Combinatorial multi-parametric quadratic programming with saturation matrix based pruning

Christian Feller$^{1,2}$, Tor Arne Johansen$^2$, Sorin Olaru$^3$

Abstract—The goal of multi-parametric quadratic programming (mpQP) is to compute analytic solutions to parameter-dependent constrained optimization problems, e.g., in the context of explicit linear MPC. We propose an improved combinatorial mpQP algorithm based on a saturation matrix pruning criterion which uses geometric properties of the constraint polyhedron for excluding infeasible constraint combinations from the candidate active set enumeration. Performance improvements are discussed for both practical and random example problems from the area of explicit linear MPC.

I. INTRODUCTION

Multi-parametric programming (mpP) techniques allow to compute explicit solutions to parameter-dependent constrained optimization problems. These explicit (or analytic) solutions can be computed a priori and allow to replace computationally demanding on-line optimization procedures, whose application may be problematic in real-time applications or in the presence of hardware restrictions. In particular, mpP allows to derive explicit solutions to different types of open-loop optimal control problems as they occur in the area of model predictive control (MPC). Using explicit mpP solutions, the on-line optimization involved in the MPC approach can be moved off-line, which extends the possible application area of receding horizon control. In this paper, we will focus on strictly convex multi-parametric quadratic programming (mpQP) problems, which are related to linear MPC based on a quadratic cost function, i.e., the constrained finite-horizon LQR problem. In general, the solution has the form of a piecewise affine function over a polyhedral partition of the parameter space into so-called critical regions, where each region corresponds to a set of optimal active constraints.

Most of the mpQP algorithms reported in the literature are based on geometric methods and apply recursive exploration strategies in order to identify all critical regions of the explicit solution. In [1], the authors propose a simple algorithm that subdivides the parameter space into polyhedral regions by reversing recursively the facet-defining hyperplanes of all previously identified regions. Unfortunately, this approach introduces artificial cuts in the parameter space, which can result in unnecessary and redundant partitioning. More efficient exploration strategies were presented in [2], [3], and [4], based on the assumption that for each facet of a critical region there exists only one neighboring critical region that is adjacent to this facet. The parameter space is then explored iteratively by stepping over all the facets of already identified regions and solving the mpQP problem for new parameter vectors, or by examining the type of the facet-defining hyperplanes, respectively. In [5], the authors propose an algorithm that combines the approaches from [1] and [3] in order to handle situations in which this facet-to-facet property does not hold. Moreover, additional geometric approaches were proposed in [6] and [7]: in [6], the authors consider all possible configurations of the constraint polyhedron in order to induce a partition of the input space, while a parametrized polyhedra approach in the combined (input+parameter) space is used in [7]. Then, both approaches use projection techniques to construct all full-dimensional critical regions of the parameter space partition.

Recently, a new combinatorial mpQP approach that is based on an implicit enumeration of all possible constraint combinations in form of candidate active sets has been proposed in [8]. In this approach, the candidate active sets are tested for optimality by solving a linear program (LP) and pruning criteria are used to reduce the number of constraint combinations that need to be considered in the enumeration process. In contrast to most of the existing geometric approaches, the combinatorial approach does not rely on an explicit geometric exploration strategy. Moreover, it inherently allows easy identification and handling of degeneracies while guaranteeing partitioning of the complete parameter space. However, despite the pruning criteria, one disadvantage of the approach is the combinatorial complexity with respect to the number of possible candidate active sets, i.e., the number of LPs that need to be solved in the solution process.

