Error-Free Approximation of Explicit Linear MPC Through Lattice Piecewise Affine Expression

Xu, Jun; Lou, Yunjiang; De Schutter, Bart; Xiong, Zhenhua

DOI
10.1109/TAC.2024.3466869

Publication date
2025

Document Version
Final published version

Published in
IEEE Transactions on Automatic Control

Error-Free Approximation of Explicit Linear MPC Through Lattice Piecewise Affine Expression

Jun Xu, *Senior Member, IEEE*, Yunjiang Lou, *Senior Member, IEEE*, Bart De Schutter, *Fellow, IEEE*, and Zhenhua Xiong, *Member, IEEE*

***Abstract*—In this article, the disjunctive and conjunctive lattice piecewise affine (PWA) approximations of explicit linear model predictive control (MPC) are proposed. Training data consisting of states and corresponding affine control laws are generated in a control invariant set, and redundant sample points are removed to simplify the construction of lattice PWA approximations. Resampling is proposed to guarantee the equivalence of lattice PWA approximations and optimal MPC control law at the sample points. Under certain conditions, the disjunctive lattice PWA approximation constitutes a lower bound, whereas the conjunctive version formulates an upper bound of the original optimal control law. The equivalence of the two lattice PWA approximations then guarantees error-free approximations in the domain of interest, which is tested through a statistical guarantee. The performance of the proposed approximation strategy is tested through two simulation examples, and the results show that error-free lattice PWA approximations can be obtained with low offline complexity and small storage requirements. Besides, the online complexity is less compared with the state-of-the-art method.**

***Index Terms*—Error-free approximation, lattice piecewise affine (PWA), linear model predictive control (MPC).**

I. Introduction

THE impact of model predictive control (MPC) on the industry has been recognized widely [1], and the complexity of online optimization restricts the prevalent application of MPC [2]. To alleviate this situation, explicit MPC was proposed [3], in which the linear MPC problem is formulated as a multiparametric quadratic programming (mpQP) problem and solved offline. The optimal control law is proved to be continuous piecewise affine (PWA) with respect to the state. The subregions, as well as the corresponding affine functions defined on them, are recorded. For online implementation, given the current state, one must only find the subregion in which the state lies, and the function evaluation of the corresponding affine function gives rise to the optimal control law.

However, the offline construction of such subregions, the memory required to store them, and the online search for the right subregion are the main limitations of explicit MPC [4]. Much work has been done to solve these three problems. To overcome the complex offline geometric computations, combinatorial approaches are proposed based on implicitly enumerating all possible combinations of active constraints [5], [6]. To reduce the memory required to store the information of subregions and control laws, region-free explicit MPC is proposed [4], [7]. Moreover, the online search complexity can be reduced by storing additional information [8], [9], introducing an improved binary search tree (orthogonal truncated binary search tree) [10], or resorting to the method of convex lifting [11], [12]. The lattice PWA representation has also been used to exactly express the explicit MPC control law, resulting in a much lower storage requirement [13], [14]. As the complexity of solving the explicit MPC problem increases exponentially with size of the optimization problem, all the above methods can only alleviate the computational burden to some extent.

Another idea is to formulate the approximate MPC controller [15], [16] or semiexplicit MPC controller [17]. In these methods, training data containing the values of states and corresponding optimal control laws of the MPC problem are generated, and the approximated controller is constructed using these data. In general, the samples are required to be distributed sufficiently evenly over the domain [18]. Different approaches have been used to generate the approximation, such as the canonical piecewise linear function [19], radial basis functions [20], wavelets [21], and so on. In addition, reinforcement learning has also been used to derive a data-driven MPC control law in [22]. In the work of [16], [21], and [23], the approximations are based on particular partitions of the domain of interest, and the interpolation-based algorithm can be developed [24]. Approximations by neural networks or basis functions generally require computationally expensive optimization-based training, and the interpolation-based algorithm always introduces partitions of

Received 8 February 2024; revised 13 July 2024; accepted 15 September 2024. Date of publication 24 September 2024; date of current version 28 February 2025. This work was supported in part by the National Natural Science Foundation of China under Grant 62173113, in part by the Science and Technology Innovation Committee of Shenzhen Municipality under Grant GXWD20231129101652001, and in part by Natural Science Foundation of Guangdong Province of China under Grant 2022A1515011584. An earlier version of this paper was presented in part at the 60th IEEE Conference on Decision and Control (CDC), [DOI: 10.1109/CDC45484.2021.9683051]. Recommended by Associate Editor E. C. Kerrigan. *(Corresponding authors: Jun Xu; Yunjiang Lou.)*

Jun Xu and Yunjiang Lou are with the Department of Automation, Harbin Institute of Technology, Shenzhen 518055, China (e-mail: xujunqgy@gmail.com; louyj@hit.edu.cn).

Bart De Schutter is with the Delft Center for Systems and Control, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: b.deschutter@tudelft.nl).

Zhenhua Xiong is with the State Key Laboratory of Mechanical Systems and Vibration, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: mexiong@sjtu.edu.cn).

Digital Object Identifier 10.1109/TAC.2024.3466869

the domain that are different from the domain partitions in the explicit linear MPC law.

In our previous work [14], the lattice PWA *representation* of the explicit MPC control law was derived, which, however, also depends on the result of explicit MPC, and scales poorly with the size of MPC control problem. To resemble explicit MPC control law to a larger extent and also to handle the issue of high offline complexity in explicit MPC, this article proposes an approximation approach *in the domain of interest*, in which the explicit MPC control law needs not be obtained in advance. Specifically, data containing sampled states and corresponding local affine functions are employed to derive the lattice PWA approximation, which can be guaranteed to be statistically *error free*, i.e., coincide with the explicit MPC control under certain assumptions. Moreover, the worst-case offline calculation complexity depends mainly on the number of sample points, which scales well with the state dimension. A preliminary thought of the disjunctive lattice PWA approximation of explicit linear MPC was presented in [25], in which the approximated control law is not guaranteed to be error free. However, in this work, under certain assumptions, the disjunctive and conjunction lattice PWA approximations constitute the lower and upper bounds of the optimal MPC control law, and the equivalence of the two approximations ensures that the two approximations are identical to the optimal control law in the domain of interest. The approximation can also be simplified to lower storage requirements and online computational complexity.

The rest of this article is organized as follows. Section II gives the preliminaries about the explicit linear MPC problem and the lattice PWA representation. The offline approximations of the explicit linear MPC control law through the lattice PWA expression are given in detail in Section III, in which the construction of the sample domain, sampling and resampling procedures, simplifications of the approximations, online evaluation of the approximations, and the complexity analysis are provided. In Section IV, the approximation error is analyzed. Section V provides simulation results. Finally, Section VI concludes this article.

II. Preliminaries

A. Explicit Linear MPC Problem

In particular, MPC for a discrete-time linear time-invariant system can be cast as the following optimization problem at time step t:

$$\min_{U} J(U, \boldsymbol{x}(t)) = V(\boldsymbol{x}_{t+N_p|t}) + \sum_{k=0}^{N_p-1} v(\boldsymbol{x}_{t+k|t}, \boldsymbol{u}_{t+k|t}) \quad (1a)$$

$$\text{s.t. } \boldsymbol{x}_{t+k+1|t} = A\boldsymbol{x}_{t+k|t} + B\boldsymbol{u}_{t+k|t}, \quad k = 0, \ldots, N_p - 1 \quad (1b)$$

$$\boldsymbol{x}_{t+k|t} \in \mathbb{X}, \quad k = 0, \ldots, N_p \quad (1c)$$

$$\boldsymbol{u}_{t+k|t} \in \mathbb{U}, \quad k = 0, \ldots, N_p - 1 \quad (1d)$$

$$\boldsymbol{x}_{t+N_p|t} \in \mathbb{X}_f \quad (1e)$$

in which the optimized variable is $U = [\boldsymbol{u}_{t|t}^T, \ldots, \boldsymbol{u}_{t+N_p-1|t}^T]^T$, and N_p is the prediction horizon. The variable $\boldsymbol{x}_{t|t} = \boldsymbol{x}(t) \in \mathbb{R}^{n_x}$, and $\boldsymbol{x}_{t+k|t} \in \mathbb{R}^{n_x}$ and $\boldsymbol{u}_{t+k|t} \in \mathbb{R}^{n_u}$ denote the predicted state and input at time step $t+k$, respectively, using (1b). The terminal penalty is denoted as V, $v(\cdot,\cdot)$ is the stage cost, and $\mathbb{X}, \mathbb{X}_f$, and $\mathbb{U}$ are full-dimensional polyhedral sets of appropriate dimensions. In this article, we assume a strictly convex cost, i.e., $V(\boldsymbol{x}_{t+N_p|t}) = \boldsymbol{x}_{t+N_p|t}^T Q_{N_p} \boldsymbol{x}_{t+N_p|t}$, $v(\boldsymbol{x}_{t+k|t}, \boldsymbol{u}_{t+k|t}) = \boldsymbol{x}_{t+k|t}^T Q_k \boldsymbol{x}_{t+k|t} + \boldsymbol{u}_{t+k|t}^T Q_u \boldsymbol{u}_{t+k|t}$, in which $Q_u \succ 0, Q_k, Q_{N_p} \succeq 0$. After solving the optimization problem (1), the optimal $U^* = [(\boldsymbol{u}_{t|t}^*)^T, \ldots, (\boldsymbol{u}_{t+N_p-1|t}^*)^T]^T$ is obtained, and only $\boldsymbol{u}_{t|t}^*$ is applied to the system. The optimization problem is subsequently reformulated and solved at the next time steps by updating the given state vector $\boldsymbol{x}(t)$.

It has been proved in [3] that the solution U^* is a *continuous PWA function* of the state $\boldsymbol{x}(t)$, and we use $\boldsymbol{x}$ instead hereafter in this article. The definition of a continuous PWA function is presented as follows.

Definition 1 (see [26]): A function $f : \Omega \to \mathbb{R}^m$, where $\Omega \subseteq \mathbb{R}^{n_x}$ is convex, is said to be continuous PWA if it is continuous on the domain Ω and the following conditions are satisfied.

1) The domain space Ω is divided into a finite number of nonempty convex polyhedra, i.e., $\Omega = \cup_{i=1}^{\hat{N}} \Omega_i$, $\Omega_i \neq \emptyset$, the polyhedra are closed and have nonoverlapping interiors, $\text{int}(\Omega_i) \cap \text{int}(\Omega_j) = \emptyset \ \forall i, j \in \{1, \ldots, \hat{N}\}, i \neq j$, in which $\text{int}(\cdot)$ denotes the interior of a set. These polyhedra are also called local regions. The boundaries of the polyhedra are nonempty sets in $(n-1)$-dimensional space.
2) In each local region Ω_i, f equals a local affine function $\ell_{\text{loc}(i)}$

$$f(\boldsymbol{x}) = \ell_{\text{loc}(i)}(\boldsymbol{x}) \quad \forall x \in \Omega_i$$

in which the subscript $\text{loc}(i)$ denotes the index of local affine function in Ω_i.

The local affine function, as well as local regions in the continuous PWA function $U^*(\boldsymbol{x})$, is obtained through the Karush–Kuhn–Tucker (KKT) conditions of the following mpQP problem:

$$\min_{\boldsymbol{z}} \frac{1}{2} \boldsymbol{z}^T H \boldsymbol{z} \quad (2a)$$

$$\text{s.t.} G\boldsymbol{z} \leq \boldsymbol{w} + S\boldsymbol{x} \quad (2b)$$

in which $\boldsymbol{z} = U + H^{-1} F^T \boldsymbol{x}$, and the matrices $H, F, G,$ and S as well as the vector w are obtained through (1).

For a given state $\boldsymbol{x}$, solving (2) yields the optimal solution $\boldsymbol{z}^*$, which together with $\boldsymbol{x}$ can determine the active and inactive constraints in (2b). Assuming that $G \in \mathbb{R}^{p \times N_p \cdot n_u}$, $\boldsymbol{w} \in \mathbb{R}^p$, $S \in \mathbb{R}^{p \times n_x}$, and G_i, w_i, and S_i denote the ith row of G, w, and S, respectively. If the ith constraint is active, we have $G_i z = w_i + S_i \boldsymbol{x}$; if it is inactive, we have $G_i z < w_i + S_i \boldsymbol{x}$. Then, the active as well as inactive index sets can be written as

$$\mathcal{A}^* = \{j \in \{1, \ldots, p\} | G_j \boldsymbol{z}^* = \boldsymbol{w}_j + S_j \boldsymbol{x}\}$$

and

$$\mathcal{N}^* = \{j \in \{1, \ldots, p\} | G_j \boldsymbol{z}^* < \boldsymbol{w}_j + S_j \boldsymbol{x}\}$$

respectively. It is apparent that $\mathcal{A}^* = \{1, \ldots, p\} \setminus \mathcal{N}^*$.

For a particular $\mathcal{A}^*$, assume that $G_{\mathcal{A}^*}$ is full-row rank, according to the KKT conditions evaluated at $\boldsymbol{x}$ and $\boldsymbol{z}^*$,

we have

$$\boldsymbol{z}^* = H^{-1}G_{\mathcal{A}^*}^T(G_{\mathcal{A}^*}H^{-1}G_{\mathcal{A}^*}^T)^{-1}(\boldsymbol{w}_{\mathcal{A}^*} + S_{\mathcal{A}^*}\boldsymbol{x}). \quad (3)$$

The local region for which the local affine function (3) is defined is a polyhedron in the feasible state region, which is called *critical region* and can also be constructed through the KKT conditions, i.e.,

$$\mathcal{CR}(\boldsymbol{x}) = \left\{ \boldsymbol{x} \left| \begin{bmatrix} Gz^* \le W + S\boldsymbol{x} \\ (G_{\mathcal{A}^*}H^{-1}G_{\mathcal{A}^*}^T)^{-1}(\boldsymbol{w}_{\mathcal{A}^*} + S_{\mathcal{A}^*}\boldsymbol{x}) \le 0 \end{bmatrix} \right. \right\}. \quad (4)$$

Once the optimal solution $\boldsymbol{z}^*$ of the optimization problem (2) is available, we can easily obtain the optimal U^* as

$$U^* = \boldsymbol{z}^* - H^{-1}F^T\boldsymbol{x} \quad (5)$$

which is also affine within the critical region $\mathcal{CR}(\boldsymbol{x})$.

Remark 1: For the case in which $G_{\mathcal{A}^*}$ is not full-row rank, and assuming that the rank of $G_{\mathcal{A}^*}$ is r, we can then arbitrarily select r independent constraints and proceed with the new reduced active index set [27].

In order to obtain the continuous PWA control law, one must obtain all the local affine functions and critical regions, i.e., one must enumerate all possible active index sets $\mathcal{A}^*$, apply the KKT conditions accordingly [5], [6], [28], and the multiparametric toolbox 3.0 (MPT3) [29] can be used to fulfill this.

B. Lattice PWA Representation

For the continuous PWA control law obtained through MPT3, the lattice PWA representation is presented in our previous work [14] to express this continuous PWA function.

