MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Sequential Quadratic Programming Algorithm for Real-Time Mixed-Integer Nonlinear MPC

Quirynen, Rien; Di Cairano, Stefano

TR2021-147 January 11, 2022

Abstract

Nonlinear model predictive control (NMPC) has grown mature and algorithmic techniques exist, e.g., based on sequential quadratic programming (SQP) methods, to handle relatively complex constrained control systems. In addition, model predictive control for hybrid dynamics systems, including both continuous and discrete decision variables, can be implemented efficiently based on state of the art mixed-integer quadratic programming (MIQP) algorithms. This paper proposes a novel mixed-integer SQP (MISQP) optimization algorithm as a heuristic search technique to find feasible, but possibly suboptimal, solutions for real-time implementations of mixed-integer NMPC (MI-NMPC). Two particular variants of the MISQP algorithm are described and motivated. Based on a preliminary software implementation, the real-time MISQP performance is illustrated for closed-loop MINMPC simulations on a nontrivial vehicle control case study, featuring worst-case computation times below 30 milliseconds.

IEEE Conference on Decision and Control (CDC) 2021

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

Sequential Quadratic Programming Algorithm for Real-Time Mixed-Integer Nonlinear MPC

Rien Quirynen[1] and Stefano Di Cairano[1]

***Abstract*— Nonlinear model predictive control (NMPC) has grown mature and algorithmic techniques exist, e.g., based on sequential quadratic programming (SQP) methods, to handle relatively complex constrained control systems. In addition, model predictive control for hybrid dynamical systems, including both continuous and discrete decision variables, can be implemented efficiently based on state of the art mixed-integer quadratic programming (MIQP) algorithms. This paper proposes a novel mixed-integer SQP (MISQP) optimization algorithm as a heuristic search technique to find feasible, but possibly suboptimal, solutions for real-time implementations of mixed-integer NMPC (MINMPC). Two variants of the MISQP algorithm are described and motivated. Based on a preliminary software implementation, the real-time MISQP performance is illustrated for closed-loop MINMPC simulations on a nontrivial vehicle control case study, featuring worst-case computation times below 30 milliseconds.**

I. INTRODUCTION

Model predictive control (MPC) allows enforcing constraints and optimizing a performance criterion by solving a constrained optimal control problem (OCP) at each time step [1]. This framework also applies to hybrid dynamical systems that include both continuous and discrete decision variables, providing a powerful model-based control design for a large class of problems, e.g., including switched dynamical systems [2], discrete or quantized actuation [3], motion planning [4], logic rules and temporal logic specifications [5]. In case of nonlinear objective and constraint functions, nonlinear MPC (NMPC) with discrete decision variables requires the online solution of mixed-integer nonlinear programming (MINLP) problems, which is known to be $\mathcal{NP}$-hard, in general [6].

In this work, we propose to solve mixed-integer optimal control problems (MIOCPs) of the following form

$$\min_{X,U,W} \sum_{k=0}^{N-1} l_k(x_k,u_k) + m(x_N) + \sum_{k=0}^{N} c_k^\top w_k \tag{1a}$$

$$\text{s.t.} \quad x_0 = \hat{x}_t, \tag{1b}$$

$$x_{k+1} = \psi_k(x_k,u_k) + D_k w_k, \quad \forall k \in \mathbb{Z}_0^{N-1}, \tag{1c}$$

$$0 \geq h_k(x_k,u_k) + E_k w_k, \; \forall\, k \in \mathbb{Z}_0^{N-1}, \tag{1d}$$

$$0 \geq h_N(x_N) + E_N w_N, \tag{1e}$$

$$w_{k,j} \in \mathbb{Z}, \quad \forall j \in \mathbb{Z}_1^{n_\mathrm{w}}, \; k \in \mathbb{Z}_0^N, \tag{1f}$$

where $\mathbb{Z}_a^b$ denotes the range of integers $a, a+1, \ldots, b$, $x_k \in \mathbb{R}^{n_\mathrm{x}}$ are real-valued states, $u_k \in \mathbb{R}^{n_\mathrm{u}}$ are real-valued controls and $w_k \in \mathbb{Z}^{n_\mathrm{w}}$ are integer decision variables. Therefore, the optimization variables include all states $X = [x_0^\top, \ldots, x_N^\top]^\top$, control inputs $U = [u_0^\top, \ldots, u_{N-1}^\top]^\top$ and integer variables $W = [w_0^\top, \ldots, w_N^\top]^\top$. In the MIOCP, (1a) defines the cost, the initial state value $\hat{x}_t$ is set by (1b), (1c) are the discrete-time system dynamics, (1d)-(1e) are the inequality constraints, and (1f) imposes the integrality constraints.

The MIOCP (1) is a non-convex MINLP, i.e., the optimization problem is non-convex even after relaxing the integrality constraints in (1f), e.g., due to the nonlinear system dynamics in (1c). As discussed in [7], global optimization algorithms for MINLPs often require convexity of the objective and constraint functions. For example, `BONMIN` [8] is a global solver for convex MINLPs and becomes a heuristic for non-convex MINLPs. Even though global optimization algorithms exist for non-convex MINLPs, e.g., using relaxations of factorable problems [9], they are usually computationally very expensive and hence generally not yet practical for real-time implementations of mixed-integer NMPC (MINMPC) [6].

In this paper, we instead focus on approximate or heuristic techniques to find feasible but (possibly) suboptimal solutions of (1) to enable real-time MINMPC applications. Global algorithms for convex MINLPs can be used to find approximate solutions to non-convex MINLPs, e.g., using outer approximation or hybrid branch-and-bound (hB&B) methods [8]. Recent work in [10] proposed a variant of the hB&B algorithm for real-time MINMPC based on convex MINLPs. Specifically for non-convex MINMPC, a variant of the real-time iterations (RTI) algorithm has been proposed based on outer convexification in combination with rounding schemes in [6]. However, when inequality constraints depend directly on the discrete decision variables as in (1), the latter approach requires solving mathematical programs with vanishing constraints, which are particularly challenging.

