Proceedings of the
47th IEEE Conference on Decision and Control
Cancun, Mexico, Dec. 9-11, 2008

A Dual Gradient Projection Quadratic Programming Algorithm Tailored for Model Predictive Control

Daniel Axehill and Anders Hansson

Abstract— **The objective of this work is to derive a QP algorithm tailored for MPC. More specifically, the primary target application is MPC for discrete-time hybrid systems. A desired property of the algorithm is that warm starts should be possible to perform efficiently. This property is very important for on-line linear MPC, and it is crucial in branch and bound for hybrid MPC. In this paper, a dual active set-like QP method was chosen because of its warm start properties. A drawback with classical active set methods is that they often require many iterations in order to find the active set in optimum. Gradient projection methods are methods known to be able to identify this active set very fast and such a method was therefore chosen in this work. The gradient projection method was applied to the dual QP problem and it was tailored for the MPC application. Results from numerical experiments indicate that the performance of the new algorithm is very good, both for linear MPC as well as for hybrid MPC. It is also noticed that the number of QP iterations is significantly reduced compared to classical active set methods.**

I. Introduction

The main motivation for this work is control of discrete-time hybrid systems in Mixed Logical Dynamical (MLD) form, [1], using Model Predictive Control (MPC). In the basic linear setup, the MPC problem can be cast in the form of a Quadratic Programming (QP) problem. When a hybrid system is to be controlled, the corresponding optimization problem is changed from a QP problem into a Mixed Integer Quadratic Programming (MIQP) problem, and hence, the term Mixed Integer Predictive Control (MIPC) is sometimes used. The MIQP problem is usually solved using branch and bound, where sometimes a large number of QP problems have to be solved. In the MIPC application, these QP problems are in the form of linear MPC problems. The focus in this work is to solve linear MPC problems efficiently, especially in the case when several similar problems are solved consecutively. This property is not only useful for hybrid MPC, but also for linear MPC where the QP problem is resolved in each time instant, and in (smooth) nonlinear MPC where solvers based on Sequential Quadratic Programming (SQP) can be used. Also in SQP, several similar linear MPC problems have to be solved before the solution to the original nonlinear problem is found.

In this work, the problem structure is utilized in two ways in order to improve performance. First, since the difference between the optimization problems to be solved often is small (especially in branch and bound), the solution from a previously solved problem is reused as a starting point in a new problem. This procedure is often called a warm start of the solver. In previous work by the authors, [2], [3], [4], work from several other researchers, *e.g.*, [5], [6], [7], [8], working with QP methods has been summarized. Based on their work and experience, the conclusion was drawn that a dual active set method is the best choice for the MIPC application, where numerous *similar* QP problems have to be solved in order to solve the original MIQP problem. Early work on dual active set solvers can be found in, *e.g.*, [9]. A more recent method is found in [5], which has been refined in, *e.g.*, [10]. Second, in the dual QP solver to be presented, the Karush-Kuhn-Tucker (KKT) system is solved using a Riccati recursion. This has previously been done in primal active set QP solvers, *e.g.*, [11], and in interior point solvers, *e.g.*, [12]. The material presented in this paper is based on an extension of the work presented in [2] and in [3]. In that work, a dual active set QP solver tailored for MIPC built on a classical active set method is presented. In this new paper, the classical active set method used in [2], [3] has been replaced by a gradient projection method which has the potential to give better performance, especially for problems with many active constraints at the optimum, [13]. Early work on gradient projection methods can be found in [14] and in [15]. Many articles have been written about gradient projection. The method is used for optimal control in, *e.g.*, [13] and [16]. In [17], a method is presented where basic gradient projection iterations are combined with conjugated gradient iterations.

The contribution in this paper is a gradient projection algorithm working on the dual MPC problem. Gradient projection methods have previously been recognized as very appropriate for optimal control problems with simple constraints, but not for problems with general constraints. In this paper, it is shown that the dual MPC problem always get simple bound constraints, independently of the primal constraints. Consequently, the gradient projection method is expected to be efficient when applied to the dual problem. This is confirmed by numerical experiments in this work.

Because of the limited space in this paper, only the linear MPC problem is considered in detail. The QP relaxations used in branch and bound for MIPC will also be of linear MPC type, but the details are left out. A compact introduction to the basics of MIQP can be found in [3]. For a complete treatment, see [4] or Paper D in [18] (available on-line).

A. Notation

In this paper, $\mathbb{S}^n_{++}$ ($\mathbb{S}^n_{+}$) denotes the set of symmetric positive (semi) definite matrices with n rows. Furthermore, let $\mathbb{Z}$ be the set of integers, $\mathbb{Z}_{++}$ be the set of positive (non-zero) integers, and $\mathbb{Z}_{i,j} = \{i, i+1, \ldots, j\}$.

D. Axehill and A. Hansson are with the Division of Automatic Control, Linköping University, SE-581 83 Linköping, Sweden, {daniel,hansson}@isy.liu.se.

B. Problem Definition

There exist different equivalent optimization problem formulations of the linear MPC problem. For example, it can be written as a QP problem with only control signals as free variables, or it can be written as a QP problem where control signals, states and control errors all are free variables. The derivations of both formulations for a general linear MPC problem can be found in [19], or in [18, pp. 65–67]. In this paper, it will be seen that the second alternative is advantageous from a computational point of view and this formulation is used in this work. Using this second form, the MPC optimization problem for a linear time-variant system can be written as

$$\begin{aligned} \underset{\mathsf{x},\mathsf{u},\mathsf{e}}{\text{minimize}} \quad & \frac{1}{2}\sum_{t=0}^{N-1} e^T(t)Q_e(t)e(t) + u^T(t)Q_u(t)u(t) + \\ & + \frac{1}{2}e^T(N)Q_e(N)e(N) \\ \text{subject to} \quad & x(0) = x_0 \\ & x(t+1) = A(t)x(t) + B(t)u(t),\ t \in \mathbb{Z}_{0,N-1} \\ & e(t) = M(t)x(t),\ t \in \mathbb{Z}_{0,N} \\ & h(0) + H_u(0)u(0) \le 0 \\ & h(t) + H_x(t)x(t) + H_u(t)u(t) \le 0, \\ & t \in \mathbb{Z}_{1,N-1} \\ & h(N) + H_x(N)x(N) \le 0 \end{aligned} \tag{1}$$

where $\mathsf{e} = \left[e^T(0), \ldots, e^T(N)\right]^T$, $\mathsf{x} = \left[x^T(0), \ldots, x^T(N)\right]^T$, $\mathsf{u} = \left[u^T(0), \ldots, u^T(N-1)\right]^T$, and where the matrices $A(t) \in \mathbb{R}^{n\times n}$, $B(t) \in \mathbb{R}^{n\times m}$ and $M(t) \in \mathbb{R}^{p\times n}$ define the system. Furthermore, $x(t)$ denotes the n states of the system, $u(t)$ denotes the m control inputs, and $e(t)$ denotes the p controlled outputs. Moreover, $H_x(t) \in \mathbb{R}^{c(t)\times n}$, $H_u(t) \in \mathbb{R}^{c(t)\times m}$ and $h(t) \in \mathbb{R}^{c(t)}$ define the inequality constraints, where $c(t)$ denotes the number of inequality constraints at time t. Furthermore, the following assumptions are made

Assumption 1: $Q_e(t) \in \mathbb{S}^p_{++}$, $t = 0, \ldots, N$

Assumption 2: $Q_u(t) \in \mathbb{S}^m_{++}$, $t = 0, \ldots, N-1$

Note that, when a hybrid MPC problem is solved, the resulting non-convex MIQP problem can be solved as a sequence of linear MPC problems in the form in (1), [18].

