Contract-based Predictive Control for Modularity in Hierarchical Systems

Baethge, Tobias; Kogel, Markus; Di Cairano, Stefano; Findeisen, Rolf

TR2018-123 August 25, 2018

Abstract

Hierarchical control architectures pose challenges for control, as lower-level dynamics, such as from actuators, are often unknown or uncertain. If not considered correctly in the upper layers, requested and applied control signals will differ. Thus, the actual and the predicted plant behavior will not match, likely resulting in constraint violation and decreased control performance. We propose a model predictive control scheme in which the upper and lower levels - the controller and the actuator - agree on a "contract" that allows to bound the error due to neglected dynamics. The contract allows to guarantee a desired accuracy, enables modularity, and breaks complexity: Components can be exchanged, vendors do not need to provide in-depth insights into the components' working principle, and complexity is reduced, as upper-level controllers do not need full model information of the lower level - the actuators. The approach allows to consider uncertain actuator dynamics with flexible, varying sampling times. We prove repeated feasibility and input-to-state stability and illustrate the scheme in an example for a hierarchical controller/plant cascade.

IFAC Conference on Nonlinear Model Predictive Control

Contract-based Predictive Control for Modularity in Hierarchical Systems

Tobias Bäthge * **Markus Kögel** * **Stefano Di Cairano** ** **Rolf Findeisen** *

* *Laboratory for Systems Theory and Automatic Control, Otto von Guericke University, Magdeburg, Germany.* {tobias.baethge, markus.koegel, rolf.findeisen}@ovgu.de

** *Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, USA.* dicairano@ieee.org

Abstract: Hierarchical control architectures pose challenges for control, as lower-level dynamics, such as from actuators, are often unknown or uncertain. If not considered correctly in the upper layers, requested and applied control signals will differ. Thus, the actual and the predicted plant behavior will not match, likely resulting in constraint violation and decreased control performance. We propose a model predictive control scheme in which the upper and lower levels—the controller and the actuator—agree on a "contract" that allows to bound the error due to neglected dynamics. The contract allows to guarantee a desired accuracy, enables modularity, and breaks complexity: Components can be exchanged, vendors do not need to provide in-depth insights into the components' working principle, and complexity is reduced, as upper-level controllers do not need full model information of the lower level—the actuators. The approach allows to consider uncertain actuator dynamics with flexible, varying sampling times. We prove repeated feasibility and input-to-state stability and illustrate the scheme in an example for a hierarchical controller/plant cascade.

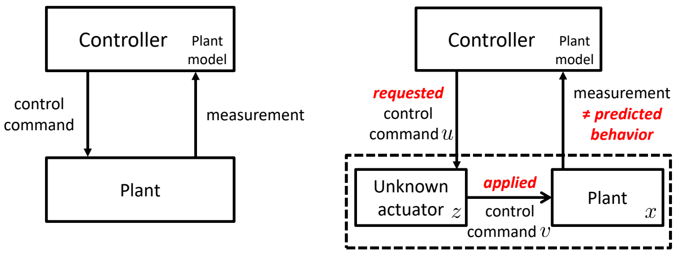
Keywords: Hierarchical control; predictive control; modularization; contracts; robustness.

1. INTRODUCTION

Control problems in modern applications like Automatic Driving or the "Internet of Things" are often highly interconnected, as they involve combinations of supervisory controllers and lower-level actuators, see e.g. Campbell et al. (2010); Di Cairano and Borrelli (2016); Lucia et al. (2016). Such hierarchical structures, spanning multiple levels, often consist of controllers, sensors, and actuators from different manufacturers. Combining components from multiple vendors often has an impact on the available knowledge of and the communication between these components: Exact dynamics of subsystems or neighboring components might be unknown, as companies want to protect proprietary knowledge, or might change when replacing a component with a model from a different vendor. Designing model-based controllers without such knowledge is difficult, especially in the case of Model Predictive Control (MPC). MPC schemes rely on sufficiently correct and detailed system models for the prediction of the future system behavior, see e.g. Maciejowski (2002); Rawlings et al. (2017); Mayne (2014)), to achieve good control performance.

We consider the case of unknown actuator dynamics, as shown in Fig. 1, to outline the appearing challenges and

Major parts of this work where developed during a research stay of TB and RF at MERL. TB, MK, and RF are members of the International Max Planck Research School for Advanced Methods in Process and Systems Engineering (IMPRS), Magdeburg, Germany.

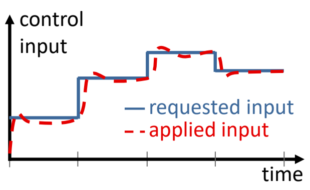


(a) Simple hierarchy with exact plant knowledge. (b) An unknown actuator introduces model-plant mismatch.

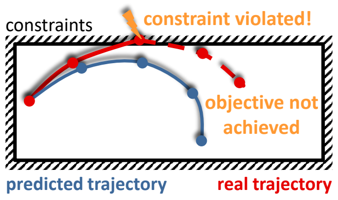
Fig. 1. Nominal plant structure and hierarchical structure including an unknown actuator.

provide an MPC strategy that overcomes these, enables modularity, allows privacy between the components, and breaks complexity. The unknown or uncertain dynamics can "slow down", delay, or modify the requested control input, leading to a real input that is different to the one that the controller commanded. As a consequence, the desired optimal behavior will not be reached and constraints might be violated, cf. Fig. 2. Neglecting this mismatch can result in increased conservatism, higher energy consumption, or even instability.