In this paper, we propose an improved combinatorial mpQP algorithm that uses some of the underlying geometric properties of the involved problem constraints in order to increase the efficiency of the combinatorial active set enumeration. The algorithm is based on a new pruning mechanism that allows to detect infeasible combinations of active constraints by a simple row sum check on the so-called saturation matrix of the constraint polyhedron. This allows to reduce the computational complexity of the combinatorial mpQP approach while preserving its structural advantages. Two benchmark problems as well as a selection of random problems are used to demonstrate the achieved performance improvements.
II. Multi-parametric Quadratic Programming

The relevance of parametric programming to control theory results from the fact that it is possible to express special classes of constrained optimal control problems as multi-parametric programs [1]. In the following, we will focus on standard mpQP problems of the form

\[
V^*_z(x) = \min_{z} \frac{1}{2} z^T H z - G^T z, \quad \text{s. t.} \quad G z \leq W + S x, \tag{1a}
\]

\[
\text{where } z \in \mathbb{R}^m \text{ and } x \in \mathbb{R}^n \text{ denote the vectors of optimization variables and parameters, and } H \in \mathbb{R}^{m \times m}, G \in \mathbb{R}^{q \times m}, W \in \mathbb{R}^q, S \in \mathbb{R}^{q \times n} \text{ are real matrices. We assume that all constraints on } x \text{ are included in (1b) and that the problem is strictly convex, i.e., that } H \succ 0. \text{ Without loss of generality, we further assume that (1b) does not contain any redundant constraints.}
\]

A. Analytic solutions to mpQP problems

As shown in [1] and [3], we can solve (1) by applying the Karush-Kuhn-Tucker (KKT) conditions, which for this problem are given by

\[
H z + G^T \lambda = 0, \quad \lambda \in \mathbb{R}^q, \tag{2a}
\]

\[
\lambda^i (G^i z - W^i - S^i x) = 0, \quad i = 1, \ldots, q, \tag{2b}
\]

\[
\lambda \geq 0, \tag{2c}
\]

\[
G z \leq W + S x, \tag{2d}
\]

where the superscript index \(i\) denotes the \(i\)th row of a matrix or vector and \(\lambda\) refers to the vector of Lagrangian multipliers. Before proceeding further in the description of the analytic solution, we introduce some definitions [1], [3].

Definition 1 (Optimal active set) Let \(z^*(x)\) be the optimal solution to (1) for a given parameter vector \(x\) and let \(Q := \{1, \ldots, q\}\) denote the set containing the indices of all constraints in (1). Then, the optimal active set \(A^*(x)\) is defined as the index set of active constraints at the optimum:

\[
A^*(x) := \left\{ i \in Q : G^i z^*(x) - W^i - S^i x = 0 \right\}. \tag{3}
\]

Definition 2 (Weak (strong) activity) We define as weakly (strongly) active constraint an active constraint with an associated Lagrange multiplier \(\lambda^i = 0 (\lambda^i > 0)\).

Assuming that we know the optimal active set \(A^*(x)\) for a given \(x\), we can form submatrices \(G^{A^*(x)}, W^{A^*(x)}, S^{A^*(x)}\) that contain the constraints associated to the indices in \(A^*(x)\). In the following, we drop the explicit parametrization of \(A^*(x)\) for the ease of notation.

Definition 3 (LICQ) For an index set \(A \subseteq Q\), the linear independence constraint qualification (LICQ) holds if the gradients of the corresponding constraints are linearly independent, i.e., if \(G^A\) has full row rank.

When assuming that LICQ holds for a given optimal active set \(A\), we can use the first two equations of the KKT conditions to derive the parameter-dependent optimizer

\[
z_A(x) = H^{-1}(G^A)^T H^{-1}_{G^A} (W^A + S^A x) \tag{4}
\]

for a fixed parameter vector \(x\), where existence of

\[
H^{-1}_{G^A} := (G^A H^{-1}(G^A)^T)^{-1} \tag{5}
\]

is guaranteed since \(H \succ 0\) and LICQ holds [1], [3]. Moreover, the two remaining KKT inequality conditions characterize the so-called critical region \(CR_A\) in which the solution (4) remains optimal when varying the parameter \(x\):

\[
-H_{G^A}^{-1} (W^A + S^A x) \geq 0 \tag{6a}
\]

\[
GH^{-1}(G^A)^T H^{-1}_{G^A} (W^A + S^A x) \leq W + S x. \tag{6b}
\]

This polyhedral region in the state space is the largest set of parameters for which the combination of active constraints at the optimizer remains unchanged, i.e., for which \(A\) remains the optimal active set. Thus, by identifying all optimal active sets \(A_i\), the parameter space is implicitly partitioned into several critical regions \(CR_{A_i}\), and the optimizer can be represented as a continuous piecewise affine function of the parameter \(x\) [1], [3].

