2025 25th International Conference on Process Control (PC)
June 3–6, 2025, Štrbské Pleso, Slovakia

Quadcopter altitude control using model predictive control with online constraint removal

Nora Lindner
Automatic Control and Systems Theory
Ruhr University Bochum
Bochum, Germany
nora.lindner@rub.de

Juraj Holaza
Institute of Information Engineering, Automation, and Mathematics
Slovak University of Technology in Bratislava
Bratislava, Slovak Republic
juraj.holaza@stuba.sk

Juraj Oravec
Institute of Information Engineering, Automation, and Mathematics
Slovak University of Technology in Bratislava
Bratislava, Slovak Republic
juraj.oravec@stuba.sk

Martin Mönnigmann
Automatic Control and Systems Theory
Ruhr University Bochum
Bochum, Germany
martin.moennigmann@rub.de

***Abstract*—Model predictive control (MPC) is an established and acclaimed method. Because MPC is based on solving an optimal control problem (OCP) in every sampling interval, MPC incurs costly calculations at runtime, however. Consequently, a lot of research effort has been and still is devoted to reducing the computational effort of MPC. Most approaches attempt to reduce the maximum computing time of the underlying OCP because of the immediate importance of this quantity for realtime applicability of MPC. In contrast, we devote our efforts to reducing the computational effort of MPC on average or in the large in the present paper, because the average computing time determines the average power requirements of MPC, which, in turn, are crucial for use of MPC in battery-powered devices. Because battery power is a limiting factor for drones, and because control algorithms for drones require sampling times of milliseconds or below, they are an excellent application example. Specifically, we demonstrate that the average computational effort of MPC applied to the hovering control of a quadcopter can be reduced considerably with a technique known as online constraint removal.**

***Index Terms*—model predictive control, unmanned aerial vehicles, online constraint removal**

I. Introduction

Model predictive control (MPC) is a control method that periodically solves a constrained optimal control problem (OCP) at each time step to determine the optimal input signal. Because it is based on optimization, MPC provides a natural way to incorporate constraints on inputs, states and outputs of the system, a feature that distinguishes MPC from most other established control methods. On the other hand, the requirement of solving an optimization problem in every sampling interval makes MPC computationally expensive. This drawback has prompted numerous investigations on how to reduce the computational effort of MPC. Many investigations addressed the computational efficiency by exploiting the structure of the underlying OCP that results from the repetition of the dynamics along the horizon and the relation of feedback to warm starting the numerical solver at the next time instance. Another branch of research attempts to solve the OCP analytically, or analytically to within some controllable approximation. Notably, this led to the extension of the state feedback law that solves the unconstrained linear-quadratic regulator to the constrained case [1], [2]. Finally, the recent surge in popularity of machine learning methods has induced many research projects towards reducing the computational effort of MPC. We refer to [3] for a recent summary of efforts focused on accelerating MPC with a focus on use on embedded devices and other devices with limited memory and computational power.

Here we focus on a particular variant of methods for reducing the computational effort of MPC called constraint removal. Constraint removal aims to detect obsolete constraints at runtime. By removing these constraints from the OCP, the computational effort is reduced. We stress that constraint removal goes beyond removing constraints that are redundant in that they can never, i.e. for no current state or equivalently no initial condition to the OCP, become active. In contrast, constraint removal removes constraints depending on the current state and thus in a time-variant fashion.

Various methods for constraint removal have been explored in the meantime, starting with the case of linear-quadratic constrained OCPs and MPC [4]–[7], and later extended to nonlinear cases [8]. The MPC design method proposed in this paper builds on a variant that first precomputes bounds on the cost function values that can be attained if a constraint is active, and subsequently detects constraints cannot become active anymore depending on the current cost function value [7]. This method, which calculates the cost function bounds offline and before the runtime of the actual MPC controller, is explained in more detail in Section II-B.

The first applications of constraint removal to real systems were reported in [9], [10] to the knowledge of the authors. Recently, constraint removal has also been applied to hyperthermia control for cancer treatment [9], [11]. The particular variant of constraint removal used in these works is based on calculating forward and backward reachable sets

979-8-3315-2530-9/25/$31.00 ©2025 IEEE

and determining obsolete constraints based on these sets [12], [13].

We present the preliminaries, including a brief introduction into the variant of constraint removal used here, in Section II. Section III reports the results achieved with constraint removal for the simple drone altitude control problem considered here. Conclusions and a brief outlook are stated in Section IV.

