2024 IEEE 63rd Conference on Decision and Control (CDC)
December 16-19, 2024. MiCo, Milan, Italy

Recursive feasibility guarantees in multi-horizon MPC

Varsha N. Behrunani[1,2], Philipp Heer[2], Roy S. Smith[1] and John Lygeros[1]

***Abstract*— Multi-Horizon model predictive control (MPC) is a method that uses time coarsening to increase the prediction horizon by using several models, each with a different sampling time that gradually increases later in the horizon. This facilitates having a longer prediction interval without significantly impacting the computational load or compromising the response time. However, the use of models with different granularity make guaranteeing recursive feasibility challenging as conventional approaches cannot be applied directly. This work proposes a constraint tightening strategy to enforce recursive feasibility in time-invariant multi-horizon MPC schemes. The state constraint in the optimization is replaced by adaptive state and input constraint at each time step that depend on the sampling time to ensure that the trajectory remains feasible between two sampling points even as the sampling time increases. An extensive numerical study illustrates the effectiveness and scalability of our approach and compares its performance to standard MPC and multi-horizon MPC controllers without any constraint tightening.**

Keywords: Multi-horizon MPC, constraint tightening, optimization, recursive feasibility

I. INTRODUCTION

MPC has been well established in various applications for the optimal control of constrained systems due its ability to explicitly incorporate hard state and input constraints in the optimization. MPC uses a dynamic model of a system along with real-time feedback to repeatedly optimize control inputs by making predictions of future responses. MPC operates in a receding horizon manner wherein at each time step, the optimal control inputs are obtained by solving a constrained finite horizon optimal control problem for the current state of the plant and only the first control input is applied to the system. MPC has been shown to improve the performance of the overall system as the repeated optimization brings feedback into the process through the measurements and allows the controller to continuously adapt to updated measurements and estimations, and suppress the effect of model mismatch, disturbances, and exogenous inputs [2] [3].

The key trade-offs of the MPC problem are the sampling time and prediction horizon. Ideally, one would like to make the prediction horizon as long as possible to give more preview to the decisions with an small sampling time to detect changes in the system and provide a fast response; on the other hand, doing so requires solving to a larger optimisation problem which is more difficult to solve in real time [8]. Methods such as move blocking MPC have previously been used to extend the horizon while reducing the computational complexity by fixing the control inputs to be constant over several steps in the horizon [11]. Multi-horizon MPC (MHMPC) addresses this trade-off by using models of different granularities to extend the horizon without increasing the computational load, that is, covering a longer horizon span with fewer states. It results in less conservatism in the solution than move blocking MPC as it does not restrict the inputs. Each model predicts system responses for different parts of the horizon. These predictions are combined to predict system responses of the entire horizon. The sampling time is small at the start of the horizon thereby maintaining a high resolution and gradually increases later in the horizon which allows us to increase the length of the prediction interval. MHMPC has been proposed in various works [4][5][9] for several applications and further extended in [8] with distributed MPC. This approach relies on the exponential decay of sensitivity [6], [7] property of optimal control problems which states that the impact of perturbations in the future on the current control action is inversely proportional to how far in the future it occurs.

One key aspect of MPC is recursive feasibility which states that if the solution of the MPC exists at the initial time, then the MPC remains feasible at all future time steps and the closed loop trajectories never reaches infeasible states [18]. It has been studied extensively for discrete-time systems and standard move-blocking MPC with different prediction horizons [10]. Recursive feasibility is typically enforced by constraining the terminal state of the finite-horizon optimal control problem to a controlled invariant set [12][13]. However, this strategy does not work for MHMPC as it does not ensure adherence to the state constraints between two states in the horizon with increasing sampling time. The optimization is implemented with a receding horizon and propagated forward with the smallest sampling time so it is crucial to guarantee that the trajectory does not violate the state constraints at any sampling time and does not steer into a direction where the problem becomes infeasible at a later time. Another common technique is the soft constraint method where the state constraints are relaxed or simply removed for some portion of the prediction horizon and the size of the violation is penalized in the cost function [15]. This however may result in large closed-loop violations of the state constraints which are unsuitable for application where state constraints are hard.

In [4], the authors propose a strategy to ensure recursive feasibility by imposing that the state at the end of the 1st sub-interval with the smallest sampling time should be in the maximally control invariant set. While this method does ensure recursive feasibility, it is an extremely restrictive approach and gravely reduces the feasible set and results in an extremely conservative solution. Other approaches

This research is supported by the SNSF through NCCR Automation (Grant Number 180545).

[1]V. Behrunani, J. Lygeros and Roy S. Smith are with the Automatic Control Laboratory, ETH Zurich, Switzerland (email: bvarsha@ethz.ch, jlygeros@ethz.ch, rsmith@ethz.ch)

[2]V. Behrunani and P. Heer are with the Urban Energy Eystems Laboratory, Swiss Federal Institute of Material Science and Technology (Empa) Dubendorf, Switzerland.(email: varsha.behrunani@empa.ch, philipp.heer@empa.ch)

for ensuring recursive feasibility for sub-optimal MPC are presented in [16][17]. However, these techniques rely on time invariant system matrices in the optimization problem or are conservative in their approach as they consider the maximum worst case input.