II. Quadratic Programming

In this section, the QP problem is introduced and some basic properties are discussed. Furthermore, the gradient projection algorithm used in this work is presented. For an extensive bibliography on QP, see [20]. The MPC problem in (1) is a QP problem with $\bar{n}$ variables, $\bar{p}$ equality constraints and $\bar{m}$ inequality constraints in the form

$$\begin{aligned} \underset{x_1,x_2}{\text{minimize}} \quad & \frac{1}{2}\begin{bmatrix} x_1^T & x_2^T \end{bmatrix} \underbrace{\begin{bmatrix} \tilde{H} & 0 \\ 0 & 0 \end{bmatrix}}_{H} \underbrace{\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}_{x} + \begin{bmatrix} \tilde{f}^T & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\ \text{subject to} \quad & \underbrace{\begin{bmatrix} A_{1\mathcal{E}}^T \\ A_{2\mathcal{E}}^T \end{bmatrix}^T}_{A_\mathcal{E}} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = b_\mathcal{E}, \quad \underbrace{\begin{bmatrix} A_{1\mathcal{I}}^T \\ A_{2\mathcal{I}}^T \end{bmatrix}^T}_{A_\mathcal{I}} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \le b_\mathcal{I} \end{aligned} \tag{2}$$

where $\bar{n} = \bar{n}_1 + \bar{n}_2$, $x_1 \in \mathbb{R}^{\bar{n}_1}$, $x_2 \in \mathbb{R}^{\bar{n}_2}$, $\tilde{H} \in \mathbb{S}^{\bar{n}_1}_{++}$, $\tilde{f} \in \mathbb{R}^{\bar{n}_1}$, $A_1 \in \mathbb{R}^{\bar{p}+\bar{m}\times\bar{n}_1}$, $A_2 \in \mathbb{R}^{\bar{p}+\bar{m}\times\bar{n}_2}$ and $b \in \mathbb{R}^{\bar{p}+\bar{m}}$. Furthermore, $\mathcal{E} \in \mathbb{Z}^{\bar{p}}_{++}$ and $\mathcal{I} \in \mathbb{Z}^{\bar{m}}_{++}$ denote sets of indices to rows representing equality constraints and inequality constraints respectively in $A \in \mathbb{R}^{\bar{p}+\bar{m}\times\bar{n}}$ and $b \in \mathbb{R}^{\bar{p}+\bar{m}}$. The dual problem to the problem in (2) can be found by forming the Lagrange dual function and performing maximization over λ and ν. This is thoroughly described in [2]. By reformulating the resulting dual maximization problem as a minimization problem and by removing a constant in the objective function, the result is a new equivalent QP problem in the form

$$\begin{aligned} \underset{\lambda,\nu}{\text{minimize}} \quad & \frac{1}{2}Q_D(\lambda,\nu) \\ \text{subject to} \quad & A_{2\mathcal{I}}^T\lambda + A_{2\mathcal{E}}^T\nu = 0, \quad \lambda \ge 0 \end{aligned} \tag{3}$$

where $\lambda \in \mathbb{R}^{\bar{m}}$, $\nu \in \mathbb{R}^{\bar{p}}$ and

$$\begin{aligned} Q_D(\lambda,\nu) = & \begin{bmatrix} \lambda \\ \nu \end{bmatrix}^T \begin{bmatrix} A_{1\mathcal{I}} \\ A_{1\mathcal{E}} \end{bmatrix} \tilde{H}^{-1} \begin{bmatrix} A_{1\mathcal{I}} \\ A_{1\mathcal{E}} \end{bmatrix}^T \begin{bmatrix} \lambda \\ \nu \end{bmatrix} + \\ & + 2\left(\tilde{f}^T\tilde{H}^{-1} \begin{bmatrix} A_{1\mathcal{I}} \\ A_{1\mathcal{E}} \end{bmatrix}^T + \begin{bmatrix} b_\mathcal{I} \\ b_\mathcal{E} \end{bmatrix}^T \right) \begin{bmatrix} \lambda \\ \nu \end{bmatrix} \end{aligned} \tag{4}$$

Apart from a known constant and a change of sign, strong duality holds for the primal problem in (2) and the (with a slight abuse of notation) dual problem in (3), [4]. The idea in this work is to solve a primal QP problem in the form in (2) by solving a dual problem in the form in (3) and then compute the primal optimal solution from the dual optimal solution.

A. Gradient Projection for QP

The algorithm presented in this paper is based on a gradient projection method, which in principle works as an active set method. However, large changes of the working set are possible in each iteration. Most properties of active set methods, like the possibility of efficient warm starts also hold for this improved method. For an introduction to active set methods, see, *e.g.*, [21].

In this section, the gradient projection algorithm used in this work is presented. It is presented for a general problem, and the efficient computations that utilizes problem structure will be presented later in the paper.

1) Introduction: A drawback with a classical active set method is that the working set is changing very slowly. For each change in the working set, a system of equations for a Newton step has to be solved. If the initial working set is very different from the optimal active set, it will take a lot of effort to reach this set. The idea in a gradient projection method is to allow a more rapid change of the working set, which in turn implies that often less Newton systems have to be solved before the optimal active set is found. However, when this method is applied to a QP problem with general inequality constraints, the projection operation performed in each iteration can become very computationally expensive. An exception is when the inequality constraints only consist of upper and lower bounds on variables. An important example of a problem that has constraints of this type is the dual QP problem, [21].

Algorithm 1 Gradient projection algorithm for QP, [21]

```
1: Compute a feasible starting point $x_0$.
2: Define the maximum number of iterations as $k_{max}$.
3: $k \leftarrow 0$
4: while $k < k_{max}$ do
5:   if $x_k$ satisfies the KKT conditions for (5) then
6:     $x^* \leftarrow x_k$
7:     STOP
8:   end if
9:   "Main step 1" (Gradient projection): Starting in $x_k$,
     find the Cauchy point $x^c$.
10:  "Main step 2" (Improvement): Find an approximate
     minimizer $x^+$ to the subproblem in (6), such that
     $Q(x^+) \leq Q(x^c)$ and such that $x^+$ is feasible with
     respect to the constraints in the problem in (5).
11:  $x_{k+1} \leftarrow x_k$
12:  $k \leftarrow k + 1$
13: end while
14: No solution was found in $k_{max}$ iterations.
```

Gradient projection methods are suitable for problems with many active constraints in the optimum. This property is important in optimal control applications where often many control inputs are at their boundaries at the optimum, [13]. The algorithm presented in this work shares some similarities with the one presented in [17], where the gradient projection method is combined with a conjugated gradient method. The differences between the algorithms are that the algorithm presented in this paper works on the dual QP problem and the Newton step is computed using a Riccati recursion instead of conjugated gradient iterations.

The algorithm presented in this work is inspired by the one in [21]. To simplify the presentation, a generic QP problem in the form

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & Q(x) = \frac{1}{2}x^T H x + f^T x \\ \text{subject to} \quad & x \geq 0 \end{aligned} \tag{5}$$

is considered, where $H \in \mathbb{S}_+^{\bar{n}}$ and $f \in \mathbb{R}^{\bar{n}}$. In the algorithm presented in this work, analogous ideas are applied to a dual QP problem in the form in (3).