Sequential quadratic programming (SQP) methods form a popular technique to solve nonlinear programs (NLPs), e.g., within a B&B method for MINLPs [11]. A mixed-integer SQP (MISQP) algorithm was proposed in [12], [13] for general MINLPs, based on the solution of mixed-integer quadratic programming (MIQP) subproblems and a trust region method. Even though MIQPs are still $\mathcal{NP}$-hard in general, state of the art algorithms can efficiently solve a large range of MIQPs, e.g., using B&B-type methods [14], [15]. In addition, more recently, tailored methods have been developed for warm starting, early termination and pre-solve techniques in B&B methods for MIQP-based MPC of linear or piecewise-linear systems [16], [17].

The MISQP method in [12], [13] requires the use of a trust region radius for both continuous and integer op-

[1]Mitsubishi Electric Research Laboratories, Cambridge, MA, 02139, USA {quirynen,dicairano}@merl.com.

timization variables, and it relies on the assumption that integer variables have a *smooth* influence on the MINLP, i.e., incrementing an integer variable by one leads to a small change of function values. However, the latter assumption is generally not true for the MIOCP (1) of interest, as for example, it may include binary variables that have a large influence on the optimal control trajectories. In the present paper, based on our assumption of linear dependency on integer variables in (1), without loss of generality, we do not require the smoothness assumption from [12], [13] and we propose instead a novel variant of the MISQP algorithm, including a line search or trust region method applied only to the continuous optimization variables.

The paper is organized as follows. Section II introduces our proposed MISQP heuristic framework and presents the theoretical motivation. Section III proposes two particular variants of the MISQP algorithm, and Section IV describes the real-time feasible software implementation. Section V illustrates the closed-loop performance for an MINMPC case study, and Section VI concludes the paper.

II. MIXED-INTEGER SEQUENTIAL QUADRATIC PROGRAMMING (MISQP)

We rely on the following two assumptions.

Assumption 1: The MIOCP is formulated as in (1), where integer variables w_k enter linearly in the objective (1a), equality (1c) and inequality constraints (1d)-(1e).

Assumption 2: The nonlinear functions $l_k(\cdot)$, $m(\cdot)$ in the cost (1a), $\psi_k(\cdot)$ in the discrete-time dynamics (1c) and $h_k(\cdot)$, $h_N(\cdot)$ in the inequality constraints (1d)-(1e) are assumed to be C^2, i.e., twice continuously differentiable.

Assumption 1 will be important for our proposed MISQP approach but it can be imposed without loss of generality, since any MIOCP can be written in the structured form (1), possibly by defining auxiliary optimization variables. Assumption 2 is common in the literature to describe convergence results for standard SQP methods in constrained nonlinear optimization, e.g., see [18], [19].

Let us introduce the following compact notation for the MIOCP in (1) that needs to be solved

$$
\begin{aligned}
\mathcal{P}(y,z) \;:=\; \min_{y,z} \quad & f(y) + c^\top z && \text{(2a)}\\
\text{s.t.} \quad & g(y) + D\,z \;=\; 0, && \text{(2b)}\\
& h(y) + E\,z \;\leq\; 0, && \text{(2c)}\\
& z_j \in \mathbb{Z}, \qquad \forall j \in \mathbb{Z}_1^{n_z}, && \text{(2d)}
\end{aligned}
$$

where $y \in \mathbb{R}^{n_y}$ and $z \in \mathbb{Z}^{n_z}$ denote the continuous and integer optimization variables, i.e.,

$$
y = [x_0^\top, u_0^\top, \ldots, u_{N-1}^\top, x_N^\top]^\top, \quad z = [w_0^\top, \ldots, w_N^\top]^\top. \quad (3)
$$

Based on Assumption 1 and 2, we know that the integer optimization variables z enter linearly in the cost (2a) and constraints in (2b) and (2c), and that the nonlinear functions $f(\cdot)$, $g(\cdot)$ and $h(\cdot)$ in (2) are C^2 functions, respectively.

A. MIQP Subproblem for MISQP Algorithm

We propose an MISQP algorithm to solve the MINLP in (2) by solving a sequence of MIQP approximations,

$$
\begin{aligned}
&\mathcal{P}_{\mathrm{miqp}}(y^k, z^k) :=\\
&\min_{\Delta y, \Delta z} \quad \frac{1}{2}\Delta y^\top B(y^k)\,\Delta y + \nabla_y f(y^k)^\top \Delta y\\
&\qquad\qquad + c^\top (z^k + \Delta z) && \text{(4a)}\\
&\text{s.t.} \quad g(y^k) + \frac{\partial g}{\partial y}(y^k)\Delta y + D\,(z^k + \Delta z) \;=\; 0, && \text{(4b)}\\
&\qquad h(y^k) + \frac{\partial h}{\partial y}(y^k)\Delta y + E\,(z^k + \Delta z) \;\leq\; 0, && \text{(4c)}\\
&\qquad z_j^k + \Delta z_j \in \mathbb{Z}, \qquad \forall j \in \mathbb{Z}_1^{n_z}, && \text{(4d)}
\end{aligned}
$$

given the current solution guess (y^k, z^k) at the k^{th} iteration, and where $B(y^k) \succ 0$ is a positive definite matrix that approximates the Hessian of the Lagrangian [20]. Integer feasibility is imposed at each MISQP iteration due to the integrality constraints in (4d). In addition, the linear-quadratic objective approximation in (4a) and the constraint linearizations in (4b) and (4c) depend only on the continuous optimization variables y, due to Assumption 1.

Let us define a merit function for the MISQP algorithm, using the standard ℓ_1 penalty function to quantify constraint satisfaction applied to the MINLP (2)

$$
\phi(y; z, \rho) = F(y,z) + \rho\, \|G(y,z)\|_1 + \rho \sum_i \max\left(H_i(y,z), \epsilon\right), \quad (5)
$$

where $\rho > 0$ is the penalty parameter value, $\epsilon \geq 0$ is a small positive value and

$$
\begin{aligned}
&F(y,z) = f(y) + c^\top z,\\
&G(y,z) = g(y) + D\,z, \quad H(y,z) = h(y) + E\,z,
\end{aligned} \quad (6)
$$

are the compact notation for the objective and constraints. Our proposed merit function in (5) does not directly incorporate the integrality constraints (2d), since they are imposed exactly by each MIQP approximation in (4d). The ℓ_1 merit function in (5) is not differentiable but a directional derivative exists, which is sufficient for our purpose. The aim of the MISQP algorithm is to decrease the merit function along the search direction in order to ensure convergence.

