Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

Accelerating Explicit Model Predictive Control by Constraint Sorting ⋆

Juraj Holaza ∗, **Juraj Oravec** ∗, **Michal Kvasnica** ∗, **Raphael Dyrska** ∗∗, **Martin Mönnigmann** ∗∗, **Miroslav Fikar** ∗

∗ *Institute of Information Engineering, Automation, and Mathematics, Slovak University of Technology in Bratislava, Slovakia, (e-mail: juraj.holaza@gmail.com, {juraj.oravec, michal.kvasnica, miroslav.fikar}@stuba.sk)*
∗∗ *Automatic Control and Systems Theory, Ruhr-Universität Bochum, Germany, (e-mail: {raphael.dyrska, martin.moennigmann}@rub.de)*

Abstract: Explicit MPC represents one of the fastest ways of real-time MPC implementation. As the explicit MPC policy is optimization-free in real-time control, its efficiency is determined by solving a point location problem. This paper proposes the novel concept of accelerating explicit MPC that significantly speeds up the real-time evaluation of the point location problem. The introduced strategy has two layers: (i) an offline phase determines a *smart* order of the regions to be explored, and (ii) an online phase removes further regions to be explored on the fly based on the current value of the value function. The main advantage of layer (i) is that the order is evaluated offline, therefore, it does not increase the real-time implementation of explicit MPC. The implementation of layer (ii) slightly increases the real-time evaluation but leads to further speed-up of the point location problem. As the proposed layers are based just on the evaluation of some appropriate value function, the main benefit is that these layers are fully applicable also for higher-dimensional systems. Although the accelerated explicit MPC variant does not reduce the worst-case time of solving the point location problem, an extensive case study demonstrates the efficiency of the proposed strategy.

Keywords: Explicit model predictive control, point location problem, sequential search

1. INTRODUCTION

Model Predictive Control (MPC) (García et al., 1989) attracted considerable interest of both, academia and industry, during the past three decades. As MPC provides *optimal* control action taking into account a wide class of constraints, and thanks to its robustness and ease of tuning, MPC has become an alternative to PID and unconstrained LQR control. Therefore, there has been significant effort to formulate MPC in a way suitable for implementation on embedded hardware. Although the industrial applications of MPC originate in petrochemical industries (Cai et al., 2014) for plants with relatively slow dynamics, recent MPC formulations consider MPC implementations for plants with much faster dynamics. Distributed MPC (Wang and Ong, 2014) simplifies and speeds up solving of the optimization problem, on the other hand, there is still a need to solve the optimization problem during real-time control. The nonnegative-least-squares-based method was introduced in Bemporad (2016) to eliminate redundant constraints and to speed-up MPC design.

Explicit MPC represents an effective alternative for MPC implementation, especially, when hardware resources are scarce (Bemporad et al., 2002a). This methodology is based on offline pre-calculating an optimal feedback law for all feasible initial conditions via exploiting parametric programming (Willner, 1967). For the type of MPC problem treated here, the solution takes the form of a piecewise affine (PWA) function defined over a polytopic partition (Bemporad et al., 2002a). Online evaluation of optimal control actions is then restricted only to a mere function evaluation that can be carried out on low-level control platforms without a need for an additional optimization solver (see, e.g., Rauová et al. (2011) for use on a programmable logic controller).

Explicit MPC arguably suffers from two shortcomings. The first one regards the offline computation of the analytical solution, i.e., the explicit feedback law. It is known that this task is computationally expensive and thus restricts the usability of the explicit MPC strategies to small or moderate-sized problems. Even though new techniques have been proposed in the literature (Borrelli et al., 2010; Kvasnica et al., 2015a; Gupta et al., 2011; Mönnigmann, 2019), the dimension of decision variables still has to be

⋆ This publication is the result of a research group linkage cooperation between the Slovak University of Technology in Bratislava and the Ruhr-Universität Bochum supported by the Alexander von Humboldt Foundation. The funding is provided in its entirety by the Federal Ministry of Education and Research. Additionally, the work of STU co-authors was partly supported by the Slovak Research and Development Agency (APVV-15-0007) and the Scientific Grant Agency of the Slovak Republic under the grants (1/0545/20, 1/0585/19). The work of RUB co-authors was partly supported by the German Federal Ministry for Economic Affairs and Energy under grant 0324125C.

kept small. The second shortcoming arises, because identifying the affine piece that is optimal for the current state, the *point location problem*, may take as long as solving the corresponding online optimization problem, even though sophisticated approaches exist for it (Christophersen et al., 2007; Tøndel et al., 2002; Herceg et al., 2013b; Baotić et al., 2008). This paper aims to address the second issue.