B. Degeneracies in mpQP

So far, we have assumed that LICQ holds, i.e., that the rows of the constraint matrix \(G^A\) are linearly independent. However, in situations for which this assumption is not true, \(H^{-1}_{G^A}\) does not exist and the vector of Lagrangian multipliers \(\lambda^A\) might not be uniquely defined. If such a case of primal degeneracy occurs inside a full-dimensional critical region, the KKT conditions will only lead to a polyhedron expressed in the \((\lambda, x)\)-space, and further methods have to be applied in order to obtain a representation of \(z_A\) and \(CR_A\). Two different methods on how to deal with such situations have been proposed in the literature: (i) take a full-rank subset of the active constraints and proceed with this new optimal active set, or (ii) use a projection algorithm in the \((\lambda, x)\)-space to obtain the full-dimensional critical region. The first method, which is used in [1] and also in the Multi-Parametric Toolbox (MPT) [9], is rather easy to implement. However, it is not suited for all mpQP algorithms and can lead to overlapping or even missing critical regions. The second method, which may be computationally more demanding, is used in [4] in order to circumvent the problems of the first one. As will be discussed in the next section, the procedure of considering all full-rank subsets of an optimal active set with LICQ failure is inherently employed in the combinatorial mpQP approach.

C. Combinatorial mpQP

The goal in multi-parametric programming is usually to identify all optimal active sets corresponding to full-dimensional critical regions, as these implicitly define the explicit solution to the mpQP problem (1). Whereas most of the existing geometric mpQP algorithms construct the
solution by identifying all full-dimensional critical regions in a recursive parameter space exploration, the combinatorial mpQp algorithm presented in [8] operates directly on the level of possible optimal active sets. The main idea of the approach is the implicit enumeration of all possible combinations of active constraints, which we will shortly summarize in the following.

Consider again the set \( \mathcal{Q} = \{1, \ldots, q\} \) referring to the constraint indices in (1b). Then, the active set \( \mathcal{A}(z, x) \) can be described as

\[
\mathcal{A}(z, x) := \{i \in \mathcal{Q} \mid G^i z - W^i - S^i x = 0\},
\]

while the corresponding set of inactive constraints \( \mathcal{J}(z, x) \) is given by the set difference of \( \mathcal{Q} \) and \( \mathcal{A} \)

\[
\mathcal{J}(z, x) := \mathcal{Q} \setminus \mathcal{A}(z, x).
\]

As pointed out in [8], only a maximum of \( m \) linearly independent constraints can be strongly active at the optimal solution of a quadratic program with \( m \) optimization variables and \( q \geq m \) inequality constraints. Hence, all possible optimal active sets are included in the set

\[
\mathcal{P}'(\mathcal{Q}) := \left\{ \mathcal{A}_1 = \emptyset, \mathcal{A}_2 = \{1\}, \ldots, \mathcal{A}_{q+1} = \{q\}, \ldots, \mathcal{A}_{q+2} = \{1, 2\}, \ldots, \mathcal{A}_{n\mathcal{Q}} = \{q - m + 1, \ldots, q\} \right\},
\]

which is a subset of the power set \( \mathcal{P}(\mathcal{Q}) \) and consists of

\[
n'_{\mathcal{Q}} = \sum_{i=0}^m \binom{q}{i} \leq 2^q \text{ index sets. Here, } q \text{ and } m \text{ are defined as } q = \max\{m, q\} \text{ and } m = \min\{m, q\}, \text{ respectively.}
\]

