2019 American Control Conference (ACC)
Philadelphia, PA, USA, July 10-12, 2019

Modular Design for Constrained Control of Actuator-Plant Cascades

Stefano Di Cairano, Tobias Bäthge, Rolf Findeisen

***Abstract*— We consider layered control architectures where a constraint-enforcing upper layer is cascaded with a lower layer controlled actuator. As we aim for a design where each layer requires as little knowledge as possible of the other, the upper layer is based on a model that neglects the lower layer dynamics, and includes instead additive uncertainty. The uncertainty set is constructed and "declared" by the lower layer based only on constraints on the command "declared" by the upper layer. This results in a contract between upper layer and lower layer guaranteeing a bound on the prediction error if the command satisfies the declared constraints. The command and plant constraints are robustly enforced by model predictive control with a robust control invariant set. The stability properties are analyzed, and a case study of vehicle steering control is shown.**

I. Introduction

High performance mechatronic devices, such as satellites, (semi)autonomous vehicles and robots for flexible manufacturing, require complex control architectures to achieve their operating goals. Such architectures are composed of multiple layers that operate with different time scales and abstractions, i.e., models, of the plant. An example is an autonomous vehicle control architecture, which may be composed of three layers: (i) a path planner (PP) computes the vehicle trajectory based on the road and other vehicles; (ii) a vehicle controller (VC) tracks the trajectory by issuing commands, e.g., steering angle, wheel torque, brake force; (iii) an actuator controller (AC) layer regulates the actuators, e.g., steering and engine, to achieve the commands.

Structuring a control system into multiple independent layers, i.e., modular layered control design, has practical benefits. Components can be re-used, flexibility in maintenance and upgrading increases, and obfuscation ("privacy") is provided, which allows for independent development of sub-components. There are also technical benefits, such as operating different layers at different rates, with different plant abstractions, and different decision horizons: moving from top to bottom layers, the abstraction becomes less coarse and the control rate increases, while the decision horizon shortens. Instead, a monolithic controller would need to operate at the highest rate, with the finest abstraction, and the longest horizon, which is often impractical.

The overall behavior of a multi-layer architecture is largely dependent on how well the layers are integrated. Today, verifying and validating the correct integration gives rise to a fairly long and expensive trial-and-error procedure. An alternative is to design each layer to be robust to the effects of the neighbors, while using minimal information from other layers, to ensure fast integration and retaining of isolation (modularity) and obfuscation (independent development).

We consider the modular design of an upper layer, the plant controller, i.e., VC in the vehicle example, to be integrated with with a lower layer, the controlled actuator, i.e., AC in the example. The upper layer must enforce plant constraints and certain stability properties with only minimal information on the lower layer, during design time, i.e., of its dynamics, and execution, i.e., of its state.

The design is based on deriving error sets for the difference between the plant response with actual and ideal, i.e., instantaneous, lower layer dynamics. The upper layer considers the plant subject to additive uncertainty bounded in such sets. The sets depend on the commands issued by the upper layer so that, by imposing control constraints, their size can be modified. After selecting the command constraints and determining as a consequence the corresponding error sets, the upper layer can be designed as a model predictive control (MPC) enforcing constraints on the command and a robust control invariant (RCI) set [1], [2] for satisfying constraints despite ignoring the lower layer.

In the proposed design, the upper layer needs to "declare" only the command constraints, and the lower layer needs to "declare" only the corresponding error sets. Any other information, at design and during operation, remains private. The relation between command constraints and error sets provides an "assume-guarantee" condition, also called a "contract" [3]–[6], where if the upper layer enforces the declared command constraints, then the lower layer guarantees that the deviation from the instantaneous response is within the error set. The RCI set is also exploited to determine whether the design contract is acceptable, or the upper layer needs to modify the constraints thus modifying the error set.

Thus, our design is a hierarchical MPC architecture achieved with minimal shared information between layers without assuming a sharp time scale separation, which is recognized to be an open problem [7, Sec. 5.2].

In what follows, Section II describes the models of plant and actuator, and the control-oriented model, Section III describes the control design, Section IV analyzes closed-loop stability, Section V presents a case study related to the vehicle example and Section VI discusses future work.

Notation: $\mathbb{R}$, $\mathbb{R}_{0+}$, $\mathbb{R}_+$, are the sets of real, nonnegative real, positive real numbers, and similarly for integer numbers $\mathbb{Z}$. We denote intervals by, e.g., $\mathbb{Z}_{[a,b)} = \{z \in \mathbb{Z} : a \leq z < b\}$. $\mathcal{X} \oplus \mathcal{Y}$, $\mathcal{X} \sim \mathcal{Y}$, $\alpha\mathcal{X}$, $\alpha \in \mathbb{R}_+$, are set sum, difference, scaling, and $A \circ \mathcal{X}$ is the set image through the (linear) map

S. Di Cairano is with Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA. Email: dicairano@ieee.org.

T. Bäthge and R. Findeisen are with the Inst. for Automation Engineering, Otto von Guericke University Magdeburg, Germany. Email: {tobias.baethge, rolf.findeisen}@ovgu.de.

978-1-5386-7928-9/$31.00 ©2019 AACC

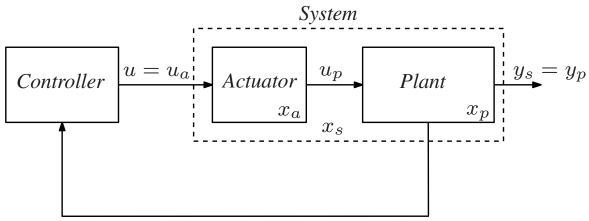

Fig. 1. Schematics of the actual system

A. For matrices, inequalities indicate (semi)definiteness, and $\bar{\lambda}(M)$, $\underline{\lambda}(M)$ are the largest and smallest eigenvalue, $[M]_i$, $[M]^j$, $[M]_i^j$ are the i^{th} row, the j^{th} column, and the element at i^{th} row and j^{th} column. For vectors, $(x, y) = [x' \ y']'$ is the "stacking", and inequalities and absolute value $|x|$ are intended componentwise.

