2024 IEEE 63rd Conference on Decision and Control (CDC)
December 16-19, 2024. MiCo, Milan, Italy

A Gap Penalty Method for Optimal Control of Linear Complementarity Systems

Kangyu Lin and Toshiyuki Ohtsuka

***Abstract*— In this study, we propose a novel penalty reformulation and numerical solution method for optimal control problems (OCPs) of linear complementarity systems. The proposed reformulation aims to construct a penalty term tailored to the complementarity constraints using the D-gap function. The proposed penalty term is nonconvex but exhibits a convexity structure that can be utilized by convexity-exploiting solution methods. To solve the reformulated OCP efficiently, we propose a solution method using the sequential convex quadratic programming framework. The convexity of subproblems is guaranteed by a pre-computed regularization matrix using the parameter of the D-gap function. The proposed method is globalized using a merit line search strategy. We confirmed the effectiveness of the proposed method using a benchmark test in comparison with several state-of-the-art methods.**

I. Introduction

A. Background

This study considers optimal control problems (OCPs) for a class of non-smooth dynamical systems governed by the *linear complementarity system* (LCS). Briefly, the LCS involves a linear ordinary differential equation (ODE) and a *linear complementarity problem* (LCP) [4]. The theoretical properties of the LCS, such as the existence, uniqueness, and non-Zeno behavior of the solution trajectory, have been intensively discussed in [1], [3], [10]. Benefiting from the powerful modeling capability of LCPs, LCSs have been widely applied in modeling non-smooth systems arising in engineering and economics [1], [3], [13].

OCPs of LCSs have recently garnered significant attention. The first-order necessary optimality conditions for the continuous-time OCP of the LCS were studied in [22] based on the results reported in [8]. Direct methods (i.e., first discretize then optimize) and indirect methods (i.e., first optimize then discretize) for the OCP of LCS have also been developed in [22]. From the perspective of practical applications, direct methods are more favored as the discretized OCP can often be solved efficiently by well-developed nonlinear programming (NLP) solvers. However, the discretized OCP of LCS belongs to a class of notorious NLP problems known as *mathematical programming with complementarity constraints* (MPCCs), which violates almost all constraint qualifications at every feasible point. The lack of constraint regularity often leads to failure when directly using off-the-shelf NLP solvers to solve MPCC. Nevertheless, NLP solvers are still widely applied in specific methods for solving MPCC. In these MPCC-tailored methods, the complementarity constraints are first reformulated as parameterized cost terms or constraints, and the MPCC solution can then be obtained using a continuation method that solves a sequence of parameterized NLP problems. These MPCC-tailored methods, such as penalty methods [14], relaxation methods [11], and smoothing methods [12], are practical for MPCC arising in OCP of non-smooth dynamical systems and are briefly reviewed in Subsection II-B.

Recently, researchers have focused on a subclass of MPCCs known as *linear complementarity quadratic programming* (LCQP), which is the discretized OCP of LCS. In [9], an efficient solution method for LCQP is proposed based on complementarity penalty reformulation and a sequential convex quadratic programming (SCQP) framework. This method ensures the subproblem convexity by ignoring the constant indefinite Hessian of the penalty function.

B. Motivation and contributions

The motivation is in the nonlinear model predictive control (MPC) of a general non-smooth dynamical system, such as *dynamical complementarity systems* and *differential variational inequalities*. In each MPC update, the dynamics can be linearly approximated by an LCS near the reference trajectory [13]. Thus, a method capable of efficiently solving the OCP of LCS during each sampling period is required to achieve feedback control.

The contributions of this study are mainly on two aspects. First, we propose a novel penalty reformulation for the OCP of LCS by using the *D-gap function* [20]. The proposed penalty reformulation exhibits two favorable properties:

- Its constraint system is differentiable, satisfies constraint qualifications, and is more concise than that of the classical MPCC-tailored methods;
- Its penalty term exhibits a convexity structure that can be utilized by convexity-exploiting solution methods.

Second, we present an efficient SCQP-type method based on the proposed penalty reformulation. The proposed SCQP-type method is globalized using a dedicated merit line search strategy. A benchmark test revealed that the proposed method outperformed state-of-the-art MPCC-tailored methods.

C. Outline

The remainder of this study is organized as follows: Section II provides the problem formulation and reviews MPCC-tailored methods; Section III presents a new penalty reformulation that uses the D-gap function; Section IV

This work was supported in part by JSPS KAKENHI (Grant Number JP22H01510). The author Kangyu Lin was supported by the CSC scholarship (No. 201906150138). The authors are with the Department of Systems Science, Graduate School of Informatics, Kyoto University, Kyoto, Japan. Email: k-lin@sys.i.kyoto-u.ac.jp and ohtsuka@i.kyoto-u.ac.jp

explores the convexity of the penalty term, and proposes a convexity-exploiting solution method; Section V presents numerical experiments; Section VI concludes this study.

D. Notation

Given an Euclidean n-dimensional vector space $\mathbb{R}^n$, we denote its nonnegative orthant by $\mathbb{R}^n_+$. Given two variables $v, w \in \mathbb{R}^n$, we denote their element-wise concatenation by $\xi = \xi(v,w) = [\xi_1^T, \cdots, \xi_i^T, \cdots, \xi_n^T]^T \in \mathbb{R}^{2n}$ with $\xi_i = [v_i, w_i]^T$, and their element-wise product (i.e., the Hadamard product) by $u = v \odot w$, where $u \in \mathbb{R}^n$ has elements given by $u_i = v_i w_i$. Given a differentiable function $f(x)$, we denote its Jacobian as $\nabla_x f \in \mathbb{R}^{m \times n}$ with $f : \mathbb{R}^n \to \mathbb{R}^m$, and its Hessian as $\nabla^2 f \in \mathbb{R}^{n \times n}$ with $f : \mathbb{R}^n \to \mathbb{R}$.

II. Problem Formulation

A. Optimal control of linear complementarity system

We consider the OCP of LCS in the form of:

$$\min_{x(\cdot),u(\cdot),\lambda(\cdot)} \int_0^T \underbrace{\frac{1}{2}(\|x(t)\|^2_{Q_x} + \|u(t)\|^2_{Q_u} + \|\lambda(t)\|^2_{Q_\lambda})}_{:=L_S(x(t),u(t),\lambda(t))} dt \tag{1a}$$

$$\text{s.t. } \dot{x}(t) = \underbrace{Ax(t) + Bu(t) + E\lambda(t)}_{:=f(x(t),u(t),\lambda(t))}, \tag{1b}$$

$$\eta(t) = \underbrace{Cx(t) + Du(t) + F\lambda(t)}_{:=g(x(t),u(t),\lambda(t))}, \tag{1c}$$