B. Descent Property of MIQP Search Direction

The step search direction $(\Delta y, \Delta z)$, computed by solving the MIQP in (4), is not necessarily a descent direction for a merit function of the original MINLP in (2), due to the non-convexity of the integrality constraints in (4d). However, we will show that Δy is a descent direction for a merit function of the NLP that is obtained in (2) when fixing the values of the integer optimization variables to $z^{k+1} = z^k + \Delta z^k$. The latter property is important, since it can be used by line search or trust-region methods to guarantee progress with respect to the merit function.

Lemma 1: Given the solution $(\Delta y^k, \Delta z^k)$ to problem $\mathcal{P}_{\mathrm{miqp}}(y^k, z^k)$ in (4), the step direction Δy^k is equal to the step direction Δy that is the solution to problem $\mathcal{P}_{\text{miqp}}(y^k, z^{k+1})$, where $z^{k+1} = z^k + \Delta z^k$.

Proof: This result follows directly from z entering the MINLP (2) linearly, such that $(\Delta y^k, 0)$ is the solution to problem $\mathcal{P}_{\text{miqp}}(y^k, z^{k+1})$ if $(\Delta y^k, \Delta z^k)$ is the solution to problem $\mathcal{P}_{\text{miqp}}(y^k, z^k)$ in (4) and $z^{k+1} = z^k + \Delta z^k$. Specifically, the optimal values for Δy from the MIQP in (4) do not depend on the values for z^k. ∎

Based on the result in Lemma 1, it can be shown that the following descent property holds for the MIQP search direction and the ℓ_1 merit function in (5).

Theorem 1: The MISQP step direction Δy^k, computed as the solution to problem $\mathcal{P}_{\text{miqp}}(y^k, z^k)$ in (4), is a descent direction for the merit function $\phi(y; z^{k+1}, \rho)$ in (5), i.e., for a sufficiently large parameter value $\rho > 0$,

$$\nabla_y \phi(y^k; z^{k+1}, \rho)^\top \Delta y^k < 0. \tag{7}$$

Proof: Let us show this result for the proposed MISQP algorithm, considering an equality constrained MINLP (2), even though it can easily be extended to the inequality constrained case, e.g., by using a slack reformulation as in [20]. For this case, the ℓ_1 penalty function reads as

$$\phi(y; z, \rho) = F(y, z) + \rho \, \|G(y, z)\|_1. \tag{8}$$

Following the standard SQP result in [20, Theorem 18.2], the directional derivative can be shown to read as

$$\begin{aligned} \nabla_y \phi(y^k; z^{k+1}, \rho)^\top \Delta y^k = \nabla_y F(y^k, z^{k+1})^\top \Delta y^k \\ - \rho \, \|G(y^k, z^{k+1})\|_1. \end{aligned} \tag{9}$$

Based on the optimality conditions for the MIQP in (4)

$$\begin{aligned} B(y^k)\, \Delta y^k - \frac{\partial g}{\partial y}(y^k)^\top \lambda^{k+1} &= -\nabla_y F(y^k, z^{k+1}) \\ \frac{\partial g}{\partial y}(y^k) \Delta y^k &= -G(y^k, z^{k+1}), \end{aligned} \tag{10}$$

the following inequality can be derived from (9)

$$\begin{aligned} \nabla_y \phi(y^k; z^{k+1}, \rho)^\top \Delta y^k \leq -{\Delta y^k}^\top B(y^k)\, \Delta y^k \\ - \left(\rho - \|\lambda^{k+1}\|_\infty\right) \, \|G(y^k, z^{k+1})\|_1. \end{aligned} \tag{11}$$

It follows from (11) that Δy^k is a descent direction for the merit function $\phi(y; z^{k+1}, \rho)$ if $\Delta y^k \neq 0$, the Hessian approximation $B(y^k) \succ 0$ is positive definite and the penalty parameter value $\rho > \|\lambda^{k+1}\|_\infty$. ∎

C. MISQP Convergence and Heuristics

As pointed out in [12], [13], it is important to stress that one cannot prove convergence of an MISQP algorithm to the global solution of the MINLP (2). Instead, we propose a novel heuristic MISQP algorithm based on the descent property described in Theorem 1. Assuming that the sequence of values for the integer optimization variables converges to a fixed set of values, i.e., $z^k \to \overline{z}$ based on a sequence of updates $z^{k+1} = z^k + \Delta z^k$ where Δz^k corresponds to the MIQP solution in (4), then the following result for the convergence of the continuous variables y holds.

Proposition 1: If an index $\overline{k}$ exists for which $z^{k+1} = z^k = \overline{z}$ for all $k \geq \overline{k}$, and there exists at least one set of values $\overline{y}$ for which $G(\overline{y}, \overline{z}) = 0$ and $H(\overline{y}, \overline{z}) \leq 0$, then the MISQP method converges to a local minimizer $(y^\star, \overline{z})$ of the MINLP (2) for fixed integer values $z = \overline{z}$, i.e., $y^\star$ is a local minimizer for the continuous program $\mathcal{P}(y, \overline{z})$.

Proof: The latter proposition follows directly from Theorem 1 and the convergence analysis of standard SQP methods [20], using line search or trust-region globalization techniques, applied to the continuous nonlinear program $\mathcal{P}(y, \overline{z})$ in (2) for fixed integer values. ∎

The proposition is based on two main assumptions, namely that the sequence of integer values converges $z^k \to \overline{z}$ and that a feasible solution exists for the continuous nonlinear program $\mathcal{P}(y, \overline{z})$ in (2). Therefore, our proposed heuristic MISQP algorithm works well when an integer feasible solution guess $z^{k+1} = z^k + \Delta z^k$ to the MIQP in (4) is likely to result in a feasible nonlinear optimization problem $\mathcal{P}(y, z^{k+1})$. In this case, one could enforce the integer variables to remain fixed in the subsequent MISQP iterations in order to quickly find a feasible, but possibly suboptimal, solution guess for the original MINLP in (2). A similar strategy of (temporarily) fixing the integer variables was used in [13] to prevent cycling in the algorithm.