Several approaches to solving the point location problem exist. The simplest one is the sequential search algorithm that traverses through regions until a region containing the current state measurement is located. In Herceg et al. (2013a) authors extend this concept by adding outer box constraints for each region and exploiting them to perform a prior check. In Borrelli et al. (2001) authors suggest to discard the polytopic partition and evaluate optimal control actions solely based on the PWA value function. Binary trees, which were originally proposed around the time the explicit law was first described (Tøndel et al., 2002), are still competitive. The downside, however, of this approach is that well-balanced binary search tree is usually difficult to construct, especially, if the explicit feedback law is complex. Needlessly to say, the main attribute that increases the evaluation time in all of the aforementioned techniques is the number of regions of the analytical solution. Therefore, in order to accelerate the point location problem, one can also consider applying advanced inner/outer approximation methods (Oravec et al., 2013) or memory reduction techniques, e.g., Geyer et al. (2008), Jones and Morari (2010), Holaza et al. (2015), Bakaráč et al. (2018), to name a few.

In this paper, a novel acceleration technique of explicit MPC feedback laws is proposed [1]. One can easily embed the technique in a majority of the commonly used online evaluation methods. This approach is motivated by the work of Jost et al. (2017), where it was shown that constraints known to be inactive can be removed from the optimization problem to reduce its complexity. To detect inactive constraints, Jost et al. (2017) use offline calculated bounds for every constraint that represents the smallest cost function value for which this constraint can be active. Online, the cost function value is compared to these bounds and if it drops below one or more bounds, the corresponding constraints are detected to be inactive and can be removed for future time steps. This idea of using bounds for an auxiliary function is adopted and transferred to the point location problem arising in explicit MPC. Concretely, the aim is to calculate bounds offline to identify the regions that form the solution over the feasible set. Note, that the proposed strategy does not reduce the worst-case time. However, it will be shown by means of illustrative examples that the point location problem can be accelerated even by a factor of three.

2. PROBLEM STATEMENT

Let *polyhedron* refer to a set that is defined by the intersection of a finite number of halfspaces. A polyhedron is called *polytope* if it is bounded. Let a finite number of polyhedra $\mathcal{R}_1, ..., \mathcal{R}_s$ be abbreviated by $\mathcal{R}_{\{1,...,s\}}$.

[1] The extended version of this work is available as a Technical Note at: `https://www.uiam.sk/assets/publication_info.php?id_pub=2131`.

The aim is to control linear discrete-time time-invariant systems given as

$$x(k+1) = Ax(k) + Bu(k),\ k \geq 0, \tag{1}$$

where $x(k) \in \mathbb{R}^n$, $u(k) \in \mathbb{R}^m$ are the state and input vectors at time k, respectively, and A, B have the obvious dimensions. Assume that (A, B) is a controllable pair and the system in (1) is subjected to linear constraints

$$x(k) \in \mathbb{X},\ u(k) \in \mathbb{U}, \quad \forall k \geq 0, \tag{2}$$

where $\mathbb{X} \subset \mathbb{R}^n$ and $\mathbb{U} \subset \mathbb{R}^m$ are compact convex sets containing the origin in their interiors.

To regulate the system (1) to the origin, the optimal control problem of the form

$$\min_{\substack{u(k),\, x(k+1),\\ k=0,\dots,N-1}} \quad \|x_N\|_P^2 + \sum_{k=0}^{N-1} \left(\|x_k\|_Q^2 + \|u_k\|_R^2\right) \tag{3a}$$

$$\text{s.t.}: \ x(k+1) = Ax(k) + Bu(k), \tag{3b}$$

$$x(k) \in \mathbb{X},\ \ u(k) \in \mathbb{U},\ k = 0, \dots, N-1, \tag{3c}$$

$$x_N \in \mathbb{X}_\mathrm{N}, \tag{3d}$$

is considered for a given initial condition $x(0)$ and matrices $P \succ 0$, $Q \succ 0$, $R \succ 0$ with the obvious dimensions. The terminal set $\mathbb{X}_\mathrm{N} \subset \mathbb{R}^n$ is assumed to be a compact and convex set that contains the origin in its interior. Furthermore, assume that (3) has a solution for all initial conditions $x(0)$ from a useful set $\mathbb{F} \supset \mathbb{X}_\mathrm{N}$. Let $U^\star = (u^\star(0), \dots, u^\star(N-1))$ refer to the sequence of optimal input signals for an arbitrary but fixed $x(0) \in \mathbb{F}$. The MPC-controlled system results from solving (3) on a receding horizon and applying $u^\star(0)$ in every time step. The resulting feedback control law by is denoted by $\kappa : \mathbb{F} \to \mathbb{R}^m$, where $\kappa(\theta) \in \mathbb{R}^m$ is equal to $u(0)$ that results from solving (3) for parametric constraint $x(0) = \theta$. Note, the optimal cost function (3a) is a Lyapunov function for the closed-loop system if $\mathbb{X}_\mathrm{N}$ is the maximal set on which the constrained problem (3) and the corresponding unconstrained problem have the same solution, and P is set to the solution of the discrete-time algebraic Riccati equation for the unconstrained problem.

