Control Engineering Practice 97 (2020) 104352

Contents lists available at ScienceDirect

Control Engineering Practice

journal homepage: www.elsevier.com/locate/conengprac

Quasi-translations for fast hybrid nonlinear model predictive control ☆

Anson Maitland *, John McPhee

Department of Systems Design Engineering, University of Waterloo, 200 University Ave W, Waterloo, ON N2L 3G1, Canada

ARTICLE INFO

Keywords:
Model predictive control
Hybrid control
Nonlinear control

ABSTRACT

We present the quasi-translation (QT) strategy for mixed-integer model predictive control (MPC) problems. This strategy handles the complexity introduced by discrete controls in a simple manner that takes advantage of the sequential nature of MPC optimization problems. Some general criteria is provided guiding their application. The QT strategy can be used to balance controller performance, chatter and turnaround time. We also introduce an approach to handle hybrid models with state-dependent switches for use with a symbolic formulation of the MPC problem that integrates seamlessly with the QT strategy.

We investigate the QT strategy for a variety of hybrid nonlinear MPC problems including the classic benchmark Lotka–Volterra fishing problem, a diesel airpath control problem and longitudinal vehicle control. Results show the QT strategy consistently yields a reduction in turnaround times with excellent or minimally degraded performance as compared to competing strategies.

1. Introduction

Nonlinear Model Predictive Control (NMPC) is a feedback-based control strategy that has garnered significant attention from academia and industry. A serious hurdle to its widespread adoption is the hard real-time constraint of any real system, which conflicts with the increased computational burden of NMPC compared to classical controllers like PID (Cairano & Kolmanovsky, 2018). Significant work has been devoted to accelerating the turnaround times (TATs) of NMPCs, primarily at the solver level (Andersson, 2013; Diehl, Ferreau, & Haverbeke, 2009; Frasch, Sager, & Diehl, 2015; Kirches, Wirsching, Sager and Bock, 2010; Quirynen, Vukov, & Diehl, 2015; Vukov, Domahidi, Ferreau, Morari, & Diehl, 2013). However, there remain classes of problems that are still very challenging for real-time NMPCs, such as those with hybrid behaviour. The quasi-translation (QT) approach is introduced in effort to alleviate this challenge and focuses on the hybrid aspect of the problem meaning previous work on specialized methods to handle the nonlinear programming (NLP) problem of MPCs can be used in tandem.

Traditionally, the tools of control theory have been developed to handle systems governed by smooth dynamics with smooth inputs. In recent years, much attention has been devoted to the optimal control of hybrid systems that exhibit both smooth and discrete behaviours; see Zhu and Antsaklis (2015) and the references therein. These systems combine regimes with smooth dynamics along with a discrete switching logic. Controlling hybrid systems in an optimal fashion presents both a theoretical and practical challenge as such systems are ubiquitous. In the field of MPC, hybrid system control has also been a recent focus of research, though much of the theoretical results are limited, as they pertain only to piecewise affine systems (Camacho, Ramírez, Limón, La Peña, & Alamo, 2010). One strategy in MPC has been to approximate a nonlinear model by a switched linear model in order to achieve fast TATs, *e.g.* Andrikopoulos, Nikolakopoulos, and Manesis (2013). Applications of hybrid MPC are refrigeration control (Ricker, 2010), long-haul truck energy management (Johannesson, Murgovski, Jonasson, Hellgren, & Egardt, 2015), treatment of fibromyalgia (Deshpande, Nandola, Rivera, & Younger, 2014), mineral processing (Karelovic, Putz, & Cipriano, 2015), marine boiler control (Solberg, Andersen, Maciejowski, & Stoustrup, 2010) and aerial robot control with contact (Alexis, Huerzeler, & Siegwart, 2014). Such a wide variety of examples demonstrate the ubiquity of hybrid systems and the need for flexible control strategies.

It is popular to formulate an NMPC problem by the direct transcription of a continuous time optimal control problem, *i.e.* using a discretize then optimize approach (Biegler, 2007; Kameswaran & Biegler, 2006; Rao, 2009). The receding horizon principle of NMPC generates a feedback control strategy in which a parameterized, NLP problem is solved at each timestep. Care must be taken by the control engineer in order to ensure this problem is feasible but, in general, the online optimization

☆ This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and Maplesoft, Canada™.
* Corresponding author.
E-mail address: anson.maitland@uwaterloo.ca (A. Maitland).

https://doi.org/10.1016/j.conengprac.2020.104352
Received 1 October 2019; Received in revised form 19 February 2020; Accepted 23 February 2020
Available online xxxx
0967-0661/

problem of an NMPC often takes the following form

$$\begin{aligned} \min_{X,U} \; & \left\{ \mathcal{J}(X,U) = \ell_H(X) + \sum_{k=0}^{H-1} \ell_k(X,U) \right\} \\ \text{subject to} \quad & X \in \mathcal{X}(U; x(0)) \\ & U \in \mathcal{U} \end{aligned} \tag{1}$$

where $X = [x(1), \dots, x(H)]$ and $U = [u(0), \dots, u(H-1)]$ are the states and controls over the horizon, respectively, H is the length of the horizon, $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^p$ are the plant state and control inputs, respectively, ℓ_k are the stage costs, $\mathcal{X} \subset \mathbb{R}^{nH}$ captures the state constraints including the model dynamics given by some discrete time model $x(k+1) = f(x(k), u(k); \Delta t)$ with initial state $x(0)$, timestep Δt and $\mathcal{U} \subset \mathbb{R}^{pH}$ captures the control constraints.

Difficulties arise in solving Problem (1) when discrete components are present in the problem. Two such situations that we address in this paper are: 1. when some components of the control u belong to a countable set (*i.e.*are discrete-valued) or, 2. when the model f has state-dependent switches. We note these situations are not mutually exclusive. For situation 1, for notational purposes, we split the controls into two components. The discrete controls are denoted by v and the continuous controls by u. We may also refer to the discrete inputs v as integer-controls since any countable set is in bijection with some subset of the integers.

The QT strategy presented in this paper was introduced by the authors in Maitland and McPhee (2018a) for quickly solving Problem (1) in the first situation. In this paper we extend this strategy to the second case of fast hybrid system control using NMPC. Both situations in fact fall under the umbrella of hybrid systems since discrete behaviour is present in the models of both. In the optimal control literature, hybrid systems have been classified as either externally forced systems (EFS), those with discrete inputs (situation 1), or internally forced systems (IFS), those with state dependent switching (situation 2) (Sager, Bock, & Reinelt, 2009; Zhu & Antsaklis, 2015). An EFS is said to contain explicit (or controllable) switches while an IFS has implicit (or state-dependent) switches. In the optimal control literature, IFS systems are considered more difficult to handle but recent methods have been proposed to mitigate some of the issues (Bock, Kirches, Meyer, & Potschka, 2018). In the MPC setting the problem is simplified and we propose a fast strategy to handle the implicit switching behaviour of these systems.

For externally forced systems it is straightforward to cast Problem (1) as a Mixed-Integer Nonlinear Programming (MINLP) problem. However, the discrete controls introduce complexity that must be handled by a MINLP solver; it is known that MINLPs are NP-Hard and can even be undecidable (Garey & Johnson, 2002; Jeroslow, 1973; Murty & Kabadi, 1987). There are a multitude of MINLP solver methods, see Grossmann (2002) and Schlüter, Gerdts, and Rückmann (2012), but these can suffer from an exponential explosion in computation time making them unsuitable for real-time MPC applications. The QT strategy provides a method with a hard upper bound on the solution complexity in terms of design parameters. Essentially, this strategy uses realistic assumptions about the discrete control inputs to dramatically reduce the computational burden of solving a MINLP online.

The paper is organized as follows: in Section 2 we present some background on the QT strategy and provide some conditions under which we may expect it to perform well. In Section 3 we share a simple strategy to handle models with internal switches for use with a symbolic MPC formulation. In Section 4 we investigate the QT strategy for a variety of sample problems compared to alternative methods and report on the results. We draw conclusions and provide further avenues of research in Section 5.