We propose an MPC scheme that takes the additional dynamics—without in-depth knowledge—into account as an additional uncertainty. The uncertainty is bounded in form of a suitable maximum error or an accuracy



(a) Requested (solid, blue line) and applied input, due to the additional actuator dynamics (dashed, red line).



(b) Degraded controller performance: Accuracy is lost, constraints can be violated, and control energy might be increased.

Fig. 2. Influence of unknown actuator dynamics on controller performance and constraint satisfaction.

around a "nominal" solution, requested by the controller, and for the actuator to guarantee. Achieving the desired accuracy error despite the actuator dynamics demands either a reduced range of feasible control inputs or control input changes, or an increased or decreased control input change frequency. The actuator, as an "active" component, provides this information to the controller in form of a "contract", by calculating an upper bound of the resulting mismatch between the nominal and real dynamics.

The agreement of an accuracy and corresponding input constraints forms a contract between the controller and the actuator. This allows to construct a modular control structure: The internal behavior of a control loop component can change, e.g. when an actuator is exchanged for a different type, model, or brand, as long as the agreed-on contract is enforced. Furthermore, the overall complexity is reduced, as the upper level does not need a complete model of the lower-level actuator. It also enables hiding proprietary information, which vendors often demand.

MPC schemes that exploit the idea of contracts for complex systems exist, cf. Lucia et al. (2015) and the references therein. We focus here on the hierarchical case, i.e., we consider contracts for accuracy and input constraints between components on different hierarchy levels. Furthermore, our contracts are static, i.e., they do not involve a variation in time. In comparison to Di Cairano et al. (2018), we focus on a discrete-time formulation, with the flexibility of different sampling times of controller and actuator. Furthermore, we use a robust, tube-based MPC scheme that allows an expansion towards additional model uncertainty. Summarizing, the contribution of this work is an MPC scheme that

- allows taking into account additional unknown and uncertain actuator dynamics, by agreeing on a contract, consisting of accuracy bounds and input (rate) constraints,
- that guarantees constraint satisfaction, employing a tube-based robust MPC controller, and that is formulated
- in a discrete-time setting that allows for different sampling times for actuator and controller.

Such an approach of "modularity in hierarchy", where model complexity is hidden or approximated by artificial uncertainties, can be compared and further expanded to exploiting the granularity (see e.g. Bäthge et al. (2016)) of different system models over the prediction, where different model complexities are captured by an extra uncertainty. Similarities also exist to model reduction approaches and decentralized MPC without communication, e.g. Kögel and Findeisen (2015, 2018).

The remainder of this paper is structured as follows: The general framework is introduced in Section 2. Section 3 presents an approach to bound the error that is introduced by the additional actuator dynamics in the control loop. The design of a robust MPC scheme that exploits the bound in form of a contract is presented in Section 4. A simulation example is shown in Section 5. Section 6 closes with a summary and suggestions for future work.

We use standard notation: For two sets $\mathbb{A}$, $\mathbb{B}$ and a matrix M, $\mathbb{A} \oplus \mathbb{B}$, $\mathbb{A} \ominus \mathbb{B}$, $M\mathbb{A}$, $\times$ denote the Minkowski sum, the Minkowski difference, the set multiplication, and the Cartesian product, respectively, see e.g. Blanchini and Miani (2015). A set $\mathbb{S}$ is called robust positive invariant (RPI) under $s_{k+1} = Ss_k + e_k$, $e_k \in \mathbb{E}$, where $\mathbb{E}$ is a convex, compact set with $0 \in \mathbb{E}$, if $\forall s_k \in \mathbb{S}$, $e_k \in \mathbb{E}$: $s_{k+1} \in \mathbb{S}$. $a_{i|j}$ denotes the value of a at time t_i, calculated at time t_j.

2. HIERARCHICAL CONTROL ARCHITECTURE

We consider a hierarchical control architecture in which an upper-level controller interacts with a plant and a lower-level actuator. The plant dynamics, which the upper-level controller knows and can use, cf. Fig. 1, are given by

$$\dot{x}(t) = Ax(t) + Bv(t), \qquad y(t) = Cx(t) + Dv(t), \tag{1}$$

where $x(t) \in \mathbb{R}^{n_x}$ denotes the states, $v(t) \in \mathbb{R}^{n_u}$ the inputs applied by the actuator, and $y(t) \in \mathbb{R}^{n_y}$ the plant outputs. The plant states, inputs, and outputs need to fulfill state, input, and output constraints,

$$x(t) \in \mathbb{X}, \qquad v(t) \in \mathbb{U}, \qquad \text{and } y(t) \in \mathbb{Y}, \tag{2}$$

respectively. We assume that $\mathbb{X}$, $\mathbb{U}$, and $\mathbb{Y}$ are compact, convex polytopes containing the origin in their interior.

The upper-level controller requests a command signal $u(t) \in \mathbb{U}$ from the actuator, which results in the applied input $v(t)$. Ideally,

$$v(t) = u(t), \tag{3}$$

i.e., the requested and the applied input signals match. However, in reality, the actuator introduces additional dynamics, which we consider to be of the form

$$\dot{z}(t) = A_{\mathrm{a}}z(t) + B_{\mathrm{a}}u(t), \qquad v(t) = C_{\mathrm{a}}z(t). \tag{4}$$

Here, $z(t) \in \mathbb{R}^{n_z}$ denotes the actuator states, $u(t) \in \mathbb{R}^{n_u}$ the inputs commanded to the actuator, and $v(t) \in \mathbb{R}^{n_u}$ the actuator outputs that are applied to the plant as inputs. We assume that the actuator satisfies:

Assumption 1 (Stable actuator dynamics): The actuator dynamics (2) are asymptotically stable, i.e., A_{a} is Hurwitz.