The feedback law $\kappa : \mathbb{F} \to \mathbb{R}^m$ is known to be a continuous piecewise affine (PWA) function on a partition of $\mathbb{F}$ into polytopes (Bemporad et al., 2002b). More precisely, there exist $F_i \in \mathbb{R}^{m\times n}$ and $f_i \in \mathbb{R}^m$, $i = 1, \dots, r$ such that

$$\kappa(\theta) = F_i\theta + f_i \quad \text{if} \quad \theta \in \mathcal{P}_i, \tag{4}$$

where $\mathcal{P}_i \subset \mathbb{R}^n$ are bounded polytopes with pairwise disjoint interiors such that $\mathbb{F} = \cup_{i=1}^r \mathcal{P}_i$, and r is number of polytopes. Furthermore, the optimal value function, which is denoted by $J^\star : \mathbb{F} \to \mathbb{R}$, is convex, continuous, and piecewise quadratic.

If the optimal feedback law (4) is known, the solution to the optimal control problem (3) is found in two stages. In the first step, for a given initial condition θ, an active region $\mathcal{P}_i$ is determined such that $\theta \in \mathcal{P}_i$ holds. In this paper, the *sequential search* outlined in Algorithm 1 is considered. In the second step, the corresponding affine function $\kappa(\theta) = F_{i^\star}\theta + f_{i^\star}$ from (4) is evaluated yielding the optimal feedback signal.

The objective of this paper is to accelerate the sequential search Algorithm 1, i.e., to reduce the number of explored

Algorithm 1 Sequential search point location problem.

Inputs: list of indices $\mathcal{I}$, feedback law $\kappa(\theta)$ as in (4)
Output: index $i^\star$ of the active region
for $i \in \mathcal{I}$ **do**
  if $\theta \in \mathbb{P}_i$ **then**
    $i^\star \leftarrow i$
    break
  end if
end for

regions, via rearranging and reducing the set of indices $\mathcal{I}$ to be considered.

Remark 2.1. The proposed strategy is *not* limited to the formulation of (3), as the control law in (4) can be obtained from (3) considering various MPC setups, e.g., *robust* explicit MPC for models with parametric and/or additive disturbances (Kvasnica et al., 2015b); explicit MPC for piecewise affine models (Cseko et al., 2015), *hybrid* explicit MPC (Oberdieck and Pistikopoulos, 2015), etc. This requires replacing the linear dynamics in (3b) with an appropriate nonlinear model.

Setting up the binary tree-based search involves finding an optimal (generally at most locally optimal) sequence of halfspaces, a problem which can in general not be solved to global optimality. A simple heuristics for speeding up this problem would be to consider the halfspaces in the two orders suggested here. This way one would have to look for an optimal order of the halfspaces that define the central LQR polytope first, then for an optimal order of the halfspaces of the next layer of polytopes surrounding the central one, and so on. The order by which the polytopes are selected would again be induced by either of the two methods used here. After all, this is only a heuristic, but then the problem of setting up the binary tree always involves some sort of heuristic.

3. ACCELERATING EXPLICIT MPC

The main bottleneck of the sequential search Algorithm 1 is that to allocate the active region $\theta \in \mathcal{P}_{i^\star}$ one needs to verify also regions $\mathcal{P}_i$ for all $i \in \mathcal{I}$ with $i < i^\star$. Obviously, by avoiding exploration of some of these inactive regions, the point location problem speeds up. Two techniques, how to achieve this goal, are introduced in this section. The basic idea is to calculate minimal/maximal values of a support function $V(\theta)$, over each region $\mathcal{P}_i$, and exploit them to exclude redundant regions. It will be shown that these techniques can be combined to even further accelerate the search for the index $i^\star$. Both techniques are based on the following assumption:

Assumption 3.1. Let $V : \mathbb{F} \rightarrow \mathbb{R}$ be a strictly convex function. □

Note, the different functions $V(\theta)$ lead to various accelerations.

3.1 Minimal Value Approach

Each region $\mathcal{P}_i$ of the optimal control law (4) is characterized by the lowest value σ_i the optimal value function attains on it. Since V is strictly convex on $\mathbb{F}$ by assumption, it has a unique minimum on every $\mathcal{P}_i$, $i \in \mathcal{I}$. Let

$$\underline{d}_i := \min_{\theta \in \mathcal{P}_i} V(\theta), \tag{5}$$

for every $i \in \mathcal{I}$. Determining the bounds $\underline{d}_i$ is a tractable problem, since a finite number of convex quadratic problems needs to be solved offline, i.e., before the runtime of MPC. The cornerstone of the accelerating technique is based on our first main result.

