Convex Liftings: Theory and Control Applications

Ngoc Anh Nguyen, Martin Gulan, Sorin Olaru, Pedro Rodriguez-Ayerbe

Convex Liftings: Theory and Control Applications

N. A. Nguyen[1], M. Gulan[2], S. Olaru[3], P. Rodriguez-Ayerbe[3]

***Abstract*—This paper presents the so-called *convex lifting* concept which will be proven to enable significant implementation benefits for the class of piecewise affine controllers. Accordingly, two different algorithms to construct a convex lifting for a given polyhedral/polytopic partition will be presented. These two algorithms rely on either the vertex or the halfspace representation of the related polyhedra. Also, we introduce an algorithm to refine a polyhedral partition, which does not admit a convex lifting, into a convexly liftable one. Furthermore, two different schemes will be put forward to considerably reduce both the memory footprint and the runtime complexity which play a key role in implementation of piecewise affine controllers. Finally, these results will be illustrated via numerical examples and a complexity analysis.**

***Index Terms*—Convex lifting, model predictive control, explicit solutions, parametric programming.**

I. Motivation

Explicit model predictive control (MPC) has received significant attention in control community due to its relevance for rather small-dimensional systems [10], [17], [36], [37], [41]. However, even if the controllers are explicitly obtained, there exist major problems in terms of implementation once the number of regions in the state-space partition becomes large. In particular, they require storing all the regions at the hardware level, making their implementation, namely on embedded computing platforms, difficult due to their limited memory storage and computational performance.

Various efficient implementation algorithms have been put forward [24], [25], however the requirement of substantial memory for storing the given partition is inevitable. Another contribution about efficient storage strategy has been presented in [7]. This proposal can avoid storing the state-space partition, however the point-location problem, determining which region the current state belongs to, becomes more demanding; see among the other point-location algorithms [8], [9], [42]. Therefore, it is necessary to investigate other implementation approaches for this class of controllers which can avoid storing the state-space partition and possibly to facilitate the point-location problem. This work presents a convex lifting concept which allows for efficient implementation of piecewise affine (PWA) controllers. We have recently learned that an independent study in [1] exploits the *linear machine* concept which is actually similar to the convex lifting notion exploited in this paper. It is worth noting that existence conditions for convex liftings are not available in this reference. Also, the treatment of convexly non-liftable partitions therein has not been investigated. In this paper, we present constructions based on two different polyhedra representations—the vertex and the halfspace one. Also, an algorithm to refine a convexly non-liftable partition into a convexly liftable one is introduced.

For ease of presentation, let us start with some special cases of parametric linear programming problem where the optimal cost function presents an interesting property. To illustrate these points, consider a linear MPC problem with respect to a linear cost function. Such a problem can easily be transformed into a parametric linear programming problem as follows:

$$u^*(x) = \arg\min_u C^T u \quad \text{subject to} \quad Du \leq W + Ex. \tag{1}$$

It has been shown through Theorems IV-3 and IV-4 in [16] that $C^T u^*(x)$ is a convex, continuous, PWA function defined over a polyhedral partition $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$, where $\mathcal{I}_N = \{1, 2, \ldots, N\}$. Let us denote the optimal cost function and optimal solution of (1) as follows:

$$\begin{aligned} C^T u^*(x) &= a_i^T x + b_i \quad \text{for} \quad x \in \mathcal{X}_i, \\ u^*(x) &= H_i x + G_i \quad \text{for} \quad x \in \mathcal{X}_i. \end{aligned} \tag{2}$$

Since $C^T u^*(x)$ is also a convex function, this optimal cost function can alternatively be written in the following form:

$$C^T u^*(x) = \max_{j\in\mathcal{I}_N} (a_j^T x + b_j). \tag{3}$$

Accordingly, as advocated in [7], if the optimal solution to the parametric linear programming problem (1) is *unique*, then implementation of the optimal control law $u^*(x)$ can be carried out according to Algorithm 1.

Algorithm 1 Efficient implementation of PWA controllers

1: Store (H_j, G_j) and (a_j, b_j)
2: At each sampling time, obtain the current state x
3: Find index $i \in \mathcal{I}_N$ such that:

$$a_i^T x + b_i = \max_{j\in\mathcal{I}_N} (a_j^T x + b_j).$$

4: Evaluate the controller $u^*(x) = H_i x + G_i$.
5: Return to step 2.

A significant advantage of this implementation is that it enables to avoid storing the state-space partition and facilitates the point-location problem. However, as emphasized above, this implementation only holds if the optimal solution to (1) is unique, because in this case for any pair of different regions $(\mathcal{X}_i, \mathcal{X}_j)$, their optimal cost function satisfies $(a_i, b_i) \neq$

[1]N. A. Nguyen is with Institute for Design and Control of Mechatronical Systems, Johannes Kepler University, Linz, Austria. Ngocanh.Nguyen.rs@gmail.com

[2]M. Gulan is with Institute of Automation, Measurement and Applied Informatics, Faculty of Mechanical Engineering, Slovak University of Technology, Bratislava, Slovakia. martin.gulan@stuba.sk

[3]S. Olaru, P. Rodriguez-Ayerbe are with Laboratory of Signals and Systems (L2S, UMR CNRS 8506), CentraleSupélec-CNRS-Université Paris Sud, Gif-sur-Yvette, France. Sorin.Olaru, Pedro.Rodriguez@centralsupelec.fr

(a_j, b_j). Note also that in the case the uniqueness of the optimal solution to (1) is fulfilled, the optimal cost function of the parametric linear programming problem (1) is, from a geometrical point of view, nothing other than a convex lifting associated with the state-space partition $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ (this will be formally proven later in Theorem II.7), since a convex lifting can equivalently be understood as a convex, continuous, piecewise affine function defined over a polyhedral partition such that any pair of different regions is lifted onto two distinct hyperplanes. Otherwise, in case the optimal solution to (1) is not unique or the state-space partition obtained from a linear MPC problem with respect to a quadratic cost function, this implementation is no longer applicable, even if the state-space partition admits a convex lifting. Motivated by this limitation, algorithms to verify the existence of a convex lifting and to construct it for a given partition are required. It is worth recalling that an algorithm to construct a convex lifting based on the vertex representation has recently been presented in [30], [32]. However, efficiency of this algorithm is limited to rather small-dimensional partitions, since it requires vertex enumeration. To overcome this issue, in this paper we complete the convex lifting methodology with an algorithm relying on the halfspace representation. Accordingly, the feasibility of the formulated optimization problem is shown to represent another necessary and sufficient condition for the existence of a convex lifting. This paper further addresses the convexly non-liftable partitions by introducing an algorithm to refine a given convexly non-liftable partition into a convexly liftable one. Moreover, applications of these results in implementation of PWA controllers are put forward. Both numerical example and complexity analysis show promising results of the proposed implementations in comparison with some existing methods. This allows PWA control laws to be implemented even on embedded hardware with low computation performance and available memory.

II. Preliminaries

$\mathbb{R}, \mathbb{R}_+, \mathbb{N}_{>0}$ denote the field of real numbers, the set of non-negative real numbers and the positive integer set, respectively.

Given an arbitrary set $\mathcal{S}$, $\text{conv}(\mathcal{S})$ denotes the convex hull of $\mathcal{S}$; $\text{aff}(\mathcal{S})$ denotes the affine hull of $\mathcal{S}$. Also, $\dim(\mathcal{S})$ stands for the dimension of $\text{aff}(\mathcal{S})$. If $\mathcal{S}$ is a full-dimensional set, then $\text{int}(\mathcal{S})$ denotes its interior. Given a set $\mathcal{S} \subseteq \mathbb{R}^d$ and a subspace $\mathbb{S}$ of $\mathbb{R}^d$, then $\text{Proj}_{\mathbb{S}}\mathcal{S}$ denotes the orthogonal projection of $\mathcal{S}$ onto the space $\mathbb{S}$. We use $|\mathcal{S}|$ to denote the cardinality of the set $\mathcal{S}$. If Ω denotes a polyhedral partition, then $|\Omega|$ denotes the number of its regions.

A polyhedron is defined as the intersection of finitely many closed halfspaces. A polytope is defined as a bounded polyhedron. Given a polyhedron $\mathcal{S}$, we use $\mathcal{V}(\mathcal{S})$ to denote the set of its vertices and $\mathcal{R}(\mathcal{S})$ denotes the set of its extreme rays. Further, if $\mathcal{S} \subseteq \mathbb{R}^d$ is a full-dimensional polyhedron, a face of $\mathcal{S}$ is the intersection of $\mathcal{S}$ and one of its supporting hyperplanes. $k-$face represents a face of dimension k. A $0-$face is called a vertex, a $1-$face is called an edge, a $(d-1)-$face is called a facet. Also, $\mathcal{F}(\mathcal{S})$ denotes the set of all facets of the polyhedron $\mathcal{S}$. Given two sets S_1, S_2, we use $S_1 \backslash S_2$ to denote the following set: $S_1 \backslash S_2 := \{x : x \in S_1,\ x \notin S_2\}$.

Given a matrix $G \in \mathbb{R}^{m\times n}$, we denote $\text{size}(G, 1) = m$ and $\text{size}(G, 2) = n$. Also, $G(i, \cdot)$ denotes the i-th row of matrix G, while $G(i, j : k)$ denotes the i-th row of G, truncated from the j-th column to the k-th column, for $1 \le i \le m$ and $1 \le j \le k \le n$. If $j = k$, then $G(i, j)$ denotes the element of the i-th row and the j-th column. Let us recall also some useful definitions.

Definition II.1 A collection of $N \in \mathbb{N}_{>0}$ full-dimensional polyhedra $\mathcal{X}_i \subset \mathbb{R}^d$, denoted by $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$, is called a *polyhedral partition of a polyhedron* $\mathcal{X} \subseteq \mathbb{R}^d$ if:

1) $\mathcal{X} = \bigcup_{i\in\mathcal{I}_N} \mathcal{X}_i$,
2) $\text{int}(\mathcal{X}_i) \bigcap \text{int}(\mathcal{X}_j) = \emptyset$ with $i \neq j,\ (i, j) \in \mathcal{I}_N^2$,

$(\mathcal{X}_i, \mathcal{X}_j)$ are called neighboring or adjacent if $(i, j) \in \mathcal{I}_N^2$, $i \neq j$ and $\dim(\mathcal{X}_i \cap \mathcal{X}_j) = d - 1$. Also, if $\mathcal{X}$ is a polytope then $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ is called a *polytopic partition*.

In the case $\mathcal{X}$ is not a polyhedron, $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ is still called a polyhedral/polytopic partition but of a *nonconvex polyhedral set.*

The definition of a *cell complex* was presented by Grünbaum in [18]. For simplicity, a *cell complex* should be hereafter understood as a polyhedral partition whose face-to-face property is fulfilled, i.e., the intersection of any pair of regions is either empty or a common face. Also, if $\mathcal{X}$ is a polyhedron, then a cell complex of $\mathcal{X}$ is understood as a polyhedral partition whose facet-to-facet property is satisfied, meaning any pair of neighboring regions share a common facet. For illustration, the polytopic partition in Fig.1 is a cell complex.

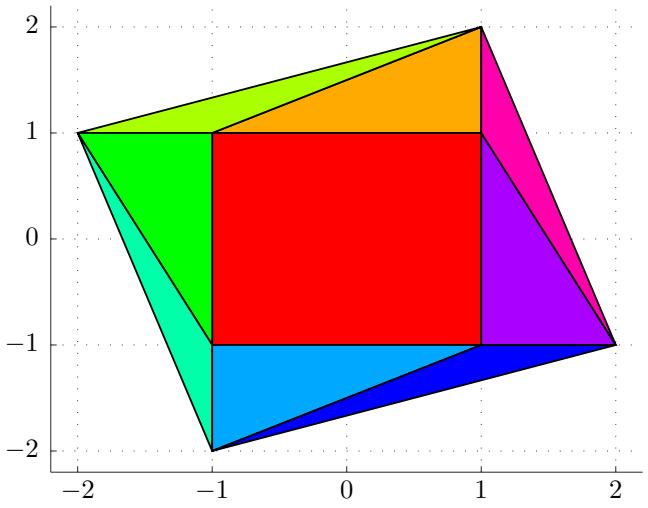

Fig. 1: A cell complex of a polytope in $\mathbb{R}^2$.

Definition II.2 For a given polyhedral partition $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ of a polyhedron $\mathcal{X} \subseteq \mathbb{R}^d$, a *piecewise affine lifting* is described by a function $z : \mathcal{X} \to \mathbb{R}$ with:

$$z(x) = a_i^T x + b_i \quad \text{for any } x \in \mathcal{X}_i, \tag{4}$$

and $a_i \in \mathbb{R}^d$, $b_i \in \mathbb{R}$, $\forall i \in \mathcal{I}_N$.

Definition II.3 Given a polyhedral partition $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ of a polyhedron $\mathcal{X}\subseteq\mathbb{R}^d$, a piecewise affine lifting

$$z(x)=a_i^Tx+b_i \text{ for } x\in\mathcal{X}_i,$$

is called a *convex piecewise affine lifting* if the following conditions hold true:

- $z(x)$ is continuous over $\mathcal{X}$,
- for each $i\in\mathcal{I}_N$, $z(x)>a_j^Tx+b_j$ for all $x\in\mathcal{X}_i\backslash\mathcal{X}_j$ and all $j\neq i$, $j\in\mathcal{I}_N$.