Since the solution to (1) is implicitly defined if all optimal active sets are known, it can be constructed by checking for each candidate set \( \mathcal{A}_i \in \mathcal{P}'(\mathcal{Q}) \) whether there exists a feasible point in the parameter space for which \( \mathcal{A}_i \) is the optimal active set. Clearly, this is equivalent to checking if there exist vectors \( z, x, \lambda \) for which the KKT conditions (2) are satisfied. Adapting this idea, the authors of [8] suggest to choose candidate active sets \( \mathcal{A}_i \in \mathcal{P}'(\mathcal{Q}) \) in the order of increasing cardinality and use the linear program

\[
\begin{align*}
\max_{z, x, \lambda} & \quad t \\
\text{s. t.} & \quad te_1 \leq \lambda A_1, \quad te_2 \leq sJ_i \\
& \quad t \geq 0, \quad \lambda A_1 \geq 0, \quad sJ_i \geq 0 \\
& \quad Hz + (G^A_1)^T \lambda A_1 = 0 \\
& \quad G^A_1 z - S^A_1 x - W^A_1 = 0 \\
& \quad G^J_1 z - S^J_1 x - W^J_1 + sJ_i = 0
\end{align*}
\]

(10a) to check whether \( \mathcal{A}_i \) is indeed an optimal active set. Here, in addition to the already introduced variables and matrices, \( t \) is a scalar optimization variable and \( e_1 = [1, \ldots, 1]^T, e_2 = [1, \ldots, 1]^T \) are vectors of appropriate sizes corresponding to the vector of Lagrangian multipliers \( \lambda \), and the vector of slack variables \( s^J_i \), respectively. Clearly, if the LP (10) has a feasible solution, then there exist feasible \( z_{A_i}, x_{A_i}, \lambda^A_i, s^J_i \), satisfying the KKT conditions, and \( \mathcal{A}_i \) is an optimal active set. In this case, (4) and (6) can be used to compute the affine solution and the corresponding critical region. While the resulting regions will be unique for \( t > 0 \), some constraints will be weakly active for \( t = 0 \) \( (\lambda^A \lor s^J_i = 0) \), which may lead to non-full-dimensional or overlapping critical regions (see [8], [10] for details). On the other hand, infeasibility of the LP (10) implies that \( \mathcal{A}_i \) is not an optimal active set, and \( z_{A_i}, CR_{A_i} \) need not be computed. Moreover, if the LP (10) is also infeasible when only feasibility constraints are considered, i.e., when all constraints related to \( \lambda^A_i \) are discarded, then \( \mathcal{A}_i \) represents an infeasible combination of active constraints. Note that it may be useful to impose an artificial, possibly very large, upper bound on \( t \) in order to get bounded LP solutions. Since checking the LP (10) for every candidate set by enumerating \( \mathcal{P}'(\mathcal{Q}) \) explicitly may be impractical even for relatively small values of \( m \) and \( q \), the authors of [8] propose in addition the following pruning criterion which reduces the number of candidate sets and makes the enumeration of \( \mathcal{P}'(\mathcal{Q}) \) implicit.

**Criterion 1 (Pruning of candidate active sets)**

If a candidate active set \( \mathcal{A}_i \in \mathcal{P}'(\mathcal{Q}) \)

(i) leads to violation of the LICQ condition, or

(ii) represents an infeasible constraint combination,

then \( \mathcal{A}_i \) and all its supersets can be excluded from further consideration in the enumeration of \( \mathcal{P}'(\mathcal{Q}) \).