2) The two main steps of the algorithm: Each iteration of the algorithm can be considered to consist of two steps; "Main step 1" and "Main step 2". The point found in the first step is called a Cauchy point and it has the property that it is good enough to guarantee global convergence, [21]. The purpose of the second step is to improve the convergence rate. The algorithm applied to a problem in the form in (5) is outlined in Algorithm 1. In "Main step 1", the gradient is computed at the current point. After the gradient has been computed, a line search optimization along the negative gradient (*i.e.*, steepest descent) direction is performed. If an inequality constraint is encountered before a minimizer is found along the line, the search direction is bent-off such that the constraint remains satisfied. The idea to project the search direction onto the feasible set is in this paper used for different types of search directions and the operation is here called projected line search. This procedure is illustrated

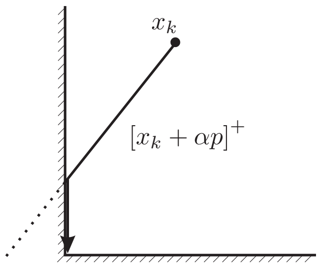

Fig. 1: The figure illustrates how the points along the line starting in the point x_k in the direction p are projected back onto the positive orthant during the projected line search operation. The resulting path is piecewise linear.

in Figure 1. The search is continued until either a local optimal solution is found along the resulting piecewise linear path, the search is stopped by constraints in sufficiently many directions to make it impossible to continue in any of the initial directions, or the step size tends to infinity without any constraints blocking the way. In the latter case, an eigenvector corresponding to a zero eigenvalue has been found and the problem is unbounded. In "Main step 2", a smaller optimization problem is defined from the original one, where all constraints that are active after the first step are kept *locked*. During this part of the algorithm, subproblems in the form

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & Q_s(x) = \frac{1}{2}x^T H x + f^T x \\ \text{subject to} \quad & x_i = x_i^c,\ i \in \mathcal{A}(x^c) \\ & x_i \geq 0,\ i \notin \mathcal{A}(x^c) \end{aligned} \tag{6}$$

are solved approximately, where $\mathcal{A}(x^c)$ denotes the active set in the last Cauchy point. To obtain global convergence of Algorithm 1, it is only necessary that the approximate solution is feasible with respect to the constraints of the original problem in (5) and that the objective function value of the approximate solution x^+ is not worse than the already found Cauchy point x^c, [21]. A common choice is to run a conjugated gradient method on the subspace defined by the locked constraints, [21], [17]. In this work, an alternative approach has been used. The search direction is taken as the vector from the current point toward the minimizer of a problem in the form

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & Q_s(x) \\ \text{subject to} \quad & x_i = x_i^c,\ i \in \mathcal{A}(x^c) \end{aligned} \tag{7}$$

which is the Newton step for the problem in (7) since the problem is quadratic. In this problem, the inequality constraints in (6) have been disregarded, and it is in this work solved directly using a Riccati recursion. This is explained in detail in Section IV-C. The computed step is projected onto the inequality constraints of the problem in (6). Hence, this part of the algorithm can be interpreted as a projected Newton step, which has been previously discussed in, *e.g.*, [13]. Note that, it is not in general true that the projection of the Newton step from (7) onto the feasible set of the problem in (6) will lead to the true minimizer of (6) in one, or several, iterations. This potential problem is discussed in [16], where also a remedy is presented. In this paper, this problem is avoided in alternative ways. Partly this is performed by *alternating* between iterations where the Newton step is projected and gradient projection iterations. The negative gradient direction does not suffer from the problem and it ensures convergence (under certain assumptions) as described in [21]. For details, see [18, pp. 155–159].

3) Gradient Projected onto the Nullspace of the Hessian: In this section, the case when the problem in (7) is unbounded is discussed. When this occurs, there does not exist any point where the KKT conditions are satisfied. For simplicity, the equality constraints in (7) are now eliminated as described in [21, pp. 428–434] and an equivalent problem in the form

$$\underset{x}{\text{minimize}} \quad \hat{Q}_s(x) = \tfrac{1}{2}x^T\hat{H}x + \hat{f}^Tx \tag{8}$$

is considered. The KKT system for this reduced problem is

$$\hat{H}x + \hat{f} = 0 \tag{9}$$

If $\hat{f} \notin \operatorname{range}\hat{H}$ there is no solution to the system of equations in (9). In such a case, an alternative search direction has to be chosen. One possibility is to choose the steepest descent direction, but then the convergence rate will be rather slow. In this work, the search direction is chosen as the projection of the steepest descent direction onto the nullspace of the Hessian $\hat{H}$. This choice is motivated in [18, pp. 192–193] and the resulting direction (with unit length) in the point x^0 can be written as

$$-\hat{g}(x^0) = -\mathcal{P}\left(\hat{H}x^0 + \hat{f}\right) \Big/ \left\|\mathcal{P}\left(\hat{H}x^0 + \hat{f}\right)\right\| \tag{10}$$

where $\mathcal{P} = Z\left(Z^TZ\right)^{-1}Z^T$. The columns in Z form a basis for the nullspace of $\hat{H}$.

III. The Dual MPC Problem

The optimization problem in (1) is in the form in (2). Hence, the dual optimization problem to (1) is in the form in (3). Then, without going into details, the dual problem to (1) can be written as

$$\begin{aligned}
\underset{\tilde{\mathsf{x}},\tilde{\mathsf{u}}}{\text{minimize}} \quad & J_D(\tilde{\mathsf{x}},\tilde{\mathsf{u}}) = \frac{1}{2}\tilde{u}^T(-1)\tilde{Q}_{\tilde{u}}(-1)\tilde{u}(-1) + \tilde{q}_{\tilde{u}}^T(-1)\tilde{u}(-1) \\
& + \frac{1}{2}\sum_{\tau=0}^{N-1}\Big(\tilde{x}^T(\tau)\tilde{Q}_{\tilde{x}}(\tau)\tilde{x}(\tau) + \tilde{u}^T(\tau)\tilde{Q}_{\tilde{u}}(\tau)\tilde{u}(\tau) \\
& + 2\tilde{x}^T(\tau)\tilde{Q}_{\tilde{x}\tilde{u}}(\tau)\tilde{u}(\tau) + 2\tilde{q}_{\tilde{u}}^T(\tau)\tilde{u}(\tau)\Big) + \tilde{q}_{\tilde{x}}^T(N)\tilde{x}(N) \\
\text{subject to} \quad & \tilde{x}(0) = \tilde{B}(-1)\tilde{u}(-1) \\
& \tilde{x}(\tau+1) = \tilde{A}(\tau)\tilde{x}(\tau) + \tilde{B}(\tau)\tilde{u}(\tau),\ \tau \in \mathbb{Z}_{0,N-1} \\
& \begin{bmatrix} 0 & -I_{c(N-\tau-1)} \end{bmatrix}\tilde{u}(\tau) \le 0,\ \tau \in \mathbb{Z}_{-1,N-1}
\end{aligned} \tag{11}$$

where $\tilde{\mathsf{x}} = \left[\tilde{x}^T(0),\ldots,\tilde{x}^T(N)\right]^T$, $\tilde{\mathsf{u}} = \left[\tilde{u}^T(-1),\ldots,\tilde{u}^T(N-1)\right]^T$ and where $\tilde{x}(\tau) \in \mathbb{R}^{\tilde{n}}$ and $\tilde{u}(\tau) \in \mathbb{R}^{\tilde{m}(\tau)}$. For a detailed derivation of the dual problem in (11), see [2], where also detailed relations to the primal variables are presented.

