Robust Maneuver Planning with Scalable Prediction Horizons: A Move Blocking Approach

Philipp Schitz, Johann C. Dauer and Paolo Mercorelli

Abstract— **Implementation of Model Predictive Control (MPC) on hardware with limited computational resources remains a challenge. Especially for long-distance maneuvers that require small sampling times, the necessary horizon lengths prevent its application on onboard computers. In this paper, we propose a computationally efficient tube-based shrinking horizon MPC that is scalable to long prediction horizons. Using move blocking, we ensure that a given number of decision inputs is efficiently used throughout the maneuver. Next, a method to substantially reduce the number of constraints is introduced. The approach is demonstrated with a helicopter landing on an inclined platform using a prediction horizon of 300 steps. The constraint reduction decreases the computation time by an order of magnitude with a slight increase in trajectory cost.**

Index Terms— **Predictive control for linear systems, computational methods, autonomous systems, robotics.**

I. INTRODUCTION

MODEL Predictive Control (MPC) is a popular control method for handling systems subject to state and input constraints [1], [2]. In regular MPC, a trajectory optimization problem is solved over a fixed receding horizon at every time step. In order to ensure closed-loop stability, the terminal state is usually constrained to lie within an invariant terminal set [3]. However, invariance of the terminal set may be restrictive in certain applications [4]. For example, reaching a certain position with a non-zero target velocity can not be encoded as an invariant set. It is well known that finite-time arrival to arbitrary terminal sets can be achieved by varying the prediction horizon length [4]–[9]. Usually, the resulting optimization is a mixed-integer program (MIP) where the horizon length acts as the integer decision variable. Finite-time completion and recursive feasibility require that the target is reachable from the current state. As a result, large horizons are often needed to achieve practically relevant regions of

P. Schitz and J. C. Dauer are with the Institute of Flight Systems, German Aerospace Center (DLR), 38108 Braunschweig, Germany. P. Schitz is also with the Institute for Production Technology and Systems, Leuphana University of Lueneburg, 21335 Lueneburg, Germany (e-mail: philipp.schitz@dlr.de; johann.dauer@dlr.de).

P. Mercorelli is with the Institute for Production Technology and Systems, Leuphana University of Lueneburg, 21335 Lueneburg, Germany (e-mail: paolo.mercorelli@leuphana.de).

attraction (ROA). In [7], the required horizon length is reduced by introducing intermediate waysets. By placing these sets in a way that ensures the reachability from one wayset to another, an unmanned aircraft is shown to maneuver to the target region while avoiding obstacles. However, the resulting MIPs need to be solved online which is usually prohibitively expensive on onboard computers. Another strategy [8], [9], often referred to as shrinking horizon MPC (SHMPC), treats the horizon length as a parameter so the problem reduces to a continuous program. Consequently, optimality with respect to horizon length is sacrificed for computational tractability. In the aforementioned approaches, the number of decision variables and thus the computational load vary drastically with horizon length. Since an onboard computer needs to be able to handle the worst-case scenario, the maximal horizon is limited by hardware capabilities. The approach by [6] allows to set a maximal number of decision inputs, regardless of horizon length, using move blocking [10], a scheme where inputs are held constant over some number of steps to reduce the amount of decision variables. The trajectory is then optimized over feasible blocking strategies in a MIP.

A. Main Contribution and Structure of the Paper

In this paper, we present a computationally efficient and robust SHMPC that is scalable to long prediction horizons. In contrast to [6], we simplify the problem to a shrinking horizon formulation in which the horizon length at a given time step is known a priori. On this basis, we present a constructive method to generate time-varying blocking strategies while efficiently utilizing a given maximum number of decision inputs and ensuring recursive feasibility. For long prediction horizons, the number of constraints becomes another significant source of complexity. We therefore reformulate the standard move blocking optimization problem to enable the reduction of constraints. An optimization-based approach is developed that enables the approximation of the constraint set with substantially fewer halfspaces. The efficacy of our approach is demonstrated on a helicopter landing example. The constraint reduction is shown to decrease the computation time by an order of magnitude.

The rest of this paper is structured as follows. Section II introduces the control problem and reviews some preliminaries. In Section III, the proposed SHMPC with move blocking is presented. The constraint reduction is developed in Section IV. Section V demonstrates the approach on a numerical example. Section VI provides a conclusion and outlook.

B. Notation

Given two sets $\mathcal{S}_1$, $\mathcal{S}_2$, set addition and difference are defined as $\mathcal{S}_1 \oplus \mathcal{S}_2 = \{a + b \,|\, a \in \mathcal{S}_1, b \in \mathcal{S}_2\}$ and $\mathcal{S}_1 \ominus \mathcal{S}_2 = \{a \,|\, a \oplus \mathcal{S}_2 \subseteq \mathcal{S}_1\}$, respectively. Given matrices or vectors A, we interpret $A\mathcal{S}$ as $\{As \,|\, s \in \mathcal{S}\}$. We further denote a block diagonal stacking by $\mathrm{diag}(A_1, \ldots, A_n)$, a vertical concatenation by $(A_1, \ldots, A_n)$ and the Kronecker product by $A_1 \otimes A_2$. The n-dimensional identity matrix is denoted by I_n. We interpret $\mathbf{0}_{n \times m}$ and $\mathbf{1}_{n \times m}$ as $n \times m$-dimensional matrix of zeros and ones, respectively. The set of natural numbers ranging from l to u is written as $\mathbb{N}_l^u$.