II. Problem statement

A. Linear-quadratic MPC problem

We consider linear time-invariant discrete-time systems

$$x(k+1) = Ax(k) + Bu(k), \tag{1a}$$
$$y(k) = Cx(k) \tag{1b}$$

for time steps $k \in \mathbb{N}$ with the state $x(k) \in \mathbb{R}^n$, the manipulated variables $u(k) \in \mathbb{R}^m$, the measured variables $y(k) \in \mathbb{R}^p$ and the matrices $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$ and $C \in \mathbb{R}^{p\times n}$. We assume the pair (A, B) to be stabilizable. The state variables and manipulated variables are constrained by upper bounds and lower bounds

$$\underline{x}_i \leq x_i(k) \leq \overline{x}_i, \tag{2a}$$
$$\underline{u}_j \leq u_j(k) \leq \overline{u}_j \tag{2b}$$

for all $i = 1, \ldots, n$ and $j = 1, \ldots, m$. The system (1) is regulated to the origin by solving the OCP

$$\min_{X,U} \quad V_f\left(x(N), u(N)\right) + \sum_{i=0}^{N-1} l\left(x(i), u(i)\right) \tag{3a}$$
$$\text{s.t. } x(0) = x, \tag{3b}$$
$$x(k+1) = Ax(k) + Bu(k), \tag{3c}$$
$$x(k) \in \mathcal{X}, \tag{3d}$$
$$u(k) \in \mathcal{U}, \tag{3e}$$
$$x(N) \in \mathcal{T} \tag{3f}$$

for $k = 0, \ldots, N-1$ and the decision variables summarized by $X = \left(x^{\mathrm{T}}(1), \ldots x^{\mathrm{T}}(N)\right)^{\mathrm{T}}$ and $U = \left(u^{\mathrm{T}}(0), \ldots u^{\mathrm{T}}(N-1)\right)^{\mathrm{T}}$, on a receding horizon with length N, given the current state x. In (3d) and (3e), the sets $\mathcal{X} \subseteq \mathbb{R}^n$ and $\mathcal{U} \subseteq \mathbb{R}^m$ are defined by (2a) and (2b), respectively. The cost function in (3a) consists of the terms $V_f\left(x(N), u(N)\right) = x^{\mathrm{T}}(N)Px(N)$ and $l\left(x(i), u(i)\right) = x^{\mathrm{T}}(i)Qx(i) + u^{\mathrm{T}}(i)Ru(i)$. We assume the weighting matrices $Q \in \mathbb{R}^{n\times n}$ to be positive semi-definite and $R \in \mathbb{R}^{m\times m}$ and $P \in \mathbb{R}^{n\times n}$ to be positive definite, where the terminal penalty matrix P is the solution to the algebraic Riccati equation of (3) without inequality constraints. The terminal set $\mathcal{T}$ (3f) is assumed to be a full dimensional, polyhedral set, containing the origin in its interior. The set can be calculated as proposed in [14], using the closed loop system $x(k+1) = (A - BK)\,x(k)$, where $K = \left(B^{\mathrm{T}}PB + R\right)^{-1} B^{\mathrm{T}}PA$. We note that this OCP comprises

$$q = 2N\,(n+m) + r \tag{4}$$

constraints, where r is the number of constraints resulting from the polyhedral terminal set $\mathcal{T}$.

By substituting the system equation (1a) in (3), the optimal control problem (3) can be stated as an equivalent quadratic program (QP) of the form

$$\min_{U} \; V(x, U) \tag{5a}$$
$$\text{s.t. } GU - Ex \leq w, \tag{5b}$$

where $V(x,U) = \frac{1}{2}x^{\mathrm{T}}Yx + x^{\mathrm{T}}FU + \frac{1}{2}U^{\mathrm{T}}HU$, $Y \in \mathbb{R}^{n\times n}$, $F \in \mathbb{R}^{n\times mN}$, $H \in \mathbb{R}^{mN\times mN}$, $G \in \mathbb{R}^{q\times mN}$, $E \in \mathbb{R}^{q\times n}$ and $w \in \mathbb{R}^q$. The weighting matrix H is positive definite, thus the solution to (5) is unique [1]. Solving the quadratic program (5) results in an open-loop optimal input sequence U^* for the given initial state x. Closed-loop control with MPC then results from applying the first element $u^*(0)$ of U^* and periodically solving (5) for the system state at every time step k and applying the first optimal input signal. We refer to the set of states $x \in \mathcal{X}$, for which (5) has a solution, as the feasible set $\mathcal{F}$. For any $i \in \mathcal{Q} = \{1, \ldots, q\}$, G_i, E_i and w_i denote the ith row of the respective matrix or vector in the sequel. We call the ith constraint active at the optimal solution U^* for an $x \in \mathcal{F}$ if $G_iU^* - E_ix = w_i$ and inactive otherwise, and collect the indices of the active and inactive constraints in $\mathcal{A} \subseteq \mathcal{Q}$ and $\mathcal{I} \subseteq \mathcal{Q}$, respectively.