By Assumption 1 and Assumption 2, it can be shown, [18, p. 164], that the following inequality holds

$$\begin{bmatrix} \tilde{Q}_{\tilde{x}}(\tau) & \tilde{Q}_{\tilde{x}\tilde{u}}(\tau) \\ \tilde{Q}_{\tilde{x}\tilde{u}}^T(\tau) & \tilde{Q}_{\tilde{u}}(\tau) \end{bmatrix} \succeq 0,\ \tau = -1,\ldots,N-1 \tag{12}$$

Once the optimal solution to the dual problem in (11) is known, the optimal solution to the primal problem in (1) can easily be computed.

IV. Tailored Computations

In this section, tailoring of the most computationally demanding parts of the algorithm presented in Section II-A for the specific application MPC is discussed. Because of the limited space in this paper, some of the algorithms are only briefly discussed, while others are discussed in detail. For a complete treatment of all algorithms, including formal algorithm descriptions, see Paper D in [18].

A. Tailored Projected Line Search

The projected line search operation is performed by first searching for breakpoints where the search direction is bent and second performing one-dimensional optimizations along segments between breakpoints until the first local minimizer is found. See Figure 1. The one-dimensional objective function on each of those segments is a convex quadratic function. This makes it easy to find the exact optimizer for a segment. In this work, the solution of these one-dimensional optimization problems have been tailored for the MPC application with a resulting computational complexity of $\mathcal{O}(N)$. This is described in detail in [18, pp. 165–167].

B. Computation of Steepest Descent Direction

The steepest descent direction is the direction of the negative gradient. The computation of this search direction has been tailored for the dual MPC problem, and an algorithm with complexity $\mathcal{O}(N)$ has been found. This is described in detail in [18, pp. 167–168].

C. Newton Step Computation

The Newton step computation is an important part of the algorithm presented in this paper. In each iteration in "Main step 2", the solution to an equality constrained QP in the form in (7) has to be computed and the result gives the Newton step. This means that for a subset of the inequality constrained components in $\tilde{u}(\tau)$ in the problem in (11), the inequality constraints are temporarily considered as equality constraints. For the remaining components in $\tilde{u}(\tau)$, the inequality constraints are temporarily disregarded. Denote the part of $\tilde{u}(\tau)$ that is subject to an equality constraint $v(\tau)$ (*i.e.*, $v(\tau) = 0$) and the part that is unconstrained $w(\tau)$.

1) Efficient Factorization of the KKT System Coefficient Matrix: After a straightforward elimination of the variable v (*i.e.*, $\mathsf{v} = 0$), the KKT conditions for the subproblem in (7) are

$$\begin{bmatrix} 0 & \tilde{\mathsf{A}} & \tilde{\mathsf{B}}_{\mathsf{w}} \\ \tilde{\mathsf{A}}^T & \tilde{\mathsf{Q}}_{\tilde{\mathsf{x}}} & \tilde{\mathsf{Q}}_{\tilde{\mathsf{x}}\mathsf{w}} \\ \tilde{\mathsf{B}}_{\mathsf{w}}^T & \tilde{\mathsf{Q}}_{\tilde{\mathsf{x}}\mathsf{w}}^T & \tilde{\mathsf{Q}}_{\mathsf{w}} \end{bmatrix} \begin{bmatrix} \lambda \\ \tilde{\mathsf{x}} \\ \mathsf{w} \end{bmatrix} = \begin{bmatrix} 0 \\ -\tilde{\mathsf{q}}_{\tilde{\mathsf{x}}} \\ -\tilde{\mathsf{q}}_{\mathsf{w}} \end{bmatrix} \tag{13}$$

where $\lambda = \left[\lambda^T(0),\ldots,\lambda^T(N)\right]^T$, and where $\tilde{\mathsf{A}}$, $\tilde{\mathsf{B}}_{\mathsf{w}}$, $\tilde{\mathsf{Q}}_{\tilde{\mathsf{x}}}$, $\tilde{\mathsf{Q}}_{\tilde{\mathsf{x}}\mathsf{w}}$, $\tilde{\mathsf{Q}}_{\mathsf{w}}$, $\tilde{\mathsf{q}}_{\tilde{\mathsf{x}}}$ and $\tilde{\mathsf{q}}_{\mathsf{w}}$ are defined in Appendix A. In order to solve this system of equations efficiently, the coefficient matrix is factorized. If there exist matrices $\tilde{\mathsf{P}} \in \mathbb{S}^{(N+1)\tilde{n}}$,

$\tilde{\mathsf{G}} \in \mathbb{S}^{\sum_{\tau=-1}^{N-1} \tilde{m}_w(\tau)}$ and $\tilde{\mathsf{K}} \in \mathbb{R}^{\sum_{\tau=-1}^{N-1} \tilde{m}_w(\tau) \times (N+1)\tilde{n}}$ such that

$$\begin{aligned} \tilde{\mathsf{P}} &= \tilde{\mathsf{Q}}_{\tilde{\mathsf{x}}} + \big(I + \tilde{\mathsf{A}}\big)^T \tilde{\mathsf{P}} \big(I + \tilde{\mathsf{A}}\big) - \tilde{\mathsf{K}}^T \tilde{\mathsf{G}} \tilde{\mathsf{K}} \\ \tilde{\mathsf{G}}\tilde{\mathsf{K}} &= -\big(\tilde{\mathsf{Q}}_{\tilde{\mathsf{x}}\mathsf{w}}^T + \tilde{\mathsf{B}}_{\mathsf{w}}^T \tilde{\mathsf{P}} \big(I + \tilde{\mathsf{A}}\big)\big) \\ \tilde{\mathsf{G}} &= \tilde{\mathsf{Q}}_{\mathsf{w}} + \tilde{\mathsf{B}}_{\mathsf{w}}^T \tilde{\mathsf{P}} \tilde{\mathsf{B}}_{\mathsf{w}} \end{aligned} \tag{14}$$

then the following factorization holds

$$\begin{bmatrix} \tilde{\mathsf{Q}}_{\tilde{\mathsf{x}}} & \tilde{\mathsf{A}}^T & \tilde{\mathsf{Q}}_{\tilde{\mathsf{x}}\mathsf{w}} \\ \tilde{\mathsf{A}} & 0 & \tilde{\mathsf{B}}_{\mathsf{w}} \\ \tilde{\mathsf{Q}}_{\tilde{\mathsf{x}}\mathsf{w}}^T & \tilde{\mathsf{B}}_{\mathsf{w}}^T & \tilde{\mathsf{Q}}_{\mathsf{w}} \end{bmatrix} = \begin{bmatrix} (\tilde{\mathsf{A}} + \tilde{\mathsf{B}}_{\mathsf{w}}\tilde{\mathsf{K}})^T & -\tilde{\mathsf{P}} & -\tilde{\mathsf{K}}^T \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \cdot \\ \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ \tilde{\mathsf{B}}_{\mathsf{w}}^T & 0 & I \end{bmatrix} \begin{bmatrix} -\tilde{\mathsf{P}} & I & 0 \\ I & 0 & 0 \\ 0 & 0 & \tilde{\mathsf{G}} \end{bmatrix} \begin{bmatrix} I & 0 & \tilde{\mathsf{B}}_{\mathsf{w}} \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} \tilde{\mathsf{A}} + \tilde{\mathsf{B}}_{\mathsf{w}}\tilde{\mathsf{K}} & 0 & 0 \\ -\tilde{\mathsf{P}} & I & 0 \\ -\tilde{\mathsf{K}} & 0 & I \end{bmatrix} \tag{15}$$