Lemma 3.2. Assume an explicit MPC law (4), a function $V(\cdot)$ that satisfies Assumption 3.1 and $\underline{d}_i$ for all $i \in \mathcal{I}$ are known. Let $\theta \in \mathbb{F}$ be arbitrary. Then, for any $i \in \mathcal{I}$, $V(\theta) < \underline{d}_i$ implies $\theta \notin \mathcal{P}_i$.

Proof. Since $\underline{d}_i$ is the global minimum of $V(\cdot)$ on $\mathcal{P}_i$, there exists no $\tilde{\theta} \in \mathcal{P}_i$ such that $V(\tilde{\theta}) < \underline{d}_i$.

The implementation of the proposed approach is formulated in Algorithm 2. Specifically, at each sample instant, the function $V(\cdot)$ is evaluated yielding a scalar $\sigma(k)$. The algorithm then traverses through indices $i \in \mathcal{I}$ verifying the primal condition $\sigma(k) \geq \underline{d}_i$ and the secondary condition $\theta \in \mathcal{P}_i$ until both of them are satisfied at the same time. Algorithm 2 is, generally, more efficient than Algorithm 1 as the primal condition decreases the number of evaluated secondary conditions. Moreover, the primal condition is much cheaper to compute than the secondary one as only a scalar comparison is performed.

Algorithm 2 Sequential search algorithm with $\underline{d}_i$.

Inputs: list of indices $\mathcal{I}$, feedback law $\kappa(\theta)$ as in (4), local minimizers $\underline{d}_i$ for all $i \in \mathcal{I}$
Output: index $i^\star$ of the active region
Initialization: compute $\sigma(k) = V(\theta)$
for $i \in \mathcal{I}$ **do**
  if $\sigma(k) \geq \underline{d}_i$ **then**
    if $\theta \in \mathcal{P}_i$ **then**
      $i^\star \leftarrow i$
      break
    end if
  end if
end for

Now note that there exists a sequence of the indices

$$\underline{\mathcal{I}} := (i_1, i_2, \ldots, i_r),\ i_j \in \mathcal{I}, \tag{6}$$

such that

$$\underline{d}_{i_1} \leq \underline{d}_{i_2} \leq \cdots \leq \underline{d}_{i_r}, \tag{7}$$

i.e., let us rearrange the order of indices in $\mathcal{I}$ based on the ascending values of minimizers $\underline{d}_i$. Now recall that (4) is a stabilizing controller that drives the system to the origin. As a result, the first polytopes in the sequence $\underline{\mathcal{I}}$ occur more frequently in the point location problem. Subsequently, the primal condition $\sigma(k) \geq \underline{d}_i$ becomes redundant and one can substitute Algorithm 2 by Algorithm 1 with $\mathcal{I} \leftarrow \underline{\mathcal{I}}$.

By substituting $\mathcal{I} \leftarrow \underline{\mathcal{I}}$ in Algorithm 2 the number of scalar comparisons can therefore be further reduced.

Remark 3.3. Assumption 3.1 admits a wide variety of functions, but the function used for calculating the bounds and the function used for evaluating a current cost function value has to be the same and is denoted as V. Note, however, that the structure of $V(\cdot)$ influences the efficiency of the proposed approach. For example, $V(\theta) = \|\theta\|_2^2$ is a candidate that penalizes the distance of regions from the origin. On the other hand, $V(\theta) \leftarrow J^\star$ minimizing (3a), with $J^\star$ a PWQ function, represents another suitable candidate that takes into account the entire open-loop behavior. Recall that $J^\star$ requires an additional inner point location problem to be evaluated as it is a PWQ function. This problem, nevertheless, is an easy workaround if the sorted list of indices $\underline{\mathcal{I}}$ is used instead of $\mathcal{I}$. Subsequently, no additional computation is required in the online phase of the point location problem.

3.2 Maximal Value Approach

The maximal value approach exploits the same principle as the technique in the previous subsection. The difference, however, is that here each region is associated with a local maximizer $\overline{d}_i \in \mathbb{R}$ that is computed via

$$\overline{d}_i := \max_{\theta \in \mathcal{P}_i} V(\theta), \tag{8}$$

for all $i \in \mathcal{I}$. The acceleration of this technique is then based on the following Lemma.

Lemma 3.4. Assume an explicit MPC policy (4), local maximizers $\overline{d}_i$ for all $i \in \mathcal{I}$, parameter vector $\theta(k) \in \mathbb{F}$, and a function $V(\cdot)$ satisfying Assumption 3.1 are given. If $V(\theta(k)) > \overline{d}_i$ then $\theta(k) \notin \mathcal{P}_i$.

