag{20}$$

whereby $\mathbf{p}(t) \in \mathbb{R}^n$ are adjoint variables. Notice that the state equations (zero terms) are only added up to t_M here since there is no influence of the control after t_M on the cost functional in Eq. (19). A similar behavior of the adjoint variables has already been discussed in [20]. Analogously to the previous section, a variation of the control $\mathbf{u}(t)$ by $\delta\mathbf{u}(t)$ leads to the variation of the extended cost functional

$$\delta\hat{J} = h_\mathbf{x}\Big|_{t_\mathrm{M}} \delta\mathbf{x}_\mathrm{M} + h_\mathbf{u}\Big|_{t_\mathrm{M}} \delta\mathbf{u}_\mathrm{M} + h_t\Big|_{t_\mathrm{M}} \delta t_\mathrm{M} + \int_{t_0}^{t_\mathrm{M}} \left[\mathbf{p}^\mathsf{T} (\mathbf{f}_\mathbf{x}\delta\mathbf{x} + \mathbf{f}_\mathbf{u}\delta\mathbf{u} - \delta\dot{\mathbf{x}}) \right] \mathrm{d}t, \tag{21}$$

in which $h_\mathbf{x}|_{t_\mathrm{M}}$ is the partial derivative of h with respect to $\mathbf{x}(t)$ evaluated at time t_M. An integration by parts is applied to the terms containing the variation $\delta\dot{\mathbf{x}}$ for each integral as follows

$$\int_{t_0}^{t_\mathrm{M}} \mathbf{p}^\mathsf{T} \delta\dot{\mathbf{x}}\, \mathrm{d}t = \left(\mathbf{p}^\mathsf{T} \delta\mathbf{x} \right) \Big|_{\cancel{t_0}}^{t_\mathrm{M}} - \int_{t_0}^{t_\mathrm{M}} \dot{\mathbf{p}}^\mathsf{T} \delta\mathbf{x}\, \mathrm{d}t, \tag{22}$$

whereby $\delta\mathbf{x}(t_0)$ is omitted due to the defined initial state in Eq. (1). Consequently, the variation of the extended cost functional $\delta\hat{J}$ in Eq. (21) becomes

$$\begin{aligned} \delta\hat{J} = {} & h_\mathbf{x}\Big|_{t_\mathrm{M}} \delta\mathbf{x}_\mathrm{M} + h_\mathbf{u}\Big|_{t_\mathrm{M}} \delta\mathbf{u}_\mathrm{M} + h_t\Big|_{t_\mathrm{M}} \delta t_\mathrm{M} - \left(\mathbf{p}^\mathsf{T} \delta\mathbf{x} \right) \Big|_{t_\mathrm{M}} \\ & + \int_{t_0}^{t_\mathrm{M}} \left[(\dot{\mathbf{p}}^\mathsf{T} + \mathbf{p}^\mathsf{T}\mathbf{f}_\mathbf{x})\delta\mathbf{x} + \mathbf{p}^\mathsf{T}\mathbf{f}_\mathbf{u}\delta\mathbf{u} \right] \mathrm{d}t. \end{aligned} \tag{23}$$

As stated in [16], the variations $\delta\mathbf{x}_\mathrm{M}$ and $\delta\mathbf{x}(t_\mathrm{M})$ are generally not equivalent, but are related to each other as follows

$$\delta\mathbf{x}_\mathrm{M} = \dot{\mathbf{x}}(t_\mathrm{M})\, \delta t_\mathrm{M} + \delta\mathbf{x}(t_\mathrm{M}). \tag{24}$$

Using this relationship for the variations in the state $\mathbf{x}(t)$ and analogously for the system output h and the control $\mathbf{u}(t)$, the variation of the extended cost functional results in

$$\begin{aligned}\delta\hat{J} = \left[\left(h_{\mathbf{x}} - \mathbf{p}^{\mathsf{T}}\right)\delta\mathbf{x} + h_{\mathbf{u}}\delta\mathbf{u}\right]\Big|_{t_{\mathrm{M}}} + \left(h_{\mathbf{x}}\dot{\mathbf{x}} + h_{\mathbf{u}}\dot{\mathbf{u}} + h_t\right)\Big|_{t_{\mathrm{M}}}\delta t_{\mathrm{M}} \\ + \int_{t_0}^{t_{\mathrm{M}}}\left[\left(\dot{\mathbf{p}}^{\mathsf{T}} + \mathbf{p}^{\mathsf{T}}\mathbf{f}_{\mathbf{x}}\right)\delta\mathbf{x} + \mathbf{p}^{\mathsf{T}}\mathbf{f}_{\mathbf{u}}\delta\mathbf{u}\right]\mathrm{d}t.\end{aligned} \tag{25}$$

Note that the variation δt_{M} vanishes since it is multiplied by the total time derivative of h, which must be zero at time t_{M} due to Eq. (17), i.e., $\dot{h}(t_{\mathrm{M}}) = (h_{\mathbf{x}}\dot{\mathbf{x}} + h_{\mathbf{u}}\dot{\mathbf{u}} + h_t)|_{t_{\mathrm{M}}} = 0$. The variation of $\hat{J}$ is further simplified in case the variation $\delta\mathbf{x}(t)$ is eliminated by defining the adjoint systems of equations

$$\dot{\mathbf{p}}(t) = -\mathbf{f}_{\mathbf{x}}^{\mathsf{T}}\mathbf{p}(t) \quad \text{with} \quad \mathbf{p}(t_{\mathrm{M}}) = h_{\mathbf{x}}^{\mathsf{T}}\Big|_{t_{\mathrm{M}}}, \tag{26}$$

with the conditions at $t = t_{\mathrm{M}}$ for the adjoint variables $\mathbf{p}(t)$. By solving the adjoint system of equations in Eq. (26) for $\mathbf{p}(t)$ backward in time, the variation of $\hat{J}$ reduces to

$$\delta\hat{J} = \left(h_{\mathbf{u}}\delta\mathbf{u}\right)\Big|_{t_{\mathrm{M}}} + \int_{t_0}^{t_{\mathrm{M}}}\mathbf{p}^{\mathsf{T}}\mathbf{f}_{\mathbf{u}}\delta\mathbf{u}\,\mathrm{d}t. \tag{27}$$

In the same manner as in [21], the control $\mathbf{u}(t)$ is discretized in terms of a spline parametrization as in Eq. (12) leading to

$$\delta\hat{J} = \left[\left(h_{\mathbf{u}}\mathbf{C}\right)\Big|_{t_{\mathrm{M}}} + \int_{t_0}^{t_{\mathrm{M}}}\mathbf{p}^{\mathsf{T}}\mathbf{f}_{\mathbf{u}}\mathbf{C}\,\mathrm{d}t\right]\delta\bar{\mathbf{u}}. \tag{28}$$

Similar as presented in Sect. 2, the adjoint gradient for $\hat{J}$ with respect to $\bar{\mathbf{u}}$ can be defined by

$$\nabla_{\bar{\mathbf{u}}}\hat{J}^{\mathsf{T}} = \left(h_{\mathbf{u}}\mathbf{C}\right)\Big|_{t_{\mathrm{M}}} + \int_{t_0}^{t_{\mathrm{M}}}\mathbf{p}^{\mathsf{T}}\mathbf{f}_{\mathbf{u}}\mathbf{C}\,\mathrm{d}t. \tag{29}$$

It has to be emphasized that the proposed adjoint gradient here evaluates the system output h at $t = t_M$ with regard to the extremal value of the system output, whereas the proposed adjoint gradient in [21, Eq. (28)] considers the corresponding term in the entire time interval $t \in [t_0, t_{\mathrm{F}}]$. A fruitful discussion on the interpretation of the adjoint variables in this context can be found as well in [20, Sect. 5.1.2].