The existence and uniqueness of such matrices has already been considered in [22] under the assumption that $\tilde{\mathsf{Q}}_{\mathsf{w}} \succ 0$ and it was shown that $\tilde{\mathsf{P}} \succeq 0$ and $\tilde{\mathsf{G}} \succ 0$. In this work, the result is generalized to the singular case, *i.e.*, when $\tilde{\mathsf{Q}}_{\mathsf{w}} \succeq 0$ is singular. Note that, it follows directly from the equation in (14) that $\tilde{\mathsf{G}}$ is uniquely determined by $\tilde{\mathsf{P}}$, and that $\tilde{\mathsf{G}} \succeq 0$ if $\tilde{\mathsf{P}} \succeq 0$ since $\tilde{\mathsf{Q}}_{\mathsf{w}} \succeq 0$. Furthermore, the outer four matrices in (15) are non-singular. Note especially that $\tilde{\mathsf{A}} + \tilde{\mathsf{B}}_{\mathsf{w}}\tilde{\mathsf{K}}$ is invertible and lower triangular with diagonal elements equal to -1 for any choice of $\tilde{\mathsf{A}}$, $\tilde{\mathsf{B}}_{\mathsf{w}}$ and $\tilde{\mathsf{K}}$, [18, p. 191]. This implies that the KKT system is non-singular if and only if the center matrix is non-singular. Notice that, the upper left block $\begin{bmatrix} -\tilde{\mathsf{P}} & I \\ I & 0 \end{bmatrix}$ in this matrix is non-singular. Hence, the KKT system is non-singular if and only $\tilde{\mathsf{G}}$ is non-singular.

It will now be shown that there exist matrices $\tilde{\mathsf{P}} \succeq 0$ and $\tilde{\mathsf{K}}$ such that the equations in (14) hold. These equations can be expressed in the block matrices of which the involved matrices consist. The result is

$$\begin{aligned} \tilde{P}(N) &= 0 \\ \tilde{G}(0) &= \tilde{Q}_w(-1) + \tilde{B}_w^T(-1)\tilde{P}(0)\tilde{B}_w(-1) \\ \tilde{F}(\tau+1) &= \tilde{Q}_{\tilde{x}}(\tau) + \tilde{A}^T(\tau)\tilde{P}(\tau+1)\tilde{A}(\tau) \\ \tilde{G}(\tau+1) &= \tilde{Q}_w(\tau) + \tilde{B}_w^T(\tau)\tilde{P}(\tau+1)\tilde{B}_w(\tau) \\ \tilde{H}(\tau+1) &= \tilde{Q}_{\tilde{x}w}(\tau) + \tilde{A}^T(\tau)\tilde{P}(\tau+1)\tilde{B}_w(\tau) \\ \tilde{G}(\tau+1)\tilde{K}(\tau+1) &= -\tilde{H}^T(\tau+1) \\ \tilde{P}(\tau) &= \tilde{F}(\tau+1) - \tilde{K}^T(\tau+1)\tilde{G}(\tau+1)\tilde{K}(\tau+1) \end{aligned} \tag{16}$$

where all equations are to be satisfied for $\tau = 0, \ldots, N-1$ unless otherwise stated. It will now be shown that there exist matrices $\tilde{P}(\tau) \succeq 0$, and $\tilde{K}(\tau+1)$ such that $\tilde{G}(\tau+1)\tilde{K}(\tau+1) = -\tilde{H}^T(\tau+1)$. Furthermore, it will be shown that $\tilde{P}(\tau)$ is unique, also in the case when the KKT system is singular. It follows directly from (16) that $\tilde{P}(N) \succeq 0$. Now, assume that $\tilde{P}(\tau+1) \succeq 0$ for an arbitrary $\tau \in \{0, \ldots, N-1\}$. Then, it follows from the equations in (16) and (12) that $\tilde{F}(\tau+1) \succeq 0$. Furthermore,

$$\begin{bmatrix} \tilde{F}(\tau+1) & \tilde{H}(\tau+1) \\ \tilde{H}^T(\tau+1) & \tilde{G}(\tau+1) \end{bmatrix} = \begin{bmatrix} \tilde{Q}_{\tilde{x}}(\tau) & \tilde{Q}_{\tilde{x}w}(\tau) \\ \tilde{Q}_{\tilde{x}w}^T(\tau) & \tilde{Q}_w(\tau) \end{bmatrix} \\ + \begin{bmatrix} \tilde{A}(\tau) & \tilde{B}_w(\tau) \end{bmatrix}^T \tilde{P}(\tau+1) \begin{bmatrix} \tilde{A}(\tau) & \tilde{B}_w(\tau) \end{bmatrix} \succeq 0 \tag{17}$$

by the equation in (12), the assumption that $\tilde{P}(\tau+1) \succeq 0$, and the fact that the last term in the expression is quadratic. By the Schur complement formula for positive semidefinite matrices the following holds

$$\begin{bmatrix} \tilde{F}(\tau+1) & \tilde{H}(\tau+1) \\ \tilde{H}^T(\tau+1) & \tilde{G}(\tau+1) \end{bmatrix} \succeq 0 \Leftrightarrow \\ \tilde{G}(\tau+1) \succeq 0,\ \Big(I - \tilde{G}(\tau+1)\tilde{G}^\dagger(\tau+1)\Big)\tilde{H}^T(\tau+1) = 0, \\ \tilde{F}(\tau+1) - \tilde{H}(\tau+1)\tilde{G}^\dagger(\tau+1)\tilde{H}^T(\tau+1) \succeq 0 \tag{18}$$

where † denotes the pseudoinverse. Furthermore, notice that

$$\tilde{P}(\tau) = \tilde{F}(\tau+1) - \tilde{H}(\tau+1)\tilde{G}^\dagger(\tau+1)\tilde{H}^T(\tau+1) \tag{19}$$

which follows from the equations in (16), a basic property of the pseudoinverse (*i.e.*, $\tilde{G} = \tilde{G}\tilde{G}^\dagger\tilde{G}$) and the symmetry of $\tilde{G}(\tau+1)$. By combining (18) and (19), it directly follows that $\tilde{P}(\tau) \succeq 0$, and by induction it follows that this is true for all $\tau = N, \ldots, 0$. Furthermore, since there exists a solution $\tilde{K}(\tau+1)$ to a system of equations in the form $\tilde{G}(\tau+1)\tilde{K}(\tau+1) = -\tilde{H}^T(\tau+1)$ if

$$\Big(I - \tilde{G}(\tau+1)\tilde{G}^\dagger(\tau+1)\Big)\tilde{H}^T(\tau+1) = 0 \tag{20}$$

it is possible to conclude that there exist matrices $\tilde{K}(\tau+1)$ for all $\tau = N-1, \ldots, 0$. Note, however, that $\tilde{K}(\tau+1)$ is not unique in the case when the KKT system is singular since $\tilde{G}(\tau)$ is singular for at least one τ in that case. Finally, it follows from (19) that $\tilde{P}(\tau)$ is independent of the choice of $\tilde{K}(\tau+1)$, and is hence unique.

Summarizing, the factorization exists also in the case when the KKT system is singular. However, the factorization is not unique in that case because there is a freedom in the choice of $\tilde{K}(\tau+1)$. Despite this, $\tilde{P}(\tau)$ and $\tilde{G}(\tau)$ are unique. The factorization can be performed very efficiently as the well-known Riccati recursion, which is known to have linear computational complexity in the prediction horizon. For further details see, *e.g.*, [22].

If the KKT system is singular, this will be found during the solution process of the equation $\tilde{G}(\tau+1)\tilde{K}(\tau+1) = -\tilde{H}^T(\tau+1)$ in the Riccati recursion. Generally, $\tilde{G}(\tau+1)$ can be factored using, *e.g.*, the QR factorization or the SVD factorization. Both work also in the singular case and they can be used to compute the nullspace of $\tilde{G}(\tau+1)$, which is needed in the computation of the nullspace of the entire KKT system. This will be considered in detail in Section IV-D. An alternative is to use the Cholesky factorization as in [22] and monitor if it breaks down.