Assumption 2 (Unity gain of the actuator dynamics): The actuator has a steady state unity gain: A constant $u(t)$ implies $v(t) \to u(t)$ and $\forall u^{\mathrm{s}} : 0 = A_{\mathrm{a}}z^{\mathrm{s}} + B_{\mathrm{a}}u^{\mathrm{s}}$, $u^{\mathrm{s}} = C_{\mathrm{a}}z^{\mathrm{s}}$.

Note that Assumption 2 can be satisfied easily by a suitable choice/scaling of the inputs $u(t)$.

Remark 3 (Actuator constraints): For simplicity and as we assume a well-behaving and stable actuator, we only consider input constraints $\mathbb{U}$ for the actuator, but no actuator state constraints.

As outlined, the exact lower-level actuator dynamics might be unknown or unavailable, e.g. for proprietary or privacy reasons. Thus, the upper-level controller has no knowledge of the actuator model (2). However, it can negotiate a "contract" with the actuator during the design phase or at specific times during the operation phase, which defines allowable input changes and input limits to achieve a desired actuator-plant error.

Given this setup, we want to solve the following problem:

Problem 4 (Modular contract-based controller design): Design an upper-level controller that robustly stabilizes the lower-level plant (2) *and achieves constraint satisfaction despite limited knowledge of the actuator dynamics.*

To tackle this problem, we suggest that the controller treats the unknown actuator dynamics as an additional uncertainty that is directly considered in the prediction. The bounding set for the uncertainty, the accuracy, together with the necessary constraints on the control input form an actuator-controller contract. To calculate these sets, we derive bounds for the error that is introduced by the additional dynamics.

Note that, for clarity of presentation, we focus on a single actuator-controller configuration. Expansion to the multiple-actuator case is easily possible.

3. BOUNDING THE ACTUATOR ERROR

The contract between controller and actuator is based on an upper bound for the error that the actuator dynamics (2) cause in comparison to the ideal actuator dynamics (3). We assume that the upper-level controller is a sampled-data controller based on a discrete-time model, while the system itself operates in continuous time. To this end, we introduce discrete-time formulations for the combined systems, which are then used to obtain a bound on the resulting error. By applying standard discretization techniques, cf. Ogata (1995), using a sampling time $T > 0$ and a zero-order hold $u(t) = u(t_k)$, for $t \in [kT, kT + T)$, $k = 0, 1, \ldots$, one obtains the following discrete-time models:

Model of the plant with ideal actuator: The ideal system, consisting of the plant dynamics (2) and the ideal actuator dynamics (3), is given by

$$\hat{x}(t_{k+1}) = \hat{F}\hat{x}(t_k) + \hat{G}u(t_k) \tag{5a}$$

$$\hat{y}(t_k) = C\hat{x}(t_k) + Du(t_k). \tag{5b}$$

Here, $\hat{\cdot}$ denotes the variables of the combination of plant and ideal actuator. This model, expanded by an error term, forms the basis for the upper-level controller.

Model of the plant and the actuator: The real system, combining the plant dynamics (2) and the actuator dynamics (2), is given by

$$\dot{\xi}(t) = \begin{pmatrix} A & BC_\mathrm{a} \\ 0 & A_\mathrm{a} \end{pmatrix} \xi(t) + \begin{pmatrix} 0 \\ B_\mathrm{a} \end{pmatrix} u(t), \quad \xi = \begin{pmatrix} x \\ z \end{pmatrix}. \tag{6}$$

The corresponding discrete-time system, using the sampling time T, is given by

$$\xi(t_{k+1}) = \begin{pmatrix} F & F_\mathrm{c} \\ 0 & F_\mathrm{a} \end{pmatrix} \xi(t_k) + \begin{pmatrix} G_x \\ G_z \end{pmatrix} u(t_k) \tag{7a}$$

$$y(t_k) = (C\ 0)\,\xi(t_k) + Du(t_k). \tag{7b}$$

As a consequence of the made assumptions, we furthermore have:

Proposition 5 (Basic properties of the discretized systems): The matrices $\hat{F}$ and F satisfy $\hat{F} = F = \mathrm{e}^{AT}$. If Assumption 1 holds, then $F_\mathrm{a} = \mathrm{e}^{A_\mathrm{a}T}$ is Schur stable, there exists an RPI set $\mathbb{Z}$ for the actuator dynamics, such that

$$\mathbb{Z} \subseteq F_\mathrm{a}\mathbb{Z} \oplus G_z\mathbb{U}, \tag{8}$$

and the steady state z_∞ of the actuator for a given steady state input u_∞ satisfies

$$z_\infty = (I - F_\mathrm{a})^{-1} G_z u_\infty. \tag{9}$$

If Assumptions 1 and 2 hold, then

$$\Delta G = G_x - \hat{G} = -F_\mathrm{c}(I - F_\mathrm{a})^{-1} G_z, \tag{10}$$

as, from Assumption 2, any constant $u(t) = u_\mathrm{c}$ gives $v(t) = u(t) = u_\mathrm{c}$ and $Bv(t) = Bu(t) = Bu_\mathrm{c}$ (compare linear dynamics of normal and augmented system). Then, $\hat{G}u_\mathrm{c} = G_x u_\mathrm{c} + F_\mathrm{c}(I - F_\mathrm{a})^{-1} G_z u_\mathrm{c}$.