2. Quasi-translations for externally forced hybrid NMPC

QTs were developed with the simultaneous goals of reducing the computational burden of solving Externally Forced Hybrid NMPC problems online and reducing undesirable chatter in their solutions. The tradeoff in their use is a guaranteed loss in optimality for many problems. However, in practise optimal MINLP solutions are often unobtainable when the turnaround time of the solver must meet real-time constraints. The use of QTs thus allow us to extend the domain of real-time Hybrid NMPC by providing a hard upper bound on the computation time of the online MINLP at the cost of a potentially sub-optimal solution. From our simulations it appears this loss in optimality often results in a qualitatively minimal loss of performance. The details of QT development and two example strategies were presented by the author in Maitland and McPhee (2018a).

2.1. Review of quasi-translations

We briefly review the basics of QTs here. For purposes of exposition we consider purely binary controls in the following. This is not as restrictive as it may first seem since all integer controls can be rewritten using a combination of binary ones (see page 24 of Kirches (2010)).

Let us denote a binary sequence by $B = [b(0), \dots, b(H-1)] \in \{0,1\}^H$. A *switch* occurs when consecutive elements differ and a consecutive subsequence of equal elements of maximal length is called a *block*. An internal block is a block that does not share an element at either the beginning or the end of the sequence. We draw QTs from the parameterized set

$$\mathcal{B}(s,l) = \left\{ B \in \{0,1\}^H \;\middle|\; \begin{array}{l} \text{number of switches in } B \le s \text{ and} \\ \text{length of all internal blocks in } B \ge l \end{array} \right\}$$

to control the chatter of our controller by setting an upper bound on the number of switches over the horizon and a lower bound on the timegap between them.

An Externally Forced Hybrid NMPC can often be written as a parameterized MINLP of the following form

$$\min_{(U,V) \in \mathcal{U} \times \mathcal{V}} \mathcal{J}(U, V; x(0)) \tag{2}$$

where $\mathcal{J}$ is the cost, $U = [u(0), \dots, u(H-1)]$ are the continuous controls over the horizon with $u \in \mathbb{R}^{p_u}$, $V = [v(0), \dots, v(H-1)]$ are the integer controls over the horizon and, without loss of generality, $v \in \mathbb{Z}^{p_v}$ such that $p_v + p_u = p$ and $x(0)$ is the initial state. We note that the states have been eliminated in this formulation and so $\mathcal{U}$ now encompasses the state and control constraints of (1). In this presentation we have eliminated the states by absorbing the plant dynamics directly into the cost, although it should be noted this is not necessary for using QTs. In Section 3.1 we discuss how this can be done in theory and practise.

To use QTs in Problem (2) we replace $\mathcal{V}$ with $\mathcal{V} \cap \mathcal{QT}(V^-)$ where $\mathcal{QT}(\cdot)$ denotes the set of QTs, given the selected strategy, and V^- is the integer solution found at the previous timestep. QTs act as constraints on the integer solution of the problem which can change over time. They enforce a degree of continuity between integer solutions over sequential timesteps. In Maitland and McPhee (2018a) we introduced two strategies to determine $\mathcal{QT}(\cdot)$. These strategies parameterize the mapping $\mathcal{QT}(\cdot)$ so we can control the quality and cost of our solutions. Briefly, the first strategy, called Crab-Walk QTs (CWQTs), constrains the QTs to be stiff translations of the input sequence. For a single step the QTs are $C(V; s, l, 1) = \{V, [v(0), V[0, H-2]], [V[1, H-1], 0], [V[1, H-1], 1]\} \cap \mathcal{B}(s,l)$ where $V[i,j] = v(i), v(i+1), \dots, v(j)$. The second strategy, called Inchworm QTs (IQTs), generates QTs that are generated from bounded translations of the switches of the input sequence. For a single swap the QTs are $\mathcal{I}(V; s, l, 1) = \{V, \cup_{z \in \mathcal{G}} V[\bar{z}]\} \cap \mathcal{B}(s,l)$ where $V[\bar{i}] = [v(0), \dots, v(i-1), 1 - v(i), v(i+1), \dots, v(H-1)]$ and $\mathcal{G} = \{i | v(i) \neq v(i+1)$ and/or $v(i) \neq v(i-1)\} \cup H - 1$. For example, if $V = [1,0,0,1,1]$ then

$$\begin{aligned} C(V;2,1,1) = \{&[0,0,1,1,0], \\ &[0,0,1,1,1], \\ &[\mathbf{1},\mathbf{0},\mathbf{0},\mathbf{1},\mathbf{1}], \\ &[1,1,0,0,1]\}, \end{aligned} \qquad \begin{aligned} \mathcal{I}(V;2,1,1) = \{&[0,0,0,1,1], \\ &[1,0,0,0,1], \\ &[1,0,1,1,1], \\ &[\mathbf{1},\mathbf{0},\mathbf{0},\mathbf{1},\mathbf{1}], \\ &[1,1,0,1,1]\}. \end{aligned}$$

We define the sets

$$C(V;s,l,r) = \bigcup_{V' \in C(V;s,l,r-1)} C(V';s,l,1)$$
$$\mathcal{I}(V;s,l,r) = \bigcup_{V' \in \mathcal{I}(V;s,l,r-1)} \mathcal{I}(V';s,l,1)$$

where r is an upper bound on the number of steps or swaps . Both strategies enforce the rule that switches cannot be introduced arbitrarily at the beginning of a sequence but must come from a step or swap of the input sequence. See Maitland and McPhee (2018a) for a more detailed definition of these strategies, as well as, a QT search strategy that scales linearly with parameter r.

2.2. When are QTs appropriate?

Since QTs are enforced by constraints the necessary conditions for an optimum (see Theorem 3 of Ioffe and Tihomirov (2009)) will be satisfied as long as the original QT-free problem satisfies those conditions as well. At the computational level the inclusion of QT constraints does not pose a problem. For example, consider the Generalized Benders Decomposition (GBD) as an example MINLP method. GBD is known to have finite convergence, see Theorem 6.3.4 of Floudas (1995). It is straightforward to see that as long as the original QT-free problem satisfies the conditions of Theorem 6.3.4 then the problem with QT constraints will satisfy those same conditions as well. One is free to use any MINLP method when including QT constraints, however given their recursive structure we can use greedy search methods to bound the computation time (Maitland & McPhee, 2018a).

In the following we examine under what conditions we may expect a QT strategy to perform well. Consider the following basic proposition on parameterized mixed-integer optimization problems. This proposition gives us an upper bound on the difference between solutions of perturbed mixed-integer optimization problems.

Proposition 1. *Consider the optimization problem*

$$\underset{(W,V)\in\mathcal{W}\times\mathcal{V}}{\arg\min}\ C(W,V;z) \tag{3}$$

where $\mathcal{W} \subset \mathbb{R}^w$ is convex and compact, $\mathcal{V} \subset \mathbb{Z}^m$, $\langle\mathcal{V}\rangle$ (the convex hull of $\mathcal{V}$) is compact and $z \in \mathcal{Z} \subset \mathbb{R}^n$ where $\mathcal{Z}$ is open. For brevity we denote $Y = (W,V)$. Now suppose the solution of (3) is unique and denoted $Y^*(z)$ and that the solution of the relaxation of (3) is in the interior of $\mathcal{W}\times\langle\mathcal{V}\rangle$. Further, suppose $C : \mathcal{W}\times\langle\mathcal{V}\rangle\times\mathcal{Z} \to \mathbb{R}$ is strongly convex in the first two arguments and twice continuously differentiable in all arguments so that for all $z \in \mathcal{Z}$ there exist constants $\lambda, \mu > 0$ such that

$$|C(Y;z) - C(Y';z)| \le \lambda\|Y - Y'\|_2 \tag{4}$$
$$\nabla^2_Y C(Y;z) \succeq \mu I \tag{5}$$

for all $Y, Y' \in \mathcal{W}\times\langle\mathcal{V}\rangle$. Then for all $z \in \mathcal{Z}$ there exists a neighbourhood $\mathcal{N}$ of z so that for all $z + \delta z \in \mathcal{N}$ and some constant $\beta > 0$

$$\|Y^*(z) - Y^*(z+\delta z)\|_2 \le 2\frac{\lambda}{\mu} + \beta\|\delta z\|_2. \tag{6}$$

Proof. First, we consider the relaxation of (3):

$$\underset{(\bar{W},\bar{V})\in\mathcal{W}\times\langle\mathcal{V}\rangle}{\arg\min}\ C(\bar{W},\bar{V};z) \tag{7}$$

which has a unique solution $\bar{Y}^*(z)$ by convexity. Further from the properties of C, we know $\nabla_Y C(\bar{Y}^*(z);z) = 0$ and $\nabla^2_Y C(\bar{Y}^*(z);z)$ is invertible. Thus by the implicit function theorem for some neighbourhood $\mathcal{N}$ containing z, $\bar{Y}^*(z)$ is a unique, continuous differentiable function. Thus for all $z + \delta z \in \mathcal{N}$

$$\|\bar{Y}^*(z) - \bar{Y}^*(z+\delta z)\|_2 \le \beta\|\delta z\|_2 \tag{8}$$

for some $\beta > 0$.