Proof. Follows directly from the proof of Lemma 3.2.

Remark 3.5. Notice that the optimization problem (8) is no longer convex. There are two principal ways how to solve the non-convex optimization problem in (8). The first approach is based on *vertex enumeration*. It is suitable for the modest complexity of the parametric optimization problem in (3). The local maximizers $\overline{d}_i$ are computed via evaluating $V(\cdot)$ at the vertices of $\mathcal{P}_i$ and by taking the maximal value among them. However, such vertex enumeration technique becomes quickly intractable for higher dimensions. The second approach is *global non-convex optimization* if, for some reason, the vertices cannot be evaluated.

The implementation of Lemma 3.4 is straightforward. One needs to modify the Algorithm 2 by substituting $\sigma(\theta) \geq \underline{d}_i$ with $\sigma(\theta) \leq \overline{d}_i$. Needlessly to say, the set of indices $\mathcal{I}$ can be also a priori sorted with descending order

$$\overline{\mathcal{I}} := (k_1, k_2, \ldots, k_r),\ k_l \in \mathcal{I}, \tag{9}$$

such that

$$\overline{d}_{k_1} \geq \overline{d}_{k_2} \geq \cdots \geq \overline{d}_{k_r}. \tag{10}$$

Subsequently, the Algorithm 2 boils down to the standard sequential search Algorithm 1 with $\mathcal{I} \leftarrow \overline{\mathcal{I}}$.

3.3 Bounded Value Approach

The bounded value approach combines both of our results from Lemma 3.2 and Lemma 3.4 to achieve even greater acceleration of the point location problem.

Lemma 3.6. Assume an explicit MPC policy (4), local minimizers $\underline{d}_i$ and maximizers $\overline{d}_i$ for all $i \in \mathcal{I}$, parameter vector $\theta(k) \in \mathbb{F}$, and a function $V(\cdot)$ satisfying Assumption 3.1 are given. If

$$\overline{d}_{k_l} < V(\theta(k)) < \underline{d}_{i_j}, \tag{11}$$

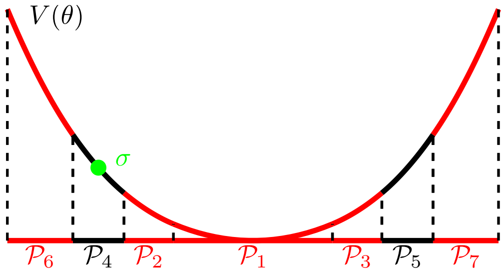

Fig. 1. Illustrative example of the bounded value approach.

for some $k_l \in \overline{\mathcal{I}}$ and some $i_j \in \underline{\mathcal{I}}$, then

$$\theta \notin \mathcal{P}_{k_{l'}}\ \forall k_{l'} \geq k_l, \qquad \theta \notin \mathcal{P}_{i_{j'}}\ \forall i_{j'} \leq i_j.$$

Proof. By definition of $\underline{d}_{i_j}$ in (5), $V(\theta(k)) < \underline{d}_{i_j}$ implies $\theta \notin \mathcal{P}_{i_j}$. Since the order $\overline{\mathcal{I}}$ implies $V(\theta(k)) < \underline{d}_{i_{j'}}$ for all $i_{j'} \leq i_j$, implies $\theta \notin \mathcal{P}_{i_{j'}}$ by the same argument. The statements for the upper bounds $\overline{d}_{k_l}$ can be shown accordingly.

The implementation of the proposed approach is compactly formulated in Algorithm 3, where the restriction (11) is simplified to $V(\theta(k)) \leq \overline{d}_i$ via exploiting the ordered list $\underline{\mathcal{I}}$. The reason behind choosing the ascending order $\underline{\mathcal{I}}$ instead of $\overline{\mathcal{I}}$ is that the controller drives the system's states to the origin. This means that the parameter vector θ will converge to the terminal set that is indexed as the first index in $\underline{\mathcal{I}}$. In such case, the Algorithm 3 requires the fewest operations to determine the index of the active region, i.e., $i^\star = 1$.

Algorithm 3 Bounded sequential search algorithm.