Remark 6 (Flexible sampling time): In principle, there is no need to use a constant input signal for the full sampling time T in the actuator. One could use a faster sampling time, e.g. $h = \frac{T}{H}$, $H \in \mathbb{N}$, allowing a faster actuator controller of the form

$$u^\mathrm{c}(t) = \begin{cases} K_1 \begin{pmatrix} z(t_k) \\ u(t_k) \end{pmatrix}, & t \in [t_k, t_k + h) \\ K_2 \begin{pmatrix} z(t_k + h) \\ u(t_k + h) \end{pmatrix}, & t \in [t_k + h, t_k + 2h) \\ \vdots & . \end{cases} \tag{11}$$

Here, u^c is the command sent to the plant in this case, which can be reformulated in a similar form as (7).

The controller does not know the actuator dynamics (2). Thus, the predicted state at the next time instant, $\hat{x}(t_{k+1})$, will in general not match the real state $x(t_{k+1})$, even if $\hat{x}(t_k) = x(t_k)$. To account for this mismatch, we introduce an artificial disturbance $w(t_k)$ that will capture the error. We aim to find a set $\mathbb{W}$ such that

$$\begin{aligned} \exists w(t_k) \in \mathbb{W} : x(t_{k+1}) &= Fx(t_k) + F_\mathrm{c}z(t_k) + G_x u(t_k) \\ &= \underbrace{\hat{F}x(t_k) + \hat{G}u(t_k)}_{=\hat{x}(t_{k+1})} + w(t_k). \end{aligned} \tag{12}$$

In other words, the set-based dynamics

$$\tilde{x}(t_{k+1}) = \hat{F}\tilde{x}(t_k) + \hat{G}u(t_k) + w(t_k), \quad w(t_k) \in \mathbb{W}, \tag{13}$$

outer-bound the behavior of the real system.

We first establish a way to find such a set $\mathbb{W}$ and that the error approaches zero as the input goes to zero:

Proposition 7 (Bounding the actuator error): Under Assumption 1 and if $z(t_0) \in \mathbb{Z}$, where $\mathbb{Z}$ satisfies (8), *a $\mathbb{W}$ such that* (12) *holds is given by*

$$\mathbb{W} = F_\mathrm{c}\mathbb{Z} \oplus \Delta G\mathbb{U}. \tag{14}$$

Furthermore, if $u(t_k) \to 0$, then $w(t_k) \to 0$.

Proof. Exploiting equation (12) and Proposition 5, it is clear that

$$w(t_k) = x(t_{k+1}) - \hat{x}(t_{k+1}) = F_\mathrm{c}z(t_k) + \Delta G u(t_k).$$

This yields equation (14) using the bounds on $z(t_k)$ and $u(t_k)$. Furthermore, note that F_a is Schur stable. Thus, $u(t_k) \to 0$ in (7) implies $z(t_k) \to 0$, thus, $F_\mathrm{c}z(t_k) + \Delta G u(t_k) \to 0$. □

Therefore, the error w caused by the actuator vanishes for small inputs u. In general, however, the error caused by the actuator can be large.

To further decrease the actuator uncertainties, it is possible to restrict the rate of input changes,

$$\Delta u(t_k) = u(t_k) - u(t_{k-1}) \in \Delta\mathbb{U}, \tag{15}$$

where $\Delta\mathbb{U}$ is a compact, convex polytope containing the origin.

Considering input change constraints $\Delta\mathbb{U}$ allows to derive results similar to Proposition 7. To do so, we define the difference between actuator state steady state at t_i and the applied input $u(t_{i-1})$:

$$\Delta z(t_i) = z(t_i) - (I - F_{\mathrm{a}})^{-1} G_z u(t_{i-1}). \tag{16}$$

This enables the establishment of the following:

Proposition 8 (Error in Δu formulation): Let Assumptions 1 and 2 and $\Delta z(t_0) \in \Delta\mathbb{Z}$ hold, where $\Delta\mathbb{Z}$ satisfies

$$\Delta\mathbb{Z} \subseteq F_{\mathrm{a}}\Delta\mathbb{Z} \oplus -F_{\mathrm{a}}(I - F_{\mathrm{a}})^{-1} G_z \Delta\mathbb{U}. \tag{17}$$

If $\Delta u(t_i) \in \Delta\mathbb{U}$, $i = 0, \ldots, k$, then a $\mathbb{W}$ leading to satisfaction of (12) *is given by*

$$\mathbb{W} = F_{\mathrm{c}}\Delta\mathbb{Z} \oplus \Delta G \Delta\mathbb{U}. \tag{18}$$

Proof. Proposition 5 and (16) yield

$$\begin{aligned} \Delta G u(t_{i-1}) &= -F_{\mathrm{c}}(I - F_{\mathrm{a}})^{-1} G_z u(t_{i-1}) \\ &= F_{\mathrm{c}}(\Delta z(t_i) - z(t_i)). \end{aligned} \tag{19}$$

Thus, we obtain from (12) that

$$\begin{aligned} w(t_i) &= F_{\mathrm{c}} z(t_i) + \Delta G(u(t_{i-1}) + \Delta u(t_i)) \\ &= F_{\mathrm{c}}\Delta z(t_i) + \Delta G \Delta u(t_i), \end{aligned} \tag{20}$$

which establishes (18) using the bounds on $\Delta z(t_i)$ and $\Delta u(t_i)$.