The first of these pruning conditions becomes apparent when considering the structure of the combinatorial enumeration process. Since the algorithm proceeds through the elements of \( \mathcal{P}'(\mathcal{Q}) \) in order of increasing cardinality, all full-rank subsets of a candidate set with LICQ failure will be covered automatically. Thus, the method of computing the control law and the critical regions for cases of LICQ failure in full-dimensional regions by considering all full-rank subsets is, in a sense, inherently included in the combinatorial approach. Therefore, every candidate active set \( \mathcal{A}_i \) for which \( G^A_1 \) does not have full row rank can be discarded, and, since LICQ will of course also be violated for all \( \mathcal{A}_j \supset \mathcal{A}_i \), the same holds for all its supersets. In such a case, however, there will be weakly active constraints and overlapping critical regions may occur. This point is discussed in more detail in [10], where also an algorithm modification is presented that allows to avoid overlapping critical regions by merging them into their convex union. The second pruning criterion follows directly from the fact that an infeasible system of equality and inequality constraints, i.e., here the LP (10), will still be infeasible when some of the inequalities are treated as equalities [8].

Using these results, the combinatorial mpQP algorithm proposed in [8] can be summarized as follows.

**Algorithm 1 Combinatorial mpQP algorithm from [8]**

1. choose \( \mathcal{A}_i \in \mathcal{P}'(\mathcal{Q}) \) in order of increasing cardinality;
2. if \( \mathcal{A}_i \) not pruned and \( G^A_1 \) has full row rank, solve (10) \( \lor \) if feasible, use (4) and (6) to construct \( z_{A_i}, CR_{A_i} \) \( \lor \) if infeasible, solve (10) without optimality constraints \( \lor \) if infeasible, add all \( \mathcal{A}_j \supset \mathcal{A}_i \) to the pruned sets;
3. return to 1. until the whole set \( \mathcal{P}'(\mathcal{Q}) \) is explored.
A graphical illustration of the combinatorial enumeration strategy and the involved pruning process is given in Fig. 1 in form of a tree diagram. Note that it is important to prune infeasible constraint combinations globally in the whole tree, since otherwise the efficiency of the enumeration would be suboptimal and dependent on the arrangement of the constraints in the matrices $G$, $W$, and $S$. By investigating the performance of Algorithm 1 for a selection of random mpQP problems, the authors of [8] showed that the proposed pruning criteria considerably reduce the computational complexity of the combinatorial mpQP approach.

III. AN IMPROVED COMBINATORIAL MPQP ALGORITHM

In the previous section we have reviewed the combinatorial mpQP approach proposed in [8]. Advantages of this approach are that no geometric exploration strategy is needed, that degeneracies are detected and handled easily, and that, by considering all possible constraint combinations as candidate active sets, the complete parameter space is explored implicitly and without introducing artificial cuts. However, the main disadvantage of the approach is given by its combinatorial complexity. Although the described pruning criterion usually reduces the number of candidate sets considerably, the computational cost of the algorithm, i.e., the number of optimization problems solved, may still grow exponentially with $n$ and $q$. In the following, we will present an improved combinatorial mpQP algorithm that is also based on the implicit enumeration approach but uses a simple matrix check to exclude infeasible constraint combinations. The underlying idea is to use some of the geometric properties of the mpQP problem constraints in order to increase the efficiency of the combinatorial enumeration process.

A. An infeasibility check based on the constraint polyhedron geometry – saturation matrix pruning

Consider the mpQP problem (1). For an arbitrary parameter vector $x$, the constraints on the optimization variable $z$ are given by the parametrized constraint polyhedron

$$P(x) = \{ z \in \mathbb{R}^m \mid Gz \leq W + Sx \}, \quad x \in \mathbb{R}^n. \quad (11)$$

In [7], the authors showed how the parametrized vertices of $P(x)$ can be used to construct a hierarchical partition of the parameter space. Here, however, we want to exploit the fact that the constraints (1) can also be represented as a non-parametrized polyhedron in the augmented (variable+parameter) space:

$$\tilde{P} = \left\{ [z, x] \in \mathbb{R}^{m+n} \left| \begin{bmatrix} G & -S \end{bmatrix} \begin{bmatrix} z \\ x \end{bmatrix} \leq W \right. \right\}. \quad (12)$$