For a vector, a signal, a system, $\|a\|_p$, $\|a(\cdot)\|_p$, $\|G(t)\|_p$ denote the p-norm, respectively; subscripts are dropped when clear from context/irrelevant. For a system/signal, $*$ is the time convolution. For signal $x(t)$, $t \in \mathbb{R}_{0+}$, the sampled signal $x(k)$, $k \in \mathbb{Z}_{0+}$, with sampling period T_s is $x(k) = x(kT_s)$, $x_{h|k}$ is the value of x predicted h steps ahead based on data at sample k, i.e., $x(k + h)$, and $x_{0|k} = x(k)$. Function $\alpha : \mathbb{R}_{0+} \to \mathbb{R}_{0+}$ is of class $\mathcal{K}$ if it is continuous, strictly increasing, $\alpha(0) = 0$; if in addition $\lim_{c\to\infty} \alpha(c) = \infty$, α is of class $\mathcal{K}_\infty$.

II. System Description and Control Model

We consider a system composed of plant and actuators, see Fig. 1. The plant is described by

$$\dot{x}_p(t) = A_p x_p(t) + B_p u_p(t) \quad (1a)$$
$$y_p(t) = C_p x_p(t) + D_p u_p(t), \quad (1b)$$

where $x_p \in \mathbb{R}^{n_p}$, $u_p \in \mathbb{R}^{m_p}$, $y_p \in \mathbb{R}^{p_p}$, are the plant state, input, and constrained output vectors, respectively. The plant (1) is subject to constraints

$$u_p \in \mathcal{U}_p, \qquad y_p \in \mathcal{Y}_p, \quad (2)$$

that should be satisfied by the upper layer controller, from now on simply the *controller*.

The controller command is actuated as the input to the plant by an actuator in closed-loop with a lower layer controller, that together from now on are simply called the *actuator*, and are described by

$$\dot{x}_a(t) = A_a x_a(t) + B_a u_a(t) \quad (3a)$$
$$y_a(t) = C_a x_a(t) + D_a u_a(t) \quad (3b)$$

where $x_a \in \mathbb{R}^{n_a}$, $u_a \in \mathbb{R}^{m_a}$, are the actuator state and input vectors, respectively, and the closed-loop actuator output is the plant input, i.e., $y_a = u_p \in \mathbb{R}^{m_p}$. We assume that (3) is asymptotically stable and has unitary dc-gain.

From the controller perspective, the full system (1), (3) is,

$$\dot{x}_s(t) = A_s x_s(t) + B_s u_s(t) \quad (4a)$$
$$y_s(t) = C_s x_s(t) + D_s u_s(t) \quad (4b)$$

where $x_s = (x_p, x_a)$, $u_s = u_a$, $y_s = y_p$, and $A_s = \left[\begin{smallmatrix} A_p & B_p C_a \\ 0 & A_a \end{smallmatrix}\right]$, $B_s = \left[\begin{smallmatrix} B_p D_a \\ B_a \end{smallmatrix}\right]$, $C_s = \left[\begin{smallmatrix} C_p & D_p C_a \end{smallmatrix}\right]$, $D_s = D_p D_a$.

We want to design a controller for (4) satisfying (2) pointwise in time with period T_s and achieving stability, in an appropriate sense, without detailed information on the actuator state, x_a, and dynamics (3). For designing such controller we only use a discrete-time model of (1),

$$x_m(k+1) = A_m x_m(k) + B_m u_m(k) \quad (5a)$$
$$y_m(k) = C_m x_m(k) + D_m u_m(k) \quad (5b)$$
$$u(k) \in \mathcal{U}, \ y_m(k) \in \mathcal{Y}, \quad (5c)$$

where x_m, $u_m = u$, y_m, are the predicted state, input, and constrained output vectors, with the same dimension of the corresponding in (1), $\mathcal{U} \subseteq \mathcal{U}_p$, $\mathcal{Y} \subseteq \mathcal{Y}_p$ are the constraints enforced by the controller that imply the plant constraints (2), the sampling period is T_s, $A_m = e^{A_p T_s}$, $B_m = \int_0^{T_s} e^{A_p(T_s - \tau)} B_p \mathrm{d}\tau$, $C_m = C_p$, $D_m = D_p$, and $u_a(t)$ is obtained by a zero order hold on $u(k)$. Thus, at sampling instants and in the ideal case, i.e., with infinitely fast actuator dynamics, $x_m = x_p$, $y_m = y_p = y$, $u_m = u_p = u$.

Ignoring the actuator dynamics (3) in the controller, i.e., using only the control-design model (5), leads in general to constraint violations in the actual system (4). To guarantee constraint satisfaction, we extend (5) by including the additive uncertainty vectors w_x, w_u,

$$x_m(k+1) = A_m x_m(k) + B_m u(k) + w_x(k) \quad (6a)$$
$$y_m(k) = C_m x_m(k) + D_m u(k) + D_m w_u(k) \quad (6b)$$
$$u(k) + w_u(k) \in \mathcal{U}, \ y(k) \in \mathcal{Y}. \quad (6c)$$

Here, w_x, w_u represent the errors due to neglecting the actuator dynamics. Sets bounding such errors are the only information needed from the actuator for the controller design based on (6). Since the actuator is driven by the command signal, the bounds on w_x, w_u are related to it,

$$w_i(k) \in \tilde{\mathcal{W}}_i(\{(u(h), \Delta u(h))\}_{h=0}^k), \ i \in \{x, u\}. \quad (7)$$

As shown in details in the next section, these bounds depend on the command step change $\Delta u(k) = u(k) - u(k-1)$.

