Cloud-Assisted Nonlinear Model Predictive Control for Finite-Duration Tasks

Nan Li, *Member, IEEE*, Kaixiang Zhang, Zhaojian Li, *Senior Member, IEEE*, Vaibhav Srivastava, *Senior Member, IEEE*, and Xiang Yin, *Member, IEEE*

***Abstract*—Cloud computing creates new possibilities for control applications by offering powerful computation and storage capabilities. In this article, we propose a novel cloud-assisted model predictive control (MPC) framework in which we systematically fuse a cloud MPC that leverages the computing power of the cloud to compute optimal control based on a high-fidelity nonlinear model (thus, more accurate) but is subject to communication delays with a local MPC that relies on simplified linear dynamics due to limited local computation capability (thus, less accurate) while has timely feedback. Unlike traditional cloud-based control that treats the cloud as a powerful, remote, and sole controller in a networked control system setting, the proposed framework aims at seamlessly integrating the two controllers for enhanced performance. In particular, we formalize the fusion problem for finite-duration tasks with explicit consideration for model mismatches and errors due to request-response communication delays. We analyze stability-type properties of the proposed cloud-assisted MPC framework and establish approaches to robustly handling constraints within this framework in spite of plant-model mismatch and disturbances. A fusion scheme is then developed to enhance control performance while satisfying stability-type conditions, the efficacy of which is demonstrated with multiple simulation examples, including an automotive control example to show its industrial application potentials.**

***Index Terms*—Cloud computing, control fusion, model predictive control (MPC).**

I. Introduction

CLOUD computing is a computing paradigm that has evolved significantly over the past few years. In general, cloud computing is the availability of computer system resources to provide on-demand computing power and data storage services to users [1]. The "infinite" computing capacity of the cloud has opened up new possibilities for industrial automation [2], [3] and control applications [4], [5], [6], [7], especially for optimization-based and learning-based control strategies, such as model predictive control (MPC), that are computation and/or data intensive [8], [9]. Meanwhile, moving onboard computations to the cloud has given rise to new problems related to communication delays [10], [11], connection and/or packet losses [12], [13], [14], as well as privacy and cybersecurity issues [15].

The majority of existing work on cloud-based control systems has focused on high-level abstraction and design of computation and control architectures [16], [17], [18], [19], with scarce attention paid to field-level approaches and solutions. "Field-level" refers to the level of physical devices and functions, and the main goal of this level is "to provide interactions between the cyberspace (cloud) and the physical world (real field devices)" [2]. In particular, comprehensive theoretical study on field-level cloud-enabled/assisted control strategies is largely missing.

One notable exception is the cloud-assisted MPC approach proposed in [12] and [13]. In this approach, a constrained linear-quadratic MPC problem is solved in the cloud to generate constraint-admissible control at each sample time instant. An unconstrained linear-quadratic regulator is used as a backup local controller to ensure uninterrupted stabilization of the plant when connectivity is lost or the cloud MPC control is not returned within a prescribed deadline. To address the same issue, in [14], a remote linear-quadratic MPC executes in the cloud at high frequency as the nominal controller, while the same MPC problem is solved at low frequency on the local device as a backup controller. The work of [12], [13], [14] has focused on practical strategies to handle connectivity loss and cloud service latency. However, control performance and constraint enforcement in the presence of plant-model mismatch as well as connectivity, latency, or feasibility issues are not addressed.

Note that networked control system (NCS) is a related but different concept from cloud-assisted control. An NCS is a spatially distributed system for which the communication between sensors, actuators, and controllers is supported by a (typically wireless) network [20], [21]. Although NCS and cloud-assisted control both use wireless networks for communication and data exchange, one distinguishing feature of cloud-assisted control is that cloud computing is used as the remote, sole controller in

Manuscript received 4 August 2022; accepted 26 October 2022. Date of publication 3 November 2022; date of current version 30 August 2023. This work was supported by National Science Foundation under Grant 2045436. Recommended by Associate Editor C. Mahulea. *(Nan Li and Kaixiang Zhang contributed equally to this work.) (Corresponding author: Zhaojian Li.)*

Nan Li is with the Department of Aerospace Engineering, Auburn University, Auburn, AL 36849 USA (e-mail: nanli@auburn.edu).

Kaixiang Zhang and Zhaojian Li are with the Department of Mechanical Engineering, Michigan State University, East Lansing, MI 48824 USA (e-mail: zhangk64@msu.edu; lizhaoj1@egr.msu.edu).

Vaibhav Srivastava is with the Department of Electrical and Computer Engineering, Michigan State University, East Lansing, MI 48824 USA (e-mail: vaibhav@egr.msu.edu).

Xiang Yin is with the Department of Automation and Key Laboratory of System Control and Information Processing, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: yinxiang@sjtu.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TAC.2022.3219293.

Digital Object Identifier 10.1109/TAC.2022.3219293

NCS while cloud-assisted control aims at integrating both cloud and local computations for enhanced performance.

In this article, we propose a novel cloud-assisted nonlinear MPC approach for finite-duration control tasks. Although nonlinear MPC can explicitly handle system constraints for complex nonlinear systems, its implementations require substantial computation power to address nonconvex optimization problems, which hinders the deployment of nonlinear MPC in many cyber-physical systems with limited onboard resource. Considering this issue, in the proposed approach the cloud is introduced to execute the nonlinear MPC that relies on a high-fidelity model. This cloud-based nonlinear MPC is then systematically combined with a computationally lighter linear MPC that is executed on the local device to mitigate issues related to communication delays, disturbances, and plant-model mismatch and thereby achieve enhanced control performance. This article focuses on finite-duration control tasks. Note that finite-duration control tasks are frequently encountered in the automotive and aerospace fields (e.g., autonomous vehicle lane change [22], spacecraft orbital transfer and landing [23], [24], etc.), indicating that the proposed approach has broad application potentials. Compared to previous work, our approach is unique by the following features: First, different from the approaches of [19], [25] that solely rely on cloud-computed control, we investigate a systematic fusion of cloud-computed control and locally-computed control, where the cloud is used as a value-added service to improve control performance when connectivity is available. Second, in our approach, the system requests a service from the cloud only at the initial time of the control task, instead of requesting cloud service at every sample time instant over the control task duration as in the approaches considered in [12], [13], [14]. This strategy is motivated by the widely adopted pay-per-use model of cloud computing services [1], i.e., the user pays for every cloud request and thereby it is desirable to minimize cloud request times. This strategy also implies that our approach does not rely on a low-latency connection to the cloud over the entire control task duration, i.e., our approach requires a lower connection quality and thereby has a wider range of applications. Note that this strategy does not imply our control approach is open-loop. Instead, the local MPC provides feedback by recomputing control based on new measurement of system state at every sample time instant, and therefore, after cloud-local MPC fusion, the resulting control has feedback and is closed-loop. Third, our approach considers a general nonlinear system, with no restrictions to linear models and quadratic cost functions as considered in [12], [13], [14]. Furthermore, our approach explicitly addresses model prediction errors due to plant-model mismatch and disturbances, including constraint enforcement in the presence of such prediction errors, which is not addressed in [12], [13], [14].