Summarizing, this proposed adjoint gradient for the cost functional in Eq. (29) and the adjoint gradient for the final constraints in Eq. (16) can now be directly provided in this form to any NLP software package for constraint optimization instead of utilizing the default numerical gradient. Replacing the gradient computation is considerably more time efficient, as the evaluation of the adjoint gradient does not depend on the number of grid nodes k selected for the control due to the fact that the size of the adjoint system does not increase with the number of grid nodes, as compared to direct or numerical differentiation. In order to clarify the discussion on different system sizes in the comparison of gradient computation utilizing the adjoint method with direct differentiation, the reader is referred to [22, Fig. 4]. There, a clear depiction of a fair comparison is given.

Moreover, the computational time reduces furthermore due to the fact that the computation of the gradient in Eq. (29) is reduced to the time interval $t \in [t_0, t_{\mathrm{M}}]$. Hence, also the forward solution of the state equations and the backward solution of the adjoint equations are only necessary in $t \in [t_0, t_\mathrm{M}]$ here. Notice, a more detailed discussion on the final constraints using the gradient in Eq. (16) can be found in [9].

4 Application of the proposed adjoint gradient to multiple extremal values

In the previous section, only one extremal value of the system output h has been assumed in the time interval $t \in [t_0, t_\mathrm{F}]$. As illustrated in Fig. 1, several local extremal values may generally occur within the time interval. The goal is then to minimize the global extremal values while preventing all other local extremal values from exceeding the global one during an optimization step. Therefore, the cost functional J in Eq. (19) is extended taking into account for several extremal values

$$J = \max\left(h_1, \ldots, h_N\right), \tag{30}$$

where $h_i = h(\mathbf{x}(t_i), \mathbf{u}(t_i), t_i)$ and $\dot{h}(\mathbf{x}(t_i), \mathbf{u}(t_i), t_i) = 0$, with $i \in \{1, \ldots, N\}$.

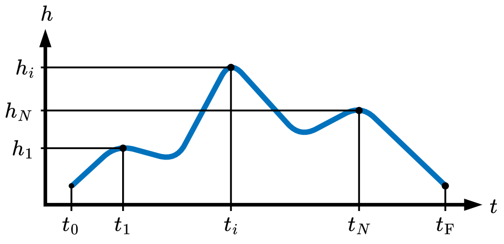


Fig. 1 Several local extremal values of the system output h may be present in the specified time interval $t \in [t_0, t_\mathrm{F}]$

The adjoint gradient $\nabla_{\bar{\mathbf{u}}} \hat{J}$ in Eq. (29) allows the minimization of individual local extremal values. However, the challenge is to consider all gradients simultaneously in order to achieve the optimization goal including several extremal values. A promising approach is the goal attainment method, which originates from Gembicki's PhD thesis [12] which presents an integrated, multi-objective optimization based on a vector index approach. This multi-objective optimization strategy is implemented in MATLAB's `fminimax` function [32] which is designed to minimize the maximum value of a set of objective functions $F_i(x)$, while allowing for general constraints. Specifically, it solves optimization problems in the form of

$$\min_x \max_i F_i(x). \tag{31}$$

To solve this so-called minimax problem, `fminimax` reformulates Eq. (31) as an unscaled goal attainment problem that seeks to reduce a set of nonlinear functions $F_i(x)$ below specified target values $F_i^\star$, taking into account user-defined scalar weights w_i, i.e., a degree of under or over attainment of the desired level. The objective is to minimize the maximum normalized deviations $(F_i(x) - F_i^\star)/w_i$. This formulation ensures that the problem remains well-defined, even if not all goals can be simultaneously satisfied. The routine `fminimax` uses this goal attainment method with targets $F_i^\star = 0$ and weights $w_i = 1$. In addition to this setting, the minimax problem is formulated as a smooth goal attainment problem. This

MATLAB routine can be utilized for a multi-objective optimization considering the cost functional in Eq. (30) with

$$\min_{\bar{\mathbf{u}}} \max_{i} h_i = \min_{\bar{\mathbf{u}}} \max_{i} \hat{J}_i(\bar{\mathbf{u}}), \tag{32}$$

for computing the optimal grid nodes $\bar{\mathbf{u}}$ of the control. The routine `fminimax` provides numerical gradients as fully-automated procedure using finite differences. However, providing the proposed adjoint gradients derived in Sect. 2 and Sect. 3 for the final constraints and the cost functional, respectively, the multi-objective optimization benefits from analytical descent directions. Hence, the adjoint gradient $\nabla_{\bar{\mathbf{u}}}\hat{J}$ in Eq. (29) can be reused in the `fminimax` routine in the form

$$\nabla_{\bar{\mathbf{u}}}\hat{J}_i^{\mathsf{T}} = (h_{\mathbf{u}}\mathbf{C})\Big|_{t_i} + \int_{t_0}^{t_i} \mathbf{p}_i^{\mathsf{T}}\mathbf{f}_{\mathbf{u}}\mathbf{C}\,\mathrm{d}t, \tag{33}$$

whereby $\nabla_{\bar{\mathbf{u}}}\hat{J}_i$ is the gradient of the local extremal value h_i, for a number of N local extremal values. Note that each individual extremal value h_i corresponds to an individual adjoint system of equations and therefore providing N individual adjoint gradients. Since the adjoint differential equations are solved in different time intervals $t \in [t_0, t_i]$ for each extremal value for $i = 1, \ldots, N$, the adjoint variables $\mathbf{p}(t)$ have to be defined for each extremal value, abbreviated by $\mathbf{p}_i(t)$ and given by the adjoint system of equations

$$\dot{\mathbf{p}}_i(t) = -\mathbf{f}_{\mathbf{x}}^{\mathsf{T}}\mathbf{p}_i(t) \quad \text{with} \quad \mathbf{p}_i(t_i) = h_{\mathbf{x}}^{\mathsf{T}}\Big|_{t_i}. \tag{34}$$