Inputs: list of indices $\underline{\mathcal{I}}$, feedback law $\kappa(\theta)$ as in (4), local maximizers $\overline{d}_i$ for all $i \in \underline{\mathcal{I}}$
Output: index $i^\star$ of the active region
Initialization: compute $\sigma(k) = V(\theta)$
for $i \in \underline{\mathcal{I}}$ **do**
  if $\sigma(k) \leq \overline{d}_i$ **then**
    if $\theta \in \mathcal{P}_i$ **then**
      $i^\star \leftarrow i$
      break
    end if
  end if
end for

Example 3.7. Algorithm 3 is illustrated in Figure 1. Here, the explicit MPC policy is defined over 7 polyhedral regions $\mathcal{P}_i$ with $i \in \mathcal{I}$. By evaluating the convex function $V(\theta)$ for a given parameter vector θ, one obtains η (denoted by green color). This scalar is then compared with local minimizers $\sigma \geq \underline{d}$ admitting only regions $\mathcal{P}_{\{1,2,3,4,5\}}$. Regions $\mathcal{P}_{\{6,7\}}$ are omitted as, by Lemma 3.2, they do not contain θ. Equivalently, by validating with local maximizers $\sigma \geq \overline{d}$, regions $\mathcal{P}_{\{1,2,3\}}$ are omitted via Lemma 3.4. By combining both Lemmas, a restricted search interval of only two regions $\mathcal{P}_{\{4,5\}}$ is obtained, and they are shown in black color. All redundant regions are red. In summary, the proposed Algorithm 3 determines the active region $\theta \in \mathcal{P}_4$ after one region exploration and four logical comparisons.

The approach proposed in this section requires storing the list of indices $\underline{\mathcal{I}}$ and the ordered bounds (10). The

Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

additional online computations amount to comparisons of real numbers and bit-shifting operations.

Remark 3.8. Although all proposed approaches aim to decrease the average online computation of explicit MPC policies, the worst-case evaluation time is not improved. On the other hand, as system states converge to the origin, the worst-case time of each iteration is reduced in each control step.

Remark 3.9. As the implementation of accelerating explicit MPC ensures decreased average online computation time, the proposed strategy leads to improvements in terms of energy savings. E.g., implementation of explicit MPC on embedded hardware may significantly prolong the battery life, i.e., the total operation time of the autonomous vehicle or unmanned aerial vehicle, etc.

Remark 3.10. The bounds $\overline{d}_{k_l}$, $\underline{d}_{i_j}$ required in Lemma 3.6 can efficiently be determined by interval bisection: assume $V(\theta)$ is given, then by checking $V(\theta) < \underline{d}_{\lfloor r/2 \rfloor}$ one can determine whether, roughly speaking, the lower or upper half of the bounds (7) needs to be analyzed. In every subsequent step, approximately half of the remaining bounds (7) can be discarded until only one of them remains, which is the desired $\underline{d}_{i_j}$. Note that the required $\lfloor \cdot/2 \rfloor$ operations can efficiently be carried out by a bit shifting operation, thus no divisions are necessary.

4. CASE STUDIES

Two extensive case studies were evaluated to investigate the properties of the proposed strategy [2]: (i) a *double integrator* system and (ii) a set of 20 randomly generated systems in $\theta \in \mathbb{R}^5$.

4.1 Double Integrator

The double integrator system, in the discrete-time domain with $T_s = 0.1$ seconds, is represented by (1), (2), considering

$$A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \; B = \begin{bmatrix} 1 \\ 0.5 \end{bmatrix}, \; -5 \preceq x \preceq 5, \; -1 \leq u \leq 1, \quad (12)$$

for all k. MPC policy (3) was designed with prediction horizon $N = 6$, weighting matrices $Q = \text{diag}(1, 1)$ and $R = 0.1$. Moreover, terminal penalty P and terminal set $\mathbb{X}_N$ were computed using the LQR-based strategy. By solving (3) parametrically, for all feasible initial conditions θ, explicit MPC policy (3) defined over $r = 55$ polytopic regions was obtained. We used the MPT toolbox (Herceg et al., 2013a) with the `PLCP`, `enum`, and `mpqp` parametric solvers. [3]

For illustrative purposes, Figure 2 depicts the value function $J^\star$ defined over the polytopic partition with $r = 55$. It can be observed that the most exploited regions, in the neighborhood of the origin, are associated with low values of the support value function $V(x)$. Therefore, by exploring these regions with the ascending order, the sequential search by Algorithm 1 was accelerated.

[2] Also an inverted pendulum case study can be found in the Technical Note at: `https://www.uiam.sk/assets/publication_info.php?id_pub=2131`.

[3] Note that parametric solver plays a crucial part in indexing of the polytopic partition, i.e., by using different parametric solvers one can obtain various set of indices $\mathcal{I}$.

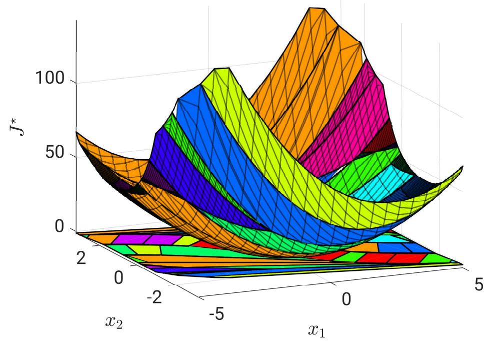

Fig. 2. PWQ Lyapunov cost function above the feedback partition.