By the strong convexity of C for all $z \in \mathcal{Z}$

$$\|Y^*(z) - \bar{Y}^*(z)\|_2^2 \le \frac{C(Y^*(z);z) - C(\bar{Y}^*(z);z)}{\mu},$$

and thus

$$\|Y^*(z) - \bar{Y}^*(z)\|_2 \le \frac{\lambda}{\mu} \tag{9}$$

for all $z \in \mathcal{Z}$. Combining (8) and (9) via the triangle inequality we acquire (6), which was to be proved. □

We can use Proposition 1 to provide some insight into when we may expect QTs to work well in the nominal case. Firstly, in the MPC setting the optimization problem is parameterized by the plant's state whose evolution is captured by a discrete-time model of the form $x^+ = f(x,u,v;\Delta t)$. These models often have bounded derivatives over the state–space of interest so that they satisfy $\|x^+ - x\|_2 \le \gamma\Delta t$ for some $\gamma > 0$. Thus if one selects the timestep Δt small enough we can ensure that $x^+ \in \mathcal{N}(x)$ for given state x where $\mathcal{N}$ is determined by the local properties of $\mathcal{J}(\cdot;x)$.

If we interpret $\mathcal{W}$ as $\mathcal{U}$, as in (2), then the assumptions are quite restrictive for general NMPC problems. In particular, the assumption of convexity on the cost function with embedded nonlinear dynamics will likely be valid only within a small neighbourhood of a solution Y^*, if valid at all. However, if we interpret $\mathcal{W}$ as $\mathcal{X}\times\mathcal{U}$, as in (1), then the assumptions are applicable to many NMPC problems since they are designed with convex costs in mind. Putting these elements together, provided that $\mathcal{J}$ satisfies the conditions of Proposition 1, then we can conclude that

$$\|V^*(x) - V^*(x^+)\|_1 \le \sqrt{H(p+n)}\left(2\frac{\lambda}{\mu} + \beta\|x - x^+\|_2\right) \tag{10}$$

for some $\beta > 0$. Thus we have an upper bound on the difference between the integer solutions of consecutive MPC optimization problems. The smaller the bound (10) the better we expect QTs to perform. Thus problems with costs that have small Lipschitz constant λ, large μ and take small steps are expected to have better performance with QTs.

With the above understanding we can better interpret the QT strategies introduced. For the sake of clarity suppose v is a single binary input. Then given a control sequence V we have $\|V - V'\|_1 \le 3r$ for $V' \in C(V;s,l,r)$ and $\|V - V'\|_1 \le p$ for $V' \in \mathcal{I}(V;s,l,p)$. Both strategies provide sequences within $\mathcal{B}(s,l)$ that are within a parameterized bound of the input sequence. If p is selected to be greater than the upper bound in (10) then $\mathcal{I}(V;s,l,p)$ contains the optimal integer solution within $\mathcal{B}(s,l)$. Thus the IQT strategy if parameterized appropriately will always contain the optimal solution for problems that satisfy the assumptions of Proposition 1. The CWQT strategy is not complete as it cannot guarantee the inclusion of the optimal solution. However, it's success in practise suggests a hybrid of the two approaches should be considered. Further, the search methods proposed in Maitland and McPhee (2018a) operate by greedily searching through $\mathcal{QT}(\cdot)$ with monotonically increasing distance (in $\|\cdot\|_1$) from the input sequence.

3. Internally forced hybrid systems NMPC

Internally forced hybrid systems (or implicitly switched systems) are notoriously difficult to handle in an optimal control setting. In the following we describe a method to handle such systems by introducing artificial controls.

A discrete-time hybrid model with implicit switches can be described by the model subsystems (or modes) f_m and switching functions σ^i_m, $m \in \mathcal{M}, i \in \{1,\dots,m_n\}$. In the following we assume there are no state jumps (or discontinuities) at the mode transitions. In more detail we can describe such a system by

$$x(k+1) = f_m(x(k),u(k);\Delta t) \tag{11}$$

where $m \in \mathcal{M}$ is the active mode. A mode m is active when

$$(x,u) \in \mathcal{A}_m = \{(x,u)|\sigma^i_m(x,u) \le 0, i = 1,\dots,m_n\}. \tag{12}$$

We assume the system model is well-defined so that all $\mathcal{A}_m, m \in \mathcal{M}$ are disjoint and that the model domain is contained in $\bigcup_{m\in\mathcal{M}} \mathcal{A}_m$. We shall see that this type of problem when combined with a Symbolic Single-Shooting formulation gives rise to a discontinuous objective function.

3.1. Symbolic single-shooting formulation of NMPC

Symbolic Single-Shooting (symSS) is a formulation of the online NLP (1) of an NMPC that yields an unconstrained optimization problem where only the controls are optimization variables. This approach takes advantage of symbolic computing tools to embed the model simulation directly into the problem's objective function and eliminate the dynamic constraints. The symSS formulation has been used by the authors in Batra, Maitland, McPhee, and Azad (2018), Maitland, Batra, and McPhee (2018) and Maitland and McPhee (2018b).

Since the novelty of this formulation is in terms of handling the dynamics, for ease of presentation, we suppose the state constraints of (1) are given by

$$\mathcal{X}(U; x(0)) = \{X \,|x(k+1) = f(x(k), u(k); \Delta t), k = 0, \dots, H-1\}$$

and there are no control constraints (if control constraints are present they can be handled by penalty functions without increasing the order of the FHOCP). Then using symSS we rewrite (1) as

$$\min_U \left\{ \mathcal{J}(U; x(0)) = \ell_H(\tilde{X}) + \sum_{k=0}^{H-1} \ell_k(\tilde{X}, U) \right\} \tag{13}$$

where $\tilde{X} = [\tilde{x}(0), \dots, \tilde{x}(H)]$ and $\tilde{x}(k) = f \circ^k (x(0), U)$ which is defined by the rules

$$f \circ^0 (x(0), U) = x(0)$$
$$f \circ^{n+1} (x(0), U) = f(f \circ^n (x(0), U), u(n); \Delta t).$$

These expressions provide an explicit method to compute the states over the prediction horizon and thus embed the model simulation directly into the objective function.

By taking advantage of symbolic computing combined with automatic differentiation we can compute the exact gradient and Hessian of (13) offline to produce fast solvers and eliminate the need for computationally expensive and possibly unstable numerical differentiation and integration steps. The key step in this process is to realize (13) as a computational sequence whose execution computes (13) by introducing intermediate variables.

3.2. Symbolic single-shooting for internally forced hybrid NMPC

We can handle the difficulties presented by hybrid systems in the symSS formulation by introducing artificial controls w corresponding to the hybrid system's modes. These controls are used at the computational level to enforce the evaluation of a particular mode sequence. We denote these controls over the prediction horizon by $W = [w(0), \dots, w(H-1)]$. It should be noted that the artificial controls are not genuine degrees of freedom since we must have consistency between (x, u) and w, *i.e.*w activates mode m if and only if $(x, u) \in \mathcal{A}_m$. Thus given an initial state $x(0)$ and U we can determine W by a forward simulation of the hybrid model, *e.g.*$w(0)$ is determined by $(x(0), u(0))$, $w(1)$ is determined by $(x(1), u(1))$, where, with some abuse of notation, $x(1) = f_{w(0)}(x(0), u(0); \Delta t)$ (we have used w in place of the mode m it activates), and so forth. For an MPC problem with fixed timestep and finite horizon, finding a consistent W is easier than searching for the optimal switching points in the general setting of hybrid optimal control.