B. Constraint removal in linear MPC

We summarize the particular constraint removal method first introduced in [7] as needed for the present paper. The aim of this method is to remove constraints that have no influence on the optimal solution from the optimization problem, thus reducing complexity of the OCP.

Assume the optimal solution U^* has been determined for an initial condition $x \in \mathcal{F}$. Then the inactive constraints $G_iU^* - E_ix < w_i$, $i \in \mathcal{I}$ remain inactive on a sufficiently small but finite ball around x and therefore do not influence the solution. If the set of inactive constraints $\mathcal{I}$, or a subset thereof, had been known in advance, these inactive constraints could have been removed from the OCP. The following proposition from [7] states this concisely.

Proposition 1: [7] Let $x \in \mathcal{F}$ be arbitrary and let $\tilde{\mathcal{I}} \subseteq \mathcal{I}$ be an arbitrary subset of the inactive constraints. Consider the reduced optimization problem

$$\begin{aligned} &\min_{\hat{U}} \; V(x, \hat{U}) \\ &\text{s.t. } G_{\mathcal{Q}\setminus\tilde{\mathcal{I}}}\hat{U} - E_{\mathcal{Q}\setminus\tilde{\mathcal{I}}}x \leq w_{\mathcal{Q}\setminus\tilde{\mathcal{I}}}. \end{aligned} \tag{6}$$

Then (6) has a unique solution, which we denote by $\hat{U}^*$. This solution is equal to the solution obtained from (5), i.e., $\hat{U}^* = U^*$ and $V(x, \hat{U}^*) = V(x, U^*)$.

Obviously, Proposition 1 is only useful if some or all inactive constraints are known in advance. Following the method outlined in [7], we apply precomputed characteristic bounds on the optimal cost function to detect inactive constraints

for a given $x \in \mathcal{F}$ before solving the OCP. Specifically, we calculate, for every $i \in \mathcal{Q}$, the bound $\sigma_i \in \mathbb{R}$ defined by

$$\begin{aligned} \sigma_i := \min_{x,U} \; & V(x,U) \\ \text{s.t. } & G_i U - E_i x = w_i, \\ & G_{\mathcal{Q}\setminus\{i\}} U - E_{\mathcal{Q}\setminus\{i\}} x < w_{\mathcal{Q}\setminus\{i\}}, \\ & x \in \mathcal{F}. \end{aligned} \tag{7}$$

Note that (7) is infeasible if the constraint i is not active for any $x \in \mathcal{F}$. In this case, we formally set

$$\sigma_i := \infty. \tag{8}$$

The properties of these σ_i bounds are summarized in the following lemma.

Lemma 1: [7] Let $i \in \mathcal{Q}$ be arbitrary and consider the QP (7). Then the following statements hold:

(i) If QP (7) has a solution, then this solution is unique.
(ii) If (7) has a solution, then $0 < \sigma_i < \infty$.
(iii) If (7) does not have a solution, then constraint i is always inactive in (5), or equivalently $i \in \mathcal{I}(x)$ for all $x \in \mathcal{F}$.

By its definition in (7), σ_i^* essentially is a lower bound on the value the cost function $V(x,U)$ can attain for any initial condition if the ith constraint is active. Consequently, the ith constraint cannot be active and thus can be removed, whenever the optimal cost function value drops below the bound σ_i. Because the optimal cost function is a Lyapunov function under the assumptions stated in Section II-A, the optimal cost function value is nonincreasing along closed-loop trajectories (see, e.g. [15]). Therefore, the ith constraint can be removed for all subsequent time steps for the nominal system once the optimal cost function value drops below σ_i. More precisely, the following statement holds.

Proposition 2: [7] Let $i \in \mathcal{Q}$ and $x \in \mathcal{F}$ be arbitrary, and assume σ_i, as defined in (7) and (8), has been determined. If $V(x, U^*(x)) < \sigma_i$, then constraint i is inactive at the optimal solution of (5) for the current state x. Furthermore, $V(x(k_0), U^*(x(k_0))) < \sigma_i$ implies $i \in \mathcal{I}(x(k))$ for all $k > k_0$.

Algorithm 1 summarizes the MPC with constraint removal according to Proposition 2. We abbreviate the optimal cost function value at iteration k as $V\left(x(k), U^*(x(k))\right)$ by V^*. The set $\mathcal{R}$ denotes the set of indices corresponding to the constraints that have not been removed, and thus can be active or inactive. The QP (5) is initially solved at iteration $k = 1$. At each iteration the optimal cost function value V^* is compared to the precalculated bounds σ_i for $i \in \mathcal{Q}$. If the value of σ_i is larger than the optimal cost function value V^* the corresponding constraint will remain inactive and thus can be removed from (5). In the next iteration the successor state x^+ becomes the new current state x and the QP (5) is solved for the reduced number of constraints.