$$0 \le \lambda(t) \perp \eta(t) \ge 0, \tag{1d}$$

where $x : [0,T] \to \mathbb{R}^{n_x}$ is the differential state, $u : [0,T] \to \mathbb{R}^{n_u}$ is the control input, $\lambda : [0,T] \to \mathbb{R}^{n_\lambda}$ is the algebraic variable, and $\eta : [0,T] \to \mathbb{R}^{n_\lambda}$ is the auxiliary variable. The quadratic stage cost term $L_S : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_\lambda} \to \mathbb{R}$ in (1a) is formed by positive semidefinite diagonal matrices $Q_x \in \mathbb{R}^{n_x \times n_x}, Q_u \in \mathbb{R}^{n_u \times n_u}, Q_\lambda \in \mathbb{R}^{n_\lambda \times n_\lambda}$. We called (1b) - (1d) a LCS, with the ODE right-hand side function $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_\lambda} \to \mathbb{R}^{n_x}$ in (1b) formed by matrices $A \in \mathbb{R}^{n_x \times n_x}, B \in \mathbb{R}^{n_x \times n_u}, E \in \mathbb{R}^{n_x \times n_\lambda}$, and the affine function $g : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_\lambda} \to \mathbb{R}^{n_\lambda}$ in (1c) formed by matrices $C \in \mathbb{R}^{n_\lambda \times n_x}, D \in \mathbb{R}^{n_\lambda \times n_u}, F \in \mathbb{R}^{n_\lambda \times n_\lambda}$. The notation $\lambda \perp \eta$ in the complementarity condition (1d) means that $\lambda \odot \eta = 0$. Since the algebraic trajectory $\lambda(t)$ generally does not possess continuity properties, the LCS is a nonlinear and non-smooth dynamical system. The smoothness of the state trajectory $x(t)$ strongly depends on the *relative degree* r, which is the number of times to differentiate $\eta(t)$ (regarded as output) w.r.t. time until $\lambda(t)$ (regarded as input) exists explicitly. The higher the relative degree, the less smooth the system becomes, see section 2.4.1 in [3] for a qualitative description of the influence of r on the smoothness of $x(t)$. The affine path constraints in x and u can be incorporated but we ignore them to streamline the presentation.

We numerically solve the continuous-time OCP (1) using the direct multiple shooting method [7]. Specifically, the ODE (1b) is discretized using the implicit Euler method, and the complementarity conditions (1d) are enforced at each time point $t_n \in [0,T]$, leading to a NLP problem:

$$\min_{\boldsymbol{x},\boldsymbol{u},\boldsymbol{\lambda},\boldsymbol{\eta}} \sum_{n=1}^{N} L_S(x_n, u_n, \lambda_n)\Delta t, \tag{2a}$$

$$\text{s.t. } x_{n-1} - x_n + (Ax_n + Bu_n + E\lambda_n)\Delta t = 0, \tag{2b}$$

$$Cx_n + Du_n + F\lambda_n - \eta_n = 0, \tag{2c}$$

$$0 \le \lambda_n \perp \eta_n \ge 0, \quad n = 1, \ldots, N, \tag{2d}$$

with given x_0, where $x_n \in \mathbb{R}^{n_x}$, $\lambda_n \in \mathbb{R}^{n_\lambda}$, and $\eta_n \in \mathbb{R}^{n_\lambda}$ are the values of $x(t)$, $\lambda(t)$, and $\eta(t)$ at time point t_n, respectively, $u_n \in \mathbb{R}^{n_u}$ is the piecewise constant approximation of $u(t)$ in the interval $(t_{n-1}, t_n]$, N is the number of stages, and $\Delta t := T/N$ is the time step. Vectors $\boldsymbol{x} = [x_1^T, \cdots, x_N^T]^T \in \mathbb{R}^{Nn_x}$, $\boldsymbol{u} = [u_1^T, \cdots, u_N^T]^T \in \mathbb{R}^{Nn_u}$, $\boldsymbol{\lambda} = [\lambda_1^T, \cdots, \lambda_N^T]^T \in \mathbb{R}^{Nn_\lambda}$, and $\boldsymbol{\eta} = [\eta_1^T, \cdots, \eta_N^T]^T \in \mathbb{R}^{Nn_\lambda}$ are defined to collect all states, controls, algebraic variables, and auxiliary variables along the horizon, respectively.

B. Penalty, relaxation, and smoothing method for MPCC

The NLP problem (2) is an MPCC, which is extremely difficult to solve because the complementarity constraint (2d) lacks constraint regularity required by the NLP theory. Three MPCC-tailored methods that use off-the-shelf NLP solvers exist. The first involves penalizing the complementarity term $\lambda \perp \eta$ in the cost function rather than formulating it as a constraint. This leads to the following parameterized NLP, denoted by $\mathcal{P}_{comp}(\mu)$, where $\mu > 0$ is a penalty parameter:

$$\min_{\boldsymbol{x},\boldsymbol{u},\boldsymbol{\lambda},\boldsymbol{\omega}} \sum_{n=1}^{N} L_S(x_n, u_n, \lambda_n)\Delta t + \mu \sum_{n=1}^{N} \lambda_n^T \eta_n, \tag{3a}$$

$$\text{s.t. } x_{n-1} - x_n + (Ax_n + Bu_n + E\lambda_n)\Delta t = 0, \tag{3b}$$

$$Cx_n + Du_n + F\lambda_n - \eta_n = 0, \tag{3c}$$

$$\lambda_n \ge 0,\ \eta_n \ge 0, \quad n = 1, \ldots, N. \tag{3d}$$

The second involves recovering the constraint regularity using MPCC-tailored relaxation strategies. For instance, the Scholtes relaxation strategy [21] reformulates $0 \le \lambda \perp \eta \ge 0$ as $\lambda_i, \eta_i \ge 0, s - \lambda_i \eta_i \ge 0$ in an element-wise manner, where $s \ge 0$ is a relaxation parameter. This leads to the following parameterized NLP, denoted by $\mathcal{P}_{scholtes}(s)$:

$$\min_{\boldsymbol{x},\boldsymbol{u},\boldsymbol{\lambda},\boldsymbol{\omega}} \sum_{n=1}^{N} L_S(x_n, u_n, \lambda_n)\Delta t, \tag{4a}$$

$$\text{s.t. } x_{n-1} - x_n + (Ax_n + Bu_n + E\lambda_n)\Delta t = 0, \tag{4b}$$

$$Cx_n + Du_n + F\lambda_n - \eta_n = 0, \tag{4c}$$

$$sI_{n_\lambda \times 1} - \lambda_n \odot \eta_n \ge 0, \tag{4d}$$

$$\lambda_n \ge 0,\ \eta_n \ge 0, \quad n = 1, \ldots, N. \tag{4e}$$

The third involves reformulating the complementarity constraints as a system of smoothed equations using a smoothed C-function. One popular smoothed C-function is the Fisher–Burmeister (FB) smoothed function (Chapter 11, [6]):

$$\psi_{FB}(u, v, s) = \sqrt{u^2 + v^2 + s^2} - u - v, \tag{5}$$

where u, v are scalar variables and $s \geq 0$ is a smoothing parameter. It has the following properties:

$$\psi_{FB}(u, v, s) = 0 \;\Leftrightarrow\; u \geq 0,\; v \geq 0,\; uv = \frac{1}{2}s^2. \quad (6)$$

Employing ψ_{FB} to (2d) in an element-wise manner leads to the following parameterized NLP, denoted by $\mathcal{P}_{FB}(s)$:

$$\min_{\boldsymbol{x},\boldsymbol{u},\boldsymbol{\lambda},\boldsymbol{\omega}} \sum_{n=1}^{N} L_S(x_n, u_n, \lambda_n)\Delta t, \quad (7a)$$

$$\text{s.t.} \quad x_{n-1} - x_n + (Ax_n + Bu_n + E\lambda_n)\Delta t = 0, \quad (7b)$$

$$Cx_n + Du_n + F\lambda_n - \eta_n = 0, \quad (7c)$$

$$\psi_{FB}(\lambda_n, \eta_n, s) = 0, \quad n = 1, \ldots, N. \quad (7d)$$

Consequently, the solutions to the NLP problem (2) can be obtained using a continuation method that solves a sequence of parameterized NLP problems $\mathcal{P}_{comp}(\mu)$ (resp. $\mathcal{P}_{scholtes}(s)$ and $\mathcal{P}_{FB}(s)$), with $\mu \to \infty$ (resp. $s \to 0$). Nonetheless, for $\mathcal{P}_{comp}(\mu)$, the Hessian of product $\lambda_n^T \eta_n$ is constant and indefinite. Thus, directly ignoring this matrix or forcibly modifying it into a positive semidefinite matrix severely distorts the curvature of the cost function. Furthermore, such modifications often require multiple matrix factorizations, which are expensive; Regarding $\mathcal{P}_{scholtes}(s)$, NLP solvers may stall or fail when relaxation parameter s is closed to zero because the feasible interior constructed by (4d) (4e) shrinks toward an empty set as $s \to 0$. Regarding $\mathcal{P}_{FB}(s)$, ψ_{FB} tends to be non-smooth at the origin as $s \to 0$; thus, the fast-changing gradient may destabilize gradient-based NLP solvers. These limitations highlight the need for more efficient reformulations and methods to solve MPCCs.