II. Problem Statement and Preliminaries

Let $x_k \in \mathbb{R}^n$ denote the state at time $k\tau$ with $k \in \mathbb{N}$ and sample time τ. Consider the linear discrete-time system

$$x_{k+1} = Ax_k + Bu_k + w_k, \tag{1}$$

subject to state-input and terminal constraints

$$\forall k \in \mathbb{N}_0^{N_0 - 1} : (u_k, x_k) \in \mathcal{F}, \ x_{N_0} \in \mathcal{X}_T, \tag{2}$$

where $u \in \mathbb{R}^m$ is the input, $w_k \in \mathcal{W} \subseteq \mathbb{R}^l$ is an unknown but bounded disturbance and $N_0 \in \mathbb{N}^+$ is the number of time steps within the maneuver. We assume a maximum number of decision inputs $\bar{N}_{\max}$ (e.g. based on hardware capabilities). We aim to compute finite-time trajectories with a number of decision inputs $\bar{N} \leq \bar{N}_{\max}$, irrespective of N_0. The cost function to be minimized is of the form

$$J = \sum_{i=0}^{N-1} [x_i, u_i]^T H [x_i, u_i] + x_N^T P x_N, \tag{3}$$

where H, P are positive definite matrices. We assume that (i) the pair (A, B) is stabilizable and (ii) the sets $\mathcal{F}$, $\mathcal{X}_T$, and $\mathcal{W}$ are polytopes containing the origin. In the following, the fundamentals of tube MPC and move blocking are discussed.

a) Tube MPC: In order to deal with uncertainties, tube-based approaches [11], [12] provide a way to achieve stability guarantees while keeping the resulting optimization computationally tractable. The concept is based on an ancillary controller K that keeps the states of the uncertain system within a tube centered around the states of a nominal system

$$z_{k+1} = Az_k + Bv_k. \tag{4}$$

In [11], the tube size is constant and given by a Robust Positively Invariant (RPI) set $\mathcal{Z}$.

Definition 1 (Robust positively invariant (RPI) set): Let $A_K = A - BK$ be such that $x_{k+1} = A_K x_k$ is stable. A set $\mathcal{Z}$ is robust positively invariant if $A_K \mathcal{Z} \oplus \mathcal{W} \subseteq \mathcal{Z}$.

The following proposition establishes that states starting in this tube will in fact remain inside of it for all time under an appropriate control law.

Proposition 1 (Proposition 1 in [11]): Let $\mathcal{Z}$ be a RPI set for system (1). If $x_0 \in z_0 \oplus \mathcal{Z}$ and we choose $u_k = v_k - K(x_k - z_k)$, then $x_k \in z_k \oplus \mathcal{Z}$ for all $w_k \in \mathcal{W}$ and $k \in \mathbb{N}$.

Based on Proposition 1, it is therefore possible to ensure (2) despite the influence of w by performing the trajectory optimization on the nominal system with tightened constraints

$$\bar{\mathcal{F}} = \mathcal{F} \ominus (\mathcal{Z} \times K\mathcal{Z}), \ \bar{\mathcal{X}}_T = \mathcal{X}_T \ominus \mathcal{Z}. \tag{5}$$

The nominal system then acts as a reference for the ancillary controller K. By adding a constraint to assure that the measured state x_k lies within the tube $z_0 \oplus \mathcal{Z}$ at the start of the trajectory, we obtain the optimization problem $\mathbb{P}_0(x_k)$:

$$\min_{V, z_0} \quad \sum_{i=0}^{N-1} (z_i, v_i)^T H(z_i, v_i) + z_N^T P z_N$$

$$\text{s.t.} \quad x_k \in z_0 \oplus \mathcal{Z}, \ z_N \in \bar{\mathcal{X}}_T$$

$$z_{i+1} = Az_i + Bv_i, \ (z_i, v_i) \in \bar{\mathcal{F}}, \ i \in \mathbb{N}_0^{N-1}, \tag{6}$$

where i denotes the prediction time step. With the optimal solution at time k given as $V^*(k) = (v_0^*(k), \ldots, v_{N-1}^*(k))$, $z_0^*(k)$, the control law is

$$u_k = v_0^*(k) - K(x_k - z_0^*(k)). \tag{7}$$

b) Move blocking: The computational complexity of the optimization problem is heavily influenced by the number of decision variables. In regular MPC, this corresponds to the number of decision inputs within the horizon. One strategy to reduce complexity is to hold the inputs constant over a certain number of steps using a blocking matrix.

Definition 2 (Blocking matrix): A blocking matrix $M \in \mathbb{R}^{N \times \bar{N}}$ is defined as

$$M = \mathrm{diag}(\mathbf{1}_{s_1 \times 1}, \ldots, \mathbf{1}_{s_{\bar{N}} \times 1}) =: \mathbf{M}(s),$$

where the blocking vector $s = (s_1, \ldots, s_{\bar{N}})$ stores the individual blocking lengths $s_i \in \mathbb{N}^+$.

The matrix M relates the blocked decision input vector $\bar{U} := (\bar{u}_0, \ldots, \bar{u}_{\bar{N}-1})$ to the original one by $U = (M \otimes I_m)\bar{U}$. The number of inputs within the optimization is therefore reduced from Nm to $\bar{N}m$. For example, a system with $m = 1$ and $N = 4$ can be reduced to $\bar{N} = 2$ decision inputs via

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = M \begin{bmatrix} \bar{u}_1 \\ \bar{u}_2 \end{bmatrix} \text{ with } M = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, \ s = (3, 1).$$

III. Efficient Move Blocking for SHMPC

In this section, we present a time-varying move blocking scheme for SHMPC that efficiently utilizes a given maximum number of decision inputs, regardless of the horizon length. We then prove finite-time completion and recursive feasibility.