Combining symSS with internally forced hybrid systems leads to a non-smooth optimization problem. To handle models of the form (11)–(12) problem (13) is updated by redefining $\tilde{x}(k) = f_W \circ^k (x(0), U)$ where

$$f_W \circ^0 (x(0), U) = x(0)$$
$$f_W \circ^{n+1} (x(0), U) = f_{w(n)}(f_W \circ^n (x(0), U), u(n); \Delta t).$$

We note $w(\cdot)$ is a function of $(x(0), U)$ because of the above relations and the consistency constraint. Thus the updated (13) is an optimization problem wholly in terms of U parameterized by $x(0)$. Potential discontinuities in $\mathcal{J}$ and its derivatives have been introduced via the consistency conditions corresponding to the disjoint modes. The proper analysis of such problems belongs to the field of non-smooth analysis (Clarke, 1990; Clarke, Ledyaev, Stern, & Wolenski, 2008) and will not be discussed here.

At the computational level we use these artificial controls to embed the hybrid model into the larger class of continuous systems allowing us to generate exact derivatives despite the discrete behaviour of the system. The resulting computation then contains derivatives of all branches and we select our modes by choice of w. Computational issues may arise when computing the derivatives for inactive modes as they may not be defined, *e.g.*$1/\sqrt{y}$ when $y < 0$, but with the inclusion of appropriate safeguards in our generated code these can be safely mitigated.

If we consider the hybrid model as a computational sequence then the switching functions can be seen as conditions for the different branches of the intermediate variable evaluation. For example, the function

$$f(x, u) = \begin{cases} x^2 - u^2 & |x| \geq |u| \\ \frac{1}{u^2 - x^2} & |x| < |u| \end{cases}$$

can be written as the sequence

$$[t_1 \leftarrow u^2, t_2 \leftarrow x^2, t_3 \leftarrow t_2 - t_1, \textbf{ if } t_3 \geq 0 \textbf{ then } f \leftarrow t_3 \textbf{ else } f \leftarrow -1/t_3 \textbf{ endif}] \tag{14}$$

Using an artificial variable $w \in \{0, 1\}$ we can rewrite (14) as

$$[t_1 \leftarrow u^2, t_2 \leftarrow x^2, t_3 \leftarrow t_2 - t_1, f \leftarrow w t_3 + (w-1)/t_3] \tag{15}$$

which is equivalent to (14) when the following consistency condition is satisfied $w = 1 \Leftrightarrow |x| \geq |u|$.

For intermediate variables with multiple branches we need only introduce a single artificial variable $w \in \{w_1, \dots, w_m\}$ whose range is equal in cardinality to the number of branches m, *e.g.*the branching sequence

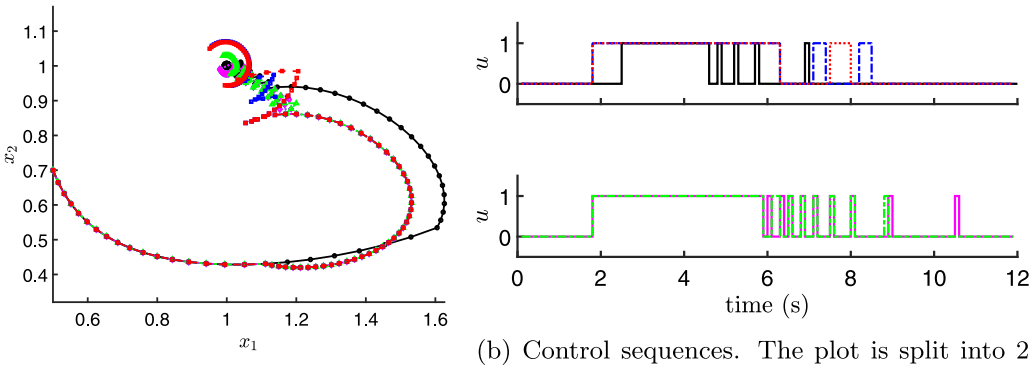
$$\begin{aligned} &\textbf{if}[\sigma_1^1(x, u) \leq 0] \wedge \cdots \wedge [\sigma_1^{1_n}(x, u) \leq 0] \textbf{ then } t_j \leftarrow \beta_1(x, u) \\ &\textbf{elseif}[\sigma_2^1(x, u) \leq 0] \wedge \cdots \wedge [\sigma_2^{2_n}(x, u) \leq 0] \textbf{ then } t_j \leftarrow \beta_2(x, u) \\ &\quad\vdots \\ &\textbf{elseif}[\sigma_m^1(x, u) \leq 0] \wedge \cdots \wedge [\sigma_m^{m_n}(x, u) \leq 0] \leq 0 \textbf{ then } t_j \leftarrow \beta_m(x, u) \\ &\textbf{endif} \end{aligned} \tag{16}$$

can be rewritten as

$$t_j \leftarrow \sum_{i=1}^{m} L_i(w)\beta_i(x, u) \tag{17}$$

where $L_i(w) = \prod_{k=1, k\neq i}^{m} \frac{w - w_k}{w_i - w_k}$ is the ith Lagrange basis polynomial so that $L_i(w_k) = \delta_{ik}$. Statements (16) and (17) are equivalent when $w = w_i \Leftrightarrow [\sigma_i^1(x, u) \leq 0] \wedge \cdots \wedge [\sigma_i^{i_n}(x, u) \leq 0]$.

The above method allows us to compute exact derivatives despite the hybrid nature of the model. Knowing $(x(0), U)$ we can compute the consistent sequence of artificial controls W by simply simulating the original hybrid model. Thus during the iterations within a solver we can maintain consistency as U is updated. For example, NLP iterations are often of the form $U^+ \leftarrow U + \alpha \Delta U$ where ΔU is based on local derivative information and α is selected via a line search or trust-region method. We can always find a $W(\alpha)$, via forward simulation, to be consistent with $(x(0), U^+)$ and thus maintain consistency at every iteration. This consistency condition only enforces the correct evaluation of the cost when there is an underlying internally switched hybrid model and does not change the convergence properties of the method. Ideally the optimization method should be chosen to handle the non-smooth nature of the optimization problem.



(a) Trajectories.

(b) Control sequences. The plot is split into 2 to ease readability.

Fig. 1. (black solid [circles]: DP, red dashed [squares]: IQT, blue dashed [squares]: CWQT, dot-dashed green [triangles]: C&R, magenta dotted [diamonds]: BONMIN-OA, cyan dotted [diamonds]: BONMIN-BB). Note that the BONMIN solutions are overlapping and hence indistinguishable in this figure. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

4. Case studies

We have selected a variety of problems to both demonstrate and examine the QT strategy for hybrid NMPC problems. The first is the Lotka–Volterra fishing problem, a classical optimal control problem with a binary input, second is a diesel airpath problem that has a binary control with internal switches and last is a longitudinal vehicle control problem with exclusive pedal action.

To evaluate the performance of the QT strategy we compare it to solutions generated using the Basic Open-source Nonlinear Mixed Integer programming (BONMIN) solver (Bonami et al., 2008) where possible, chosen as a representative MINLP solver, and the convexification and relaxation (C&R) method of Kirches (2010), Kirches, Sager, Bock and Schlöder (2010). For all solvers we pre-computed and optimized the appropriate exact gradients and Hessians using the symSS formulation described in Section 3. Thus for BONMIN and the C&R solvers this led to larger dimensional NLPs since the discrete controls were relaxed and included as optimization variables. Constraints were handled using penalty functions for the QT strategy and the C&R method and were passed directly to the BONMIN solver. The BONMIN solver was run using default settings with either the Branch and Bound (BB) or Outer-Approximation (OA) algorithm. The details of these algorithms can be found in Bonami et al. (2008).

All simulations were run using MATLAB 2019a on a desktop with an Intel(R) Core(TM) i7-4790 CPU. BONMIN was run on MATLAB using the OPTI Toolbox v2.27 (Currie, Wilson, et al., 2012). Unless otherwise stated the NLPs were solved using a generic Newton-based solver for unconstrained problems written by the authors.