By imposing command and command rate constraints, from (7) we obtain time-invariant disturbance sets

$$u(k) \in \mathcal{U}, \Delta u(k) \in \Delta\mathcal{U}, \forall k \in \mathbb{Z}_{0+} \implies w_i(k) \in \mathcal{W}_i, \forall k \in \mathbb{Z}_{0+} \quad (8)$$

for $i \in \{x, u\}$. At design time, the controller specifies bounds on the command and its change, and the actuator responds with the corresponding error sets $\mathcal{W}_x$, $\mathcal{W}_u$. If the controller enforces the bounds on the command and its step change, the error due to ignoring the actuator dynamics is bounded in the error sets by (8). Thus, (8) can be seen as a design contract between controller and actuator, and is exploited to ensure robustness against the ignored actuator dynamics.

III. Modular Controller Design

Definition 1: A set $\mathcal{S} \subseteq \mathbb{R}^n$ is robust control invariant (RCI) for $x(k+1) = f(x(k), u(k), w(k))$ with input $u \in \mathcal{U} \subseteq \mathbb{R}^m$ and disturbance $w \in \mathcal{W} \subseteq \mathbb{R}^d$, iff for all $x \in \mathcal{S}$

there exists $u \in \mathcal{U}$ such that $f(x,u,w) \in \mathcal{S}$, for all $w \in \mathcal{W}$. If $w = \{0\}$, $\mathcal{S}$ is simply said control invariant (CI). □

For autonomous systems, i.e., $u = 0$, the sets in Definition 1 with/without disturbances are called robust positive invariant (RPI) and positive invariant (PI), respectively. Furthermore, given the RCI $\mathcal{S} \subseteq \mathbb{R}^n$, the robust admissible input (RAI) set is $\mathcal{S}_u(x) = \{u \in \mathcal{U} : \ f(x,u,w) \in \mathcal{S}, \ \forall w \in \mathcal{W}\}$.

A. Error Induced by Neglecting Linear Actuators

Next, we characterize the prediction error due to ignoring the actuator dynamics, which justifies (7) and enables (8). We consider $D_a = 0$, for simplicity.

Let $\varepsilon = |x_m^+ - x_p^+|$ denote the prediction error vector between (5) and (4) after one sampling period for an input change Δu starting from input u, and actuator state x_a,

$$x_m^+ = e^{A_p T_s} x_m + \int_0^{T_s} e^{A_p(T_s-\tau)} B_p(u+\Delta u)\mathrm{d}\tau$$
$$x_p^+ = e^{A_p T_s} x_p + \int_0^{T_s} e^{A_p(T_s-\tau)} B_p C_a \Big(e^{A_a \tau} x_a + \int_0^{\tau} e^{A_a(\tau-\sigma)} B_a(u+\Delta u)\mathrm{d}\sigma \Big)\mathrm{d}\tau.$$

Considering that at the sampling instants $x_m = x_p$, we can decompose the error as $\varepsilon = \varepsilon_s + \varepsilon_c$, where the step input error ε_s is simply the difference between the response of (1) and (4) to step changes Δu,

$$\varepsilon_s = \left| \int_0^{T_s} e^{A_p(T_s-\tau)} B_p \Big(I - C_a \cdot \int_0^{\tau} e^{A_a(\tau-\sigma)} B_a \mathrm{d}\sigma \Big) \mathrm{d}\tau \Delta u \right| \leq M_s |\Delta u|, \tag{10}$$

where $M_s \in \mathbb{R}^{n_p \times m_p}$ has nonnegative elements.

The cumulative input error ε_c is due to the difference between the expected and actual input at each sampling instant due to the non-ideal response of the actuator to the sequence of previous commands and can be reformulated as,

$$\varepsilon_c = \left| \int_0^{T_s} e^{A_p(T_s-\tau)} B_p C_a e^{A_a \tau} \delta x_a^{(u)} \mathrm{d}\tau \right|, \tag{11}$$

where $\delta x_a^{(u)} = x_a - x_a^e(u)$ denotes the error between the current actuator state and the equilibrium state associated to u, $x_a^e(u)$, which is the expected state if the previous command was perfectly reached. Note that (11) is the free response of the full system (4) from $(0, \delta x_a^{(u)})$.

Next we obtain a bound on $\delta x_a^{(u)}$. To this end, we consider the discrete time representation of (3) with sampling period T_s, and an incremental input formulation,

$$x_a(k+1) = \bar{A}_a x_a(k) + \bar{B}_a v_a(k) + \bar{B}_a \Delta u(k) \tag{12a}$$
$$v_a(k+1) = v_a(k) + \Delta u(k) \tag{12b}$$
$$\tilde{y}_a(k) = x_a(k) - (I - \bar{A}_a)^{-1} \bar{B}_a v_a(k). \tag{12c}$$