Algorithm 1 MPC with constraint removal (see [7])

Input: $x^+, V^*(x), \mathcal{R} = \mathcal{Q} \setminus \tilde{\mathcal{I}}, \sigma_i, i \in \mathcal{R}$
for all $i \in \mathcal{R}$ **do**
  if $V^* < \sigma_i$ **then**
    ith constraint will remain inactive: $\tilde{\mathcal{I}} \leftarrow \tilde{\mathcal{I}} \cup \{i\}$.
  end if
end for
Remove inactive constraints: $\mathcal{R} \leftarrow \mathcal{R} \setminus \tilde{\mathcal{I}}$
Solve reduced optimization problem in (5) for $x = x^+$ and reduced set of constraints $\mathcal{R}$.
Output: Updated $U^*, V^*, x^+, \mathcal{R}$

C. Quadcopter altitude model

In this section the quadcopter model for the case study is introduced. The quadcopter model is derived with classical modelling approaches using Euler angles (see e.g. [16]). The model is based on the following assumptions. The quadcopter structure and the rotors are rigid bodies. The quadcopter is modelled as a point mass, which lies in the center of gravity of the symmetrical structure. Thrust and drag moment, which are caused by the rotor movement, are proportional to the square of the rotor rotational speed. Aerodynamic drag is assumed to be linear. Other aerodynamic effects, such as the ground effect, are neglected. The differential equations of the nonlinear state space model describe the quadcopters translational movement in terms of position $x(t)$, $y(t)$ and the altitude $z(t)$ of the center of gravity in an earth-fixed coordinate frame I, and its attitude by the roll angle $\phi(t)$, the pitch angle $\theta(t)$ and the yaw angle $\psi(t)$, which describe the rotation of a body-fixed frame M against the earth-fixed frame I.

To extract a model that considers only the movement in direction of the z-axis, the angles $\phi(t)$, $\theta(t)$, $\psi(t)$ and the position in the x-y-plane $x(t)$, $y(t)$, as well as the derivatives of these quantities are assumed to be zero. This allows neglecting all but the equations for movement in direction of the z-axis. This results in the state space model

$$\begin{aligned} \dot{x}_{\mathrm{nl}}(t) &= \begin{pmatrix} -\frac{c_{\mathrm{z}}}{m} & 0 \\ 1 & 0 \end{pmatrix} x_{\mathrm{nl}}(t) + \begin{pmatrix} -1 \\ 0 \end{pmatrix} F_{\mathrm{z}}(t) + \begin{pmatrix} g \\ 0 \end{pmatrix} \\ y_{\mathrm{nl}}(t) &= \begin{pmatrix} 0 & 1 \end{pmatrix}^T x_{\mathrm{nl}}(t), \quad x_{\mathrm{nl}}(0) = x_{\mathrm{nl0}} \end{aligned} \tag{9}$$

with

$$F_{\mathrm{z}}(t) = \frac{4\pi^2 c_{\mathrm{th}}}{m} \left(n_1^2(t) + n_2^2(t) + n_3^2(t) + n_4^2(t) \right), \tag{10}$$

$x_{\mathrm{nl}}(t) = [\dot{z}(t),\ z(t)]^{\mathrm{T}}$, and $u_{\mathrm{nl}}(t) = [n_1(t), \ldots, n_4(t)]^{\mathrm{T}}$. The model (9), (10) is nonlinear due to the nonlinear dependence of the thrust force $F_{\mathrm{z}}(t)$ on the rotational speeds $n_i(t)$ in (10). Since (10) is bijective for the relevant nonnegative rotational speeds, and since we can assume all $n_i(t)$ to be equal for our investigation of motions along the z-axis, we consider $F_{\mathrm{z}}(t)$ to be our input. Denoting the common value of $n_i(t),\ i \in \{1, \ldots, 4\}$ by $n(t)$ for simplicity, this results in

$$u(t) = F_{\mathrm{z}}(t) = \frac{16\pi^2 c_{\mathrm{th}}}{m} n^2(t). \tag{11}$$

which now is a single linear input for the system (9).