Lemma 1 (see [14]): Letting f be a continuous PWA function defined in Definition 1, then f can be represented as

$$f(\boldsymbol{x}) = f_{\mathrm{L,d}}(\boldsymbol{x}) = \max_{i=1,\ldots,N}\left\{\min_{j\in I_{\ge,i}}\{\ell_j(\boldsymbol{x})\}\right\} \quad \forall \boldsymbol{x}\in\Gamma \quad (6)$$

or

$$f(\boldsymbol{x}) = f_{\mathrm{L,c}}(\boldsymbol{x}) = \min_{i=1,\ldots,N}\left\{\max_{j\in I_{\le,i}}\{\ell_j(\boldsymbol{x})\}\right\} \quad \forall \boldsymbol{x}\in\Gamma \quad (7)$$

in which $I_{\ge,i} = \{j|\ell_j(\boldsymbol{x}) \ge \ell_{\mathrm{loc}}(i)(\boldsymbol{x}), \forall \boldsymbol{x}\in\Gamma_i\}$, $I_{\le,i} = \{j|\ell_j(\boldsymbol{x}) \le \ell_{\mathrm{loc}}(i)(\boldsymbol{x}), \forall \boldsymbol{x}\in\Gamma_i\}$, and the expressions $\min_{j\in I_{\ge,i}}\{\ell_j(\boldsymbol{x})\}$ and $\max_{j\in I_{\le,i}}\{\ell_j(\boldsymbol{x})\}$ are called terms of $f_{\mathrm{L,d}}$ and $f_{\mathrm{L,c}}$, respectively. The affine function $\ell_{\mathrm{loc}}(i)(\boldsymbol{x})$ is called a literal, representing the local affine function in Γ_i. The region Γ_i is a *base region* with $\bigcup_{i=1}^N \Gamma_i = \Gamma$ and $\Gamma_i \cap \Gamma_j = \emptyset \forall i \ne j$. The base region is a subset of the local region, and no other affine functions intersect with $\ell_i(\boldsymbol{x})$ at some point in the interior of Γ_i, i.e.,

$$\{\boldsymbol{x}|\ell_j(\boldsymbol{x}) = \ell_{\mathrm{loc}(i)}(\boldsymbol{x}), j\ne i\} \cap \mathrm{int}(\Gamma_i) = \emptyset. \quad (8)$$

Here, to avoid excessive use of notation, we just use $\mathrm{loc}(i)$ to denote the local affine functions of some region, whether it is a local region in Definition 1 or base region, which is part of the local region and additionally satisfying (8).

The expressions (6) and (7) are called full disjunctive and conjunctive lattice PWA representations, respectively, in which the names "disjunctive" and "conjunctive" come from the terminology in Boolean algebra.

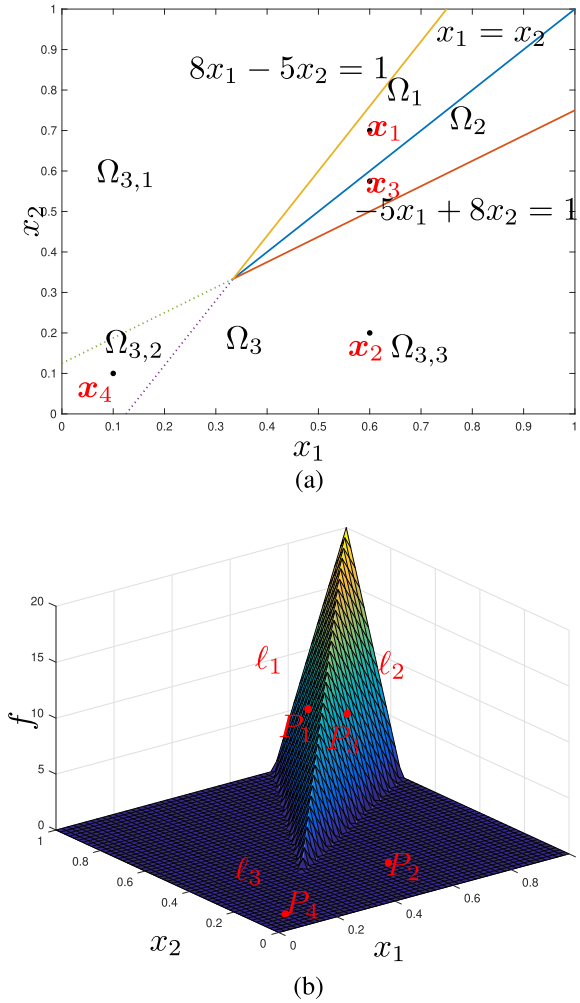


Fig. 1. Continuous PWA function in Example 1. (a) Subregion. (b) Function.

Considering a 2-D continuous PWA function (9) with three affine pieces, Fig. 1 illustrates the base region and corresponding PWA function.

Example 1: The expression for the 2-D continuous PWA function with three local affine functions is as follows:

$$f = \begin{cases} \ell_1(\boldsymbol{x}) = 80x_1 - 50x_2 - 10, & \text{if } \boldsymbol{x}\in\Omega_1 \\ \ell_2(\boldsymbol{x}) = -50x_1 + 80x_2 - 10, & \text{if } \boldsymbol{x}\in\Omega_2 \\ \ell_3(\boldsymbol{x}) = 0, & \text{if } \boldsymbol{x}\in\Omega_3 \end{cases} \quad (9)$$

the polyhedral regions and the 2-D function f are shown in Fig. 1. This example has also been studied in [30]. For this simple example, we have three polyhedral regions, i.e., Ω_1, Ω_2, and Ω_3, with local affine functions $\ell_{\mathrm{loc}(1)} = \ell_1, \ell_{\mathrm{loc}(2)} = \ell_2$, and $\ell_{\mathrm{loc}(3)} = \ell_3$. The polyhedral regions Ω_1 and Ω_2 are base regions and Ω_3 is not, i.e., in $\mathrm{int}(\Omega_1)$ and $\mathrm{int}(\Omega_2)$, no other affine functions intersect with ℓ_1 and ℓ_2, respectively, but the intersection of ℓ_1, ℓ_3 and ℓ_2, ℓ_3 are in the interior of Ω_3, shown as the green and purple dashed lines, respectively. Then, Ω_3 can be partitioned into three base regions $\Omega_{3,1}, \Omega_{3,2}$, and $\Omega_{3,3}$ by these two dashed lines, and we have $\ell_{\mathrm{loc}(3,1)} = \ell_{\mathrm{loc}(3,2)} = \ell_{\mathrm{loc}(3,3)} = \ell_3$.

According to the full disjunctive lattice PWA representation, we have five index sets $I_{\ge,i}$ corresponding to the five base

regions $\Omega_1, \Omega_2, \Omega_{3,1}, \Omega_{3,2}$, and $\Omega_{3,3}$, which can be expressed as follows:

$$I_{\geq,1} = \{1,2\}, I_{\geq,2} = \{1,2\}, I_{\geq,3} = \{2,3\}$$
$$I_{\geq,4} = \{3\}, I_{\geq,5} = \{1,3\}.$$

And we have the following full disjunctive lattice PWA representation:

$$f(\boldsymbol{x}) = \max\{\min\{\ell_1,\ell_2\}, \min\{\ell_1,\ell_2\}, \min\{\ell_3,\ell_2\}, \ell_3, \min\{\ell_3,\ell_1\}\}.$$

After removing redundant terms, we have

$$f_{\mathrm{L,d}} = \max\{\min\{\ell_1,\ell_2\},\ell_3\}. \tag{10}$$

Similarly, we can obtain the full conjunctive lattice PWA representation as

$$f_{\mathrm{L,c}} = \min\{\max\{\ell_1,\ell_3\}, \max\{\ell_2,\ell_3\}, \max\{\ell_1,\ell_3\} \max\{\ell_1,\ell_2,\ell_3\}, \max\{\ell_2,\ell_3\}\}$$

which can be then simplified to

$$f_{\mathrm{L,c}} = \min\{\max\{\ell_1,\ell_3\}, \max\{\ell_2,\ell_3\}\}. \tag{11}$$

According to Lemma 1, we can represent a continuous PWA control law using a lattice PWA function (either disjunctive or conjunctive). However, as explained in Section II-A, for problems with a large number of constraints and a high-dimensional state, the number of possible combinations of active constraints increases exponentially, and the derivation of the explicit MPC solution is extremely computationally expensive. Besides, partitioning the critical region into base regions is not possible for even medium-size problems, i.e., in [14], for a 4-D system, partitioning 890 critical regions into base regions is prohibitive.

Hence, in this article, we propose an approximated continuous PWA control law by sampling only a series of states in the domain of interest, which avoids the usage of MPT3 and the partitioning of the domain into base regions.

III. Lattice PWA Approximation of Explicit Linear MPC Control Law

A. Generation of Sample Points

As mentioned before, the explicit linear MPC control law U^* is a continuous PWA function with respect to the state $\boldsymbol{x}$. Then, the first element of U^*, which is $\boldsymbol{u}_0^*$, is also a continuous PWA function of $\boldsymbol{x}$, i.e., $\boldsymbol{u}_0^*$ is affine in the local regions that $\boldsymbol{x}$ lies in. Denote the domain of $\boldsymbol{x}$ defining $\boldsymbol{u}_0^*$ as Γ, in this article, we focus on the approximation of the PWA function $\boldsymbol{u}_0^*$ on Γ, and the affine functions and base regions are recorded accordingly. For simplicity, we omit the subscript in $\boldsymbol{u}_0^*$ hereafter in the article and use $\boldsymbol{u}^*$ instead, besides, we set $n_u = 1$; however, the proposed methodology can be easily extended to the case when $n_u > 1$.

1) Domain of Interest: The sample points are generated in the domain of interest, which is a polyhedron control invariant set Ω of the linear system (1b). Specifically, we set N_r initial sample points, and for each initial point, we solve the MPC problem (2)

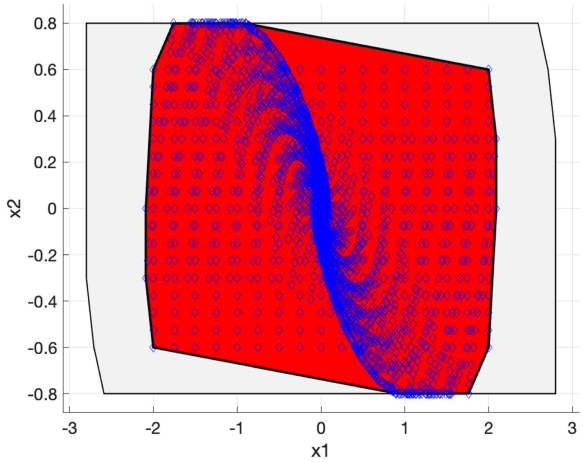


Fig. 2. Sample points, the domain of interest, and the feasible region for a 2-D example.

at subsequent time instants to generate N_r convergent closed-loop trajectories. As shown in Section II-A, the optimal control law is an affine function of the state within the corresponding critical region; the state points and corresponding affine functions on the trajectories are recorded. Assume that all the points on the N_r trajectories constitute a set $\mathcal{X}_1 = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_{s1}}\}$, in general, $N_r \ll N_{s1}$. For all $\boldsymbol{x}_i \in \mathcal{X}_1$, the affine function $\ell_{\mathrm{act}(i)}(\boldsymbol{x})$ is recorded, i.e.,

$$\ell_{\mathrm{act}(i)}(\boldsymbol{x}_i) = u_i = u^*(\boldsymbol{x}_i) \tag{12}$$

in which $\mathrm{act}(i)$ denotes the index for the affine function corresponding to $\boldsymbol{x}_i$, and the set $\{\ell_{\mathrm{act}(i)} | i \in \{1, \ldots, N_{s1}\}\}$ constitutes the set $\mathcal{U}_1$. The domain of interest is then set as the convex hull of all the sample points $\boldsymbol{x}_i \in \mathcal{X}_1$, i.e.,

$$\Omega = \mathrm{conv}(\mathcal{X}_1).$$

We demonstrate that the set Ω is *control invariant*. Actually, for any $\boldsymbol{x} \in \mathrm{conv}(\mathcal{X}_1)$, according to the definition of the convex hull, there is a nonzero $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_{N_{s1}}]^T$ with $\sum_{i=1}^{N_{s1}} \lambda_i = 1$, such that

$$\boldsymbol{x} = \sum_{\boldsymbol{x}_i \in \mathcal{X}_1} \lambda_i \boldsymbol{x}_i.$$

For each $\boldsymbol{x}_i \in \mathcal{X}_1$, as it is a point on a specific trajectory, the succeeding state $A\boldsymbol{x}_i + Bu_i$ is also on the same trajectory, in which u_i is calculated through solving (2), meaning that

$$A\boldsymbol{x}_i + Bu_i \in \mathrm{conv}(\mathcal{X}_1)$$

then by taking

$$u = \sum_{\boldsymbol{x}_i \in \mathcal{X}_1} \lambda_i u_i$$

we have

$$A\boldsymbol{x} + Bu = \sum_{\boldsymbol{x}_i \in \mathcal{X}_1} \lambda_i (A\boldsymbol{x}_i + Bu_i) \in \mathrm{conv}(\mathcal{X}_1)$$

i.e., the set $\Omega = \mathrm{conv}(\mathcal{X}_1)$ is *control invariant*.

Example 2: Fig. 2 shows the domain of interest for a 2-D system, which is introduced in [31]. The system dynamics can be written as

$$\boldsymbol{x}_{k+1} = \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix} \boldsymbol{x}_k + \begin{bmatrix} T_s^2 \\ T_s \end{bmatrix} u_k$$

where the sampling interval T_s is 0.3. Considering the MPC problem with $Q = \text{diag}(1, 0)$, $R = 1$, and P is the solution of the discrete-time algebraic Riccati equation, the prediction horizon is set to be 10. The system constraints are $-1 \leq u_k \leq 1, -2.8 \leq x_{k,1} \leq 2.8$, and $-0.8 \leq x_{k,2} \leq 0.8$. In total, 289 trajectories are generated in the region $[-2, 2] \times [-0.6, 0.6]$, resulting in 9112 sample points, shown in blue diamond, and the domain of interest is the convex hull of all the sample points, shown in red. The feasible region calculated through MPT3 is shown in gray.

Remark 2: The construction of Ω is computationally challenging for high-dimensional problems. In this article, we restrict the initial points of the sampled trajectories within a user-defined area Ω_0, e.g., a hyperrectangle or some other regular convex set. The domain of interest Ω is then set as the convex combination of all convergent trajectories starting from points in Ω_0.

2) Obtaining the Affine Control Laws: According to Section II-A, for a feasible state $\boldsymbol{x}_i$, the affine function $\ell_{\text{act}(i)}(\boldsymbol{x}_i)$ can be calculated through solving (2), (3), and (5), i.e.,

$$U^*(\boldsymbol{x}_i) = \boldsymbol{z}^*(\boldsymbol{x}_i) - H^{-1}F^T\boldsymbol{x}_i \tag{13}$$

and

$$\ell_{\text{act}(i)}(\boldsymbol{x}_i) = \begin{bmatrix} \mathbf{I}_{n_u \times n_u} & \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix} U^*(\boldsymbol{x}_i) \tag{14}$$

in which $\mathbf{I}_{n_u \times n_u}$ is the identity matrix with size $n_u \times n_u$. It is noted that (14) holds at the point $\boldsymbol{x}_i$, as U^* and z^* are continuous PWA functions and reduce to particular affine functions at the sample point $\boldsymbol{x}_i$.