Here, the output vector $\tilde{y}_z \in \mathbb{R}^{n_a}$ is the difference between the current state and the steady state associated to the past input $v_a(k) = u(k-1)$. Considering the ℓ_∞ gain for each input-output pair in (12), $\ell_\infty(i,j)$, and $[\upsilon]_j = \|[\Delta u(\cdot)]_j\|_\infty$,

$$\|[\delta x_a(\cdot)]_i\|_\infty = \sum_{j=1}^{m_p} \ell_\infty(i,j) \|[\Delta u(\cdot)]_j\|_\infty = [M_\delta \upsilon]_i, \tag{13}$$

where $[M_\delta]_i^j = \ell_\infty(i,j)$. Combining (13) and (11), we compute a matrix $M_c \in \mathbb{R}_{0+}^{n_p \times m_p}$ such that

$$\varepsilon_c \leq M_c \upsilon. \tag{14}$$

Similarly, we compute the maximum error on the input, i.e., the difference between u and u_a, by replacing (12c) with

$$\hat{y}_a(k) = C_a x_a(k) - C_a(I - \bar{A}_a)^{-1} \bar{B}_a v_a(k)$$

and repeating the steps to obtain $M_u \in \mathbb{R}_{0+}^{m_p \times m_p}$ such that

$$\varepsilon_u \leq M_u \upsilon. \tag{15}$$

From (10), (14), (15), the error due to unmodeled actuators depends on $\Delta u(\cdot)$. If the actuator can only achieve commands in $\mathcal{U}_a \subset \mathcal{U}$, there will be an additional error $\varepsilon_r(u) = |\delta x_{\mathcal{U}_a}(u)| = |B(u - u_p^e(u))|$, where $u_p^e(u)$ is the steady state input achieved for u, which results in $\mathcal{W}_i$, $i \in \{x, u\}$, depending on u as in (7). In practice, the actuator should declare $\mathcal{U}_a$ so that the controller enforces $\mathcal{U} \subseteq \mathcal{U}_a \cap \mathcal{U}_p$.

B. Robust Constrained Control Design

Let $\mathcal{U} \subseteq \mathcal{U}_p \cap \mathcal{U}_a$ and $\Delta\mathcal{U}$ be given. By the error bounds (10), (14), (15), the sets

$$\mathcal{W}_x = \{\tilde{x} : \ |\tilde{x}| \leq \sum_{j=1}^{m_p} ([M_c]^j + [M_s]^j) \max_{\Delta\mathcal{U}} |[\Delta u]_j|\}, \tag{16a}$$
$$\mathcal{W}_u = \{\tilde{u} : \ |\tilde{u}| \leq \sum_{j=1}^{m_p} [M_u]^j \max_{\Delta\mathcal{U}} |[\Delta u]_j|\}, \tag{16b}$$

bound the uncertainty due to neglecting the actuator such that, given $x_p(t)$, $u(t) = u(t - T_s) + \Delta u(t)$, $\Delta u(t) \in \Delta\mathcal{U}$, the plant and actuator dynamics satisfy

$$x_p(t+T_s) \in (A_m x_p + B_m u) \oplus \mathcal{W}_x,$$
$$y_p(t+T_s) \in (C_m x_p + B_m u) \oplus D_m \circ \mathcal{W}_u,$$
$$u_p(t+T_s) \in u(t) \oplus \mathcal{W}_u.$$

To ensure constraint satisfaction, we construct an RCI to be used in the controller. Since bounds (16) are constructed based on $\Delta\mathcal{U}$, we re-formulate (6) in incremental form by adding the command dynamics $u(k) = u(k-1) + \Delta u(k)$, with the previous command $v(k) = u(k-1)$ as state,

$$x(k+1) = Ax(k) + B\Delta u(k) + B_w w_x(k) \tag{17a}$$
$$y(k) = Cx(k) + D\Delta u(k) + D w_u(k) \tag{17b}$$
$$w_x \in \mathcal{W}_x, w_u \in \mathcal{W}_u, y \in \bar{\mathcal{Y}}, v + \Delta u \in \bar{\mathcal{U}}, \tag{17c}$$

where $x = (x_m, v)$, B_w is a vector of 0 and 1 that selects the states affected by the uncertainty, and $\bar{\mathcal{U}} = \mathcal{U} \sim \mathcal{W}_u$, $\bar{\mathcal{Y}} = \mathcal{Y} \sim D \circ \mathcal{W}_u$, to account for the error between the commanded input u and the actual plant input u_p.

Based on (17) and starting from $\mathcal{X}_0 = \{x : \ \exists \Delta u \in$

$\Delta\mathcal{U},\ v+\Delta u \in \bar{\mathcal{U}},\ Cx + D\Delta u \in \bar{\mathcal{Y}}\}$, the maximal RCI set $\mathcal{C}$ can be computed by iteratively constructing backward reachable sets until reaching a fixed point [1], which terminates with two possible outcomes. If $\mathcal{C} = \emptyset$, no full-dimensional RCI set exists. Otherwise, if $\mathcal{C} \neq \emptyset$, for all $x(k) \in \mathcal{C}$, there exists $\Delta u(k) \in \Delta\mathcal{U}$, such that $y(k) \in \bar{\mathcal{Y}}$ and $y_p(kT_s) \in \mathcal{Y}_p$, $u(k) \in \bar{\mathcal{U}}$ and $u_p(kT_s) \in \mathcal{U}_p$, $x(k+1) \in \mathcal{C}$, for all $w_x \in \mathcal{W}_x$, $w_u \in \mathcal{W}_u$.

From the RCI set $\mathcal{C}$, we construct the RAI set $\mathcal{C}_{\Delta u}(x)$. By its definition, any command step change in the RAI satisfies the constraints and maintains the state in the RCI, robustly. Hence, we formulate the optimal control problem,

$$\mathcal{V}(x(k)) = \min_{\Delta u^{(N)}(k)} \ x'_{N|k} P x_{N|k} + \sum_{h=0}^{N-1} x'_{h|k} Q x_{h|k} + \Delta u'_{h|k} R \Delta u_{h|k} \tag{18a}$$

$$\text{s.t. } x_{h+1|k} = A x_{h|k} + B\Delta u_{h|k} \tag{18b}$$

$$x_{0|k} = x(k) \tag{18c}$$

$$\Delta u_{h|k} \in \mathcal{C}_{\Delta u}(x_{h|k}) \tag{18d}$$

$$x_{N|k} \in \mathcal{X}_N \tag{18e}$$

where $\Delta u^{(N)}(k) = \{\Delta u_{0|k}, \ldots, \Delta u_{N-1|k}\}$, $\mathcal{X}_N \subseteq \mathcal{X}$, $P, R > 0$, $Q \geq 0$. Let $\Delta u^{(N)^*}(k) = \{\Delta u^*_{0|k}, \ldots, \Delta u^*_{N-1|k}\}$ denote the optimal solution at step k, the MPC law is