Furthermore, it is immediately evident from (9) that any $x^{\mathrm{e}} \in \left\{ x^{\mathrm{e}} \in \mathbb{R}^2 \mid \dot{z}^{\mathrm{e}} = 0 \right\}$ is a steady state for the constant

input $u^{\mathrm{e}} = g$. Obviously, this matches the observation that the drone can hover at an arbitrary height if the thrust compensates for the gravitational acceleration. This implies the constant term on the right-hand side of the differential equations (9) disappears if we consider deviations from an arbitrary but fixed hover state x^{e}. Specifically, the model in $u^\delta(t) = u(t) - u^{\mathrm{e}}$ and $x^\delta(t) = x(t) - x^{\mathrm{e}}$ reads

$$\begin{aligned} \dot{x}^\delta(t) &= \begin{pmatrix} -\frac{c_{\mathrm{z}}}{m} & 0 \\ 1 & 0 \end{pmatrix} x^\delta(t) + \begin{pmatrix} -1 \\ 0 \end{pmatrix} u^\delta(t), \\ y^\delta(t) &= \begin{pmatrix} 0 & 1 \end{pmatrix}^T x^\delta(t), \quad x^\delta(0) = x_0^\delta. \end{aligned} \tag{12}$$

For the output signal $y^\delta(t) = y(t) - y^{\mathrm{e}}$ applies similarly. Finally, we discretize (12) in time with the exact discretization method, which yields

$$\begin{aligned} x^\delta(k+1) &= \begin{pmatrix} e^{-\frac{c_{\mathrm{z}}}{m}T} & 0 \\ \frac{m}{c_{\mathrm{z}}}\left(1 - e^{-\frac{c_{\mathrm{z}}}{m}T}\right) & 1 \end{pmatrix} x^\delta(k) \\ &\quad + \begin{pmatrix} -\frac{m}{c_{\mathrm{z}}}\left(1 - e^{-\frac{c_{\mathrm{z}}}{m}T}\right) \\ -\frac{m}{c_{\mathrm{z}}}\left(T - \frac{m}{c_{\mathrm{z}}}\left(1 - e^{-\frac{c_{\mathrm{z}}}{m}T}\right)\right) \end{pmatrix} u^\delta(k), \\ y^\delta(k) &= \begin{pmatrix} 0 & 1 \end{pmatrix}^T x^\delta(k), \quad x^\delta(0) = x_0^\delta. \end{aligned} \tag{13}$$

III. Case study

A. Model parametrization

The quadcopter model (13) is configured using the parameter values of a Crazyflie 2.1 quadcopter, which were published in [17] and are listed in Table I. We choose the sample time $T = 0.01\,\mathrm{s}$, which is a typical update rate for quadcopter altitude control (e.g., used in the Bitcraze framework[1] and the ArduPilot ArduCopter flight software[2]). This results in the system matrices

$$A = \begin{pmatrix} a_{11} & 0 \\ a_{21} & 1 \end{pmatrix}, \quad b = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}, \quad c = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \tag{14}$$

with $a_{11} = 0.996$, $a_{21} = b_1 = -9.981 \times 10^{-3}$ and $b_2 = -4.994 \times 10^{-5}$.

The MPC problem (3) is considered with the model (13) and the matrices (14). We set a prediction horizon of $N = 17$ and use the penalty matrices

$$P = \begin{pmatrix} 51.630 & 100.011 \\ 100.011 & 554.071 \end{pmatrix}, \; Q = \begin{pmatrix} 1 & 0 \\ 0 & 10 \end{pmatrix}, \; R = 0.1\,. \tag{15}$$

The weighting matrices Q and R were tuned manually. The terminal penalty P as well as the terminal set $\mathcal{T}$ were determined using the Multi-Parametric Toolbox [18]. The rotational speed of the rotors is bounded from below by $n_{\min}$ and above by $n_{\max}$ (Table I). Converting these values into lower and upper bounds on the thrust and subtracting the thrust required to compensate gravitational acceleration yields the bounds

$$-9.810 \leq u^\delta(k) \leq 5.652. \tag{16}$$

on the input. Furthermore, we apply the state constraints

$$\begin{pmatrix} -2 \\ -2 \end{pmatrix} \leq x^\delta(k) \leq \begin{pmatrix} 2 \\ 2 \end{pmatrix} \tag{17}$$

to limit the operation range to a range applicable in our laboratory. The resulting OCP comprises 148 constraints.

TABLE I
Model parameters for a Crazyflie 2.1 quadcopter

c_{th}	$2.287 \times 10^{-8}\,\mathrm{kg\,m}$	thrust constant
c_{z}	$14 \times 10^{-3}\,\mathrm{kg\,s^{-1}}$	air drag constant
m	$37 \times 10^{-3}\,\mathrm{kg}$	quadcopter mass
g	$9.81\,\mathrm{m\,s^{-2}}$	gravitation constant
$n_{\min}$	$0\,\mathrm{s^{-1}}$	minimum applicable rotor speed
$n_{\max}$	$398\,\mathrm{s^{-1}}$	maximum applicable rotor speed

B. Constraint removal

We applied the constraint removal strategy summarized in Section II to the quadcopter (13) for randomly chosen feasible initial values. Figure 1 shows the trajectories that result under MPC control along for the initial condition $x(0) = (-1.5,\ -1)^{\mathrm{T}}$. The figure also shows the optimal cost function value V^* and the number of considered constraints q required by MPC with (blue) and without constraint removal (red). The constraints on the system state (17) and input (16) are indicated by grey lines. The number of considered constraints drops significantly after the first iteration and then decreases further until the terminal set is reached and all constraints are removed. By construction of the MPC problem (3) with its terminal cost and terminal constraint, the optimal solution to the unconstrained OCP is then the optimal feedback of the unconstrained linear-quadratic regulator.