Our approach is based on the fact that a combination of active constraints will not be feasible if it is not active at any vertex of the constraint polyhedron $\tilde{P}$, which is formulated and proven in the following theorem:

**Theorem 1 (Infeasibility of candidate active sets)**

Let $A_i \in \mathcal{P}(Q)$ be a candidate active set related to the constraint matrices $G$, $W$, and $S$ of problem (1). If and only if the constraints associated to $A_i$ are not active together at any vertex $v_k \in \mathbb{R}^{m+n}$ of $\tilde{P}$, i.e.,

$$\nexists v_k \ s.t. \ [G^{A_i} - S^{A_i}] v_k - W^{A_i} = 0 \quad (13)$$

for $k = 1, \ldots, n_v$, where $n_v$ denotes the number of vertices of $\tilde{P}$, then $A_i$ represents an infeasible combination of active constraints.

**Proof:** If the constraint combination corresponding to $A_i$ is not active at any vertex of $\tilde{P}$, it is also not active at any facet of $\tilde{P}$. Hence, the constraint hyperplanes related to $A_i$ do not intersect in the feasible part of the augmented space $\mathbb{R}^{m+n}$, which means that $A_i$ does not represent a feasible combination of active constraints. The reverse direction follows trivially.

Thus, we can obtain knowledge about the infeasibility of candidate active sets by computing all vertices of $\tilde{P}$ and checking whether any of them saturate the corresponding constraints. Note, however, that Theorem 1 can only provide a sufficient condition for infeasibility of (10) since optimality is not taken into account. In order to exploit Theorem 1 in the combinatorial enumeration process, we make use of the saturation matrix $S$ of the constraint polyhedron $\tilde{P}$:

**Definition 4 (Saturation matrix $S$ of $\tilde{P}$ [11])**

As saturation matrix of the constraint polyhedron $\tilde{P}$ we denote the binary matrix $S \in \{0,1\}^{n_v \times q}$ defined as

$$S_{kj} = \begin{cases} 1 & \text{if } [G^{j} - S^{j}] v_k - W^j = 0 \\ 0 & \text{if } [G^{j} - S^{j}] v_k - W^j \neq 0 \end{cases}, \quad (14)$$

where $k = 1, \ldots, n_v$, $j = 1, \ldots, q$. Hence, the entry $S_{kj}$ indicates whether constraint $j$ is active at vertex $v_k$ of $\tilde{P}$.

Combining Theorem 1 and Definition 4, we can conclude that a candidate set $A_i$ can only represent a feasible combination of active constraints, if and only if there exists at least one row in the saturation matrix $S$ that contains only nonzero elements in the columns related to the indices in $A_i$. Hence, we can formulate the following corollary which allows to identify infeasible candidate active sets by performing a simple row sum check on $S$.
Corollary 1 (Infeasibility condition for $A_i$)
Let $A_i \in P'(Q)$ be a candidate active set and let $S$ denote the saturation matrix of the constraint polyhedron $\tilde{P}$. Then, a necessary and sufficient condition for the infeasibility of $A_i$ is given by
\[
\sum_{j \in A_i} S_{kj} < |A_i| \quad \forall k \in \{1, \ldots, n_v\},
\]
where $|A_i|$ denotes the cardinality of $A_i$, i.e., the number of constraint indices in the candidate active set.

A simple example that demonstrates the use of the saturation matrix is given in Fig. 2. The constraint polyhedron $\tilde{P}$ is assumed to be a cuboid in $\mathbb{R}^3$, which is formed by six intersecting hyperplanes $h_j$ and eight vertices $v_k$. It is obvious that the sets $\{1, 3\}$, $\{2, 4\}$, and $\{5, 6\}$ represent infeasible combinations of active constraints, as they refer to opposing constraint hyperplanes. Since the respective columns of the saturation matrix $S$ do not have coinciding 1-elements in the same row, these infeasible candidate sets can also be detected by making use of Corollary 1. Of course, this is only a very simple example. However, Corollary 1 does hold in general and can also be used for more complex constraint polyhedron configurations.