Remark 1: If a problem $\mathcal{P}(y, \overline{z})$ is detected to be infeasible, additional constraints could be added to the MIQP approximation in (4) to avoid revisiting the integer solution $\overline{z}$ in subsequent MISQP iterations. An example of such a cutting strategy for MINLPs can be found in [13]. However, this is outside the scope of the present paper.

Remark 2: Similar to the discussion in [20] for standard SQP methods, one can show that the merit function (5) is exact, i.e., a local minimizer $(y^\star, \overline{z})$ of the MINLP (2), for fixed integer values $z = \overline{z}$, is a local minimizer of $\phi(y; \overline{z}, \rho)$ for a sufficiently large penalty value $\rho > \rho^\star$.

III. MISQP ALGORITHM IMPLEMENTATION

In this section, we present two alternative implementations of the MISQP algorithm that are based on standard line search and trust-region methods, as can be found in [20], adapted to our proposed MISQP framework for MINLPs.

A. Line Search MISQP Method (MISQP-LS)

Based on the descent property in Theorem 1, a line search method computes a step size $\alpha^k \in (0, 1]$ in order to update the continuous optimization variables $y^{k+1} = y^k + \alpha^k \Delta y^k$, for which the sufficient decrease condition holds

$$\begin{aligned} \phi(y^k + \alpha^k \Delta y^k; z^{k+1}, \rho) \leq \phi(y^k; z^{k+1}, \rho) \\ + \alpha^k \eta \, \nabla_y \phi(y^k; z^{k+1}, \rho)^\top \Delta y^k, \quad \eta \in (0, 1), \end{aligned} \tag{12}$$

which is based on the Armijo condition for unconstrained optimization [20]. A full step update for the integer optimization variables, i.e., $z^{k+1} = z^k + \Delta z^k$ is used to satisfy the integrality constraints (2d) at each MISQP iteration. Algorithm 1 describes the resulting line search MISQP method that aims at finding a feasible, but possibly suboptimal, solution to the MINLP in (2). As discussed in Section II-C, Algorithm 1 includes a heuristic that enforces the integer variables to remain fixed after a particular number N_{miqp}

Algorithm 1 Line Search MISQP Method for MINLP (2).

Input: Initial guess (y^0, z^0), $\eta, \beta \in (0, 1)$ and $\epsilon_{\mathrm{tol}} > 0$.
$k \leftarrow 0$.
while $\|r(y^k, z^k)\| > \epsilon_{\mathrm{tol}}$ **do**
    if $k \geq N_{\mathrm{miqp}}$ **then**
        Solve QP (4) for fixed $\Delta z = 0$ to compute Δy^k.
    else
        Solve MIQP in (4) to compute $(\Delta y^k, \Delta z^k)$.
    end if
    Update integer variables $z^{k+1} \leftarrow z^k + \Delta z^k$.
    Compute ρ to ensure (7) and $\alpha^k \leftarrow 1$.
    while (12) not satisfied **do**
        $\alpha^k \leftarrow \tilde{\beta}\alpha^k$ for $\tilde{\beta} \in (0, \beta]$.
    end while
    $y^{k+1} \leftarrow y^k + \alpha^k \Delta y^k$ and $k \leftarrow k + 1$.
end while

of MISQP iterations. If the value $N_{\mathrm{miqp}} = \infty$, then the integer variables are updated as $z^{k+1} = z^k + \Delta z^k$, based on the MIQP solution of (4) at each MISQP iteration. The termination condition in Algorithm 1 is based on the norm of the Karush-Kuhn-Tucker (KKT) residual, $\|r(y^k, z^k)\|$, excluding the integrality conditions in (2d) that are satisfied at each MISQP iteration.

B. Trust-Region MISQP Method (MISQP-TR)

A second approach is the trust-region MISQP method, where at each step we solve the MIQP subproblem

$$\min_{\Delta y, \Delta z} \quad (4a) \tag{13a}$$

$$\text{s.t.} \quad (4b) - (4d), \tag{13b}$$

$$\|M\Delta y\|_p \leq d_k, \tag{13c}$$

including additional constraints on the size of the update step Δy in (13c), where $d_k > 0$ is the trust-region radius, $M \succeq 0$ denotes the scaling matrix, and $p \in [1, \infty]$. In what follows $p = \infty$, i.e., we use the ∞-norm. The ratio R_k of actual to predicted reduction, which plays a critical role in standard trust-region methods [19], is defined as

$$R_k = \frac{\phi(y^k; z^{k+1}, \rho) - \phi(y^k + \Delta y^k; z^{k+1}, \rho)}{\phi^k_{\mathrm{QP}}(0; z^{k+1}, \rho) - \phi^k_{\mathrm{QP}}(\Delta y^k; z^{k+1}, \rho)}, \tag{14}$$

where $\phi(\cdot)$ denotes the merit function in (5) and $\phi^k_{\mathrm{QP}}(\cdot)$ denotes the linearization-based merit function as follows

$$\begin{aligned} \phi^k_{\mathrm{QP}}(\Delta y; z, \rho) &= \frac{1}{2}\Delta y^\top B(y^k)\,\Delta y + \nabla_y f(y^k)^\top \Delta y \\ &+ \rho \left\| G(y^k, z) + \frac{\partial G}{\partial y}(y^k, z)\,\Delta y \right\|_1 \\ &+ \rho \sum_i \max\left(H_i(y^k, z) + \frac{\partial H_i}{\partial y}(y^k, z)\,\Delta y, \epsilon \right). \end{aligned} \tag{15}$$

Algorithm 2 describes the resulting trust-region MISQP method. The trust-region radius update can be performed in many different ways, see [18], [19]. Unlike standard trust-region SQP methods, including the MISQP algorithm in [12], Algorithm 2 aims at shrinking the trust-region radius if the inequality constraint (13c) is inactive at the MIQP solution, i.e., $\|M\Delta y^k\|_p < \gamma_1\, d_k < d_k$. The latter modification enforces a desirable property of the trust-region MISQP method in Algorithm 2 that tight inequality constraints $\|M\Delta y\|_p \leq d_k$ in (13c) can be used to solve the MIQP more efficiently, as discussed also below.