Compared with ordinary sampling, in which only the value of u_i is available, here the corresponding affine function $\ell_{\text{act}(i)}(\boldsymbol{x}_i)$ is also recorded. The resulting sample dataset is recorded as $\mathcal{X}_1 \times \mathcal{U}_1$ with data length N_{s1}, in which $\mathcal{U}_1$ is a set of affine functions $\ell_{\text{act}(i)}(\boldsymbol{x})$, and in Section III-B, we will explain in detail how to build a lattice PWA approximation.

B. Lattice PWA Approximation Based on Sample Points

We now derive the disjunctive and conjunctive lattice PWA approximations based on the sample dataset $\mathcal{X}_1 \times \mathcal{U}_1$.

The disjunctive lattice PWA approximation is constructed via the sample points and can be expressed as follows:

$$\hat{f}_{\text{L,d}}(\boldsymbol{x}) = \max_{i=1,\ldots,N_{s1}} \left\{ \min_{j \in J_{\geq,i}} \{\ell_j(\boldsymbol{x})\} \right\} \tag{15}$$

in which the index set $J_{\geq,i}$ is described as follows:

$$J_{\geq,i} = \{j | \ell_j(\boldsymbol{x}_i) \geq \ell_{\text{act}(i)}(\boldsymbol{x}_i)\} \tag{16}$$

and $\ell_{\text{act}(i)}$ is defined in (12), representing the affine function at $\boldsymbol{x}_i$.

Similarly, the conjunctive lattice PWA approximation can be described as follows:

$$\hat{f}_{\text{L,c}}(\boldsymbol{x}) = \min_{i=1,\ldots,N_{s1}} \left\{ \max_{j \in J_{\leq,i}} \{\ell_j(\boldsymbol{x})\} \right\} \tag{17}$$

in which the index set $J_{\leq,i}$ is defined as

$$J_{\leq,i} = \{j | \ell_j(\boldsymbol{x}_i) \leq \ell_{\text{act}(i)}(\boldsymbol{x}_i)\}. \tag{18}$$

Compared with the full disjunctive and conjunctive lattice PWA representations (6) and (7), we can see that the lattice PWA approximations (15) and (17) only consider the order of local affine control laws at each sample point $\boldsymbol{x}_i \in \mathcal{X}_1$.

Following we will explain that under certain conditions as shown in Assumption 1, the lattice PWA approximations (15) and (17) coincide with the explicit linear MPC control law at the sample points.

Assumption 1: The terms in disjunctive lattice PWA approximations satisfy

$$\min_{j \in J_{\geq,i}} \{\ell_j(\boldsymbol{x}_k)\} \leq \ell_{\text{act}(k)}(\boldsymbol{x}_k) \quad \forall i, k \in \{1, \ldots, N_{s1}\} \tag{19}$$

and the terms in conjunctive lattice PWA approximations satisfy

$$\max_{j \in J_{\leq,i}} \{\ell_j(\boldsymbol{x}_k)\} \geq \ell_{\text{act}(k)}(\boldsymbol{x}_k) \quad \forall i, k \in \{1, \ldots, N_{s1}\}. \tag{20}$$

This assumption means that when evaluated at the sample points, the terms in the disjunctive lattice PWA approximations are not larger than, while the terms in the conjunctive lattice PWA approximations are not smaller than the active affine functions. Assumption 1 is not difficult to check, as it requires only the information of the sample points, and later in Section III-D, we will give solutions to guarantee the validity of Assumption 1. With this assumption, we can show in Theorem 1 the conclusion about the equivalence of the approximations and optimal MPC control law.

Theorem 1: Assuming that the disjunctive and conjunctive lattice PWA approximations are constructed through (15) and (17), respectively. Supposing that Assumption 1 holds, then we have

$$\hat{f}_{\text{L,d}}(\boldsymbol{x}_k) = \ell_{\text{act}}(\boldsymbol{x}_k) = \hat{f}_{\text{L,c}}(\boldsymbol{x}_k) \quad \forall \boldsymbol{x}_k \in \mathcal{X}_1. \tag{21}$$

Furthermore, if for all $i, k \in \{1, \ldots, N_{s1}\}$, we have

$$\min_{j \in J_{\geq,i}} \{\ell_j(\boldsymbol{x}_k)\} < \ell_{\text{act}(k)}(\boldsymbol{x}_k), \text{ or } \text{act}(k) \in J_{\geq,i} \tag{22}$$

and

$$\max_{j \in J_{\leq,i}} \{\ell_j(\boldsymbol{x}_k)\} > \ell_{\text{act}(k)}(\boldsymbol{x}_k), \text{ or } \text{act}(k) \in J_{\leq,i}$$

then for all points $\boldsymbol{x}_k \in \mathcal{X}_1$, the following equalities hold for the base regions including $\boldsymbol{x}_k$, i.e.,

$$\hat{f}_{\text{L,d}}(\boldsymbol{x}) = u^*(\boldsymbol{x}) = \hat{f}_{\text{L,c}}(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \Gamma_{\text{pnt}(k)} \tag{23}$$

in which $\Gamma_{\text{pnt}(k)} \subset \Gamma$ is the base region contains the sample point $\boldsymbol{x}_k$ such that

$$u^*(\boldsymbol{x}) = \ell_{\text{act}(k)}(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \Gamma_{\text{pnt}(k)}.$$

Proof: See the Appendix. ■

According to Theorem 1, the validity of Assumption 1 guarantees the equivalence of the lattice PWA approximation and the optimal control law at the sample points. Besides, if a stricter condition (22) is imposed, the lattice PWA approximations equal the original control law not only at the sample points but also in the base regions containing the sample points, as (23) shows.

In order to better describe the equivalence of $\hat{f}_{\mathrm{L,d}}(\boldsymbol{x})$ ($\hat{f}_{\mathrm{L,c}}(\boldsymbol{x})$) and $u^*(\boldsymbol{x})$ in base regions, we introduce the concept of covered base regions for terms generated through sample points $\boldsymbol{x}_i$, i.e., $\min_{j\in J_{\geq,i}}\{\ell_j(\boldsymbol{x})\}$ and $\max_{j\in J_{\leq,i}}\{\ell_j(\boldsymbol{x})\}$, and denote them as $\mathcal{C}(\min_{j\in J_{\geq,i}}\{\ell_j(\boldsymbol{x})\})$ or $\mathcal{C}(\max_{j\in J_{\leq,i}}\{\ell_j(x)\})$. This means that

$$u^*(\boldsymbol{x}) = \min_{j\in J_{\geq,i}}\{\ell_j(\boldsymbol{x})\} \quad \forall \boldsymbol{x}\in\mathcal{C}\left(\min_{j\in J_{\geq,i}}\{\ell_j(\boldsymbol{x})\}\right)$$

or

$$u^*(\boldsymbol{x}) = \max_{j\in J_{\leq,i}}\{\ell_j(x)\} \quad \forall \boldsymbol{x}\in\mathcal{C}\left(\max_{j\in J_{\leq,i}}\{\ell_j(x)\}\right).$$

It should be pointed out that the covered base regions $\mathcal{C}(\min_{j\in J_{\geq,i}}\{\ell_j(\boldsymbol{x})\})$ and $\mathcal{C}(\max_{j\in J_{\leq,i}}\{\ell_j(x)\})$ are with respect to the continuous PWA function $u^*(\boldsymbol{x})$, and are subsets of Γ. From the proof of Theorem 1, we have

$$\Gamma_{\mathrm{pnt}(k)}\subset\mathcal{C}\left(\min_{j\in J_{\geq,k}}\{\ell_j(\boldsymbol{x})\}\right),\Gamma_{\mathrm{pnt}(k)}\subset\mathcal{C}\left(\max_{j\in J_{\leq,k}}\{\ell_j(x)\}\right)$$

which means that the terms generated through sample points can cover the entire base region the sample points lie in. And we will show in Section III-D that the terms $\min_{j\in J_{\geq,k}}\{\ell_j(\boldsymbol{x})\}$ ($\max_{j\in J_{\leq,k}}\{\ell_j(\boldsymbol{x})\}$) can cover base regions other than $\Gamma_{\mathrm{pnt}(k)}$.

Remark 3: It should be noted that the computational complexity for constructing a lattice PWA *approximation* is much less than that for constructing a lattice PWA *representation*, in which all the base regions as well as all the distinct affine functions should be derived. In this article, however, the optimal continuous PWA controller information is not needed beforehand to construct a lattice PWA approximation, i.e., preprocessing by MPT3 is not needed. All we need to do is to sample points, record states and corresponding affine functions, and construct the lattice PWA approximations (15) and (17).

C. Simplification of Lattice PWA Approximation

When the number of sample points is large, say tens of thousands, the evaluation of (15) and (17) is not easy, and hence the simplification is considered in this section.

The simplification of a disjunctive lattice PWA function was addressed in [14], for which the detailed subregions of the PWA function are known. In this article, the information of the subregions of the PWA function is unknown. It is also challenging to derive the expression of the subregion polyhedra through lattice PWA approximation.

Hence, this section simplifies the disjunctive and conjunctive lattice PWA approximations according to the following rule. Assuming that the set C denotes the codomain of affine functions $u_1, u_2, \ldots$, the operations $\bigvee$ and $\bigwedge$ are defined as follows:

$$u_i\bigvee u_j=\max\{u_i,u_j\},u_i\bigwedge u_j=\min\{u_i,u_j\}.$$

It has been shown in [32] that the set C, together with the operations $\bigvee$ and $\bigwedge$, constitutes a distributive lattice, and the following property holds for all $u_i, u_j \in C$:

$$R1: \begin{matrix} u_i\bigvee(u_i\bigwedge u_j)=u_i \\ u_i\bigwedge(u_i\bigvee u_j)=u_i. \end{matrix} \tag{24}$$

Concerning the covered base regions in disjunctive lattice PWA approximation, we have the following result, as shown in Lemma 2. For the conjunctive lattice PWA approximation $\hat{f}_{\mathrm{L,c}}$, things are similar, and we omit them here for the sake of conciseness.

Lemma 2: Given two terms $\min_{j\in J_{\geq,i}}\{u_j(\boldsymbol{x})\}$ and $\min_{j\in J_{\geq,k}}\{u_j(\boldsymbol{x})\}$ generated from sample points $\boldsymbol{x}_i$ and $\boldsymbol{x}_k$, in which $J_{\geq,k}\subset J_{\geq,i}$, supposing that Assumption 1 and (22) hold, we have

$$\mathcal{C}\left(\min_{j\in J_{\geq,i}}\{u_j(\boldsymbol{x})\}\right)\subset\mathcal{C}\left(\min_{j\in J_{\geq,k}}\{u_j(\boldsymbol{x})\}\right).$$

Furthermore, if $\mathrm{act}(i)\in J_{\geq,t}, t\in\{1,\ldots,N_{s1}\}\setminus\{i\}$, then $\boldsymbol{x}_i$ can be removed from $\mathcal{X}_1$ without affecting the function value of $\hat{f}_{\mathrm{L,d}}$.

Proof: See the Appendix. ■

Lemma 2 means that as long as $J_{\geq,k}\subset J_{\geq,i}$ and $\mathrm{act}(i)$ appears in other terms, the sample point $\boldsymbol{x}_i$ is redundant in constructing lattice PWA approximations. Actually, multiple base regions may share the same term $J_{\geq,k}$, besides, the condition $J_{\geq,k}\subset J_{\geq,i}$ is even easier to be satisfied for different base regions. Hence, one term may cover many base regions, as Example 1 in Section III-D illustrates, and in general, the number of sample points needed to construct the lattice PWA approximation is much less than that of base regions and polyhedral regions, and simulation results in Section V confirm this.

D. Resampling

In the process of generating sample points, (19) and (20) may not be valid, and the following gives a resampling method such that both (19) and (20) hold.

Taking the disjunctive lattice PWA approximation as an example, if (19) is violated for some $\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta \in \mathcal{X}_1$, i.e.,

$$\min_{j\in J_{\geq,\alpha}}\{\ell_j(\boldsymbol{x}_\beta)\}>u^*(\boldsymbol{x}_\beta)=\ell_{\mathrm{act}(\beta)}(\boldsymbol{x}_\beta) \tag{25}$$

we can add sample points in the line segment

$$\mathcal{L}(\boldsymbol{x}_\alpha,\boldsymbol{x}_\beta)=\lambda\boldsymbol{x}_\alpha+(1-\lambda)\boldsymbol{x}_\beta,\lambda\in(0,1) \tag{26}$$

such that (19) is satisfied for $i=\alpha$ and $k=\beta$, which is shown in Lemma 3.

Lemma 3: Assuming that there are two points $\boldsymbol{x}_\alpha$ and $\boldsymbol{x}_\beta$ such that (25) holds, then there must be some points $\boldsymbol{x}_\gamma\in\mathcal{L}(\boldsymbol{x}_\alpha,\boldsymbol{x}_\beta)$, which is defined in (26), and the corresponding affine control law $\ell_{\mathrm{act}(\gamma)}$, such that the inequality

$$\ell_{\mathrm{act}(\gamma)}(\boldsymbol{x}_\alpha)\geq\ell_{\mathrm{act}(\alpha)}(\boldsymbol{x}_\alpha),\ell_{\mathrm{act}(\gamma)}(\boldsymbol{x}_\beta)\leq\ell_{\mathrm{act}(\beta)}(\boldsymbol{x}_\beta) \tag{27}$$

holds.

Furthermore, by adding $\boldsymbol{x}_\gamma$ to the sample dataset, we have $\mathrm{act}(\gamma)\in J_{\geq,\alpha}$ and

$$\min_{j\in J_{\geq,\alpha}}\{\ell_j(\boldsymbol{x}_\beta)\}\leq\ell_{\mathrm{act}(\beta)}(\boldsymbol{x}_\beta). \tag{28}$$

Algorithm 1: Recursive partitioning of line segment $\mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$ in case that (25) or (29) is violated.

Input: Linear MPC problem, initial sample dataset $\mathcal{X}_1 \times \mathcal{U}_1$, the line segment $\mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$.
Output: Additional sample dataset $\mathcal{X}_2 \times \mathcal{U}_2$.