III. Proposed Penalty Reformulation

A. Motivation: D-gap function for complementarity problem

Algorithms for solving the complementarity problem, that is, finding a pair (λ, η) that satisfies $0 \leq \lambda \perp \eta \geq 0$, generally fall into two categories based on various reformulations of the complementarity problem. One is the *equation-based* algorithm, which reformulates the complementarity problem as a system of equations using the C-function (Chapter 9, [6]). The other is the *optimization-based* algorithm, which reformulates the complementarity problem as a differentiable optimization problem that minimizes a tailored merit function (Chapter 10, [6]). The proposed penalty reformulation was motivated by the latter approach.

This study considers a differentiable merit function, known as the *D-gap function*, which reformulates the complementarity problem as an *unconstrained* optimization problem. The D-gap function was first introduced in [20] and then extended to the solving algorithm for *variational inequalities* [19], [24]. Its definition is provided below:

Definition 1 (D-gap function): Let $\lambda, \eta \in \mathbb{R}^{n_\lambda}$ be two variables, a, b be two given constants satisfying $b > a > 0$, and $\varphi^{ab} : \mathbb{R}^{n_\lambda} \times \mathbb{R}^{n_\lambda} \to \mathbb{R}$ be a function given by:

$$\varphi^{ab}(\lambda, \eta) = \varphi^a(\lambda, \eta) - \varphi^b(\lambda, \eta), \quad (8)$$

where $\varphi^a, \varphi^b : \mathbb{R}^{n_\lambda} \times \mathbb{R}^{n_\lambda} \to \mathbb{R}$ are the functions defined as follows:

$$\varphi^a(\lambda, \eta) = \frac{1}{2a}(\|\eta\|_2^2 - \|\max(0, \eta - a\lambda)\|_2^2), \quad (9a)$$

$$\varphi^b(\lambda, \eta) = \frac{1}{2b}(\|\eta\|_2^2 - \|\max(0, \eta - b\lambda)\|_2^2). \quad (9b)$$

We call $\varphi^{ab}(\lambda, \eta)$ the *D-gap function* for the complementarity problem, where D stands for Difference.

The properties of φ^{ab} on which we concentrate are summarized in the following theorem (Theorem 10.3.3, [6]).

Theorem 1: The following two statements are valid for the D-gap function $\varphi^{ab}(\lambda, \eta)$ given by (8):

- (*Equivalence*) $\varphi^{ab}(\lambda, \eta) \geq 0, \forall \lambda, \eta \in \mathbb{R}^{n_\lambda}$. Furthermore, $\varphi^{ab}(\lambda, \eta) = 0$ if and only if $0 \leq \lambda \perp \eta \geq 0$. Hence, a pair (λ, η) satisfies $0 \leq \lambda \perp \eta \geq 0$ if and only if it is the global solution to the following *unconstrained* optimization problem:

$$\min_{\lambda, \eta \in \mathbb{R}^{n_\lambda}} \; \varphi^{ab}(\lambda, \eta). \quad (10)$$

- (*Differentiability*) $\varphi^{ab}(\lambda, \eta)$ is a continuously differentiable function.

B. Gap penalty reformulation

By replacing the complementarity constraints (2d) with the D-gap function, we propose a new penalty reformulation for (2), which is the following parameterized NLP problem, denoted by $\mathcal{P}_{gap}(\mu)$, where $\mu > 0$ is the penalty parameter:

$$\min_{\boldsymbol{x},\boldsymbol{u},\boldsymbol{\lambda},\boldsymbol{\omega}} \sum_{n=1}^{N} L_S(x_n, u_n, \lambda_n)\Delta t + \mu \sum_{n=1}^{N} \varphi^{ab}(\lambda_n, \eta_n), \quad (11a)$$

$$\text{s.t.} \quad x_{n-1} - x_n + (Ax_n + Bu_n + E\lambda_n)\Delta t = 0, \quad (11b)$$

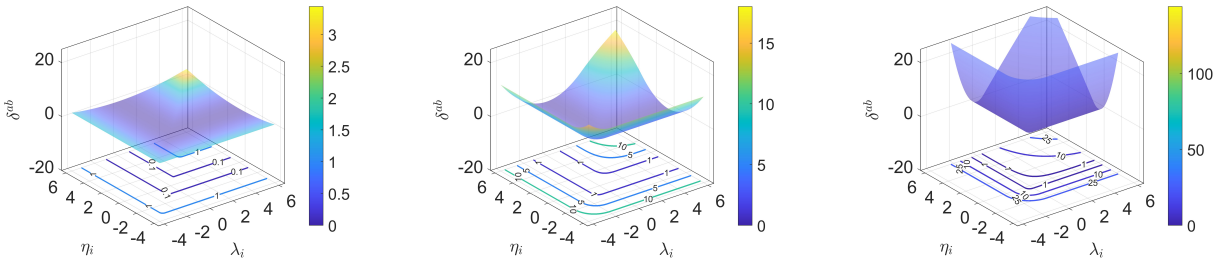
$$Cx_n + Du_n + F\lambda_n - \eta_n = 0, \quad n = 1, \ldots, N. \quad (11c)$$

We call $\mathcal{P}_{gap}(\mu)$ the *gap penalty reformulation*. Similarly, we can obtain the solutions to the NLP problem (2) by solving a sequence of $\mathcal{P}_{gap}(\mu)$ with $\mu \to \infty$. Furthermore, $\mathcal{P}_{gap}(\mu)$ exhibits the following two favorable properties: First, compared with $\mathcal{P}_{comp}(\mu)$ and $\mathcal{P}_{scholtes}(s)$, it does not involve inequality constraints $\lambda_n, \eta_n \geq 0$ and thereby possesses a more concise constraint system; Second, unlike relaxation or smoothing reformulations, it possesses a differentiable constraint system with constraint regularity regardless of the choice of μ. The main difficulty is that $\varphi^{ab}(\lambda, \eta)$ is *nonconvex*, and as stated in Theorem 1, we have to *globally* minimize $\varphi^{ab}(\lambda, \eta)$ to guarantee the satisfaction of $0 \leq \lambda \perp \eta \geq 0$. Therefore, in the next section, we aim to mitigate the nonconvexity of $\varphi^{ab}(\lambda, \eta)$, and then solve $\mathcal{P}_{gap}(\mu)$ efficiently with a given μ using convexity-exploiting solution methods.