The main contribution of this paper is to provide a practical solution to the problem of guaranteeing recursive feasibility for an MHMPC scheme via constraint tightening. We formulate the problem of MHMPC to showcase how it differs from standard MPC and propose a strategy to compute a tightened constraint set to replace the original state constraints in the optimization. Constraint tightening approaches to guarantee recursive feasibility have previously been adopted for continuous time MPC [22] as well as to mitigate disturbances and uncertainty in robust MPC [20] and stochastic MPC [21]. In this study, we impose adaptive joint state and input constraints that get more restrictive as the sampling time gets larger. For each sampling time, a different constraint set is determined which ensures that the trajectory is also feasible when the optimal input is sampled with the smallest sampling time. In a numerical example, we compare the performance of the MHMPC both with and without the tightened constraint set to the standard MPC illustrating the need for constraint tightening in MPC and the impact of the additional constraints on the cost and computation time.

The paper is structured as follows. Section II presents the problem setting, and Section III presents the MHMPC scheme. Section IV presents our main result, a sufficient condition for recursive feasibility and the modified constrained set. Our strategy is illustrated by numerical examples in Section V and conclusions are given in Section VI.

II. Preliminaries

A. Notation

The set of real numbers and integers are denoted by $\mathbb{R}$ and $\mathbb{Z}$, respectively. $\mathbb{Z}_{a:b}$ denotes the set of numbers $\{a, a+1, ..., b\}$. The set of all non-negative integers is denoted $\mathbb{Z}_{\geq 0}$. $\mathbb{R}^n$ denotes the n-dimensional real vector space.

B. System Model

Consider a discrete-time, time-invariant linear system with n states and m inputs:

$$x_{k+1} = A_k x_k + B_k u_k \tag{1}$$

where $x_k \in \mathbb{R}^n$ and $u_k \in \mathbb{R}^m$ are the state and input variables at time k, respectively. $A_k \in \mathbb{R}^{n\times n}$ and $B_k \in \mathbb{R}^{n\times m}$ are the time-varying state and input matrices of appropriate dimensions that may depend on sampling time.

The states and control inputs are required to satisfy the general constraints:

$$\begin{aligned} x_k \in \mathbb{X} \subseteq \mathbb{R}^n, \quad \forall k \in \mathbb{Z}_{\geq 0} \\ u_k \in \mathbb{U} \subseteq \mathbb{R}^m, \quad \forall k \in \mathbb{Z}_{\geq 0} \end{aligned} \tag{2}$$

where $\mathbb{X}$ and $\mathbb{U}$ are polytopic state and input constraint sets containing the origin in their interior:

$$\begin{aligned} \mathbb{X} = \{x \in \mathbb{R}^n | G_x x \leq W_x\}, \\ \mathbb{U} = \{u \in \mathbb{R}^m | G_u u \leq W_u\}. \end{aligned} \tag{3}$$

The control objective is to determine an input trajectory, which minimizes the finite horizon quadratic cost over the total prediction time T divided into N prediction time steps

$$J_T = \sum_{k=0}^{N-1} \left(x_k^\top Q_k x_k + u_k^\top R_k u_k \right) + x_N^\top Q_H x_N \tag{4}$$

where $Q_k \in \mathbb{R}^{n\times n}, Q_r \succ 0$, and $R_k \in \mathbb{R}^{m\times m}, R_k \succ 0$ are possibly time varying state and input cost matrices, respectively.

Definition 2.1: Controlled Invariance A non-empty set $\mathbb{X}_\mathrm{I} \subseteq \mathbb{R}^n$ is a controlled invariant set of the system (1) if and only if for all $x \in \mathbb{X}_\mathrm{I}$, there exist $u \in \mathbb{U}$ such that $Ax + Bu \in \mathbb{X}_\mathrm{I}$ [14][13]. The *maximal controlled invariant* set of the system (1) within $\mathbb{X}$ is defined by $\mathbb{X}_\mathrm{CI} \equiv \{x_0 \in \mathbb{X} \mid \exists u_k \in \mathbb{U} \text{ s.t. } x_{k+1} = Ax_k + Bu_k \in \mathbb{X} \quad \forall k \in \mathbb{Z}_{\geq 0}\}$.

III. Multi-Horizon MPC

MHMPC is a variant of the MPC in which the prediction horizon is divided into multiple sub-intervals, each with a different sampling time. Several dynamic models are used, each discretized with a different sampling time that gradually increases further down the horizon. Each model predicts system responses for different parts of the horizon and the predictions are combined to predict system responses over the entire horizon.