1: Initialize $flag = 1$, $\mathcal{X}_2 = \emptyset, \mathcal{U}_2 = \emptyset$.
2: **while** flag **do**
3: $N_a = 0$;
4: **for** $\boldsymbol{x}_i \in \mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta) \cap (\mathcal{X}_1 \cup \mathcal{X}_2)$ **do**
5: Select corresponding $\ell_{\text{act}(i)}(\boldsymbol{x}_i) \in \mathcal{U}_1 \cup \mathcal{U}_2$.
6: **if** $\text{sign}(\ell_{\text{act}(i)}(\boldsymbol{x}_i) - \ell_{\text{act}(i+1)}(\boldsymbol{x}_i)) = \text{sign}(\ell_{\text{act}(i)}(\boldsymbol{x}_{i+1}) - \ell_{\text{act}(i+1)}(\boldsymbol{x}_{i+1}))$ **then**
7: $N_a = N_a + 1$.
8: Add a point $\boldsymbol{x}_{\text{new}} = 0.5(\boldsymbol{x}_i + \boldsymbol{x}_{i+1})$ to $\mathcal{X}_2$.
9: Calculate corresponding affine function $u^*(\boldsymbol{x}_{\text{new}})$ through (3), (13)-(14), add to $\mathcal{U}_2$.
10: **end if**
11: **end for**
12: **if** $Na = 0$ **then**
13: $flag = 0$.
14: **end if**
15: **end while**

Proof: See the Appendix. ■

For the conjunctive lattice PWA approximation, in case (20) is violated, i.e., there are two points $\boldsymbol{x}_\alpha$ and $\boldsymbol{x}_\beta$ such that the following inequality holds:

$$\max_{j \in J_{\leq,\alpha}} \{\ell_j(\boldsymbol{x}_\beta)\} < u^*(\boldsymbol{x}_\beta) = \ell_{\text{act}(\beta)}(\boldsymbol{x}_\beta) \tag{29}$$

similar results can be obtained, and we omit here due to space limitation.

When (25) or (29) holds, in order to construct a continuous PWA function and to ensure the validity of (19) and (20) for every *sample point* in the line segment $\mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$, the line segment is recursively partitioned to generate new sample points, as Lines 4–11 in Algorithm 1 show, resulting in an additional sample dataset $\mathcal{X}_2 \times \mathcal{U}_2$. In Algorithm 1, $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i+1}$ refer to sample points in $\mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta) \cap (\mathcal{X}_1 \cup \mathcal{X}_2)$, arranged according to the sample sequence in $\mathcal{X}_1 \cup \mathcal{X}_2$.

Lemma 4 shows that if we add points according to Algorithm 1, resulting in an additional sample dataset $\mathcal{X}_2 \times \mathcal{U}_2$, (19) and (20) are satisfied for all the sample points in $\mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$.

Lemma 4: Given the line segment $\mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$ such that (19) or (20) is violated, if we add points according to Algorithm 1, then we have

$$\min_{j \in J_{\geq,i}} \{u_j(\boldsymbol{x}_k)\} \leq u_k(\boldsymbol{x}_k) \quad \forall \boldsymbol{x}_i, \boldsymbol{x}_k \in \mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta) \cap (\mathcal{X}_1 \cup \mathcal{X}_2) \tag{30}$$

and

$$\max_{j \in J_{\leq,i}} \{u_j(\boldsymbol{x}_k)\} \geq u_k(\boldsymbol{x}_k) \quad \forall \boldsymbol{x}_i, \boldsymbol{x}_k \in \mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta) \cap (\mathcal{X}_1 \cup \mathcal{X}_2) \tag{31}$$

in which $\mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta) \cap (\mathcal{X}_1 \cup \mathcal{X}_2)$ denotes the sample points that are within the line segment $\mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$ after resampling.

Proof: See the Appendix. ■

Algorithm 1 can be run repeatedly until, for all the sample points $(\boldsymbol{x}_i, u_i) \in (\mathcal{X}_1 \cup \mathcal{X}_2) \times (\mathcal{U}_1 \cup \mathcal{U}_2)$, we have (19) and (20); thus, Assumption 1 is satisfied and the resulting lattice PWA approximations equal the original control solution at all sample points.

The 2-D function in Example 1 illustrates the process of constructing the disjunctive and conjunctive lattice PWA approximations and the resampling procedure.

Example 1 (Continued)*:* The disjunction and conjunctive lattice PWA representations for Example 1 have been derived in Section II-B. Here, we resort to an approximation strategy, i.e., several points are sampled in the function domain (9) to illustrate the disjunctive and conjunctive lattice PWA approximations.

Given two sample points $\boldsymbol{x}_1 = (0.6, 0.7)^T$ and $\boldsymbol{x}_2 = (0.6, 0.2)^T$, which are shown in Fig. 1(a), the sampled affine functions are $\ell_{\text{act}(1)} = \ell_1$ and $\ell_{\text{act}(2)} = \ell_3$, respectively. Fig. 1(b) also shows the position of the points $P_1 = (\boldsymbol{x}_1, \ell_1(\boldsymbol{x}_1))$ and $P_2 = (\boldsymbol{x}_2, \ell_3(\boldsymbol{x}_2))$. The index sets are then expressed as (the affine function ℓ_2 has not been sampled yet)

$$J_{\geq,1} = \{1\}, J_{\geq,2} = \{1, 3\}, J_{\leq,1} = \{1, 3\}, J_{\leq,2} = \{3\}.$$

Apparently Assumption 1 is not satisfied, i.e., (19) is violated for $i = 1, k = 2$, and (20) is violated for $i = 2, k = 1$. According to Algorithm 1, the point $\boldsymbol{x}_3 = (0.6, 0.575)^T$ is added to the sample points set with ℓ_2 being identified, i.e., $\ell_{\text{act}(3)} = \ell_2$, and at this time, Assumption 1 is satisfied. The disjunctive and conjunctive lattice PWA approximations can be written and simplified as

$$\begin{aligned}\hat{f}_{\text{L,d}}(\boldsymbol{x}) &= \max\{\min\{\ell_1, \ell_2\}, \min\{\ell_1, \ell_2\}, \min\{\ell_1, \ell_3\}\} \\ &= \max\{\min\{\ell_1, \ell_2\}, \min\{\ell_1, \ell_3\}\}\end{aligned} \tag{32}$$

and

$$\begin{aligned}\hat{f}_{\text{L,c}}(\boldsymbol{x}) &= \min\{\max\{\ell_1, \ell_3\}, \max\{\ell_2, \ell_3\}, \max\{\ell_2, \ell_3\}\} \\ &= \min\{\max\{\ell_1, \ell_3\}, \max\{\ell_2, \ell_3\}\}\end{aligned} \tag{33}$$

respectively.

Readers can verify that as (22) holds, the lattice PWA approximations coincide with the function (9) in base regions containing the sample points $\boldsymbol{x}_1, \boldsymbol{x}_2$, and $\boldsymbol{x}_3$. For the conjunctive approximation (33), we have

$$\mathcal{C}(\max\{\ell_1, \ell_3\}) = \{\Omega_1, \Omega_{3,1}, \Omega_{3,2}\}$$
$$\mathcal{C}(\max\{\ell_2, \ell_3\}) = \{\Omega_2, \Omega_{3,2}, \Omega_{3,3}\}$$

as $\mathcal{C}(\max\{\ell_1, \ell_3\}) \cup \mathcal{C}(\max\{\ell_2, \ell_3\})$ covers the entire domain, the conjunctive representation (11) and approximation (33) are identical. Besides, we can see that the term $\max\{\ell_1, \ell_3\}$, which is generated through the sample point $\boldsymbol{x}_1$, covers base regions $\Omega_{3,1}$ and $\Omega_{3,2}$ other than Ω_1, in which $\boldsymbol{x}_1$ lies in. Apart from Ω_2, the term $\max\{\ell_2, \ell_3\}$ also covers base regions $\Omega_{3,2}$ and $\Omega_{3,3}$. Therefore, in general, in generating sample points to approximate a continuous PWA function, we need far fewer sample points than the number of base regions. We will also show this conclusion in Section V.

For the disjunctive approximation (10), as

$$\mathcal{C}(\min\{\ell_1, \ell_2\}) = \{\Omega_1, \Omega_2\}, \mathcal{C}(\min\{\ell_1, \ell_3\}) = \{\Omega_{3,3}\}$$

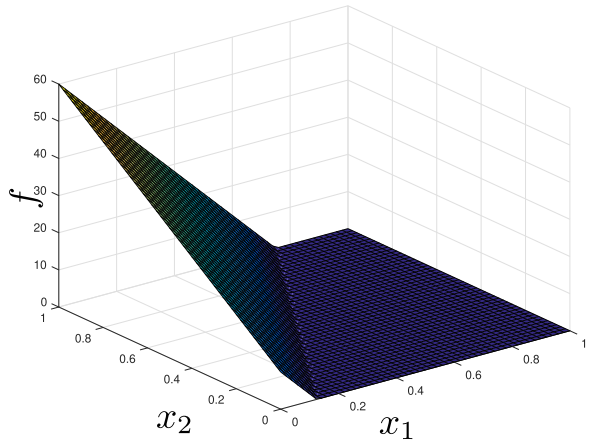


Fig. 3. Deviation between the disjunctive lattice PWA approximation and function (9) in Example 1.

Algorithm 2: Construction and simplification of lattice PWA approximations.

Input: Linear MPC problem.
Output: Simplified disjunctive and conjunctive lattice PWA approximations.

1: Generate sample dataset $\mathcal{X}_1 \times \mathcal{U}_1$.
2: Remove unnecessary sample points according to Lemma 2.
3: Generate additional sample dataset $\mathcal{X}_2 \times \mathcal{U}_2$ according to Algorithm 1. Let $\mathcal{X} = \{\mathcal{X}_1, \mathcal{X}_2\}, \mathcal{U} = \{\mathcal{U}_1, \mathcal{U}_2\}$.
4: **for** $\boldsymbol{x}_i \in \mathcal{X}$ **do**
5: Calculate the index sets $J_{\geq,i}$ and $J_{\leq,i}$ at the point $\boldsymbol{x}_i$ according to Equation (16) and (18), respectively.
6: **end for**
7: Construct the disjunctive and conjunctive lattice PWA approximations according to Equation (15) and (17).
8: Simplify the lattice PWA approximations according to the rule R1 (24).

there are base regions $\Omega_{3,1}$ and $\Omega_{3,2}$ that have not been covered, hence the disjunctive approximation and representation are not identical. The deviation of the disjunctive approximation (32) and representation is shown in Fig. 3, which is not zero in base regions $\Omega_{3,1}$ and $\Omega_{3,2}$. Later, we will revisit this example in Section IV-A to show the conditions for error-free approximations.

The process of obtaining disjunctive and conjunctive lattice PWA approximations and simplifying them can be summarized in Algorithm 2.

E. Online Evaluation of Lattice PWA Approximation

Supposing that we have obtained a simplified lattice PWA approximation for a linear MPC problem, the online evaluation reduces to affine functions evaluation, comparing affine functions in each term and comparing terms in a lattice PWA approximation. The comparison is fulfilled through a so-called structure matrix [33], which is a binary matrix containing the elements zero and one, i.e., $\mathcal{S}_{i,j} = 1$ indicates that the literal ℓ_j appears in the ith term, and $\mathcal{S}_{i,j} = 0$ shows otherwise. The size of the structure matrix is $N_t \times M$, in which N_t and M are the number of terms and literals in the lattice PWA approximation obtained through Algorithm 2.

Hence, for the lattice PWA approximations in Example 1, the structure matrices for $\hat{f}_{\mathrm{L,d}}$ and $\hat{f}_{\mathrm{L,c}}$ are

$$\mathcal{S}_{\mathrm{L,d}} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}, \quad \mathcal{S}_{\mathrm{L,c}} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

respectively. For the online evaluation of $\hat{f}_{\mathrm{L,d}}$ at some state $\boldsymbol{x}(t)$, one can construct another parameter-structure matrix $\mathcal{P}_{\mathrm{L,d}}$, in which the one element $\mathcal{S}_{i,j} = 1$ in $\mathcal{S}_{\mathrm{L,d}}$ is replaced with $\ell_j(\boldsymbol{x}(t))$. Then, one can perform a min–max operation on the matrix

$$\mathcal{P}_{\mathrm{L,d}} = \begin{bmatrix} \ell_1 & \ell_2 & 0 \\ \ell_1 & 0 & \ell_3 \end{bmatrix}$$

i.e., minimum of the columns and then maximum of the rows, to evaluate $\hat{f}_{\mathrm{L,d}}$. The evaluation of $\hat{f}_{\mathrm{L,c}}$ is similar, in which the maximum across the columns is taken first, and then the minimum is performed among the rows.

F. Complexity Analysis

1) Online Complexity: Assuming that there are N_t terms in the final approximation, according to Section III-E, take disjunctive lattice PWA approximation for example, the online evaluation of lattice PWA expression involves calculating the value of $\ell_j(\boldsymbol{x}(t)) \forall j = 1, \ldots, M$, minimum among the columns and then maximum of the rows. To calculate $\ell_j(\boldsymbol{x}(t)) \forall j = 1, \ldots, M$, $(n_x + 1) \cdot M$ multiplications and $n_x \cdot M$ additions are needed. For the comparison, the minimum takes at most $N_t \cdot (M - 1)$ comparisons while the maximum takes at most $N_t - 1$ comparisons. Therefore, the worst-case online complexity can be written as $O((n_x + 1) \cdot M + n_x \cdot M + N_t(M - 1) + N_t - 1)$. In general, $n_x \ll N_t$, hence the worse-case online complexity can be roughly approximated as $O(N_t \cdot M)$. In general, we have $N_t \ll N_s$, so the online evaluation is speedy.

As the comparisons are performed through the matrix $\mathcal{P}_{\mathrm{L,d}}$, they can be accelerated by parallel computing. Hence, the actual complexity for online evaluation is generally a fraction of $O(N_t \cdot M)$, depending on the number of cores in the processor. The online complexity is similar for evaluating conjunctive lattice PWA approximation.

2) Storage Requirements: Take the disjunctive lattice PWA approximation for example, supposing that it has N_t terms; we must store $(n_x + 1) \cdot M$ real numbers for affine functions and $\sum_{i=1}^{N_t} |J_{\geq,i}|$ binary numbers for the structure matrix $\mathcal{S}_{\mathrm{L,d}}$, in which $|J_{\geq,i}|$ is the number of elements in the set $J_{\geq,i}$.

As $|J_{\geq,i}| \leq M$, in total at most $(n_x + 1) \cdot M$ real numbers and $M \cdot N_t$ binary numbers must be stored.

In many cases, we have $N_t \ll N_s$, so the storage requirement for the disjunctive lattice PWA approximation is very small.

For the conjunctive lattice PWA approximation, we achieve the same result.

3) Offline Complexity: The offline time complexity for deriving disjunctive and conjunctive lattice PWA approximations, i.e., the running time of Algorithm 2, consists of two parts. One

concerns the training points sampling and resampling to obtain preliminary lattice PWA approximations, and the other concerns the complexity of construction and simplification of lattice PWA approximations through sample points obtained.

Lemma 5: The worst-case complexity of deriving the disjunctive and conjunctive lattice PWA approximations is $O(M \cdot N_s^2)$, in which M is the number of distinct affine functions, and N_s is the number of sample points in $\mathcal{X}$.

Proof: See the Appendix. ■

It is noted that $O(M \cdot N_s^2)$ is the worst-case complexity. In the simulation results, we can see that the offline calculation is not time-consuming.

IV. Approximation Error

A. Deviations Between the Disjunctive and Conjunctive Approximations

After sampling and resampling, as shown in Algorithm 2, we obtain the sample point set $\mathcal{X} \times \mathcal{U}$. Assuming that $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_s}\}$, given both the disjunctive and conjunctive lattice PWA approximations, under the following assumption, the deviation between the approximations and the optimal control law can be derived.