The second condition in the above definition implies that $z(x)$ is a convex function defined over $\mathcal{X}$. Moreover, the strict inequalities ensure that any pair of neighboring regions is lifted onto two distinct hyperplanes. For ease of presentation, a slight abuse of notation is used henceforth: a *convex lifting* is understood as a convex piecewise affine lifting.

With respect to the above definition, if a polyhedral partition $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ admits a convex lifting, then it has to be a cell complex. This observation was stated via Proposition 2.1 in [30]. It is recalled in the sequel for completeness, the interested reader is referred to this reference for the proof.

Proposition II.4 *A polyhedral partition of a polyhedron, which admits a convex lifting, is a cell complex.*

According to Proposition II.4, a convex lifting is always defined over a cell complex. However, the cell complex characterization of $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ is a necessary condition for the existence of a convex lifting, but not a sufficient one.

Remark II.5 Note also that Proposition II.4 does not necessarily restrict $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ to polyhedral partition of a polyhedron. In other words, a polyhedral partition of a suitable set $\mathcal{X}\subseteq\mathbb{R}^d$, which admits a convex lifting, should also be a cell complex.

Definition II.6 A given cell complex $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ in $\mathbb{R}^d$ has an *affinely equivalent polyhedron* if there exists a polyhedron $\widetilde{\mathcal{X}}\subset\mathbb{R}^{d+1}$ such that for each $i\in\mathcal{I}_N$:

1) $\exists F_i\in\mathcal{F}(\widetilde{\mathcal{X}})$ satisfying: $\text{Proj}_{\mathbb{R}^d}F_i=\mathcal{X}_i$,
2) if $\underline{z}(x)=\min\limits_z z$ s.t. $\begin{bmatrix}x^T & z\end{bmatrix}^T\in\widetilde{\mathcal{X}}$, then $\begin{bmatrix}x\\ \underline{z}(x)\end{bmatrix}\in F_i$ for $x\in\mathcal{X}_i$.

An illustration can be found in Fig.2 where a cell complex in $\mathbb{R}$ consists of the multicolored segments along the horizontal axis. One of its affinely equivalent polyhedra in $\mathbb{R}^2$ is the pink shaded region. Moreover, the lower facets of this polytope are an illustration of the facets F_i appearing in Definition II.6. Note that given a polyhedron $\widetilde{\mathcal{X}}\subset\mathbb{R}^{d+1}$, if z denotes the last coordinate of $\widetilde{\mathcal{X}}$ such that $\begin{bmatrix}x^T & z\end{bmatrix}^T\in\widetilde{\mathcal{X}}$, then the optimal solution to the following parametric linear programming problem:

$$z^*(x)=\min_z z \text{ subject to } \begin{bmatrix}x^T & z\end{bmatrix}^T\in\widetilde{\mathcal{X}}. \tag{5}$$

is nothing other than a convex lifting for the cell complex associated with this optimal solution. This observation will be proven in the sequel.

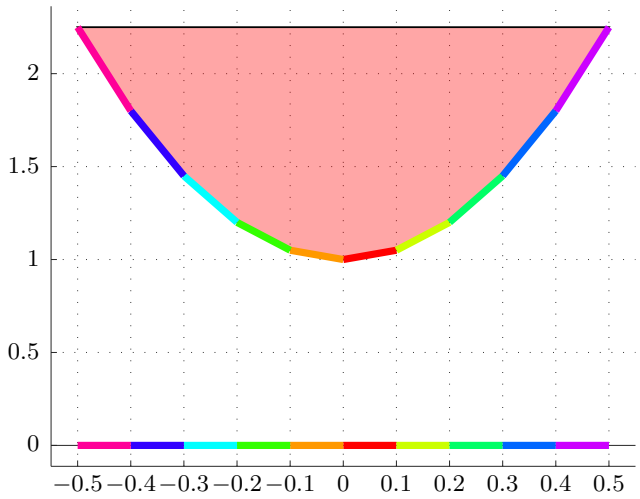

Fig. 2: An illustration of an affinely equivalent polyhedron.

First, consider the parametric linear programming problem (1) and its optimal solution (2). We will prove that if the optimal solution to (1) is unique, then the optimal cost function of (1) is a convex lifting for the polyhedral partition associated with the optimal solution (2).

Theorem II.7 *If the optimal solution to the parametric linear programming problem* (1) *is unique, then the optimal cost function $C^Tu^*(x)$ is a convex lifting for the polyhedral partition associated with $u^*(x)$.*

Proof: According to Theorems IV-3 and IV-4 in [16], $C^Tu^*(x)$ is a convex, continuous and piecewise affine function defined over the polyhedral partition $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$, associated with the optimal solution $u^*(x)$. Accordingly, to prove $C^Tu^*(x)$ to be a convex lifting for $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$, it suffices to show that for any pair of different regions $(\mathcal{X}_i,\mathcal{X}_j)$,

$$(C^TH_i,C^TG_i)\neq(C^TH_j,C^TG_j).$$

Suppose the converse situation happens, i.e., there exist two different regions $(\mathcal{X}_i,\mathcal{X}_j)$, $i\neq j$, $(i,j)\in\mathcal{I}_N^2$ such that $(C^TH_i,C^TG_i)=(C^TH_j,C^TG_j)$. We prove that both cases $(H_i,G_i)=(H_j,G_j)$ and $(H_i,G_i)\neq(H_j,G_j)$ never happen.

If $(H_i,G_i)=(H_j,G_j)$, we denote the set of constraints active at $u^*(x)=H_ix+G_i$ for $x\in\mathcal{X}_i$ as below

$$D^{[i]}u\leq W^{[i]}+E^{[i]}x. \tag{6}$$

As $(H_i,G_i)=(H_j,G_j)$, we then obtain $D^{[i]}(H_jx+G_j)=W^{[i]}+E^{[i]}x$. Note that this end holds for all $x\in\mathcal{X}_i$ as a full-dimensional polyhedron, which thus yields

$$D^{[i]}H_j=E^{[i]},\ D^{[i]}G_j=W^{[i]}. \tag{7}$$

Note however that for $x\in\text{int}(\mathcal{X}_j)$, there exists at least one constraint in (6) which is inactive at $u^*(x)=H_jx+G_j$, due to $\mathcal{X}_j\neq\mathcal{X}_i$ and the uniqueness of the optimal solution. Without loss of generality, suppose this constraint is $D^{[i]}(1,\cdot)u\leq W^{[i]}(1)+E^{[i]}(1,\cdot)x$. Accordingly, it yields:

$$D^{[i]}(1,\cdot)(H_jx+G_j)<W^{[i]}(1)+E^{[i]}(1,\cdot)x. \tag{8}$$

Inclusions (7) and (8) are clearly contradictory. In other words, the case $(H_i, G_i) = (H_j, G_j)$ cannot happen.

Otherwise, if $(H_i, G_i) \neq (H_j, G_j)$, consider $x_1 \in \text{int}(\mathcal{X}_i)$, $x_2 \in \mathcal{X}_j$ and a scalar $\alpha \in [0, 1]$. Due to the convexity of $C^T u^*(x)$, we can see that

$$\begin{aligned} &C^T u^*(\alpha x_1 + (1-\alpha)x_2) \\ &\quad \leq \alpha C^T(H_i x_1 + G_i) + (1-\alpha)C^T(H_j x_2 + G_j). \end{aligned} \tag{9}$$

If we choose α close enough to 1 such that $\alpha x_1 + (1-\alpha)x_2 \in \mathcal{X}_i$, then

$$C^T u^*(\alpha x_1 + (1-\alpha)x_2) = C^T(H_i(\alpha x_1 + (1-\alpha)x_2) + G_i). \tag{10}$$

Note also that according to the assumption $(C^T H_i, C^T G_i) = (C^T H_j, C^T G_j)$, it follows that

$$\begin{aligned} &\alpha C^T(H_i x_1 + G_i) + (1-\alpha)C^T(H_j x_2 + G_j) \\ &= C^T H_i(\alpha x_1 + (1-\alpha)x_2) + C^T G_i. \end{aligned} \tag{11}$$

Also, since $H_i x_1 + G_i, H_j x_2 + G_j$ satisfy the constraint set in (1), so does $\alpha(H_i x_1 + G_i) + (1-\alpha)(H_j x_2 + G_j)$. According to (9), (10), (11), $\alpha(H_i x_1 + G_i) + (1-\alpha)(H_j x_2 + G_j)$ is also an optimal solution to (1). Due to the uniqueness of the optimal solution to (1), we obtain the following:

$$H_i(\alpha x_1 + (1-\alpha)x_2) + G_i = \alpha(H_i x_1 + G_i) + (1-\alpha)(H_j x_2 + G_j),$$

leading to

$$H_i x_2 + G_i = H_j x_2 + G_j. \tag{12}$$

It is worth emphasizing that (12) holds true for all $x_2 \in \mathcal{X}_j$. Since $(H_i, G_i) \neq (H_j, G_j)$, the set of $x \in \mathbb{R}^d$ satisfying $H_i x + G_i = H_j x + G_j$ represents a polyhedron of dimension lower than d, while $\mathcal{X}_j$ is a full-dimensional polyhedron in $\mathbb{R}^d$. This is clearly contradictory. Therefore, the initial hypothesis is not true. In other words, if the optimal solution to (1) is unique, the optimal cost function $C^T u^*(x)$ describes a convex lifting for the associated polyhedral partition $\{\mathcal{X}_i\}_{i \in \mathcal{I}_N}$. ■

Now, let us come back to prove that the optimal solution to (5) stands for a convex lifting of the associated polyhedral partition.

Lemma II.8 *Given the parametric linear programming problem* (5)*, if $\{\mathcal{X}_i\}_{i \in \mathcal{I}_N}$ denotes the polyhedral partition associated with $z^*(x)$, then $z^*(x)$ is a convex lifting for $\{\mathcal{X}_i\}_{i \in \mathcal{I}_N}$.*

Proof: First, we will prove that the optimal solution to (5) is unique. Indeed, suppose there exist two optimal solutions to (5), denoted by $z_1^*(x)$ and $z_2^*(x)$, respectively. Without loss of generality, suppose $z_1^*(x), z_2^*(x)$ are defined on the same polyhedral partition $\{\mathcal{X}_i\}_{i \in \mathcal{I}_N}$. Consider a region $\mathcal{X}_i$ in this polyhedral partition and denote these optimal solutions over $\mathcal{X}_i$ as follows:

$$\begin{aligned} z_1^*(x) &= (a_i^{(1)})^T x + b_i^{(1)} \quad \text{for} \quad x \in \mathcal{X}_i, \\ z_2^*(x) &= (a_i^{(2)})^T x + b_i^{(2)} \quad \text{for} \quad x \in \mathcal{X}_i. \end{aligned}$$

Since $z_1^*(x), z_2^*(x)$ represent optimal cost function of (5), we thus obtain:

$$(a_i^{(1)})^T x + b_i^{(1)} = (a_i^{(2)})^T x + b_i^{(2)}. \tag{13}$$

Note that (13) holds for all $x \in \mathcal{X}_i$, as a full-dimensional polyhedron. Accordingly, this holds only if $(a_i^{(1)}, b_i^{(1)}) = (a_i^{(2)}, b_i^{(2)})$. In other words, the optimal solution to (5) is unique. According to Theorem II.7, the optimal cost function of (5) represents a convex lifting of $\{\mathcal{X}_i\}_{i \in \mathcal{I}_N}$. ■

III. Constructions of convex liftings

A. Existing results on convex liftings

The definition of a convex lifting has been presented earlier. In control theory, so far, convex liftings have been used to solve the inverse parametric linear/quadratic programming problem [28]–[35]. Many necessary and sufficient conditions for the existence of convex liftings for cell complexes were investigated in different studies [3]–[5], [13], [14], [27], [38], [40]. It is shown in [38] that there exists a convex lifting for a cell complex in $\mathbb{R}^d$ if and only if one of the following holds:

- it admits a strictly positive $d-$stress;
- it is an additively weighted Dirichlet-Voronoi diagram;
- it is an additively weighted Delaunay decomposition.

The interested reader is referred to [28] for further details of the above notation and to [3]–[5], [38] for other related results. Note that the above results cover the general class of cell complexes in $\mathbb{R}^d$. Unfortunately, despite the mathematical completeness of the existing results, the verification of these conditions is in general expensive. Furthermore, they do not provide any hint for the construction of a convex lifting. On the other hand, control applications require specific algorithms to verify the convex liftability of the cell complexes and construct their convex liftings if they exist. These elements are detailed in the following subsections. We remark that the construction of convex liftings for some special cases, e.g., Voronoi diagrams and Delaunay triangulations and their recognition were already investigated in [5], [15], [21].

B. Construction of convex liftings based on the vertex representation

The main objective of this subsection is to present an algorithm for the construction of a convex lifting for a given cell complex via linear/quadratic programming. Given a cell complex $\{\mathcal{X}_i\}_{i \in \mathcal{I}_N}$ of a polytope $\mathcal{X} \subset \mathbb{R}^d$, let a convex lifting $z(x)$ of $\{\mathcal{X}_i\}_{i \in \mathcal{I}_N}$ be denoted by $z(x) = a_i^T x + b_i$ for $x \in \mathcal{X}_i$, one needs to determine (a_i, b_i) for all $i \in \mathcal{I}_N$. The construction of $z(x)$ based on the vertex representation is presented in [30] and is recalled in Algorithm 2 for completeness. We remark that this construction is limited to polytopic partitions since it hinges on suitable constraints imposed at the vertices of this partition. Extension of this construction to polyhedral partitions of unbounded polyhedra can be found in [32].