IV. Proposed Solution Method

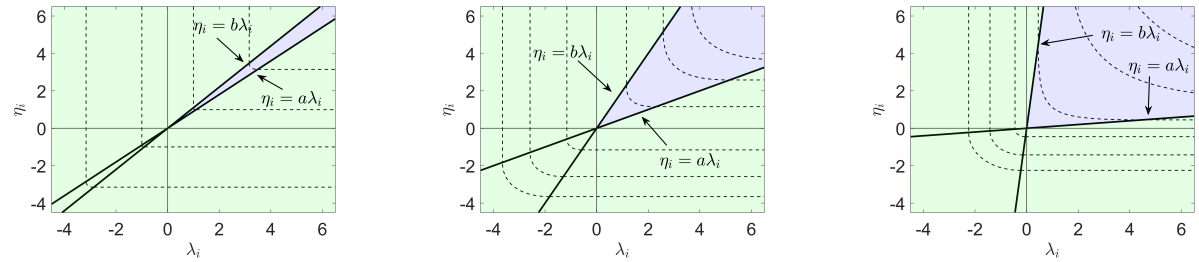
A. Convexity structure of the D-gap function

First, we discuss certain properties of the D-gap function $\varphi^{ab}(\lambda, \eta)$. Note that, $\varphi^{ab}(\lambda, \eta)$ given by (8) exhibits *partial*



(a) $a = 0.9, b = 1.1$ (b) $a = 0.5, b = 2$ (c) $a = 0.1, b = 10$

Fig. 1. Contour of $\delta^{ab}(\lambda_i, \eta_i)$ under various parameter combinations.



(a) $a = 0.9, b = 1.1$ (b) $a = 0.5, b = 2$ (c) $a = 0.1, b = 10$

Fig. 2. Feasible region partitioning under various parameter combinations: Hessian $\nabla^2\delta^{ab}(\xi_i)$ with $\xi_i = [\lambda_i, \eta_i]^T$ is indefinite when ξ_i is in the blue region and is positive semidefinite when ξ_i is in the green region.

separability (Definition 7.1, [7]) and can be decomposed as the sum of n_λ scalar subfunctions:

$$\varphi^{ab}(\lambda, \eta) = \sum_{i=1}^{n_\lambda} \delta^{ab}(\lambda_i, \eta_i), \tag{12}$$

where $\delta^{ab} : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$ is the subfunction given by:

$$\begin{aligned}\delta^{ab}(\lambda_i, \eta_i) =& \frac{b-a}{2ab}\eta_i^2 - \frac{1}{2a}\{\max(0, \eta_i - a\lambda_i)\}^2 \\ &+ \frac{1}{2b}\{\max(0, \eta_i - b\lambda_i)\}^2,\end{aligned} \tag{13}$$

which can be further expanded as a piecewise function:

$$\delta^{ab}(\lambda_i, \eta_i) = \begin{cases} \frac{b-a}{2}\lambda_i^2, & \eta_i \geq b\lambda_i \text{ and } \eta_i \geq a\lambda_i, \\ -\frac{a}{2}\lambda_i^2 + \lambda_i\eta_i - \frac{1}{2b}\eta_i^2, & b\lambda_i > \eta_i > a\lambda_i, \\ \frac{b-a}{2ab}\eta_i^2, & \eta_i \leq b\lambda_i \text{ and } \eta_i \leq a\lambda_i, \\ \frac{b}{2}\lambda_i^2 - \lambda_i\eta_i + \frac{1}{2a}\eta_i^2, & b\lambda_i < \eta_i < a\lambda_i. \end{cases} \tag{14}$$

δ^{ab} is also a nonnegative function, and its contour and feasible region partitioning under various parameter combinations are shown in Fig 1 and 2, respectively.

Let $\mathcal{I} = \{1, \cdots, n_\lambda\}$ be a finite set of indices, with two subsets $\mathcal{I}_{ind}$ and $\mathcal{I}_{psd}$ defined by: $\mathcal{I}_{ind} = \{i \in \mathcal{I} \mid b\lambda_i > \eta_i > a\lambda_i\}$ and $\mathcal{I}_{psd} = \mathcal{I} \setminus \mathcal{I}_{ind}$, respectively. Let $\xi = \xi(\lambda, \eta) \in \mathbb{R}^{2n_\lambda}$ be the element-wise concatenation of λ, η with $\xi_i = [\lambda_i, \eta_i]^T$. From (14), we have that the Hessian $\nabla^2\delta^{ab}(\xi_i)$ is indefinite only when $i \in \mathcal{I}_{ind}$, and is positive semidefinite when $i \in \mathcal{I}_{psd}$. Because the Hessian of $\varphi^{ab}(\lambda, \eta)$ is block diagonal in terms of the variable ξ:

$$\nabla^2\varphi^{ab}(\xi) = \text{diag}\{\nabla^2\delta^{ab}(\xi_1), \cdots, \nabla^2\delta^{ab}(\xi_{n_\lambda})\}, \tag{15}$$

the negative eigenvalues of $\nabla^2\varphi^{ab}(\xi)$ can only originate from the indefinite submatrices $\nabla^2\delta^{ab}(\xi_i), i \in \mathcal{I}_{ind}$, which are constant and parameterized by $b > a > 0$:

$$\nabla^2\delta^{ab}(\xi_i) = \begin{bmatrix} -a & 1 \\ 1 & -\frac{1}{b} \end{bmatrix}, i \in \mathcal{I}_{ind}, \tag{16}$$

with two eigenvalues $\kappa_{min}, \kappa_{max}$ that can be pre-computed as follows:

$$\kappa_{min} = -\frac{1}{2}(a + \frac{1}{b}) - \frac{1}{2}\sqrt{(\frac{1}{b} - a)^2 + 4} < 0, \tag{17a}$$

$$\kappa_{max} = -\frac{1}{2}(a + \frac{1}{b}) + \frac{1}{2}\sqrt{(\frac{1}{b} - a)^2 + 4} > 0. \tag{17b}$$

The definiteness of submatrices $\nabla^2\delta^{ab}(\xi_i)$ motivates us to explore the convexity structure of $\varphi^{ab}(\xi)$. Specifically, we construct a positive semidefinite matrix $B(\xi) \in \mathbb{R}^{2n_\lambda \times 2n_\lambda}$ to approximate the Hessian $\nabla^2\varphi^{ab}(\xi)$ by reserving positive semidefinite submatrices $\nabla^2\delta^{ab}(\xi_i), i \in \mathcal{I}_{psd}$, and regularizing only those indefinite submatrices $\nabla^2\delta^{ab}(\xi_i), i \in \mathcal{I}_{ind}$. Regularization is achieved by defining a regularization matrix $B_{reg} \in \mathbb{R}^{2n_\lambda \times 2n_\lambda}$ using the negative eigenvalue κ_{min} and indicator variable $\omega \in \mathbb{R}^{n_\lambda}$:

$$B_{reg} = -\kappa_{min}\text{diag}\{\omega_1 I_{2\times 2}, \cdots, \omega_{n_\lambda} I_{2\times 2}\}, \tag{18}$$

where the elements of the indicator variable ω are given by:

$$\omega_i = \begin{cases} 0, & \text{if } i \in \mathcal{I}_{psd}, \\ 1, & \text{if } i \in \mathcal{I}_{ind}. \end{cases} \tag{19}$$

Then, the Hessian approximation matrix $B(\xi)$ is given by:

$$B(\xi) = \nabla^2 \varphi^{ab}(\xi) + B_{reg}. \tag{20}$$

This Hessian modification strategy (20) is practical for the convexity-exploiting solution method of $\mathcal{P}_{gap}(\mu)$. On the one hand, if set $\mathcal{I}_{ind}$ is empty, then $B(\xi)$ equals to the exact Hessian $\nabla^2\varphi^{ab}(\xi)$ which is positive semidefinite; On the other hand, if set $\mathcal{I}_{ind}$ is nonempty, then B_{reg} ensures $B(\xi) \succeq 0$ by employing the *diagonal modification* strategy (Section 3.4, [17]) to regularize indefinite submatrices $\nabla^2\delta^{ab}(\xi_i), i \in \mathcal{I}_{ind}$ to be positive semidefinite. Here, the modification strategy uses only a pre-computed regularization matrix $-\kappa_{min} I_{2\times 2}$ and does not require matrix factorization of $\nabla^2\varphi^{ab}(\xi)$. These advantages can improve the computation efficiency of the solution methods that use this Hessian modification strategy.