Assumption 2: We assume that all the distinct affine functions defining the first input have been sampled in the domain of interest Ω.

All the distinct affine functions for the first input in a PWA MPC optimal controller can be obtained by collecting all critical regions Ω_i and corresponding local affine functions $\ell_{\text{loc}(i)}(x)$ as defined in Definition 1 and selecting the distinct ones.

Assumption 2 can be explained as follows. For example, as Example 1 shows, there are five regions $\Omega_1, \Omega_2, \Omega_{3,1}, \Omega_{3,2}$, and $\Omega_{3,3}$, and the corresponding local affine functions $\ell_{\text{loc}(1)}(\boldsymbol{x}) = \ell_1(\boldsymbol{x}), \ell_{\text{loc}(2)}(\boldsymbol{x}) = \ell_2(\boldsymbol{x})$, and $\ell_{\text{loc}(3,1)}(\boldsymbol{x}) = \ell_{\text{loc}(3,2)}(\boldsymbol{x}) = \ell_{\text{loc}(3,3)}(\boldsymbol{x}) = \ell_3(\boldsymbol{x})$, then the distinct affine functions are $\ell_1(\boldsymbol{x})$ and $\ell_2(\boldsymbol{x}), \ell_3(\boldsymbol{x})$, and by sampling points in Ω_1, Ω_2, and $\Omega_{3,1}$, we have sampled all the distinct affine functions.

Theorem 2 bounds the error between the lattice PWA approximations and the original optimal control law.

Theorem 2: Supposing that

$$\hat{f}_{\text{L,d}}(\boldsymbol{x}) = \max_{i \in \{1,\ldots,N_s\}} \left\{ \min_{j \in J_{\geq,i}} \{u_j(\boldsymbol{x})\} \right\}$$

and

$$\hat{f}_{\text{L,c}}(\boldsymbol{x}) = \min_{i \in \{1,\ldots,N_s\}} \left\{ \max_{i \in J_{\leq,i}} \{u_i(\boldsymbol{x})\} \right\}$$

are the disjunctive and conjunctive approximations of the optimal control law $u^*(\boldsymbol{x})$ over the domain Ω, assuming that Assumption 2 holds, and defining

$$\varepsilon = \max_{\boldsymbol{x} \in \Omega} \left\{ \hat{f}_{\text{L,c}}(\boldsymbol{x}) - \hat{f}_{\text{L,d}}(\boldsymbol{x}) \right\} \tag{34}$$

we then have

$$-\varepsilon \leq \hat{f}_{\text{L,d}}(\boldsymbol{x}) - u^*(\boldsymbol{x}) \leq 0 \tag{35}$$

and

$$0 \leq \hat{f}_{\text{L,c}}(\boldsymbol{x}) - u^*(\boldsymbol{x}) \leq \varepsilon. \tag{36}$$

Furthermore, if $\varepsilon = 0$, we have

$$\hat{f}_{\text{L,d}}(\boldsymbol{x}) = \hat{f}_{\text{L,c}}(\boldsymbol{x}) = u^*(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \Omega. \tag{37}$$

Proof: See the Appendix. ■

Theorem 2 provides the sufficient conditions for error-free lattice PWA approximations, and we claim that the two conditions, i.e., the validity of Assumption 2 and $\varepsilon = 0$, are also necessary. This is apparent, as if Assumption 2 is violated, the constructed lattice PWA approximations miss some affine pieces, and if $\varepsilon \neq 0$, at least one lattice PWA approximation introduces error.

Readers can verify that if an additional point is sampled for Example 1, see $\boldsymbol{x}_4$ in Fig. 1(a) with $\ell_{\text{act}(4)} = \ell_3$. Then, we have $J_{\geq,4} = \{3\}$, $J_{\leq,4} = \{1, 2, 3\}$, $\mathcal{C}(\ell_3) = \{\Omega_{3,1}, \Omega_{3,2}, \Omega_{3,3}\}$, and $\mathcal{C}(\max\{\ell_1, \ell_2, \ell_3\}) = \{\Omega_{3,2}\}$. In this case, the disjunctive and conjunctive lattice PWA approximations cover the entire domain and both equal the 2-D continuous PWA function (9).

Remark 4: For Example 1, four sample points are generated to obtain error-free approximations, and for the base region $\Omega_{3,1}$, there is no sample point. As a necessary condition for the validity of (37), i.e., the lattice PWA approximations are error free, Assumption 2 requires that all the distinct affine functions have been sampled. It is noted that this is not the same as the condition that all the base regions should be identified. As the work in [14] shows, the number of distinct affine functions in explicit MPC is generally far less than that of critical regions, as some critical regions can share the same affine function. Besides, the critical regions should be partitioned to get the base regions satisfying (8); hence, the number of critical regions is much less than that of the base regions. Therefore, for an error-free lattice PWA approximation, the prerequisite is that all the distinct affine functions have been sampled, and in general, sampling only a portion of base regions can achieve this.

B. Probabilistic Guarantees of Error-Free Approximations

In order to check whether (34) is zero, as both the disjunctive approximation $\hat{f}_{\text{L,d}}$ and conjunctive approximation $\hat{f}_{\text{L,c}}$ are continuous PWA functions, so whether $\hat{f}_{\text{L,d}} = \hat{f}_{\text{L,c}}$ can be checked in each linear subregion of both $\hat{f}_{\text{L,d}}$ and $\hat{f}_{\text{L,C}}$. Or we can check the equivalence of the disjunctive and conjunctive PWA approximations by solving a continuous piecewise linear programming problem, i.e., the difference of the two kinds of approximations is set as the cost, which is continuous PWA as the difference of two continuous PWA functions is still continuous PWA. If the minimum of the cost is zero in the whole domain Ω, the disjunctive and conjunctive lattice PWA approximations are equivalent.

In general, the continuous piecewise linear programming is not convex, hence here for simplicity we resort to a statistical method, i.e., generating N_v independent identically distributed (i.i.d.) sample points, which constitute a validation dataset $\mathcal{X}_{\text{validate}} = \{\boldsymbol{x}_i, i = 1, \ldots, N_v\}$. For each sample point, we

define an indicator function as

$$I(\boldsymbol{x}_i) := \begin{cases} 1 & \text{if } \hat{f}_{\mathrm{L,d}}(\boldsymbol{x}_i) = \hat{f}_{\mathrm{L,c}}(\boldsymbol{x}_i) \\ 0 & \text{if } \hat{f}_{\mathrm{L,d}}(\boldsymbol{x}_i) \neq \hat{f}_{\mathrm{L,c}}(\boldsymbol{x}_i) \end{cases}$$

and then the random variables $I(\boldsymbol{x}_i), i = 1, \ldots, N_v$ are also i.i.d. Denoting the probability for $I(\boldsymbol{x}_i = 1)$ as μ, i.e., $\mathbb{P}[I(\boldsymbol{x}_i) = 1] = \mu$, then according to Hoeffding's inequality [34], we have

$$\mathbb{P}[|\mu - \bar{I}| \geq \epsilon] \leq 2\exp(-2N_v\epsilon^2)$$

in which $\bar{I} = \frac{1}{N_v}\sum_{k=1}^{N_v} I(\boldsymbol{x}_k)$. Therefore, we have

$$\mathbb{P}[\mu \geq \bar{I} - \epsilon] > 1 - 2\exp(-2N_v\epsilon^2)$$

meaning that with confidence $1 - 2\exp(-2N_v\epsilon^2)$, the probability that the lattice PWA approximations $\hat{f}_{\mathrm{L,d}}$ and $\hat{f}_{\mathrm{L,c}}$ are identical is larger than $\bar{I} - \epsilon$. If $\bar{I} = 1$, then by setting a small enough threshold ϵ, we can say that with confidence $1 - 2\exp(-2N_v\epsilon^2)$, the lattice PWA approximations $\hat{f}_{\mathrm{L,d}}$ and $\hat{f}_{\mathrm{L,c}}$ are almost identical, and thus both equal the optimal control law. For example, if $\epsilon = 10^{-2}$, then $N_v \geq 5 \times 10^4$ can ensure that the confidence is almost 1.

V. Simulation Results

Example 3: Consider a 4-D example taken from [21]

$$\boldsymbol{x}(t+1) = A\boldsymbol{x}(t) + Bu(t)$$

where the matrices A and B can be expressed as

$$A = \begin{bmatrix} 0.4035 & 0.3704 & 0.2935 & -0.7258 \\ -0.2114 & 0.6405 & -0.6717 & -0.0420 \\ 0.8368 & 0.0175 & -0.2806 & 0.3808 \\ -0.0724 & 0.6001 & 0.5552 & 0.4919 \end{bmatrix}$$

$$B = \begin{bmatrix} 1.6124 & 0.4806 & -1.4512 & -0.6761 \end{bmatrix}^T.$$

The constraints for the state and input are $\|\boldsymbol{x}\|_\infty^T \leq 5$ and $|u| \leq 0.2$, respectively. The prediction horizon is taken to be $N_p = 17$. The value of matrices in the cost function is $Q = \mathrm{diag}\{1, 1, 1, 1\}$, $R = 0.2$, and $P = 0$. According to MPT3, the optimal control solution is a PWA function of the state $\boldsymbol{x}$, with 28 246 polyhedral regions. As stated in [14], for a 4-D system with 890 polyhedral regions, the process of partitioning the polyhedral regions into base regions is prohibitive, and for these 28 246 polyhedral regions, there will be an exponential number of base regions. Hence, the method in [14] is not applicable to the examples listed in this article.

To construct the disjunctive and conjunctive lattice PWA approximations, the domain of interest Ω is created by generating 2000 feasible trajectories starting in the user-defined region $\Omega_0 = [-2.5\ 2.5] \times [-0.5\ 0.5] \times [-0.5\ 0.5] \times [-0.5\ 0.5]$, which results in 36 060 sample points and takes 42.29 s. The 2000 initial points are selected uniformly in Ω_0, and if some point is not feasible, it is replaced with a random initial point that conforms to uniform distribution and is feasible. Among the 36 060 sample points, there are 103 distinct affine functions, and according to Lemma 2, only 4575 points are useful in constructing lattice PWA approximations, i.e., only 4575 base regions have sample points in it.

TABLE I
PARAMETERS OF LATTICE PWA APPROXIMATIONS ON EXAMPLE 3, IN WHICH N_{s1}, N_b, AND N_s ARE THE NUMBER OF SAMPLE POINTS GENERATED ON TRAJECTORIES, REMAINED AFTER REMOVING REDUNDANT ONES, AFTER RESAMPLING, RESPECTIVELY, AND N_t AND M ARE THE NUMBER OF TERMS AND DISTINCT AFFINE FUNCTIONS

Method	N_{s1}	N_b	N_s	N_t	M	#para
Disjunctive	36 060	4575	4576	118	104	12 792
Conjunctive				115		12 480

TABLE II
PERFORMANCE OF LATTICE PWA APPROXIMATIONS ON EXAMPLE 3, IN WHICH t_{samp}, t_{L}, t_{off}, t_{on}, AND $t_{\mathrm{M,on}}$ DENOTE THE COMPUTATION TIME FOR OFFLINE SAMPLING, OFFLINE CONSTRUCTION OF LATTICE PWA APPROXIMATIONS, TOTAL OFFLINE, AVERAGE ONLINE CALCULATION, MAXIMUM ONLINE CALCULATION, RESPECTIVELY

Method	t_{samp}[s]	t_{L}[s]	t_{off}[s]	$t_{\mathrm{on}}[\mu\mathrm{s}]$	$t_{\mathrm{m,on}}[\mu\mathrm{s}]$
Disjunctive	42.29	3.4	45.69	26.37	376
Conjunctive				26.26	140
Online MPC				304.6	16,900
Method in [21]					11.9

For the 4575 sample points, Algorithm 1 is performed to sample one additional sample and get an additional distinct affine function to ensure the satisfaction of Assumption 1.

Then, the disjunctive and conjunctive lattice PWA approximations are constructed, of which the detailed numbers of parameters are given in Table I. Table II lists offline as well as online complexity of lattice PWA approximations, and the comparisons with the state-of-the-art online MPC solver qpOASES [35] and method in [21] are also included. For online MPC and the method in [21], there are no need to sample, hence the fields $t_{\mathrm{samp}}, t_{\mathrm{L}},$ and t_{off} are empty. Besides, as the work in [21] only lists the maximum online calculation complexity, the mean online calculation complexity for the method in [21] is also not listed in Table II. The online evaluation time is the mean of those 30 000 trials, and the worst-case online complexity is also listed. All the computations in this article are implemented through MATLAB 2024a (MathWorks, USA) on an Apple M3 Max computer.

It can be seen from Table II that the offline calculation time is mainly for the sampling on the trajectories, and the construction of lattice PWA approximation only takes 3.4 s. It is noted that the online evaluation time in [21] is obtained through computing the flops of a binary search tree

$$\mathrm{flops}_{\mathrm{tree}} = (n_x + 1) \cdot 2^{n_x} + n_x \cdot (3 + l_{\max} - l_0) - 1 \quad (38)$$

where $l_{\max}$ and l_0 are the finest and coarsest levels of detail, respectively, and n_x is the dimension. The online calculation time is calculated by assuming a processor speed of 1 Gflops/s. Hence, for real applications in more sophisticated computers, the online evaluation time is shorter than the value 11.9 μs, which outperforms our online speed.

Recall that the worst-case complexity for the online evaluation of lattice PWA approximation is $O(N_t \cdot M)$, in which N_t and M are the number of terms and distinct affine functions, respectively. For this example, take the disjunctive lattice PWA approximation for instance, we have $N_t = 118$ and $M = 104$, and the worst-case complexity for online evaluation is $O(12\,272)$,

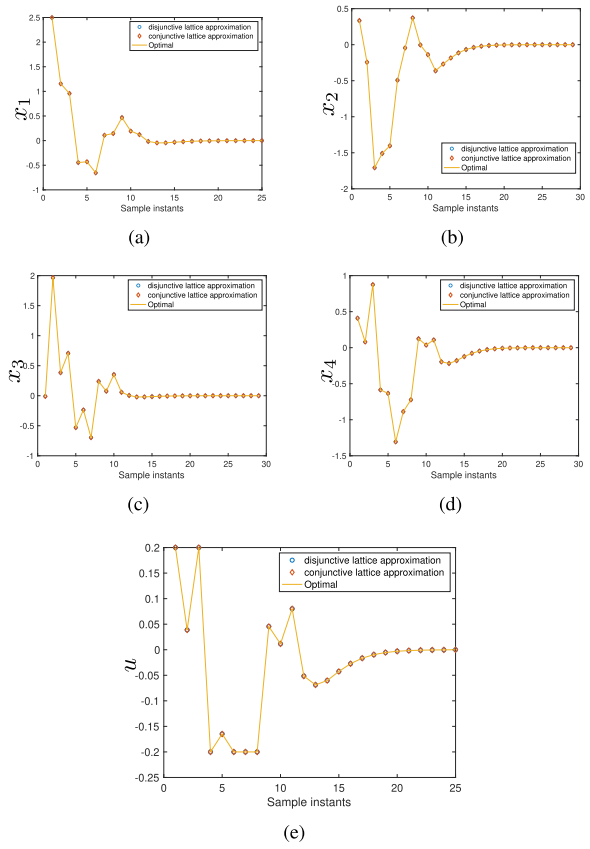


Fig. 4. One exemplary closed-loop simulation of Example 3.

whereas the flops of the binary search tree is 119. However, as can be seen from (38), the calculation time for binary search increases exponentially with n_x, and if we aim to derive a more accurate approximation, $l_{\max}$ is also large. Hence, for the next 10-D example, the worst-case online complexity of lattice PWA approximations is less than that in [21].