Algorithm 2 Trust-Region MISQP Method for MINLP (2).

Input: Initial guess (y^0, z^0), values $0 < \eta_1 \leq \eta_2 < 1$, $0 < \gamma_1 \leq \gamma_2 < 1 \leq \gamma_3$, $0 < \underline{d} < \overline{d}$ and $d_0 \in [\underline{d}, \overline{d}]$.
$k \leftarrow 0$.
while $\|r(y^k, z^k)\| > \epsilon_{\mathrm{tol}}$ **do**
    if $k \geq N_{\mathrm{miqp}}$ **then**
        Solve QP (13) for fixed $\Delta z = 0$ to compute Δy^k.
    else
        Solve MIQP in (13) to compute $(\Delta y^k, \Delta z^k)$.
    end if
    Compute ratio R_k in (14).
    if $R_k \geq \eta_1$ **then** ▷ Accept step
        $y^{k+1} \leftarrow y^k + \Delta y^k$ and $z^{k+1} \leftarrow z^k + \Delta z^k$.
    else ▷ Reject step
        $y^{k+1} \leftarrow y^k$ and $z^{k+1} \leftarrow z^k$.
    end if
    `Trust-region radius update:`
    $d_{k+1} \leftarrow d_k$.
    if $R_k < \eta_1$ **then**
        $d_{k+1} \leftarrow \max\left(\gamma_1 \|M\Delta y^k\|_p, \underline{d}\right)$. ▷ Shrink
    else if $\|M\Delta y^k\|_p < \gamma_1\, d_k$ **then**
        $d_{k+1} \leftarrow \max\left(\gamma_2\, d_k, \underline{d}\right)$. ▷ Shrink
    else if $R_k > \eta_2$ and $\|M\Delta y^k\|_p = d_k$ **then**
        $d_{k+1} \leftarrow \min\left(\gamma_3\, d_k, \overline{d}\right)$. ▷ Grow
    end if
end while

For simplicity of presentation, Algorithm 1 and 2 assume that the MIQP/QP subproblem at each iteration is feasible, which can be guaranteed by introducing slack variables and an exact penalty function in the objective [12], [13].

C. Discussion on MISQP Variants for MINLPs

Algorithm 1 and 2 each have their own advantages and disadvantages as MISQP-type heuristics to approximately solve the MINLP in (2). In the line search MISQP of Algorithm 1, each MIQP solution is used to try and reduce the merit function, which is desirable under the assumption that the computational cost for the MIQP solution is large compared to that of the line search. However, the computational cost of solving the MIQP at each iteration of Algorithm 1 can still be relatively large due to its combinatorial nature, even when the optimal solution to (4) corresponds to the trivial solution $\Delta z = 0$. For example, branch-and-bound methods may spend the majority of their time proving that a particular solution is globally optimal [14].

The trust-region MISQP method of Algorithm 2 includes the additional constraints on the step size $\|M\Delta y\|_p$ for the continuous optimization variables in (13c). State of the art

MIQP solvers can use the latter constraints to considerably reduce the search space for the integer optimization variables, e.g., using domain propagation in the pre-solve routine to fix variables before solving the MIQP or before solving the relaxed problem in any tree node [15]. In Section V, we compare both MISQP variants and their performance based on numerical simulation results.

IV. REAL-TIME MIXED-INTEGER NMPC

In this section, we discuss how the proposed MISQP framework in Algorithm 1 and 2 can result in real-time feasible implementations of MINMPC.

A. MISQP Software Implementation

First, we briefly describe the MISQP software implementation that is used in the numerical results of Section V. The efficient evaluation of nonlinear functions and their derivatives, for the preparation of the MIQP/QP subproblem, is based on algorithmic differentiation (AD) and C code generation in `CasADi` [21]. Each MIQP is solved using a branch-and-bound method, including warm starting and pre-solve techniques in [16], and using the active-set based interior point method in `ASIPM` [22] to solve QP relaxations, both of which have been designed specifically for real-time MPC in embedded platforms. In addition, early termination and infeasibility detection for `ASIPM` is implemented based on our recent work in [17].

B. Real-time MISQP Algorithm

SQP methods are popular for the implementation of NMPC, due to their desirable warm starting properties [1]. In addition, based on the realization that NMPC requires the solution of a parametric optimization problem at each sampling instant, tailored continuation methods exist for NMPC that adapt the solution guess to the most recent state estimate in real time. For example, the real-time iteration (RTI) algorithm was originally proposed in [23], based on a single SQP iteration at each control time step.

Warm started from the solution at the previous control time step, it can be expected that the proposed MISQP algorithm relatively quickly finds good feasible solutions to the MIOCP in (1). Under strict timing constraints, and motivated by the RTI algorithm in [23], one could perform a single iteration of Algorithm 1 or 2 at each sampling instant of MINMPC. Instead, in the present paper, we aim to let Algorithm 1 or 2 converge to a feasible solution of the MIOCP (1), potentially restricting the number of MIQP solutions at each control time step. For example, when setting $N_{\mathrm{miqp}} = 1$, Algorithm 1 and 2 become heuristics that compute an integer solution guess z^1 based on the MIQP in the first MISQP iteration, warm started from the solution at the previous control time step, and then solve the resulting nonlinear program $\mathcal{P}(y, z^1)$ by a sequence of relatively cheap QP solutions.

In Section V, we illustrate the performance of the resulting MINMPC implementations for different choices of the parameter value N_{miqp} in Algorithm 1 and 2.

C. Alternative Smooth NLP Relaxations

In what follows, we compare our proposed MISQP methods against `BONMIN` [8], which is a global solver for convex MINLPs but it becomes a heuristic for non-convex MINLPs. In addition, we compare against two alternative implementations based on the solution of a smooth NLP at each control time step, using the state of the art `Ipopt` solver [24]. The first approach solves the NLP that results from relaxing the integrality constraints in (1f) at each MINMPC time step, e.g., $w_{k,j} \in \{0, 1\}$ is relaxed as $w_{k,j} \in [0, 1] \subset \mathbb{R}$. A rounding strategy can be used in order to ensure integer feasibility [6]. The second approach solves an NLP that enforces the integrality constraints in (1f) based on smooth nonlinear equality constraints. For example, a binary variable $w_{k,j} \in \{0, 1\}$ can be reformulated as

$$w_{k,j}(1 - w_{k,j}) = 0, \quad w_{k,j} \in [0, 1] \subset \mathbb{R}. \tag{16}$$

As discussed in [6, Section 2.2.5], the latter approach could be improved by using a homotopy technique at the cost of solving multiple NLPs at each MINMPC time step.