A. Optimization Problem Formulation

With move blocking, we can ensure that the number of decision inputs will not exceed a given bound $\bar{N}_{\max}$ by choosing an appropriate blocking matrix at each time step. In order to achieve robustness against disturbances, we combine move blocking with the tube-based MPC formulation (6) to arrive at $\mathbb{P}(x_k, M_k)$:

$$\min_{\bar{V}, z_0} \quad \sum_{i=0}^{N_k - 1} (z_i, v_i)^T H(z_i, v_i) + z_{N_k}^T P z_{N_k},$$

$$\text{s.t. } x_k \in z_0 \oplus \mathcal{Z}, \ z_{N_k} \in \bar{\mathcal{X}}_T,$$

$$V = (v_0, \ldots, v_{N_k - 1}) = (M_k \otimes I_m)\bar{V},$$

$$z_{i+1} = Az_i + Bv_i,$$

$$(z_i, v_i) \in \bar{\mathcal{F}}, \ i \in \mathbb{N}_0^{N_k - 1}, \tag{8}$$

where $\bar{V} = (\bar{v}_0, \dots, \bar{v}_{\bar{N}})$ is the reduced input vector, $N_k :=
N_0 - k$ and the control is given by (7). Since the horizon N_k
shrinks at every time step, the blocking matrix $M_k \in \mathbb{R}^{N_k \times \bar{N}}$
and the corresponding blocking vectors $s_k = (s_{1,k}, \dots, s_{\bar{N},k})$
are now time-varying. Optimal values are denoted by a star
as their superscript, e.g. $\bar{V}^*(k)$, $z_0^*(k)$. Note that while the
RPI set $\mathcal{Z}$ ensures invariance with respect to the trajectory
tracking error $x_k - z_0^*(k)$, invariance of the nominal trajectory
z_i is not required due to the use of shrinking horizons [4].
In the following, we outline a procedure to obtain blocking
matrices that ensure recursive feasibility and efficiently utilize
the available number of decision inputs.

B. Recursive Feasibility via Truncation

The move blocking scheme is built on the fact that, when
using shrinking horizons, a feasible solution at the next time
step is easily obtained by truncating the previous solution.

Proposition 2: Consider (1) controlled by (7). Let $V^*(k) =
(M_k \otimes I_m)\bar{V}^*(k)$ denote the optimal input trajectory of
$\mathbb{P}(x_k, M_k)$ for $k \in \mathbb{N}_0^{N_0-1}$. Let $R = [\mathbf{0}_{(N_k-1)\times 1}, I_{N_k-1}] \in
\mathbb{R}^{(N_k-1)\times N_k}$ denote a truncation matrix. If there exists an
input trajectory W such that

$$(M_{k+1} \otimes I_m)W = (R \otimes I_m)V^*(k) = (v_1^*, \dots, v_{N-1}^*), \quad (9)$$

then $\bar{V}(k+1) = W$ and $z_0(k+1) = z_1^*(k)$ are a feasible
solution to $\mathbb{P}(x_{k+1}, M_{k+1})$.

Proof: We first show that the truncated input trajectory
$(R \otimes I_m)V^*(k)$ is feasible at $k+1$. Based on Proposition
1, $x_{k+1} \in z_1^*(k) \oplus \mathcal{Z}$. Thus, $z_0(k+1) = z_1^*(k)$ satisfies the
initial state constraint at $k+1$. Since there is no disturbance
acting on the nominal system (4), truncating the previous input
trajectory $V^*(k)$ via $(R \otimes I_m)V^*(k)$ yields a feasible solution.
Therefore, if (9) holds, then $\mathbb{P}(x_{k+1}, M_{k+1})$ is feasible for
$\bar{V}(k+1) = W$ and $z_0(k+1) = z_1^*(k)$. ∎

If such a W exists for a particular blocking matrix M_{k+1}, we
refer to M_{k+1} as a feasible blocking matrix. A simple way to
construct a feasible blocking matrix is to choose $W = \bar{V}^*(k)$
and then truncate the first row of M_k as follows:

$$\begin{aligned}
(R \otimes I_m)V^*(k) &= (R \otimes I_m)(M_k \otimes I_m)\bar{V}^*(k) \\
&= (RM_k \otimes I_m)\bar{V}^*(k),
\end{aligned} \quad (10)$$

where we used $(A \otimes B)(C \otimes D) = AC \otimes BD$ for matrices
A, B, C, D of appropriate size. Thus, choosing $M_{k+1} = RM_k$
results in a feasible blocking matrix. However, repeatedly
truncating M eventually leads to zero columns, effectively
nullifying the corresponding reduced input. Consequently, the
available number of decision inputs is not fully utilized.

C. Recursively Feasible Interval Splits

In this section, we expand the previous strategy to enable
the efficient usage of the available decision inputs. Our goal
is to design a function $\Gamma(M_k)$ that only behaves like RM_k
when $s_{1,k} > 1$, i.e. when RM_k does not lead to a reduction
of effective decision inputs. However, when $s_{1,k} = 1$ and the
horizon length is larger than the maximum number of available
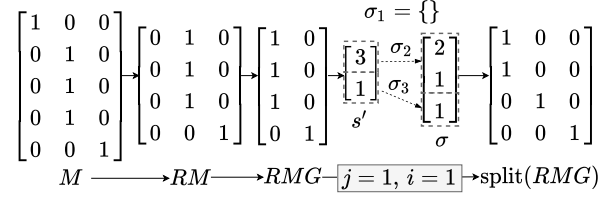decision inputs $\bar{N}_{\max}$, we split an existing blocking interval

Fig. 1. An example of (12) with $s = (1, 3, 1)$. The blocking vector s'
corresponds to the blocking matrix RMG. Since $j = 1$, σ_1 is empty.