Note that the cost function in (16) is chosen so as to avoid the unboundedness of optimal solution. Other choices are possible as long as the boundedness of optimal solution is guaranteed. Also, as seen in (15), the strict inequality in the second condition of Definition II.3 can easily be transformed into inequality constraints of an optimization problem by adding a positive constant c on the right-hand side of (15), thus $>$ can be replaced with $\geq$.

Algorithm 2 Construction of a convex lifting for a given cell complex $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ of a polytope $\mathcal{X}\subset\mathbb{R}^d$.

Input: $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ and a given constant $c>0$.
Output: $(a_i,\, b_i)$, $\forall i\in\mathcal{I}_N$.

1: Register all pairs of neighboring regions in $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$.
2: For each pair of neighboring regions $(\mathcal{X}_i,\, \mathcal{X}_j)$,
- add continuity conditions $\forall v\in\mathcal{V}(\mathcal{X}_i\cap\mathcal{X}_j)$:

$$a_i^T v+b_i=a_j^T v+b_j; \tag{14}$$

- add convexity conditions $\forall v\in\mathcal{V}(\mathcal{X}_i)\backslash\mathcal{V}(\mathcal{X}_j)$:

$$a_i^T v+b_i\geq a_j^T v+b_j+c. \tag{15}$$

3: Solve the following convex optimization problem by minimizing a chosen cost function, e.g.,

$$\min_{a_i,\,b_i}\ \sum_{i=1}^{N}(a_i^T a_i+b_i^T b_i)\ \text{ subject to }\ (14),(15). \tag{16}$$

Now, we will step by step prove that the feasibility of the optimization problem (16) serves as another necessary and sufficient condition for the convex liftability of the given polytopic partition of a polytope.

Proposition III.1 *If problem* (16) *is feasible, then function*

$$z(x)=a_i^T x+b_i\ \ \textit{for}\ \ x\in\mathcal{X}_i$$

is a convex lifting over the given cell complex $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$.

The interested reader is referred to Theorem 4.1 in [30] for the proof of Proposition III.1. We remark that this proposition only provides a sufficient condition for the existence of a convex lifting, while no result has been presented in [30] to clarify how the choice of the scalar constant c in Algorithm 2 affects the feasibility of the optimization problem (16). Theoretically, if the given cell complex is convexly liftable, then any positive value of c does not have any effect on the feasibility of the optimization problem (16).

Proposition III.2 *Given a cell complex* $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ *of a polyhedron* $\mathcal{X}\subseteq\mathbb{R}^d$, *if* $z(x)=a_i^T x+b_i$ *for* $x\in\mathcal{X}_i$ *is a convex lifting for this cell complex, then so is* $\tilde{z}(x)=(\alpha a_i)^T x+(\alpha b_i)+\beta$ *for* $x\in\mathcal{X}_i$, *for any* $\alpha>0,\beta\in\mathbb{R}$.

Proof: In fact, if $z(x)$ represents a convex lifting for the given cell complex $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$, then according to the definition of a convex lifting, for each pair of neighboring regions $(\mathcal{X}_i,\mathcal{X}_j)$, it follows that:

$$\begin{aligned} &a_i^T x+b_i=a_j^T x+b_j\ \ \text{for}\ \ x\in\mathcal{X}_i\cap\mathcal{X}_j\\ &a_i^T x+b_i>a_j^T x+b_j\ \ \text{for}\ \ x\in\mathcal{X}_i\backslash\mathcal{X}_j. \end{aligned} \tag{17}$$

Accordingly, for any $\alpha>0,\beta\in\mathbb{R}$, (17) amounts to:

$$\begin{aligned} &(\alpha a_i)^T x+\alpha b_i+\beta=(\alpha a_j)^T x+\alpha b_j+\beta\ \ \text{for}\ \ x\in\mathcal{X}_i\cap\mathcal{X}_j,\\ &(\alpha a_i)^T x+\alpha b_i+\beta>(\alpha a_j)^T x+\alpha b_j+\beta\ \ \text{for}\ \ x\in\mathcal{X}_i\backslash\mathcal{X}_j. \end{aligned} \tag{18}$$

Inclusion (18) means that $\tilde{z}(x)=(\alpha a_i)^T x+(\alpha b_i)+\beta$ for $x\in\mathcal{X}_i$ is also a convex lifting for the given cell complex for any $\alpha>0,\beta\in\mathbb{R}$. ■

We now prove that the feasibility of the optimization problem (16) serves as a necessary and sufficient condition for the convex liftability of the given polytopic partition $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ of polytopes.

Theorem III.3 *The given polytopic partition* $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ *of a polytope* $\mathcal{X}$, *is convexly liftable if and only if the optimization problem* (16) *is feasible for any constant* $c>0$.

Proof: $\longleftarrow$ This inclusion directly follows according to Proposition III.1.
$\longrightarrow$ If the given polytopic partition, denoted by $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$, is convexly liftable, then there exists a constant $\tilde{c}>0$ and a function $z(x)=a_i^T x+b_i$ for $x\in\mathcal{X}_i$ such that for any pair of neighboring regions $(\mathcal{X}_i,\mathcal{X}_j)$, the following holds:

$$\begin{aligned} &a_i^T v+b_i=a_j^T v+b_j\ \ \text{for}\ \ v\in\mathcal{V}(\mathcal{X}_i\cap\mathcal{X}_j)\\ &a_i^T v+b_i\geq a_j^T v+b_j+\tilde{c}\ \ \text{for}\ \ v\in\mathcal{V}(\mathcal{X}_i)\backslash\mathcal{V}(\mathcal{X}_j). \end{aligned} \tag{19}$$

According to Proposition III.2, if we choose $\alpha=c/\tilde{c}>0,\beta=0$, (19) is equivalent to:

$$\begin{aligned} &(\alpha a_i)^T v+(\alpha b_i)=(\alpha a_j)^T v+(\alpha b_j)\ \ \text{for}\ \ v\in\mathcal{V}(\mathcal{X}_i\cap\mathcal{X}_j)\\ &(\alpha a_i)^T v+(\alpha b_i)\geq(\alpha a_j)^T v+(\alpha b_j)+c\\ &\qquad\qquad\qquad\qquad\text{for}\ \ v\in\mathcal{V}(\mathcal{X}_i)\backslash\mathcal{V}(\mathcal{X}_j). \end{aligned}$$

In other words, $(\alpha a_i,\alpha b_i)$ for all $i\in\mathcal{I}_N$ also make the constraint set (14) and (15) feasible. Therefore, the optimization problem (16) is feasible with any given constant $c>0$. ■

Remark III.4 Note that Theorem III.3 holds true not only for polytopic partitions of polytopes but also for cell complexes of nonconvex polyhedral sets in $\mathbb{R}^d$.

Remark III.5 According to Proposition II.4, if a polyhedral partition is convexly liftable, then it should be a cell complex. Therefore, the optimization problem (16) is infeasible for the polytopic partitions of polytopes whose facet-to-facet property is not fulfilled.

To illustrate Algorithm 2, a cell complex of a polytope is shown in Fig.3. One of its convex liftings is also presented therein. Further, Fig.4 depicts a cell complex of a nonconvex set which is the underlying partition. One of its convex liftings is also illustrated above.

C. Construction of convex liftings based on the halfspace representation

Recall that Algorithm 2 relies on the vertex representation of related polytopes. Note also that the pivoting algorithm by Avis and Fukuda in [6] can carry out the vertex enumeration in time $\mathcal{O}(ndv)$, where d denotes the dimension of the given polytope, v represents the number of vertices of this polytope and n denotes the number of facets of this polytope. However, the vertex enumeration is not necessary in many cases, particularly

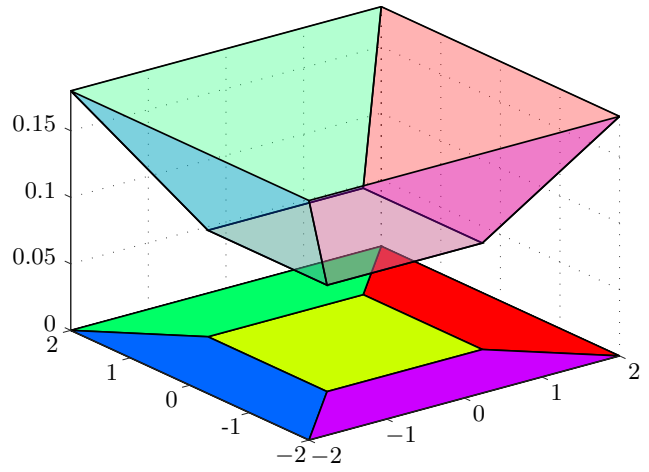

Fig. 3: A cell complex of a polytope and its convex lifting resulted from Algorithm 2.

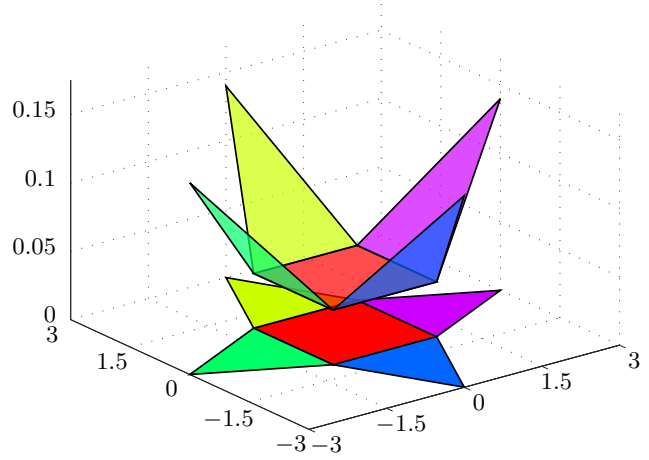

Fig. 4: A cell complex of a nonconvex polyhedral set and its convex lifting resulted from Algorithm 2.

in control theory where the implementation of state-space partition mostly relies on the halfspace representation. Moreover, the construction of convex liftings based on the vertex representation is limited to polytopic partitions, therefore, this construction for partitions of unbounded polyhedra may cause computational complications, see [32]. Motivated by these limitations, this subsection presents an approach to construct convex liftings based on the halfspace representation.

Given a convexly liftable cell complex $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ of a polyhedron $\mathcal{X}\subseteq\mathbb{R}^d$, as denoted in Definition II.3, we use

$$z(x)=a_i^Tx+b_i \quad \text{for} \quad x\in\mathcal{X}_i \tag{20}$$

to denote a convex lifting for $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$. Since a convex lifting $z(x)$ has to fulfill the continuity and convexity conditions, for any pair of neighboring regions $(\mathcal{X}_i, \mathcal{X}_j)$, the corresponding affine functions $a_i^Tx+b_i$ and $a_j^Tx+b_j$ have to satisfy:

$$a_i^Tx+b_i\geq a_j^Tx+b_j \quad \text{for all} \quad x\in\mathcal{X}_i, \tag{21a}$$
$$a_j^Tx+b_j\geq a_i^Tx+b_i \quad \text{for all} \quad x\in\mathcal{X}_j. \tag{21b}$$

It can be easily observed that according to (21a) and (21b)

$$a_i^Tx+b_i=a_j^Tx+b_j \quad \text{for all} \quad x\in\mathcal{X}_i\cap\mathcal{X}_j, \tag{22}$$

implies the continuity of $z(x)$ at any point $x\in\mathcal{X}_i\cap\mathcal{X}_j$. Furthermore, given the halfspace representation of region $\mathcal{X}_i$, i.e., $\mathcal{X}_i=\left\{x\in\mathbb{R}^d: R_ix\leq K_i\right\}$, (21a) holding for all $x\in\mathcal{X}_i$, leads to:

$$\begin{aligned}\mathcal{X}_i&=\left\{x\in\mathbb{R}^d: R_ix\leq K_i\right\}\\&\subseteq P=\left\{x\in\mathbb{R}^d:(a_j-a_i)^Tx\leq b_i-b_j\right\}.\end{aligned} \tag{23}$$

According to the extended Farkas lemma [39], (23) leads to the existence of a suitable vector λ_{ij} such that

$$\lambda_{ij}\geq 0,\ \lambda_{ij}R_i=(a_j-a_i)^T,\ \lambda_{ij}K_i\leq b_i-b_j. \tag{24}$$

Similarly, given the halfspace representation of region $\mathcal{X}_j$, i.e., $\mathcal{X}_j=\left\{x\in\mathbb{R}^d: R_jx\leq K_j\right\}$, (21b) leads to the existence of a suitable vector λ_{ji} such that

$$\lambda_{ji}\geq 0,\ \lambda_{ji}R_j=(a_i-a_j)^T,\ \lambda_{ji}K_j\leq b_j-b_i. \tag{25}$$

It should be emphasized that the constraints in (21) cannot guarantee that the affine functions corresponding to regions $\mathcal{X}_i$ and $\mathcal{X}_j$ are distinct, i.e., $(a_i, b_i)\neq(a_j, b_j)$. Therefore, in order to ensure this property of a convex lifting, one needs to impose additional constraints. A simple way to avoid nonlinear constraints is to require:

$$a_i^Tx_0+b_i\geq a_j^Tx_0+b_j+c, \tag{26}$$

for a given scalar constant $c>0$ and $x_0\in\mathrm{int}(\mathcal{X}_i)$. Constraint (26) is meaningful to guarantee $(a_i, b_i)\neq(a_j, b_j)$. In fact, if the converse situation happens, constraint (26) will be infeasible. Also, x_0 can be arbitrarily chosen as long as it lies in the interior of $\mathcal{X}_i$; the Chebyshev center is also a possible candidate. Recall that Chebyshev center of a polyhedron $\mathcal{X}$ is the center of the largest inscribed ball of $\mathcal{X}$. More precisely, finding Chebyshev center x_c of polyhedron $\mathcal{X}$ amounts to solving the following problem

$$\begin{aligned}&\max_{x_c,r}\ r \quad \text{s.t.}\\&x_c\in\mathcal{X},\ \left\{x\in\mathbb{R}^d:(x-x_c)^T(x-x_c)\leq r\right\}\subseteq\mathcal{X}.\end{aligned}$$

Note also that this problem can easily be transformed into a linear programming problem, see [12].