4.1. Lotka–Volterra fishing problem

The Lotka–Volterra Fishing Problem is a classic benchmark problem in controls that can be found in Sager, Bock, Diehl, Reinelt, and Schloder (2006) and described on `mintoc.de`. The goal is to find an optimal fishing strategy to bring both predator and prey fish to a specific steady state. In the following formulation we regard the control to be binary, *i.e.* either no fishing or maximal fishing. The finite horizon NLP of the MPC is given by

$$\min_{V} J(V; x(0)) = \sum_{k=1}^{H} \|x(k) - x^r\|^2$$

$$\text{subject to} \quad x(k+1) = x(k) + \Delta t F(x(k), v(k)), k = 0, \dots, H-1$$
$$V \in \{0,1\}^H$$

where $H = 10, \Delta t = 0.01$ s, and the target state is $x^r = \begin{pmatrix} 1 & 1 \end{pmatrix}^T$. The simulations are run for 12 s and started at $x = \begin{pmatrix} 0.5 & 0.7 \end{pmatrix}^T$. The model arises from an explicit Euler discretization of the continuous time ODE model, $\dot{x} = F(x, v) = \begin{pmatrix} x_1(1 - x_2 - \frac{2}{5}v) \\ x_2(-1 + x_1 - \frac{1}{5}v) \end{pmatrix}$.

A dynamic programming (DP) solution was found using (Sundstrom & Guzzella, 2009) where the defined state grid contained 1000 equidistant points on the interval $[0, 2]$. The C&R solution used the IPOPT solver (interfaced with MATLAB using the OPTI Toolbox v2.27) with `maxiter` = 10 and sum up rounding (SUR) strategy with a threshold of 0.5 (Kirches, 2010). Since there are no continuous optimization variables for this problem, no derivative information was needed for the QT methods only — the cost function itself.

In Fig. 1a we compare the trajectories of all solution methods.

The parameter settings of the QT solutions are $s = 3, l = 1$, IQT: $r = 3$ and CWQT: $r = 5$. The two BONMIN solutions are equal and overlap entirely. We can see all solutions approach the point x^r. In Fig. 1b we display the control sequences of the different solutions. We can see the DP solution, with its global knowledge, is able to switch the least and achieve the best performance. All NMPC solutions first switch at the same timestep, before 2 s, whereas the DP solution switches later. Surprisingly, the BONMIN and C&R solutions arrive at very similar switching behaviour as can be seen in the second plot.

In terms of performance, the DP solution performed best (with an overall cost of 1.55 where the overall cost is defined by $\Delta t \sum \|x(k) - x^r\|^2$ where the sum is over the entire simulation). Next best was the C&R and BONMIN solutions which were 34% worse (at 2.08) followed by the CWQT, IQT solutions which were 35% (2.10) and 36% (2.11) worse, respectively. Relative to the DP optimum the QT solutions were at most 2% worst than the BONMIN solution.

The BONMIN and C&R solutions switch 20 times over the 12 s simulation and the DP only 10. The CWQT and IQT solutions switch 10 and 6 times, respectively. Thus the QT solutions reduce control switching by a factor of at least 2 (compared to BONMIN) with a small performance degradation.

In Fig. 2 we display a sample of the turnaround times (TATs) of the solvers over a simulation. The QT solutions have at least an order of magnitude smaller average TATs. The BONMIN solutions are the slowest. This large gap in computing time can justify the use of the QT strategy despite the small performance degradation. For reference the global DP solution took 99s to compute.

To better understand the role of the QT parameters we now examine the role of the parameters on the performance of the CWQT solutions. Given $H = 10$, we consider $s, r \in \{1, \dots, H-1\}, l \in \{1, \dots, H-2\}$. Over this parameter range the best performance of CWQT using full enumeration is 2.0755 and the best performance of the greedy search is only 1.07% worse. Thus the use of the greedy optimization scheme does not significantly degrade the performance over a naive full enumeration approach. In the following, we utilize the greedy QT scheme by default.

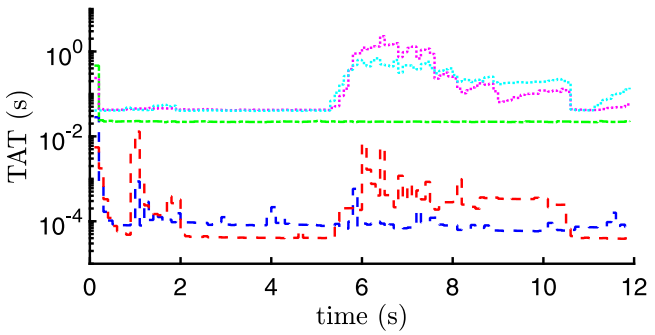


Fig. 2. Sample TATs of the solvers (red dashed: IQT, blue dashed: CWQT, dot-dashed green: C&R, magenta dotted: BONMIN-OA, cyan dotted: BONMIN-BB). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

In Fig. 3 we display the role the parameters play in controller performance while using the CWQT strategy.

In Fig. 3a we display the cost performance of CWQT relative to the best performance over the set of parameters. As can be seen, performance degrades steadily with larger l, is largely unaffected by r and is worst for $s = 1$. In Fig. 3b we display the maximum number of NLP subproblems solved (an architecture-free measure of TAT) during a timestep over the set of parameters. Only for small values of l do s, r have a significant impact. In Fig. 3c we display the number of switches that occurred over the simulation. We see that for larger l chatter is kept to a near minimum, independent of s, r.

To minimize TAT, chatter and cost we want to pick a parameter set that approximately minimizes all 3 plots. From the trends present in these plots it appears that small values of all parameters with $s > 1$ are near optimal depending on the relative importance of TAT, chatter and cost.

4.2. Diesel airpath model

The main goal of this controller is to track a reference output trajectory where $y = \begin{pmatrix} p_{int} & \phi_{EGR} \end{pmatrix}^T$, the intake manifold pressure and exhaust gas recirculation (EGR) rate, are the outputs of interest. Tracking these two outputs allows the engine to meet the driver's demands while satisfying emission requirements. The diesel airpath (DAP) model under consideration is a 9 state, nonlinear, physics based-model with 3 continuous controls, 1 binary control, 2 disturbance inputs and 2 internal switches. The disturbance inputs of the model were held constant in our simulations. More information on the model and its derivation can be found in Choroszucha, Sun, and Butts (2016) and the references therein.

A further objective of the controller is to minimize the rate of change of the continuous controls $u = \begin{pmatrix} \theta & u_{EGR} & u_{VGT} \end{pmatrix}^T$, throttle command [% closed], EGR valve command [% open], and variable-geometry turbocharger (VGT) command [% normalized], while satisfying the control bounds $\theta \in [0, 100], u_{EGR} \in [0, 70]$ and $u_{VGT} \in [40, 80]$. The problem cost is given by

$$\mathcal{J}(U, V; x(0)) = \sum_{k=0}^{H-1} \|y(k) - y^r(k)\|_{\mathbf{Q}}^2 + \|\Delta u(k)\|_{\mathbf{R}}^2 + \|\rho_p(u(k))\|^2$$

where the single binary control $v \in \{0, 1\}$ is the EGR cooler bypass setting , y^r is a reference output trajectory, $\Delta a(k) = a(k) - a(k-1)$ (with $\Delta a(0) = 0$), $\mathbf{Q} = \text{diag}\begin{pmatrix} 1 & 10^4 \end{pmatrix}$ and $\mathbf{R} = \text{diag}\begin{pmatrix} 10 & 10 & 10 \end{pmatrix}$. To enforce the box constraints we used a penalty function of the form ρ_p, for $p \in \mathbb{N}$, given by

$$\rho_p(z)^2 = \left(\frac{2z - (z^{\max} + z^{\min})}{z^{\max} - z^{\min}} \right)^{2p}$$

Table 1
Mean RMS tracking error of 6 simulations.

	p_{int}[kPa]	ϕ_{EGR}[−]
QT	0.2125	0.0012
C&R	10.9865	0.0719

which enforces constraints of the form $z \in [z^{\min}, z^{\max}]$ in the limit $p \to \infty$. We evaluate ρ_p entry-wise for vector-valued arguments. In our simulations we set $p = 4$.