Note that the cloud-assisted MPC framework proposed in this article is also different from the classic two-layer integration of real-time optimization (RTO) and MPC in the process industries. First, the proposed scheme aims at integrating two MPC-based controllers, one executed in the cloud and the other executed on the local device, to leverage one's strengths to offset the other's weaknesses and hence achieve enhanced performance

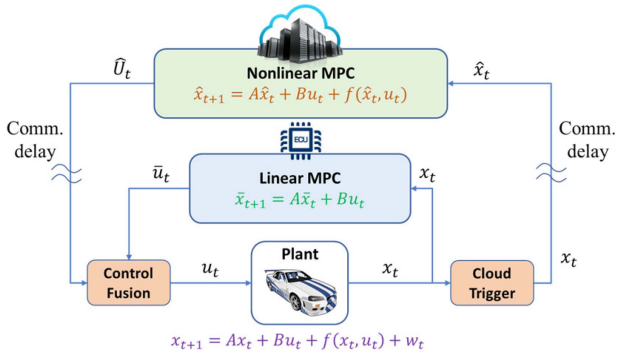


Fig. 1. Illustration of the architecture of the proposed cloud-assisted control framework based on MPC.

after the integration. In this scheme, the two MPCs are in the same layer and compensate each other. In typical integration of RTO and MPC, RTO is in the upper layer and determines the targets/set-points for the MPC in the lower layer. Second, the cloud and local MPCs in the proposed scheme both use dynamic models for prediction and control optimization. In typical integration of RTO and MPC, RTO makes higher-level decisions based on a steady-state model and MPC takes care of dynamic control. Furthermore, the control signals generated by the cloud and local MPCs in the proposed scheme have the same time scale. Specifically, each element of the cloud/local MPC generated control sequence represents a control signal over one sampling period. In typical applications of RTO + MPC in the process industries, RTO and MPC execute on different time scales: RTO usually executes on a hour time scale and MPC on a minute time scale [26]. We refer the readers to [26] for a comprehensive review of RTO.

With the aforementioned distinguishing features, the contributions of this article include.

1) We propose a unified framework for cloud-assisted MPC design (see Fig. 1), with a focus on finite-duration control tasks. The proposed novel framework systematically integrates cloud and local controls to achieve improved performance.
2) We rigorously analyze the feasibility and robust constraint satisfaction for the considered paradigm in the presence of disturbances and cloud/local model prediction errors.
3) We develop a switching-based fusion policy to systematically combine cloud-computed control trajectory with local shrinking-horizon MPC solutions to minimize the worst-case cost-to-go for enhanced performance.
4) We verify our theoretical results and demonstrate the effectiveness of our cloud-assisted MPC approach in terms of improving control performance with multiple simulation examples, including an automotive control example to illustrate potential practical application of our approach.

The rest of this article is organized as follows. Section II introduces the types of systems and control tasks to be treated as well as the models used by the cloud and local devices to compute controls. We describe the cloud MPC and local MPC

designs and analyze their properties in Section III. We then develop a switching-based fusion scheme to systematically combine cloud and local MPC controls for enhanced performance in Section IV. We use multiple simulation examples to verify the theoretical results and demonstrate the effectiveness in terms of improving overall control performance of our cloud-assisted MPC approach in Section V. Finally, Section VI concludes this article.

The notations used in this article are standard. In particular, we use $\|\cdot\|$ to denote an arbitrary vector norm and its induced matrix norm, and use $\|\cdot\|_Q$ with a positive-definite symmetric matrix Q to denote a quadratic norm, i.e., $\|\cdot\|_Q = \sqrt{(\cdot)^\top Q(\cdot)}$. The symbols $\oplus$ and $\sim$ denote, respectively, the Minkowski sum operation and the Minkowski/Pontryagin difference operation between sets [27].

II. System and Models

In this article, we investigate a new cloud-assisted control paradigm that is illustrated in Fig. 1. Unlike conventional networked control systems that exploit only one (remote) controller, the considered paradigm seamlessly integrates two complementary computing platforms (cloud and onboard computation units). Specifically, the MPC optimizations are performed both on the cloud and on the local processor (e.g., automotive electronic control unit). On the cloud side, the cloud can support an MPC with a high-fidelity model but is subject to communication delays. On the local agent side, the onboard controller runs an MPC with a simplified model due to limited onboard computations but with negligible time delays. In this work, the two controllers are systematically designed and integrated for enhanced performance.

More specifically, we consider a plant that can be represented by the following discrete-time state-space model:

$$x_{t+1} = Ax_t + Bu_t + f(x_t, u_t) + w_t \tag{1}$$

where $x_t \in \mathbb{R}^n$ denotes the state of the system at the discrete time instant $t \in \mathbb{N}_0$, $u_t \in \mathbb{R}^m$ denotes the control input at t, A and B are matrices of appropriate dimensions, $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is a nonlinear function, and $w_t \in \mathbb{R}^p$ represents unmeasured disturbances acting on the system.

We make the following assumptions about system (1):

A1: f is a Lipschitz continuous function such that $\|f(x,u) - f(x,'u')\| \le L_f\|x - x'\| + M_f\|u - u'\|$ for all $x, x' \in \mathbb{R}^n$ and $u, u' \in \mathbb{R}^m$, where $L_f, M_f \ge 0$ are known Lipschitz constants of f; and $f(0,0) = 0$.

Note that the second condition of **A1**, $f(0,0) = 0$, is equivalent to saying that $(x,u) = (0,0)$ is a nominal steady-state pair and the matrices A, B correspond to a linear model of the system around the steady-state pair $(x,u) = (0,0)$.

A2: The disturbance input w_t takes values in a known bounded set $W \subset \mathbb{R}^p$ and $\sup_{w\in W} \|w\| = \omega$.

The control objective is to minimize the following cost function

$$J = \sum_{t=0}^{N-1} \phi(x_t, u_t) + \psi(x_N) \tag{2}$$

where N is the optimization horizon that corresponds to the end time of a finite-duration control task (i.e., $t = 0, 1, \ldots, N$ corresponds to the control task duration). The following assumption is made about the cost function (2):

A3: $\phi(\cdot,\cdot)$ and $\psi(\cdot)$ are both Lipschitz continuous in x, i.e., satisfy $|\phi(x,u) - \phi(x,'u)| \le L_\phi\|x - x'\|$ and $|\psi(x) - \psi(x')| \le L_\psi\|x - x'\|$ for all $x, x' \in \mathbb{R}^n$, where $L_\phi, L_\psi \ge 0$ are known Lipschitz constants of ϕ and ψ, respectively.

Being globally Lipschitz is in general a strong assumption on the functions ϕ and ψ. For example, the commonly used quadratic cost function is not globally Lipschitz. However, if the system state is known to stay within or needs to be constrained within a bounded domain during the system operation, then the global Lipschitz conditions in **A3** can be replaced with local Lipschitz conditions for later developments, which are easier to satisfy. For instance, every continuously differentiable function is Lipschitz continuous on every compact set [28].