4.1 Alternative solution strategy

The handling of the optimization task in the form of Eq. (32) requires specially tailored optimization toolboxes, as e.g., the MATLAB function `fminimax`. By applying an appropriate solution strategy for minimax problems, an equivalent optimization problem is obtained for which any NLP software package can be chosen. Following the formulation of [5], the cost functional in Eq. (30) is transformed into

$$J \cong \left(\sum_{i=1}^{N} h_i^{\alpha}\right)^{\frac{1}{\alpha}}, \tag{35}$$

whereby the parameter α has to be selected large enough to particularly emphasize the largest value, i.e., max $(h_1, \ldots, h_N)$, as illustrated in the Appendix. Notice, in this formulation each extremal value h_i is assumed to be positive, otherwise an adjustment is required as given in [5]. Each summand in Eq. (35) corresponds to Eq. (19), thus the adjoint gradient for $\hat{J}$ with respect to $\bar{\mathbf{u}}$ results in

$$\nabla_{\bar{\mathbf{u}}}\hat{J} = \left(\sum_{i=1}^{N} h_i^{\alpha}\right)^{\frac{1-\alpha}{\alpha}} \sum_{i=1}^{N} h_i^{\alpha-1}\,\nabla_{\bar{\mathbf{u}}}\hat{J}_i. \tag{36}$$

4.2 Algorithm

Both approaches are applicable for solving the optimization task. As both procedures are almost identical, they are summarized together in Algorithm 1, whereby differences are

Algorithm 1: Minimize extremal value of system output with adjoint gradients

```
opt ← optimoptions(SolverName,'SpecifyObjectiveGradient', true,
                   'SpecifyConstraintGradient',true) ;                // Cf. [32]
$(k, \bar{\mathbf{u}}) \leftarrow (\mathbb{N}, \bar{\mathbf{u}}_{\text{init}})$ ;                                       // Step 1
while criteria_stop $\neq$ fulfilled do
    $\mathbf{x}, \mathbf{t} \leftarrow$ solve(Eq. (1), $[t_0, t_F]$, $\mathbf{x}_0$) ;                     // Step 2
    $\mathbf{t}_{\dot{h}=0} \leftarrow \{t_i \mid i \in \{1, \dots, N\}, \dot{h}(\mathbf{x}(t_i), \mathbf{u}(t_i), t_i) = 0\}$ ;       // Step 3
    $\mathbf{h} \leftarrow \{h(\mathbf{x}(t_i), \mathbf{u}(t_i), t_i) \mid i \in \{1, \dots, N\}\}$ ;                  // Step 3
    if Algorithm == $(a)$ then
        $\mathbf{J} \leftarrow \mathbf{h}$ ;                                               // Step 3(a)
    else
        $J \leftarrow$ Eq. (35) ;                                          // Step 3(b)
    end
    $\boldsymbol{\phi} \leftarrow$ Eq. (3) ;                                            // Step 4
    for $i \leftarrow 1$ to $N$ do
        $\mathbf{p}_i \leftarrow$ solve(Eq. (34), $[t_0, t_i]$, $h_{\mathbf{x}}(\mathbf{x}(t_i), \mathbf{u}(t_i), t_i)$) ;           // Step 5
    end
    $\mathbf{R} \leftarrow$ solve(Eq. (11), $[t_0, t_F]$, $\boldsymbol{\phi}_{\mathbf{x}}(\mathbf{x}(t_F), t_F)$) ;                // Step 6
    for $i \leftarrow 1$ to $N$ do
        $\nabla_{\bar{\mathbf{u}}} \hat{J}_i \leftarrow$ Eq. (33) ;                                     // Step 7
    end
    if Algorithm == $(b)$ then
        $\nabla_{\bar{\mathbf{u}}} \hat{J} \leftarrow$ Eq. (36) ;                                   // Step 7(b)
    end
    $\nabla_{\bar{\mathbf{u}}} \bar{\boldsymbol{\phi}} \leftarrow$ Eq. (16) ;                                       // Step 8
    if Algorithm == $(a)$ then
        $\bar{\mathbf{u}} \leftarrow$ fminimax($\mathbf{J}, \nabla_{\bar{\mathbf{u}}} \hat{\mathbf{J}}, \bar{\mathbf{u}}, \boldsymbol{\phi}, \nabla_{\bar{\mathbf{u}}} \bar{\boldsymbol{\phi}}$, opt) ;              // Step 9(a)
    else
        $\bar{\mathbf{u}} \leftarrow$ fmincon($J, \nabla_{\bar{\mathbf{u}}} \hat{J}, \bar{\mathbf{u}}, \boldsymbol{\phi}, \nabla_{\bar{\mathbf{u}}} \bar{\boldsymbol{\phi}}$, opt) ;               // Step 9(b)
    end
end
return $\bar{\mathbf{u}}^{\star}$
```

explicitly indicated by sub-item (a) for the MATLAB function `fminimax` and (b) for the alternative solution strategy. The necessary steps are as follows:

1. Define the number of grid nodes k for $\bar{\mathbf{u}}$ and assign initial values $\bar{\mathbf{u}}_{\text{init}}$.
2. Solve the state equations in form of Eq. (1) for the prescribed initial values $\mathbf{x}_0$ in the given time interval $t \in [t_0, t_F]$.
3. Determine the extremal values h_i and corresponding time instances t_i.
   (a) The MATLAB function `fminimax` internally evaluates Eq. (30).
   (b) Compute the equivalent cost functional by Eq. (35).
4. Evaluate the final constraints $\boldsymbol{\phi}$ at time t_F
5. Compute adjoint variables $\mathbf{p}_i(t)$ backwards in time in the interval $t \in [t_0, t_i]$ for each occurring maximum h_i by Eq. (34).
6. Compute adjoint variables $\mathbf{R}(t)$ backwards in time in the interval $t \in [t_0, t_F]$ by Eq. (11).
7. Compute the adjoint gradient $\nabla_{\bar{\mathbf{u}}} \hat{J}_i$ for each occurring maximum h_i by Eq. (33).

   (b) Combine these gradients into the adjoint gradient $\nabla_{\bar{\mathbf{u}}}\hat{J}$ by Eq. (36).
8. Compute the adjoint gradient $\nabla_{\bar{\mathbf{u}}}\bar{\boldsymbol{\phi}}$ by Eq. (16).
9. Compute the update of $\bar{\mathbf{u}}$
   (a) with the MATLAB function `fminimax`.
   (b) with any NLP software package, e.g., MATLAB function `fmincon`.
10. Repeat steps $2-9$ until the required optimality or function tolerances for constrained optimization problems are satisfied, e.g., following [15, 18] for the Karush–Kuhn–Tucker conditions.

Note that in step 1, any values can be set for $\bar{\mathbf{u}}_{\text{init}}$, even a trivial zero control. In order to illustrate the basic idea of this work which focuses on the minimization of an extremal value, a proper initialization for $\bar{\mathbf{u}}_{\text{init}}$ is given as follows. The MATLAB function `fmincon` with sequential quadratic programming (SQP) option is utilized to find a control $\mathbf{u}$, which only has to fulfill the final constraints $\boldsymbol{\phi}$. For this task, the adjoint gradient $\nabla_{\bar{\mathbf{u}}}\bar{\boldsymbol{\phi}}$ from step 8 is applied.