In Fig. 4 we display the results of the different solvers for a sample reference trajectory. We note that only a single internal switch was activated over this simulation. Here we used a CWQT strategy as a representative for the QT strategy since both strategies performed very similarly. It is clear from the figure that the QT strategy performed best in tracking error compared to the C&R solution. The values of this difference are captured in Table 1 which summarizes the results of 6 different simulations (including the one shown in Fig. 4a). From Table 1 we see that the QT method is more than an order of magnitude better at tracking than the C&R strategy.

From Fig. 4a we can see that the C&R strategy is unable to take advantage of the binary input (this was tested for a variety of SUR thresholds). We note that the solutions are similar when the bypass mode is equal. It is a shortcoming of the C&R strategy that a rounding strategy is unable to take advantage of the binary input.

We found that the QT parameters did not affect their solutions significantly. This is largely due to the relatively slow switching frequency of the problem. We do not report TATs for this problem since we could not get a fair comparison of the two approaches. The TAT of the QT solution is on the order of 1 ms and the consistency enforcement function is on the order of 0.01 ms.

4.3. Longitudinal vehicle control

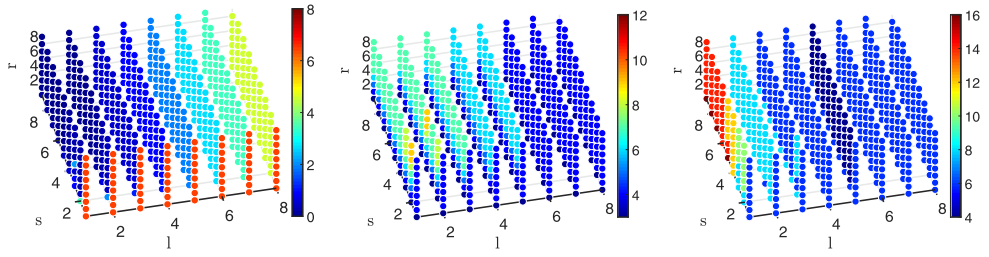
The objective of this controller is to track a reference speed using exclusive accelerator and brake pedal actions. The control inputs of the system are pedal depth $u \in [0, 1]$ and pedal choice $v \in \{0, 1\}$. From these values we can acquire the brake pedal position $BPP = 0.3815u(1 - v) + 0.1385$ and accelerator pedal position $APP = 100uv$ required of the high-fidelity plant model.

The longitudinal control oriented model of the vehicle is given by

$$\begin{aligned}
\dot{s}_x &= s_y\dot{\psi} + \frac{\sum_{\bullet\in\{l,r\}} F_{long,\bullet f}\cos\delta - F_{lat,\bullet f}\sin\delta + F_{long,\bullet r}}{m} - d_C\frac{s_x^2}{m} \\
\dot{s}_y &= -s_x\dot{\psi} + \frac{\sum_{\bullet\in\{l,r\}} F_{long,\bullet f}\sin\delta + F_{lat,\bullet f}\cos\delta + F_{lat,\bullet r}}{m} \\
\dot{\omega}_{\bullet f} &= \frac{\rho_2 T_{mot} - \frac{3}{8}T_B - r_{eff}\sum_{\bullet'\in\{l,r\}} F_{long,\bullet' f}}{\rho_2 I_{mot} + \rho_1 I_{gen} + 2J_w} \\
\dot{\omega}_{\bullet r} &= -\frac{\frac{1}{8}T_B + r_{eff}F_{long,\bullet r}}{J_w} \\
\dot{\lambda}_{\bullet f} &= \frac{r_{eff}\omega_{\bullet f} - (1+\lambda_{\bullet f})[(s_y + \dot{\psi}L_f)\sin\delta + (s_x \mp \dot{\psi}\frac{L_w}{2})\cos\delta]}{\beta} \\
\dot{\lambda}_{\bullet r} &= \frac{r_{eff}\omega_{\bullet r} - (1+\lambda_{\bullet r})[s_x \mp \dot{\psi}\frac{L_w}{2}]}{\beta}
\end{aligned} \tag{18}$$

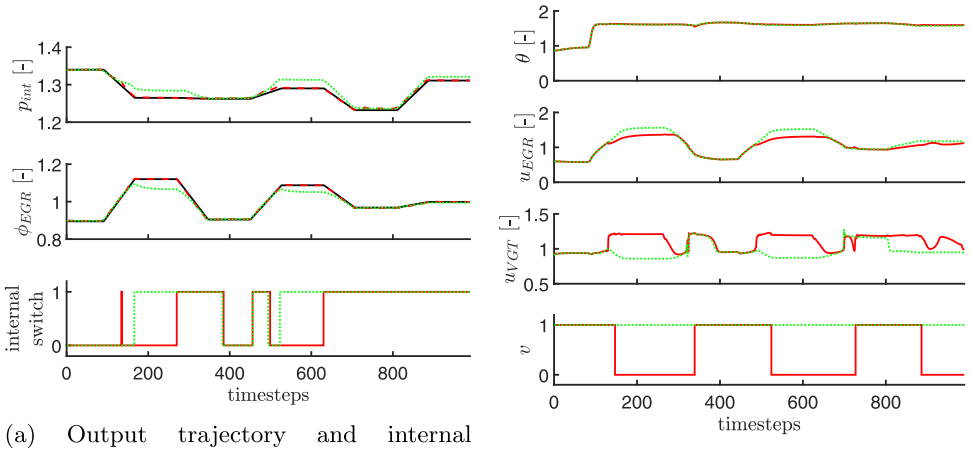
where the states $x = \begin{pmatrix} s_x & s_y & \omega_{lf} & \omega_{rf} & \omega_{lr} & \omega_{rr} & \lambda_{lf} & \lambda_{rf} & \lambda_{lr} & \lambda_{rr} \end{pmatrix}^T$ are longitudinal and lateral speed, and all four wheel spins and slip ratios, respectively. The brake torque, motor torque and longitudinal tire forces are given by

$$\begin{aligned}
T_B &= a_B[1 + e^{b_B BPP + c_B}]^{-\frac{1}{d_B}} \\
T_{mot} &= C_{mot} + \sum_{i\in\{1,2\}} a_{mot,i}[1 + e^{b_{mot,i}APP + c_{mot,i} + d_{mot,i}\sqrt{s_x^2 + s_y^2} + e_{mot,i}T_{mot}^-}]^{-1} \\
F_{long,\bullet\circ} &= -d_{p,\bullet\circ}\sin(c_{p,\bullet\circ}\arctan(b_{p,\bullet\circ}\lambda_{\bullet\circ}) + e_{p,\bullet\circ}(b_{p,\bullet\circ}\lambda_{\bullet\circ} - \arctan(b_{p,\bullet\circ}\lambda_{\bullet\circ}))) \\
&\quad \times F_{z,\bullet\circ}
\end{aligned}$$



(a) Relative CWQT performance (%). (b) CWQT max number NLP subproblems. (c) CWQT number of realized switches.

Fig. 3. Role of parameters in CWQT performance.



(a) Output trajectory and internal switches. (b) Controls.

Fig. 4. Solver performance (reference: black, C&R: green, QT: red). All values have been normalized and offset. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

where $\bullet \in \{l,r\}, \circ \in \{f,r\}$. The torques are determined by shallow neural network mappings (Hosking, 2018) and the longitudinal forces by the Pacejka magic formula (Pacejka & Bakker, 1992). The inputs of this model are: $\dot{\psi}$ yaw rate, δ steering angle, T^-_{mot} previous timesteps motor torque, $F_{z,\bullet\circ}$ wheel normal forces and $F_{lat,\bullet\circ}$ lateral forces. The parameters of the model are: m mass, r_{eff} tire radius, d_C drag coefficient, J_w wheel inertia, L_w vehicle width, L_f vehicle length from front to CG, ρ_1, ρ_2 gear ratios, β relaxation length, I_{mot} motor inertia, I_{gen} generator inertia, $b_{p,\bullet\circ}, c_{p,\bullet\circ}, d_{p,\bullet\circ}, e_{p,\bullet\circ}$ Pacejka tire parameters and $a_B, \dots, d_B, a_{mot,i}, \dots, e_{mot,i}, C_{mot}$ neural network parameters.