Meanwhile, constraints that represent hard operational specifications are ubiquitous. Two types of constraints are most common: 1) pointwise-in-time constraints, i.e., constraints that must be satisfied for all time $t = 0, \ldots, N$; 2) terminal constraints, i.e., constraints that must be satisfied at the terminal time $t = N$. For instance, pointwise-in-time constraints are frequently used to represent safety-related requirements, such as variable bounds in chemical processes and collision avoidance in autonomous driving, while terminal constraints are frequently used to represent terminal control objectives in the case of finite-duration tasks and used for stability reasons. In this article, we consider state constraints taking the following form:

$$x_T \in \mathcal{X}_T = \left\{x \in \mathbb{R}^n : G_{T,j}^\top x \le g_{T,j},\ j = 1, \ldots, p_T\right\} \tag{3}$$

where T denotes the time instant where the constraints are imposed and it takes values in $1, 2, \ldots, N$, $G_{T,j} \in \mathbb{R}^n$, $g_{T,j} \in \mathbb{R}$, and p_T is the total number of linear inequalities defining the constraint set X_T. Note that in what follows we treat the case where state constraints are imposed at a single time instant T to simplify the exposition. The case where state constraints are imposed at multiple time instants can be treated similarly (for instance, if (3) needs to be satisfied at $T = T_1$ and $T = T_2$, then in the cloud MPC problem the tightened constraint (17) will be imposed for both $T = T_1$ and $T = T_2$). In this case, since T can take any values in $1, 2, \ldots, N$, (3) can represent both pointwise-in-time and terminal state constraints.

As shown in Fig. 1, the cloud MPC exploits the following higher-fidelity model to compute control for the system:

$$\Sigma^c :\ \hat{x}_{t+1} = A\hat{x}_t + Bu_t + f(\hat{x}_t, u_t). \tag{4}$$

Here, we use the "hat" notation to represent predicted states in the cloud MPC. This model includes the nonlinear term $f(\hat{x}_t, u_t)$ and necessarily leads to a nonlinear and nonconvex optimization problem, which can be computationally intensive. Nevertheless, due to access to powerful cloud computations, we assume this can be solved efficiently on the cloud.

On the other side, due to limited onboard resources, the following lower-fidelity model is used by the local MPC to

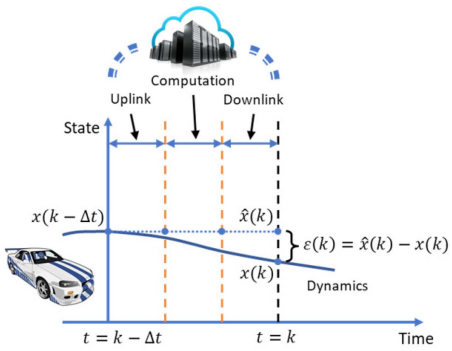


Fig. 2. Illustration of cloud state propagation error due to request-response delays.

compute control for the system:

$$\Sigma^l: \ \bar{x}_{t+1} = A\bar{x}_t + Bu_t \tag{5}$$

which is a linear model and thereby yields easier optimization problems that can be handled by onboard computing devices. We use the "bar" notation to represent predicted states in the local MPC.

Our cloud-assisted nonlinear MPC approach is featured by a cloud MPC problem solved at the initial time of the control task, a sequence of shrinking-horizon local MPC problems solved at each sample time instant over the control task duration, and a fusion strategy to systematically combine cloud and local MPC solutions. These components are introduced and analyzed sequentially in the following two sections.

III. Cloud and Local MPC Designs

In this section, we introduce the cloud MPC and the local MPC designs and analyze their properties.

A. Cloud MPC Design

When a control task is assigned, we assume that the local system can request the cloud to solve a nonlinear MPC problem based on the higher-fidelity model (4). Note that this process incurs a "request-response delay," i.e., there will be a time difference between the local system submitting its computation request to the cloud and receiving the computation results from the cloud, as illustrated in Fig. 2. In the sequel, we refer to such delays as communication delays. While there exist different ways to handle the incurred delays, in this article, we adopt a "prediction-ahead-of-time" approach. Specifically, let Δt denote the maximum delay due to cloud communication. Upon cloud MPC request at time instant $t = k - \Delta t$, the cloud will predict the state Δt time ahead, $\hat{x}((k-\Delta t)+\Delta t) = \hat{x}(k)$, by assuming state remaining constant over the time interval Δt or by running forward simulation based on system model (4). The predicted state $\hat{x}(k)$ is then used as the initial condition for the cloud MPC and the obtained optimal control trajectory is downloaded to the local system for use starting at time instant $t = (k-\Delta t)+\Delta t = k$ to avoid any outdated control inputs. This delay treatment strategy essentially converts the effect of communication delays to initial state uncertainty, which is illustrated in Fig. 2. We note that there exist reliable methods for estimating the request-response delay Δt in a given cloud service scenario, for instance, estimating the uplink + downlink delay through a "ping" test and estimating the computation delay according to the cloud available computing power and the problem complexity [29].

Without loss of generality, we consider that at time instant $t = -\Delta t$, the local system requests the cloud to perform a nonlinear MPC based on the cost function (2), the model (4), and an estimate of state at $t = 0$, $\hat{x}_0$. For now, we assume generic constraints on the predicted states $\hat{x}_\tau$ and controls u_τ. In particular, the cloud is requested to solve the following optimization problem:

$$\min \quad J = \sum_{\tau=0}^{N-1} \phi(\hat{x}_\tau, u_\tau) + \psi(\hat{x}_N) \tag{6a}$$

$$\text{s.t.} \quad \hat{x}_{t+1} = A\hat{x}_t + Bu_t + f(\hat{x}_t, u_t) \tag{6b}$$

$$\left(\{\hat{x}_\tau\}_{\tau=1}^{N}, \{u_\tau\}_{\tau=0}^{N-1}\right) \in \Xi^c \tag{6c}$$

with the initial condition $\hat{x}_0$. Note that (6c) can represent any constraints on $\hat{x}_\tau$ and u_τ. We will elaborate (6c) later on to enforce specific state constraints in the form of (3) in a robust manner (i.e., ensuring satisfaction of (3) by the actual system in spite of model prediction errors).