We consider H sub intervals labeled by $i \in \mathbb{H} := \{1, \ldots, H\}$. The system and cost matrices differ in each sub-interval owing to the different sampling time and $\{A_i, B_i, Q_i, R_i\}$ are the corresponding matrices associated with the sub-interval i with the sampling time t_i. Let $\mathbb{K}_i$ be a set of all the time steps k within the sub-interval i and N_i be the cardinality of set $\mathbb{K}_i$. The set $\mathbb{K}_i$ only includes the initial state for each sub-interval and excludes the terminal state as the terminal state is included in the set of the next sub-interval. The terminal state of the complete optimization $k = N$ is not included in any sub-interval. The total time spanned by sub-interval i is T_i. Hence, $\sum_{i\in\mathbb{H}} N_i = N - 1$ and $\sum_{i\in\mathbb{H}} T_i = T$ and the N non-uniform prediction time steps cover the complete prediction interval T. Here, $\mathbb{X}_\mathrm{CI} \subseteq \mathbb{X}$ is the control invariant set.

The MHMPC problem can be compactly written as:

$$\begin{aligned} \min_{\{x_k,u_k\}_{k=0}^N} \quad & \sum_{i\in\mathbb{H}} \left(\sum_{k\in\mathbb{K}_i} \left(x_k^\top Q_i x_k + u_k^\top R_i u_k \right) \right) + x_N^\top Q_H x_N \\ \text{s.t.} \quad & x_{k+1} = A_i x_k + B_i u_k, \quad \forall k \in \mathbb{K}_i, \ \forall i \in \mathbb{H} \\ & x_k \in \mathbb{X}, \quad \forall k \in \mathbb{Z}_{0:N} \\ & u_k \in \mathbb{U}, \quad \forall k \in \mathbb{Z}_{0:N-1} \\ & x_N \in \mathbb{X}_\mathrm{CI}, \end{aligned} \tag{5}$$

Let the sampling time of the 1st sub-interval $i = 1$ be t_1 and assume that the sampling time of the i^{th} sub-interval is $t_i = \alpha_i t_1$ ($\alpha_1 = 1$) where $\alpha_i \in \mathbb{Z}_{\geq 1}$ and increasing such that $\alpha_1 < \alpha_2 \cdots < \alpha_{H-1} < \alpha_H$. Then set,

$$\begin{aligned} A_i = A_1^{\alpha_i}, \quad B_i = \sum_{j=0}^{\alpha_i - 1} A_1^j B_1, \\ Q_i = \alpha_i Q_1, \quad R_i = \alpha_i R_1 \end{aligned} \tag{6}$$

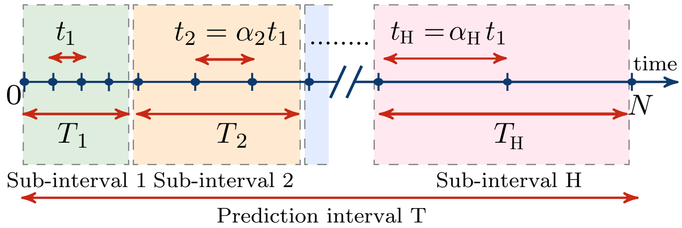

Fig. 1. Sketch of the MHMPC control scheme. A detailed model is used in sub-interval 1 with a small sampling time t_1 predicting the outcome for T_1. The models gets progressively coarser as the sampling time increases in subsequent sub-intervals as $\alpha_1 \leq \alpha_2 < \alpha_3 < \cdots < \alpha_H$. The largest sampling time of $t_{\rm H}$ is used in final sub-interval H.

This assumption only requires the sampling time to be increasing ($t_1 < t_2 < \cdots < t_{\rm H}$) and does not impose any restrictions on the number of time steps and length of each sub-interval. Fig. 1 illustrates how MHMPC is formulated by splitting the prediction interval into several sub-intervals and the corresponding sampling times and time steps associated with each sub-interval.

A. Recursive Feasibility

Definition 3.1: Recursive Feasibility An MPC problem is called recursively feasibly if and only if for all initially feasible states x_0, the closed loop trajectory remains feasible for all time [10].

Recursive feasibility implies that if an MPC optimization has a solution at $k = 0$, then there exists a solution at $k = 1, \ldots, \infty$. This is classically obtained by introducing a terminal constraint at the end of the prediction horizon of the form $x_N \in \mathbb{X}_{\rm CI}$. Summarising briefly, starting from an initially feasible state, the optimal input sequence computed and the resulting trajectory remains feasible throughout the prediction interval even as the horizon moves forward. The terminal constraint guarantees all future states stay within the controlled invariant set after the end of the prediction interval and a feasible control input can be computed. Hence, a feasible trajectory at time k can be extended by appending a feasible input at its end at time $k+1$. Therefore, the optimization is recursively feasible.

Conversely, terminal constraints are not sufficient for enforcing recursive feasibility for MHMPC. Although the state constraints are satisfied at each time step, the trajectory may not be feasible with a smaller sampling time in between two time steps when the sampling time is large later in the horizon. As the horizon moves forward in each time step, the optimal control input computed at the first time step can no longer be applied to the system directly with the smallest sampling time as it does not guarantee state constraint satisfaction in the subsequent sub-intervals. This can eventually result in a trajectory reaching an state in the future where the optimization has no feasible solution. This is also illustrated using the numerical example in Fig. 5.