Using (7), (22), and (16), we obtain

$$\begin{aligned} \Delta z(t_{i+1}) &= F_{\mathrm{a}} z(t_i) + G_z u(t_i) - (I - F_{\mathrm{a}})^{-1} G_z u(t_i) \\ &= F_{\mathrm{a}} z(t_i) - F_{\mathrm{a}}(I - F_{\mathrm{a}})^{-1} G_z u(t_i) \\ &= F_{\mathrm{a}}\Delta z(t_i) - F_{\mathrm{a}}(I - F_{\mathrm{a}})^{-1} G_z \Delta u(t_i), \end{aligned} \tag{21}$$

leading to (17). □

Thus, to achieve the desired actuator accuracy error, we need to enforce input and input rate constraints in the upper-level controller. We do so using an MPC scheme in input-change formulation, as often done in MPC, see e.g. Maciejowski (2002): The input $u(t_k)$ can be written as a sum of input *changes* and the input of the actuator at the time shortly before the initial time $t = 0$, i.e.,

$$u(t_k) = u(t_{-1}) + \sum_{i=0}^{k} \Delta u(t_i). \tag{22}$$

Here, $u(t_{-1}) = \lim_{t \to 0^-} z(t)$ is the actuator state just before $t = 0$.

Remark 9 (Uncertain actuator dynamics): We assume that the dynamics to establish the contract in the actuator are exactly known, which in practice, however, is often not the case. A straightforward extension to cover "multiplicative uncertainties" is to assume that the matrices F_{a}, F_{c}, G_x, and G_z are given by

$$F_{\mathrm{a}} = \sum_{i=1}^{V} \lambda_i {F_{\mathrm{a}}}^i, \qquad F_{\mathrm{c}} = \sum_{i=1}^{V} \lambda_i {F_{\mathrm{c}}}^i, \tag{23a}$$

$$G_x = \sum_{i=1}^{V} \lambda_i {G_x}^i, \qquad G_z = \sum_{i=1}^{V} \lambda_i {G_z}^i, \tag{23b}$$

where $\lambda_i \geq 0$ and $\sum_{i=1}^{V} \lambda_i = 1$. ${F_{\mathrm{a}}}^i$, ${F_{\mathrm{c}}}^i$, ${G_x}^i$, and ${G_z}^i$ form the corners defining the uncertainty.

It is easy to show that $\mathbb{W} \subseteq \mathbb{W}^i$, $i = 1, \ldots, V$, where $\mathbb{W}^i$ are the sets obtained from Proposition 7 (or Proposition 8) for the matrices ${F_{\mathrm{a}}}^i$, ${F_{\mathrm{c}}}^i$, ${G_x}^i$, and ${G_z}^i$. This allows to determine a bound on $w(t_k)$ such that equation (12) *holds for such multiplicative uncertainties* (23).

As an immediate consequence, a bounding set $\mathbb{W}$ can be obtained from (14). $\mathbb{W}$ then needs to be the desired accuracy bound $\mathbb{W}_{\mathrm{request}}$ that the upper-level controller wants to be guaranteed. During the contract negotiation, the actuator obtains a request $\mathbb{W}_{\mathrm{request}}$ from the controller. This can be rejected or confirmed by suitable sets $\mathbb{U}$, $\Delta\mathbb{U}$, and $\mathbb{W}$. The controller can then certify whether it accepts the sets $\mathbb{U}$ and $\Delta\mathbb{U}$ or asks for a different $\mathbb{W}_{\mathrm{request}}$.

4. CONTROLLER DESIGN

We use a tube-based controller in the upper level to achieve constraint satisfaction, exploiting the controller-actuator accuracy-input contract. In tube-based MPC, the set of perturbed trajectories is bounded by a "tube" around the nominal trajectory, cf. Mayne et al. (2005, 2006).

To consider the input rate constraints, we embed the input rate in virtual dynamics,

$$\tilde{x} = \begin{pmatrix} \hat{x}_k \\ u_{k-1} \end{pmatrix}, \quad \tilde{y} = \begin{pmatrix} y_k \\ u_k - u_{k-1} \end{pmatrix} = \begin{pmatrix} C & 0 \\ 0 & -I \end{pmatrix} \tilde{x} + \begin{pmatrix} D \\ I \end{pmatrix} u,$$

for which we enforce the constraints $\tilde{\mathbb{Y}} = \mathbb{Y} \times \Delta\mathbb{U}$. Here, $\mathbb{Y}$ are general "output" constraints.

The resulting tube-based MPC scheme solves

$$\min_{\mathbf{x}_k, \mathbf{u}_k} J(\mathbf{x}_k, \mathbf{u}_k) \tag{24}$$

in a receding-horizon fashion, where

$$\mathbf{x}_k = \left\{x_{k|k}, \ldots, x_{k+N|k}\right\}, \tag{25a}$$

$$\mathbf{u}_k = \left\{u_{k|k}, \ldots, u_{k+N-1|k}\right\}, \tag{25b}$$

and $N \in \mathbb{N}$, $N > 1$, is the length of the prediction horizon, and where the cost function is given by

$$J(\mathbf{x}_k, \mathbf{u}_k) = x_{k+N|k}^{\mathsf{T}} P x_{k+N|k} + \sum_{i=k}^{k+N-1} x_i^{\mathsf{T}} Q x_i + u_i^{\mathsf{T}} R u_i \tag{25c}$$

with positive definite weighting matrices Q, P, and R. The dynamics and constraints are given by

$$x_{i+1|k} = \hat{F} x_{i|k} + \hat{G} u_{i|k},\ i = k, \ldots, k+N-1, \tag{25d}$$