2) Solving the KKT System Using a Riccati Recursion: Once the KKT coefficient matrix has been factorized using the Riccati recursion, the system in (13) can be solved using backward and forward substitutions, all with linear computational complexity in the prediction horizon length, as described in, *e.g.*, [22]. The result is basically the Newton step, and this computation is only performed if the factorization step terminates without any detection of singularity. If singularity is detected, the search direction presented in the next section is used. For further details, see [18, p. 172].

D. Inconsistent KKT System

If the KKT system is inconsistent, an alternative search direction has to be found as discussed in Section II-A.3. In this section, it is described how such a direction can be computed efficiently.

1) Nullspace Computation: Consider the factorization of the KKT system coefficient matrix in (15) and multiply out the two outer matrices on each side of the center one. The result is

$$\begin{bmatrix} \tilde{\mathsf{Q}}_{\tilde{\mathsf{x}}} & \tilde{\mathsf{A}}^T & \tilde{\mathsf{Q}}_{\tilde{\mathsf{x}}\mathsf{w}} \\ \tilde{\mathsf{A}} & 0 & \tilde{\mathsf{B}}_{\mathsf{w}} \\ \tilde{\mathsf{Q}}_{\tilde{\mathsf{x}}\mathsf{w}}^T & \tilde{\mathsf{B}}_{\mathsf{w}}^T & \tilde{\mathsf{Q}}_{\mathsf{w}} \end{bmatrix} = \begin{bmatrix} \tilde{\mathsf{A}}^T & -\tilde{\mathsf{P}}^T & -\tilde{\mathsf{K}}^T \\ 0 & I & 0 \\ \tilde{\mathsf{B}}_{\mathsf{w}}^T & 0 & I \end{bmatrix} \begin{bmatrix} -\tilde{\mathsf{P}} & I & 0 \\ I & 0 & 0 \\ 0 & 0 & \tilde{\mathsf{G}} \end{bmatrix} \begin{bmatrix} \tilde{\mathsf{A}} & 0 & \tilde{\mathsf{B}}_{\mathsf{w}} \\ -\tilde{\mathsf{P}} & I & 0 \\ -\tilde{\mathsf{K}} & 0 & I \end{bmatrix} \triangleq \Pi^T \Sigma \Pi \tag{21}$$

Note that

$$\xi \in \operatorname{null}(\Pi^T \Sigma \Pi) \Leftrightarrow \Pi^T \Sigma \Pi \xi = 0 \Leftrightarrow \Sigma \Pi \xi = 0 \tag{22}$$

since Π is non-singular. Furthermore,

$$\Sigma \Pi \xi = 0 \Leftrightarrow \Sigma \eta = 0 \text{ and } \xi = \Pi^{-1} \eta \tag{23}$$

Let the columns of the matrix N_Σ be a basis for the nullspace of Σ, and the columns of $N_{\Sigma\Pi}$ a basis for the nullspace of $\Pi^T\Sigma\Pi$. It is straightforward to generalize the ideas in (22) and in (23) in order to compute an entire basis for the nullspace of $\Pi^T\Sigma\Pi$ using a basis for the nullspace of Σ. That is, $N_{\Sigma\Pi}$ can be computed as

$$N_{\Sigma\Pi} = \Pi^{-1} N_\Sigma \tag{24}$$

Note that, Π is non-singular independently of which solution $\tilde{K}(\tau+1)$ that is used. Furthermore, since Π depends on $\tilde{K}(\tau+1)$, Π is non-unique. However, for every choice of $\tilde{K}(\tau+1)$, the columns of $N_{\Sigma\Pi}$ form *one* basis for the nullspace of $\Pi^T\Sigma\Pi$ since Π is always non-singular. The computations in (24) can be performed very efficiently thanks to the structure of N_Σ and Π. For further details, see [18, p. 174].

2) Projection of the Steepest Descent Direction: In this section, it will be shown how the search direction discussed in Section II-A.3 can be computed efficiently for this application. The desired search direction is the projection of the negative gradient in the current point onto the nullspace of the reduced Hessian of the dual subproblem in the form in (7). By comparing the nullspace equation for the reduced Hessian (states eliminated) with the nullspace equation for the KKT system coefficient matrix in (15) it can be shown, [18, p. 175], that the nullspace of the reduced Hessian can be computed by computing the nullspace of the KKT system coefficient matrix in (15). How this can be efficiently computed has already been shown in the previous section.

After the nullspace has been computed, the projection of the gradient onto the nullspace spanned by the columns in $N_{\Sigma\Pi}$ can be found as

$$\hat{g} = N_{\Sigma\Pi} \left(N_{\Sigma\Pi}^T N_{\Sigma\Pi} \right)^{-1} N_{\Sigma\Pi}^T g \tag{25}$$

where g is the gradient, and where $N_{\Sigma\Pi}^T N_{\Sigma\Pi}$ is non-singular since the columns of $N_{\Sigma\Pi}$ are linearly independent because they are basis vectors. In [18, pp. 192–193], it is shown that $-\hat{g}$ is the direction that gives the fastest descent possible in the nullspace of the Hessian. Due to the structure in $N_{\Sigma\Pi}$, the matrix $N_{\Sigma\Pi}^T N_{\Sigma\Pi}$ is often sparse and the computations in (25) can therefore often be performed very efficiently. A truly tailored version of this projection operation is left as future work.

V. Numerical Experiments

In this section, the QP algorithm presented in this paper is applied to random linear MPC problems. Since the main objective of this work is control of hybrid systems, also an MIPC problem is considered. All computational performance tests have been performed on a computer with two processors of the type Dual Core AMD Opteron 270 sharing 4 GB RAM (the code was not written to utilize multiple cores) running CentOS release 4.6 (Final) Kernel 2.6.9-55.ELsmp and MATLAB 7.2.0.294. Computational times have been measured using the MATLAB command `cputime`.

The random linear MPC problems are in the form in (1) with $n = 10, p = 10, m = 5$ and $c(t) = 10, t = 0, \ldots, N-1$ for different values of N in the range 50 to 450. For each prediction horizon length, 10 stable random systems are found using the MATLAB function `drss`. The constraints are bound constraints on the control signals and are chosen such that both feasible as well as infeasible problems are present among the test problems. Out of the 50 problems, 38 are feasible. The reference signal is chosen as a vector of sinusoids with random phases; one for each output of the system. Furthermore, the cost matrices in the objective function have also been chosen randomly, but in a way that they are symmetric and positive definite.

In the examples, dense and sparse formulations of the MPC problem have been solved. The sparse formulation is a formulation in the form in (1). The objective function's Hessian and the equality constraints in this problem are sparse. After eliminating x and e, the result is a new equivalent QP problem without equality constraints. The objective function's Hessian of this problem is dense but of smaller size since the only free variables are the control signals u. Warm starts have not been considered in the linear experiments.