By solving (6), the cloud computes and transmits to the local system an optimal control trajectory available starting from time 0, $\{\hat{u}_0, \hat{u}_1, \ldots, \hat{u}_{N-1}\}$, an associated prediction of state trajectory, $\{\hat{x}_0, \hat{x}_1, \ldots, \hat{x}_N\}$, and a corresponding sequence of cost-to-go, $\{\hat{J}_0, \hat{J}_1, \ldots, \hat{J}_N\}$, where

$$\hat{J}_k = \sum_{\tau=k}^{N-1} \phi(\hat{x}_\tau, \hat{u}_\tau) + \psi(\hat{x}_N). \tag{7}$$

Note that if the controls $\{\hat{u}_0, \hat{u}_1, \ldots, \hat{u}_{N-1}\}$ are applied to the actual system (1), the resulting actual state trajectory $\{x_0, x_1, \ldots, x_N\}$ will be different from the predicted trajectory $\{\hat{x}_0, \hat{x}_1, \ldots, \hat{x}_N\}$, due to the disturbances w_t acting on the actual system (1) and errors between $\hat{x}_0$ and x_0 caused by imperfect prediction (shown in Fig. 2). Correspondingly, the predicted cost-to-go $\hat{J}_k$, for $k = 0, \ldots, N$, will have some errors from the actual cumulative cost over the steps $\tau = k, \ldots, N$, which is defined as follows:

$$J_k^c = \sum_{\tau=k}^{N-1} \phi(x_\tau, \hat{u}_\tau) + \psi(x_N) \tag{8}$$

where x_τ denotes the actual state at time τ under the control input sequence $\{\hat{u}_0, \hat{u}_1, \ldots, \hat{u}_{\tau-1}\}$.

The following proposition establishes bounds on the errors between $\hat{x}_\tau$ and x_τ and between $\hat{J}_k$ and J_k^c, which will be exploited later to develop an approach for robust constraint enhancement and to design our cloud-local MPC fusion scheme.

Proposition 1: Suppose at time k, $k \in \{0, 1, \ldots, N-1\}$, the difference between the predicted state $\hat{x}_k$ and the actual state x_k is $\varepsilon_k = \hat{x}_k - x_k$ and the control sequence $\{\hat{u}_k, \ldots, \hat{u}_{N-1}\}$ is applied to the actual system (1) over all future steps $\tau = k$,

$\ldots, N-1$. Then, the error between the predicted state $\hat{x}_\tau$ and the actual state x_τ, for $\tau = k+1, \ldots, N$, can be bounded as

$$\|\hat{x}_\tau - x_\tau\| \leq (a+L_f)^{\tau-k}\epsilon_k + \sum_{l=k}^{\tau-1}(a+L_f)^{\tau-l-1}\omega \tag{9}$$

where $a = \|A\|$, $\epsilon_k = \|\varepsilon_k\|$, and $\omega = \sup_{w\in W}\|w\|$ (see **A2**). Meanwhile, the error between the predicted cost-to-go $\hat{J}_k$ and the actual cumulative cost J_k^c can be bounded as

$$\begin{aligned}&|\hat{J}_k - J_k^c| \\ &\leq \frac{M_k\left((a+L_f)^{N-k}-1\right)-(N-k)L_\phi\omega}{(a+L_f)-1} + L_\psi\epsilon_k(a+L_f)^{N-k}\end{aligned} \tag{10}$$

where

$$M_k = L_\phi\epsilon_k + L_\psi w + \frac{L_\phi w}{(a+L_f)-1}. \tag{11}$$

Proof: Mathematical induction is exploited to prove that (9) holds for $\tau = k+1, \ldots, N$. Firstly, using (1) and (4) we obtain

$$\begin{aligned}&\|\hat{x}_{k+1} - x_{k+1}\| \\ &= \|A\hat{x}_k + B\hat{u}_k + f(\hat{x}_k,\hat{u}_k) - Ax_k - B\hat{u}_k - f(x_k,\hat{u}_k) - w_k\| \\ &\leq \|A(\hat{x}_k - x_k)\| + \|f(\hat{x}_k,\hat{u}_k) - f(x_k,\hat{u}_k)\| + \|w_k\| \\ &\leq (\|A\| + L_f)\|\hat{x}_k - x_k\| + \|w_k\| \\ &\leq (a+L_f)\epsilon_k + \omega\end{aligned} \tag{12}$$

where we have used the definition of matrix norm and the Lipschitz continuity assumption of f (i.e., **A1**) to derive the inequality in the fourth line. This proves (9) for $\tau = k+1$. We now assume (9) holds for $\tau = \sigma$, i.e., $\|\hat{x}_\sigma - x_\sigma\| \leq (a+L_f)^{\sigma-k}\epsilon_k + \sum_{l=k}^{\sigma-1}(a+L_f)^{\sigma-l-1}\omega$. In this case, following (12) we can obtain:

$$\begin{aligned}&\|\hat{x}_{\sigma+1} - x_{\sigma+1}\| \leq (a+L_f)\|\hat{x}_\sigma - x_\sigma\| + \omega \\ &\leq (a+L_f)\left((a+L_f)^{\sigma-k}\epsilon_k + \sum_{l=k}^{\sigma-1}(a+L_f)^{\sigma-l-1}\omega\right) + \omega \\ &= (a+L_f)^{\sigma+1-k}\epsilon_k + \sum_{l=k}^{\sigma-1}(a+L_f)^{\sigma-l}\omega + \omega \\ &= (a+L_f)^{\sigma+1-k}\epsilon_k + \sum_{l=k}^{\sigma}(a+L_f)^{\sigma-l}\omega\end{aligned} \tag{13}$$

where we have used the upper bound of $\|\hat{x}_\sigma - x_\sigma\|$ (i.e., the inequality (9) for $\tau = \sigma+1$) to derive the second line. This proves (9) for $\tau = \sigma+1$. Combining the base case in (12) and the induction step in (13), (9) is proved for all $\tau = k+1, \ldots, N$ by induction.

We now estimate the error between $\hat{J}_k$ and J_k^c

$$\begin{aligned}&|\hat{J}_k - J_k^c| \leq \sum_{\tau=k}^{N-1}|\phi(\hat{x}_\tau,\hat{u}_\tau) - \phi(x_\tau,\hat{u}_\tau)| + |\psi(\hat{x}_N) - \psi(x_N)| \\ &\leq \sum_{\tau=k}^{N-1} L_\phi\|\hat{x}_\tau - x_\tau\| + L_\psi\|\hat{x}_N - x_N\| \\ &\leq \sum_{\tau=k}^{N-1} L_\phi\left((a+L_f)^{\tau-k}\epsilon_k + \sum_{l=k}^{\tau-1}(a+L_f)^{\tau-l-1}\omega\right) \\ &\quad + L_\psi\left((a+L_f)^{N-k}\epsilon_k + \sum_{l=k}^{N-1}(a+L_f)^{N-l-1}\omega\right) \\ &= L_\phi\epsilon_k\frac{(a+L_f)^{N-k}-1}{(a+L_f)-1} + L_\phi\omega\sum_{\tau=k}^{N-1}\frac{(a+L_f)^{\tau-k}-1}{(a+L_f)-1} \\ &\quad + L_\psi\epsilon_k(a+L_f)^{N-k} + L_\psi\omega\frac{(a+L_f)^{N-k}-1}{(a+L_f)-1} \\ &= \left(L_\phi\epsilon_k + L_\psi\omega + \frac{L_\phi\omega}{(a+L_f)-1}\right)\frac{(a+L_f)^{N-k}-1}{(a+L_f)-1} \\ &\quad - \frac{(N-k)L_\phi\omega}{(a+L_f)-1} + L_\psi\epsilon_k(a+L_f)^{N-k}.\end{aligned} \tag{14}$$

Therefore, we have shown the bound in (10). ■

Note that the above results hold true for generic constraints on the predicted states $\hat{x}_\tau$ and controls u_τ imposed in the cloud MPC optimization problem (6). We now consider specific state constraints in the form of (3). Note that imposing the constraint (3) directly in the cloud MPC problem (6) does not guarantee (3) will be satisfied by the actual state, due to prediction errors discussed above. Therefore, in what follows we develop an approach to robustly enforcing (3).