V. MINMPC CASE STUDY: VEHICLE CONTROL

We illustrate the performance of the proposed MISQP heuristic algorithm for a vehicle control case study based on real-time MINMPC, using a piecewise-affine (PWA) approximation of the tire force model as in [25].

A. Vehicle Control Problem Formulation

We use a single-track vehicle model that includes the position $(p^{\mathrm{x}}, p^{\mathrm{y}})$, the longitudinal velocity v^{x}, lateral velocity v^{y}, yaw angle ψ and yaw rate $\dot{\psi}$ as states, i.e., $n_{\mathrm{x}} = 6$. The inputs to the vehicle model are the front and rear wheel speeds ω_f, ω_r and the tire-wheel angle δ, i.e., $n_{\mathrm{u}} = 3$. The single-track model lumps together the left and right wheel on each axle, and the resulting nonlinear system dynamics are described in detail in [26]. The slip angles α_i and slip ratios λ_i are defined as

$$\alpha_i = -\arctan\left(\frac{v_i^{\mathrm{y}}}{v_i^{\mathrm{x}}}\right), \quad \lambda_i = \frac{R_w \omega_i - v_i^{\mathrm{x}}}{v_i^{\mathrm{x}}}, \; i \in \{f, r\}. \tag{17}$$

For simplicity, the longitudinal tire forces are modeled as the linear functions $F_i^{\mathrm{x}} = C_i^{\mathrm{x}} \lambda_i$ based on the longitudinal stiffness C_i^{x} and the slip ratio λ_i for front and rear tires, $i \in \{f, r\}$. Similar to the PWA model based on experimental tire friction data in [25], we use a PWA approximation of a nonlinear Pacejka curve for the lateral tire force with respect to the slip angle α_i:

$$F_i^{\mathrm{y}} = \begin{cases} d_{i,-j} + C_{i,-j}^{\mathrm{y}} \alpha_i, & \text{if } \alpha_i \in -[\overline{\alpha}_{i,j}, \overline{\alpha}_{i,j-1}), \; \forall j \in \mathbb{Z}_1^n, \\ d_{i,0} + C_{i,0}^{\mathrm{y}} \alpha_i, & \text{if } \alpha_i \in [-\overline{\alpha}_{i,0}, \overline{\alpha}_{i,0}), \\ d_{i,j} + C_{i,j}^{\mathrm{y}} \alpha_i, & \text{if } \alpha_i \in [\overline{\alpha}_{i,j-1}, \overline{\alpha}_{i,j}), \; \forall j \in \mathbb{Z}_1^n, \end{cases} \tag{18}$$

with $2\,n + 1$ regions or modes and slip angle values $\overline{\alpha}_{i,0} < \cdots < \overline{\alpha}_{i,n-1} < \overline{\alpha}_{i,n}$, for $i \in \{f, r\}$. Similar to [25], we further use a PWA approximation with only 3 regions, i.e., $n = 1$, to represent negative saturation, linear region and positive saturation of the lateral tire force. The use of the PWA tire model in (18) considerably reduces the nonlinearity of the OCP formulation, but it requires the use of binary decision variables and results in an MIOCP of the form in (1). This hybrid system is used as benchmark example for the proposed MINMPC implementation, and it is outside the scope of the present paper whether the MINMPC outperforms an NMPC controller based directly on the Pacejka curve [26].

B. Convex-Hull Problem Formulation

We introduce a convex-hull formulation for the PWA model in (18) to obtain tight convex relaxations of the resulting MIOCP as discussed, e.g., in [2]. We first define the continuous optimization variables $\alpha_{i,-n}, \dots, \alpha_{i,0}, \dots, \alpha_{i,n} \in \mathbb{R}$ and the binary optimization variables $b_{i,-n}, \dots, b_{i,0}, \dots, b_{i,n} \in \{0,1\}$ for each of the $2\,n+1$ regions of the PWA model in (18). The lateral tire force can then be defined as

$$F_i^{\mathrm{y}} = \sum_{j=-n}^{n} d_{i,j}\, b_{i,j} + C_{i,j}^{\mathrm{y}}\, \alpha_{i,j}, \tag{19}$$

where $b_{i,j}=1$ if slip angle $\alpha_{i,j}=\alpha_i$ lies in the j^{th} region, otherwise $b_{i,j}=0$ and $\alpha_{i,j}=0$. Therefore, each of the variables is constrained as follows

$$-\overline{\alpha}_{i,j}\, b_{i,-j} \;\le\; \alpha_{i,-j} \;\le -\overline{\alpha}_{i,j-1}\, b_{i,-j},\ \forall j \in \mathbb{Z}_1^n, \tag{20a}$$

$$-\overline{\alpha}_{i,0}\, b_{i,0} \;\le\; \alpha_{i,0} \;\le\; \overline{\alpha}_{i,0}\, b_{i,0}, \tag{20b}$$

$$\overline{\alpha}_{i,j-1}\, b_{i,j} \;\le\; \alpha_{i,j} \;\le\; \overline{\alpha}_{i,j}\, b_{i,j},\ \forall j \in \mathbb{Z}_1^n, \tag{20c}$$

where the sum of binary variables $b_{i,j} \in \{0,1\}$ is equal to one and the slip angle definition from (17) reads as

$$\sum_{j=-n}^{n} b_{i,j} = 1, \quad \sum_{j=-n}^{n} \alpha_{i,j} = -\arctan\left(\frac{v_i^{\mathrm{y}}}{v_i^{\mathrm{x}}}\right). \tag{21}$$