For completeness, a procedure to construct convex liftings based on the halfspace representation for a given convexly liftable cell complex is summarized in Algorithm 3.

Remark III.6 Note that Chebyshev center of a polyhedron may not always be unique or may lie at infinity. As emphasized above, other candidate of this point is possible as long as it lies in the interior of $\mathcal{X}_i$.

Remark III.7 We remark that for each pair of neighboring regions, two additional variables are added in the problem formulation, as shown in constraints (27), (28). Therefore, the number of additional variables, besides a_i, b_i, scales quadratically with the number of regions, since this number is bounded above by $N(N-1)$.

Algorithm 3 Construction of a convex lifting for a given convexly liftable cell complex $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ of a polyhedron $\mathcal{X}\subseteq\mathbb{R}^d$

Input: $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ of a polyhedron $\mathcal{X}\subseteq\mathbb{R}^d$, the halfspace representation of $\mathcal{X}_i=\left\{x\in\mathbb{R}^d : R_i x\leq K_i\right\}$ and a scalar constant $c>0$.

Output: gains a_i, b_i.

1: Find Chebyshev center for each region $\mathcal{X}_i$, denoted by x_i.

2: Register all pairs of neighboring regions in $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$.

3: For each pair of neighboring regions $(\mathcal{X}_i, \mathcal{X}_j)$, add the following constraints:

$$\lambda_{ij}\geq 0,\ \lambda_{ij}R_i=(a_j-a_i)^T,\ \lambda_{ij}K_i\leq b_i-b_j; \quad (27)$$
$$\lambda_{ji}\geq 0,\ \lambda_{ji}R_j=(a_i-a_j)^T,\ \lambda_{ji}K_j\leq b_j-b_i; \quad (28)$$
$$a_i^Tx_i+b_i\geq a_j^Tx_i+b_j+c. \quad (29)$$

4: Solve the following convex optimization problem by minimizing a chosen cost function, e.g.,

$$\min_{a_i,b_i,\lambda_{ij},\lambda_{ji}} \sum_{i=1}^{N}(a_i^Ta_i+b_i^Tb_i) \ \text{ s.t. } (27),(28),(29). \quad (30)$$

The following results present important formal properties of the construction in Algorithm 3.

Proposition III.8 *If the optimization problem* (30) *is feasible, then the function $z(x)=a_i^Tx+b_i$ for $x\in\mathcal{X}_i$ represents a convex lifting for cell complex $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$.*

Proof: If the optimization problem (30) is feasible, then the constraints (27), (28) and (29) are all feasible. According to the extended Farkas lemma [39], constraint (27) leads to:

$$a_i^Tx+b_i\geq a_j^Tx+b_j \ \text{ for all } \ x\in\mathcal{X}_i. \quad (31)$$

Similarly, it follows from constraint (28) that

$$a_j^Tx+b_j\geq a_i^Tx+b_i \ \text{ for all } \ x\in\mathcal{X}_j. \quad (32)$$

According to (31) and (32), the continuity of $z(x)$ at the common boundary of $\mathcal{X}_i$ and $\mathcal{X}_j$ is verified by

$$a_i^Tx+b_i=a_j^Tx+b_j \ \text{ for all } \ x\in\mathcal{X}_i\cap\mathcal{X}_j.$$

This leads to the following inclusions for the vertices and the extreme rays of $\mathcal{X}_i\cap\mathcal{X}_j$:

$$\begin{aligned} a_i^Tv+b_i=a_j^Tv+b_j \ \text{ for all } \ v\in\mathcal{V}(\mathcal{X}_i\cap\mathcal{X}_j),\\ a_i^Tr=a_j^Tr \ \text{ for all } \ r\in\mathcal{R}(\mathcal{X}_i\cap\mathcal{X}_j).\end{aligned} \quad (33)$$

Moreover, constraint (29) implies

$$a_i^Tx_i+b_i\geq a_j^Tx_i+b_j+c>a_j^Tx_i+b_j. \quad (34)$$

From (33) and (34), any point x, described in the following form:

$$x=\gamma x_i+\sum_{v\in\mathcal{V}(\mathcal{X}_i\cap\mathcal{X}_j)}\alpha(v)v+\sum_{r\in\mathcal{R}(\mathcal{X}_i\cap\mathcal{X}_j)}\mu(r)r$$

with $\alpha(v),\mu(r)\in\mathbb{R},\gamma+\sum_{v\in\mathcal{V}(\mathcal{X}_i\cap\mathcal{X}_j)}\alpha(v)=1$ satisfies:

$$a_i^Tx+b_i>a_j^Tx+b_j \ \text{ for all } \ \gamma>0, \quad (35a)$$
$$a_i^Tx+b_i=a_j^Tx+b_j \ \text{ for } \ \gamma=0. \quad (35b)$$

In other words, any point x, in the halfspace containing $\mathcal{X}_i$ but not in $\mathrm{aff}(\mathcal{X}_i\cap\mathcal{X}_j)$, satisfies (35a). Otherwise, any point $x\in\mathrm{aff}(\mathcal{X}_i\cap\mathcal{X}_j)$, satisfies (35b).

The same inclusion holds for the other pairs of neighboring regions, leading to the fact that:

$$a_i^Tx+b_i>a_j^Tx+b_j \ \text{ for all } \ x\in\mathcal{X}_i\backslash\mathcal{X}_j \ \text{ and } \ j\neq i. \quad (36)$$

Therefore, function $z(x)=a_i^Tx+b_i$ for $x\in\mathcal{X}_i$ represents a convex lifting for $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ according to Definition II.3. ■

Similar to Subsection III-B, any value of the given scalar c in Algorithm 3 does not affect the feasibility of the optimization problem (30) as long as $c>0$.

Theorem III.9 *The given cell complex $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ of a polyhedron $\mathcal{X}\subseteq\mathbb{R}^d$, is convexly liftable if and only if the optimization problem* (30) *is feasible for any constant $c>0$.*

Proof: $\longleftarrow$ This inclusion directly follows according to Proposition III.8.

$\longrightarrow$ If the given cell complex $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ is convexly liftable, then there exists a function $z(x)=a_i^Tx+b_i$ for $x\in\mathcal{X}_i$ such that for any pair of neighboring regions $(\mathcal{X}_i,\mathcal{X}_j)$, inclusions (31) and (32) hold. Accordingly, the extended Farkas lemma leads to the existence of two suitable vectors $\lambda_{ij},\lambda_{ji}$ such that:

$$\lambda_{ij}\geq 0,\ \lambda_{ij}R_i+(a_i-a_j)^T=0,\ \lambda_{ij}K_i\leq b_i-b_j, \quad (37a)$$
$$\lambda_{ji}\geq 0,\ \lambda_{ji}R_j+(a_j-a_i)^T=0,\ \lambda_{ji}K_j\leq b_j-b_i, \quad (37b)$$

where $\mathcal{X}_i,\mathcal{X}_j$ are again given as $\mathcal{X}_i=\left\{x\in\mathbb{R}^d : R_ix\leq K_i\right\}$, $\mathcal{X}_j=\left\{x\in\mathbb{R}^d : R_jx\leq K_j\right\}$.

Also, since $z(x)$ is a convex lifting for $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$, there exists a constant $c_{ij}>0$ for each pair of neighboring regions $(\mathcal{X}_i,\mathcal{X}_j)$ such that

$$a_i^Tx_i+b_i\geq a_j^Tx_i+b_j+c_{ij}, \quad (38)$$

where x_i represents Chebyshev center of $\mathcal{X}_i$. Let $\tilde{c}$ be the minimal value of c_{ij} for the pairs of neighboring regions $(\mathcal{X}_i,\mathcal{X}_j)$, i.e.,

$$\tilde{c}=\min_{(i,j)\in\mathcal{I}_N^2 \,|\, \dim(\mathcal{X}_i\cap\mathcal{X}_j)=d-1} c_{ij}.$$

Accordingly, for any pair of neighboring regions $(\mathcal{X}_i,\mathcal{X}_j)$, we obtain:

$$a_i^Tx_i+b_i\geq a_j^Tx_i+b_j+\tilde{c}.$$

By choosing $\delta=c/\tilde{c}>0$, it follows that

$$(\delta a_i)^Tx_i+(\delta b_i)\geq(\delta a_j)^Tx_i+(\delta b_j)+c. \quad (39)$$

According to (37a), (37b) and (39), it can be deduced that $(\delta a_i,\delta b_i)$ for all $i\in\mathcal{I}_N$ make the constraints (27), (28) and (29) feasible, since $\delta\lambda_{ij},\delta\lambda_{ji}\geq 0$. In other words, the optimization problem (30) is feasible with any given constant $c>0$. ■

Remark III.10 Note that according to Theorem III.9, the feasibility of the optimization problem (30) serves as another necessary and sufficient condition for the convex liftability of the polyhedral partitions of polyhedra.

Remark III.11 As proven in Proposition II.4, a polyhedral partition admitting a convex lifting should be a cell complex. Accordingly, for any polyhedral partition of polyhedron whose facet-to-facet property is not fulfilled, the optimization problem (30) is infeasible.

D. Convexly non-liftable partitions

This subsection addresses polyhedral partitions whose convex liftability is not fulfilled. This is usually the case in control theory, in particular for polyhedral partitions obtained from linear MPC problems with respect to quadratic cost functions. It is worth emphasizing that rearranging a given polyhedral partition is possible. However, any modification of the initial boundaries of the given polyhedral partition is not allowed due to the fact that it destroys the original structure of PWA controller. This may lead to the case where two different affine control laws are defined over the same region of state space. Therefore, the problem is formulated as follows: by preserving the internal boundaries, is it possible to refine a given polyhedral partition in order to recover the convex liftability property?

It will be proven that there exists at least one subdivision which can retrieve the convex liftability for a given polyhedral partition. The proof shows that the so-called *hyperplane arrangement* technique, defined as the decomposition of a space by a set of hyperplanes, can be used to perform this subdivision.

Theorem III.12 *Given a convexly non-liftable polyhedral partition $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ of a polyhedron $\mathcal{X}\subseteq\mathbb{R}^d$, there exists at least one subdivision, preserving the internal boundaries of this partition, such that the new cell complex is convexly liftable.*

The proof is referred to Appendix for reading ease.

Remark III.13 Note that Theorem III.12 states the existence of a suitable refinement, while the proof points to a specific technique for the refinement. In a broader perspective, for a given polyhedral partition which does not admit a convex lifting, there exist multiple practical solutions for suitable refinements into a convexly liftable cell complex, hyperplane arrangement is only one of them. An alternative, *fitting* planar cell complexes into Voronoi diagrams, can be found in [2].

Return to the hyperplane arrangement technique, an algorithm to carry out this decomposition is presented in Algorithm 4 for a given polyhedral partition.

To illustrate Algorithm 4, consider the cell complex in Fig.5, the result is depicted in Fig.6. Again, the convex liftability of this cell complex can be verified by the feasibility of problem (16) or (30).

Algorithm 4 An algorithm to carry out the hyperplane arrangement technique for a given polyhedral partition.

Input: Convexly non-liftable partition $\Omega=\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ in $\mathbb{R}^d$.
Output: Convexly liftable cell complex $\widetilde{\Omega}=\{\mathcal{Y}_j\}_{j\in\mathcal{I}_M}$.