B. Algorithm overview

By employing the Hessian modification strategy (20) in a convexity-exploiting method known as *sequential convex quadratic programming* [16], we propose an efficient method for solving the NLP problem $\mathcal{P}_{gap}(\mu)$ with a given μ. For brevity, we first rewrite the problem $\mathcal{P}_{gap}(\mu)$ as a general NLP problem. We collect all decision variables into a vector $\boldsymbol{z} = [z_1^T, \cdots z_n^T, \cdots z_N^T]^T$ with $z_n = [x_n^T, u_n^T, \xi_n^T]^T$, and $\xi_n = \xi(\lambda_n, \eta_n) \in \mathbb{R}^{2n_\lambda}$ is the element-wise concatenation of λ_n, η_n. We collect all equality constraints into a vector $\boldsymbol{h}(\boldsymbol{z}) = [h_1^T, \cdots h_n^T, \cdots h_N^T]^T$ with

$$h_n = \begin{bmatrix} x_{n-1} - x_n + (Ax_n + Bu_n + E\lambda_n)\Delta t \\ Cx_n + Du_n + F\lambda_n - \eta_n \end{bmatrix}.$$

We define the shorthand $\boldsymbol{J}_S(\boldsymbol{z}) = \sum_{n=1}^N L_S(x_n, u_n, \lambda_n)\Delta t$, $\boldsymbol{J}_P(\boldsymbol{z}, \mu) = \mu \sum_{n=1}^N \varphi^{ab}(\lambda_n, \eta_n)$, and $\boldsymbol{J}(\boldsymbol{z}, \mu) = \boldsymbol{J}_S(\boldsymbol{z}) + \boldsymbol{J}_P(\boldsymbol{z}, \mu)$, then $\mathcal{P}_{gap}(\mu)$ can be rewritten as the following general NLP problem:

$$\min_{\boldsymbol{z}} \ \boldsymbol{J}(\boldsymbol{z}, \mu), \tag{21a}$$
$$\text{s.t.} \quad \boldsymbol{h}(\boldsymbol{z}) = 0. \tag{21b}$$

Let the Lagrangian of the NLP problem (21) be:

$$\mathcal{L}(\boldsymbol{z}, \boldsymbol{\gamma}_h, \mu) = \boldsymbol{J}(\boldsymbol{z}, \mu) + \boldsymbol{\gamma}_h^T \boldsymbol{h}(\boldsymbol{z}), \tag{22}$$

where $\boldsymbol{\gamma}_h \in \mathbb{R}^{n_h}$ is the Lagrangian multipliers for constraints $\boldsymbol{h}$. The Karush–Kuhn–Tucker (KKT) conditions associated with the NLP problem (21) are defined as follows:

$$\nabla_{\boldsymbol{z}} \mathcal{L}(\boldsymbol{z}, \boldsymbol{\gamma}_h, \mu) = 0, \tag{23a}$$
$$\boldsymbol{h}(\boldsymbol{z}) = 0. \tag{23b}$$

The pair $(\boldsymbol{z}^*, \boldsymbol{\gamma}_h^*)$ satisfying (23) is referred to as the KKT point of the NLP problem (21). The SCQP-type method is an iterative method that solves a sequence of convex QP approximations for an NLP problem (21). It terminates at an iterate $(\boldsymbol{z}^k, \boldsymbol{\gamma}_h^k)$ which is a KKT point of the NLP problem (21) with the given tolerance:

$$\|\nabla_{\boldsymbol{z}} \mathcal{L}(\boldsymbol{z}^k, \boldsymbol{\gamma}_h^k, \mu)\|_\infty \le \epsilon_D, \tag{24a}$$
$$\|\boldsymbol{h}(\boldsymbol{z}^k)\|_\infty \le \epsilon_P, \tag{24b}$$

Algorithm 1 Proposed SCQP-type method to solve the penalty problem (21) with a given $\mu > 0$

Input: μ, $\boldsymbol{z}^0$
Output: $\boldsymbol{z}^*$

Initialization: $\boldsymbol{z}^k \leftarrow \boldsymbol{z}^0, \boldsymbol{\gamma}_h^k \leftarrow 0$.
Initialization: pre-compute $\kappa_{min}, \boldsymbol{H}_S, \nabla_{\boldsymbol{z}}\boldsymbol{h}$
for $k = 1$ to k_{max} **do**
  Step 1. evaluate functions and derivatives:
  $\boldsymbol{J}^k, \boldsymbol{h}^k, \nabla_{\boldsymbol{z}}\boldsymbol{J}^k, \boldsymbol{H}_P^k \leftarrow \boldsymbol{z}^k, \mu, \kappa_{min}$.
  Step 2. check termination condition:
  if (24) is satisfied **then**
    $\boldsymbol{z}^* \leftarrow \boldsymbol{z}^k$, and break iteration routine.
  end if
  Step 3. evaluate search direction and multiplier:
  $\Delta\boldsymbol{z}, \hat{\boldsymbol{\gamma}}_h^{k+1} \leftarrow$ solve a sparse convex QP (25).
  Step 4. merit line search globalization:
  $\beta \leftarrow$ using (30)
  $\alpha \leftarrow \nu_D \alpha$ until (31) is satisfied
  if $\alpha = \alpha_{min}$ but still fails to satisfy (31) **then**
    $\alpha \leftarrow \alpha_{opt}$ using (35)
  end if
  Step 5. update iterate:
  $\boldsymbol{z}^{k+1} \leftarrow \boldsymbol{z}^k + \alpha\Delta\boldsymbol{z}$, $\boldsymbol{\gamma}_h^{k+1} \leftarrow \boldsymbol{\gamma}_h^k + \alpha(\hat{\boldsymbol{\gamma}}_h^{k+1} - \boldsymbol{\gamma}_h^k)$
end for

where $\epsilon_P, \epsilon_D > 0$ are the primal and dual residual tolerance, respectively. Algorithm 1 summarizes the proposed SCQP-type method and the details are presented below.