$$u(k) = \kappa_{\text{MPC}}(x(k)) = v(k) + \Delta u^*_{0|k}. \tag{19}$$

Next, we show that the closed-loop system satisfies the constraints. First, the following assumption ensures that the entire RCI set $\mathcal{C}$ is the feasible region of (18) and that the nominal MPC, where the full system and the prediction model in (18b) are equal, i.e., the actuator dynamics are infinitely fast, is asymptotically stable (AS) with the value function $\mathcal{V}$ in (18a) being a Lyapunov function (LF).

Assumption 1: P in (18a) and $\mathcal{X}_N$ in (18e) are such that for all $x \in \mathcal{X}_N$, there exists Δu such that

1) $\Delta u \in \mathcal{C}_{\Delta u}(x)$
2) $Ax + B\Delta u \in \mathcal{X}_N$
3) $x'Px - (Ax + B\Delta u)'P(Ax + B\Delta u) \leq x'Qx + \Delta u' R \Delta u$

Furthermore, $N \in \mathbb{R}_+$ in (18) is such that

4) for every $x_0 \in \mathcal{C}$, there exists a $\Delta u^{(N)}$ such that for all $h \in \mathbb{Z}_{[0,N-1]}$, $\Delta u_h \in \mathcal{C}_{\Delta u}(x_h)$ and $x_N \in \mathcal{X}_N$. □

Remark 1: Assumption 1 can be satisfied by designing the horizon, terminal cost and terminal set by the steps:

1) design terminal weight $P > 0$ and associated terminal controller $\kappa_N(x)$.
2) design $\mathcal{X}_N \subseteq \{x : \ \mathcal{C}_{\Delta u}(x) \in \Delta\mathcal{U}\}$ to be positive invariant for $Ax + B\kappa_N(x)$.
3) Select $N \in \mathbb{Z}_+$ such that N-steps backward reachable set of $\mathcal{X}_N$ for $x(k+1) = Ax(k) + Bu(k)$ subject to $\Delta u(k) \in \mathcal{C}_{\Delta u}(x(k))$ covers $\mathcal{C}$.

This design process is similar to [8], for instance. □

Theorem 1: Consider system (4), the contract (8), and the MPC (19) that solves (18). At time kT_s, let $x(k) \in \mathcal{C}$. Then, for every $\zeta T_s \geq kT_s$, (18) is feasible and (2) is satisfied.

Proof (Sketch): By invariance, if $x_{h|k} \in \mathcal{C}$, there exists $\Delta u_{h|k} \in \mathcal{C}_{\Delta u}$ such that $x_{h+1|k} \in \mathcal{C}$, $u_{h|k} \in \bar{\mathcal{U}}$, $y_{h|k} \in \bar{\mathcal{Y}}$ for every $w_i \in \mathcal{W}_i$, $i \in \{x, u\}$. By assumption, $x_{0|k} \in \mathcal{C}$, and by construction $\bar{\mathcal{U}} = \mathcal{U} \sim \mathcal{W}_u$, $\bar{\mathcal{Y}} = \mathcal{Y} \sim D \circ \mathcal{W}_u$, so that

$$x(k) \in \mathcal{C},\ w_i(\zeta) \in \mathcal{W}_i,\ i \in \{x,u\},\ \forall \zeta \geq k \implies x(\zeta) \in \mathcal{C},\ y_p(\zeta T_s) \in \mathcal{Y}_p,\ u_p(\zeta T_s) \in \mathcal{U}_p,\ \forall \zeta \geq k.$$

since $\mathcal{U} \subseteq \mathcal{U}_p$, $\mathcal{Y} \subseteq \mathcal{Y}_p$. Furthermore, according to (8)

$$\Delta u(\zeta) \in \Delta\mathcal{U},\ \forall \zeta \in \mathbb{Z}_{0+} \implies w_i(\zeta) \in \mathcal{W}_i,\ \forall \zeta \in \mathbb{Z}_{0+},$$

and hence

$$x(k) \in \mathcal{C},\ \Delta u(\zeta) \in \Delta\mathcal{U},\ \forall \zeta \geq k \implies x(\zeta) \in \mathcal{C},\ y_p(\zeta T_s) \in \mathcal{Y}_p,\ u_p(\zeta T_s) \in \mathcal{U}_p,\ \forall \zeta \geq k.$$

By the constraints enforced in (18), where (18d) is always feasible inside the RCI set, and (18e) is feasible from within the RCI set by the choice of N, we have that $\Delta u(\zeta) \in \Delta\mathcal{U},\ \forall \zeta \geq k$, and hence

$$x(k) \in \mathcal{C} \implies x(\zeta) \in \mathcal{C},\ y_p(\zeta T_s) \in \mathcal{Y}_p,\ u_p(\zeta T_s) \in \mathcal{U}_p,\ \forall \zeta \geq k.$$

■

Establishing (8) amounts to selecting $\Delta\mathcal{U}$ so that $\mathcal{C} \neq \emptyset$, i.e., the design is feasible. The only actuator information needed to compute $\mathcal{C}$ is the range $\mathcal{U}_a$ and the uncertainty sets (16). To construct the latter ones, the actuator only needs $\Delta\mathcal{U}$. This provides a modular design based on limited information, which hides the implementation of each module to the other and requires no actuator real-time information (i.e., $x_a(t)$) in the controller. Algorithm 1 sketches how to iteratively construct (8) by exchanging such information.