The average computational times measured during this experiment are shown in the left plot in Figure 2. The algorithm presented in this paper is implemented in the function `drgpqp`. "CPLEX sparse" denotes CPLEX given a sparse representation of the MPC problem, and the default settings in CPLEX are used. This means that a primal Interior Point (IP) solver will be used to solve the problem. "CPLEX dual sparse" is the sparse formulation solved using a dual active set solver in CPLEX. "CPLEX dual dense" is the dense version of the MPC problem solved by a dual active set solver in CPLEX. As been mentioned previously, dual active set solvers are preferable in applications where warm starts are used. Consequently, CPLEX uses its dual solver as the default solver for the node problems in branch and bound. Hence, the comparisons between `drgpqp` and CPLEX's *dual* solvers are the most important ones in this application where warm starts will be frequently performed. The primal solver is only shown as a reference. As can be seen in Figure 2, `drgpqp` has lower computational complexity compared to CPLEX's dual solvers for large values of N. Note that `drgpqp` is implemented entirely in m-code, while CPLEX is running in compiled code. Hence, the *trends* are most interesting in this experiment, and the absolute times are of minor interest. According to the result in this experiment, the m-coded solver built on the ideas in

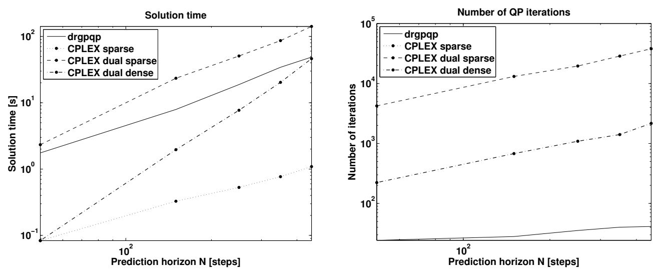

Fig. 2: These plots show the computational times and number of active set QP iterations for different QP solvers. The QP algorithm described in this section is `drgpqp`. The conclusion drawn is that the computational time as well as the number of QP iterations grows more slowly for the algorithm presented in this work compared to other *dual* solvers.

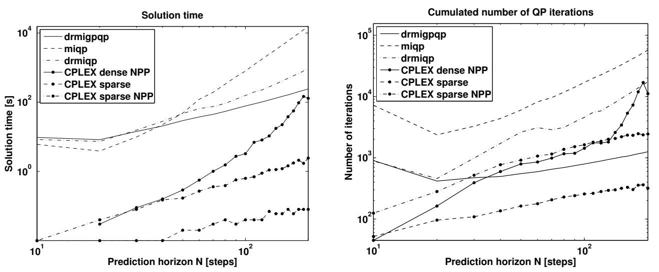

Fig. 3: These plots show the computational times and number of cumulated QP iterations for six different MIQP solvers. The result is that the algorithm presented in this paper is the m-code implementation with the best performance for large values of N. Even though the implementation of the algorithm in this paper is far behind CPLEX when absolute computational times are compared, its computational time *grows* similar compared to the one of CPLEX, also when sparsity is utilized by the latter. Furthermore, the right plot gives an indication of that the algorithm presented in this work is rather efficient in terms of number of QP iterations.

this work has lower *absolute* computational time compared to the dual solver in CPLEX applied to the sparse problem. For some reason, the absolute performance is rather bad for this configuration. Even though the solution returned is correct, it cannot be excluded that some error occurs, or some significant overhead is added, *e.g.*, in CPLEXINT during the call to CPLEX. Another possible explanation is that CPLEX does not solve this problem formulation efficiently from scratch, and that the dual solver in CPLEX is optimized for warm starts rather than solving a problem from scratch. The computational complexities in this experiment grow approximately as $\mathcal{O}(N^{1.5})$ for `drgpqp`, $\mathcal{O}(N^{1.2})$ for "CPLEX sparse", $\mathcal{O}(N^{1.9})$ for "CPLEX dual sparse", and $\mathcal{O}(N^{2.9})$ for "CPLEX dual dense". In the right plot in Figure 2, the average numbers of QP iterations are compared. Since CPLEX is not open source software, it is hard to make a fair comparison. The number of iterations presented for CPLEX is the status variable denoted "simplex iterations", which is assumed to be proportional to the number of Newton steps computed. Note that, the number of reported active set QP iterations for "CPLEX sparse" is zero, since it is an IP solver rather than an active set solver. The result from this experiment indicates that the method used in this work significantly cuts down the number of QP iterations needed to reach the optimal solution. It should, however, be mentioned that each iteration in the algorithm presented in this work is more expensive than in a classical active set solver. In future research, an attempt will be made to reduce the computational time by updating factorizations to a higher extent than today rather than computing them from scratch. Furthermore, inexact line searches will be tested instead of the exact line searches that are used in the current algorithm.

In the second part of the simulations, the algorithm is applied to an MIPC problem where the attitude of a satellite is to be controlled. The control signals consist of one real-valued control signal and two binary-valued control signals. The discrete-time model has three states. The setup, except for the QP solver used in branch and bound, is identical to what has been previously presented in [3], where also further details about the experiment setup can be found. The average computational times are presented in the left plot in Figure 3 and the average accumulated (in the branch and bound tree for each problem) number of QP iterations are presented in the right plot in Figure 3. Since the branch and bound part of CPLEX is much more advanced with highly developed pre-processing and heuristics, *etc.*, than the one in `drmigpqp`, `drmiqp` and `miqp`, CPLEX has been "detuned" in order to make the comparison more fair. This is indicated by the acronym "NPP" (No PreProcessing) in the plots, and the parameters modified are listed in [18, pp. 191–192].

The MIQP algorithm using the QP solver presented in this paper is referred to as `drmigpqp`. Furthermore, `miqp` is the freely available solver described in [23] solving a dense (states and control error eliminated) formulation of the problem, `drmiqp` is an implementation of the algorithm presented in [3], "CPLEX dense NPP" is CPLEX when given a dense version of the problem and preprocessing has been turned off, "CPLEX sparse" is CPLEX when given a sparse version (states kept) of the problem and default settings are used (*i.e.*, basically all features in CPLEX are enabled), and finally, "CPLEX sparse NPP" is CPLEX when given a sparse formulation of the problem and preprocessing has been turned off. All solvers except `miqp` are using warm starts in branch and bound. In this example, for prediction horizons longer than 20 time steps, the computational complexity for the algorithm presented in this paper grows approximately as $\mathcal{O}(N^{1.5})$, while for the generic solver `miqp`, using the QP solver `quadprog`, it grows approximately as $\mathcal{O}(N^{3.6})$. The computational complexity for CPLEX grows in the default configuration approximately as $\mathcal{O}(N^{0.9})$. After detuning, CPLEX computational time grows approximately as $\mathcal{O}(N^{1.8})$ using the sparse formulation and $\mathcal{O}(N^{3.6})$ using the dense formulation. The implementation of the branch and bound algorithm used in this work is the one originally implemented in `miqp`. Hence, it is exactly the same branch and bound code used in the results for `drmigpqp`, `drmiqp` and `miqp`.

The algorithm has also been applied to *random* problems similar to those used in the first part of this section, but with integer variables included in the problem. The performance in these tests has not been as good as in the satellite example. The major explanation might be that for more advanced problems, more nodes have to be explored during branch and bound and it gets more crucial that the implementation of the branching is performed efficiently. This will be improved in the future. Furthermore, even though CPLEX is "detuned", it still explores a significant smaller number of nodes than `drmigpqp`. This is not a result of the algorithm presented in this paper, but a result of that the branch and bound part, and possibly some heuristic, of the code in CPLEX still cuts away large parts of the tree. The random problems are thoroughly discussed in [18, pp. 181–183].

Summarizing, the result presented in this paper should be interpreted like a "proof of concept". The intention is not to present *the* QP solver for MPC and MIPC. To be useful in practice, there are some features that remain to be added. For example, for the MIPC application, an efficient preprocessing algorithm has to be used in and before branch and bound. Despite that there are improvements that could be made, the main goal in this work has in large been reached. Especially, the number of QP iterations has been significantly reduced compared to a classical active set method.