Based on this saturation matrix pruning criterion, we propose the following combinatorial mpQP algorithm:

Algorithm 2 New combinatorial mpQP algorithm
1. compute the saturation matrix $S$;
2. choose $A_i \in P'(Q)$ in order of increasing cardinality;
3. if $G^{A_i}$ has full row rank, check condition (15)
   - if $A_i$ is identified as infeasible, go to 4;
   - else, try to solve LP (10)
     - if feasible, use (4) and (6) to construct $z_{A_i}$, $CR_{A_i}$;
     - if infeasible, go to 4;
4. return to 2, until the whole set $P'(Q)$ is explored.

As in Algorithm 1, the enumeration proceeds through $P'(Q)$ in order of increasing cardinality and exploits Criterion 1 to reduce the number of candidate active sets. However, by making use of Corollary 1, all candidate sets that are related to infeasible combinations of active constraints can be excluded by simply checking the row sums of the saturation matrix, which considerably reduces the number of LPs that need to be solved. Furthermore, since infeasibility of the LP (10) will now in all cases only arise from suboptimality, it is obviously not worthwhile to solve the modified second LP and perform an additional pruning of infeasible candidate sets. This results in a further reduction in the number of LPs and eliminates the need for an explicit pruning mechanism. Moreover, in contrast to the algorithm proposed in [8], each candidate set $A_i$ can now be checked independently from all other sets $A_j \in P'(Q)$, which would allow easy parallelization of the enumeration procedure in Algorithm 2. However, one disadvantage of the approach is that constructing and handling the saturation matrix $S$ may become computationally demanding with increasing complexity of the constraint polyhedron $\tilde{P}$. In such situations it might be helpful to minimize the number of saturation matrix checks by pruning infeasible constraint combinations explicitly. Note that further complexity reduction in the combinatorial mpQP approach could be achieved by exploiting symmetries that may exist in the mpQP problem formulation, see [10] for more details.

IV. NUMERICAL EXAMPLES
In order to compare Algorithm 2 with the combinatorial mpQP algorithm from [8], we implemented both algorithms in MATLAB and performed benchmark tests for both practical examples and random problems from the area of linear MPC. In our implementation of Algorithm 2, the extreme point solver included in the MPTP [9] is used to compute the vertices of the constraint polyhedron $\tilde{P}$, which are needed for the construction of the saturation matrix $S$. All computations were performed on a 3 GHz Dual Core PC with 8 GB RAM, running MATLAB 7.11 and MPT 2.6.3. For the following results, MATLAB’s “linprog” LP solver was used for solving the LPs of type (10).

Example 1. Considered is the discrete-time double integrator system discussed in [3] with a discretization time of $T_s = 0.3$ s. For this system, the linear MPC open-loop optimal control problems were formulated using a quadratic cost function with the weight matrices $Q = \text{diag}(1, 0)$, $R = 1$, $P = P_{LQR}$ and the input and state constraints $|u| \leq 1$, $|x_2| \leq 0.8$, $x(t + N) \in \Omega_{LQR}$. Here, $P_{LQR}$ has been computed from the algebraic Riccati equation and $\Omega_{LQR}$ is the positively invariant set of the LQR.

Example 2. As a more specific second example, we considered the 3-DOF laboratory model helicopter described in [3], which is given in form of a linear state space model involving six system states and two inputs. The MPC problems were formulated using $Q = \text{diag}(100, 100, 10, 10, 400, 200)$, $R = I_{2 \times 2}$, $P = P_{LQR}$, and the constraints $-1 \leq u_i \leq 3$, $i = 1, 2$.