IV. Recursive Feasibility via Constraint Tightening

This section describes the proposed strategy to ensure recursive feasibility for the MHMPC scheme. To achieve this, a tightened constraint set is used that guarantees recursive feasibility and stability of the control problem as the sampling times increase.

The goal is to design a constraint set $\mathbb{X}_{\rm RF} \subseteq \mathbb{X}$ such that the MHMPC is recursively feasible if and only if $\{x\}_{k=0}^{T} \in \mathbb{X}_{\rm RF}$. The constraint set requires that for the optimal input, the state constraints are satisfied at time steps of duration t_1 throughout the prediction interval irrespective of the sampling time. The trajectory remains feasible between two time steps in all sub-intervals, T_i, when propagated forward with time steps of t_1. This ensures that the optimal input computed at the initial time step can directly be applied at each step without violating any state constraints even as the horizon shifts forward by t_1 at the next iteration of MPC. Hence, the same solution continues to remain feasible within the complete prediction interval. Similar to the standard MPC, the terminal constraint imposes that the final state must lie in a control invariant set so that the state trajectory stays within this set. The control invariant set is computed with the dynamics corresponding to the sampling time t_1. Hence, recursive feasibility is guaranteed for MHMPC.

Consider the MHMPC problem formulated in Section III. In each sub-interval $i \in \mathbb{H}_{\geq 2}$ with sampling time t_i, the time t_i can be divided into α_i intervals of time t_1 which results in $\alpha_i - 1$ intermediate time steps between any two points when discretised with t_1. To ensure the constraints are met with the smallest sampling time, the following constraints are imposed on all states k associated with sub interval i:

$$\left(A_1^s x_k + \sum_{j=0}^{s-1} A_1^j B_1 u_k \right) \in \mathbb{X}, \quad \forall s \in \mathbb{Z}_{1:\alpha_i}, \forall k \in \mathbb{K}_i \quad (7)$$

These constraints ensure that any state within the sub-interval i when propagated forward with the system dynamics associated with t_1 will continue to satisfy the state constraints until the next state is reached. At $s = \alpha_i$, the constraint (7) when imposed on all the sub-intervals results in the following:

$$\begin{aligned} &\left(A_1^{\alpha_i} x_k + \sum_{j=0}^{\alpha_i - 1} A_1^j B_1 u_k \right) \in \mathbb{X}, \ \forall k \in \mathbb{K}_i, \ \forall i \in \mathbb{H} \\ &\text{From (6)} \\ &\equiv (A_i x_k + B_i u_k) \in \mathbb{X}, \ \forall k \in \mathbb{K}_i, \ \forall i \in \mathbb{H} \\ &\equiv x_{k+1} \in \mathbb{X}, \ \forall k \in \mathbb{K}_i, \ \forall i \in \mathbb{H} \\ &\equiv x_k \in \mathbb{X}, \ \forall k \in \mathbb{Z}_{1:N} \end{aligned} \quad (8)$$

Hence, starting from an initially feasible state $x_0 \in \mathbb{X}$, the constraint set (7) is a subset of the state constraint set in the MHMPC optimization (5).

In addition to the states, the constraints (7) at each step also depend on the corresponding input. To integrate this input constraints into the MHMPC problem, we define the following joint input and state constraint for each sub interval i by substituting the constraint definition from (3) in (7):

$$\mathbb{X}_i^{\rm RF} \equiv \left\{ \begin{array}{c} x \in \mathbb{R}^n \\ u \in \mathbb{R}^m \end{array} \middle| \begin{array}{r} G_x\left(A_1^s x + \sum_{j=0}^{s-1} A_1^j B_1 u \right) \leq W_x, \\ \forall s \in \mathbb{Z}_{1,\alpha_i} \end{array} \right\}.$$

The constraint set can then be compactly written as (9). The matrices $\widetilde{G_i}$ and $\widetilde{W_i}$ can be computed offline for each sub-interval i and remain constant within a sub-interval.

$$\mathbb{X}_i^{\mathrm{RF}} \equiv \left\{ \begin{array}{l} x \in \mathbb{R}^n \\ u \in \mathbb{R}^m \end{array} \middle| \; \widetilde{G_i} \begin{bmatrix} x \\ u \end{bmatrix} \leq \widetilde{W_i} \right\}, \tag{9}$$

$$\text{where } \widetilde{G_i} = \begin{bmatrix} G_x A_1 & G_x B_1 \\ \vdots & \vdots \\ G_x A_1^s & G_x \sum_{j=0}^{s-1} A_1^j B_1 \\ \vdots & \vdots \\ G_x A_1^{\alpha_i} & G_x \sum_{j=0}^{\alpha_i - 1} A_1^j B_1 \end{bmatrix} \forall s \in \mathbb{Z}_{1,\alpha_i},$$

$$\widetilde{W_i} = \left.\begin{bmatrix} W_x \\ \vdots \\ W_x \end{bmatrix}\right\} \alpha_i \text{ rows}$$