Figure 2 shows the closed-loop control trajectories, the optimal cost function value V^* and the number of considered constraints q for some additional initial conditions $x(0)$, where the various initial conditions are distinguished by color. Constraints are again indicated by the grey lines. Just as for the results in Figure 1, the cost function value decreases with each iteration for all initial conditions. All constraints are satisfied, see e.g. the lower state constraint $\underline{\dot{z}}^\delta$ for the yellow trajectory or the input constraints for the yellow, red and blue trajectories. In all cases the constraint removal strategy again removes a significant amount of the constraints after the first iteration. As expected, the number of removed constraints increases as the trajectories advance the origin.

We further evaluated the MPC with constraint removal for 400 randomly chosen, feasible initial conditions. Figure 3 shows the mean percentage of removed constraints per iteration (blue). The red line denotes the percentage of constraints, for which $\sigma_i = \infty$ holds. These constraints can never be active

[1] https://www.bitcraze.io
[2] https://ardupilot.org

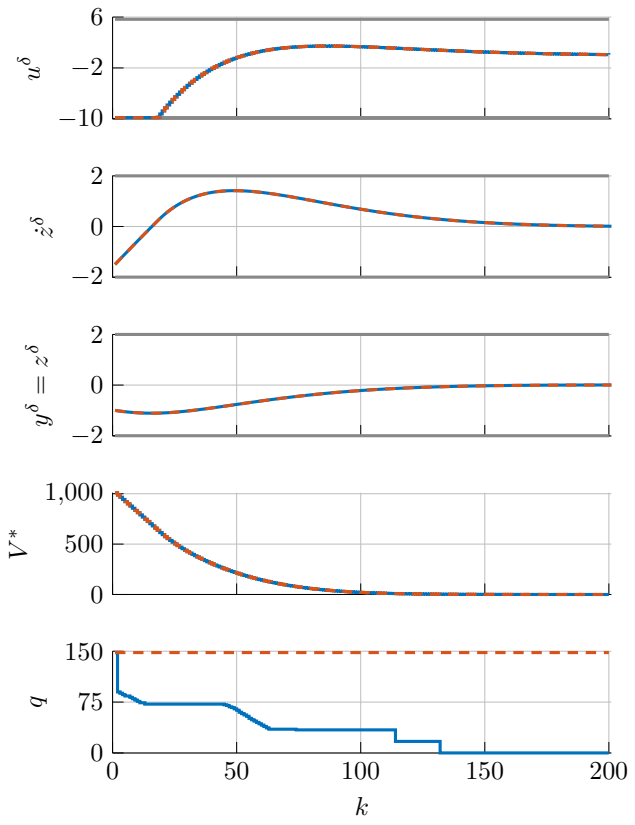

Fig. 1. Closed-loop control trajectories comparing results of conventional MPC (red, dashed) and MPC with constraint removal (blue, full). The constraints on the system state and input are indicated by grey lines.

and can thus be removed from the OCP, even before the first iteration.

IV. Conclusions and Outlook

We demonstrated that online constraint removal efficiently simplifies the constrained optimal control problems that form the core of MPC. In contrast to approaches for reducing the maximum computation time of the underlying optimal control problem, constraint removal reduces the computational effort on average by simplifying the optimal control problem whenever possible even if the current computation time falls well below the maximum admissible computation time. As a result, the overall computational complexity and, in turn, the energy consumption of MPC can be reduced, which is obviously of interest for battery-powered devices such as drones.

Future work will quantify the energy savings that can be achieved with constraint removal. Furthermore, the method will be implemented on real drones and assessed in a flight lab.

Acknowledgements

This work was funded by the European Commission under the grant no. 101079342 (Fostering Opportunities Towards Slovak Excellence in Advanced Control for Smart Industries). We also acknowledge the contribution of the project 09I01-03-V04-00024 (Slovak Research Excellence in Advanced Control for Smart Industries). JO and JH gratefully acknowledge the contribution of the Scientific Grant Agency of the Slovak Republic under grant 1/0297/22.

References

[1] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3–20, 2002.

[2] M. M. Seron, G. C. Goodwin, and J. A. D. Doná, "Characterisation of receding horizon control for constrained linear systems," *Asian Journal of Control.*, vol. 5, no. 2, pp. 271–286, 2003.