Besides, for the method in [21], one has to store the information of a series of hypercubic regions, which will result in a large storage requirement, especially for high-dimensional systems. In contrast, we have to store at most $(n_x + 1) \cdot M$ real number and $M \cdot N_t$ binary numbers, and in this example, we only have to store 520 real numbers and 118×104 binary numbers.

To check whether the lattice PWA approximations are error free in the domain of interest Ω, as Ω has not been explicitly expressed due to the complexity of calculating the convex combination of the 36 060 sample points in $\mathbb{R}^4$, we then generate 4000 convergent trajectories with initial points in Ω_0, resulting in 75 064 test points. It has been tested through those 75 064 test data points that the two lattice PWA approximations are identical in the region Ω. By setting $\epsilon = 10^{-2}$, it can be concluded that with confidence 1, the probability that the approximated lattice PWA control laws equal the optimal control is larger than 0.99. For the 75 064 points, the optimal explicit linear MPC control law is also calculated, and it is found that the lattice PWA approximations are error free in Ω.

Fig. 4 shows one exemplary closed-loop simulation of the example, and we can see from the figure that the optimal state

TABLE III
PARAMETERS OF LATTICE PWA APPROXIMATIONS ON EXAMPLE 4, IN WHICH N_{s1}, N_b, AND N_s ARE THE NUMBER OF SAMPLE POINTS GENERATED, AFTER REMOVING REDUNDANT ONES, AND AFTER RESAMPLING, RESPECTIVELY, AND N_t AND M ARE THE NUMBER OF TERMS AND LITERALS

Method	N_p	N_{s1}	N_b	N_s	N_t	M	#para
Disjunctive	10	76,322	118	118	13	24	576
Conjunctive					11		528
Disjunctive	20	75,163	118	123	13	23	552
Conjunctive					13		552

TABLE IV
PERFORMANCE OF LATTICE PWA APPROXIMATIONS ON EXAMPLE 4, IN WHICH t_{samp}, t_{L}, t_{off}, t_{On}, AND $t_{\text{m,on}}$ DENOTE THE SAMPLING, LATTICE PWA APPROXIMATION CONSTRUCTION, TOTAL OFFLINE, AVERAGE ONLINE, AND MAXIMUM ONLINE COMPUTATION TIME, RESPECTIVELY

Method	N_p	t_{samp}[s]	t_{L}[s]	t_{off}[s]	$t_{\text{on}}[\mu s]$	$t_{\text{m,on}}[\mu s]$
Disjunctive	10	139.1	0.032	139.64	3.28	499
Conjunctive					3.14	480
Online MPC					219.4	9,600
Method in [6]				72.85		
Disjunctive	20	1223.14	0.016	1223.16	3.14	204
Conjunctive					3.15	110
Online MPC					932.22	29,200
Method in [6]						

trajectory and the trajectory with the lattice PWA approximations as inputs are identical. Compared with the result in [21], here we have obtained an error-free solution.

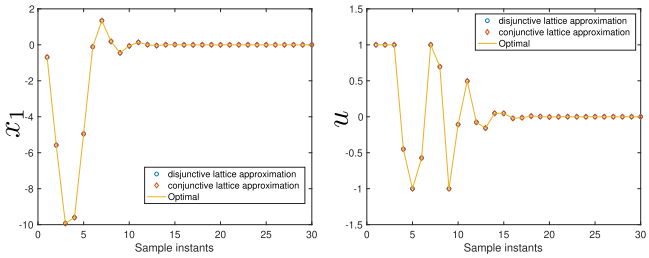
Example 4: Consider an example taken from [6], which is an mpQP problem constructed from the typical MPC setup with $\boldsymbol{x} \in \mathbb{R}^{10}$, $u \in \mathbb{R}$, $P = Q = I_{10}$, $R = 1$, $\mathcal{X} = \{\boldsymbol{x} | -10 \leq x_i \leq 10, i = 1, \ldots, n_x\}$, and $\mathcal{U} = \{u | -1 \leq u \leq 1\}$. The prediction model is obtained by discretizing the model $1/(s+1)^{10}$ with sampling time of 1 s and then converting the discretized model into a state-space form. The prediction horizon is taken to be $N_p = 10$. This is a complex problem, as the dimension is much higher.

Here, to construct the disjunctive and conjunctive lattice PWA approximations, 3500 feasible trajectories are generated with initial points in the region $\Omega_0 = [-2, 2]^{10}$. The numbers of parameters in disjunctive and conjunctive lattice PWA approximations are given in Table III .

The offline and online complexity of lattice PWA approximations when the prediction horizon N is 10 are given in Table IV . Comparisons with qpOASES [35] are also listed. Similar to Table II, the fields $t_{\text{samp}}, t_{\text{L}},$ and t_{off} are empty for online MPC. Besides, as the method in [6] focused on the traversing of the critical regions of the optimal control law, the fields $t_{\text{samp}}, t_{\text{L}}, t_{\text{on}},$ and $t_{\text{m,on}}$ are left empty.

It can be seen from Tables III and IV that lattice PWA approximations for Example 4 require small storage requirements and exhibit low online complexity. According to Section III-F, here the worst-case online complexity is $O(13 \times 24)$, which is $O(312)$. In contrast, the worst-case online flops for the method in [21] is 11 363 by using (38), hence for this high-dimensional example, the online speed of lattice PWA approximations will outperform that in [21]. It is apparent that the lattice PWA approximations result in a very low online computational burden.

It has been tested through 76 765 test data points that the two lattice PWA approximations are identical and equal the optimal



(a) (b)

Fig. 5. One exemplary closed-loop simulation of Example 4.

linear MPC control law in the control invariant set formulated by convergent trajectories starting from points in the region $\Omega_0 = [-2, 2]^{10}$. By setting $\epsilon = 10^{-2}$, it can be concluded that with confidence 1, the probability that the approximated lattice PWA control laws equal the optimal control law is larger than 0.99.

Fig. 5 shows one exemplary closed-loop simulation of the example (only the first state variable is listed), and we can see from the figure that the optimal state trajectory and the trajectory with the lattice PWA approximations as inputs are identical.

As indicated in [6], MPT3 failed to solve the problem. Moreover, the combinatorial approach proposed in [6] successfully enumerates all optimal active sets and creates all the 573 critical regions, taking 72.85 s. According to the definition of base region, the number of base regions for this example is far more than 573, which is much larger than the number of useful sample points, i.e., 118 in Table III. It is notable that although the calculation time for our procedure is longer, we obtain the final continuous PWA expression of the optimal control law in a simplified form, which is much easier to implement online.

To demonstrate more clearly the efficacy of the lattice PWA approximations, the prediction horizon N_p is extended to 20, and the number of parameters and complexity of lattice PWA approximations are also listed in Tables III and IV, respectively. When $N_p = 20$, Ahmadi-Moshkenani et al. [6] did not provide a result either, hence corresponding fields in Table II are left blank. In this case, we generate 75 643 validation points to show that with confidence 1, the probability that the approximated lattice PWA control laws equal the optimal control law is larger than 0.99.

VI. Conclusion and Future Work

This article presents disjunctive and conjunctive lattice PWA approximations of the explicit linear MPC control law by sampling, and resampling in the state domain. Under certain conditions, the lattice PWA approximated and exact control laws are identical in base regions that contain the sample points. Furthermore, assuming that all the affine functions have been identified in the domain of interest, the disjunctive lattice PWA approximation is always smaller than the original optimal control law, whereas the conjunctive lattice PWA approximation is always larger. Then, the equivalence of the disjunctive and conjunctive lattice PWA approximations guarantees the equivalence to the optimal control law. The two kinds of lattice PWA approximations have been simplified to reduce the storage and online evaluation complexity further. The complexity of the online and offline approximation and the storage requirements have been analyzed. Simulation results show that we can obtain statistically error-free lattice PWA approximations calculated with relatively small computational costs. Besides, the online computational complexity is much less than the state-of-the-art methods for a 10-D system.

It is noted that the error-free lattice PWA approximations equal the optimal MPC controller only in a part of the feasible region, i.e., the domain of interest is set as the combination of sample points in convergent trajectories that originate in a user-specified region. This applies when the system states are not far away from the operation points. If the state varies largely, a more general domain of interest should be considered, which may result in thousands of distinct affine functions. In this case, it is not easy to construct error-free approximations, and the approximation error will be considered in the future. Corresponding feasibility and stability analysis will also be investigated.

Appendix A
Proofs

Proof of Theorem 1.

Proof: Following gives the proof for the disjunctive case, i.e., the first equality in (21) and (23), and the proof for the conjunctive case, i.e., the second equality in (21) and (23), follows similarly.

Substituting $\boldsymbol{x}_k \forall k \in \{1, \ldots, N_{s1}\}$, into the right-hand side of (15) and we get

$$\max_{i=1,\ldots,N_{s1}} \left\{ \min_{j \in J_{\geq,i}} \{\ell_j(\boldsymbol{x}_k)\} \right\}$$

which equals $\ell_{\text{act}(k)}(\boldsymbol{x}_k)$ as (19) holds. Hence, the equivalence of disjunctive lattice PWA approximation and the optimal control law at $\boldsymbol{x}_k$ is guaranteed, and the first equality of (21) is proved.

Then, we prove the validity of the first equality in (23) under a stricter condition (22), other than (19). As (22) holds, we have $\forall i, k$, either

$$\min_{j \in J_{\geq,i}} \{\ell_j(\boldsymbol{x}_k)\} = \ell_{\text{act}(k)}(\boldsymbol{x}_k), \text{act}(k) \in J_{\geq,i} \tag{39}$$

or

$$\min_{j \in J_{\geq,i}} \{\ell_j(\boldsymbol{x}_k)\} < \ell_{\text{act}(k)}(\boldsymbol{x}_k). \tag{40}$$

For these two cases, we will show that the following inequality holds:

$$\min_{j \in J_{\geq,i}} \{\ell_j(\boldsymbol{x})\} \leq \ell_{\text{act}(k)}(\boldsymbol{x}) \quad \forall i \in \{1, \ldots, N_{s1}\}, \forall \boldsymbol{x} \in \Gamma_{\text{pnt}(k)}. \tag{41}$$

Case 1: (39) is valid for some i, k. Then, as $\text{act}(k) \in J_{\geq,i}$, we have (41).

Case 2: (40) holds for some i, k. In this case, we prove by contradiction. Assuming that (41) does not hold, then there is some $\boldsymbol{x}_0 \in \Gamma_{\text{pnt}(k)}$ with $\hat{f}_{\text{L,d}}(\boldsymbol{x}_0) = \ell_{\text{act}(k)}(\boldsymbol{x}_0)$, such that

$$\min_{j \in J_{\geq,i}} \{\ell_j(\boldsymbol{x}_0)\} > \ell_{\text{act}(k)}(\boldsymbol{x}_0).$$

As both sides of the above inequality are continuous, we can then find a neighborhood of $\boldsymbol{x}_0$, say $\mathcal{B}(\boldsymbol{x}_0) \cap \text{int}(\Gamma_{\text{pnt}(k)})$, in which $\text{int}(\Gamma_{\text{pnt}(k)})$ denotes the interior of $\Gamma_{\text{pnt}(k)}$, such that $\forall \boldsymbol{x} \in \mathcal{B}(\boldsymbol{x}_0) \cap \text{int}(\Gamma_{\text{pnt}(k)})$, we have

$$\min_{j \in J_{\geq,i}} \{\ell_j(\boldsymbol{x})\} > \ell_{\text{act}(k)}(\boldsymbol{x}).$$

Randomly choosing a point $\bar{\boldsymbol{x}}_0 \in \mathcal{B}(\boldsymbol{x}_0) \cap \text{int}(\Gamma_{\text{pnt}(k)})$, and the following conclusion follows directly:

$$\ell_j(\bar{\boldsymbol{x}}_0) > \ell_{\text{act}(k)}(\bar{\boldsymbol{x}}_0) \quad \forall j \in J_{\geq,i}. \tag{42}$$

Considering the line segment

$$\mathcal{L}(\boldsymbol{x}_k, \bar{\boldsymbol{x}}_0) = \lambda \boldsymbol{x}_k + (1-\lambda)\bar{\boldsymbol{x}}_0, \lambda \in [0,1]$$

as the base region $\Gamma_{\text{pnt}(k)}$ is convex, we have

$$\mathcal{L}(\boldsymbol{x}_k, \bar{\boldsymbol{x}}_0) \subset \Gamma_{\text{pnt}(k)}.$$

According to (40), there should be some $j_0 \in J_{\geq,i}$ such that

$$\ell_{j_0}(\boldsymbol{x}_k) < \ell_{\text{act}(k)}(\boldsymbol{x}_k).$$

Combined with (42), there must be some point $\hat{\boldsymbol{x}} \in \mathcal{L}(\bar{\boldsymbol{x}}_0, \boldsymbol{x}_k)$ such that

$$\ell_{j_0}(\hat{\boldsymbol{x}}) = \ell_{\text{act}(k)}(\hat{\boldsymbol{x}}). \tag{43}$$

As $\bar{\boldsymbol{x}}_0 \in \text{int}(\Gamma_{\text{pnt}(k)})$, we have $\hat{\boldsymbol{x}} \in \text{int}(\Gamma_{\text{pnt}(k)})$. Then, the validity of (43) contradicts the definition of the base region (8). Hence, (41) holds.

As the equality in (41) holds for $i = k$, i.e.,

$$\min_{j \in J_{\geq,k}} \{u_j(\boldsymbol{x})\} = \ell_{\text{act}(k)}(\boldsymbol{x}) \quad \forall k \in \{1, \ldots, N_{s1}\}, \forall \boldsymbol{x} \in \Gamma_{\text{pnt}(k)}$$

combining with the disjunctive lattice PWA approximation (15), we have

$$\hat{f}_{\text{L,d}}(\boldsymbol{x}) = \ell_{\text{act}(k)}(\boldsymbol{x}) \quad \forall k \in \{1, \ldots, N_{s1}\}, \forall \boldsymbol{x} \in \Gamma_{\text{pnt}(k)}. \tag{44}$$

For a sample point $\boldsymbol{x}_k$, according to the sampling procedure, we have

$$u^*(\boldsymbol{x}_k) = \ell_{\text{act}(k)}(\boldsymbol{x}_k) \quad \forall k \in \{1, \ldots, N_{s1}\}.$$

As indicated in Section II-A, the equivalence of $u^*(\boldsymbol{x})$ and $\ell_{\text{act}(k)}(\boldsymbol{x})$ holds for all $\boldsymbol{x} \in \mathcal{CR}(\boldsymbol{x}_k)$.