We assume that at the initial time $t = 0$, the error between the state estimate $\hat{x}_0$ (i.e., the initial condition of the cloud MPC problem (6)) and the actual state x_0 can be bounded as

$$\|\hat{x}_0 - x_0\| \leq \delta_0 \tag{15}$$

where δ_0 is a known constant. Note that when the system state is assumed to be locally measured, the error between $\hat{x}_0$ and x_0 is mainly caused by the delay due to cloud communication, which includes uplink, downlink, and computational delays, as illustrated in Fig. 2. In this case, it is possible for the cloud to estimate a bound for $\|\hat{x}_0 - x_0\|$ according to maximum delay length (which is estimable using the methods described in the first paragraph of Section III-A) plus system dynamics and disturbance characteristics (such as Lipschitz constants of the dynamic equation and disturbance bounds in an approach similar to that in Proposition 1).

Under the assumption (15), we define

$$\delta_k = (a+L_f)^k\delta_0 + \sum_{l=0}^{k-1}(a+L_f)^{k-l-1}\omega \tag{16}$$

for $k = 1, 2, \ldots, T$, and impose the following constraint in the cloud MPC problem as (6c):

$$\hat{x}_T \in \mathcal{X}_T \sim \mathcal{B}_{\delta_T} \tag{17}$$

where $\sim$ denotes the Minkowski/Pontryagin difference operation [27], and $\mathcal{B}_{\delta_T} = \{x \in \mathbb{R}^n : \|x\| \leq \delta_T\}$.

By tightening the constraint according to (17), we can guarantee the original constraint (3) to be satisfied by the actual state x_T. This result is formalized in Lemma 1 in Section IV. Furthermore, suppose the support function for unit ball $\mathcal{B} = \{x \in \mathbb{R}^n : \|x\| \leq 1\}$, $h_{\mathcal{B}} : \mathbb{R}^n \to \mathbb{R}$, is available, the constraint (17) can be expressed as

$$G_{T,j}^\top \hat{x}_T \leq g_{T,j} - \delta_T h_{\mathcal{B}}(G_{T,j}), \quad j = 1, \ldots, p_T \tag{18}$$

which are linear inequality constraints on the predicted state $\hat{x}_T$.

We note that constraint tightening is a common approach to realizing robust constraint enforcement in the MPC literature [30], [31], [32]. Although for linear systems it is possible to accurately estimate the worst-case effect of a bounded disturbance via direct propagation of the disturbance set W through the dynamic equation [30], for nonlinear systems this is typically not possible and therefore it is common to leverage function characteristics such as Lipschitz constants to overbound the disturbance effect and tighten the constraints accordingly [31], [32], which is also pursued in our approach [see (16) and (17)]. From an MPC practical implementation perspective, various strategies are available to reduce disturbance effect and the required amount of constraint tightening. As one strategy, one can first stabilize the system using a local controller[1] and treat the stabilized system as the plant (1) where u_t represents an adjustment to the nominal control signal that is determined by MPC to achieve optimal control and constraint enforcement [33]. When designing the local controller, one can enforce $a + L_f < 1$. In this case, the error bound δ_k in (16) grows slowly and is upper bounded by $\delta_0 + \frac{\omega}{1-a-L_f}$ (i.e., $\delta_k < \delta_0 + \frac{\omega}{1-a-L_f}$ for all k). Such a strategy can effectively reduce the constraint tightening in (17) and avoid infeasibility of the cloud MPC problem.

B. Local MPC Design

At each time instant $t = 0, 1 \ldots, N-1$, the plant computes locally an optimal control based on the cost function (2), the model (5), and a measurement of current state x_t. As before, we start with considering the case with generic constraints on the predicted states $\bar{x}_{\tau|t}$ and controls $u_{\tau|t}$. In this case, the following shrinking-horizon local MPC problem is solved at each t to compute control:

$$\min \quad J_t = \sum_{\tau=t}^{N-1} \phi(\bar{x}_{\tau|t}, u_{\tau|t}) + \psi(\bar{x}_{N|t}) \tag{19a}$$

$$\text{s.t.} \quad \bar{x}_{\tau+1|t} = A\bar{x}_{\tau|t} + Bu_{\tau|t} \tag{19b}$$

$$\bar{x}_{t|t} = x_t \tag{19c}$$

$$\left(\{\bar{x}_{\tau|t}\}_{\tau=1}^{N}, \{u_{\tau|t}\}_{\tau=0}^{N-1}\right) \in \Xi_t^l. \tag{19d}$$

In the above expressions, we use $(\cdot)_{\tau|t}$ to denote a predicted value of the variable $(\cdot)_\tau$ with the prediction made at the time instant t.[2] The constraint (19d) can represent any constraints on $\bar{x}_{\tau|t}$ and $u_{\tau|t}$ and will be elaborated to robustly enforce specific state constraints in the form of (3) later on. By solving (19), an optimal control trajectory, $\{\bar{u}_{t|t}, \bar{u}_{t+1|t}, \ldots, \bar{u}_{N-1|t}\}$, an associated prediction of state trajectory, $\{\bar{x}_{t|t}, \bar{x}_{t+1|t}, \ldots, \bar{x}_{N|t}\}$, and a corresponding cost value, $\bar{J}_t$, are obtained.

Note that similar to the case of cloud MPC, if the control sequence $\{\bar{u}_{t|t}, \bar{u}_{t+1|t}, \ldots, \bar{u}_{N-1|t}\}$ is applied to the actual system (1), the resulting actual state trajectory $\{x_t, x_{t+1}, \ldots, x_N\}$ will be different from the predicted trajectory $\{\bar{x}_{t|t}, \bar{x}_{t+1|t}, \ldots, \bar{x}_{N|t}\}$, due to plant-model mismatch and the disturbances w_τ acting on the actual system (1). Correspondingly, the predicted cost value $\bar{J}_t$ will have some error from the actual cumulative cost over the steps $\tau = t, \ldots, N$, which is defined as follows:

$$J_t^l = \sum_{\tau=t}^{N-1} \phi(x_\tau, \bar{u}_{\tau|t}) + \psi(x_N) \tag{20}$$

where x_τ denotes the actual state at time τ under the control input sequence $\{\bar{u}_{t|t}, \bar{u}_{t+1|t}, \ldots, \bar{u}_{\tau-1|t}\}$.

The following proposition establishes bounds on the errors between $\bar{x}_{\tau|t}$ and x_τ and between $\bar{J}_t$ and J_t^l.