Algorithm 1 Constraint negotiation

1: Actuator declares $\mathcal{U}_a$.
2: Controller selects $\mathcal{U} \subseteq \mathcal{U}_p \cap \mathcal{U}_a$, $\overline{\Delta\mathcal{U}} \subset \mathbb{R}^m$, bounded.
3: Controller chooses $\Delta\mathcal{U} \subseteq \overline{\Delta\mathcal{U}}$.
4: Controller declares $\Delta\mathcal{U}$ to actuator.
5: Actuator determines the sets $\mathcal{W}_x$, $\mathcal{W}_u$ by (16)
6: Controller receives $\mathcal{W}_x$, $\mathcal{W}_u$, and constructs $\mathcal{C}$.
7: If $\mathcal{C} \neq \emptyset$ terminate. Otherwise, Controller selects $\overline{\Delta\mathcal{U}} \subset \Delta\mathcal{U}$ and goes to Step 3.

The $\overline{\Delta\mathcal{U}}$ update at Step 7 of Algorithm 1 generates a monotonically decreasing sequence of sets $\Delta\mathcal{U}$. Other methods may be applied, such as an increasing, or a "bisection-like" sequence. The termination condition at Step 7 of Algorithm 1, can also be evaluated on whether the obtained $\mathcal{C}$ includes a range of desired states. For instance, a set of equilibria $\mathcal{X}_e$ may be required to be feasible, i.e., $\mathcal{X}_e \subset \mathcal{C}$.

IV. Input to State Stability Properties

We recall some definitions and results on regional stability and LFs, see, e.g., [2, Appendix B].

Definition 2: Given $x(k+1) = f(x(k), w(k))$, $x \in \mathbb{R}^n$, $w \in \mathcal{W} \subseteq \mathbb{R}^d$, and a RPI set $\mathcal{S}$ for f, $0 \in \mathcal{S}$, a function $\mathcal{V} : \mathbb{R}^n \to \mathbb{R}_{0+}$ such that there exists $\alpha^{(1)}, \alpha^{(2)}, \alpha^{(3)} \in$

$\mathcal{K}_\infty$ and $\gamma \in \mathcal{K}$ such that $\alpha^{(1)}(\|x\|) \leq \mathcal{V}(x) \leq \alpha^{(2)}(\|x\|)$, $\mathcal{V}(f(x)) - \mathcal{V}(x) \leq -\alpha^{(3)}(\|x\|) + \gamma(\|w\|)$ for all $x \in \mathcal{S}$, $w \in \mathcal{W}$ is an ISS-LF for f in $\mathcal{S}$ with respect to w. □

In the disturbance-free case, $w = 0$, the ISS-LF definition is equivalent to that of LF, since $\gamma(0) = 0$.

Result 1: Given $x(k+1) = f(x(k))$, $x \in \mathbb{R}^n$, and a PI $\mathcal{S}$ for f, $0 \in \mathcal{S}$, if there exists a LF for f in $\mathcal{S}$, the origin is AS for f in $\mathcal{S}$. Given $x(k+1) = f(x(k), w(k))$, $x \in \mathbb{R}^n$, $w \in \mathcal{W} \subseteq \mathbb{R}^d$, and a RPI $\mathcal{S}$ for f, $0 \in \mathcal{S}$, if there exists an ISS-LF for f in $\mathcal{S}$, the origin is input-to-state-stable (ISS) for f in $\mathcal{S}$ with respect to w. □

Next we analyze ISS properties when the full system (4) is in closed-loop with the MPC (19), whose prediction model is based on the plant (1) only, and ignores the actuator (3). Due to space limitations, we list here the results without (or with minimal sketches) of the proofs. We start with ISS with respect to the maximum input step change, which relates the closed-loop ultimate bound to the design contract (8), and with respect to the actual commanded input step change.

Result 2: Let Assumption 1 hold. Then, the plant dynamics (1) in closed-loop with (19) and subject to actuator dynamics (3) are ISS with respect to $\varsigma = \max_{\Delta\mathcal{U}} \|\Delta u\|$. □

Result 2 follows from $\mathcal{V}$ being a LF when $w = 0$ and Lipschitz continuous, and from $\|w_x(k)\| \leq c \max_{\Delta\mathcal{U}} \|\Delta u(k)\|$, for some $c \in \mathbb{R}_+$, for all $k \in \mathbb{Z}_+$.

Result 3: Let Assumption 1 hold. Then, the plant dynamics (1) in closed-loop with (19) and subject to the actuator dynamics (3) are ISS with respect to $\|\Delta u(k)\|$. □

Result 3 follows from the same arguments as Result 2 and $\|w_x(k)\| = \||\tilde{M}_c \delta x_a^{(u)}(k)| + |M_s \Delta u(k)|\|$ for a suitable matrix $\tilde{M}_c$. Then, the system can be represented as the cascade of two ISS systems, the plant and the actuator dynamics, combined with a static map, which is ISS [9].

The last result shows that controller (19) can be designed so that the full closed-loop system is AS.