As the work in [3] shows, critical regions are also obtained through the KKT conditions (4). Different critical regions that share the same affine function $\ell_{\text{act}(k)}$ are combined such that the resulting combination is convex, then according to the definition of base region, $\Gamma_{\text{pnt}(k)}$ is a subset of this combination and

$$u^*(\boldsymbol{x}) = \ell_{\text{act}(k)}(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \Gamma_{\text{pnt}(k)}.$$

The above equation together with (44) yields (23).

The conjunctive case can be proved similarly.

Proof of Lemma 2. ■

Proof: According to the definition of covered base regions, we have

$$\min_{j \in J_{\geq,i}} \{u_j(\boldsymbol{x})\} = u^*(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \mathcal{C}\left(\min_{j \in J_{\geq,i}} \{u_j(\boldsymbol{x})\}\right).$$

Then, for any $\Gamma_t \subset \mathcal{C}(\min_{j \in J_{\geq,i}} \{u_j(\boldsymbol{x})\})$, we have

$$\min_{j \in J_{\geq,i}} \{u_j(\boldsymbol{x})\} = \ell_{\text{act}(t)}(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \Gamma_t.$$

As $J_{\geq,k} \subset J_{\geq,i}$, we then have

$$\min_{j \in J_{\geq,k}} \{u_j(\boldsymbol{x})\} \geq \min_{j \in J_{\geq,i}} \{u_j(\boldsymbol{x})\} = \ell_{\text{act}(t)}(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \Gamma_t. \tag{45}$$

According to (22) and the proof of Theorem 1, we have

$$\min_{j \in J_{\geq,k}} \leq \ell_{\text{act}(t)}(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \Gamma_t$$

which together with (45) yield the following:

$$\min_{j \in J_{\geq,k}} = \ell_{\text{act}(t)}(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \Gamma_t$$

meaning that

$$\mathcal{C}\left(\min_{j \in J_{\geq,i}} \{u_j(\boldsymbol{x})\}\right) \subset \mathcal{C}\left(\min_{j \in J_{\geq,k}} \{u_j(\boldsymbol{x})\}\right).$$

Moreover, if there is $t \in \{1, \ldots, N_{s1}\} \setminus \{i\}$ such that $\text{act}(i) \in J_{\geq,t}$, then removing $\boldsymbol{x}_i$ will just removing corresponding term $\min_{j \in J_{\geq,i}} \{\ell_j(\boldsymbol{x})\}$ without eliminating any affine functions in other terms, thus the function value of $\hat{f}_{\text{L,d}}$ will not change according to Rule (24), which also means that the covered base regions do not change.

Proof of Lemma 3. ■

Proof: As the optimal control solution u^* is continuous PWA, and the feasible domain is convex, it is still continuous PWA when restricted to the line segment $\mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$.

Defining an index set $\text{aff}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$ as

$$\text{aff}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta) = \{j | \exists \boldsymbol{x} \in \mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta) \text{ such that } u^*(\boldsymbol{x}) = \ell_j(\boldsymbol{x})\}$$

i.e., the index set $\text{aff}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$ includes all the indices of affine functions in $u^*(\boldsymbol{x})$ when restricted to $\mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$. It is noted that the index set $\text{aff}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$ may contain affine functions that have not been sampled yet. In the following, the index set $\text{aff}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$ is used to illustrate the existence of $\boldsymbol{x}_\gamma \in \mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$ such that (27) holds, and there is no need to identify the details of $\text{aff}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$.

According to [14], we have

$$\min_{j \in S_{\geq,\alpha}} \{\ell_j(\boldsymbol{x})\} \leq \ell_{\text{act}(\beta)}(\boldsymbol{x}_\beta) \quad \forall \boldsymbol{x} \in \mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$$

in which $S_{\geq,\alpha}$ is the index set such that

$$S_{\geq,\alpha} = \{j \in \text{aff}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta) | \ell_j(\boldsymbol{x}_\alpha) \geq \ell_{\text{act}(\alpha)}(\boldsymbol{x}_\alpha)\}.$$

Clearly, there will be some $\text{act}(\gamma) \in S_{\geq,\alpha}$, such that (27) holds.

Therefore, if we add one of these $\boldsymbol{x}_\gamma$ to the sample point set, as (27) is valid, we have $\text{act}(\gamma) \in J_{\geq,\alpha}$, and (28) is valid.

Proof of Lemma 4. ■

Proof: The condition $\text{sign}(\ell_{\text{act}(i)}(\boldsymbol{x}_i) - \ell_{\text{act}(i+1)}(\boldsymbol{x}_i)) = \text{sign}$ $(\ell_{\text{act}(i)}(\boldsymbol{x}_{i+1}) - \ell_{\text{act}(i+1)}(\boldsymbol{x}_{i+1}))$ is equivalent to the condition

$$(\ell_{\text{act}(i)}(\boldsymbol{x}_i) - \ell_{\text{act}(i+1)}(\boldsymbol{x}_i)) \cdot (\ell_{\text{act}(i)}(\boldsymbol{x}_{i+1}) - \ell_{\text{act}(i+1)}(\boldsymbol{x}_{i+1})) > 0$$

which indicates either

$$\ell_{\text{act}(i)}(\boldsymbol{x}) > \ell_{\text{act}(i+1)}(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \mathcal{L}(\boldsymbol{x}_i, \boldsymbol{x}_{i+1})$$

or

$$\ell_{\text{act}(i)}(\boldsymbol{x}) < \ell_{\text{act}(i+1)}(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \mathcal{L}(\boldsymbol{x}_i, \boldsymbol{x}_{i+1}).$$

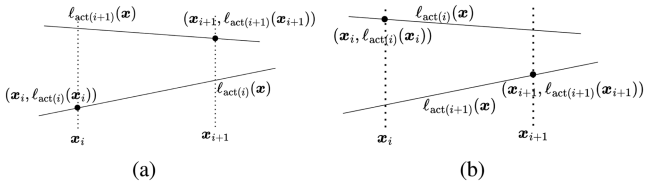


Fig. 6. Cases when $\text{sign}(\ell_{\text{act}(i)}(\boldsymbol{x}_i) - \ell_{\text{act}(i)+1}(\boldsymbol{x}_i)) = \text{sign}(\ell_{\text{act}(i)}(\boldsymbol{x}_{i+1}) - \ell_{\text{act}(i+1)}(\boldsymbol{x}_{i+1}))$. (a) $\ell_{\text{act}(i)}(\boldsymbol{x}) \le \ell_{\text{act}(i+1)}(\boldsymbol{x})$. (b) $\ell_{\text{act}(i)}(\boldsymbol{x}) \ge \ell_{\text{act}(i+1)}(\boldsymbol{x})$.

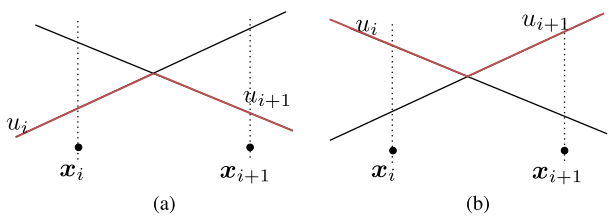


Fig. 7. Two cases satisfying (46). (a) Case 1. (b) Case 2.

Fig. 6 shows these two cases. This means that there will be no continuous PWA functions constructed to connect the two points $(\boldsymbol{x}_i, \ell_{\text{act}(i)}(\boldsymbol{x}_i))$ and $(\boldsymbol{x}_{i+1}, \ell_{\text{act}(i+1)}(\boldsymbol{x}_{i+1}))$. Hence, new points should be added in $\mathcal{L}(\boldsymbol{x}_i, \boldsymbol{x}_{i+1})$.

After evaluating Algorithm 1, the condition

$$(\ell_{\text{act}(i)}(\boldsymbol{x}_i) - \ell_{\text{act}(i+1)}(\boldsymbol{x}_i)) \cdot (\ell_{\text{act}(i)}(\boldsymbol{x}_{i+1}) - \ell_{\text{act}(i+1)}(\boldsymbol{x}_{i+1})) \le 0 \tag{46}$$

is satisfied for all points $\boldsymbol{x}_i \in \mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$, which corresponds to two cases, as shown in Fig. 7. To generalize, here we only consider the case when $\ell_{\text{act}(i)} \neq \ell_{\text{act}(i+1)}$; for the case $\ell_{\text{act}(i)} = \ell_{\text{act}(i+1)}$, the affine function $\ell_{\text{act}(i)}$ can connect the two points. Then, we can construct a continuous PWA function connecting the two points $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i+1}$, i.e., $\min\{\ell_{\text{act}(i)}, \ell_{\text{act}(i+1)}\}$ for case 1 and $\max\{\ell_{\text{act}(i)}, \ell_{\text{act}(i+1)}\}$ for case 2. Like the two cases in Fig. 7, we can construct a continuous PWA function, say $\hat{f}_1$, that connects all the points $(\boldsymbol{x}_i, \ell_{\text{act}(i)}(\boldsymbol{x}_i))$ for all $\boldsymbol{x}_i \in \mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$. It is noted that $\hat{f}_1$ may be different from u^*, and $\hat{f}_1$ is just used as a supplementary continuous PWA function to show the validity of (30) and (31). For this continuous PWA function $\hat{f}_1$, the affine pieces are $\ell_{\text{act}(i)}(\boldsymbol{x})$, $\boldsymbol{x}_i \in \mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$.

Define the index set

$$\text{aff}_1(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta) = \{j | \exists \boldsymbol{x} \in \mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta) \text{ such that } \hat{f}_1(\boldsymbol{x}) = \ell_j(\boldsymbol{x})\}$$

then $\text{aff}_1 = \{\text{act}(i) | \forall \boldsymbol{x}_i \in \mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta) \cap (\mathcal{X}_1 \cup \mathcal{X}_2)\}$.

According to [14], the continuous PWA function $\hat{f}_1$ can be represented using a lattice PWA expression, and take disjunctive lattice PWA representation for example, the conjunctive case can be proved similarly. For all $\boldsymbol{x}_i \in \mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta) \cap (\mathcal{X}_1 \cup \mathcal{X}_2)$, let $\bar{\Gamma}_{\text{pnt}(i)}$ is a base region that satisfies

$$\boldsymbol{x}_i \in \bar{\Gamma}_{\text{pnt}(i)}, \bar{\Gamma}_{\text{pnt}(i)} \subset \mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta).$$

For the disjunctive lattice PWA representation, the term concerning the base region $\bar{\Gamma}_{\text{pnt}(i)}$ is calculated as

$$\min_{j \in \bar{I}_{\ge,\text{pnt}(i)}} \{\ell_j\}$$

in which the index set $\bar{I}_{\ge,\text{pnt}(i)}$ can be expressed as

$$\bar{I}_{\ge,\text{pnt}(i)} = \{j \in \text{aff}_1(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta) | \ell_j(\boldsymbol{x}) \ge \ell_{\text{act}(i)}(\boldsymbol{x}), \forall \boldsymbol{x} \in \bar{\Gamma}_{\text{pnt}(i)}\}.$$

As $\boldsymbol{x}_i \in \bar{\Gamma}_{\text{pnt}(i)}$, we have $\bar{I}_{\ge,\text{pnt}(i)} \subset \bar{J}_{\ge,i}$, in which $\bar{J}_{\ge,i}$ is defined as

$$\bar{J}_{\ge,i} = \{j \in \text{aff}_1(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta) | \ell_j(\boldsymbol{x}_i) \ge \ell_{\text{act}(i)}(\boldsymbol{x}_i)\}.$$

As aff_1 is a subset of the entire affine function index set, we have

$$\bar{I}_{\ge,\text{pnt}(i)} \subset \bar{J}_{\ge,i} \subset J_{\ge,i}.$$

According to [14], the following inequality holds:

$$\min_{j \in \bar{I}_{\ge,\text{pnt}(i)}} \{\ell_j(\boldsymbol{x}_k)\} \le \ell_{\text{act}(k)}(\boldsymbol{x}_k) \quad \forall \boldsymbol{x}_i, \boldsymbol{x}_k \in \mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta) \cap (\mathcal{X}_1 \cup \mathcal{X}_2)$$

and as $\bar{I}_{\ge,\text{pnt}(i)} \subset J_{\ge,i}$, we have (30).

The validity of (31) can be proved similarly by referring to the conjunctive lattice PWA representation of $\hat{f}_1$.

Proof of Lemma 5.

Proof: As Algorithm 2 shows, the offline complexity comes from generating the sample set $\mathcal{X}_1$, and resampling to make (19) and (20) hold, and simplification according to the rule R1 (24).

Assuming that there are N_{s1} sample points generated on trajectories, as shown in Section III-A, the complexity of sampling N_{s1} points, i.e., evaluating Line 1 in Algorithm 2, includes solving N_{s1} convex quadratic programming (QP) problems and corresponding KKT conditions, which are solving linear equations. The solving of N_{s1} convex QP problems with $N_p \cdot n_u$ decision variables is approximately $O(N_{s1} \cdot L^2 (N_p \cdot n_u)^4)$ by using an interior-point algorithm [36], in which L is the bit length of the QP problem. The dominant algorithmic operation in solving the KKT conditions is solving N_{s1} matrix inversion problems, the worst-case complexity of which is $O(N_{s1}|\mathcal{A}^*|^3)$ using the Gauss–Jordan elimination algorithm, in which $|\mathcal{A}^*|$ is the number of active constraints. As $|\mathcal{A}^*| \le p$, where p is the number of constraints in QP (2), the worst-case complexity for solving the KKT conditions is $O(N_{s1} p^3)$.

We now discuss the worst-case complexity of evaluating Algorithm 1, i.e., Line 3 in Algorithm 2. For two points $\boldsymbol{x}_\alpha$ and $\boldsymbol{x}_\beta$, if (19) or (20) is violated, the evaluation of Algorithm 1 is a binary search method for identifying additional affine functions, which in the worst case requires determining all the omitted subregions in the line segment $\mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$. The maximum number of subregions appearing in $\mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$ is $\frac{d_{\alpha,\beta}}{\delta_M}$, where $d_{\alpha,\beta}$ is the length of $\mathcal{L}(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$ and δ_M is the minimum measure of subregions. The binary searching of the subregions yields a worst-case complexity of $O(\log_2 \frac{d_{\alpha,\beta}}{\delta_M})$. Supposing that there are N_a point pairs such that (19) is violated, then the worst-case complexity is $O(N_a \cdot \log_2 \frac{d_{\alpha,\beta}}{\delta_M})$. The number N_a is closely related to the number of sample points generated previously, i.e., N_{s1}. A larger N_{s1} will result in a smaller number of N_a and $d_{\alpha,\beta}$; hence, the complexity of Algorithm 1 can be decreased by increasing N_{s1}.