[3] R. Findeisen, A. Rose, K. Graichen, and M. Mönnigmann, "Embedded optimization in control: An introduction, opportunities, and challenges," in *Encyclopedia of Systems and Control Engineering*. Elsevier, to appear 2025.

[4] M. Jost and M. Mönnigmann, "Accelerating online MPC with partial explicit information and linear storage complexity in the number of constraints," in *Proceedings of the 2013 European Control Conference*, 2013, pp. 35–40.

[5] M. Jost and M. Mönnigmann, "Accelerating model predictive control by online constraint removal," in *Proceedings of the 52. IEEE Conference on Decision and Control*, 2013, pp. 5764 – 5769.

[6] M. Jost, G. Pannochia, and M. Mönnigmann, "Online constraint removal: accelerating MPC with a Lyapunov function," *Automatica*, vol. 57, pp. 164–169, 2014.

[7] M. Jost, G. Pannochia, and M. Mönnigmann, "Accelerating linear model predictive control by constraint removal," *European Journal of Control*, vol. 35, pp. 42–49, 2017.

[8] R. Dyrska and M. Mönnigmann, "Accelerating nonlinear model predictive control by constraint removal," in *Proceedings of the 7th IFAC Conference on Nonlinear Model Predictive Control*, 2021, pp. 278–283.

[9] S. A. N. Nouwens, B. de Jager, M. M. Paulides, and W. P. M. H. Heemels, "Constraint removal for MPC with performance preservation and a hyperthermia cancer treatment case study," in *2021 60th IEEE Conference on Decision and Control*, 2021, pp. 4103–4108.

[10] R. Dyrska, M. Horváthová, P. Bakaráč, M. Mönnigmann, and J. Oravec, "Heat exchanger control using model predictive control with constraint removal," *Applied Thermal Engineering*, vol. 227, p. 120366, 2023.

[11] S. A. N. Nouwens, B. de Jager, M. M. Paulides, and W. P. M. H. Heemels, "Constraint-adaptive MPC for large-scale systems: Satisfying state constraints without imposing them," in *7th IFAC Conference on Nonlinear Model Predictive Control NMPC 2021*, vol. 54, no. 6, 2021, pp. 232–237.

[12] S. A. N. Nouwens, M. M. Paulides, and W. P. M. H. Heemels, "Constraint-adaptive MPC for linear systems: A system-theoretic framework for speeding up MPC through online constraint removal," *Automatica*, vol. 157, p. 111243, 2023.

[13] Nouwens, S. A. N., Paulides, M. M., and Heemels, W. P. M. H., "Accelerating soft-constrained MPC for linear systems through online constraint removal," in *2023 62nd IEEE Conference on Decision and Control*, 2023, pp. 4273–4278.

[14] E. Gilbert and K. Tan, "Linear systems with state and control constraints: the theory and application of maximal output admissible sets," *IEEE Transactions on Automatic Control*, vol. 36, no. 9, pp. 1008–1020, 1991.

[15] D. Mayne, J. Rawlings, C. Rao, and P. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[16] S. Bouabdallah and R. Siegwart, "Full control of a quadrotor," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007, pp. 153–158.

[17] M. Schwung, "Cooperative event-based control of mobile agents," Ph.D. dissertation, Ruhr-Universität Bochum, 2021.

[18] M. Herceg, M. Kvasnica, C. N. Jones, and M. Morari, "Multi-parametric toolbox 3.0," in *2013 European Control Conference*, 2013, pp. 502–510.