Theorem 2: Let Assumption 1 hold and the equilibrium for x be in the interior of $\mathcal{X}_N$. For a proper choice of a $\eta > 0$, $Q \geq 0$, $R > 0$ the closed-loop is locally AS in an RPI around the equilibrium where the constraints of (18) are inactive. Furthermore, for proper choices of Q, R and $\max_{\Delta\mathcal{U}} \|\Delta u\|$, the closed-loop is AS in $\mathcal{C}$. □

For proving Theorem 2 one needs to prove that: (i) for any LF of the actuator dynamics $\mathcal{V}_a$ and any $\eta > 0$, $\mathcal{V}_s(\xi) = \mathcal{V}(x) + \eta \mathcal{V}_a(\delta x_a)$, $\xi = (x, x_a)$ is a LF for the disturbance-free full system; (ii) $\mathcal{V}_s(\xi)$ is a local ISS-LF in the stated RPI, by showing that we can choose [10] $\mathcal{V}_a$, η such that the decrease of $\mathcal{V}_a$ dominates the increase in $\mathcal{V}$ due to the non-ideal actuation dynamics; (iii) for a choice of Q, R, in a neighborhood $\mathcal{B}$ of the equilibrium the nominal decrease of $\mathcal{V}$ dominates the increase due to non-ideal actuation dynamics in $\mathcal{V}_s$; (iv) for additional conditions on Q, R and/or $\max_{\Delta\mathcal{U}} \|\Delta u\|$, the ultimate bound due to ISS from Result 3 is contained in $\mathcal{B}$, resulting in convergence, and, combined with the previous results, AS.

V. Case study: vehicle steering control

We demonstrate the proposed method in a case study of lateral dynamics control of a vehicle equipped with an angle-controlled steering actuator. The plant is the linearized single track model [11] with constant the longitudinal velocity v_x,

$$\dot{v}_y = -2\frac{(C_f + C_r)}{m v_x} v_y - \left(2\frac{C_f \ell_f - C_r \ell_r}{m v_x} + v_x\right)\varphi + 2\frac{C_f}{m}\delta$$
$$\dot{\varphi} = -2\frac{C_f \ell_f - C_r \ell_r}{I_z v_x} v_x - 2\frac{C_f \ell_f^2 + C_r \ell_r^2}{I_z v_x}\varphi + 2\frac{\ell_f C_f}{I_z}\delta$$

where subscripts f, r denote front and rear, C_i, ℓ_i, $i \in \{f, r\}$ are the tire stiffnesses and the semi-axles lengths from center of mass, m, I_z are the vehicle mass and inertia along the vertical axis, v_y is the lateral velocity, φ is the yaw rate, δ is the (road wheel) steering angle. The tire slip angles are

$$\alpha_f = (v_y + \ell_f \varphi)/v_x, \quad \alpha_r = (v_y - \ell_r \varphi)/v_x. \qquad (21)$$

The parameters are from a real SUV [12] on a wet road. The actuator is an angle-controlled electric power steering, where the closed-loop response from command to road wheel angle is a 2^{nd} order system with dc-gain 1, rise time about 0.35s and 17.5% overshoot. The frequency separation between plant and actuator is less than 1 order of magnitude.

We enforce constraints on the steering angle and lateral velocity, and on tire slip angles that ensure the state to remain in the region where the tire model linearization [11] is valid,

$$\delta_{\min} \leq \delta \leq \delta_{\max} \qquad v_{y_{\min}} \leq v_y \leq v_{y_{\max}} \qquad (22a)$$
$$\alpha_{f_{\min}} \leq \alpha_f \leq \alpha_{f_{\max}} \qquad \alpha_{r\min} \leq \alpha_r \leq \alpha_{r\max}. \qquad (22b)$$

We design the controller to track a yaw rate reference by commanding the steering angle, $u = \delta$, with sampling period 0.3s, where the prediction model is based only on the vehicle dynamics (20) and ignores the actuator. To ensure robust constraint enforcement despite lacking detailed knowledge of the steering actuation we establish the design "contract" between controller and actuator as in (8). To this end we limit the steering command step change $\|\Delta\delta\| \leq \Delta u_{\max}$ for different values of $\Delta u_{\max} = n \cdot 0.1$, $n \in \mathbb{Z}_{[1,5]}$ and we use the information provided by the actuator, namely $\mathcal{W}_x$, $\mathcal{W}_u$, to construct the RCI constraint (18d) for each case. Assumption 1 is enforced as per Remark 1 with $N = 10$.

The results for tracking a yaw rate reference signal in closed loop with a simulation model including the continuous time plant model (20) and the actuator model are shown in Fig. 2–5. The bottom plot of Fig. 3 shows with the same colors used for the corresponding trajectories in Fig. 2–5 the different $\Delta u_{\max}$ limits. The constraints are always enforced despite the reference being in some time intervals steady state infeasible. On the other hand if MPC is designed for ensuring nominal recursive feasibility, i.e., by a CI set, but without robustifying against the modeling error due to ignoring the actuator dynamics, the closed-loop simulation with only (20) is successful, but if the simulation model also includes the actuator dynamics the rear tire slip angle constraint (and briefly the steering angle constraint, not shown here) is violated when the yaw rate reference amplitude step is large, see Figure 4 a little after 3s. This causes infeasibility of the optimization problem and failure of the MPC law, resulting in the reference no longer being tracked.

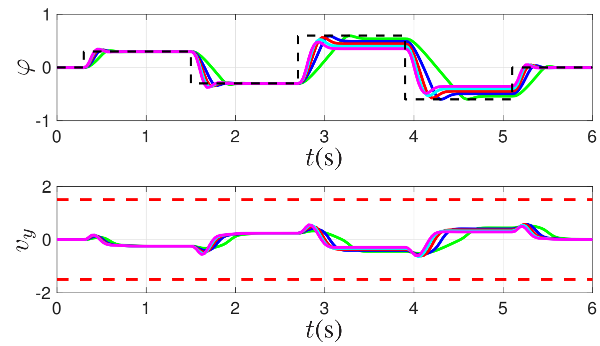

Fig. 2. Simulation results: state trajectories for different values of $\Delta u_{\max}$. Reference (dash black) and constraints (dash red).