$$x_{k|k} = \{x(t_k)\} \oplus \mathbb{D}, \tag{25e}$$

$$x_{i|k} \in \tilde{\mathbb{X}},\ i = k, \ldots, k+N-1, \tag{25f}$$

$$u_{i|k} \in \tilde{\mathbb{U}},\ i = k, \ldots, k+N-1, \tag{25g}$$

$$C x_{i|k} + D u_{i|k} \in \tilde{\mathbb{Y}},\ i = k, \ldots, k+N-1, \tag{25h}$$

$$x_{k+N|k} \in \mathbb{T}. \tag{25i}$$

$\mathbb{D}$ describes the constraint back-off that provides a safety margin, $\tilde{\mathbb{X}}$, $\tilde{\mathbb{U}}$, and $\tilde{\mathbb{Y}}$ are tightened constraint sets for states, inputs, and outputs, and $\mathbb{T}$ is the terminal set. Suitable choices for these sets that guarantee repeated feasibility and stability will be given below.

The input that is commanded to the actuator and thus the system is given by

$$u(t_k) = u^\star_{k|k} + K(x(t_k) - x^\star_{k|k}). \tag{26}$$

Here, $\cdot^\star$ denotes the optimal solution of the optimization problem (25) and K is the tube controller.

To achieve robust stability and constraint satisfaction, we require:

Assumption 10 (Conditions on sets $\mathbb{D}$, $\tilde{\mathbb{X}}$, $\tilde{\mathbb{U}}$, $\tilde{\mathbb{Y}}$, and $\mathbb{T}$): The constraint back-off set $\mathbb{D}$ satisfies

$$\mathbb{D} \supseteq (\hat{F} + \hat{G}K)\mathbb{D} \oplus \mathbb{W}. \tag{27}$$

The sets $\tilde{\mathbb{X}}$, $\tilde{\mathbb{U}}$, and $\tilde{\mathbb{Y}}$ contain a neighborhood of the origin and satisfy

$$\tilde{\mathbb{X}} = \mathbb{X} \ominus \mathbb{D}, \quad \tilde{\mathbb{U}} = \mathbb{U} \ominus K\mathbb{D}, \quad \tilde{\mathbb{Y}} = \mathbb{Y} \ominus (C + DK)\mathbb{D}. \tag{28}$$

$\mathbb{T}$ contains a neighborhood of its origin and satisfies

$$(\hat{F} + \hat{G}K)\mathbb{T} \subseteq \mathbb{T}, \ \mathbb{T} \subseteq \tilde{\mathbb{X}}, \ C\mathbb{T} + D\mathbb{T} \subseteq \tilde{\mathbb{Y}}, \ K\mathbb{T} \subseteq \tilde{\mathbb{U}}. \tag{29}$$

One can use tailored toolboxes like MPT3 (Herceg et al., 2013) and PnPMPC (Riverso et al., 2013) to compute these sets.

Assumption 11 (Conditions on the tube control gain): The control gain K is such that the control laws

$$u(t_k) = K\hat{x}(t_k), \qquad u(t_k) = Kx(t_k) \tag{30}$$

asymptotically stabilize systems (5) and (7), respectively.

Assumption 11 appears in similar fashion in works related to decentralized MPC. The control gain K can be calculated using a linear-quadratic regulator (LQR) design.

Provided that these assumptions hold, we can establish recursive feasibility:

Theorem 12 (Recursive feasibility): Let Assumptions 1, 10, and 11 and $z(t_0) \in \mathbb{Z}$ hold, where $\mathbb{Z}$ satisfies (8). If input rate constraints are used, let $\Delta z(t_0) \in \Delta\mathbb{Z}$ hold, where $\Delta\mathbb{Z}$ satisfies (17). Then, the closed-loop system (7), (26) is recursively feasible, i.e., if (25) is feasible at t_0, then for any $k \geq 0$, (25) is feasible and the constraints are satisfied: $u(t_k) \in \mathbb{U}$, $x(t_k) \in \mathbb{X}$, and $y(t_k) \in \mathbb{Y}$.

Proof. For clarity and due to space limitations, we only sketch the proof. The conditions of Proposition 8 are satisfied. Thus, the closed-loop system states satisfy (13). Following standard tube-based MPC ideas, we can guarantee recursive feasibility and constraint satisfaction robustly w.r.t. the artificial disturbance w. □

If the sets $\mathbb{D}$, $\tilde{\mathbb{X}}$, $\tilde{\mathbb{U}}$, $\tilde{\mathbb{Y}}$, and $\mathbb{T}$ are convex, closed polytopes, then (25) is a convex quadratic program, which can be solved efficiently, see e.g. Boyd and Vandenberghe (2004); Rawlings et al. (2017); Lucia et al. (2016). Alternatively, one might choose $\mathbb{D}$, $\tilde{\mathbb{X}}$, $\tilde{\mathbb{U}}$, $\tilde{\mathbb{Y}}$, and $\mathbb{T}$ as ellipsoids, in which case (25) is a convex quadratically-constained quadratic program, for which, by now, efficient, tailored solution approaches exist, as well.

To establish stability, we need further assumptions on the terminal penalty in the cost function:

Assumption 13 (Terminal penalty): The terminal penalty P in the cost function (25c) satisfies

$$P = (\hat{F} + \hat{G}K)^\mathsf{T} P (\hat{F} + \hat{G}K) + Q + K^\mathsf{T} R K. \tag{31}$$

Using this and Theorem 12, we can establish stability:

Theorem 14 (Asymptotic stability): Let Assumptions 1, 10, 11, and 13 and $z(t_0) \in \mathbb{Z}$ hold, where $\mathbb{Z}$ satisfies (8). If (25) is feasible at t_0, then the closed-loop system (7), (26) is asymptotically stable.