instead of truncating it. The number of effective decision
inputs thus remains $\bar{N}_{\max}$ instead of reducing to $\bar{N}_{\max} - 1$.
Specifically, let

$$\Gamma(M_k) = \begin{cases} \Psi(M_k) & \text{if } s_{1,k} = 1, \ N_k > \bar{N}_{\max} \\ RM_k & \text{otherwise} \end{cases}, \quad (11)$$

where

$$\Psi(M_k) = \text{split}(RM_k G), \quad G = [\mathbf{0}_{(\bar{N}-1)\times 1}, I_{\bar{N}-1}]^T. \quad (12)$$

An example of (12) is shown in Fig. 1. The function $\Psi(M_k)$
first truncates M_k using R. Since $s_{1,k} = 1$, removing the first
row results in a matrix with only zeros in the first column. This
column is removed using G, leading to a blocking matrix with
$\bar{N} - 1$ columns. The result is then split using the function split,
which we define in the following. Recalling Definition 2, M
can be represented using its blocking vector s via $M = \mathbf{M}(s)$.
Our proposed splitting procedure can be written as:

$$\text{split}(M) = \mathbf{M}(\sigma), \quad \sigma = (\sigma_1, \sigma_2, \sigma_3) \quad (13)$$

where $\sigma_1 = (s_1, \dots, s_{j-1})$, $\sigma_2 = (s_j - i, i)$, and $\sigma_3 =
(s_{j+1}, \dots, s_{\bar{N}})$ with $i \in \mathbb{N}_1^{s_j-1}$ and $j \in \mathbb{N}_1^{\bar{N}}$ s.t. $s_j > 1$. The
index j is restricted to columns with $s_j > 1$ since a column
with $s_j = 1$ can not be split further. Note that for $j = 1$
and $j = \bar{N}$, the vectors σ_1 and σ_3 are empty, respectively.
The following Proposition establishes that $\text{split}(M)$ results in
a feasible blocking matrix.

Proposition 3: For a feasible blocking matrix $M \in \mathbb{R}^{N \times \bar{N}}$,
the blocking matrix $\text{split}(M) \in \mathbb{R}^{N \times \bar{N}+1}$ defined in (13) is
feasible.

Proof: By conjoining columns j and $j+1$ of $\text{split}(M)$,
the original blocking matrix M can be recovered. Therefore,
$M = \text{split}(M)D_j$ with $D_j = \text{diag}(I_{j-1}, (1,1), I_{\bar{N}-j})$. Since
M is feasible, Proposition 2 states that there exists a feasible
input trajectory $(M \otimes I_m)\bar{V}$ so that

$$\begin{aligned}
(M \otimes I_m)\bar{V} &= (\text{split}(M)D_j \otimes I_m)\bar{V} \\
&= (\text{split}(M) \otimes I_m)(D_j \otimes I_m)\bar{V}.
\end{aligned} \quad (14)$$

It follows that $(D_j \otimes I_m)\bar{V}$ can be chosen as a feasible input
trajectory for $\text{split}(M)$, thus it is a feasible blocking matrix. ∎

Proving recursive feasibility and finite time completion is now
straightforward.

Theorem 1: Consider (1) controlled by (7), wherein $v_0^*(k)$
and $z_0^*(k)$ are part of the optimal solution to $\mathbb{P}(x_k, \Gamma(M_{k-1}))$.
Suppose that a solution to $\mathbb{P}(x_0, M_0)$ exists for some M_0 with
an initial horizon length N_0. Then

1) $\mathbb{P}(x_k, \Gamma(M_{k-1}))$ is feasible for all $k \in \mathbb{N}_1^{N_0-1}$;
2) $x_{N_0} \in \mathcal{X}_T$.

Proof: Suppose that M_{k-1} is feasible for some $k \in \mathbb{N}_1^{N_0-1}$. If $M_k = \Gamma(M_{k-1})$ is feasible, item 1) holds by induction since the base case is true by assumption. For the case $N_{k-1} > \bar{N}$ and $s_{1,k-1} = 1$ of (11), we first need to show that $M_k' := RM_kG$ in (13) is feasible in order for Proposition 3 to be applicable. Since the first column of RM_{k-1} consists of zeros, we can add a zero column to M_{k-1}' to obtain $RM_{k-1} = [\mathbf{0}_{(N_{k-1}-1)\times 1}, M_{k-1}'] = M_{k-1}'G^T$. As (10) shows that is RM_{k-1} feasible, we obtain

$$
\begin{aligned}
(RM_{k-1} \otimes I_m)\bar{V}^*(k-1) \\
= (M_{k-1}'G^T \otimes I_m)\bar{V}^*(k-1) \\
= (M_{k-1}' \otimes I_m)(G^T \otimes I_m)\bar{V}^*(k-1),
\end{aligned}
\tag{15}
$$

i.e. M_k' is feasible with $W = (G^T \otimes I_m)\bar{V}^*(k-1)$ according to Proposition 2. By Proposition 3, $\mathrm{split}(M_{k-1}')$ is then also feasible. For all other cases, we have $M_k = RM_{k-1}$ which is again shown to be feasible in (10). $\Gamma(M_{k-1})$ is thus feasible any $k \in \mathbb{N}_1^{N_0-1}$. Item 2) follows directly: At time step $k = N_0$, since $\mathbb{P}(x_k, \Gamma(M_{k-1}))$ is feasible for all $k \in \mathbb{N}_1^{N_0-1}$, Proposition 1 guarantees that $x_{N_0} \in z_1^*(N_0 - 1) \oplus \mathcal{Z} \in \mathcal{X}_T$. ∎

IV. CONSTRAINT REDUCTION

For move blocking, the feasibility of the state trajectory between two blocked inputs is ensured by encoding that every state within the interval satisfies the constraints. A key observation is that this kind of formulation leads to highly redundant constraints and thus increases the complexity of the problem unnecessarily. In this section, we reformulate $\mathbb{P}$ to allow for constraint reduction and then present an approach to efficiently approximate the constraint sets. This facilitates the scalability of our SHMPC with respect to horizon length.