Proposition 2: Suppose the local MPC control sequence $\{\bar{u}_{t|t}, \ldots, \bar{u}_{N-1|t}\}$ is applied to the actual system (1) over the steps $\tau = t, \ldots, N-1$. Then, the error between the predicted state $\bar{x}_{\tau|t}$ and the actual state x_τ, for $\tau = t+1, \ldots, N$, can be bounded as

$$\|\bar{x}_{\tau|t} - x_\tau\| \leq \sum_{l=t}^{\tau-1} (a + L_f)^{\tau-l-1} \left(L_f \|\bar{x}_{l|t}\| + M_f \|\bar{u}_{l|t}\| + \omega\right) \tag{21}$$

where $a = \|A\|$ and $\omega = \sup_{w \in W} \|w\|$ (see **A2**). Meanwhile, the error between the predicted cost value $\bar{J}_t$ and the actual cumulative cost J_t^l can be bounded as

$$\begin{aligned} &|\bar{J}_t - J_t^l| \\ &\leq L_\phi \sum_{\tau=t+1}^{N-1} \left(\sum_{l=t}^{\tau-1} (a + L_f)^{\tau-l-1} \left(L_f \|\bar{x}_{l|t}\| + M_f \|\bar{u}_{l|t}\| + \omega\right) \right) \\ &+ L_\psi \left(\sum_{l=t}^{N-1} (a + L_f)^{\tau-l-1} \left(L_f \|\bar{x}_{l|t}\| + M_f \|\bar{u}_{l|t}\| + \omega\right) \right). \end{aligned} \tag{22}$$

Proof: Mathematical induction is exploited to prove that (21) holds for $\tau = t+1, \ldots, N$. Firstly, using (1) and (5) we obtain

$$\begin{aligned} &\|\bar{x}_{t+1|t} - x_{t+1}\| \\ &= \|A\bar{x}_{t|t} + B\bar{u}_{t|t} - Ax_t - B\bar{u}_{t|t} - f(x_t, \bar{u}_{t|t}) - w_t\| \\ &\leq \|f(x_t, \bar{u}_{t|t}) - f(0,0)\| + \|w_t\| \\ &\leq L_f \|x_t\| + M_f \|\bar{u}_{t|t}\| + \omega \\ &= L_f \|\bar{x}_{t|t}\| + M_f \|\bar{u}_{t|t}\| + \omega \end{aligned} \tag{23}$$

where we have used $\bar{x}_{t|t} = x_t$ and $f(0,0) = 0$ (see **A1**) to obtain the third line, used the Lipschitz continuity assumption of f (i.e., **A1**) to obtain the fourth line, and used $\bar{x}_{t|t} = x_t$ again to obtain the last line. This proves (21) for $\tau = t+1$. We now assume (21) holds for $\tau = \sigma$, i.e., $\|\bar{x}_{\sigma|t} - x_\sigma\| \leq \sum_{l=t}^{\sigma-1} (a +$

[1]This controller is not necessarily an optimal controller and therefore does not need to be optimization-based.

[2]We did not use such notations for cloud MPC variables because the cloud MPC problem is solved only once at the initial time $t = 0$.

$L_f)^{\sigma-l-1}(L_f\|\bar{x}_{l|t}\|+M_f\|\bar{u}_{l|t}\|+\omega)$. In this case, following (23) we can obtain:

$$\begin{aligned}
&\|\bar{x}_{\sigma+1|t}-x_{\sigma+1}\| \\
&=\|A\bar{x}_{\sigma|t}+B\bar{u}_{\sigma|t}-Ax_\sigma-B\bar{u}_{\sigma|t}-f(x_\sigma,\bar{u}_{\sigma|t})-w_\sigma\| \\
&\le\|A(\bar{x}_{\sigma|t}-x_\sigma)\|+\|f(x_\sigma,\bar{u}_{\sigma|t})-f(0,0)\|+\|w_\sigma\| \\
&\le a\|\bar{x}_{\sigma|t}-x_\sigma\|+L_f\|x_\sigma-\bar{x}_{\sigma|t}+\bar{x}_{\sigma|t}\|+M_f\|\bar{u}_{\sigma|t})\|+\omega \\
&\le(a+L_f)\|\bar{x}_{\sigma|t}-x_\sigma\|+L_f\|\bar{x}_{\sigma|t}\|+M_f\|\bar{u}_{\sigma|t})\|+\omega \\
&\le(a+L_f)\left(\sum_{l=t}^{\sigma-1}(a+L_f)^{\sigma-l-1}\left(L_f\|\bar{x}_{l|t}\|+M_f\|\bar{u}_{l|t}\|+\omega\right)\right) \\
&\quad+L_f\|\bar{x}_{\sigma|t}\|+M_f\|\bar{u}_{\sigma|t})\|+\omega \\
&=\sum_{l=t}^{\sigma}(a+L_f)^{\sigma-l-1}\left(L_f\|\bar{x}_{l|t}\|+M_f\|\bar{u}_{l|t}\|+\omega\right) \qquad (24)
\end{aligned}$$

where we have used the upper bound of $\|\bar{x}_{\sigma|t}-x_\sigma\|$ (i.e., the inequality (21) for $\tau=\sigma$) to derive the fifth line. This proves (21) for $\tau=\sigma+1$. Combining the base case in (23) and the induction step in (24), (21) is proved for all $\tau=t+1,\dots,N$ by induction.

Using the Lipschitz continuity assumptions of ϕ and ψ (i.e., **A3**), we can bound the difference between $\bar{J}_t$ and J_t^l according to

$$\begin{aligned}
&|\bar{J}_t-J_t^l| \\
&=\left|\sum_{\tau=t}^{N-1}\left(\phi(\bar{x}_{\tau|t},\bar{u}_{\tau|t})-\phi(x_\tau,\bar{u}_{\tau|t})\right)+\left(\psi(\bar{x}_{N|t})-\psi(x_N)\right)\right| \\
&\le\sum_{\tau=t}^{N-1}\left|\phi(\bar{x}_{\tau|t},\bar{u}_{\tau|t})-\phi(x_\tau,\bar{u}_{\tau|t})\right|+\left|\psi(\bar{x}_{N|t})-\psi(x_N)\right| \\
&\le L_\phi\sum_{\tau=t}^{N-1}\|\bar{x}_{\tau|t}-x_\tau\|+L_\psi\|\bar{x}_{N|t}-x_N\| \\
&=L_\phi\sum_{\tau=t+1}^{N-1}\|\bar{x}_{\tau|t}-x_\tau\|+L_\psi\|\bar{x}_{N|t}-x_N\| \qquad (25)
\end{aligned}$$

where we have used $\bar{x}_{t|t}=x_t$ to drop the term for $\tau=t$ in the sum and obtain the last line. Then, using the bounds for $\|\bar{x}_{\tau|t}-x_\tau\|$ in (21) for $\tau=t+1,\dots,N$, we can obtain the bound for $|\bar{J}_t-J_t^l|$ in (22). ■

Remark 1: The bounds in (21) and (22) depend on the predicted states $\bar{x}_{l|t}$ and controls $\bar{u}_{l|t}$. Note that once the local MPC problem (19) is solved, $\bar{x}_{l|t}$ and $\bar{u}_{l|t}$, for $l=t,\dots,N$, are available. This means the bounds in (21) and (22) are available once (19) is solved.