VI. Conclusions

The main result in this work is a QP algorithm that combines several important concepts into an algorithm with very good computational performance for MPC and MIPC problems. The algorithm has warm start properties similar to a classical active set method working in the dual space. Furthermore, it identifies the active set significantly faster than a classical active set method. Moreover, all computations of major complexity, except for a projection operation, have been tailored for the MPC problem. The projection operation is not frequently used, but it would be interesting in the future to fill in this last gap to an all tailored algorithm. In numerical experiments, the performance is good; for QP problems the computational complexity grows slower than the one of CPLEX's dual solvers as the length of the prediction horizon grows. Furthermore, the numerical results indicate that the algorithm solves the problem using fewer QP iterations than the active set solvers in CPLEX. When the algorithm is applied to MIPC problems, the performance is still good on simple problems, but it seems like the branch process has to be more efficiently implemented to maintain high performance also in more difficult problems. Examples of future work are to test inexact line search methods and to test if more advanced factorization updates are necessary and how they can be efficiently implemented.

Appendix

A. Definitions of Stacked Matrices

$$
\begin{aligned}
\tilde{\mathsf{Q}}_{\tilde{\mathsf{x}}} &= \operatorname{diag}\left(\tilde{Q}_{\tilde{x}}(0), \ldots, \tilde{Q}_{\tilde{x}}(N-1)\right), \\
\tilde{\mathsf{Q}}_{\mathsf{w}} &= \operatorname{diag}\left(\tilde{Q}_{w}(-1), \ldots, \tilde{Q}_{w}(N-1)\right), \\
\tilde{\mathsf{Q}}_{\tilde{\mathsf{x}}\tilde{\mathsf{w}}} &= \operatorname{diag}\left(\tilde{Q}_{\tilde{x}w}(0), \ldots, \tilde{Q}_{\tilde{x}w}(N-1)\right), \\
\tilde{\mathsf{q}}_{\tilde{\mathsf{x}}} &= \left[0, \ldots, \tilde{q}_{\tilde{x}}^{T}(N)\right]^{T}, \ \tilde{\mathsf{q}}_{\mathsf{w}} = \left[\tilde{q}_{w}^{T}(-1), \ldots, \tilde{q}_{w}^{T}(N-1)\right]^{T}, \\
\tilde{\mathsf{B}}_{\mathsf{w}} &= \operatorname{diag}\left(\tilde{B}(-1), \ldots, \tilde{B}(N-1)\right), \\
\tilde{\mathsf{G}} &= \operatorname{diag}\left(\tilde{G}(0), \ldots, \tilde{G}(N)\right), \ \tilde{\mathsf{P}} = \operatorname{diag}\left(\tilde{P}(0), \ldots, \tilde{P}(N)\right)
\end{aligned}
\tag{26}
$$

$$
\tilde{\mathsf{A}} = \begin{bmatrix} -I & 0 & \ldots & 0 & 0 \\ \tilde{A}(0) & -I & \ldots & 0 & 0 \\ 0 & \tilde{A}(1) & \ddots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \ldots & \tilde{A}(N-1) & -I \end{bmatrix}, \ \tilde{\mathsf{K}} = \begin{bmatrix} 0 & 0 & \ldots & 0 & 0 \\ \tilde{K}(1) & 0 & \ldots & 0 & 0 \\ 0 & \tilde{K}(2) & \ddots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \ldots & \tilde{K}(N) & 0 \end{bmatrix} \tag{27}
$$

References

[1] A. Bemporad and M. Morari, "Control of systems integrating logic, dynamics, and constraints," *Automatica*, vol. 35, pp. 407 – 427, 1999.

[2] D. Axehill, "Applications of integer quadratic programming in control and communication," Licentiate's Thesis, Linköpings universitet, 2005. [Online]. Available: http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-5263

[3] D. Axehill and A. Hansson, "A mixed integer dual quadratic programming algorithm tailored for MPC," in *Proceedings of the 45th IEEE Conference on Decision and Control*, Manchester Grand Hyatt, San Diego, USA, Dec. 2006, pp. 5693–5698.

[4] ——, "A dual gradient projection quadratic programming algorithm tailored for mixed integer predictive control," Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, Tech. Rep. LiTH-ISY-R-2833, Dec. 2007.

[5] D. Goldfarb and A. Idnani, "A numerically stable dual method for solving strictly convex quadratic programs," *Math. Program.*, vol. 27, pp. 1 – 33, 1983.

[6] S. J. Wright, "Applying new optimization algorithms to model predictive control," in *Chemical Process Control – V*, J. C. Kantor, C. E. García, and B. Carnahan, Eds., 1997, pp. 147 – 155.

[7] R. Fletcher and S. Leyffer, "Numerical experience with lower bounds for MIQP branch-and-bound," *SIAM J. Optimiz.*, vol. 8, no. 2, pp. 604 – 616, May 1998.

[8] L. A. Wolsey, *Integer Programming*. John Wiley & Sons, Inc., 1998.

[9] C. van de Panne and A. Whinston, "The simplex and the dual method for quadratic programming," *Oper. Res. Quart.*, vol. 15, no. 4, pp. 355 – 388, 1964.

[10] M. J. D. Powell, "On the quadratic programming algorithm of Goldfarb and Idnani," *Math. Program. Stud.*, vol. 25, pp. 46 – 61, 1985.

[11] H. Jonson, "A Newton method for solving non-linear optimal control problems with general constraints," Ph.D. dissertation, Linköpings Tekniska Högskola, 1983.

[12] C. V. Rao, S. J. Wright, and J. B. Rawlings, "Application of interior-point methods to model predictive control," Mathematics and Computer Science Division, Argonne National Laboratory, Preprint ANL/MCS-P664-0597, May 1997.

[13] D. P. Bertsekas, "On the Goldstein – Levitin – Polyak gradient projection method," *IEEE Trans. Autom. Control*, vol. 21, no. 2, pp. 174 – 184, Apr. 1976.

[14] A. A. Goldstein, "Convex programming in Hilbert space," *Bull. Amer. Math. Soc.*, May 1964.

[15] E. S. Levitin and B. T. Polyak, "Constrained minimization problems," *Comput. Math. Math. Phys.*, vol. 6, pp. 1 – 50, 1966.

[16] D. P. Bertsekas, "Projected Newton methods for optimization problems with simple constraints," *SIAM J. Control Optim.*, vol. 20, no. 2, pp. 221 – 246, Mar. 1982.

[17] J. J. Moré, "Gradient projection techniques for large-scale optimization problems," in *Proceedings of the 28th Conference on Decision and Control*, Dec. 1989, pp. 378 – 381.

[18] D. Axehill, "Integer quadratic programming for control and communication," Ph.D. dissertation, Linköping University, 2008. [Online]. Available: http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-10642

[19] B. Lie, M. D. Díez, and T. A. Hauge, "A comparison of implementation strategies for MPC," *Modeling, identification and control*, vol. 26, no. 1, pp. 39 – 50, Jan. 2005.

[20] N. I. M. Gould and P. L. Toint, "A quadratic programming bibliography," Numerical Analysis Group, Rutherford Appleton Laboratory, Tech. Rep., Feb. 2001.

[21] J. Nocedal and S. J. Wright, *Numerical Optimization – Second Edition*. Springer Verlag, 2006.

[22] L. Vandenberghe, S. Boyd, and M. Nouralishahi, "Robust linear programming and optimal control," Department of Electrical Engineering, University of California Los Angeles, Tech. Rep., 2002.

[23] A. Bemporad and D. Mignone, "A Matlab function for solving mixed integer quadratic programs version 1.02 user guide," Institut für Automatik, ETH, Tech. Rep., 2000.