A. Transition Matrix Formulation

Let $\xi_0(i) = (z_i, \bar{v}_i) \in \bar{\mathcal{F}}$ denote the state-input pair at the start of a blocking interval $s_{i,k}$ (k is dropped in the following for brevity). The following auxiliary system describes the trajectories evolving within this interval:

$$
\xi_{j+1}(i) = \mathbf{A}\xi_j(i), \ \mathbf{A} = \begin{bmatrix} A & B \\ \mathbf{0}_{m \times n} & I_m \end{bmatrix}.
\tag{16}
$$

With $\tilde{A}(s_i) = A^{s_i}$ and $\tilde{B}(s_i) = \sum_{j=0}^{s_i-1} A^j B$, we then have $z_{i+s_i} = \tilde{A}(s_i)z_i + \tilde{B}(s_i)v_i$. Therefore, (8) can be equivalently stated as $\tilde{\mathbb{P}}(x_k, s_k)$:

$$
\begin{aligned}
\min_{\tilde{z}_0, \bar{V}} \quad & \sum_{i=0}^{\bar{N}-1} (\tilde{z}_i, \bar{v}_i)^T \tilde{H}(s_i)(\tilde{z}_i, \bar{v}_i) + \tilde{z}_{\bar{N}}^T P\tilde{z}_{\bar{N}} \\
\text{s.t.} \quad & \tilde{z}_0 \in x_k \oplus \mathcal{Z}, \ \tilde{z}_{\bar{N}} \in \mathcal{X}_T \\
& \tilde{z}_{i+1} = \tilde{A}(s_i)\tilde{z}_i + \tilde{B}(s_i)\bar{v}_i, \\
& (\tilde{z}_i, \bar{v}_i) \in \tilde{\mathcal{F}}(s_i), \ i \in \mathbb{N}_0^{\bar{N}-1}.
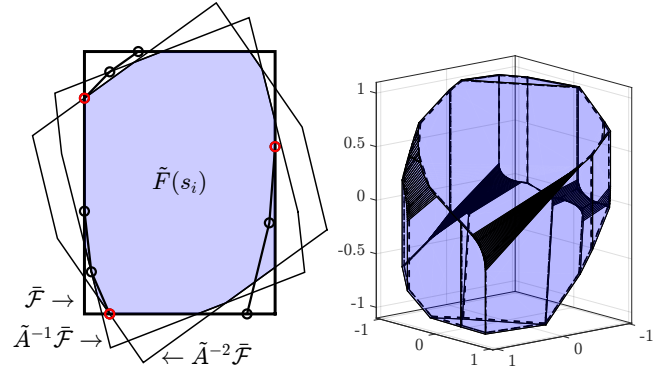\end{aligned}
\tag{17}
$$

Fig. 2. Left: An example of (18) with $s_i = 3$. Red circles denote initial states. Right: $\tilde{\mathcal{F}}$ (blue) and $\tilde{\mathcal{F}}_t$ (dashed) of a randomly generated set from Example 1 for $s_i = 30$.

where $\tilde{H}(s_i) = \sum_{j=0}^{s_i-1} (\mathbf{A}^j)^T H\mathbf{A}^j$ and

$$
\tilde{\mathcal{F}}(s_i) = \bigcap_{j=0}^{s_i-1} \mathbf{A}^{-j}\bar{\mathcal{F}},
\tag{18}
$$

which describes the state-input pairs that remain in $\bar{\mathcal{F}}$ for s_i-1 steps. Note that at time step s_i, the state is constrained by $\tilde{\mathcal{F}}(s_{i+1})$. An example of (18) is shown on the left side of Fig. 2. Note that trajectories starting in $\tilde{\mathcal{F}}$ do not leave $\bar{\mathcal{F}}$ within $s_i - 1$ steps. For polytopic sets, (18) is computed by stacking the matrices defining the halfspaces of $\mathbf{A}^{-j}\bar{\mathcal{F}}$, in which case we arrive at the same constraint matrices as in (8). However, this representation allows us to remove redundant halfspaces and thus lower the complexity of $\tilde{\mathbb{P}}$. Still, for large interval lengths s_i, even a non-redundant halfspace representation of $\tilde{\mathcal{F}}(s_i)$ is often still complex. In the next section, we introduce an approach to drastically reduce the constraints with only marginal loss of constraint set volume.

B. Constraint Set Approximation

To further reduce $\tilde{\mathcal{F}}(s_i) = \{x \mid Fx \leq f\}$, we construct a polytopic approximation $\tilde{\mathcal{F}}_t(s_i) = \{x \mid F_t x \leq f_t\}$ with fewer halfspaces. In order to ensure that all state-input pairs within $\tilde{\mathcal{F}}_t(s_i)$ also do not leave the original constraint set $\bar{\mathcal{F}}(s_i)$ within $s_i - 1$ steps, the approximation needs to be a subset of $\tilde{\mathcal{F}}(s_i)$. In other words, $\tilde{\mathcal{F}}_t(s_i)$ must be an inner-approximation of $\tilde{\mathcal{F}}(s_i)$, which of course may shrink the ROA. We therefore try to maximize the volume of $\tilde{\mathcal{F}}_t(s_i)$ while enforcing $\tilde{\mathcal{F}}_t(s_i) \subseteq \tilde{\mathcal{F}}(s_i)$ via optimization. To encode set containment within a linear program (LP), we use a generalized version of Farkas lemma:

Lemma 1 (Theorem 1 of [13]): Let $\mathbb{X} = \bar{x} + X\mathcal{H}_x$, $\mathbb{Y} = \bar{y} + Y\mathcal{H}_y$, where $\mathcal{H}_x = \{x \in \mathbb{R}^{n_x} | H_x x \leq h_x\}$ and $\mathcal{H}_y = \{y \in \mathbb{R}^{n_y} | H_y y \leq h_y\}$, where q_x, q_y are the number of rows of H_x and H_y, respectively. Then $\mathbb{X} \subseteq \mathbb{Y}$ if $\exists \Gamma \in \mathbb{R}^{n_y \times n_x}$, $\exists \beta \in \mathbb{R}^{n_y}$ and $\exists \Lambda \in \mathbb{R}_+^{q_y \times q_x}$ such that

$$
X = Y\Gamma, \ \bar{y} - \bar{x} = Y\beta, \Lambda H_x = H_y\Gamma, \ \Lambda h_x \leq h_y + H_y\beta.
$$

Proof: See Theorem 1 of [13] for a proof. ∎

The following LP uses a special case of Lemma 1 with $Y = I$ and $\bar{y} = 0$ to find an optimal scaling vector $\sigma \in \mathbb{R}_+^{n+m}$ with

$X = \text{diag}(\sigma)$ and translation $\bar{x} \in \mathbb{R}^{n+m}$ for $\tilde{\mathcal{F}}_t(s_i)$:

$$\min_{\bar{x}, \Lambda, \sigma} \quad ||\sigma||_1$$
$$\text{s.t.} \quad \Lambda F_t = F\text{diag}(\sigma), \quad \Lambda f_t \leq f - F\bar{x}, \tag{19}$$
$$\Lambda \geq 0, \quad \sigma > 0.$$

Since X is invertible by design, we can recover our inner-approximation as [13]

$$\tilde{\mathcal{F}}_t^*(s_i) = \{x \,|\, F_t\text{diag}(\sigma^*)^{-1}x \leq f_t + F_t\text{diag}(\sigma^*)^{-1}\bar{x}^*\}, \tag{20}$$

where the star denotes optimal values. The template $\tilde{\mathcal{F}}_t(s_i)$ can be any non-degenerate polytope of appropriate dimension. However, the choice of $\tilde{\mathcal{F}}_t(s_i)$ substantially influences the tightness and constraint reduction of the resulting approximation. In the following, we propose a useful heuristic to construct a $\tilde{\mathcal{F}}_t(s_i)$ that approximates the shape of $\tilde{\mathcal{F}}(s_i)$ efficiently. In particular, we use the structure of (18), but only consider a subset of the steps from 0 to $s_i - 1$:

$$\tilde{\mathcal{F}}_t(s_i) := \bigcap_{j \in \pi} \mathbf{A}^{-j}\bar{\mathcal{F}}, \ \pi \subseteq \mathbb{N}_0^{s_i-1}. \tag{21}$$

Remark 1: With the selection of π, a trade-off between volume and constraint reduction can be achieved. The more steps there are in π, the more halfspaces $\tilde{\mathcal{F}}_t(s_i)$ contains and the closer it gets to the shape of $\tilde{\mathcal{F}}(s_i)$. We found that $\pi = \{0, \lfloor \frac{s_i-1}{2} \rfloor, s_i - 1\}$, where $\lfloor \cdot \rfloor$ rounds a decimal to the nearest integer towards zero, often yields good results, even for large s_i.

Example 1: To assess the quality of the approximation in Remark 1, we generate random 100 second-order systems of the form (1) with $\mathcal{W} = \{0\}$, discretized with a sampling time of 0.05 seconds and constraint sets $\mathcal{F} = \mathcal{X} \times \mathcal{U}$. The state constraint set $\mathcal{X}$ is the convex hull of 62 random points while the input set $\mathcal{U} = \{u \,|\, ||u||_\infty < 1\}$ remains unchanged. All randomly generated coefficients are sampled from a uniform distribution and lie in the interval $[-1, 1]$. The results are shown in Fig. 3. The volume ratio between $\tilde{\mathcal{F}}(s_i)$ and $\tilde{\mathcal{F}}_t(s_i)$ shrinks with larger s_i, which is expected since the number of intersection operations in (21) is the same, regardless of s_i. As a result, the number of halfspaces of $\tilde{\mathcal{F}}(s_i)$ increases with s_i, while the complexity of $\tilde{\mathcal{F}}_t(s_i)$ stays relatively constant. Consequently, as shown in the right plot of Fig. 3, the relative constraint reduction grows as s_i increases. Note that even for large s_i, the constraint reduction retains over 96% of the volume of $\tilde{\mathcal{F}}(s_i)$ on average. Meanwhile, constraints are reduced by 80% with respect to the minimal representation and by over 94% with respect to the non-reduced representation.

A particular set from this study for $s_i = 30$ is shown on the right side of Fig. 2. It can be seen that a lot of halfspaces stem from the coupled state-input constraints on the edges of $\tilde{\mathcal{F}}(s_i)$. Our constraint reduction approach is able to approximate this shape efficiently.