$G=[\,]$
For $i=1:N$
  $\mathcal{X}_i=\{x: R_ix\le K_i\}$, $G=[G;R_i\ K_i]$
End
Remove redundant rows of matrix G.
For $i=1:\text{size}(G,1)$
  $\widetilde{\Omega}=\emptyset$
  For $j=1:|\Omega|$
    $\mathcal{X}_j=\{x: R_jx\le K_j\}$
    $\mathcal{Y}^{(1)}=\left\{x:\begin{bmatrix}R_j\\ G(i,1:d)\end{bmatrix}x\le\begin{bmatrix}K_j\\ G(i,d+1)\end{bmatrix}\right\}$
    $\mathcal{Y}^{(2)}=\left\{x:\begin{bmatrix}R_j\\ -G(i,1:d)\end{bmatrix}x\le\begin{bmatrix}K_j\\ -G(i,d+1)\end{bmatrix}\right\}$
    If $\dim(\mathcal{Y}^{(1)})=d$ & $\dim(\mathcal{Y}^{(2)})<d$ **then**
      $\widetilde{\Omega}\leftarrow\widetilde{\Omega}\cup\left\{\mathcal{Y}^{(1)}\right\}$
    Elseif $\dim(\mathcal{Y}^{(1)})<d$ & $\dim(\mathcal{Y}^{(2)})=d$ **then**
      $\widetilde{\Omega}\leftarrow\widetilde{\Omega}\cup\left\{\mathcal{Y}^{(2)}\right\}$
    Elseif $\dim(\mathcal{Y}^{(1)})=d$ & $\dim(\mathcal{Y}^{(2)})=d$ **then**
      $\widetilde{\Omega}\leftarrow\widetilde{\Omega}\cup\left\{\mathcal{Y}^{(1)},\mathcal{Y}^{(2)}\right\}$
    End
  End
  $\Omega\leftarrow\widetilde{\Omega}$
End

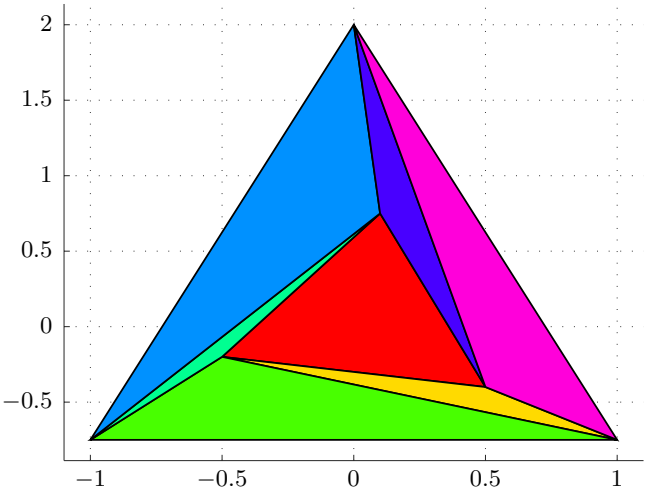

Fig. 5: A convexly non-liftable cell complex in $\mathbb{R}^2$.

IV. Applications of convex lifting in control

This section aims to employ the convex lifting concept to facilitate the implementation of PWA control laws. Note that earlier studies in this subject can be found in [19], [28]. In control theory, since the performance of physical systems is always limited, the control signal is usually bounded [25].

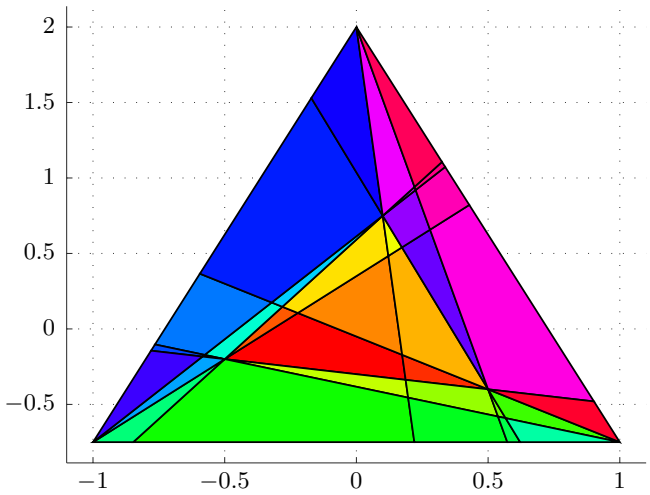

Fig. 6: Cell complex resulted from Algorithm 4.

Therefore, without loss of generality, the constraints on current control variable denoted by $u \in \mathbb{R}^{d_u}$, are assumed to be in the following form:

$$u_{\min} \leq u \leq u_{\max}.$$

For ease of presentation, we use $u^{(i)}$ to denote the $i-$th component of vector u. By an *unsaturated region*, we denote a region whose associated control law is not of componentwise saturation, i.e., $u^{(i)}_{\min} < u^{(i)} < u^{(i)}_{\max}$ for at least one $i \in \mathcal{I}_{d_u}$. Furthermore, a *saturated region* implies a region corresponding to a componentwise saturated control law, i.e., either $u^{(i)} = u^{(i)}_{\min}$ or $u^{(i)} = u^{(i)}_{\max}$ for all $i \in \mathcal{I}_{d_u}$. Accordingly, given a state-space partition, the *unsaturated partition* consists of the unsaturated regions. Such a partition may not be a partition of a polyhedron but of a nonconvex set. The developments of this section are motivated by two following observations:

- the complexity of state-space partitions is mainly due to the saturation [24], the boundaries between saturated regions of the same controller can thus be appropriately modified;
- in many practical MPC setups, the unsaturated partition is convexly liftable [19].

Note that the existence of unsaturated partition is the premise of works on complexity reduction in explicit MPC controllers, e.g., [24], [25]. For ease of presentation, the following assumptions are convenient for the next developments.

Assumption IV.1 The control input is a scalar variable, i.e., $\dim(u) = d_u = 1$.

Assumption IV.2 The unsaturated partition is a convexly liftable polytopic partition.

Assumption IV.3 The state space $\mathcal{X}$ is a polytope.

Assumption IV.4 The given PWA control law is continuous.

Assumption IV.1 is not restrictive, since the development presented in the sequel can easily be extended to the multivariable case. Note also that even if the unsaturated partition is not convexly liftable, one can use Algorithm 4 to split it into a convexly liftable cell complex. This is meaningful to avoid a complete hyperplane rearrangement of the original state-space partition. Therefore, Assumption IV.2 loses no generality of the proposed schemes. Also, Assumption IV.3 restricts our attention to polytopic partitions of the state space. This is not restrictive, since the construction can easily be extended to polyhedral partitions. Finally, we are exclusively interested in implementation of the continuous PWA controllers as presented in Assumption IV.4.

Given a PWA controller

$$u(x) = H_i x + G_i \quad \text{for} \quad x \in \mathcal{X}_i, \tag{40}$$

defined over a polytopic partition $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ of a polytope $\mathcal{X} \subset \mathbb{R}^d$ satisfying Assumptions IV.1, IV.2 and IV.4, let $\mathcal{I}^{\text{uns}} \subset \mathcal{I}_N$ denote the index set such that $\{\mathcal{X}_i\}_{i\in\mathcal{I}^{\text{uns}}}$ represents the unsaturated partition of $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ and $u(x)$. Also, we use $\ell^{\text{uns}}(x)$ to denote a convex lifting for $\{\mathcal{X}_i\}_{i\in\mathcal{I}^{\text{uns}}}$, i.e.,

$$\ell^{\text{uns}}(x) = (a_i^{\text{uns}})^T x + b_i^{\text{uns}} \quad \text{for} \quad x \in \mathcal{X}_i,\ i \in \mathcal{I}^{\text{uns}}. \tag{41}$$

In order to use Algorithm 1, we need to construct a convex lifting, denoted by $\ell(x)$, which is defined over the whole $\mathcal{X}$ and coincides with $\ell^{\text{uns}}(x)$ over $\{\mathcal{X}_i\}_{i\in\mathcal{I}^{\text{uns}}}$. To this goal, two different constructions will be presented in the sequel; the first one aims to rearrange the saturated regions to reduce the number of regions and also find a suitable convex lifting over the rearranged partition, while the second one incorporates the clipping concept with convex lifting.

A. Construction based on convex lifting for the vertices of the state space $\mathcal{X}$

The first construction aims to compute an appropriate height h^* corresponding to the vertices of $\mathcal{X}$. This height has to satisfy that the augmented vertices $\left[v^T\ \ell^{\text{uns}}(v)\right]^T$ for $v \in \bigcup_{i\in\mathcal{I}^{\text{uns}}} \mathcal{V}(\mathcal{X}_i)$ and $\left[v^T\ h^*\right]^T$ for $v \in \mathcal{V}(\mathcal{X})\backslash\bigcup_{i\in\mathcal{I}^{\text{uns}}} \mathcal{V}(\mathcal{X}_i)$ build a convex lifting $\ell(x)$ over $\mathcal{X}$ such that $\ell(x) = \ell^{\text{uns}}(x)$ for $x \in \bigcup_{i\in\mathcal{I}^{\text{uns}}} \mathcal{X}_i$ and $\ell(x) = h^*$ for $x \in \mathcal{V}(\mathcal{X})\backslash\bigcup_{i\in\mathcal{I}^{\text{uns}}} \mathcal{V}(\mathcal{X}_i)$. This construction is presented in Algorithm 5.

The following lemma represents the most important property of $\ell(x)$ resulted from Algorithm 5.

Lemma IV.5 *$\ell(x)$ obtained from Algorithm 5 satisfies:*

$$\ell(x) = \ell^{\text{uns}}(x) \text{ for all } x \in \bigcup_{i\in\mathcal{I}^{\text{uns}}} \mathcal{X}_i.$$

Proof: Consider any point $\left[x^T\ z\right]^T \in \Pi$, defined in (43). This point can be described as a convex combination of the points in Π_1, Π_2 as follows:

$$\begin{aligned}
&\alpha(v), \beta(v) \geq 0,\\
&\sum_{v\in\bigcup_{i\in\mathcal{I}^{\text{uns}}} \mathcal{V}(\mathcal{X}_i)} \alpha(v) + \sum_{v\in\mathcal{V}(\mathcal{X})\backslash\bigcup_{i\in\mathcal{I}^{\text{uns}}} \mathcal{V}(\mathcal{X}_i)} \beta(v) = 1,\\
&\left[x^T\ z\right]^T = \sum_{v\in\bigcup_{i\in\mathcal{I}^{\text{uns}}} \mathcal{V}(\mathcal{X}_i)} \alpha(v) \left[v^T\ \ell^{\text{uns}}(v)\right]^T\\
&\qquad + \sum_{v\in\mathcal{V}(\mathcal{X})\backslash\bigcup_{i\in\mathcal{I}^{\text{uns}}} \mathcal{V}(\mathcal{X}_i)} \beta(v) \left[v^T\ h^*\right]^T.
\end{aligned}$$

Algorithm 5 Construct a convex lifting over $\mathcal{X} \subset \mathbb{R}^d$, coincident with $\ell^{\rm uns}(x)$ over the unsaturated partition $\{\mathcal{X}_i\}_{i\in\mathcal{I}^{\rm uns}}$.

Input: $\{\mathcal{X}_i\}_{i\in\mathcal{I}^{\rm uns}}$, $\ell^{\rm uns}(x)$ defined in (41), $\mathcal{X}$ and a given constant $c > 0$.

Output: h^*, $\ell(x)$.

1: Solve the problem:

$$h^* = \min_h h \;\text{ s.t. }\; h \geq c + (a_i^{\rm uns})^T v + b_i^{\rm uns}, \quad \forall i \in \mathcal{I}^{\rm uns},\ \forall v \in \mathcal{V}(\mathcal{X})\backslash \bigcup_{i\in\mathcal{I}^{\rm uns}} \mathcal{V}(\mathcal{X}_i) \tag{42}$$

2: Construct the polytope

$$\begin{aligned} \Pi_1 &= \left\{ \begin{bmatrix} v \\ \ell^{\rm uns}(v) \end{bmatrix} : v \in \bigcup_{i\in\mathcal{I}^{\rm uns}} \mathcal{V}(\mathcal{X}_i) \right\} \subset \mathbb{R}^{d+1}, \\ \Pi_2 &= \left\{ \begin{bmatrix} v \\ h^* \end{bmatrix} : v \in \mathcal{V}(\mathcal{X})\backslash \bigcup_{i\in\mathcal{I}^{\rm uns}} \mathcal{V}(\mathcal{X}_i) \right\} \subset \mathbb{R}^{d+1}, \\ \Pi &= \text{conv}\,(\Pi_1 \cup \Pi_2) \end{aligned} \tag{43}$$

3: Solve the following parametric linear program:

$$\ell(x) = \arg\min_z z \text{ subject to } \begin{bmatrix} x^T & z \end{bmatrix}^T \in \Pi. \tag{44}$$

Denote also $\underline{z}(x) = \max\limits_{j\in\mathcal{I}^{\rm uns}} (a_j^{\rm uns})^T x + b_j^{\rm uns}$ for $x \in \mathcal{X}$. Clearly, $\underline{z}(x) = \ell^{\rm uns}(x)$ for $x \in \bigcup_{i\in\mathcal{I}^{\rm uns}} \mathcal{X}_i$ and is known to be a convex function over $\mathcal{X}$. According to (42), it follows that:

$$\begin{aligned} &\sum_{v\in\bigcup_{i\in\mathcal{I}^{\rm uns}} \mathcal{V}(\mathcal{X}_i)} \alpha(v)\ell^{\rm uns}(v) + \sum_{v\in\mathcal{V}(\mathcal{X})\backslash\bigcup_{i\in\mathcal{I}^{\rm uns}} \mathcal{V}(\mathcal{X}_i)} \beta(v)h^* \\ &\geq \sum_{v\in\bigcup_{i\in\mathcal{I}^{\rm uns}} \mathcal{V}(\mathcal{X}_i)} \alpha(v)\underline{z}(v) \\ &\qquad + \sum_{v\in\mathcal{V}(\mathcal{X})\backslash\bigcup_{i\in\mathcal{I}^{\rm uns}} \mathcal{V}(\mathcal{X}_i)} \beta(v)(c + \underline{z}(v)) \\ &\geq \underline{z}(x) + \sum_{v\in\mathcal{V}(\mathcal{X})\backslash\bigcup_{i\in\mathcal{I}^{\rm uns}} \mathcal{V}(\mathcal{X}_i)} \beta(v)c \geq \underline{z}(x). \end{aligned}$$

If $x \in \mathcal{X}_i$ for $i \in \mathcal{I}^{\rm uns}$, then the equality only happens when $\beta(v) = 0$ for $v \in \mathcal{V}(\mathcal{X})\backslash\bigcup_{i\in\mathcal{I}^{\rm uns}} \mathcal{V}(\mathcal{X}_i)$. In other words, when $x \in \bigcup_{i\in\mathcal{I}^{\rm uns}} \mathcal{X}_i$ the minimal cost function of (44) satisfies $\ell(x) = \underline{z}(x) = \ell^{\rm uns}(x)$. ■

Let $\{\mathcal{Y}_j\}_{j\in\mathcal{I}_M}$ denote the state-space partition associated with $\ell(x)$ obtained from Algorithm 5. The following corollary presents another property of such a function $\ell(x)$.