Furthermore, it should be noted that the analytical adjoint gradient derivations and implementations in step 7 and 8 are verified using numerical gradients computed via finite differences in the following numerical examples.

5 Numerical examples

In the following, the workflow presented in Sect. 4.2 is utilized to demonstrate the minimization of an extremal value of a system output with given final constraints using the proposed adjoint gradients. In principle, both approaches lead to similar results. The discussion is based on the results obtained by the optimization using the MATLAB function `fminimax`. In case the alternative solution strategy using the MATLAB function `fmincon` with the SQP option leads to a deviating result, comments are enclosed.

First, the maximum of the reaction force of the academic example of a one-mass oscillator is minimized. Moreover, the maximum occurring absolute velocity of the TCP of a two-axis robot is minimized which demonstrates a possible application in a HRC scenario. Finally, all results are compared and discussed in order to evaluate the impact of the presented adjoint gradient.

5.1 One-mass oscillator

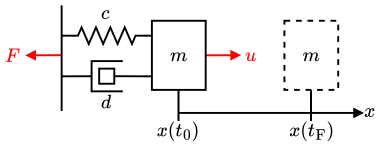
As a first example, the one-mass oscillator in Fig. 2 is examined, consisting of a mass $m = 1$ kg, a spring with stiffness $c = 1$ N/m and a damper with damping coefficient $d = 0.5$ N s/m. The mass m is at the time $t_0 = 0$ s in its equilibrium position, i.e., $x(t_0) = 0$ m, and zero velocity $v(t_0) = 0$ m/s. At the final time $t_\text{F} = 0.9$ s, the mass m is supposed to reach the specified position $x(t_\text{F}) = x_\text{F} = 1$ m with the requested velocity $v(t_\text{F}) = v_\text{F} = 0$ m/s. The control force $u(t)$ required for the displacement of the mass m is limited to ± 16 N, in order to achieve a more realistic behavior.

The state equations of the one-mass oscillator are defined by the first-order differential system

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, u) = \begin{pmatrix} v \\ \frac{1}{m}(u - dv - cx) \end{pmatrix}, \tag{37}$$

in which the state vector $\mathbf{x}(t) = (x(t), v(t))^\mathsf{T}$ includes the position $x(t)$ and velocity $v(t)$ of the mass m. The objective is to find the control force $u(t)$ that minimizes the extremal value

Fig. 2 One-mass oscillator is to be moved from its given initial position $x(t_0)$ to a defined final position $x(t_\mathrm{F})$



of the reaction force $F(x, v)$ during the displacement of the mass m. Hence, the system output $h(x, v)$ could be, e.g., defined as

$$h(x, v) = \frac{1}{2} F(x, v)^2 = \frac{1}{2} \left(c\, x(t) + d\, v(t)\right)^2 , \tag{38}$$

and the final constraints are specified as

$$\boldsymbol{\phi}(x, v) = \left. \begin{pmatrix} x(t) - x_\mathrm{F} \\ v(t) - v_\mathrm{F} \end{pmatrix} \right|_{t_\mathrm{F}} . \tag{39}$$

For the optimization in finite-dimensional space, the control $u(t)$ is discretized as defined in Eq. (12) by cubic spline interpolation, as e.g., used in [21], whereby an equidistant discretization of the time interval $t \in [t_0, t_\mathrm{F}]$ with $k = 20$ grid nodes is provided.

In Fig. 3, the comparison of the position $x(t)$ and velocity $v(t)$ of the mass m due to the initial control force u_init and optimized control force $u^\star(t)$ is shown. The time evolution of the velocity $v(t)$ changes drastically, especially the amplitude and the time of the maximum differ considerably. This behavior is due to the properties and interaction of the spring and damper. The spring force increases steadily as it expands, so the damper force has to decrease to a similar extent, in order to enable the maximum of the reaction force $F(x, v)$ to be reduced. The higher initial velocity in the optimized case leads to a significantly different displacement of the mass m as compared to the displacement due to the initial control force u_init.

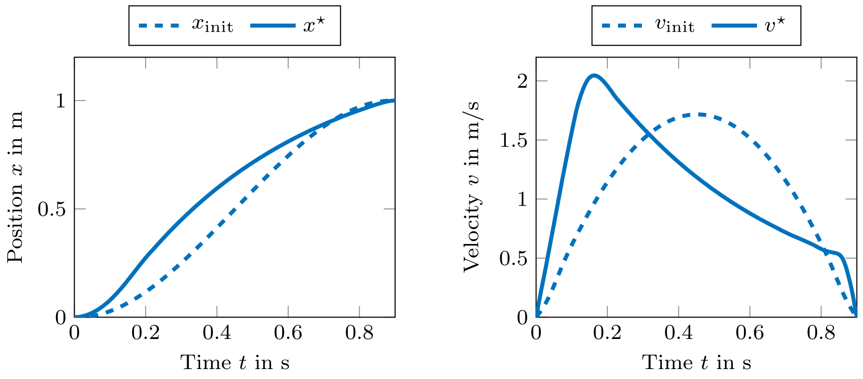


Fig. 3 Comparison of the time evolution of the states due to the initial control force and optimized control force obtained by `fminimax`: (left) position x of the mass m (right) velocity v of the mass m

Figure 4 shows the time evolution of the optimized control force $u^\star(t)$. Furthermore, the reaction force $F(x, v)$ is plotted in a comparison resulting from the initial control $u_\mathrm{init}(t)$ and final control $u^\star(t)$, respectively. As previously stated, the mass m should accelerate rapidly

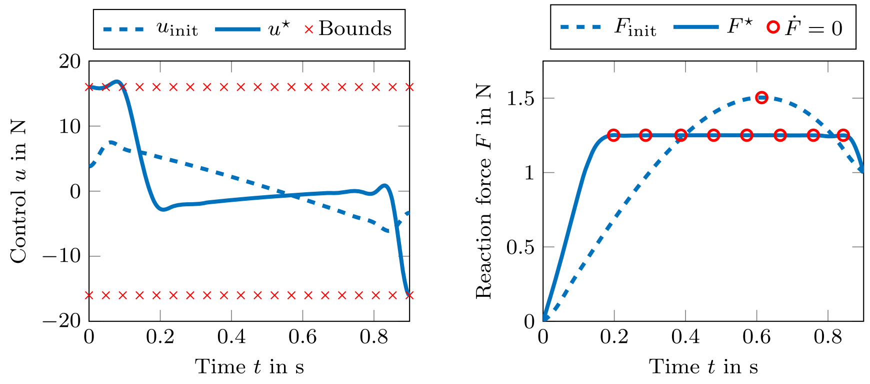


Fig. 4 (left) Comparison of the time evolution of the initial control force u_{init} and the optimized control force $u^\star$ obtained by `fminimax`, which are both limited by lower and upper bounds. (right) Comparison of the time evolution of the reaction force F due to the initial and optimized control force and hollow circles mark maxima with $\dot{F} = 0$