C. Evaluation of the search direction

Let the current iterate be $\boldsymbol{z}^k$ and γ_h^k, then a new search direction $\Delta\boldsymbol{z}$ and an estimate of the new multiplier $\hat{\gamma}_h^{k+1}$ can be obtained by solving a structured sparse convex QP:

$$\min_{\Delta\boldsymbol{z}} \ \frac{1}{2}\Delta\boldsymbol{z}^T \boldsymbol{H}^k \Delta\boldsymbol{z} + \nabla_{\boldsymbol{z}}\boldsymbol{J}^k \Delta\boldsymbol{z}, \tag{25a}$$
$$\text{s.t.} \quad \boldsymbol{h}^k + \nabla_{\boldsymbol{z}}\boldsymbol{h}\Delta\boldsymbol{z} = 0. \tag{25b}$$

Here, the constraint Jacobian $\nabla_{\boldsymbol{z}}\boldsymbol{h}$ is constant and structured sparse, and $\boldsymbol{H}^k$ is the Lagrangian Hessian at the current iterate. Because the constraints incorporated in the NLP problem (21) are affine, $\boldsymbol{H}^k$ involves only the curvature of the cost function and can be further split into two parts:

$$\boldsymbol{H}^k = \boldsymbol{H}_S + \boldsymbol{H}_P^k, \tag{26}$$

where $\boldsymbol{H}_S$ is the exact Hessian of $\boldsymbol{J}_S$ and $\boldsymbol{H}_P^k$ approximates the Hessian of $\boldsymbol{J}_P$ at the current iterate $\boldsymbol{z}^k$. Both matrices $\boldsymbol{H}_S$ and $\boldsymbol{H}_P^k$ are positive semidefinite and have a block diagonal sparse structure, where $\boldsymbol{H}_S$ is formed by Q_x, Q_u, Q_λ and is thereby constant throughout the iterations, whereas $\boldsymbol{H}_P^k$ is formed by $B(\xi_n^k)$ and must be updated at each iteration. The sparse convex QP (25) can be solved very efficiently, either by the state-of-the-art sparse QP solvers or by solving the following sparse linear system using the sparse linear algebraic routine:

$$\begin{bmatrix} \boldsymbol{H}^k & \nabla_{\boldsymbol{z}}\boldsymbol{h}^T \\ \nabla_{\boldsymbol{z}}\boldsymbol{h} & 0 \end{bmatrix} \begin{bmatrix} \Delta\boldsymbol{z} \\ \hat{\gamma}_h^{k+1} \end{bmatrix} = - \begin{bmatrix} (\nabla_{\boldsymbol{z}}\boldsymbol{J}^k)^T \\ \boldsymbol{h}^k \end{bmatrix}. \tag{27}$$

D. Globalization strategy

After obtaining the search direction $\Delta \boldsymbol{z}$, we use the merit line search globalization method to find a new iterate $\boldsymbol{z}^{k+1} = \boldsymbol{z}^k + \alpha \Delta \boldsymbol{z}$ with stepsize α. We define an ℓ_1 non-differentiable exact penalty function $\Theta(\boldsymbol{z}, \mu)$ to simultaneously measure the cost and constraint violation (Chapter 18, [17]):

$$\Theta(\boldsymbol{z}, \mu) = \boldsymbol{J}(\boldsymbol{z}, \mu) + \beta \|\boldsymbol{h}(\boldsymbol{z})\|_1, \tag{28}$$

where $\beta > 0$ is a weighting parameter. We need to specify β such that a stepsize $\alpha \in (0, 1]$ exists and step $\alpha \Delta \boldsymbol{z}$ provides a sufficient decrease in the merit function. Specifically, let $\mathbb{D}\Theta^k$ be the directional derivative of the merit function (28) at $\boldsymbol{z}^k$ along $\Delta \boldsymbol{z}$. We expect the choice of β can enforce $\mathbb{D}\Theta^k$ to be sufficiently negative:

$$\mathbb{D}\Theta^k = \nabla_{\boldsymbol{z}} \boldsymbol{J}^k \Delta \boldsymbol{z} - \beta \|\boldsymbol{h}^k\|_1 \leq -\rho \beta \|\boldsymbol{h}^k\|_1, \tag{29}$$

with parameter $\rho \in (0, 1)$, where the equality in (29) is the explicit expression of $\mathbb{D}\Theta^k$ (Theorem 18.2, [17]). This suggests that an appropriate candidate β should satisfy the following conditions:

$$\beta \geq \beta_{trail} := \frac{\nabla_{\boldsymbol{z}} \boldsymbol{J}^k \Delta \boldsymbol{z}}{(1-\rho)\|\boldsymbol{h}^k\|_1}. \tag{30}$$

We can then apply a backtracking line search strategy to evaluate a new iterate $\boldsymbol{z}^{k+1} = \boldsymbol{z}^k + \alpha \Delta \boldsymbol{z}$. The trail stepsize $\alpha \leftarrow \nu_\alpha \alpha$ is gradually reduced from $\alpha_{max} = 1$ with $\nu_\alpha \in (0, 1)$ until the Armijo condition is satisfied:

$$\Theta(\boldsymbol{z}^k + \alpha \Delta \boldsymbol{z}, \mu) \leq \Theta(\boldsymbol{z}^k, \mu) + \nu_D \alpha \mathbb{D}\Theta^k, \tag{31}$$

where $\nu_D \in (0, 1)$ is the desired reduction in Θ. In our setting $\rho = 0.1$, $\nu_\alpha = 0.5$, and $\nu_D = 10^{-4}$.

Occasionally, the backtracking strategy may stall or fail to find an appropriate new iterate if the acceptable stepsize for the Armijo condition (31) is arbitrarily small. Thus, if a trail stepsize $\alpha = \alpha_{min}$ (e.g., $\alpha_{min} = 10^{-4}$) still fails to satisfy (31), we switch to computing an optimal stepsize α_{opt} using an approximate model of the merit function $\Theta(\boldsymbol{z}, \mu)$. Specifically, we define the (piecewise) quadratic model of $\Theta(\boldsymbol{z}, \mu)$ at $\boldsymbol{z}^k$ as:

$$\begin{aligned} q_\beta(\alpha) =& \boldsymbol{J}^k + \alpha \nabla_{\boldsymbol{z}} \boldsymbol{J}^k \Delta \boldsymbol{z} + \frac{1}{2}\alpha^2 \Delta \boldsymbol{z}^T \boldsymbol{H}^k \Delta \boldsymbol{z} \\ &+ \beta \|\boldsymbol{h}^k + \alpha \nabla_{\boldsymbol{z}} \boldsymbol{h} \Delta \boldsymbol{z}\|_1. \end{aligned} \tag{32}$$

The optimal stepsize is then computed to maximize the decrease in the quadratic model:

$$\max_{\alpha \in (0,1]} \ \Delta q_\beta(\alpha) := q_\beta(0) - q_\beta(\alpha), \tag{33}$$

where $\Delta q_\beta(\alpha)$ can be explicitly written down as:

$$\Delta q_\beta(\alpha) = -\frac{1}{2}\alpha^2 \Delta \boldsymbol{z}^T \boldsymbol{H}^k \Delta \boldsymbol{z} - \alpha \underbrace{(\nabla_{\boldsymbol{z}} \boldsymbol{J}^k \Delta \boldsymbol{z} - \beta \|\boldsymbol{h}^k\|_1)}_{\mathbb{D}\Theta^k < 0}. \tag{34}$$

Thus, the analytic solution to the problem (33) is:

$$\hat{\alpha}_{opt} = -\frac{\mathbb{D}\Theta^k}{\Delta \boldsymbol{z}^T \boldsymbol{H}^k \Delta \boldsymbol{z}} > 0, \tag{35}$$

and we select $\alpha_{opt} = \min(1, \hat{\alpha}_{opt})$.

V. Numerical Simulation

A. Benchmark problems

We define a benchmark problem set including 11 different continuous-time OCPs of the LCS in the form of (1), which are taken from [22] and listed in Table I. We discretize each continuous-time OCP using six different discretization stages: $N = \{50, 80, 100, 200, 250, 400\}$. Thus, the problem set includes 66 different MPCCs in the form of (2). Problem details and solutions are available at https://github.com/KY-Lin22/Penalty-Gap-OCP-LCS.