Corollary IV.6 *$\ell(x)$ obtained from Algorithm 5, represents a convex lifting for the polytopic partition $\{\mathcal{Y}_j\}_{j\in\mathcal{I}_M}$ of the state space $\mathcal{X}$.*

Proof: The proof follows as a direct consequence of Lemma II.8. ■

According to Lemma IV.5, for each region $\mathcal{X}_i$ of the unsaturated partition $\{\mathcal{X}_i\}_{i\in\mathcal{I}^{\rm uns}}$, there exists a region $\mathcal{Y}_j$ of $\{\mathcal{Y}_j\}_{j\in\mathcal{I}_M}$ such that $\mathcal{Y}_j = \mathcal{X}_i$. If $\mathcal{I}^{\max} \subset \mathcal{I}_N$ ($\mathcal{I}^{\min} \subset \mathcal{I}_N$) denotes the set of indices such that each region $\mathcal{X}_j$, $j \in \mathcal{I}^{\max}$ ($j \in \mathcal{I}^{\min}$) is associated with saturated controller $u(x) = u_{\max}$ ($u(x) = u_{\min}$) for $x \in \mathcal{X}_j$, then $\mathcal{I}_N = \mathcal{I}^{\rm uns} \cup \mathcal{I}^{\max} \cup \mathcal{I}^{\min}$. Define the following controller, denoted by $\widetilde{f}_{\rm pwa}(x)$, associated with $\{\mathcal{Y}_j\}_{j\in\mathcal{I}_M}$:

$$\widetilde{f}_{\rm pwa}(x) = \begin{cases} u(x) \text{ if } x \in \mathcal{Y}_j \text{ s.t. } \exists i \in \mathcal{I}^{\rm uns}, \mathcal{Y}_j = \mathcal{X}_i \\ u_{\max} \text{ if } x \in \mathcal{Y}_j \text{ s.t. } \mathcal{Y}_j \subset \bigcup\limits_{i\in\mathcal{I}^{\max}} \mathcal{X}_i \\ u_{\min} \text{ if } x \in \mathcal{Y}_j \text{ s.t. } \mathcal{Y}_j \subset \bigcup\limits_{i\in\mathcal{I}^{\min}} \mathcal{X}_i \end{cases}$$

Note that the newly obtained PWA control law $\widetilde{f}_{\rm pwa}(x)$ is equivalent to the given one $u(x)$ in the sense that $\widetilde{f}_{\rm pwa}(x) = u(x)$ for all $x \in \mathcal{X}$. Therefore, it suffices to implement $\widetilde{f}_{\rm pwa}(x)$ as in Algorithm 1.

Remark IV.7 In the case the given PWA control law $u(x)$ is of multiple inputs, then implementation of this controller according to the construction of convex liftings as in Algorithm 5 can be carried out componentwise. Roughly speaking, the implementation of $u(x)$ can be summarized as follows:

- construct a convex lifting $(\ell^{\rm uns})^{(i)}(x)$ for the unsaturated partition, denoted by $\{\mathcal{X}_j\}_{j\in\mathcal{I}^{(i)}}$, of the state-space partition $\{\mathcal{X}_j\}_{j\in\mathcal{I}_N}$ ($\mathcal{I}^{(i)} \subseteq \mathcal{I}_N$) associated with the i-th component $u^{(i)}(x)$ of the given PWA controller $u(x)$;
- construct an extended convex lifting, denoted by $\ell^{(i)}(x)$ defined over $\mathcal{X}$ for $(\ell^{\rm uns})^{(i)}(x)$ as in Algorithm 5;
- rearrange each component $u^{(i)}(x)$ of the given PWA controller $u(x)$ according to $\ell^{(i)}(x)$; denote this rearranged component by $\tilde{u}^{(i)}(x)$
- implement each rearranged component $\tilde{u}^{(i)}(x)$ as in Algorithm 1.

Note also that in this multiple-input case, the unsaturated partitions $\{\mathcal{X}_j\}_{j\in\mathcal{I}^{(i)}}$ of $\{\mathcal{X}_j\}_{j\in\mathcal{I}_N}$ associated with the components $u^{(i)}(x)$ of $u(x)$, may not be identical.

B. Construction based on convex lifting and clipping

Although the construction of $\ell(x)$ in Algorithm 5 shows an advancement in terms of efficient storage, the number of affine functions composing $\ell(x)$ may be still relatively large. We now present a more efficient construction that can considerably reduce the number of affine functions. This construction employs convex liftings and the concept of clipping presented in [24].

As mentioned in the proof of Lemma IV.5, we can choose such a convex lifting $\ell(x)$ as follows:

$$\ell(x) = \max_{j\in\mathcal{I}^{\rm uns}} (a_j^{\rm uns})^T x + b_j^{\rm uns} \;\text{ for }\; x \in \mathcal{X}. \tag{45}$$

Obviously, this construction ensures that $\ell(x) = \ell^{\rm uns}(x)$ for all $x \in \bigcup_{i\in\mathcal{I}^{\rm uns}} \mathcal{X}_i$. Let $\{\mathcal{Y}_i\}_{i\in\mathcal{I}_M}$ denote the polytopic partition of $\mathcal{X}$ associated with the convex lifting $\ell(x)$ defined in (45). For ease of presentation, denote $\ell(x)$ as follows:

$$\ell(x) = \widetilde{a}_i^T x + \widetilde{b}_i \;\text{ for }\; x \in \mathcal{Y}_i. \tag{46}$$

According to the new state-space partition $\{\mathcal{Y}_i\}_{i\in\mathcal{I}_M}$, a (non-equivalent) rearrangement of the given control law $u(x)$, denoted by $\widetilde{f}_{\rm pwa}(x)$, is put forward as follows:

$$\widetilde{f}_{\rm pwa}(x) = \widetilde{H}_i x + \widetilde{G}_i = H_j x + G_j \;\; \text{for} \;\; x \in \mathcal{Y}_i \;\; \text{such that} \;\; j \in \mathcal{I}^{\rm uns},\; \mathcal{X}_j \subseteq \mathcal{Y}_i. \tag{47}$$

Note that determining the new partition $\{\mathcal{Y}_i\}_{i\in\mathcal{I}_M}$ in (46) requires solving a linear parametric programming problem with the number of constraints equal to $|\mathcal{I}^{\rm uns}| = M$ and a 1-dimensional decision variable. Accordingly, the definition of $\widetilde{f}_{\rm pwa}(x)$ in (47) just involves in set comparisons available, e.g., in MPT 3.0 [22]. The following corollary represents a property of $\widetilde{f}_{\rm pwa}(x)$ constructed in (47).

Corollary IV.8 *If $u(x)$ defined in* (40) *is continuous, then $\widetilde{f}_{\rm pwa}(x)$ is continuous over $\bigcup_{i\in\mathcal{I}^{\rm uns}}\mathcal{X}_i$.*

Proof: It can be observed that $\widetilde{f}_{\rm pwa}(x) = u(x)$ over $\bigcup_{i\in\mathcal{I}^{\rm uns}}\mathcal{X}_i$. Obviously, the proof directly follows. ■

Note also that the continuity of $\widetilde{f}_{\rm pwa}(x)$ may not be guaranteed over $\mathcal{X}\backslash\bigcup_{i\in\mathcal{I}^{\rm uns}}\mathcal{X}_i$, as this property is not accounted for in the definition of $\widetilde{f}_{\rm pwa}(x)$ in (47). However, in implementation, $\widetilde{f}_{\rm pwa}(x)$ will be saturated over this region such that the given constraints are respected. The implementation is summarized in Algorithm 6.

Algorithm 6 Efficient implementation of PWA controllers based on convex lifting and clipping

1: Store $\ell(x)$ defined in (45), denoted as in (46) and the PWA controller $\widetilde{f}_{\rm pwa}(x)$ defined as in (47).
2: At each sampling time, obtain the current state x.
3: Find index $i \in \mathcal{I}_M$ such that:
$$\widetilde{a}_i^T x + \widetilde{b}_i = \max_{j\in\mathcal{I}_M} (\widetilde{a}_j^T x + \widetilde{b}_j).$$
4: Evaluate the control law
$$u^*(x) = \begin{cases} \widetilde{H}_i x + \widetilde{G}_i & \text{if} \;\; u_{\min} \le \widetilde{H}_i x + \widetilde{G}_i \le u_{\max} \\ u_{\max} & \text{if} \;\; \widetilde{H}_i x + \widetilde{G}_i > u_{\max} \\ u_{\min} & \text{if} \;\; \widetilde{H}_i x + \widetilde{G}_i < u_{\min}. \end{cases}$$
5: Return to step 2.

Note that by the saturation in Step 4, $u^*(x)$ is equivalent to the given PWA controller, i.e., $u^*(x) = u(x)$, see [19], [24].

Remark IV.9 Algorithm 6 can be easily extended to the multiple-input case. More precisely, unlike the implementation in Subsection IV-A, this implementation, based on convex lifting and clipping, only requires the construction of a single convex lifting for the unsaturated partition of $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$ associated with $u(x)$ instead of a convex lifting for each component of $u(x)$. Accordingly, the given multiple-input $u(x)$ can be implemented as in Algorithm 6 where Step 4 has to be modified to carry out the componentwise saturation. Namely, if d_u denotes the dimension of $u(x)$, then

$$u^*(x) = \left[(u^*)^{(1)}(x) \dots (u^*)^{(d_u)}(x)\right]^T,$$

where $(u^*)^{(j)}(x)$ for $j \in \mathcal{I}_{d_u}$ are defined as follows:

$$(u^*)^{(j)}(x) = \begin{cases} \widetilde{H}_i^{(j)} x + \widetilde{G}_i^{(j)} & \text{if} \;\; u_{\min}^{(j)} \le \widetilde{H}_i^{(j)} x + \widetilde{G}_i^{(j)} \le u_{\max}^{(j)} \\ u_{\max}^{(j)} & \text{if} \;\; \widetilde{H}_i^{(j)} x + \widetilde{G}_i^{(j)} > u_{\max}^{(j)} \\ u_{\min}^{(j)} & \text{if} \;\; \widetilde{H}_i^{(j)} x + \widetilde{G}_i^{(j)} < u_{\min}^{(j)}, \end{cases}$$

and $\widetilde{H}_i^{(j)}, \widetilde{G}_i^{(j)}, u_{\max}^{(j)}, u_{\min}^{(j)}$ again denote the j-th row of matrices $\widetilde{H}_i, \widetilde{G}_i, u_{\max}, u_{\min}$.

C. Complexity analysis

In the following, we assess memory and runtime complexity of the proposed approaches in the context of implementation of PWA controllers. As a reference, the total memory consumed by the original PWA control law together with its underlying partition, $\{\mathcal{X}_i\}_{i\in\mathcal{I}_N}$, is $\sum_{i=1}^N (c_i + d_u)(d+1)$ real numbers, where c_i denotes the number of halfspaces defining the i-th region. On the other hand, storing the simpler controller obtained via Section IV-A and its associated control law requires $M(1+d_u)(d+1)$ real numbers, where M is the number of affine terms of the convex lifting, resulted from Algorithm 5. Finally, the memory footprint of the implementation proposed in Section IV-B amounts to $|\mathcal{I}^{\rm uns}|(1+d_u)(d+1) + 2d_u$ real numbers, where the second term denotes a negligible memory needed to encode the clipping function in Algorithm 6.

In addition, we quantify the necessary on-line computational effort. Specifically, the standard implementation of the original PWA control law via sequential search consists of the point location problem, i.e., finding index i of the region $\mathcal{X}_i$ that contains x, and evaluation of the corresponding control law. In the worst case, this amounts to $\sum_{i=1}^N c_i(2d+1) + 2d_u d$ floating point operations (FLOPs). Note that the proposed implementations based on convex lifting may perform this task in a very efficient way, in particular the one described by Algorithm 6 exploiting clipping, without the need to carry out expensive point location. In total, it requires a constant number of $2|\mathcal{I}^{\rm uns}|d + 2d_u d + 2d_u$ FLOPs, which is a significant reduction in runtime complexity, even if $|\mathcal{I}^{\rm uns}| = N$ was the case (typically $|\mathcal{I}^{\rm uns}| \ll \sum_{i=1}^N c_i$).

It should be noted that all the above figures do not consider evaluating and storing the full optimizer as only its first element is required for implementation of PWA controllers in a receding horizon fashion.

In terms of complexity reduction in explicit MPC, one may compare the proposed convex lifting approach, e.g., with the clipping-based implementation of [24] as they both exploit the concept of clipping. The latter, however, relies on replacing some of the saturated regions by extensions of the unsaturated ones, while the achievable reduction may range from none to the case when the new partition has $|\mathcal{I}^{\rm uns}|$ regions. Another technique of [25] in turn requires to only store the unsaturated regions by employing a separating function. Clearly, both of the aforementioned approaches necessitate storing a modified state-space partition, and hence performing the point location at each sampling instant. Alternatively, a regionless implementation of explicit MPC was proposed in [11], and recently extended in [26]. Its nature, however, renders it applicable for

MPC problems with rather larger parametric space and short prediction horizons.