The resulting recursively feasible MHMPC (rf-MHMPC) problem is:

$$\begin{aligned} \min_{\{x_k, u_k\}_{k=0}^N} \quad & \sum_{i \in \mathbb{H}} \left(\sum_{k \in \mathbb{K}_i} \left(x_k^\top Q_i x_k + u_k^\top R_i u_k \right) \right) + x_N^\top Q_H x_N \\ \text{s.t.} \quad & x_{k+1} = A_i x_k + B_i u_k, \quad \forall k \in \mathbb{K}_i, \ \forall i \in \mathbb{H} \\ & (x_k, u_k) \in \mathbb{X}_i^{\mathrm{RF}}, \quad \forall k \in \mathbb{K}_i, \ \forall i \in \mathbb{H} \\ & u_k \in \mathbb{U}, \quad \forall k \in \mathbb{Z}_{0:N-1} \\ & x_0 \in \mathbb{X}, \\ & x_N \in \mathbb{X}_{\mathrm{CI}} \end{aligned} \tag{10}$$

Note that control invariant set $\mathbb{X}_{\mathrm{CI}}$ in (9) is computed with the dynamics corresponding to the sampling time t_1. The state constraint in (5) is replaced by the rf-MHMPC constraints (9) resulting in (10). This structure increases the number of constraints compared to the original MHMPC formulation and decreases the number of decision variables compared to standard MPC. Constraint reduction methods proposed in the literature [23] can be applied to reduce the resulting constraint set to increase the computational efficiency of the optimization.

The modified constraint set replaces the state constraints in the original MHMPC problem with joint state and input constraints which increases the complexity of the constraints. Ideally, the additional constraint would only depend on the states independent of the inputs and the resulting linear matrices of the tightened constraints could be computed offline. This would minimize the additional computational overload and complexity. One strategy to achieve this would be to restrict the states in each sub-interval by considering the worst case input. This, however, may result in a very small feasible set in the later sub-intervals and an extremely conservative sub-optimal solution or the problem to become infeasible and would therefore restrict the maximum sampling time that can be used. Another approach would be to restrict the maximum input that can be used in each sub-interval. Exploring these strategies is a topic of future work.

V. Numerical Example

In this section, we demonstrate the proposed approach in simulation for guaranteeing recursive feasibility and the resulting impact on the performance by comparing the rf-MHMPC to two standard MPC and the MHMPC without any constraint tightening. All simulations were performed in MATLAB 2022a with YALMIP using the Gurobi solver. Consider the following linear mass spring damper system [19] shown in Fig. 2(a) with:

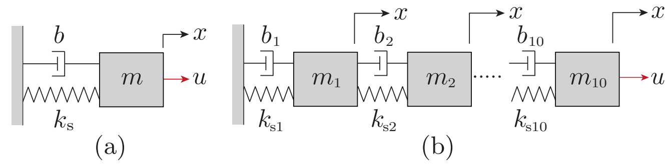

Fig. 2. Illustration of the mass spring damper system used in simulation: (a) Single mass spring damper (b) Ten mass spring dampers in series.

$$A = \begin{bmatrix} 0 & 1 \\ \frac{-k_s}{m} & \frac{-b}{m} \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} \tag{11}$$

where $m = 0.5\,\mathrm{kg}$ is the mass, $k_s = 0.5\,\mathrm{N/m}$ is the spring constant and $b = 0.25\,\mathrm{Ns/m}$ is the damping constant. The system is subject to the state and input constraint sets:

$$\begin{aligned} \mathbb{X} &\equiv \left\{ x \in \mathbb{R}^2 \mid |x| \leq 20 \right\}, \\ \mathbb{U} &\equiv \left\{ u \in \mathbb{R} \mid |u| \leq 0.5 \right\}. \end{aligned} \tag{12}$$

The system is discretized with the smallest sampling time $t_1 = 0.05\,\mathrm{s}$. For MHMPC, we consider 5 sub-intervals with sampling times ranging from up $0.05\,\mathrm{s}$ to $0.8\,\mathrm{s}$ over the complete prediction interval of $4.8\,\mathrm{s}$. The number of time steps in each sub interval with different sampling times is presented in Table I and the total number of states N, is 21 including the terminal state. The system is simulated for a total of $20\,\mathrm{s}$. The initial state x_0 is [18,-14].

Sub-interval i	t_{i} [s]	N_{i}	$\mathbb{K}_{\mathrm{i}}$	T_{i} [s]
1	0.05	4	[0,3]	0.2
2	0.1	6	[4,9]	0.6
3	0.2	4	[10,13]	0.8
4	0.4	4	[14,17]	1.6
5	0.8	2	[18,19]	1.6

TABLE I

Specifications of all the sub-intervals in MHMPC covering a total prediction interval of 4.8s.

Fig. 3 compares the optimal trajectory obtained using MHMPC and rf-MHMPC with the two standard MPC controllers with a sampling time of t_1. A standard MPC controller is used with the same prediction interval of $4.8\,\mathrm{s}$ which has almost 4 times as many decision variables as the MHMPC as well as with a smaller prediction interval of $1\,\mathrm{s}$ that has the same number of decision variables as the MHMPC optimization problem. The figure shows that the trajectory of the MHMPC and rf-MHMPC controllers match the optimal standard MPC controller with the small sampling time and a long prediction interval whereas decreasing the prediction interval in the standard MPC to reduce the number of decision variables to match the MHMPC results in a different trajectory.