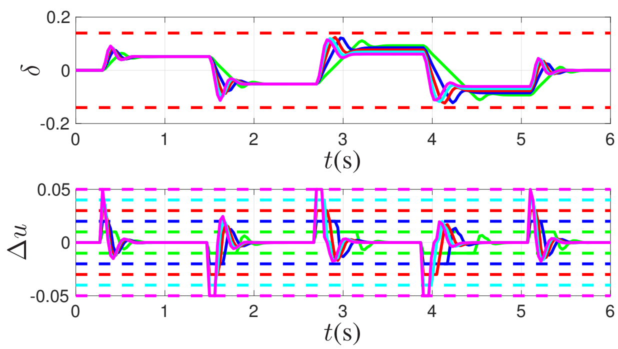

Fig. 3. Simulation results: steering angle and command step change for different values of $\Delta u_{\max}$. Constraints (dash), including values of $\Delta u_{\max}$.

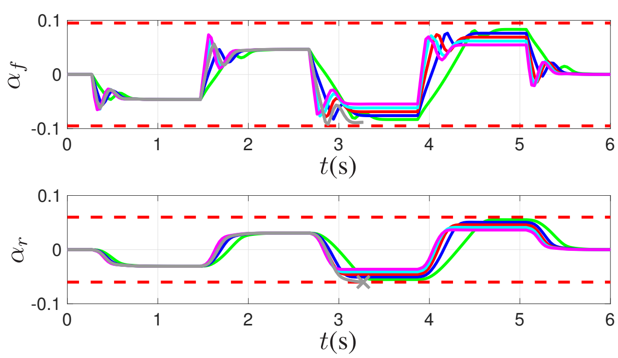

Fig. 4. Simulation results: constrained outputs, i.e., tire slip angles, trajectories for different values of $\Delta u_{\max}$. Constraints (dash red). Without robustifying against the neglected actuator dynamics (gray, $\Delta u_{\max} = 0.3$), the rear slip angle constraints is violated ($\times$-mark).

Note that when the reference is steady state feasible, the controller achieves stable, offset free tracking. This is in accordance with Theorem 2, for the chosen tuning of the cost function weights Q, R for small values of R steady state oscillations may appear. From Fig. 2 we see the trade-off in reachable setpoint and response speed due to (8). Fig. 5

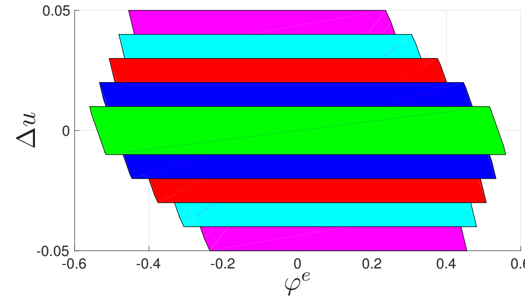

Fig. 5. Section of $\mathcal{C}_{\Delta u}$ for different values of $\Delta u_{\max}$ in the plane of equilibrium yaw rate (φ^e) and Δu.

shows section s of $\mathcal{C}_{\Delta u}$ for different $\Delta u_{\max}$ highlighting the trade-off between $\Delta\mathcal{U}$ and the achievable setpoints.

VI. Future Work

We proposed a modular design for constrained control in multi-layer architectures, where the layers are required to declare only minimal information. The relation between command change and error caused by ignoring the detailed actuator dynamics is used to derive uncertainty sets for robust invariant design. Then, MPC enforces the invariant and the command step change constraints, guaranteeing constraint satisfaction. In the future we will examine adjusting the trade-off between achievable equilibria-step change bounds by switching between different controllers, and we will analyze the case of nonlinear actuator dynamics.

References

[1] F. Blanchini and S. Miani, *Set-theoretic methods in control*. Birkhäuser Boston, 2007.
[2] J. Rawlings and D. Mayne, *Model predictive control: Theory and design*. Madison, WI: Nob Hill Publishing, LLC, 2009.
[3] A. Sangiovanni-Vincentelli, W. Damm, and R. Passerone, "Taming Dr. Frankenstein: Contract-based design for cyber-physical systems," *European Journal of Control*, vol. 18, no. 3, pp. 217–238, 2012.
[4] P. Nuzzo, H. Xu, N. Ozay, J. B. Finn, A. L. Sangiovanni-Vincentelli, R. M. Murray, A. Donzé, and S. A. Seshia, "A contract-based methodology for aircraft electric power system design," *IEEE Access*, vol. 2, pp. 1–25, 2014.
[5] S. Lucia, M. Kögel, and R. Findeisen, "Contract-based predictive control of distributed systems with plug and play capabilities," in *Proc. 5th IFAC Int. Conf. Nonlinear Model Predictive Control*, 2015.
[6] X. Xu, J. W. Grizzle, P. Tabuada, and A. D. Ames, "Correctness guarantees for the composition of lane keeping and adaptive cruise control," *IEEE Trans. Automation Science and Engineering*, 2017.
[7] R. Scattolini, "Architectures for distributed and hierarchical model predictive control–a review," *J. Process Control*, vol. 19, no. 5, pp. 723–731, 2009.
[8] S. Di Cairano, "Indirect adaptive model predictive control for linear systems with polytopic uncertainty," in *American Control Conf.*, 2016.
[9] E. D. Sontag, "The ISS philosophy as a unifying framework for stability-like behavior," in *Nonlinear control in the year 2000*, A. Isidori, F. Lamnabhi-Lagarrigue, and W. Respondek, Eds. Springer, 2001, vol. 2, pp. 443–467.
[10] E. Sontag and A. Teel, "Changing supply functions in input/state stable systems," *IEEE Trans. Automatic Control*, vol. 40, no. 8, pp. 1476–1478, 1995.
[11] R. Rajamani, *Vehicle dynamics and control*. Springer, 2011.
[12] S. Di Cairano, U. Kalabić, and K. Berntorp, "Vehicle tracking control on piecewise-clothoidal trajectories by MPC with guaranteed error bounds," in *Proc. 55th IEEE Conf. on Decision and Control*, 2016.