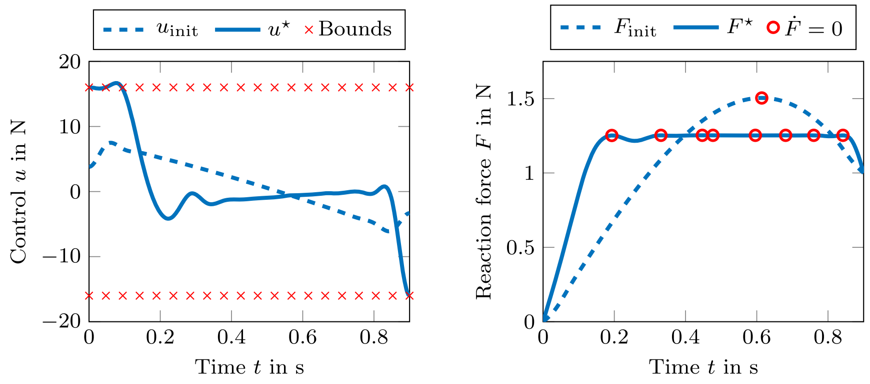


Fig. 5 (left) Comparison of the time evolution of the initial control force u_{init} and the optimized control force $u^\star$ obtained by `fmincon`, which are both limited by lower and upper bounds. (right) Comparison of the time evolution of the reaction force F due to the initial and optimized control force and hollow circles mark maxima with $\dot{F} = 0$

to a maximum value with respect to the velocity and then decelerate steadily. The time evolution of the optimal control force $u^\star(t)$ can be seen in Fig. 4 (left): the optimized control $u^\star(t)$ is initially at the upper limit and then remains at a moderate amplitude. As a result, the reaction force $F(x, v)$ in Fig. 4 (right) changes from a single occurring maximum to 8 maxima with almost identical amplitudes forming a plateau. Through the optimization, the extremal value of the reaction force is reduced from $F_{\text{init}} = 1.504$ N to $F^\star = 1.251$ N. Figure 5 shows the time evolution of the optimized control force $u^\star(t)$, but in this case computed by the alternative solution strategy with $\alpha = 1000$. In comparison to Fig. 4, the same number of maxima occur, but not all time points are identical. Moreover, the extremal value of the reaction force is reduced to $F^\star = 1.253$ N. Approximately ten additional iterations are required.

Table 1 shows the time-efficient computation in the case of using adjoint gradients instead of using numerical gradients in terms of forward finite differences in the example of the one-

Table 1 One-mass oscillator: Comparison of the performance behavior when using numerical versus adjoint gradients in a `fminimax` optimization (approx. 190 iterations)

k	Type of gradient computation	Runtime	Function evaluations
20	`fminimax` forward finite differences	3 min 46 s	4461
	proposed adjoint gradient method	16 s	406

mass oscillator. Discussing also the adjoint gradient from the alternative solution strategy presented in Sect. 4.1, the further iterations increase the runtime to 21 s.

5.2 Planar two-axis robot

In this section, a two-axis robot is investigated in an optimal control setting. The goal is to find the controls in the joints minimizing the maximum of the absolute velocity of the TCP during a rest-to-rest maneuver. Based on the study in [30] on the contact forces resulting from possible collisions at certain velocities in HRC, the maximum occurring velocity should be reduced as much as possible and be forced below a certain limit.

In the following, the two-axis robot model presented in [9] is utilized. This robot is shown in Fig. 6 and the lengths, masses, moments of inertia and the distances between joints and center of gravity are listed in Table 2. The robot is described in minimal coordinates using the joint angles $\varphi_1(t)$ and $\varphi_2(t)$. The state equations, cf. [9], are provided again as first-order differential system. The state vector is given by

$$\mathbf{x}(t) = (\varphi_1, \varphi_2, \omega_1, \omega_2)^\mathsf{T}, \tag{40}$$

whereby $\omega_1(t) := \dot{\varphi}_1(t)$ and $\omega_2(t) := \dot{\varphi}_2(t)$ are the angular velocities. At the beginning $t_0 = 0$ s the robot is at rest, i.e., $\omega_1(t_0) = \omega_2(t_0) = 0$ rad/s, and the two angles are given by $\varphi_1(t_0) = -\pi/4$ rad and $\varphi_2(t_0) = 0$ rad. In the revolute joint between the ground and the first body of the robot, the control $u_1(t)$ acts within given limits ± 4 N m. In the revolute joint between the first and second body of the robot, the control $u_2(t)$ acts within given limits ± 2 N m.

Fig. 6 Planar two-axis robot from [9] investigated in the optimal control problem

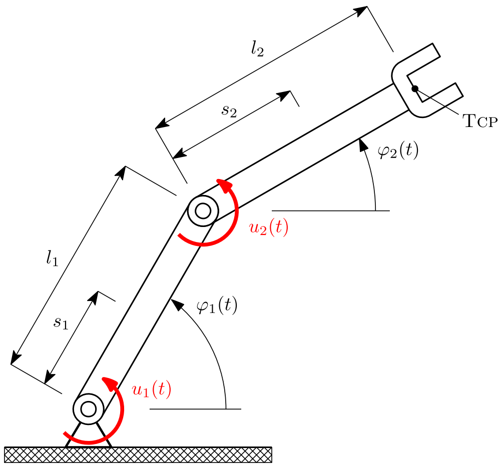


Table 2 Parameters of the two-axis robot, cf. [9]

Component	Length	Distance to center of gravity	Mass	Moment of inertia
Body 1	$l_1 = 1$ m	$s_1 = 0.5$ m	$m_1 = 1$ kg	$J_1 = 0.0833$ kg m^2
Body 2	$l_2 = 1$ m	$s_2 = 0.5$ m	$m_2 = 0.5$ kg	$J_2 = 0.0417$ kg m^2
TCP	-	-	$m_3 = 1$ kg	-

In order to realize the rest-to-rest maneuver, final constraints at time $t_F = 2$ s are specified to ensure that the TCP reaches the desired final position $x_F = 1$ m and $y_F = 1$ m as

$$\boldsymbol{\phi}\,(\varphi_1, \varphi_2, \omega_1, \omega_2) = \left.\begin{pmatrix} l_1 \cos(\varphi_1(t)) + l_2 \cos(\varphi_2(t)) - x_F \\ l_1 \sin(\varphi_1(t)) + l_2 \sin(\varphi_2(t)) - y_F \\ \omega_1(t) \\ \omega_2(t) \end{pmatrix}\right|_{t_F}. \tag{41}$$

The velocity of the TCP can be derived by

$$\mathbf{v}_{TCP}\,(\varphi_1, \varphi_2, \omega_1, \omega_2) = \begin{pmatrix} -l_1\,\omega_1(t) \sin(\varphi_1(t)) - l_2\,\omega_2(t) \sin(\varphi_2(t)) \\ l_1\,\omega_1(t) \cos(\varphi_1(t)) + l_2\,\omega_2(t) \cos(\varphi_2(t)) \end{pmatrix}. \tag{42}$$

In order to minimize the maximum of the absolute velocity $||\mathbf{v}_{TCP}||$ during this rest-to-rest maneuver of the robot, the system output is defined by

$$h\,(\varphi_1, \varphi_2, \omega_1, \omega_2) = \frac{1}{2}\,||\mathbf{v}_{TCP}||^2. \tag{43}$$

For a finite-dimensional optimization space, the control $\mathbf{u}(t)$ is discretized as defined in Eq. (12) by cubic spline interpolation, as e.g., used in [21], whereby an equidistant discretization of the time interval $t \in [t_0, t_F]$ with $k = 20$ grid nodes is provided.