It is worth recalling that the technique presented in [7] proposes for efficient implementation of PWA controllers the use of a so-called descriptor function for parametric quadratic programming and the optimal cost function for parametric linear programming, which allow to avoid storing the state-space partition. Therefore, its storage demand is $N(1+d_u)(d+1)$ real numbers which is much less than $\sum_{i=1}^{N}(c_i+d_u)(d+1)$ real numbers required by the original implementation. However, it is usually the case that $N(1+d_u)(d+1) \gg |\mathcal{I}^{\rm uns}|(1+d_u)(d+1)+2d_u$, since $N \gg |\mathcal{I}^{\rm uns}|$. Accordingly, the method in [7] requires a larger memory footprint than the one by Algorithm 6. Moreover, the online implementation of the technique in [7] requires $(2d-1)N+\sum_{i=1}^{N} c_i$ FLOPs in the worst case, which is much more demanding than the one by Algorithm 6 with $2|\mathcal{I}^{\rm uns}|d+2d_ud+2d_u$ FLOPs since $\sum_{i=1}^{N} c_i \gg N \gg |\mathcal{I}^{\rm uns}|$.

On the other hand, the method in [8] makes use of a hash table which allows for acceleration of the online evaluation, however, at the price of additional storage requirement besides the memory needed to store the state-space partition. Consequently, this method does not help reducing the memory footprint. For reading ease, a summary of these aspects is reported in Table II.

It is worth emphasizing that the method in [42] presents a construction of binary search trees which can achieve a logarithmic time for the point-location problem. However, the memory footprint requirement is still more demanding than the proposed method in Subsection IV-B, since in addition to the constructed binary search tree, it also requires to store all the unique hyperplanes of the state-space partition whose number is much larger than the number of unsaturated regions. To illustrate this point, in the example of Subsection V-A with $N_{\rm p}=20$, the implementation presented in Subsection IV-B requires to store 73 hyperplanes, while the one in [42] needs to store 3270 hyperplanes besides a binary search tree of 5189 nodes. Combination of lattice representation of PWA functions and truncated binary search tree is introduced in [9] to provide a better trade-off between the memory footprint requirement and the online evaluation over the method in [42]. For these elements, drawing a clear comparison between this method and the ones based on convex lifting is not straightforward.

Remark IV.10 We remark that the limited computational accuracy might practically lead to the case where several neighboring regions are associated with very similar affine functions of a convex lifting, if the scalar constant $c>0$ chosen in (15) or (29) is too small. This case can be avoided by increasing c to a sufficiently large value, e.g., c should be greater than a tolerable error but not large in absolute value as long as it affects the slope of the respective affine functions.

In order to assess the scalability of the proposed implementations, one can see that it primarily depends on the tractability of the construction of convex liftings. As for the constructions of convex lifting in Algorithms 2 and 3, the number of constraints of the optimization problems (16) and (30) scales quadratically with the number of regions in the given partition. Therefore, the construction of convex liftings becomes more demanding, as the number of regions and dimension increase. However, since the construction of a convex lifting is performed offline, it is reasonable to assume that sufficient computational resources are available. In addition, one may choose among a plethora of efficient linear/quadratic programming solvers. Note also that the number of unsaturated regions is usually much smaller than the one of the original partition, i.e., $|\mathcal{I}^{\rm uns}| \ll N$, therefore we only need to work with partitions of much less regions. To this end, we refer the reader to Table I for illustration.

TABLE I: Computational time for the constructions of convex liftings via different examples.

d	d_u	N	$\lvert\mathcal{I}^{\rm uns}\rvert$	Algo. 2 [s]	Algo. 3 [s]
2	1	1089	99	0.49	0.40
		3640	139	0.43	0.53
		5583	173	0.57	0.50
		5207	679	1.02	2.61
3	1	591	115	0.77	0.71
		1887	223	2.49	1.49
		2845	377	5.75	3.13
4	2	655	73	0.44	0.43
		437	149	2.36	1.15
		1681	401	18.64	1.91
		1913	1425	1205.73	67.35
5	1	57	37	0.45	0.32
	2	821	651	12.67	10.11
		963	729	24.12	15.47
		1382	901	86.41	16.26
	3	992	992	789.62	42.05

V. Illustrative examples

A. Example 1

To illustrate the above proposed schemes, consider the double integrator system:

$$x_{k+1}=\begin{bmatrix}1 & 0.5\\ 0 & 1\end{bmatrix}x_k+\begin{bmatrix}0.125\\ 0.5\end{bmatrix}u_k. \tag{48}$$

We design a PWA controller based on linear model predictive control which minimizes the following quadratic cost function:

$$\sum_{i=0}^{N_{\rm p}-1}\left(x_{k+i|k}^T Q x_{k+i|k}+u_{k+i|k}^T R u_{k+i|k}\right)+x_{k+N_{\rm p}|k}^T P x_{k+N_{\rm p}|k},$$

where $Q=I_2$, (I denotes an identity matrix of suitable dimension), $R=10$, P is set to be the solution of discrete-time Riccati equation and the prediction horizon $N_{\rm p}$ equal to 10. Note that $x_{k+i|k}, u_{k+i|k}$ denote the state and control variables at time $k+i$, predicted at instant k. This problem is subject to the following constraints:

$$\begin{aligned}&-2\le u_{k+i|k}\le 2 \ \text{ for } \ 0\le i\le N_{\rm p}-1,\\ &x_{k+N_{\rm p}|k}\in\mathcal{X}_{\rm f},\end{aligned}$$

where $\mathcal{X}_{\rm f}$ denotes the terminal constraint set as the maximal positively invariant set associated with the local control law $u=\begin{bmatrix}-0.2554 & -0.7590\end{bmatrix}x$. The above problem is explicitly solved using MPT 3.0 [22]. The resulting PWA controller is presented in Fig.7 with the red and green controllers denoting where the control action attains the minimal and maximal

TABLE II: A comparison of memory footprint and online evaluation of different methods

Method	Memory footprint [# real numbers]	Online evaluation [FLOPs]
Original explicit solution	$\sum_{i=1}^{N}(c_i+d_u)(d+1)$	$\sum_{i=1}^{N} c_i(2d+1) \; + \; 2d_u d$
Clipping-based approach [24]	$\sum_{i\in\mathcal{I}_{\tilde{N}}}(\tilde{c}_i+d_u)(d+1)$[a]	$2dd_u + \sum_{i\in\mathcal{I}_{\tilde{N}}} \tilde{c}_i(2d+1)$
Separation-based approach [25]	$\sum_{i\in\mathcal{I}^{\mathrm{uns}}}(c_i+d_u)(d+1)$	$2dd_u + \sum_{i\in\mathcal{I}^{\mathrm{uns}}} c_i(2d+1)$
CL-based approach per Sec. IV-B	$\lvert\mathcal{I}^{\mathrm{uns}}\rvert(1+d_u)(d+1)+2d_u$	$2\lvert\mathcal{I}^{\mathrm{uns}}\rvert d + 2d_u d + 2d_u$
Descriptor function [7]	$\lvert\mathcal{I}_N\rvert(1+d_u)(d+1)$	$(2d-1)\lvert\mathcal{I}_N\rvert + \sum_{i=1}^{N} c_i$

[a] $\tilde{N}$ denotes the number of regions of the modified partition and $\tilde{c}_i$ denotes the number of halfspaces of its i-th region, while $|\mathcal{I}^{\mathrm{uns}}| \leq \tilde{N} \leq N$ in general.

values, respectively. In particular, the corresponding state-space partition consists of 73 unsaturated regions, 160 regions where $u(x) = -2$ and 160 regions where $u(x) = 2$. Storing all the 393 regions in this case amounts to 4758 real numbers, with additional 1179 real numbers needed to encode the PWA control law. Assuming double precision arithmetics, the total memory footprint of the original MPC controller is 48 kilobytes. The worst-case computational effort required for its online evaluation is 7934 FLOPs.

Fig.8 in turn depicts a convex lifting obtained from Algorithms 2 and 5. The associated state-space partition $\{\mathcal{Y}_j\}_{j\in\mathcal{I}_M}$ is shown only for illustration, where the red and green regions represent the rearrangement of the partitions $\{\mathcal{X}_i\}_{i\in\mathcal{I}^{\min}}$ and $\{\mathcal{X}_i\}_{i\in\mathcal{I}^{\max}}$, respectively, according to the constructed convex lifting. The yellow regions represent $\{\mathcal{X}_i\}_{i\in\mathcal{I}^{\mathrm{uns}}}$, which does not change after the rearrangement. This regionless MPC implementation hence requires to only store the convex lifting and the corresponding PWA feedback, which in total amounts to 6.7 kB, implying a reduction by a factor of 7.2. Evaluating the optimal control action requires 559 FLOPs, which is 14.2 times less than in the case of the original explicit solution.

Finally, a convex lifting constructed per Algorithm 2 and equation (45) is depicted in Fig.9. The total memory footprint of the resulting regionless controller is 3.5 kB, i.e., the memory consumption is reduced by a factor of 13.7. Accordingly, the online evaluation effort reduces to mere 298 FLOPs, which is 26.6 times faster than the worst-case runtime of the original controller.

To assess how the controller complexity in the considered MPC example scales with problem size, in particular with the prediction horizon, we report the related memory consumption data in Table III. The proposed approaches are also compared with the clipping and separation based MPC implementations of [24] and [25], respectively. One may observe the significant complexity reduction of explicit solutions achieved via the convex lifting based techniques (denoted by CL), in particular the latter one, described in Section IV-B. The online evaluation effort is omitted here for brevity, however, it scales better in favor of the convex lifting based approach as it does not perform the traditional point location (c.f. Section IV-C). This clearly allows for controller deployment even on low-end embedded microcontroller platforms with limited storage capacity and computational power. Similarly, the offline computational time is substantially lower for the proposed approach (1-2 s to obtain $\ell^{\mathrm{uns}}(x)$ for $N_{\mathrm{p}} = 50$). We remark that the approach is accordingly applicable for problems involving a higher number of parameters or optimization variables.

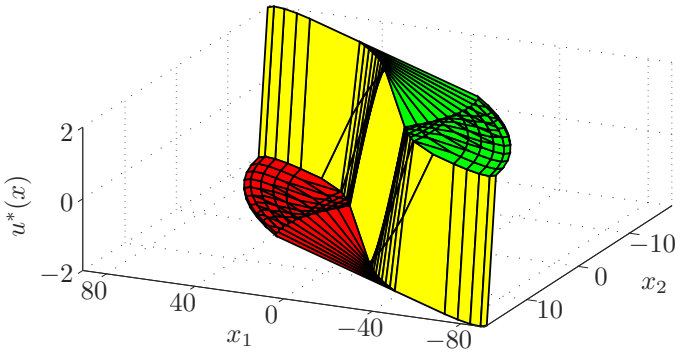

Fig. 7: The original PWA controller.

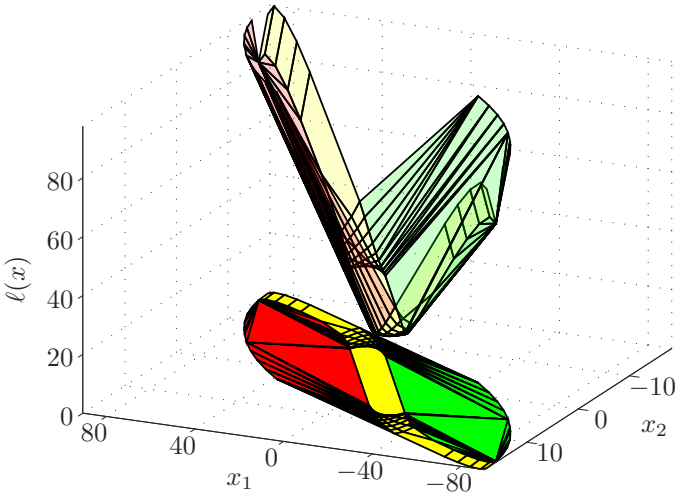

Fig. 8: A convex lifting obtained from Algorithms 2 and 5 and its associated state-space partition.

B. Example 2

In this subsection, we consider a higher-dimensional system to show more clearly the efficiency of the proposed algorithms. To this goal, the quadruple tank system in [23] is accounted

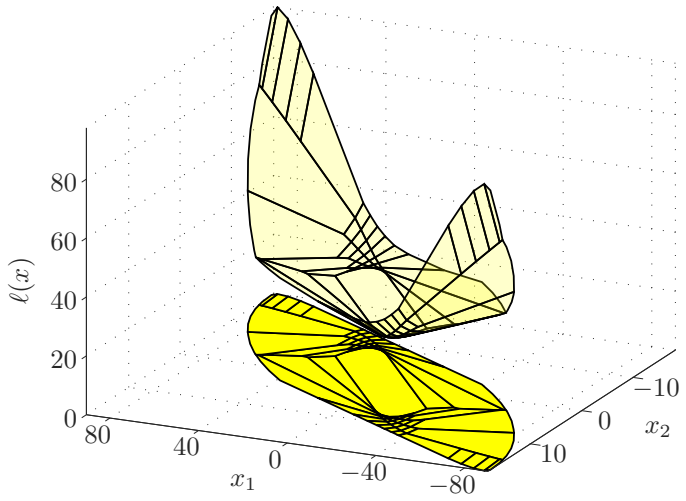

Fig. 9: A convex lifting obtained from Algorithm 2 and equation (45) and its associated state-space partition.