To assess the online evaluation complexity of the constructed explicit MPC policy, $N-$step closed-loop simulations were performed for the set of 10^2 uniformly distributed initial conditions over the feasibility parameter-space domain. Specifically, online complexity was determined w.r.t. the number of evaluated half-spaces during the point location problem. The results are summarized in Table 1. Hereafter, η denotes the number of explored half-spaces and Δ denotes relative acceleration in percentage w.r.t. Algorithm 1.

As can be seen, the *minimal value approach* from Subsection 3.1 accelerated the point location problem by 110 %, while the *bounded value approach* from Subsection 3.3 attained an acceleration of 184 %.

Considering another *distance function*, e.g., *upper approximation* of the original PWQ cost function, the acceleration of the point location problem was also attained. The technical details on how to construct such a value function can be found, e.g., in Bakaráč et al. (2018). The total number of explored half-spaces remained larger than considering original PWQ cost function, see Table 2. The *minimal value approach* from Subsection 3.1 accelerated the point location problem by 90 %, while the *bounded value approach* from Subsection 3.3 attained an acceleration of 170 %.

Table 1. Double integrator benchmark: PWQ cost function.

solver	PLCP		enum		mpqp	
	η	Δ	η	Δ	η	Δ
Alg. 1	19 356	–	15 174	–	14 774	–
Alg. 2	9 221	110	9 167	66	9 203	61
Alg. 3	6 820	184	6 452	135	6 470	128

Table 2. Double integrator benchmark: *upper* approximation of PWQ cost function.

solver	PLCP		enum		mpqp	
	η	Δ	η	Δ	η	Δ
Alg. 1	19 356	–	15 174	–	14 774	–
Alg. 2	10 188	90	10 099	50	10 197	61
Alg. 3	7 188	170	7 156	112	7 188	106

4.2 Randomly Generated Systems

For a comprehensive elaboration, the acceleration of the proposed technique was verified for 20 randomly generated

Table 3. Relative acceleration of explicit MPC of set of random systems.

solver	PLCP	enum	mpqp
Alg. 2	51	48	27
Alg. 3	57	58	35

systems of order 5, i.e., with $n = 5$ and $m = 1$. All systems were designed such that the pair (A, B) in (1) was controllable and $-5 \preceq x \preceq 5, -1 \leq u \leq 1$ hold.

The acceleration rate was also investigated analogously to the case study in Section 4.1, i.e., by the number of the reduced half-spaces that were evaluated by the point location problem. Simulations were performed considering *upper* approximation of the original PWQ cost function, within 10^4 equidistantly separated initial conditions. The generated results are summarized in Table 3. The acceleration was evaluated w.r.t. the Algorithm 1. As can be seen, an acceleration of up to 57 % was achieved.

5. CONCLUSIONS

The paper addressed the problem of speeding up the online phase of explicit MPC by introducing a novel acceleration technique. A two-layer explicit MPC scheme with constraints sorting is proposed. First, the *smart* order of regions to be explored is evaluated. Next, the real-time pruning scheme is applied for further reduction of the number of considered regions. Particularly, the speedup of the point location problem is based on the proper *distance function* sorting the regions of the explicit feedback law to be explored. This strategy is also applicable to higher-dimensional systems. The illustrative case study demonstrated the efficiency of the proposed strategy.

Further research will address several aspects. If the constraints (3c) are symmetric, the a partition is also symmetric. This *symmetry* can be exploited to further speed-up solving the point location problem. Another perspective way how to estimate the set of admissible regions to be considered by the point location problem is the offline evaluation of the *1-step maximal reachable sets*, see Kvasnica et al. (2019). Once the current system measurement is associated with the particular 1-step maximal reachable sets, the set of admissible regions where the system states can occur in the next control step is assigned.

REFERENCES

Bakaráč, P., Holaza, J., Klaučo, M., Kalúz, M., Löfberg, J., and Kvasnica, M. (2018). Explicit MPC based on approximate dynamic programming. In *Proc. of ECC*, 1172–1177. IEEE, Limassol, Cyprus.

Baotić, M., Borrelli, F., Bemporad, A., and Morari, M. (2008). Efficient on-line computation of constrained optimal control. *SIAM J. Control Optim.*, 47(5), 2470–2489. doi:10.1137/060659314.

Bemporad, A. (2016). A quadratic programming algorithm based on nonnegative least squares with applications to embedded model predictive control. *IEEE Transactions on Automatic Control*, 61(4), 1111–1116. doi:10.1109/TAC.2015.2459211.

Bemporad, A., Borrelli, F., and Morari, M. (2002a). Model Predictive Control Based on Linear Programming—The Explicit Solution. *TAC*, 47(12), 1974–1985.