The cost function of the MPC is

$$\begin{aligned}\mathcal{J}(U,V;x(0)) = &\sum_{k=0}^{H-1} Q\left(\sqrt{s_x(k)^2+s_y(k)^2}-s_r(k)\right)^2 + R_1\{vu\}(k)^2 \\ &+R_2\{[1-v]u\}(k)^2 \\ &+1/\Delta t\left(R_1'\Delta\{vu\}(k)^2 + R_2'\Delta\{[1-v]u\}(k)^2\right) \end{aligned} \tag{19}$$

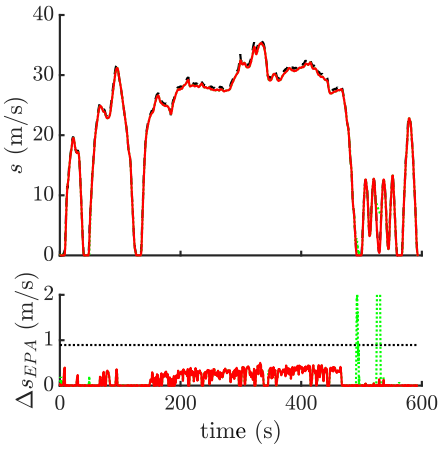
where s_r is a reference speed, Q, R_1, R_2, R_1', R_2' are weights, $\{ab\}(k) = a(k)b(k)$ and $\Delta a(k) = a(k) - a(k-1)$ (with $\Delta a(0) = 0$). For the MPC the control oriented model (18) was discretized using RK4 within the symSS formulation with a timestep $\Delta t = 0.1$s and horizon $H = 10$.

Our control simulations used a high-fidelity model of the Autonomoose, an autonomous vehicle project at the University of Waterloo, as the plant. The high-fidelity model consists of a 14 degree of freedom vehicle dynamics model with combined slip Pacejka tires (Va. Gennip, 2018) and a grey-box power-split hybrid powertrain model (Hosking, 2018) developed in MapleSim. The control oriented longitudinal model (18) is based on this high-fidelity model. The controller is run at 50 Hz. During simulation we set $\delta = 0$, and $\dot{\psi}, F_{z,\bullet\circ}, F_{lat,\bullet\circ}$ were all updated using the high-fidelity plant.

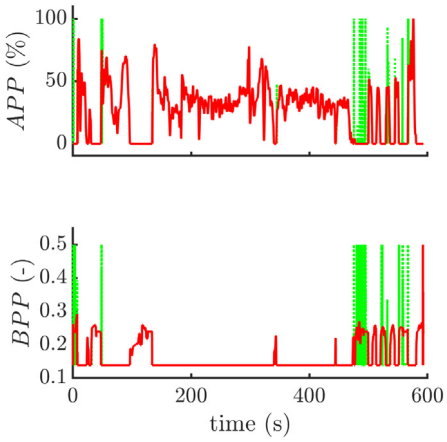
We tested the resulting MPC using the US06 Supplemental Federal Test Procedure (SFTP) drive cycle. This drive cycle is 596 s in length and represents aggressive driving behaviour. It is used for the determination of EPA on-road fuel economy ratings in the United States. During a test the allowable speed tracking error is to be below 2 mph (0.89 m/s) within a ± 1 s window of the current timestep, *i.e.* $\Delta s_{EPA}(k) = \min_{i\in\{-N,\dots,N\}}\{|\sqrt{s_x(k)^2+s_y(k)^2}-s_r(k+i)|\} < 2$ mph for all timesteps k where N is the frequency (in Hz) of the controller.

In Fig. 5a we can see the tracking performance of the QT controller as well as a C&R controller using a standard rounding procedure. The C&R strategy used the same horizon length and tuning as the QT controller. The tracking behaviour of both methods is similar for a majority of the drive cycle. The QT solution is able to maintain the tracking error below the EPA defined threshold for the entire run whereas the C&R controller fails to do so. It goes too far off track twice during the final set of acceleration/deacceleration maneuvers. This is understandable looking at the control solutions, see Fig. 5b. During these final maneuvers the C&R solution exhibits an undesirable high-frequency switching response. The QT solution by comparison exhibits a smooth control response and better tracking behaviour. Zooming in on an earlier acceleration phase of the drive cycle, see Fig. 6, we can see that this qualitative difference in control response is not isolated to the final maneuvers. During speed ups and slow downs the QT solution is smoother with fewer switches between pedal inputs. Over the three seconds shown in Fig. 6 the QT solution switches once from braking to throttle while the C&R solution switches a total of thirteen times with poorer tracking performance.

Both solutions ran faster than realtime in Simulink with the QT solution being $> 1.5\times$ faster. Although the C&R solution requires the solution of a single NLP its dimension is twice that of the sub-NLP



(a) Speed tracking results and associated EPA error (reference trajectory is given by the black dashed line in upper plot and the 2 mph limit is shown by the black dotted line in lower plot).



(b) Pedal inputs.

Fig. 5. Solver performance for US06 SFTP drive cycle (C&R: green, QT: red). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

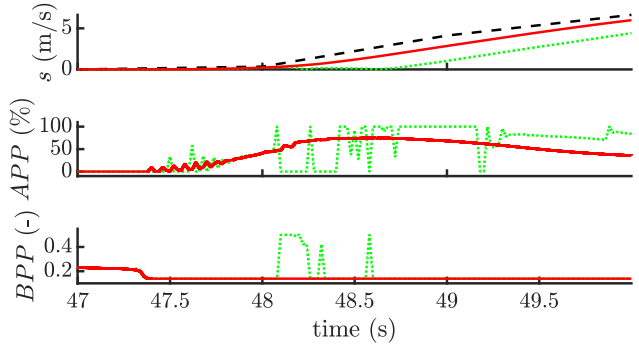


Fig. 6. Close-up of solver performance (speed tracking and controls) during the second acceleration phase (C&R: green, QT: red). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

problems of the QT method leading to a longer solution time. Overall, in this scenario the QT strategy performed better qualitatively and quantitatively with faster TATs.

5. Conclusions

We have presented a strategy that relies on simple assumptions to handle the difficulties arising from discrete control inputs. This strategy, coupled with a greedy search method, provides a hard upper bound on computational complexity that makes it suitable for real-world problems with hard real-time constraints. The QT strategy enforces some degree of continuity between timesteps of an MPC. For integer controls, utilizing the optimum of an NLP from one timestep to the next is not necessarily desirable as chatter can be introduced to the input signal. One can view the QT strategy as a modification of the receding horizon principle for integer controls. Instead of implementing only $u(0)$ at timestep t_i we implement the subsequence $u(0), u(1), \ldots, u(n)$ of the solution at t_i over the next $n+1$ timesteps, where n varies depending on the solution and QT parameters. This is done for each timestep meaning there is an enforced overlap in integer solutions over sequential timesteps.

By investigating a number of test problems we were able to demonstrate the practicality of the QT strategy. In terms of controller performance, the QT strategy performs favourably as compared to existing strategies for nonlinear problems, with a reduction in controller TATs. In more than one case the QT strategy was able to find viable solutions where other solution methods failed.

Some basic theoretical questions about the applicability of the QT strategy has been addressed. We have provided a class of problems for which the QT strategy is expected to work however, further theoretical questions raised by the QT strategy remain unexamined, such as questions about recursive feasibility or controller stability. The QT strategy shares features with the Moving Window Blocking strategy presented in Cagienard, Grieder, Kerrigan, and Morari (2007) which contains some stability and feasibility results for linear time invariant systems. However, it is expected that significant work is required to extend these results to a general hybrid nonlinear MPC setting with QTs. At this stage, the QT strategy is a heuristic strategy that provides a possible method to handle the complexities of hybrid nonlinear systems in MPC while satisfying hard real-time constraints.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Alexis, K., Huerzeler, C., & Siegwart, R. (2014). Hybrid predictive control of a coaxial aerial robot for physical interaction through contact. *Control Engineering Practice*, *32*, 96–112.

Andersson, J. (2013). *A general-purpose software framework for dynamic optimization* (Ph.D. thesis, Ph.D. thesis), Kasteelpark Arenberg 10, 3001-Heverlee, Belgium: Arenberg Doctoral School, KU Leuven, Department of Electrical Engineering (ESAT/SCD) and Optimization in Engineering Center.