TABLE III: Memory consumption in kilobytes for different implementations of the MPC example.

	$N_{\mathrm{p}}=20$	30	40	50
Original explicit solution	133.4	265.1	445.7	681.7
Clipping-based approach [24]	49.6	95.3	242.8	401.9
Separation-based approach [25]	12.1	14.5[a]	17.0[a]	21.1[a]
CL-based approach per Sec. IV-A	10.8	14.6	18.5	24.1
CL-based approach per Sec. IV-B	4.8	5.7	6.7	8.3

[a] These figures were obtained from theoretical formulas since the computation failed in these cases.

for and its mathematical model is represented as follows:

$$x_{k+1} = Ax_k + Bu_k$$

where matrices A, B are given below:

$$A = \begin{bmatrix} 0.9762 & 0 & 0.0129 & 0 \\ 0 & 0.9719 & 0 & 0.0086 \\ 0 & 0 & 0.9870 & 0 \\ 0 & 0 & 0 & 0.9913 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.0024 & 0 \\ 0 & 0.0021 \\ 0 & 0.0014 \\ 0.001 & 0 \end{bmatrix}.$$

The state and control variable are subject to the following constraints:

$$\|x_k\|_\infty \leq 20, \quad \|u_k\|_\infty \leq 2.$$

An explicit MPC controller is computed by minimizing the same quadratic cost function as in Subsection V-A with $Q = I_4$, $R = I_2$, P is also obtained from the Riccati equation and $N_{\mathrm{p}} = 20$. For simplicity, terminal constraints are not considered in this example, accordingly one obtains a controller associated with 439 regions and 57 of them are unsaturated. The memory footprint and the online evaluation of different methods are reported in Table IV. This result again emphasizes the benefit of the convex lifting based approach in implementation of PWA controllers. Application of the proposed methods on the real-time cantilever beam system is also studied in [20], the interested reader is referred to this reference for further detail.

TABLE IV: Comparison of different implementation methods where memory consumption and the worst-case online evaluation are in kilobytes and FLOPs, respectively.

Methods	Memory	Onl. evaluation
Original explicit solution	167.6	30724
Clipping-based approach [24]	26.6	5128
CL-based approach per Sec. IV-B	6.7	476

VI. Conclusion

This paper presented the concept of convex liftings and its application in control theory. Accordingly, two different algorithms to construct convex liftings have been put forward. This concept was also shown to be useful for efficient implementation of PWA controllers via two proposed schemes. They allow for significant reduction of both the storage requirement and runtime complexity. Finally, complexity analysis and numerical examples were considered to illustrate advantages of the proposed techniques compared to existing methods.

References

[1] A. Airan, M. Bhushan, and S. Bhartiya, "Linear machine solution to point location problem," *IEEE Transactions on Automatic Control*, 2016.

[2] G. Aloupis, H. Pérez-Rosés, G. Pineda-Villavicencio, P. Taslakian, and D. Trinchet-Almaguer, "Fitting voronoi diagrams to planar tesselations," in *Combinatorial Algorithms*. Springer, 2013, pp. 349–361.

[3] F. Aurenhammer, "Criterion for the affine equivalence of cell complexes in r^d and convex polyhedra in r^{d+1}," *Discrete & Computational Geometry*, vol. 2, pp. 49–64, 1987.

[4] ——, "Power diagrams: properties, algorithms and applications," *SIAM Journal on Computing*, vol. 16, no. 1, pp. 78–96, 1987.

[5] ——, "Voronoi diagramsa survey of a fundamental geometric data structure," *ACM Computing Surveys (CSUR)*, vol. 23, no. 3, pp. 345–405, 1991.

[6] D. Avis and K. Fukuda, "A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra," *Discrete & Computational Geometry*, vol. 8, no. 1, pp. 295–313, 1992.

[7] M. Baotic, F. Borrelli, A. Bemporad, and M. Morari, "Efficient on-line computation of constrained optimal control," *SIAM Journal on Control and Optimization*, vol. 47, no. 5, pp. 2470–2489, 2008.

[8] F. Bayat, T. A. Johansen, and A. A. Jalali, "Using hash tables to manage the time-storage complexity in a point location problem: Application to explicit model predictive control," *Automatica*, vol. 47, no. 3, pp. 571–577, 2011.

[9] ——, "Flexible piecewise function evaluation methods based on truncated binary search trees and lattice representation in explicit mpc," *IEEE Transactions on Control Systems Technology*, vol. 20, no. 3, pp. 632–640, 2012.

[10] 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.

[11] 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.

[12] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[13] H. Crapo and W. Whiteley, "Plane self stresses and projected polyhedra 1: the basic pattern," *Structural Topology*, vol. 19, pp. 55–73, 1993.

[14] ——, "Spaces of stresses, projections and parallel drawings for spherical polyhedra," *Contributions to Algebra and Geometry*, vol. 35, no. So. 2, pp. 259–281, 1994.

[15] H. Edelsbrunner and R. Seidel, "Voronoi diagrams and arrangements," *Discrete & Computational Geometry*, vol. 1, no. 1, pp. 25–44, 1986.

[16] T. Gal, *Postoptimal analyses, parametric programming and related topics*. Walter de Gruyter, 1995.

[17] A. Grancharova and T. A. Johansen, *Explicit Nonlinear Model Predictive Control*. Springer, 2012.

[18] B. Grünbaum, *Convex polytopes*. Wiley Interscience, 1967.

[19] M. Gulan, N. A. Nguyen, S. Olaru, P. Rodriguez-Ayerbe, and B. Rohal'-Ilkiv, "Implications of inverse parametric optimization in model predictive control." in *Developments in Model-Based Optimization and Control*, S. Olaru, A. Grancharova, and F. L. Pereira, Eds. Springer, 2015, pp. 49–70.

[20] M. Gulan, G. Takács, N. A. Nguyen, S. Olaru, P. Rodriguez-Ayerbe, and B. Roha-Ilkiv, "Embedded linear model predictive control for 8-bit microcontrollers via convex lifting," in *the 20th IFAC World Congress*, Toulouse, France, July 9–14 2017.

[21] D. Hartvigsen, "Recognizing voronoi diagrams with linear programming," *ORSA Journal on Computing*, vol. 4, no. 4, pp. 369–374, 1992.

[22] M. Herceg, M. Kvasnica, C. Jones, and M. Morari, "Multi-Parametric Toolbox 3.0," in *European Control Conference (ECC)*, Zürich, Switzerland, July 17–19 2013, pp. 502–510, http://control.ee.ethz.ch/ mpt.

[23] K. H. Johansson, "The quadruple-tank process: a multivariable laboratory process with an adjustable zero," *IEEE Transactions on Control Systems Technology*, vol. 8, no. 3, pp. 456–465, 2000.

[24] M. Kvasnica and M. Fikar, "Clipping-based complexity reduction in explicit MPC," *IEEE Transactions on Automatic Control*, vol. 57, no. 7, pp. 1878–1883, 2012.

[25] M. Kvasnica, J. Hledík, I. Rauová, and M. Fikar, "Complexity reduction of explicit model predictive control via separation," *Automatica*, vol. 49, no. 6, pp. 1776–1781, 2013.

[26] M. Kvasnica, B. Takács, J. Holaza, and S. Di Cairano, "On region-free explicit model predictive control," in *Decision and Control, 54rd IEEE Conference on*, Osaka, Japan, 2015, pp. 3669–3674.

[27] J. C. Maxwell, "On reciprocal diagrams and diagrams of forces," *Philosophical Magazine*, vol. ser. 4. 27, pp. 250–261, l864.

[28] N. A. Nguyen, "Explicit robust constrained control for linear systems: analysis, implementation and design based on optimization," Ph.D. dissertation, CentraleSupélec, Université Paris-Saclay, France, 11/2015.

[29] N. A. Nguyen, S. Olaru, P. Rodriguez-Ayerbe, M. Hovd, and I. Necoara, "Fully inverse parametric linear/quadratic programming problems via convex liftings," in *Developments in Model-Based Optimization and Control*, S. Olaru, A. Grancharova, and F. L. Pereira, Eds. Springer, pp. 27–47.

[30] ——, "Constructive solution to inverse parametric linear/quadratic programming problems via convex liftings," *Journal of Optimization Theory and Applications*, pp. 1–26, 2016.

[31] N. A. Nguyen, S. Olaru, and P. Rodriguez-Ayerbe, "Any discontinuous PWA function is optimal solution to a parametric linear programming problem," in *54th IEEE Conference on Decision and Control (CDC)*. IEEE, 2015, pp. 5926–5931.

[32] ——, "Inverse parametric linear/quadratic programming problem for continuous PWA functions defined on polyhedral partitions of polyhedra," in *54th IEEE Conference on Decision and Control (CDC)*. IEEE, 2015, pp. 5920–5925.

[33] ——, "Recognition of additively weighted voronoi diagrams and weighted delaunay decompositions," in *European Control Conference (ECC)*. IEEE, 2015, pp. 328–333.

[34] N. A. Nguyen, S. Olaru, P. Rodriguez-Ayerbe, M. Hovd, and I. Necoara, "On the lifting problems and their connections with piecewise affine control law design," in *European Control Conference (ECC)*. IEEE, 2014, pp. 2164–2169.

[35] ——, "Inverse parametric convex programming problems via convex liftings," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 2489–2494, 2014.

[36] R. Oberdieck and E. N. Pistikopoulos, "Explicit hybrid model-predictive control: The exact solution," *Automatica*, vol. 58, pp. 152 – 159.

[37] S. Olaru and D. Dumur, "A parameterized polyhedra approach for explicit constrained predictive control," in *43rd IEEE Conference on Decision and Control*, vol. 2, 2004, pp. 1580–1585.

[38] K. Rybnikov, "Polyhedral partitions and stresses," Ph.D. dissertation, Queen University, Kingston, Ontario, Canada, 1999.

[39] A. Schrijver, *Theory of linear and integer programming*. John Wiley & Sons, 1998.

[40] A. Schulz, "Lifting planar graphs to realize integral 3-polytopes and topics in pseudo-triangulations," Ph.D. dissertation, Fachbereich Mathematik und Informatik der Freien Universitat Berlin, 2008.

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

[42] P. Tøndel, T. A. Johansen, and A. Bemporad, "Evaluation of piecewise affine control via binary search tree," *Automatica*, vol. 39, no. 5, pp. 945–950, 2003.

VII. Appendix

Proof of Theorem III.12

Let $\mathcal{H}(\mathcal{X}_i)$ be the set of supporting hyperplanes of $\mathcal{X}_i$ at its facets; also define $\mathcal{H}(\mathcal{X}) = \bigcup_{i\in\mathcal{I}_N} \mathcal{H}(\mathcal{X}_i)$. We will show that the decomposition of $\mathcal{X}$ by $\mathcal{H}(\mathcal{X})$ leads to a new cell complex $\{\mathcal{Y}_j\}_{j\in\mathcal{I}_M}$ which is convexly liftable. As presented above, such a decomposition is denoted as *hyperplane arrangement*. The convex liftability of such a decomposition can be proven by returning to the concept of stresses. (Details about stresses, star, inward unit normal vector, equilibrium can be found in [28]).

We recall that the relative boundary of a cell complex $\{\mathcal{Y}_j\}_{j\in\mathcal{I}_M}$ of a polyhedron $\mathcal{X}$ is the boundary of $\mathcal{X}$. An internal face of $\{\mathcal{Y}_j\}_{j\in\mathcal{I}_M}$ is a face which does not belong to its relative boundary. Considering any internal $(d-2)-$face F_0 of $\{\mathcal{Y}_j\}_{j\in\mathcal{I}_M}$, this $(d-2)-$face F_0 is the intersection of finitely many hyperplanes in $\mathcal{H}(\mathcal{X})$. If $\mathcal{F}^{(d-1)}(F_0)$ denotes the set of all $(d-1)-$faces in the star of F_0 (i.e. the $(d-1)-$faces of $\{\mathcal{Y}_j\}_{j\in\mathcal{I}_M}$ sharing a common facet F_0), then for each $F_i^{(d-1)} \in \mathcal{F}^{(d-1)}(F_0)$, there exists a unique $F_j^{(d-1)} \neq F_i^{(d-1)}$ and $F_j^{(d-1)} \in \mathcal{F}^{(d-1)}(F_0)$ such that $F_i^{(d-1)}, F_j^{(d-1)}$ lie in a common hyperplane of $\mathcal{H}(\mathcal{X})$ and they have a common facet F_0. Accordingly, it can be seen that the inward unit normal vectors to the faces $F_i^{(d-1)}, F_j^{(d-1)}$ at their common facet F_0, denoted by $n(F_0, F_i^{(d-1)})$, $n(F_0, F_j^{(d-1)})$, respectively, satisfy:

$$n(F_0, F_i^{(d-1)}) = -n(F_0, F_j^{(d-1)}).$$

Thus, a pair of coefficients of strictly positive stresses $s(F_i^{(d-1)}), s(F_j^{(d-1)})$ exists (e.g. $s(F_i^{(d-1)}) = s(F_j^{(d-1)}) = 1$) such that:

$$s(F_i^{(d-1)})n(F_0, F_i^{(d-1)}) + s(F_j^{(d-1)})n(F_0, F_j^{(d-1)}) = 0.$$

Applying the same argument for all elements of $\mathcal{F}^{(d-1)}(F_0)$, one can obtain a strictly positive $d-$stress such that F_0 is in equilibrium. ■