B. Implementation details

The proposed SCQP-type method is implemented in MATLAB R2023b using the CasADi symbolic framework [2]. All experiments were performed on a laptop with a 1.80 GHz Intel Core i7-8550U. We solve the sparse QP (25) by using the Matlab built-in sparse linear algebraic routine to solve (27). We specify the termination tolerance as $\epsilon_P = 10^{-8}$, $\epsilon_D = 10^{-6}$, and $k_{max} = 500$. The proposed method is referred to as the *penalty (gap) method* and is compared with the four other MPCC-tailored methods listed below:

- *penalty (complementarity) method*: solving $\mathcal{P}_{comp}(\mu)$, that is, the parameterized NLP problem (3);
- *relaxation (Scholtes) method*: solving $\mathcal{P}_{Scholtes}(s)$, that is, the parameterized NLP problem (4);
- *relaxation (Lin-Fukushima) method*: solving the parameterized NLP problem that reformulates the complementarity constraints (2d) using the Lin-Fukushima relaxation strategy [15]. This strategy reformulates $0 \leq \lambda \perp \eta \geq 0$ as $s^2 - \lambda_i \eta_i \geq 0, (\lambda_i + s)(\eta_i + s) - s^2 \geq 0$ in an element-wise manner with $s \geq 0$ the relaxation parameter;
- *smoothing (FB) method*: solving $\mathcal{P}_{FB}(s)$, that is, the parameterized NLP problem (7).

The aforementioned comparison methods all utilize a well-developed NLP solver IPOPT [23] with default settings, and are employed through the CasADi interface. Overall, the selected comparison methods are among the most commonly used and effective MPCC-tailored methods currently available, based on the existing benchmark test [11], [18].

All methods are performed using the continuation method with the same initial guess (unit vector). Regarding penalty-type methods, we increase the penalty parameter by $\mu^{j+1} = \min(1.2\mu^j, \mu^*)$ from $\mu^0 = 10$ to $\mu^* = 10^5$. For the relaxation- and smoothing-type methods, we set the parameter sequence to $s^j = \frac{1}{\mu^j}$. Given a pair (λ, η), we measure the violation of the complementarity condition using the *natural residual function* $\Phi(\lambda, \eta) := \lambda - \min(0, \lambda - \eta)$. We have $\Phi = 0$ if and only if λ, η satisfy $0 \leq \lambda \perp \eta \geq 0$. The continuation step is terminated when the subproblem finds a solution whose natural residual falls below the given tolerance $\epsilon_{nat} = 10^{-2}$. Performances are compared in terms of the cost function value and computation time using the Dolan-Moré performance profiles [5]. Specifically, for the benchmark results obtained by running n_s solvers on n_p problems, let $t_{p,s}$ be the *performance measure* (e.g.,

TABLE I

DETAILS OF THE BENCHMARK PROBLEMS

No.	Name	Brief Introduction	Source
1	Vieira-LCS-Analytic-1	Analytical 1D example	Example 1 in [22], with $x_0 = 1$
2	Vieira-LCS-Analytic-2	Analytical 1D example	Example 1 in [22], with $x_0 = -1$
3	Vieira-LCS-Rel-Deg-One	Relative degree one example	Example 2 in [22]
4	Vieira-LCS-High-Dim	Higher dimensional example	Example 3 in [22]
5	Vieira-LCS-Without-Penalty	Stage cost does not penalize λ	Example 4 in [22]
6	Vieira-LCS-With-Penalty-1	Stage cost penalizes λ	Example 5 in [22], with $\alpha = 10$
7	Vieira-LCS-With-Penalty-2	Stage cost penalizes λ	Example 5 in [22], with $\alpha = 1$
8	Vieira-LCS-With-Penalty-3	Stage cost penalizes λ	Example 5 in [22], with $\alpha = 0.1$
9	Vieira-LCS-Control-Jump	Optimal control admitting jumps	Example 6 in [22]
10	Vieira-LCS-State-Jump-1	State admitting jumps	Example 7 in [22], with $\alpha = 10$
11	Vieira-LCS-State-Jump-2	State admitting jumps	Example 7 in [22], with $\alpha = 1$

computation time) of solver s on problem p, and let $r_{p,s}$ be the *performance ratio* between $t_{p,s}$ and the best performance measure of any solver on problem p:

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,i}, i = 1, \cdots, n_s\}}. \tag{36}$$

The performance profile of solver s is defined as the *(cumulative) distribution function* $\rho_s(\tau)$:

$$\rho_s(\tau) = \frac{\text{number of problem that } \ r_{p,s} \leq \tau}{n_p}, \tag{37}$$

with $\tau \geq 1$ the time factor. Intuitively, $\rho_s(1)$ is the probability that the solver s is the best.

C. Comparison between various penalty (gap) methods

From (14), it can be inferred that parameters a and b determine not only the curvature of the D-gap function but also the region where the D-gap function exhibits convexity. Therefore, we first compare the performance of the penalty (gap) method for different parameter combinations.

As demonstrated in Fig. 3(a), solutions obtained by the penalty (gap) method under various parameter combinations have very similar cost function values. Therefore, the primal focus of the comparison is the computation time. From Fig. 3(b), we can conclude that setting the values of a, b closer can speed up the computation, because these combinations, e.g., $a = 0.9, b = 1.1$, can expand the region where the D-gap function exhibits convexity and the exact Hessian can thus be used. However, in principle, they should not be set infinitely close, because the curvature of the D-gap function will be weakened. By contrast, setting a close to zero and b close to infinity, e.g., $a = 0.1, b = 10$, can achieve a larger curvature, but the function tends to be non-smooth. Furthermore, these combinations reduce the region where the D-gap function exhibits convexity, which may lead to extensive Hessian modifications during iteration. Fig. 1 and 2 provide an intuitive geometric interpretation for these conclusions. Overall, we specify $a = 0.9, b = 1.1$ as the default values of the proposed method.

D. Comparison between various MPCC-tailored methods

Finally, we compare the penalty (gap) method with four other MPCC-tailored methods. As demonstrated in Fig. 4(a), almost all methods converge to solutions with similar cost function values. Among them, the relaxation (Lin-Fukushima) method exhibits a slight advantage, whereas the penalty (gap) method appears to be trapped in solutions with slightly higher cost function values in certain problems. Similarly, the focus of the comparison is still on the computation time. As demonstrated in Fig. 4(b), the penalty (gap) method demonstrates a significant advantage: with the highest probability, which is 72.7%, of being the fastest solver to find an optimal solution. This is a promising result, primarily benefiting from a more concise constraint system and the utilization of second-order information on the penalty term.

VI. CONCLUSION

To solve the OCP of LCS efficiently, we propose a novel penalty reformulation for the complementarity constraint using the D-gap function. The proposed penalty reformulation exhibits a concise, differentiable constraint system with constraint regularity. Furthermore, the penalty term has a convexity structure that can be exploited using convexity-exploiting solution methods. We solve the reformulated OCP efficiently using an SCQP-type method. The benchmark test demonstrated the proposed SCQP-type method outperforms all other MPCC-tailored methods in terms of computation time. Future studies focus on two aspects: we aim to analyze the MPCC-tailored stationarity properties of the limit point of the proposed method, and extend the proposed gap penalty method to the more general non-smooth dynamical system.