Table II compares the total closed loop cost and average computation time of these controllers as well as other

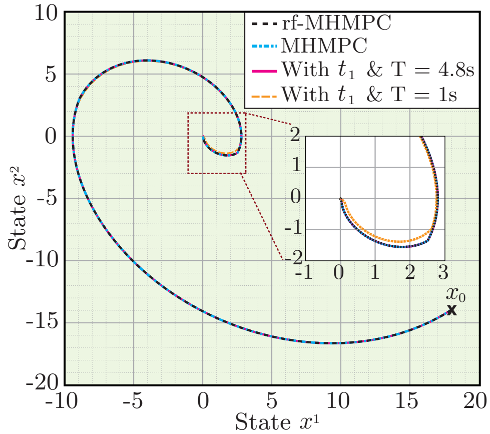

Fig. 3. Optimal trajectory obtained using MHMPC, rf-MHMPC and two standard MPC controllers with a sampling time of t_1. The green area is the feasible states and x^1 and x^2 are the first and second state respectively.

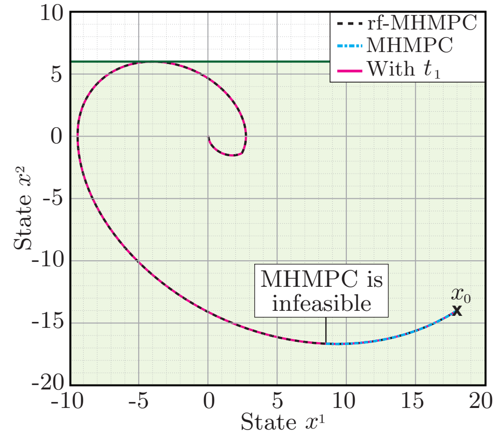

Fig. 4. Optimal trajectory obtained using MHMPC, rf-MHMPC and standard MPC controllers with a sampling time of t_1 and prediction interval of 4.8s with the reduced $\mathbb{X}$.

standard MPC controllers for the complete simulation period of 20s. The controllers were chosen to have either the same prediction horizon or the same number of decision variables as the MHMPC controllers. The cost is computed by summing the cost of the first state and input at each time step as it is the only input applied to the system. Hence, only the cost for optimal input sequence and optimal state trajectory is considered at the end of the simulation. The average computation time (Avg. time) is obtained by dividing the total computation time of the complete simulation period by the total number of time steps. For example, with a sampling time $t_1 = 0.05$s and simulation time of 20s, the total number of time steps is (20/0.05) = 400. Then, the Avg. time[s] = (Total computation time[s]/400).

The table shows that MHMPC and rf-MHMPC achieve the lowest cost similar to the best standard MPC controller whereas the avg. time is much smaller. The computation time matches the time of the standard MPC controllers with the same number of decision variables, however, these controller have a higher cost due to a shorter prediction interval and increased sampling time. While the rf-MHMPC controller requires a slightly longer time than MHMPC due to the presence of additional constraints in the optimization, the time increase is negligible.

			Original $\mathbb{X}$ in (12)		Reduced $\mathbb{X}$ in (13)	
Method	T	N	Cost	Avg. time[s]	Cost	Avg. time[s]
MPC t = 0.05	1	21	1.4507e4	0.0059	Infeasible	
	4.8	97	1.4499e4	0.0092	1.4524e4	0.0092
MPC t = 0.1	2	21	1.4765e4	0.0052	1.4786e4	0.0053
	4.8	49	1.4763e4	0.0068	1.4786e4	0.0069
MPC t = 0.2	4.8	25	1.5297e4	0.0061	1.5334e4	0.0055
MHMPC	4.8	21	1.4499e4	0.0053	Infeasible	
rf-MHMPC	4.8	21	1.4499e4	0.0059	1.4518e4	0.0060

TABLE II

COMPARISON OF COST AND COMPUTATION TIME OF DIFFERENT STANDARD MPC CONTROLLERS WITH MHMPC AND RF-MHMPC FOR BOTH THE ORIGINAL $\mathbb{X}$ AND THE REDUCED $\mathbb{X}$.