The numerical results for the two examples are presented in Table I and Table II, respectively. Here, $N$ denotes the horizon in the MPC problem formulation, $n_{c}$ the number of critical regions in the state space partition, and $n_{l, P}$ the number of optimization problems of type (10) that were solved in the combinatorial enumeration process. Furthermore, $n_v$ and $\dim_P$ refer to the number of vertices and the
TABLE I
RESULTS FOR THE DOUBLE INTEGRATOR EXAMPLE.

<table>
<thead>
<tr>
<th>N</th>
<th>n_r</th>
<th>n_r^{max}</th>
<th>n_LP,n_{Alg.1}</th>
<th>n_LP,n_{Alg.2}</th>
<th>n_v (dim_P)</th>
<th>t_2/t_1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>11</td>
<td>13</td>
<td>13</td>
<td>13</td>
<td>28 (3)</td>
<td>1.33</td>
</tr>
<tr>
<td>2</td>
<td>33</td>
<td>172</td>
<td>131</td>
<td>77</td>
<td>58 (4)</td>
<td>0.62</td>
</tr>
<tr>
<td>3</td>
<td>57</td>
<td>1794</td>
<td>651</td>
<td>383</td>
<td>138 (5)</td>
<td>0.60</td>
</tr>
<tr>
<td>4</td>
<td>83</td>
<td>17902</td>
<td>6695</td>
<td>1733</td>
<td>326 (6)</td>
<td>0.26</td>
</tr>
<tr>
<td>5</td>
<td>111</td>
<td>174437</td>
<td>30717</td>
<td>7569</td>
<td>772 (7)</td>
<td>0.24</td>
</tr>
<tr>
<td>6</td>
<td>135</td>
<td>1676116</td>
<td>263503</td>
<td>32017</td>
<td>1824 (8)</td>
<td>0.12</td>
</tr>
</tbody>
</table>

TABLE II
RESULTS FOR THE HELICOPTER EXAMPLE.

<table>
<thead>
<tr>
<th>N</th>
<th>n_r</th>
<th>n_r^{max}</th>
<th>n_LP,n_{Alg.1}</th>
<th>n_LP,n_{Alg.2}</th>
<th>n_v (dim_P)</th>
<th>t_2/t_1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9</td>
<td>11</td>
<td>9</td>
<td>9</td>
<td>256 (8)</td>
<td>1.12</td>
</tr>
<tr>
<td>2</td>
<td>81</td>
<td>163</td>
<td>81</td>
<td>81</td>
<td>1024 (10)</td>
<td>1.05</td>
</tr>
<tr>
<td>3</td>
<td>729</td>
<td>2510</td>
<td>729</td>
<td>729</td>
<td>4096 (12)</td>
<td>1.06</td>
</tr>
<tr>
<td>4</td>
<td>4461</td>
<td>39203</td>
<td>8589</td>
<td>6561</td>
<td>16384 (14)</td>
<td>0.94</td>
</tr>
<tr>
<td>5</td>
<td>18413</td>
<td>616666</td>
<td>99119</td>
<td>59049</td>
<td>65536 (16)</td>
<td>0.88</td>
</tr>
</tbody>
</table>

which may help to establish combinatorial mpQP as a real alternative to existing geometric approaches. More detailed results and computation times for different LP solvers can be found in [10].

V. CONCLUSION

In this paper, we have proposed an improved combinatorial mpQP algorithm that uses a saturation matrix pruning criterion to exclude infeasible candidate active sets. This allows to speed up the combinatorial enumeration process and eliminates the need for an explicit pruning mechanism. The results show the benefit of using geometric properties of the mpQP problem constraints for improving the efficiency of the combinatorial mpQP approach. An interesting future work could be to exploit the decoupling character of the proposed pruning mechanism in the framework of a parallelized combinatorial mpQP algorithm.

ACKNOWLEDGEMENTS

We would like to thank the authors of [8] for giving us access to the MATLAB code of their algorithm.

REFERENCES