C. Open-loop MIOCP: Convergence MISQP Algorithm

The resulting MIOCP is of the form in (1), based on an explicit 4^{th}-order Runge-Kutta discretization of the nonlinear single-track vehicle dynamics, with 3 fixed integration steps in each control interval of $T_{\mathrm{s}} = 100$ ms, and using a standard least squares tracking stage cost in (1a) as follows

$$l_k(x_k, u_k) = \|x_k - x_{\mathrm{ref},k}\|_Q^2 + \|u_k - u_{\mathrm{ref},k}\|_R^2, \tag{22}$$

corresponding to tracking of a lane change reference trajectory similar to the OCP formulation in [26]. The auxiliary variables F_i^{y} and $\alpha_{i,j}$ for $i \in \{f,r\}$ and $j \in \mathbb{Z}_{-n}^n$ are included in the continuous control inputs, $n_{\mathrm{u}} = 7 + 4\,n$. The integer decision variables in (1) consist of the binary variables $b_{i,j}$ for $i \in \{f,r\}$ and $j \in \mathbb{Z}_{-n}^n$, i.e., $n_{\mathrm{w}} = 2 + 4\,n$. The MIOCP constraints include simple bounds on the control inputs and on the lateral position of the vehicle, as well as the additional constraints in (19)-(21). We use a control horizon length $N = 10$ in the MIOCP (1), resulting in a total of 176 continuous and 66 binary decision variables.

Figure 1 shows the number of iterations and computation times for solving the resulting MIOCP, based on the proposed MISQP-LS and MISQP-TR methods in Algorithm 1 and 2, respectively. These results were obtained by solving the open-loop MIOCP for 200 randomly generated initial state values $\hat{x}_t$ in (1b). Figure 1 presents both the average (solid lines) and worst-case (dashed lines) computation times for the proposed MISQP algorithm with $N_{\mathrm{miqp}} = \infty$ (on the left) and $N_{\mathrm{miqp}} = 1$ (on the right). It can be observed from Figure 1 that both algorithms, using either the line search or trust-region method, result in a similar number of MISQP iterations for this MIOCP example.

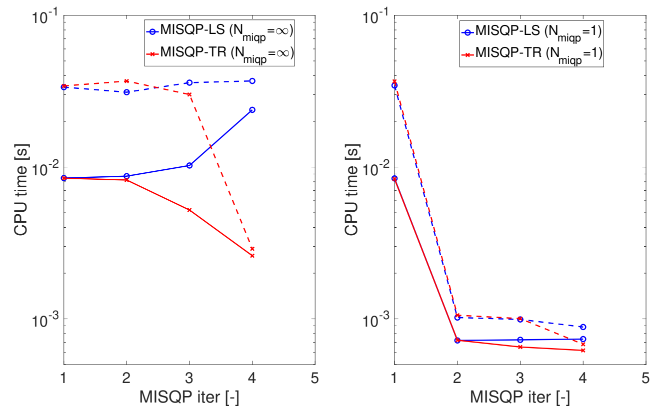

Fig. 1. Computation times for each MISQP iteration to solve the open-loop MIOCP for 200 randomly generated initial state values $\hat{x}_t$: including both average (solid lines) and maximum (dashed lines) timing results.

However, as discussed also in Section III-C, it can be observed on the left hand of Figure 1, corresponding to $N_{\mathrm{miqp}} = \infty$, that both the average and worst-case computation times decrease over subsequent MISQP iterations due to the overall decreasing trust-region radius based on Algorithm 2. The worst-case computation time is more or less constant over subsequent iterations of the MISQP-LS method in Algorithm 1 when $N_{\mathrm{miqp}} = \infty$. When $N_{\mathrm{miqp}} = 1$ in Algorithm 1 and 2, the first MISQP iteration requires the solution of an MIQP and all subsequent iterations are based on QP solutions, such that the computation times are considerably smaller for all iterations after the first one.

D. Closed-loop MINMPC Simulation Results

Finally, let us illustrate the closed-loop performance of the MINMPC controller, using our proposed MISQP heuristic, compared against `BONMIN` [8] and `Ipopt` [24] based on the relaxed NLP solution as described in Section IV-C. In addition to the number of MISQP iterations and total computation time per control time step, we are interested in the closed-loop cost defined as follows

$$\text{Cost} = \sum_{k} \left(\|x_k - x_{\mathrm{ref},k}\|_Q^2 + \|u_k - u_{\mathrm{ref},k}\|_R^2 \right), \tag{23}$$

as the control performance metric. Table I shows the closed-loop simulation results for the MINMPC-based controller to track a single lane change reference trajectory on a snow-covered road surface at a vehicle speed of 12 m/s.

The proposed MISQP algorithm results in the best closed-loop control performance for this particular case study, as

TABLE I

CLOSED-LOOP SIMULATION RESULTS FOR MINMPC TO TRACK A LANE CHANGE REFERENCE ON A SNOW-COVERED ROAD SURFACE AT A VEHICLE SPEED OF 12 M/S, USING EITHER THE PROPOSED MISQP ALGORITHMS OR THE STATE OF THE ART BONMIN [8] AND IPOPT [24] SOLVERS.

	BONMIN	IPOPT		MISQP ($N_{\text{miqp}} = \infty$)		MISQP ($N_{\text{miqp}} = 1$)	
	MINLP	NLP using (16)	NLP relax	LS	TR	LS	TR
Iterations (mean/max) [-]	-	-	-	1.9/3.0	2.1/4.0	1.9/3.0	2.1/4.0
Total closed-loop cost [-]	3.023	2.212	2.087	1.222	1.222	1.222	1.222
CPU time (mean/max) [ms]	1233.5/7762.4	187.4/588.2	165.0/237.7	19.2/74.5	18.5/81.2	8.2/27.8	7.8/26.3

can be observed from Table I based on the closed-loop cost. In addition, both the average and worst-case computation times are reduced greatly due to the MISQP algorithms compared to the BONMIN solver and even to Ipopt to solve the relaxed NLPs as described in Section IV-C. Unlike the open-loop MISQP convergence results in Figure 1, it can be observed from Table I that both Algorithm 1 and 2 result in a very similar closed-loop performance, due to the effect of warm starting the solver from one control time step to the next as discussed in Section IV-B.

Unlike the implementations using either the BONMIN or Ipopt solvers, the proposed MISQP method allows a real-time feasible MINMPC implementation with sampling time of $T_{\text{s}} = 100$ ms. In addition, setting $N_{\text{miqp}} = 1$ reduces the worst-case computation times even further below 30 ms, which would allow a sampling time of 50 ms that is desirable in real vehicle experiments [25], [26].