Remark 2: Proposition 2 represents a strategy for estimating the actual cost J_t^l corresponding to the control sequence $\{\bar{u}_{t|t},\bar{u}_{t+1|t},\dots,\bar{u}_{N-1|t}\}$ using a cost value predicted by the lower-fidelity linear model (5), $\bar{J}_t$, which is obtained along with the control sequence $\{\bar{u}_{t|t},\bar{u}_{t+1|t},\dots,\bar{u}_{N-1|t}\}$ when the local MPC problem (19) is solved. An alternative strategy is to compute another estimate of J_t^l via the higher-fidelity model (4) by applying the local MPC obtained control sequence $\{\bar{u}_{t|t},\bar{u}_{t+1|t},\dots,\bar{u}_{N-1|t}\}$ to (4). Note that this alternative strategy involves an additional step of simulating the higher-fidelity model (4), which may take nonnegligible effort of local onboard computation units especially for high-dimensional models. Further investigation of this alternative strategy is left to our future work.

Note that the above results hold true for generic constraints on the predicted states $\bar{x}_{\tau|t}$ and controls $u_{\tau|t}$ imposed in the local MPC optimization problem (19). We now elaborate (19d) to address specific state constraints in the form of (3). Similarly as in cloud MPC, in order to guarantee (3) to be satisfied by the actual system (1), robust constraint enforcement techniques for local MPC are needed and are developed in what follows.

To begin with, we consider the following polyhedral approximations of unit ball:

$$\begin{aligned}
\bar{\mathcal{X}}&=\left\{x\in\mathbb{R}^n:\bar{G}x\le\bar{g}\right\} \\
\bar{\mathcal{U}}&=\left\{u\in\mathbb{R}^m:\bar{H}u\le\bar{h}\right\} \qquad (26)
\end{aligned}$$

such that $\|x\|\le 1$ for all $x\in\bar{\mathcal{X}}$ and $\|u\|\le 1$ for all $u\in\bar{\mathcal{U}}$.

We then impose the following constraints on the predicted states and controls in the local MPC problem

$$\bar{x}_{\tau|t}\in\alpha_{\tau|t}\bar{\mathcal{X}},\quad \bar{u}_{\tau|t}\in\beta_{\tau|t}\bar{\mathcal{U}} \qquad (27)$$

for $\tau=t,\dots,T-1$, where $\alpha_{\tau|t}$ and $\beta_{\tau|t}$ are auxiliary decision variables (i.e., to be optimized together with the state and control variables in the local MPC problem). The constraints in (27) can be expressed as the following linear inequalities in $(\bar{x}_{\tau|t},\alpha_{\tau|t})$ and $(\bar{u}_{\tau|t},\beta_{\tau|t})$:

$$\begin{aligned}
\bar{G}\bar{x}_{\tau|t}-\alpha_{\tau|t}\bar{g}&\le 0 \\
\bar{H}\bar{u}_{\tau|t}-\beta_{\tau|t}\bar{h}&\le 0. \qquad (28)
\end{aligned}$$

Under (27) or (28), we have $\|\bar{x}_{\tau|t}\|\le\alpha_{\tau|t}$ and $\|\bar{u}_{\tau|t}\|\le\beta_{\tau|t}$ for $\tau=t,\dots,T-1$. Then, using (21), the difference between the predicted state $\bar{x}_{T|t}$ and the actual state x_T can be bounded as

$$\|\bar{x}_{T|t}-x_T\|\le\sum_{l=t}^{T-1}(a+L_f)^{T-l-1}\left(L_f\alpha_{l|t}+M_f\beta_{l|t}+\omega\right). \qquad (29)$$

To ensure the constraint (3) will be satisfied by the actual state x_T, we finally impose the following constraint in the local MPC problem:

$$\bar{x}_{T|t}\in\mathcal{X}_T\sim\mathcal{B}_{\xi_{T|t}} \qquad (30)$$

where $\mathcal{B}_{\xi_{T|t}}=\{x\in\mathbb{R}^n:\|x\|\le\xi_{T|t}\}$ and $\xi_{T|t}$ denotes the bound for $\|\bar{x}_{T|t}-x_T\|$ on the right-hand side of (29).

Similar to (18), the constraint (30) can be expressed as the following linear inequalities in $(\bar{x}_{T|t},\{\alpha_{\tau|t},\beta_{\tau|t}\}_{\tau=t,\dots,T-1})$ using the support function for unit ball $h_{\mathcal{B}}$:

$$\begin{aligned}
&G_{T,j}^\top\bar{x}_{T|t}+h_{\mathcal{B}}(G_{T,j})\,\xi_{T|t}=G_{T,j}^\top\bar{x}_{T|t} \\
&+h_{\mathcal{B}}(G_{T,j})\left(\sum_{l=t}^{T-1}(a+L_f)^{T-l-1}\left(L_f\alpha_{l|t}+M_f\beta_{l|t}+\omega\right)\right) \\
&\le g_{T,j},\qquad j=1,\dots,p_T. \qquad (31)
\end{aligned}$$

The robust constraint enforcement property of the (state-) constrained local MPC problem (i.e., (19) with the generic constraint (19d) elaborated as (27) and (30)) is further formalized in Lemma 2 in Section IV. Note that this problem is reformulated according to the current state measurement x_t at every sample time instant $t = 0, 1 \ldots, N-1$. It may not be feasible at all times. If the constrained local MPC problem is infeasible at some time t, as a fail-safe solution, we let

$$\bar{u}_{\tau|t} = \bar{u}_{\tau|t-1} \tag{32}$$

for $\tau = t, \ldots, N-1$, i.e., whenever the local MPC problem is infeasible, we take the remaining control sequence obtained from the previous feasible time as the solution for the current time. This fail-safe solution ensures that the local controller produces a control signal for the plant to apply at every time step.

IV. Switching-Based Control Fusion

A distinguishing component of our cloud-assisted nonlinear MPC approach is a fusion scheme to systematically combine the cloud and local MPC controls obtained in Section III to achieve enhanced performance. We describe our proposed switching-based fusion scheme in this section. We first present a simpler version of our fusion policy for the case without state constraints in Section IV-A, then present a modified version for the case with state constraints in Section IV-B, together with analysis of its feasibility and constraint satisfaction properties.

A. Switch Policy for Unconstrained Case

At each time instant $t = 0, 1 \ldots, N-1$, two candidate controls, $\hat{u}_t$ from cloud MPC and $\bar{u}_{t|t}$ from local MPC, are available. The goal is to systematically fuse them to achieve enhanced performance. The fundamental idea of the proposed switching-based fusion strategy is to minimize future cumulative cost: One should apply $\hat{u}_t$ if $J_t^c \le J_t^l$ and apply $\bar{u}_{t|t}$ otherwise. However, only the estimates of J_t^c and J_t^l, i.e., $\hat{J}_t$ and $\bar{J}_t$, but themselves, are available, and these estimates have errors. Therefore, following the idea of minimizing the worst-case cost, we propose the following switching policy for the case without state constraints:

$$u_t = \begin{cases} \hat{u}_t, & \text{if } \hat{J}_t + \hat{\eta}_t \le \bar{J}_t + \bar{\eta}_t \\ \bar{u}_{t|t}, & \text{otherwise} \end{cases} \tag{33}$$

where $\hat{\eta}_t$ denotes the bound for $|\hat{J}_t - J_t^c|$ on the right-hand side of (10) and $\bar{\eta}_t$ denotes the bound for $|\bar{J}_t - J_t^l|$ on the right-hand side of (22). This switching policy completes the unconstrained version of our proposed cloud-assisted MPC approach to finite-duration control tasks.