After evaluating Lines 1–4, there are N_s sample points. The simplification procedure requires the comparison of the sets $J_{\ge,i}$ $(J_{\le,i})$ for $i = 1, \ldots, N_s$, which at most yields $\binom{2}{N_s} = \frac{N_s(N_s-1)}{2}$ times comparisons. For each comparison, at most M literals need to be compared. Hence, the worst-case complexity for the simplification is $O(M \cdot N_s^2)$.

In general, $L, N_p, n_u, p \ll N_{s1}$, $N_a \ll N_{s1}$, and $N_{s1} < N_s$, then the total worst-case complexity is $O(M \cdot N_s^2)$, in which M is the number of literals, i.e., distinct affine functions and N_s is the number of sample points.

Proof of Theorem 2.

Proof: Assume that $\mathcal{N}$ is the index of all base regions, i.e., $\Omega = \bigcup_{i\in\mathcal{N}} \Gamma_i$, according to the conclusion in [14], we have

$$u^*(\boldsymbol{x}) = \max_{i\in\mathcal{N}} \left\{ \min_{j\in I_{\geq,i}} \{u_j(\boldsymbol{x})\} \right\} \quad \forall \boldsymbol{x} \in \Omega \tag{47}$$

$$u^*(\boldsymbol{x}) = \min_{i\in\mathcal{N}} \left\{ \max_{j\in I_{\leq,i}} \{u_j(\boldsymbol{x})\} \right\} \quad \forall \boldsymbol{x} \in \Omega \tag{48}$$

in which the index sets $I_{\geq,i}$ and $I_{\leq,i}$ are defined as

$$I_{\geq,i} = \{j | \ell_j(\boldsymbol{x}) \geq \ell_{\text{loc}(i)}(\boldsymbol{x}), \forall \boldsymbol{x} \in \Gamma_i\}$$
$$I_{\leq,i} = \{j | \ell_j(\boldsymbol{x}) \leq \ell_{\text{loc}(i)}(\boldsymbol{x}), \forall \boldsymbol{x} \in \Gamma_i\}.$$

Therefore, for all $\boldsymbol{x} \in \Omega$ and all $i \in \mathcal{N}$, we have

$$\min_{j\in I_{\geq,i}} \{\ell_j(\boldsymbol{x})\} \leq u^*(\boldsymbol{x}), \max_{j\in I_{\leq,i}} \{\ell_j(\boldsymbol{x})\} \geq u^*(\boldsymbol{x}).$$

For a sample point $\boldsymbol{x}_i$, the base region containing $\boldsymbol{x}_i$ is denoted as $\Gamma_{\text{pnt}(i)}$ and the affine function at $\boldsymbol{x}_i$ is $\ell_{\text{act}(i)}$, i.e., $u^*(\boldsymbol{x}_i) = \ell_{\text{act}(i)}(\boldsymbol{x}_i)$. According to the definition of base region and (16), we have

$$\begin{cases} J_{\geq,i} = I_{\geq,\text{pnt}(i)} & \boldsymbol{x}_i \in \text{int}(\Gamma_{\text{pnt}(i)}) \\ J_{\geq,i} \supset I_{\geq,\text{pnt}(i)} & \boldsymbol{x}_i \in \text{bd}(\Gamma_{\text{pnt}(i)}) \end{cases}$$

in which $\text{bd}(\cdot)$ denotes the boundary of some polyhedron. Then, the following is valid:

$$I_{\geq,\text{pnt}(i)} \subset J_{\geq,i}.$$

Similarly, according to (18), we have

$$I_{\leq,\text{pnt}(i)} \subset J_{\leq,i}.$$

As the sampled base regions are only a subset of all base regions, i.e.,

$$\{\text{pnt}(1), \ldots, \text{pnt}(N_s)\} \subset \mathcal{N}$$

we have $\forall \boldsymbol{x} \in \Omega, \forall i \in \{1, \ldots, N_s\}$

$$\min_{j\in J_{\geq,i}} \{\ell_j(\boldsymbol{x})\} \leq \min_{j\in I_{\geq,\text{pnt}(i)}} \{\ell_j(\boldsymbol{x})\} \leq u^*(\boldsymbol{x})$$

and

$$\max_{j\in J_{\leq,i}} \{\ell_j(\boldsymbol{x})\} \geq \max_{j\in I_{\leq,\text{pnt}(i)}} \{\ell_j(\boldsymbol{x})\} \geq u^*(\boldsymbol{x}).$$

Then, as $\hat{f}_{\text{L,d}}$ is the maximum of $\min_{j\in J_{\geq,i}} \ell_j(\boldsymbol{x}) \forall i \in \{1, \ldots, N_s\}$, and $\hat{f}_{\text{L,c}}$ is the minimum of $\max_{j\in J_{\leq,i}} \ell_j(\boldsymbol{x}) \forall i \in \{1, \ldots, N_s\}$, the following is valid:

$$\hat{f}_{\text{L,d}}(\boldsymbol{x}) \leq u^*(\boldsymbol{x}) \leq \hat{f}_{\text{L,c}}(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \Omega.$$

According to (34), we have (35) and (36).

Furthermore, if $\varepsilon = 0$, then both approximations are identical to the optimal control law in the region Ω, i.e., (37) holds. ■

References

[1] T. Samad, "A survey on industry impact and challenges thereof [Technical Activities]," *IEEE Control Syst. Mag.*, vol. 37, no. 1, pp. 17–18, Feb. 2017.

[2] M. Schwenzer, M. Ay, T. Bergs, and D. Abel, "Review on model predictive control: An engineering perspective," *Int. J. Adv. Manuf. Technol.*, vol. 117, no. 5, pp. 1327–1349, Nov. 2021.

[3] 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.

[4] M. Kvasnica, B. Takács, J. Holaza, and S. Di Cairano, "On region-free explicit model predictive control," in *Proc. IEEE 54th Conf. Decis. Control*, 2015, pp. 3669–3674.

[5] A. Gupta, S. Bhartiya, and P. Nataraj, "A novel approach to multiparametric quadratic programming," *Automatica*, vol. 47, no. 9, pp. 2112–2117, 2011.

[6] P. Ahmadi-Moshkenani, T. A. Johansen, and S. Olaru, "Combinatorial approach toward multiparametric quadratic programming based on characterizing adjacent critical regions," *IEEE Trans. Autom. Control*, vol. 63, no. 10, pp. 3221–3231, Oct. 2018.

[7] F. Borrelli, M. Baotić, J. Pekar, and G. Stewart, "On the computation of linear model predictive control laws," *Automatica*, vol. 46, no. 6, pp. 1035–1041, 2010.

[8] M. Herceg, S. Mariéthoz, and M. Morari, "Evaluation of piecewise affine control law via graph traversal," in *Proc. Eur. Control Conf.*, 2013, pp. 3083–3088.

[9] F. J. Christophersen, M. Kvasnica, C. N. Jones, and M. Morari, "Efficient evaluation of piecewise control laws defined over a large number of polyhedra," in *Proc. Eur. Control Conf.*, 2007, pp. 2360–2367.

[10] F. Bayat, T. A. Johansen, and A. Jalali, "Flexible piecewise function evaluation methods based on truncated binary search trees and lattice representation in explicit MPC," *IEEE Trans. Control Syst. Technol.*, vol. 20, no. 3, pp. 632–640, May 2012.

[11] N. A. Nguyen, M. Gulan, S. Olaru, and P. Rodriguez-Ayerbe, "Convex lifting: Theory and control applications," *IEEE Trans. Autom. Control*, vol. 63, no. 5, pp. 1243–1258, May 2018.

[12] M. Gulan, G. Takács, N. A. Nguyen, S. Olaru, P. Rodríguez-Ayerbe, and B. Rohal'-Ilkiv, "Efficient embedded model predictive vibration control via convex lifting," *IEEE Trans. Control Syst. Technol.*, vol. 27, no. 1, pp. 48–62, Jan. 2019.

[13] C. Wen, X. Ma, and B. E. Ydstie, "Analytical expression of explicit MPC solution via lattice piecewise-affine function," *Automatica*, vol. 45, no. 4, pp. 910–917, 2009.

[14] J. Xu, T. J. J. van den Boom, B. De Schutter, and S. Wang, "Irredundant lattice representations of continuous piecewise affine functions," *Automatica*, vol. 70, pp. 109–120, 2016.

[15] A. Bemporad and C. Filippi, "Suboptimal explicit MPC via approximate multiparametric quadratic programming," in *Proc. 40th IEEE Conf. Decis. Control*, 2001, pp. 4851–4856.

[16] A. Bemporad, A. Oliveri, T. Poggi, and M. Storace, "Ultra-fast stabilizing model predictive control via canonical piecewise affine approximations," *IEEE Trans. Autom. Control*, vol. 56, no. 12, pp. 2883–2897, Dec. 2011.

[17] G. Goebel and F. Allgöwer, "Semi-explicit MPC based on subspace clustering," *Automatica*, vol. 83, pp. 309–316, 2017.

[18] A. Chakrabarty, V. Dinh, M. J. Corless, A. E. Rundell, S. H. Żak, and G. T. Buzzard, "Support vector machine informed explicit nonlinear model predictive control using low-discrepancy sequences," *IEEE Trans. Autom. Control*, vol. 62, no. 1, pp. 135–148, Jan. 2017.

[19] B. Karg and S. Lucia, "Efficient representation and approximation of model predictive control laws via deep learning," *IEEE Trans. Cybern.*, vol. 50, no. 9, pp. 3866–3878, Sep. 2020.

[20] L. Csekő, M. Kvasnica, and B. Lantos, "Explicit MPC-based RBF neural network controller design with discrete-time actual Kalman filter for semiactive suspension," *IEEE Trans. Control Syst. Technol.*, vol. 23, no. 5, pp. 1736–1753, Sep. 2015.

[21] S. Summers, C. N. Jones, J. Lygeros, and M. Morari, "A multiresolution approximation method for fast explicit model predictive control," *IEEE Trans. Autom. Control*, vol. 56, no. 11, pp. 2530–2541, Nov. 2011.

[22] S. Gros and M. Zanon, "Data-driven economic NMPC using reinforcement learning," *IEEE Trans. Autom. Control*, vol. 65, no. 2, pp. 636–648, Feb. 2020.

[23] F. Scibilia, S. Olaru, and M. Hovd, "Approximate explicit linear MPC via Delaunay tessellation," in *Proc. 2009 IEEE Eur. Control Conf.*, 2009, pp. 2833–2838.

[24] A. Pavlov, I. Shames, and C. Manzie, "Minimax strategy in approximate model predictive control," *Automatica*, vol. 111, 2020, Art. no. 108649.
[25] J. Xu, "Lattice piecewise affine approximation of explicit linear model predictive control," in *Proc. 60th IEEE Conf. Decis. Control*, 2021, pp. 2545–2550.
[26] L. O. Chua and A. C. Deng, "Canonical piecewise-linear representation," *IEEE Trans. Circuits Syst.*, vol. 35, no. 1, pp. 101–111, Jan. 1988.
[27] F. Borrelli. *Constrained Optimal Control of Linear and Hybrid Systems*, vol. 290. New York, NY, USA: Springer, 2003.
[28] M. Herceg, C. N. Jones, M. Kvasnica, and M. Morari, "Enumeration-based approach to solving parametric linear complementarity problems," *Automatica*, vol. 62, pp. 243–248, 2015.
[29] M. Herceg, M. Kvasnica, C. N. Jones, and M. Morari, "Multi-parametric toolbox 3.0," in *Proc. Eur. Control Conf.*, 2013, pp. 502–510. [Online]. Available: http://control.ee.ethz.ch/mpt
[30] S. Wang and X. Sun, "Generalization of hinging hyperplanes," *IEEE Trans. Inf. Theory*, vol. 12, no. 51, pp. 4425–4431, Dec. 2005.
[31] T. A. Johansen and A. Grancharova, "Approximate explicit constrained linear model predictive control via orthogonal search tree," *IEEE Trans. Autom. Control*, vol. 48, no. 5, pp. 810–815, May 2003.
[32] J. M. Tarela, J. M. Perez, and V. Aleixandre, "Minimization of lattice polynomials on piecewise linear functions (Part I)," *Math. Comput. Simul.*, vol. 17, no. 2, pp. 79–85, 1975.
[33] S. Wang and K. S. Narendra, "Nonlinear system identification with lattice piecewise-linear functions," in *Proc. 2002 Amer. Control Conf.*, 2002, vol. 1, pp. 388–393.
[34] W. Hoeffding, "Probability inequalities for sums of bounded random variables," in *The Collected Works of Wassily Hoeffding*. Berlin, Germany: Springer, 1994, pp. 409–426.
[35] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "qpOASES: A parametric active-set algorithm for quadratic programming," *Math. Program. Comput.*, vol. 6, no. 4, pp. 327–363, 2014.
[36] Y. Ye and E. TSE, "An extension of Karmarkar's projective algorithm for convex quadratic programming," *Math. Program.*, vol. 44, no. 1, pp. 157–159, 1989.

Jun Xu (Senior Member, IEEE) received the B.S. degree in control science and engineering from the Harbin Institute of Technology, Harbin, China, in 2005, and the Ph.D. degree in control science and engineering from Tsinghua University, Beijing, China, in 2010.

She is currently an Associate Professor with the School of Mechanical Engineering and Automation, Harbin Institute of Technology, Shenzhen, China. Her research interests include piecewise linear functions and their applications in machine learning, nonlinear system identification, and control.

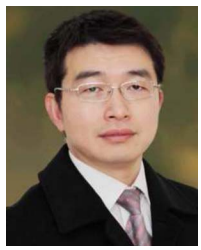
Yunjiang Lou (Senior Member, IEEE) received the B.S. and M.E. degrees in automation from the University of Science and Technology of China, Hefei, China, in 1997 and 2000, respectively, and the Ph.D. degree in electrical and electronic engineering from the Hong Kong University of Science and Technology, Hong Kong, in 2006.

He is currently with the State Key Laboratory of Robotics and Systems and the Shenzhen Key Laboratory for Advanced Motion Control and Modern Automation Equipments, School of Mechatronics Engineering and Automation, Harbin Institute of Technology, Shenzhen, China. His research interests include motion control, mechanism design, compliant actuators, and industrial robots.

Bart De Schutter (Fellow, IEEE) received the Ph.D. (summa cum laude) degree in applied sciences from KU Leuven, Leuven, Belgium, in 1996.

He is currently a Full Professor and Head of Department with the Delft Center for Systems and Control, Delft University of Technology, Delft, The Netherlands. His research interests include multilevel and multiagent control, model predictive control, learning-based control, and control of hybrid systems, with applications in intelligent transportation systems and smart energy systems.

Dr. Schutter is a Senior Editor of IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS and an Associate Editor for IEEE TRANSACTIONS ON AUTOMATIC CONTROL.



Zhenhua Xiong (Member, IEEE) received the B.E. and M.E. degrees in aircraft design and vibration engineering from the Department of Aircraft Design, Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 1995 and 1998, respectively, and the Ph.D. degree in mechatronics from the Department of Electrical and Electronics Engineering, Hong Kong University of Science and Technology, Hong Kong, in 2002.

He is currently a Professor with the State Key Laboratory of Mechanical System and Vibration, School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai, China. His research interests include servo motion control and intelligent manufacturing.