In Fig. 7, the comparison of the TCP trajectory $\mathbf{x}_{TCP}$ and absolute velocity $||\mathbf{v}_{TCP}||$ due to the initial control $\mathbf{u}_{init}(t)$ and optimized control $\mathbf{u}^\star(t)$ is shown. The result is comparable to the plateau found in the optimization of the academic example of a one-mass oscillator, which shows causal behavior for this optimization task. One single maximum enforced by the initial control is transformed to 7 maxima with almost identical amplitudes in the time evolution of the absolute velocity $||\mathbf{v}_{TCP}||$ of the TCP. Finally, the extremal value is reduced from $||\mathbf{v}_{TCP,init}|| = 1.745$ m/s to $||\mathbf{v}^\star_{TCP}|| = 1.249$ m/s. Figure 8 shows the time evolution of the control variables $\mathbf{u}(t)$ resulting from the initial control and the optimized control resulting from the multi-objective optimization. Note that the overshoot at the control limits is caused by the spline parameterization and could be reduced if more sophisticated interpolation methods are used here, as e.g., an Akima spline parameterization, which flattens the curve near local extrema by avoiding overshoots and accurately connects flat regions [1]. The results obtained by the alternative solution strategy with $\alpha = 1000$ are very similar to the graphs in Fig. 7 and Fig. 8. The main difference is the occurrence of only 6 instead of 7 maxima. Approximately 30 additional iterations are required.

Table 3 shows the time-efficient computation in the case of using adjoint gradients instead of using numerical gradients in terms of forward finite differences in the example of the two-axis robot. Discussing also the adjoint gradient from the alternative solution strategy presented in Sect. 4.1, the further iterations increase the runtime to 62 s.

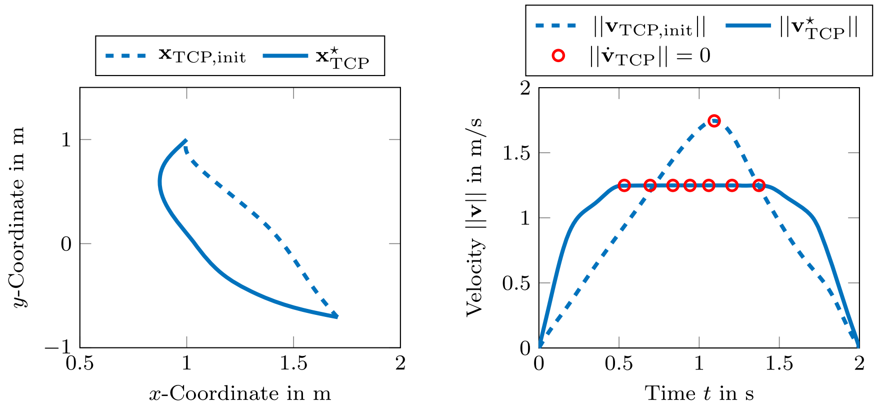


Fig. 7 (left) Comparison of the TCP trajectories due to the initial controls and the optimized controls obtained by `fminimax`. (right) Comparison of the time evolution of the TCP velocity due to the initial and optimized controls and hollow circles mark maxima with $||\dot{\mathbf{v}}_{\text{TCP}}|| = 0$

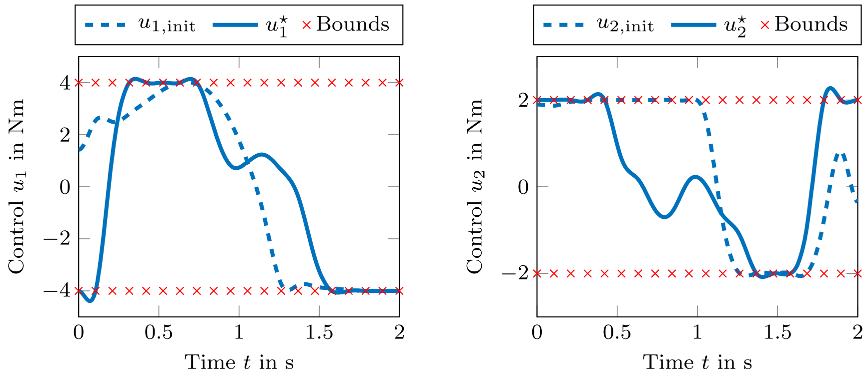


Fig. 8 Comparison of the time evolution of the initial controls $\mathbf{u}_{\text{init}}$ and the optimized controls $\mathbf{u}^\star$ obtained by `fminimax`, which are both limited by lower and upper bounds

Table 3 Two-axis robot: Comparison of the performance behavior when using numerical versus adjoint gradients in a `fminimax` optimization (approx. 100 iterations)

k	Type of gradient computation	Runtime	Function evaluations
20	`fminimax` forward finite differences	12 min 26 s	4263
	proposed adjoint gradient method	41 s	211

6 Conclusion

In this paper, an adjoint gradient approach is presented for a gradient-based optimization to minimize the extremal value of a system output while satisfying final constraints. One application for such an optimization task occurs, e.g., in robotics, when the aim is to minimize the maximum velocity of the TCP during a rest-to-rest maneuver. In order to provide the necessary gradients in a time-efficient manner, the computation by the adjoint method is investigated here. A challenging aspect in this regard concerns the presence of several extremal points in the observation period during the optimization leading to the need of a multi-objective optimization. This paper presents the derivation of the necessary adjoint gradients and shows the workflow for incorporating these gradients into a multi-objective optimization framework. Two examples show the efficient behavior when using analytically derived adjoint gradients instead of gradient computation by numerical differentiation in terms of function evaluations and runtime.

Appendix: Influence of parameter α

In order to clarify the impact of the parameter α in Sect. 4.1 more clearly, Eq. (35) is modified as follows

$$J \cong \left(\sum_{i=1}^{N} h_i^{\alpha}\right)^{\frac{1}{\alpha}} = \left(h_1^{\alpha} + \sum_{i=2}^{N} (\xi_i\, h_1)^{\alpha}\right)^{\frac{1}{\alpha}} = h_1 \left(1 + \sum_{i=2}^{N} \xi_i^{\alpha}\right)^{\frac{1}{\alpha}}, \tag{44}$$

whereby h_1 is assumed to be the global extremal value and $\xi_i =]0, 1[$ is the ratio between the local extremal value h_i and h_1. As Fig. 9 shows, ξ and α are connected. For $\xi \to 0$, small values for α are sufficient to particular emphasize the global extremal value, while much higher values for α are required for $\xi \to 1$. Once a plateau occurs during optimization, ξ tends towards the upper bound. Consequently, a too small value for α leads to a premature termination of the optimization, i.e., instead of a "smooth" plateau, a plateau with clearly isolated extremal points results.