Proof. *(Sketch).* Theorem 12 guarantees recursive feasibility and constraint satisfaction. Moreover, with the assumptions made for $x_{0|0} = x(t_0)$, the LQR controller is locally admissible. Together with the fact that all constraint sets in the optimization problem (25) are compact, $\exists c_1 > 0$ s.t. $\|x^\star_{0|0}\| \leq c_1\|x(t_0)\|$ and $\|u^\star_{0|0}\| \leq c_1\|x(t_0)\|$.

This allows, similar to standard tube-based MPC, to show that the nominal state $x^\star_{k|k}$ is asymptotically stable.

Considering the dynamics of the plant and actuator, we can furthermore derive that

$$\Delta\xi(t_k) = \xi(t_k) - \begin{pmatrix} x^\star_{k|k} \\ 0 \end{pmatrix} \tag{32a}$$

$$\Delta\xi(t_{k+1}) = \tilde{F}\Delta\xi(t_k) + \tilde{G}u^\star_{k|k} + \begin{pmatrix} \Delta x^\star \\ 0 \end{pmatrix}, \tag{32b}$$

where

$$\Delta x^\star = \hat{F}x^\star_{k|k} + \hat{G}u^\star_{k|k} - x^\star_{k+1|k+1},$$
$$\tilde{F} = \begin{pmatrix} F & F_c \\ 0 & F_a \end{pmatrix} + \begin{pmatrix} G_x \\ G_z \end{pmatrix} (K \ \ 0), \qquad \tilde{G} = \begin{pmatrix} G_x - \hat{G} \\ G_z \end{pmatrix},$$

and where $\tilde{F}$ is Schur stable. Note that the $\Delta\xi$ dynamics are input-to-state stable (ISS) in terms of the nominal state and input, and $\exists c_2 > 0$ s.t. $\|\Delta\xi(t_0)\| \leq c_2 \left\| \left(x(t_0)^\mathsf{T} \ z(t_0)^\mathsf{T} \right)^\mathsf{T} \right\|$, which implies that $\Delta\xi$ is asymptotically stable. Thus, the overall system is stable. □

Summarizing, we have established repeated feasibility and stability of the complete controller-actuator-plant system.

5. SIMULATION EXAMPLE

To illustrate the approach, we consider a double integrator as the plant, given by

$$\dot{x}(t) = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} x(t) + \begin{pmatrix} 0 \\ 1 \end{pmatrix} v(t), \quad y(t) = (1 \ \ 0)\, x(t). \tag{33}$$

The actuator is described by the first-order system

$$\dot{z}(t) = -20z(t) + 20u(t), \qquad v(t) = z(t). \tag{34}$$

Corresponding discrete-time formulations (5) and (7) are obtained using a sampling time $T = 0.3$, and states, control inputs, and control input changes are box-constrained:

$$\mathbb{X} = \left[\begin{pmatrix} -10 \\ -10 \end{pmatrix}, \begin{pmatrix} 10 \\ 10 \end{pmatrix} \right], \quad \mathbb{U} = [-2, 2], \quad \Delta\mathbb{U} = [-0.4, 0.4].$$

The weighting matrices in the cost function are chosen as $Q = \left(\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right)$ and $R = 1000$. They are used to calculate the tube controller gain K as the corresponding LQR gain and terminal penalty P to satisfy Assumption 13.

We consider that the MPC controller should bring the plant from the initial state $x(t_0) = (-7.7 \ \ 5.7)^\mathsf{T}$ to the origin. The initial state for the actuator is $z(t_0) = 0$.

The simulation results are obtained from solving optimization problem (25) in a receding-horizon manner in MATLAB. The problem is formulated with YALMIP (Löfberg, 2004). MPT3 (Herceg et al., 2013) and PnPMPC (Riverso et al., 2013) are used to determine the required sets.

Fig. 3 shows the trajectories of plant and actuator states, as well as control inputs. All constraints are satisfied and the plant is successfully stabilized at the origin. The feasible region for the plant states is shown in Fig. 4.

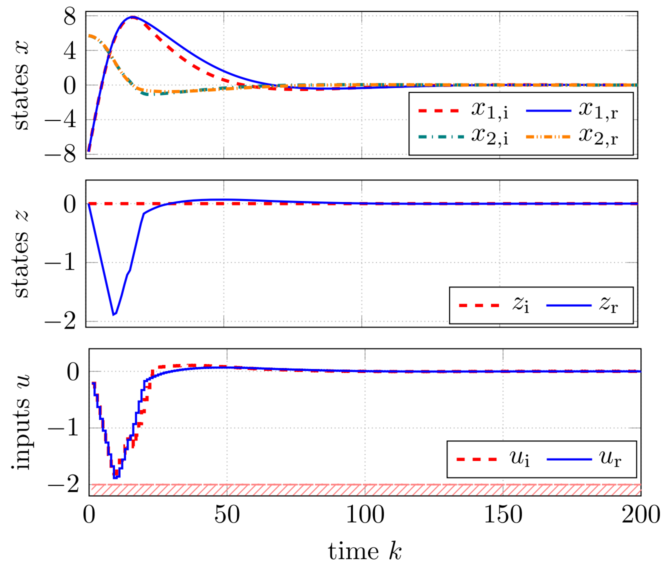


Fig. 3. Simulation results for the closed loop, consisting of the plant (5), the designed controller, and either the "ideal" actuator (subscript i) or the real actuator (5) (subscript r). Top to bottom: Evolution of the plant states, evolution of the actuator states, and control input u for the ideal system and the real system.