VI. CONCLUSIONS AND OUTLOOK

This paper presents an approach for real-time feasible implementation of mixed-integer nonlinear model predictive control (MINMPC), based on mixed-integer sequential quadratic programming (MISQP). A theoretical motivation is provided for the proposed MISQP framework, and two algorithmic variants are presented. Based on a preliminary software implementation, the performance of the MISQP algorithm is illustrated for a vehicle control case study using a piecewise-affine tire model. Since the proposed MISQP approach is a heuristic algorithm, its convergence cannot be always guaranteed, but it has shown good performance and robustness in simulations, and we plan to further validate it on challenging real-world control applications.

REFERENCES

[1] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl, *Model Predictive Control: Theory, Computation, and Design*, 2nd ed. Nob Hill, 2017.

[2] T. Marcucci and R. Tedrake, “Mixed-integer formulations for optimal control of piecewise-affine systems,” in *Proc. of the 22nd ACM Inter. Conf. on Hybrid Systems*, New York, NY, USA, 2019, p. 230–239.

[3] A. Walsh, S. Di Cairano, and A. Weiss, “MPC for coupled station keeping, attitude control, and momentum management of low-thrust geostationary satellites,” in *Proc. American Control Conf. (ACC)*, 2016, pp. 7408–7413.

[4] D. Ioan, I. Prodan, S. Olaru, F. Stoican, and S.-I. Niculescu, “Mixed-integer programming in motion planning,” *Annual Reviews in Control*, 2020.

[5] Y. E. Sahin, R. Quirynen, and S. Di Cairano, “Autonomous vehicle decision-making and monitoring based on signal temporal logic and mixed-integer programming,” in *Proc. American Control Conf. (ACC)*, 2020, pp. 454–459.

[6] C. Kirches, *Fast numerical methods for mixed-integer nonlinear model-predictive control*. Springer, 2011.

[7] P. Belotti, C. Kirches, S. Leyffer, J. Linderoth, J. Luedtke, and A. Mahajan, “Mixed-integer nonlinear optimization,” *Acta Numerica*, vol. 22, p. 1, 2013.

[8] P. Bonami, L. T. Biegler, A. R. Conn, G. Cornuéjols, I. E. Grossmann, C. D. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya, and A. Wächter, “An algorithmic framework for convex mixed integer nonlinear programs,” *Discrete Optimization*, vol. 5, no. 2, pp. 186–204, 2008.

[9] M. Tawarmalani and N. V. Sahinidis, *Convexification and global optimization in continuous and mixed-integer nonlinear programming: theory, algorithms, software, and applications*. Springer Science & Business Media, 2002, vol. 65.

[10] M. De Mauri, W. Van Roy, J. Gillis, J. Swevers, and G. Pipeleers, “Real time iterations for mixed-integer model predictive control,” in *2020 European Control Conference (ECC)*, 2020, pp. 699–705.

[11] S. Leyffer, “Integrating SQP and branch-and-bound for mixed integer nonlinear programming,” *Computational optimization and applications*, vol. 18, no. 3, pp. 295–309, 2001.

[12] O. Exler and K. Schittkowski, “A trust region SQP algorithm for mixed-integer nonlinear programming,” *Optimization Letters*, vol. 1, no. 3, pp. 269–280, 2007.

[13] O. Exler, T. Lehmann, and K. Schittkowski, “A comparative study of SQP-type algorithms for nonlinear and nonconvex mixed-integer optimization,” *Mathematical Programming Computation*, vol. 4, no. 4, pp. 383–412, 2012.

[14] C. A. Floudas, *Nonlinear and mixed-integer optimization: fundamentals and applications*. Oxford University Press, 1995.

[15] T. Achterberg, R. E. Bixby, Z. Gu, E. Rothberg, and D. Weninger, “Presolve reductions in mixed integer programming,” *INFORMS J. Comput.*, vol. 32, no. 2, pp. 473–506, 2020.

[16] P. Hespanhol, R. Quirynen, and S. Di Cairano, “A structure exploiting branch-and-bound algorithm for mixed-integer model predictive control,” in *Proc. European Control Conf. (ECC)*, 2019, pp. 2763–2768.

[17] J. Liang, S. D. Cairano, and R. Quirynen, “Early termination of convex QP solvers in mixed-integer programming for real-time decision making,” *IEEE Control Syst. Lett.*, vol. 5, no. 4, pp. 1417–1422, 2021.

[18] Y.-x. Yuan, “On the convergence of a new trust region algorithm,” *Numerische Mathematik*, vol. 70, no. 4, pp. 515–539, 1995.

[19] P. L. Toint, “Non-monotone trust-region algorithms for nonlinear optimization subject to convex constraints,” *Mathematical programming*, vol. 77, no. 3, pp. 69–94, 1997.

[20] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed., ser. Operations Research and Financial Engineering. Springer, 2006.

[21] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl., “Casadi–a software framework for nonlinear optimization and optimal control.” *Mathematical Programming Computation*, 2018.

[22] J. Frey, S. Di Cairano, and R. Quirynen, “Active-set based inexact interior point QP solver for model predictive control,” in *Proc. IFAC World Congr.*, 2020.

[23] M. Diehl, H. G. Bock, and J. P. Schlöder, “A real-time iteration scheme for nonlinear optimization in optimal feedback control,” *SIAM Journal on Control and Optimization*, vol. 43, no. 5, pp. 1714–1736, 2005.

[24] A. Wächter and L. T. Biegler, “On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming,” *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006.

[25] S. Di Cairano, H. E. Tseng, D. Bernardini, and A. Bemporad, “Vehicle yaw stability control by coordinated active front steering and differential braking in the tire sideslip angles domain,” *IEEE Trans. Control Syst. Technol.*, vol. 21, no. 4, pp. 1236–1248, 2013.

[26] K. Berntorp, R. Quirynen, T. Uno, and S. D. Cairano, “Trajectory tracking for autonomous vehicles on varying road surfaces by friction-adaptive nonlinear model predictive control,” *Vehicle System Dynamics*, vol. 58, no. 5, pp. 705–725, 2020.