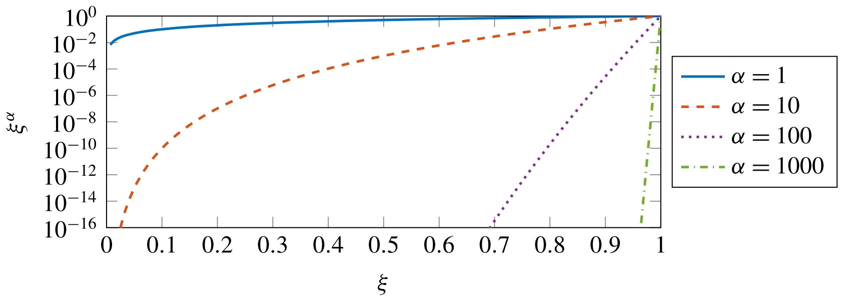


Fig. 9 Influence of parameter α with respect to ξ

Acknowledgements This research was funded in whole or in part by the Austrian Science Fund (FWF): P37110-NBL (No. 10.55776/P37110). Moreover, this project has received funding from the European Union's IBW/EFRE and Joint Transition Fund granted and managed by "Amt der Oberösterreichischen Landesregierung Abteilung Wirtschaft". Karin Nachbagauer acknowledges support from the Technical University of Munich - Institute for Advanced Study and the TUM Integrative Research Fund, provided by the seed funding initiative of the Munich Institute of Robotics and Machine Intelligence (MIRMI).

Author contributions PZ: Conceptualization, methodology, software, validation, formal analysis, investigation, writing - original draft, visualization. DL: Methodology, software, validation, formal analysis, writing - review and editing. PE: Software, validation, formal analysis, writing - review and editing. WS: Conceptualization, methodology, validation, formal analysis, writing - review and editing, supervision. KN: Conceptualization, methodology, validation, formal analysis, investigation, writing - review and editing, supervision, resources, funding acquisition.

Funding information Open access funding provided by FH Oberösterreich Studienbetriebs GmbH. Open access funding provided by University of Applied Sciences Upper Austria.

Data availability No datasets were generated or analysed during the current study.

Declarations

Competing interests The authors declare no competing interests.

References

1. Akima, H.: A new method of interpolation and smooth curve fitting based on local procedures. J. ACM **17**(4), 589–602 (1970). https://doi.org/10.1145/321607.321609
2. Barron, E., Ishii, H.: The Bellman equation for minimizing the maximum cost. Nonlinear Anal., Theory Methods Appl. **13**(9), 1067–1090 (1989). https://doi.org/10.1016/0362-546X(89)90096-5
3. Bryson, A.E., Ho, Y.C.: Applied Optimal Control: Optimization, Estimation, and Control. Taylor & Francis, New York (1975). https://doi.org/10.1201/9781315137667
4. Caramia, M., Dell'Olmo, P.: Multi-objective Management in Freight Logistics: Increasing Capacity, Service Level, Sustainability, and Safety with Optimization Algorithms. Springer, Berlin (2020). https://doi.org/10.1007/978-3-030-50812-8
5. Charalambous, C., Bandler, J.W.: Non-linear minimax optimization as a sequence of least pth optimization with finite values of p. Int. J. Syst. Sci. **7**(4), 377–391 (1976). https://doi.org/10.1080/00207727608941924
6. Daellenbach, H.G., De Kluyver, C.A.: Note on multiple objective dynamic programming. J. Oper. Res. Soc. **31**(7), 591–594 (1980). https://doi.org/10.1057/jors.1980.114
7. Edgeworth, F.Y.: Mathematical Psychics: An Essay on the Application of Mathematics to the Moral Sciences. C. Kegan Paul & Co., London (1881)
8. Eichmeir, P., Nachbagauer, K., Steiner, W.: The adjoint gradient method for time-optimal control of a moon landing: ascent, descent, and abort. In: Volume 2: 16th International Conference on Multibody Systems, Nonlinear Dynamics, and Control (MSNDC), IDETC-CIE2020. American Society of Mechanical Engineers (2020). https://doi.org/10.1115/detc2020-22034
9. Eichmeir, P., Nachbagauer, K., Lauß, T., Sherif, K., Steiner, W.: Time-optimal control of dynamic systems regarding final constraints. J. Comput. Nonlinear Dyn. **16**(3), 031003 (2021). https://doi.org/10.1115/1.4049334
10. Eichmeir, P., Nachbagauer, K., Steiner, W.: The use of slack variables in the adjoint method handling inequality constraints in optimal control and the application to tumor drug dosage. J. Comput. Nonlinear Dyn. **20**(2), 021004 (2025). https://doi.org/10.1115/1.4067128
11. Eichmeir, P., Steiner, W., Nachbagauer, K.: The Adjoint Method for Optimal Control of Multibody Systems for Free End Time and Final Constraints. Multibody Syst. Dyn. (2025). https://doi.org/10.1007/s11044-025-10114-9. Submitted
12. Gembicki, F.W.: Vector Optimization for Control with Performance and Parameter Sensitivity Indices. Ph.D. thesis, Case Western Reserve University, Cleveland, Ohio (1974)
13. Gembicki, F., Haimes, Y.: Approach to performance and sensitivity multiobjective optimization: the goal attainment method. IEEE Trans. Autom. Control **20**(6), 769–771 (1975). https://doi.org/10.1109/tac.1975.1101105
14. Gianatti, J., Aragone, L.S., Lotito, P.A., Parente, L.A.: Solving minimax control problems via nonsmooth optimization. Oper. Res. Lett. **44**(5), 680–686 (2016). https://doi.org/10.1016/j.orl.2016.08.001
15. Karush, W.: Minima of functions of several variables with inequalities as side constraints. Master's thesis, Department of Mathematics, University of Chicago (1939)