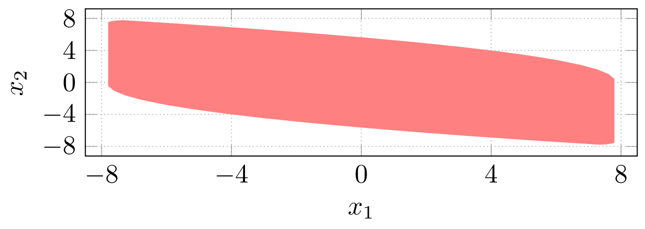


Fig. 4. Region of plant states x for which the optimization problem is feasible.

6. SUMMARY AND OUTLOOK

We considered the problem of hierarchical control, where the upper-level controller does not have a detailed model of the lower level. As a special case, we presented a Model Predictive Control scheme that stabilizes a system consisting of known plant dynamics and an actuator, whose dynamics are unknown to the MPC controller. To achieve stability and constraint satisfaction, the controller and the actuator agree on "contract" during the design phase. This contract consists of a bound on the error resulting from the unknown intermediate actuator. To achieve the desired accuracies, the actuator provides bounds on the allowable inputs and input changes. First, we derived a robust positive invariant set for the error bound, in a discrete-time setting. With this bound, we setup a tube-based MPC scheme to guarantee constraint satisfaction and stability. The simulation example provided further insights into the control scheme.

With this, we offer a method to improve controller performance in settings where not all information is available, e.g. due to privacy, legal, or modularity reasons. This is often the case in industrial applications, where different manufacturers might consider internal actuator dynamics as proprietary knowledge. Furthermore, the approach allows constructing modular controller-actuator-plant cascades, in which actuators can be exchanged—e.g. for other models or vendors with different system dynamics—as long as they satisfy the same accuracy contract.

Future work and potential extensions will consider nonlinear dynamics for plant and actuator as well as various descriptions of uncertainty.

REFERENCES

Blanchini, F. and Miani, S. (2015). *Set-Theoretic Methods in Control.* Systems & Control: Foundations & Applications. Birkhäuser, Basel, 2nd edition.

Boyd, S.P. and Vandenberghe, L. (2004). *Convex Optimization.* Cambridge University Press, Cambridge.

Bäthge, T., Lucia, S., and Findeisen, R. (2016). Exploiting Models of Different Granularity in Robust Predictive Control. In *Proc. IEEE Conf. Dec. Contr.*, 2763–2768.

Campbell, M., Egerstedt, M., How, J.P., and Murray, R.M. (2010). Autonomous driving in urban environments: approaches, lessons and challenges. *Philosophical Trans. Royal Society A*, 368(1928), 4649–4672.

Di Cairano, S. and Borrelli, F. (2016). Reference tracking with guaranteed error bound for constrained linear systems. *IEEE Trans. Autom. Contr.*, 61(8), 2245–2250.

Di Cairano, S., Bäthge, T., and Findeisen, R. (2018). Modular MPC Design for Constrained Control of Plant-Actuator Cascades. Subm. to *IEEE Conf. Dec. Contr.*

Herceg, M., Kvasnica, M., Jones, C.N., and Morari, M. (2013). Multi-Parametric Toolbox 3.0. In *Proc. Euro. Contr. Conf.*, 502–510.

Kögel, M. and Findeisen, R. (2015). Robust output feedback model predictive control using reduced order models. In *Proc. Int. Symp. Adv. Contr. Chem. Process.*, 1009–1015.

Kögel, M. and Findeisen, R. (2018). Improved Robust Decentralized MPC. In *Proc. Euro. Contr. Conf.*, 312–318.

Lucia, S., Kögel, M., and Findeisen, R. (2015). Contract-based Predictive Control of Distributed Systems with Plug and Play Capabilities. In *Proc. IFAC Conf. Nonlinear Model Predictive Contr.*, 205–111.

Lucia, S., Kögel, M., Zometa, P., Quevedo, D.E., and Findeisen, R. (2016). Predictive control, embedded cyberphysical systems and systems of systems – A perspective. *Annu. Rev. Contr.*, 41, 193–207.

Löfberg, J. (2004). YALMIP : A Toolbox for Modeling and Optimization in MATLAB. In *Proc. IEEE Int. Symp. Comp. Aided Control Syst. Design (CACSD)*, 284–289.

Maciejowski, J.M. (2002). *Predictive Control with Constraints.* Prentice Hall, Harlow.

Mayne, D.Q. (2014). Model predictive control: Recent developments and future promise. *Automatica*, 50(12), 2967–2986.

Mayne, D.Q., Raković, S.V., Findeisen, R., and Allgöwer, F. (2006). Robust output feedback model predictive control of constrained linear systems. *Automatica*, 42(7), 1217–1222.

Mayne, D.Q., Seron, M.M., and Raković, S.V. (2005). Robust model predictive control of constrained linear systems with bounded disturbances. *Automatica*, 41, 219–224.

Ogata, K. (1995). *Discrete-time Control Systems*. Prentice Hall, Englewood Cliffs, NJ, 2nd edition.

Rawlings, J.B., Mayne, D.Q., and Diehl, M.M. (2017). *Model Predictive Control: Theory, Computation, and Design*. Nob Hill Publishing, Madison, WI, 2nd edition.

Riverso, S., Battocchio, A., and Ferrari-Trecate, G. (2013). PnPMPC toolbox. URL `http://sisdin.unipv.it/pnpmpc/pnpmpc.php`.