Andrikopoulos, G., Nikolakopoulos, G., & Manesis, S. (2013). Pneumatic artificial muscles: A switching model predictive control approach. *Control Engineering Practice*, *21*(12), 1653–1664.

Batra, M., Maitland, A., McPhee, J., & Azad, N. L. (2018). Non-linear model predictive anti-jerk cruise control for electric vehicles with slip-based constraints. In *2018 annual American control conference (ACC)* (pp. 3915–3920). IEEE.

Biegler, L. T. (2007). An overview of simultaneous strategies for dynamic optimization. *Chemical Engineering and Processing: Process Intensification*, *46*(11), 1043–1053.

Bock, H. G., Kirches, C., Meyer, A., & Potschka, A. (2018). Numerical solution of optimal control problems with explicit and implicit switches. *Optimization Methods & Software*, *33*(3), 450–474.

Bonami, P., Biegler, L. T., Conn, A. R., Cornuéjols, G., Grossmann, I. E., Laird, C. D., et al. (2008). An algorithmic framework for convex mixed integer nonlinear programs. *Discrete Optimization*, *5*(2), 186–204.

Cagienard, R., Grieder, P., Kerrigan, E. C., & Morari, M. (2007). Move blocking strategies in receding horizon control. *Journal of Process Control*, *17*(6), 563–570.

Cairano, S. D., & Kolmanovsky, I. V. (2018). Real-time optimization and model predictive control for aerospace and automotive applications. In *2018 annual American control conference (ACC)* (pp. 2392–2409). IEEE.

Camacho, E. F., Ramírez, D. R., Limón, D., La Peña, D. M. D., & Alamo, T. (2010). Model predictive control techniques for hybrid systems. *Annual Reviews in Control*, *34*(1), 21–31.

Choroszucha, R., Sun, J., & Butts, K. (2016). Nonlinear model order reduction for predictive control of the diesel engine airpath. In *American control conference (ACC), 2016* (pp. 5081–5086). IEEE.

Clarke, F. H. (1990). *Optimization and nonsmooth analysis, Vol. 5*. Siam.

Clarke, F. H., Ledyaev, Y. S., Stern, R. J., & Wolenski, P. R. (2008). *Nonsmooth analysis and control theory, Vol. 178*. Springer Science & Business Media.

Currie, J., Wilson, D. I., et al. (2012). OPTI: lowering the barrier between open source optimizers and the industrial MATLAB user. *Foundations of Computer-Aided Process Operations*, *24*, 32.

Deshpande, S., Nandola, N. N., Rivera, D. E., & Younger, J. W. (2014). Optimized treatment of fibromyalgia using system identification and hybrid model predictive control. *Control Engineering Practice*, *33*, 161–173.

Diehl, M., Ferreau, H. J., & Haverbeke, N. (2009). Efficient numerical methods for nonlinear mpc and moving horizon estimation. In *Nonlinear model predictive control* (pp. 391–417). Springer.

Floudas, C. A. (1995). *Nonlinear and mixed-integer optimization: fundamentals and applications*. Oxford University Press.

Frasch, J. V., Sager, S., & Diehl, M. (2015). A parallel quadratic programming method for dynamic optimization problems. *Mathematical Programming Computation*, *7*(3), 289–329.

Garey, M. R., & Johnson, D. S. (2002). *Computers and intractability, Vol. 29*. wh freeman New York.

Grossmann, I. E. (2002). Review of nonlinear mixed-integer and disjunctive programming techniques. *Optimization and Engineering*, *3*(3), 227–252.

Hosking, B. A. (2018). *Modelling and model predictive control of power-split hybrid powertrains for self-driving vehicles* (Master's thesis), University of Waterloo.

Ioffe, A. D., & Tihomirov, V. M. (2009). *Theory of extremal problems, Vol. 6*. Elsevier.

Jeroslow, R. (1973). There cannot be any algorithm for integer programming with quadratic constraints. *Operations Research*, *21*(1), 221–224.

Johannesson, L., Murgovski, N., Jonasson, E., Hellgren, J., & Egardt, B. (2015). Predictive energy management of hybrid long-haul trucks. *Control Engineering Practice*, *41*, 83–97.

Kameswaran, S., & Biegler, L. T. (2006). Simultaneous dynamic optimization strategies: Recent advances and challenges. *Computers and Chemical Engineering*, *30*(10), 1560–1575.

Karelovic, P., Putz, E., & Cipriano, A. (2015). A framework for hybrid model predictive control in mineral processing. *Control Engineering Practice*, *40*, 1–12.

Kirches, C. (2010). *Fast numerical methods for mixed–integer nonlinear model–predictive control* (Ph.D. thesis), Ruprecht-Karls-Universität Heidelberg.

Kirches, C., Sager, S., Bock, H. G., & Schlöder, J. P. (2010). Time-optimal control of automobile test drives with gear shifts. *Optimal Control Applications & Methods*, *31*(2), 137–153.

Kirches, C., Wirsching, L., Sager, S., & Bock, H. G. (2010). Efficient numerics for nonlinear model predictive control. In *Recent advances in optimization and its applications in engineering* (pp. 339–357). Springer.

Maitland, A., Batra, M., & McPhee, J. (2018). Nonlinear model predictive control reduction using truncated single shooting. In *2018 annual American control conference (ACC)* (pp. 3165–3170). IEEE.

Maitland, A., & McPhee, J. (2018a). Fast NMPC with mixed-integer controls using quasi-translations. *IFAC-PapersOnLine*, *51*(20), 343–348.

Maitland, A., & McPhee, J. (2018b). Towards integrated planning and control of autonomous vehicles using nested MPCs. In *ASME 2018 dynamic systems and control conference*. American Society of Mechanical Engineers, V003T32A018.

Murty, K. G., & Kabadi, S. N. (1987). Some NP-complete problems in quadratic and nonlinear programming. *Mathematical Programming*, *39*(2), 117–129.

Pacejka, H. B., & Bakker, E. (1992). The magic formula tyre model. *Vehicle System Dynamics*, *21*(S1), 1–18.

Quirynen, R., Vukov, M., & Diehl, M. (2015). Multiple shooting in a microsecond. In *Multiple shooting and time domain decomposition methods* (pp. 183–201). Springer.

Rao, A. V. (2009). A survey of numerical methods for optimal control. *Advances in the Astronautical Sciences*, *135*(1), 497–528.

Ricker, N. L. (2010). Predictive hybrid control of the supermarket refrigeration benchmark process. *Control Engineering Practice*, *18*(6), 608–617.

Sager, S., Bock, H. G., Diehl, M., Reinelt, G., & Schloder, J. P. (2006). Numerical methods for optimal control with binary control functions applied to a Lotka-volterra type fishing problem. In *Recent advances in optimization* (pp. 269–289). Springer.

Sager, S., Bock, H. G., & Reinelt, G. (2009). Direct methods with maximal lower bound for mixed-integer optimal control problems. *Mathematical Programming*, *118*(1), 109–149.

Schlüter, M., Gerdts, M., & Rückmann, J.-J. (2012). A numerical study of MIDACO on 100 MINLP benchmarks. *Optimization*, *61*(7), 873–900.

Solberg, B., Andersen, P., Maciejowski, J. M., & Stoustrup, J. (2010). Optimal switching control of burner setting for a compact marine boiler design. *Control Engineering Practice*, *18*(6), 665–675.

Sundstrom, O., & Guzzella, L. (2009). A generic dynamic programming matlab function. In *Control applications,(CCA) and intelligent control,(ISIC), 2009 IEEE* (pp. 1625–1630). IEEE.

Va. Gennip, M. (2018). *Vehicle dynamic modelling and parameter identification for an autonomous vehicle* (Master's thesis), University of Waterloo.

Vukov, M., Domahidi, A., Ferreau, H. J., Morari, M., & Diehl, M. (2013). Auto-generated algorithms for nonlinear model predictive control on long and on short horizons. In *Decision and control (CDC), 2013 IEEE 52nd annual conference on* (pp. 5113–5118). IEEE.

Zhu, F., & Antsaklis, P. J. (2015). Optimal control of hybrid switched systems: A brief survey. *Discrete Event Dynamic Systems*, *25*(3), 345–364.