To see the benefits of rf-MHMPC over MHMPC, consider a reduced state constraint set:

$$\mathbb{X} \equiv \left\{ x \in \mathbb{R}^2 \middle| |x^1| \leq 20, -20 \leq x^2 \leq 6 \right\}. \qquad (13)$$

Fig. 4 shows the optimal trajectory obtained with the reduced $\mathbb{X}$. The performance of the rf-MHMPC and the standard MPC with prediction interval of 4.8s remains largely unchanged, however, the MHMPC results in a trajectory that eventually reaches a point where the optimization becomes infeasible. A better understanding of why this happens can be obtained by observing the optimal trajectory computed by both the MHMPC and the rf-MHMPC optimization at the initial time step in the first iteration shown in Fig. 5. For MHMPC, while the solution is within the state constraint set, it is not a recursively feasible solution. The optimal input applied to the system with a sampling time of $t_1 = 0.05s$ would result in the state constraints being violated for the state x^2 when the sampling time is larger. The optimization eventually becomes infeasible when such inputs are repeatedly applied to the system at each time step. Thus, the MHMPC controller does not guarantee recursive feasibility. Contrarily, the additional constraints in the rf-MHMPC ensure that any solution of the optimization is also implementable and satisfies constraints at every t_1 even as the sampling time in the subsequent sub-interval increases ($t_i > t_1$) and therefore, the optimization remains feasible.

Comparing the cost and computation time of the controllers with the reduced $\mathbb{X}$ in Table II shows that some standard MPC controllers are also infeasible due to their small prediction horizon. The cost of the rf-MHMPC matches the optimal standard MPC controller cost whereas the average computation time is still significantly lower and remains unchanged compared to the original $\mathbb{X}$. This shows the superior performance of the rf-MHMPC.

Finally, the performance of the rf-MHMPC was also compared to the standard MPC for a larger system with 10 mass spring dampers placed in series as shown in Fig.2(b), giving rise to 20 states. The system is simulated for a total of 60s to allow all states to converge. The results are shown in Table III. rf-MHMPC achieves the lowest cost and requires on average less than a third of the computation time compared to the standard MPC controller with sampling time t_1. The average computation time is comparable to the standard MPC controllers with the same N. That however lead to a larger cost.

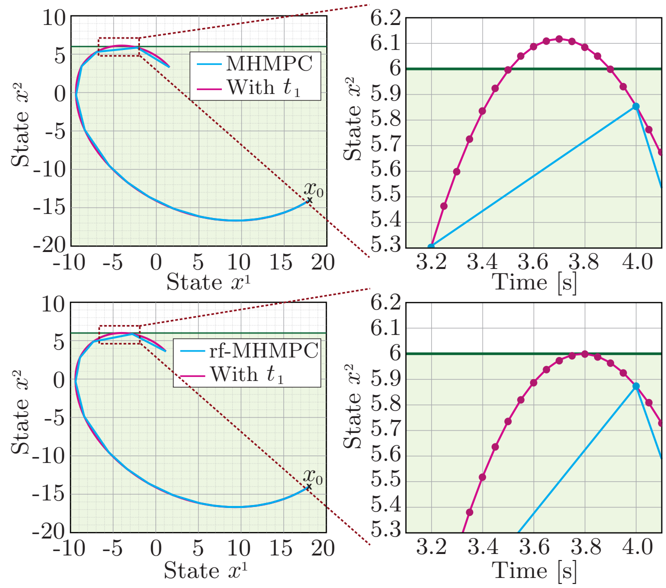

Fig. 5. The optimal trajectory computed by both the MHMPC and the rf-MHMPC optimization at the initial time step in the first iteration of the MPC and the result of implementing the computed optimal inputs with a sampling time t_1. The figure on the right zooms in on state x^2 when it approaches the limit of the state constraint set. It shows that presence of the additional constraints in rf-MHMPC ensure that the trajectory remains feasible even between two points whereas the MHMPC solution violates the constraints when implemented with a sampling time of t_1.

Method	T	N	Cost	Avg. time
MPC t = 0.05	1	21	5.47e4	0.015
	4.8	97	5.23e4	0.093
MPC t = 0.1	2	21	5.48e4	0.02
	4.8	49	5.24e4	0.037
MPC t = 0.2	4.8	25	5.27e4	0.024
rf-MHMPC	4.8	21	5.14e4	0.029

TABLE III

COMPARISON OF COST AND COMPUTATION TIME OF DIFFERENT STANDARD MPC CONTROLLERS WITH RF-MHMPC FOR A LARGER SYSTEM WITH 10 MASS SPRING DAMPERS IN SERIES.

VI. CONCLUSION

A constraint tightening scheme to enforce recursive feasibility in MHMPC is proposed. The method constrains the states and inputs in each sub-interval of the MHMPC problem based on the corresponding sampling time and the constraint sets gets progressively more restrictive for each sub-interval as the sampling time increases. The additional constraints ensure that the system remains feasible within the prediction interval even as the horizon moves forward with the smallest sampling time and always satisfies the state constraints between any two sampling points with a larger sampling time. Hence, along with the terminal constraint, the proposed strategy guarantees recursive feasibility. Numerical examples demonstrate the effectiveness and scalability of this approach and show that the recursively feasible controller has a superior performance in terms of both cost and computation time compared to standard MPC controllers. Future work aims to extend this method to non-linear systems.

REFERENCES

[1] R. Gondhalekar, J. Imura, K. Kashima, “Controlled invariant feasibility - A general approach to enforcing strong feasibility in MPC applied to move-blocking,” in Automatica, vol. 45, no. 12, 2009, pp. 2869-2875.