V. NUMERICAL EXAMPLE

In this section, we showcase our approach with a 2-dimensional unmanned helicopter landing on an inclined surface. Though helicopter dynamics are highly nonlinear, it can
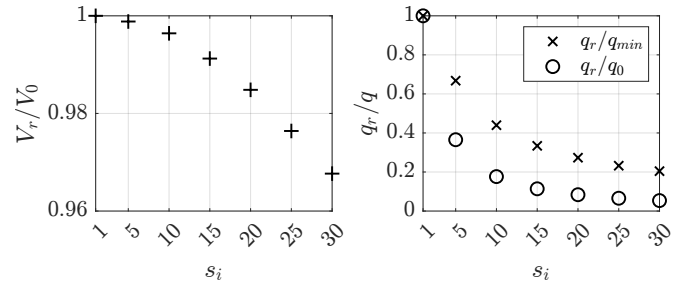

Fig. 3. Ratios of volumes V and number of halfspaces q between $\tilde{\mathcal{F}}$ and $\tilde{\mathcal{F}}_t^*$. Subscripts r, min and 0 correspond to the approximation, the minimal and the non-reduced representation of $\tilde{\mathcal{F}}$, respectively.

be shown that with some approximations, position becomes a differentially flat output [14]. We assume that there exists a sufficiently fast attitude controller so that we may consider the jerk (i.e. the 3rd derivative of position) as our input. The planning model can then be described as two independent integrator chains with $x = (p_x, v_x, a_x, p_z, v_z, a_z)$ where p, v and a denote position, velocity and acceleration, respectively. Discretizing the dynamics with a sampling time of $\tau = 0.02$ seconds yields $A = \text{diag}(A_x, A_z)$, $B = \text{diag}(B_x, B_z)$ and $w = (w_x, w_z)$, where

$$A_i = \begin{bmatrix} 1 & \tau & \tau^2/2 \\ 0 & 1 & \tau \\ 0 & 0 & 1 \end{bmatrix} B_i = \begin{bmatrix} \tau^3/6 \\ \tau^2/2 \\ \tau \end{bmatrix}, \ w_i = \begin{bmatrix} \tau^2/2 \\ \tau \\ 0 \end{bmatrix} \bar{w}_i$$

for $i = \{x, z\}$. Attitude tracking errors, model uncertainties as well as other disturbances appear as unknown accelerations $||\bar{w}_i||_\infty \leq 0.2$. The constraints $\mathcal{F}$ are given by

$$p_z \geq 0, (-4, -10) \leq (v_x, v_z) \leq (15, 5), -0.3v_x - v_z \leq 2$$
$$||[a_x, a_z, u_x, u_z]||_\infty \leq (4, 5, 3, 10), \tan(a)p_x - p_z \leq -1$$

where $a = 25°$ denotes the incline angle of the platform. The target set is given by

$$\mathcal{X}_T = \mathbf{R}(a)\{x \,|\, b_l + \mathbf{g} \leq x \leq b_u + \mathbf{g}\},$$
$$b_l = -(0.8, -1, 1, 0.9, 0.4, 4), \ b_u = (0.8, 2.2, 1, 0, 0.4, 4)$$

where $\mathbf{R}(a) = \mathcal{R}(a) \otimes I_3$, $\mathcal{R}(a)$ denotes the two-dimensional rotation matrix for an angle a and $\mathbf{g} = (\mathbf{0}_{5\times1}, 9.81)(I_6 - \mathbf{R}^T(a))$. We design the ancillary controller K via LQR with cost matrices $Q = \text{diag}(5I_4, 10I_2)$, $R = (0.1, 1)$. The RPI set $\mathcal{Z}$ is computed with the method of [15] with $\alpha = 10^{-6}$. Similar to [16], we compute the RPI set using zonotopes, which provide some computational advantages but do not alter the theoretical properties of the RPI set. We simulate three different versions of $\tilde{\mathbb{P}}$ with varying constraint sets $\tilde{\mathcal{F}}(s_i)$. MPC-0 denotes $\tilde{\mathbb{P}}$ with the full representation of $\tilde{\mathcal{F}}(s_i)$, which is equivalent to solving $\mathbb{P}$. Similarly, MPC-min and MPC-a represent $\tilde{\mathbb{P}}$ with the minimal representation of the constraint set and with its low-complexity approximation $\tilde{\mathcal{F}}_t(s_i)$, respectively. To penalize deviation from a setpoint $x_r = (-0.7, 1.4, 0.2, -0.4, -4.1, -0.9)$, we consider

$$\sum_{i=0}^{\bar{N}-1} (z_i - x_r, v_i)^T \tilde{H}(s_i)(z_i - x_r, v_i) + (z_{\bar{N}} - x_r)^T P(z_{\bar{N}} - x_r)$$

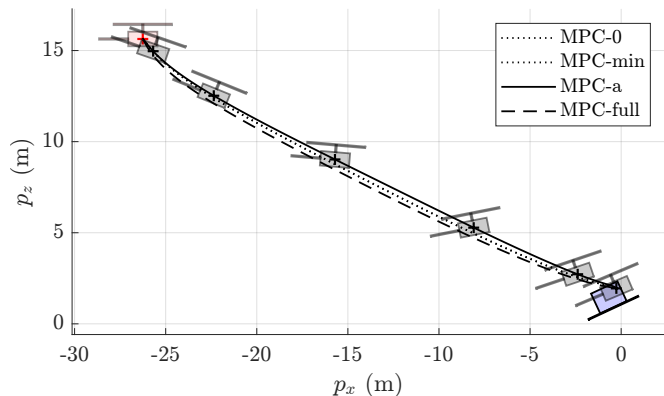|	open-loop		closed-loop	
	J/J_{full}	t_c (s)	J/J_{full}	t_c (s)
MPC-0	1.027	0.105	1.002	0.102
MPC-min	1.027	0.041	1.002	0.037
MPC-a	1.027	0.004	1.004	0.005
MPC-full	1	1.541	1	2.389

Fig. 4. Closed-loop simulation results of a helicopter landing. For MPC-a, the attitude and center of mass are plotted every second with the initial condition marked in red. The blue region denotes the projection of the target set. MPC-0 and MPC-min lie on top of each other.

as our cost function with $H = \mathrm{diag}(Q, R)$ and P as the corresponding solution to the discrete-time Riccati equation. The initial horizon length is set to $N_0 = 300$ with a maximum number of decision inputs $\bar{N}_{\max} = 10$ and initial blocking matrix $M_0 = \mathbf{1}_{30\times 1} \otimes I_{\bar{N}}$. When applying $\Psi(M_k)$ using Proposition 3, we choose $j^* = \arg\max_j s_j$ and $i = s_{j^*}/2$, i.e. we split the largest blocking interval through the middle. When s_{j^*} is odd, we round i to the nearest integer towards infinity. If there exist multiple possible j^*, we choose the largest one.