REFERENCES

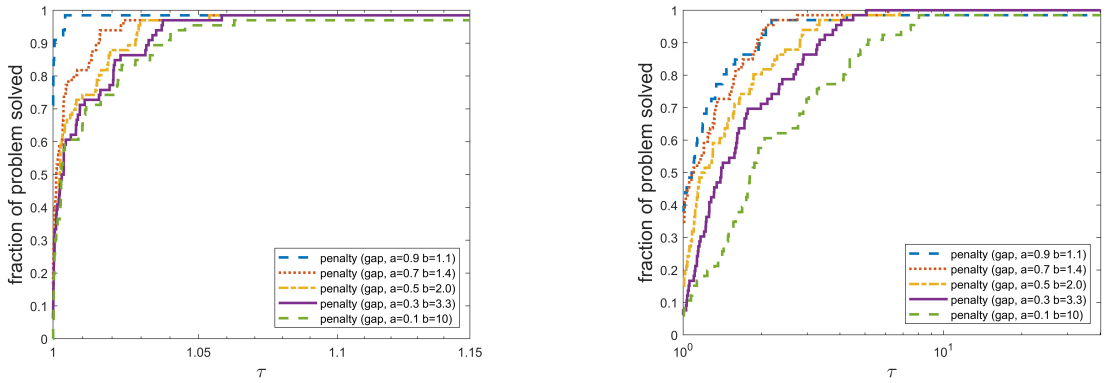
[1] V. Acary and B. Brogliato, *Numerical Methods for Nonsmooth Dynamical Systems: Applications in Mechanics and Electronics*. Springer Science & Business Media, 2008.

[2] J. Andersson, J. Gillis, G. Horn, J. Rawlings, and M. Diehl, "CasADi: a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[3] B. Brogliato and A. Tanwani, "Dynamical systems coupled with monotone set-valued operators: Formalisms, applications, well-posedness, and stability," *SIAM Review*, vol. 62, no. 1, pp. 3–129, 2020.

[4] R. W. Cottle, J.-S. Pang, and R. E. Stone, *The Linear Complementarity Problem*. SIAM, 2009.

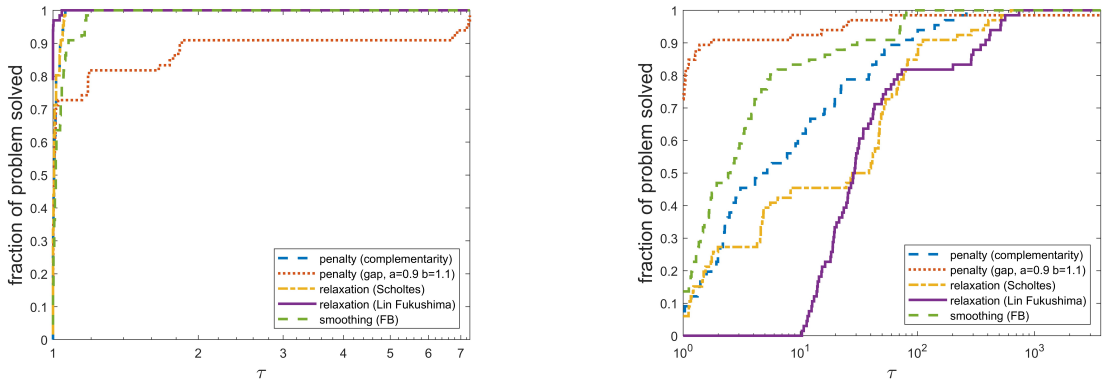
[5] E. D. Dolan and J. J. Moré, "Benchmarking optimization software with performance profiles," *Mathematical programming*, vol. 91, pp. 201–213, 2002.



(a) Comparison of cost function value.

(b) Comparison of computation time.

Fig. 3. Performance profiles of various penalty (gap) methods for the benchmark test in terms of the cost function value and computation time.



(a) Comparison of cost function value.

(b) Comparison of computation time.

Fig. 4. Performance profiles of various methods for the benchmark test in terms of the cost function value and computation time.

[6] F. Facchinei and J.-S. Pang, *Finite-Dimensional Variational Inequalities and Complementarity Problems*. Springer, 2003.

[7] S. Gros and M. Diehl, "Numerical optimal control (draft)," 2020.

[8] L. Guo and J. Ye, "Necessary optimality conditions for optimal control problems with equilibrium constraints," *SIAM Journal on Control and Optimization*, vol. 54, no. 5, pp. 2710–2733, 2016.

[9] J. Hall, A. Nurkanović, F. Messerer, and M. Diehl, "A sequential convex programming approach to solving quadratic programs and optimal control problems with linear complementarity constraints," *IEEE Control Systems Letters*, vol. 6, pp. 536–541, 2021.

[10] W. Heemels, J. M. Schumacher, and S. Weiland, "Linear complementarity systems," *SIAM Journal on Applied Mathematics*, vol. 60, no. 4, pp. 1234–1269, 2000.

[11] T. Hoheisel, C. Kanzow, and A. Schwartz, "Theoretical and numerical comparison of relaxation methods for mathematical programs with complementarity constraints," *Mathematical Programming*, vol. 137, pp. 257–288, 2013.

[12] H. Jiang and D. Ralph, "Smooth SQP methods for mathematical programs with nonlinear complementarity constraints," *SIAM Journal on Optimization*, vol. 10, no. 3, pp. 779–808, 2000.

[13] S. Le Cleac'h, T. A. Howell, S. Yang, C.-Y. Lee, J. Zhang, A. Bishop, M. Schwager, and Z. Manchester, "Fast contact-implicit model predictive control," *IEEE Transactions on Robotics*, 2024.

[14] S. Leyffer, G. López-Calva, and J. Nocedal, "Interior methods for mathematical programs with complementarity constraints," *SIAM Journal on Optimization*, vol. 17, no. 1, pp. 52–77, 2006.

[15] G.-H. Lin and M. Fukushima, "A modified relaxation scheme for mathematical programs with complementarity constraints," *Annals of Operations Research*, vol. 133, no. 1, pp. 63–84, 2005.

[16] F. Messerer, K. Baumgärtner, and M. Diehl, "Survey of sequential convex programming and generalized gauss-newton methods," *ESAIM: Proceedings and Surveys*, vol. 71, pp. 64–88, 2021.

[17] J. Nocedal and S. Wright, *Numerical Optimization*. Springer, 2006.

[18] A. Nurkanović, A. Pozharskiy, and M. Diehl, "Solving mathematical programs with complementarity constraints arising in nonsmooth optimal control," *arXiv preprint arXiv:2312.11022*, 2023.

[19] J.-M. Peng, "Equivalence of variational inequality problems to unconstrained minimization," *Mathematical Programming*, vol. 78, no. 3, pp. 347–355, 1997.

[20] J.-M. Peng and Y. Yuan, "Unconstrained methods for generalized complementarity problems," *Journal of Computational Mathematics*, pp. 253–264, 1997.

[21] S. Scholtes, "Convergence properties of a regularization scheme for mathematical programs with complementarity constraints," *SIAM Journal on Optimization*, vol. 11, no. 4, pp. 918–936, 2001.

[22] A. Vieira, B. Brogliato, and C. Prieur. (2018) Quadratic optimal control of linear complementarity systems: First order necessary conditions and numerical analysis. [Online]. Available: https://inria.hal.science/hal-01690400

[23] A. Wächter and L. 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.

[24] N. Yamashita, K. Taji, and M. Fukushima, "Unconstrained optimization reformulations of variational inequality problems," *Journal of Optimization Theory and Applications*, vol. 92, pp. 439–456, 1997.