[2] M. Arnold, R. R. Negenborn, G. Andersson, B. De Schutter, “Distributed Predictive Control for Energy Hub Coordination in Coupled Electricity and Gas Networks,” in Springer, 2010, pp. 235–273.

[3] M. Mohammadi, Y. Noorollahi, B. Mohammadi-ivatloo, M. Hosseinzadeh, H. Yousefi, S. T. Khorasani, “Optimal management of energy hubs and smart energy hubs – a review,” in Renewable and Sustainable Energy Reviews, vol. 89, 2018, pp. 33–50.

[4] T. Bäthge, S. Lucia, R. Findeisen, “Exploiting models of different granularity in robust predictive control,” in IEEE 55th Conference on Decision and Control (CDC), 2016

[5] S. Shin, V. M. Zavala, “Diffusing-horizon model predictive control,” in IEEE Transactions on Automatic Control, vol. 68, 2023, pp. 188–201.

[6] S. Na, M. Anitescu, “Exponential decay in the sensitivity analysis of nonlinear dynamic programming,” in SIAM Journal on Optimization, vol. 30, 2020, pp. 1527–1554.

[7] W. Xu, M. Anitescu, “Exponentially accurate temporal decomposition for long- horizon linear-quadratic dynamic optimization,” in SIAM Journal on Optimization, vol. 28, 2018, pp. 2541–2573.

[8] V. Behrunani, H. Cai, P. Heer, R. S. Smith, J. Lygeros, “Distributed Multi-Horizon Model Predictive Control for Network of Energy Hubs,” Control Engineering Practice, vol. 147, 2024.

[9] C. L. Kuan, K. Honda, H. Okuda and T. Suzuki, "Multi-Horizon and Multi-Rate Model Predictive Control for Integrated Longitudinal and Lateral Vehicle Control," in IEEE Intelligent Vehicles Symposium (IV), 2023, pp. 1-8.

[10] R. Gondhalekar and Jun-ichi Imura, "Recursive feasibility guarantees in move-blocking MPC," IEEE 46th Conference on Decision and Control(CDC), 2007, pp. 1374-1379.

[11] R. Cagienard, P. Grieder, E. Kerrigan, M. Morari, "Move Blocking Strategies in Receding Horizon Control," Journal of Process Control, vol. 17, no. 6, 2007, pp.563-570.

[12] E. C. Kerrigan and J. M. Maciejowski, "Invariant sets for constrained nonlinear discrete-time systems with application to feasibility in model predictive control," in Proceedings of the IEEE 39th Conference on Decision and Control (CDC), vol.5, 2000, pp. 4951-4956.

[13] D.Q. Mayne, J.B. Rawlings, C.V. Rao, P.O.M. Scokaert," Constrained model predictive control: Stability and optimality," in Automatica, vol. 36, no. 6, 2000, pp. 789-814.

[14] F. Blanchini, "Set invariance in control," in Automatica, vol. 35, no. 11, 1999, pp. 1747-1767.

[15] M. N. Zeilinger, M. Morari and C. N. Jones, "Soft Constrained Model Predictive Control With Robust Stability Guarantees," in IEEE Transactions on Automatic Control, vol. 59, no. 5, 2014, pp. 1190-1202.

[16] D. A. Allan, C. N. Bates, M. J. Risbeck, J. B. Rawlings," On the inherent robustness of optimal and suboptimal nonlinear MPC," in Systems and Control Letters, vol. 106, 2017, pp. 68-78.

[17] G. Pannocchia, J. B. Rawlings, S. J. Wright," Conditions under which suboptimal nonlinear MPC is inherently robust," in Systems and Control Letters, vol. 60, no. 9, 2011, pp. 747-755.

[18] J. Löfberg, "Oops! I cannot do it again: Testing for recursive feasibility in MPC," in Automatica, vol. 48, no. 3, 2012, pp. 550-555.

[19] A. Parsi, P. Anagnostaras, A. Iannelli and R. S. Smith, "Computationally efficient robust MPC using optimized constraint tightening," in IEEE 61st Conference on Decision and Control (CDC), 2022, pp. 1770-1775.

[20] Y. Kim, X. Zhang, J. Guanetti and F. Borrelli, "Robust Model Predictive Control with Adjustable Uncertainty Sets," in IEEE 57th Conference on Decision and Control (CDC), 2018, pp. 5176-5181.

[21] M. Lorenzen, F. Allgöwer, F. Dabbene and R. Tempo, "Scenario-based Stochastic MPC with guaranteed recursive feasibility," in IEEE 54th Conference on Decision and Control (CDC), 2015, pp. 4958-4963.

[22] W. Esterhuizen, K. Worthmann and S. Streif, "Recursive Feasibility of Continuous-Time Model Predictive Control Without Stabilising Constraints," in IEEE Control Systems Letters, vol. 5, no. 1, 2021, pp. 265-270.

[23] A.L. Tits, PA. Absil and W. P. Woessner. "Constraint reduction for linear programs with many inequality constraints." in SIAM Journal on Optimization, vol.17, no. 1, 2006, pp.119-146.