Bemporad, A., Morari, M., Dua, V., and Pistikopoulos, E.N. (2002b). The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1), 3–20.

Borrelli, F., Baotić, M., Bemporad, A., and Morari, M. (2001). Efficient On-Line Computation of Explicit Model Predictive Control. Technical report.

Borrelli, F., Baotić, M., Pekar, J., and Stewart, G. (2010). On the computation of linear model predictive control laws. *Automatica*, 46(6), 1035 – 1041.

Cai, X., Tan, Y., Li, S., and Mareels, I. (2014). On the finite time performance of model predictive control. *Systems & Control Letters*, 70, 60–68.

Christophersen, F., Kvasnica, Jones, C., and Morari, M. (2007). Efficient evaluation of piecewise control laws defined over a large number of polyhedra. In *Proceedings of the European Control Conference.*

Cseko, L., Kvasnica, M., and Lantos, B. (2015). Explicit MPC-based RBF neural network controller design with discrete-time actual kalman filter for semiactive suspension. *IEEE Transactions on Control Systems Technology*, 23(5), 1736–1753.

García, C.E., Prett, D.M., and Morari, M. (1989). Model predictive control: Theory and practice – a survey. *Automatica*, 25(3), 335 – 348. doi:https://doi.org/10.1016/0005-1098(89)90002-2.

Geyer, T., Torrisi, F.D., and Morari, M. (2008). Optimal complexity reduction of polyhedral piecewise affine systems. *Automatica*, 44(7), 1728 – 1740.

Gupta, A., Bhartiya, S., and Nataraj, P. (2011). A novel approach to multiparametric quadratic programming. *Automatica*, 47(9), 2112 – 2117. doi:10.1016/j.automatica.2011.06.019.

Herceg, M., Kvasnica, M., Jones, C., and Morari, M. (2013a). Multi-Parametric Toolbox 3.0. In *ECC*, 502–510. Zürich, Switzerland. `http://control.ee.ethz.ch/~mpt`.

Herceg, M., Mariéthoz, S., and Morari, M. (2013b). Evaluation of piecewise affine control law via graph traversal. In *ECC*, 3083–3088. Zurich, Switzerland.

Holaza, J., Takács, B., Kvasnica, M., and Di Cairano, S. (2015). Nearly optimal simple explicit MPC controllers with stability and feasibility guarantees. *Optimal Control Applications and Methods*, 35(6). doi:10.1002/oca.2131.

Jones, C. and Morari, M. (2010). Polytopic approximation of explicit model predictive controllers. *IEEE Transactions on Automatic Control*, 55. doi:10.1109/TAC.2010.2047437.

Jost, M., Pannocchia, G., and Mönnigmann, M. (2017). Accelerating linear model predictive control by constraint removal. *European Journal of Control*, 35, 42–49.

Kvasnica, M., Bakaráč, P., and Klaučo, M. (2019). Complexity reduction in explicit MPC: A reachability approach. *Systems & Control Letters*, 124, 19–26. doi:10.1016/j.sysconle.2018.12.002.

Kvasnica, M., Takács, B., Holaza, J., and Di Cairano, S. (2015a). On region-free explicit model predictive control. In *54rd IEEE CDC*, volume 54, 3669–3674. Osaka, Japan.

Kvasnica, M., Takács, B., Holaza, J., and Ingole, D. (2015b). Reachability analysis and control synthesis for uncertain linear systems in MPT. In *Proc. of IFAC ROCOND*, 8, 302–307. Elsevier, Bratislava, Slovak Republic.

Mönnigmann, M. (2019). On the structure of the set of active sets in constrained linear quadratic regulation. *Automatica*, 106, 61–69.

Oberdieck, R. and Pistikopoulos, E.N. (2015). Explicit hybrid model-predictive control: The exact solution. *Automatica*, 58, 152 – 159.

Oravec, J., Blažek, S., Kvasnica, M., and Cairano, S.D. (2013). Polygonic representation of explicit model predictive control. In *IEEE CDC*, 6422–6427. Florence, Italy.

Rauová, I., Valo, R., Kvasnica, M., and Fikar, M. (2011). Real-time model predictive control of a fan heater via PLC. In *Int. Conf. on Process Control*, 388–393. Tatranská Lomnica, Slovakia.

Tøndel, P., Johansen, T.A., and Bemporad, A. (2002). Evaluation of piecewise affine control via binary search tree.

Wang, Z. and Ong, C. (2014). Distributed MPC of constrained linear systems with time-varying terminal sets. *Systems & Control Letters*, 70, 60–68.

Willner, L. (1967). On Parametric Linear Programming. *SIAM Journal on Applied Mathematics*, 15(5), 1253–1257.