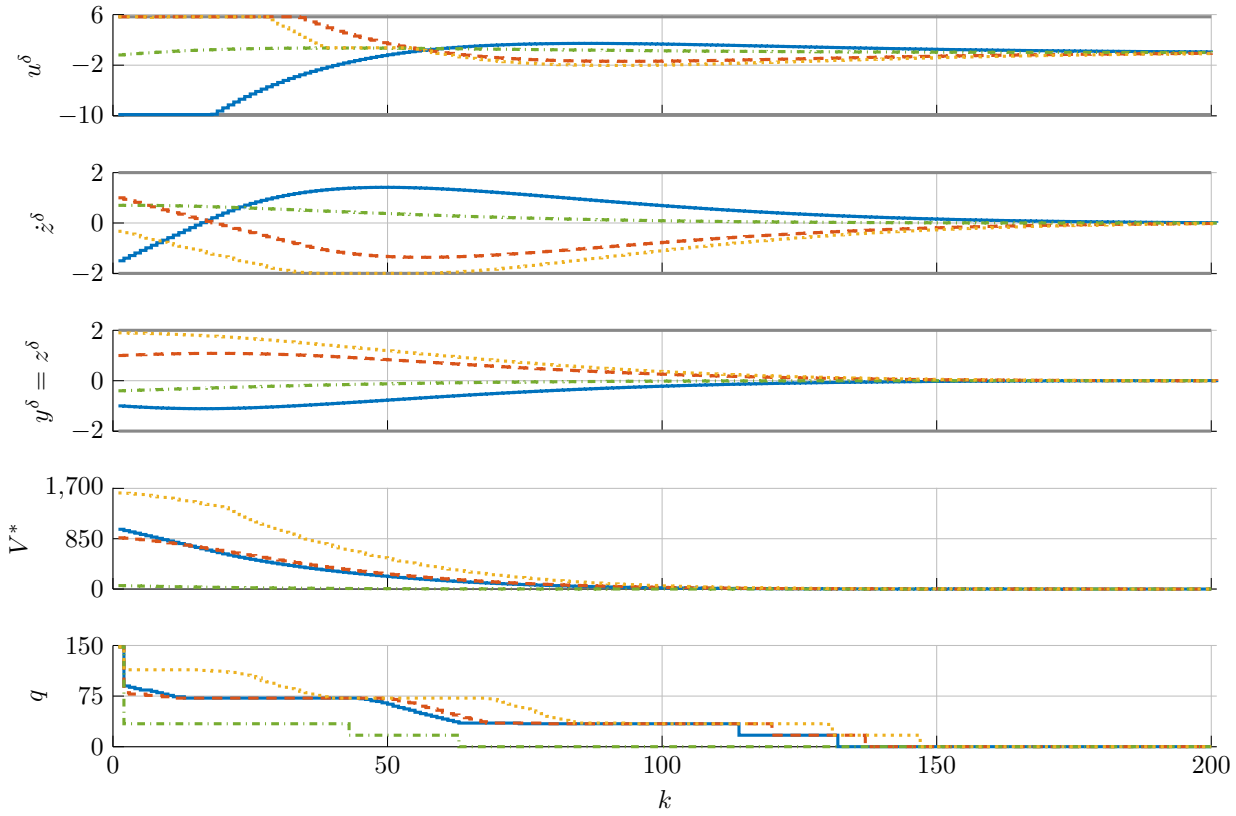

Fig. 2. Closed-loop control trajectories for MPC with constraint removal for various initial conditions. Colors correspond to initial conditions. The constraints on the system state and input are indicated by the grey lines.

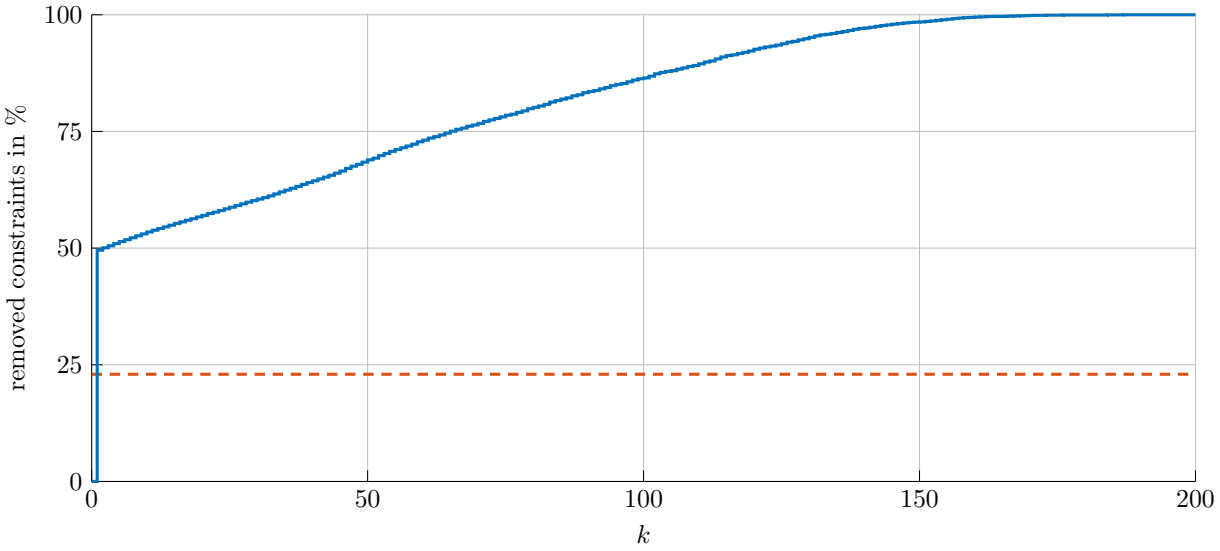

Fig. 3. Mean percentage of removed constraints per iteration (blue). The MPC with constraint removal was evaluated for 400 randomly chosen initial conditions in the feasible set. The percentage of constraints, for which $\sigma_i = \infty$ holds, is indicated by a red, dashed line.