Since the number of blocked inputs is highest at the initial condition, we first compute the trajectory costs J and computation times t_c for nominal open-loop trajectories planned at $k = 0$. We benchmark MPC-0, MPC-min and MPC-a with a non-blocked SHMPC denoted by MPC-full. The averaged results of 50 randomly generated feasible initial conditions are shown in the open-loop column of Tab. I. Afterwards, the MPC algorithms are simulated in closed-loop. Their trajectories are shown in Fig. 4 and their costs and computation times are reported in the closed-loop column of Tab. I. Overall, the costs of blocked trajectories are only slightly larger than non-blocked costs. As expected, costs for MPC-min and MPC-0 are equal since in MPC-min, only redundant halfspaces are removed. Though MPC-a uses approximative constraint sets, its cost differs marginally from MPC-min and MPC-0 while reducing computation times by orders of magnitude. Note also that the cost ratio in closed-loop is smaller than in open-loop. This showcases the efficacy of utilizing free inputs to split blocking intervals and recovering some optimality when getting closer to the target. All computations are run in MATLAB on a laptop with an Intel Core i7-11850H using the solver osqp [17].

VI. CONCLUSION AND OUTLOOK

In this paper, we proposed a shrinking horizon Model Predictive Control approach with move blocking that is scalable to large prediction horizons. A function is designed that generates blocking matrices which ensure recursive feasibility while efficiently utilizing the available number of decision inputs. Furthermore, the optimal control problem is reformulated in a way that enables the reduction of the constraint set complexity. A low-complexity inner approximation is then obtained thorough optimization. The computational efficiency of the approach is demonstrated with a helicopter landing. In the future, we plan to confirm the computational efficiency in hardware experiments and extend the approach to continuous systems.

REFERENCES

[1] D. Q. Mayne, "Model predictive control: Recent developments and future promise," *Automatica*, vol. 50, no. 12, pp. 2967–2986, 2014.

[2] U. Eren, A. Prach, B. B. Koçer, S. V. Raković, E. Kayacan, and B. Açıkmeşe, "Model Predictive Control in Aerospace Systems: Current State and Opportunities," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 7, pp. 1541–1566, 2017.

[3] 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.

[4] A. Richards and J. P. How, "Robust variable horizon model predictive control for vehicle maneuvering," *Int. J. Robust Nonlinear Control*, vol. 16, no. 7, pp. 333–351, 2006.

[5] A. Richards and J. How, "Model predictive control of vehicle maneuvers with guaranteed completion time and robust feasibility," in *2003 American Control Conference (ACC)*, vol. 5, 2003, pp. 4034–4040.

[6] R. C. Shekhar and J. M. Maciejowski, "Robust variable horizon MPC with move blocking," *Systems & Control Letters*, vol. 61, no. 4, pp. 587–594, 2012.

[7] R. C. Shekhar, M. Kearney, and I. Shames, "Robust Model Predictive Control of Unmanned Aerial Vehicles Using Waysets," *Journal of Guidance, Control, and Dynamics*, vol. 38, no. 10, pp. 1898–1907, 2015.

[8] J. P. Koeln and A. G. Alleyne, "Two-level hierarchical mission-based model predictive control," in *2018 American Control Conference (ACC)*, 2018, pp. 2332–2337.

[9] S. S. Farahani, R. Majumdar, V. S. Prabhu, and S. E. Z. Soudjani, "Shrinking horizon model predictive control with chance-constrained signal temporal logic specifications," in *2017 American Control Conference (ACC)*, 2017, pp. 1740–1746.

[10] R. Cagienard, P. Grieder, E. Kerrigan, and M. Morari, "Move blocking strategies in receding horizon control," *Journal of Process Control*, vol. 17, no. 6, pp. 563–570, 2007.

[11] D. Mayne, M. Seron, and S. Raković, "Robust model predictive control of constrained linear systems with bounded disturbances," *Automatica*, vol. 41, no. 2, pp. 219–224, 2005.

[12] L. Chisci, J. Rossiter, and G. Zappa, "Systems with persistent disturbances: Predictive control with restricted constraints," *Automatica*, vol. 37, no. 7, pp. 1019–1028, 2001.

[13] S. Sadraddini and R. Tedrake, "Linear encodings for polytope containment problems," in *IEEE 58th Conference on Decision and Control (CDC)*, 2019, pp. 4367–4372.

[14] T. Koo and S. Sastry, "Differential flatness based full authority helicopter control design," in *IEEE 38th Conference on Decision and Control (CDC)*, vol. 2, 1999, pp. 1982–1987.

[15] S. Rakovic, E. Kerrigan, K. Kouramas, and D. Mayne, "Invariant approximations of the minimal robust positively Invariant set," *IEEE Transactions on Automatic Control*, vol. 50, no. 3, pp. 406–410, 2005.

[16] V. Raghuraman and J. P. Koeln, "Set operations and order reductions for constrained zonotopes," *Automatica*, vol. 139, 2022, 110204.

[17] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: An operator splitting solver for quadratic programs," *Mathematical Programming Computation*, vol. 12, no. 4, pp. 637–672, 2020.