16. Kirk, D.E.: Optimal Control Theory: An Introduction. Dover, New York (2004)
17. Koppenborg, M., Nickel, P., Naber, B., Lungfiel, A., Huelke, M.: Effects of movement speed and predictability in human–robot collaboration. Hum. Factors Ergon. Manuf. Serv. Ind. **27**(4), 197–209 (2017). https://doi.org/10.1002/hfm.20703
18. Kuhn, H.W., Tucker, A.W.: Nonlinear programming. In: Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, pp. 481–492 (1951)
19. Lacevic, B., Zanchettin, A.M., Rocco, P.: Safe human-robot collaboration via collision checking and explicit representation of danger zones. IEEE Trans. Autom. Sci. Eng. **20**(2), 846–861 (2023). https://doi.org/10.1109/TASE.2022.3167772
20. Lichtenecker, D., Nachbagauer, K.: A discrete adjoint gradient approach for equality and inequality constraints in dynamics. Multibody Syst. Dyn. **61**(1), 103–130 (2024). https://doi.org/10.1007/s11044-024-09965-5
21. Lichtenecker, D., Rixen, D., Eichmeir, P., Nachbagauer, K.: On the use of adjoint gradients for time-optimal control problems regarding a discrete control parameterization. Multibody Syst. Dyn. **59**(3), 313–334 (2023). https://doi.org/10.1007/s11044-023-09898-5
22. Lichtenecker, D., Eichmeir, P., Nachbagauer, K.: On the usage of analytically computed adjoint gradients in a direct optimization for time-optimal control problems. In: Nachbagauer, K., Held, A. (eds.) Optimal Design and Control of Multibody Systems. IUTAM Bookseries, vol. 42. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-50000-8_14
23. Lu, P., Vinh, N.X.: Optimal control problems with maximum functional. J. Guid. Control Dyn. **14**(6), 1215–1223 (1991). https://doi.org/10.2514/3.20777
24. Miettinen, K.: Nonlinear Multiobjective Optimization. Springer (1998). https://doi.org/10.1007/978-1-4615-5563-6
25. Molina, E., Rapaport, A., Ramírez, H.: Equivalent formulations of optimal control problems with maximum cost and applications. J. Optim. Theory Appl. **195**(3), 953–975 (2022). https://doi.org/10.1007/s10957-022-02094-z
26. Nachbagauer, K., Oberpeilsteiner, S., Sherif, K., Steiner, W.: The use of the adjoint method for solving typical optimization problems in multibody dynamics. J. Comput. Nonlinear Dyn. **10**(6), 061011 (2015). https://doi.org/10.1115/1.4028417
27. Nachbagauer, K., Zallinger, P., Buchner, L., Froschauer, R.: Showcase of an optimal control problem in robotics for integrating the system matrices from a multibody simulation code to generate adjoint gradients for optimization. In: M2P 2025 - Second International Conference Math 2 Product: Emerging Technologies in Computational Science for Industry, Sustainability and Innovation, Valencia, Spain (2025)
28. Pareto, V.: Manuale di Economia Politica (1906). Societa Editrice Libraria; Translated into English by A.S. Schwier as Manual of Political Economy, Macmillan, New York (1971)
29. Pereira, J.L.J., Oliver, G.A., Francisco, M.B., Cunha, S.S., Gomes, G.F.: A review of multi-objective optimization: methods and algorithms in mechanical engineering problems. Arch. Comput. Methods Eng. **29**(4), 2285–2308 (2021). https://doi.org/10.1007/s11831-021-09663-x
30. Ponikelský, J., Chalupa, M., Černohlávek, V., Štěrba, J.: Force and pressure dependent asymmetric workspace research of a collaborative robot and human. Symmetry **16**(1), 131 (2024). https://doi.org/10.3390/sym16010131
31. Scalera, L., Giusti, A., Vidoni, R., Gasparetto, A.: Enhancing fluency and productivity in human-robot collaboration through online scaling of dynamic safety zones. Int. J. Adv. Manuf. Technol. **121**(9), 6783–6798 (2022). https://doi.org/10.1007/s00170-022-09781-1
32. The MathWorks Inc.: Optimization Toolbox version: 24.1 (R2024a). Natick, Massachusetts, United States (2024). https://de.mathworks.com/help/optim (accessed 15/04/2025)
33. Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S.J., Brett, M., Wilson, J., Millman, K.J., Mayorov, N., Nelson, A.R.J., Jones, E., Kern, R., Larson, E., Carey, C.J., Polat, I., Feng, Y., Moore, E.W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E.A., Harris, C.R., Archibald, A.M., Ribeiro, A.H., Pedregosa, F., van Mulbregt, P., Vijaykumar, A., Bardelli, A.P., Rothberg, A., Hilboll, A., Kloeckner, A., Scopatz, A., Lee, A., Rokem, A., Woods, C.N., Fulton, C., Masson, C., Häggström, C., Fitzgerald, C., Nicholson, D.A., Hagen, D.R., Pasechnik, D.V., Olivetti, E., Martin, E., Wieser, E., Silva, F., Lenders, F., Wilhelm, F., Young, G., Price, G.A., Ingold, G.L., Allen, G.E., Lee, G.R., Audren, H., Probst, I., Dietrich, J.P., Silterra, J., Webber, J.T., Slavič, J., Nothman, J., Buchner, J., Kulick, J., Schönberger, J.L., de Miranda Cardoso, J.V., Reimer, J., Harrington, J., Rodríguez, J.L.C., Nunez-Iglesias, J., Kuczynski, J., Tritz, K., Thoma, M., Newville, M., Kümmerer, M., Bolingbroke, M., Tartre, M., Pak, M., Smith, N.J., Nowaczyk, N., Shebanov, N., Pavlyk, O., Brodtkorb, P.A., Lee, P., McGibbon, R.T., Feldbauer, R., Lewis, S., Tygier, S., Sievert, S., Vigna, S., Peterson, S., More, S., Pudlik, T., Oshima, T., Pingel, T.J., Robitaille, T.P., Spura, T., Jones, T.R., Cera, T., Leslie, T., Zito, T., Krauss, T., Upadhyay,

U., Halchenko, Y.O., Vázquez-Baeza, Y.: Scipy 1.0: fundamental algorithms for scientific computing in Python. Nat. Methods **17**(3), 261–272 (2020). https://doi.org/10.1038/s41592-019-0686-2
34. Wächter, A., Biegler, L.T.: On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Math. Program. **106**(1), 15–57 (2006). https://doi.org/10.1007/s10107-004-0559-y
35. Wen, Y., Pagilla, P.: Path-constrained and collision-free optimal trajectory planning for robot manipulators. IEEE Trans. Autom. Sci. Eng. **20**(2), 763–774 (2023). https://doi.org/10.1109/TASE.2022.3169989

Authors and Affiliations

Philipp Zallinger[1,2] · Daniel Lichtenecker[3] · Philipp Eichmeir[1,4] · Wolfgang Steiner[1] · Karin Nachbagauer[1,5]

✉ P. Zallinger
philipp.zallinger@fh-wels.at

D. Lichtenecker
daniel.lichtenecker@tum.de

P. Eichmeir
philipp.eichmeir@fh-wels.at

W. Steiner
wolfgang.steiner@fh-wels.at

K. Nachbagauer
karin.nachbagauer@fh-wels.at

1 Faculty of Engineering and Environmental Sciences, University of Applied Sciences Upper Austria, Stelzhamerstraße 23, 4600 Wels, Austria

2 Institute of Mechanics and Mechatronics, TU Wien, Getreidemarkt 9/E325, Vienna 1060, Austria

3 TUM School of Engineering and Design, Department of Mechanical Engineering, Chair of Applied Mechanics, Munich Institute of Robotics and Machine Intelligence (MIRMI), Technical University of Munich, Munich, Germany

4 Institute of Mechanics and Mechatronics, Division of Process Control and Automation, TU Wien, Getreidemarkt 9/E325, Vienna 1060, Austria

5 Institute for Advanced Study, Technical University of Munich, Lichtenbergstraße 2a, 85748 Garching, Germany