The switching policy (33) uses the bounds $\hat{\eta}_t$ and $\bar{\eta}_t$ developed in (10) and (22) based on Lipschitz constants of relevant functions to estimate the worst-case costs and determine an appropriate control to apply. The proposed switching strategy is essentially based on the worst-case performance bounds and thus may not be the optimal policy (i.e., the policy that minimizes the actual cumulative cost). Note that computation of optimal policies requires exact evaluation of disturbance effects propagating through the nonlinear system (1), which is in general a difficult task, if not impossible. On the other hand, we show through simulation examples in Section V that our switching policy (33) can lead to improved control performance than using solely cloud MPC or local MPC controls. Investigation into methods to compute/approximate optimal policies is left to our future work.

B. Switch Policy Accounting for Constraints

To guarantee satisfaction of the constraint (3) by the actual state x_T, we modify the switch policy (33) for $0 \le t \le T-1$ to the following policy:

$$u_t = \begin{cases} \hat{u}_t, & \text{if } \hat{J}_t + \hat{\eta}_t \le \bar{J}_t + \bar{\eta}_t \text{ and } \|\hat{x}_t - x_t\| \le \delta_t \\ \bar{u}_{t|t}, & \text{otherwise} \end{cases} \tag{34}$$

where δ_t is defined in (16) and we let $\bar{J}_t = +\infty$ if the local MPC problem is infeasible at t.

We now discuss the constraint satisfaction property of the policy (34). Our main result, Proposition 3, is built upon two intermediate results, Lemmas 1 and 2, which formalize the robust constraint enforcement properties of the cloud and local MPC problems, respectively.

Lemma 1: Suppose (i) (17) is enforced in the cloud MPC problem (i.e., when the cloud MPC controls $\{\hat{u}_0, \ldots, \hat{u}_{N-1}\}$ are computed at the initial time of the control task), (ii) at time k, for some $0 \le k \le T-1$, the difference between the cloud predicted state $\hat{x}_k$ and the actual state x_k satisfies $\|\hat{x}_k - x_k\| \le \delta_k$, with δ_k defined in (16), and (iii) the cloud MPC controls $\{\hat{u}_k, \ldots, \hat{u}_{T-1}\}$ are applied to the actual system (1) over the steps $\tau = k, \ldots, T-1$. Then, the following two results must hold true: (I) the difference between the cloud predicted state $\hat{x}_T$ and the actual state x_T satisfies $\|\hat{x}_T - x_T\| \le \delta_T$, and (II) the constraint (3) is satisfied by the actual state x_T.

Proof: The proof of (I) follows similar steps as (12) and (13), with ϵ_k replaced by δ_k. Then, since (17) is enforced and $\|\hat{x}_T - x_T\| \le \delta_T$, we must have $x_T \in \hat{x}_T \oplus \mathcal{B}_{\delta_T} \subseteq (\mathcal{X}_T \sim \mathcal{B}_{\delta_T}) \oplus \mathcal{B}_{\delta_T} \subseteq \mathcal{X}_T$, where $\oplus$ denotes the Minkowski sum operation [27]. This proves (II). ■

Lemma 2: Suppose at time t, $0 \le t \le T-1$, the constraints (27) and (30) are enforced in the local MPC problem and a sequence of (feasible) optimal controls $\{\bar{u}_{t|t}, \ldots, \bar{u}_{N-1|t}\}$ are obtained. Then, if $\{\bar{u}_{t|t}, \ldots, \bar{u}_{T-1|t}\}$ are applied to the actual system (1) over the steps $\tau = t, \ldots, T-1$, the constraint (3) is necessarily satisfied by the actual state x_T.

Proof: Let us denote the $\alpha_{\tau|t}, \beta_{\tau|t}$ values associated with the solution $\{\bar{u}_{t|t}, \ldots, \bar{u}_{N-1|t}\}$ as $\bar{\alpha}_{\tau|t}, \bar{\beta}_{\tau|t}$ and let

$$\bar{\xi}_{T|t} = \sum_{l=t}^{T-1} (a + L_f)^{T-l-1} \left(L_f \bar{\alpha}_{l|t} + M_f \bar{\beta}_{l|t} + \omega \right). \tag{35}$$

If $\{\bar{u}_{t|t}, \ldots, \bar{u}_{T-1|t}\}$ are applied to the actual system over the steps $\tau = t, \ldots, T-1$, then according to (29) we have

$$\|\bar{x}_{T|t} - x_T\| \le \bar{\xi}_{T|t} \tag{36}$$

where $\bar{x}_{T|t}$ denotes the local MPC predicted state corresponding to the controls $\{\bar{u}_{t|t}, \ldots, \bar{u}_{T-1|t}\}$. Meanwhile, the constraint

(30) ensures

$$\bar{x}_{T|t} \in \mathcal{X}_T \sim \mathcal{B}_{\bar{\xi}_{T|t}}. \tag{37}$$

Therefore, we have $x_T \in \bar{x}_{T|t} \oplus \mathcal{B}_{\bar{\xi}_{T|t}} \subseteq (\mathcal{X}_T \sim \mathcal{B}_{\bar{\xi}_{T|t}}) \oplus \mathcal{B}_{\bar{\xi}_{T|t}} \subseteq \mathcal{X}_T$. This proves the result. ■

On the basis of the above two results, we now show that the modified switching policy (34) leads to the following constraint satisfaction result:

Proposition 3: Suppose (i) at the initial time $t = 0$, the state-constrained cloud MPC problem (i.e., (6) with the generic constraint (6c) elaborated as (17)) is feasible, and (ii) at each time instant $t = 0, \ldots, T-1$, the control that is applied to the actual system (1) is determined by the switching policy (34). Then, the constraint (3) is necessarily satisfied by the actual state x_T.

Proof: Firstly, following the proof of Lemma 1(I), it can be shown that if $\|\hat{x}_t - x_t\| \leq \delta_t$ holds at some t, $0 \leq t \leq T-1$, and $u_t = \hat{u}_t$, then the resulting actual state x_{t+1} must satisfy $\|\hat{x}_{t+1} - x_{t+1}\| \leq \delta_{t+1}$. In this case, according to the switching policy (34), if the control is switched from cloud MPC control to local MPC control at some t (i.e., $u_{t-1} = \hat{u}_{t-1}$ and $u_t = \bar{u}_{t|t}$), it must hold that $\bar{J}_t + \bar{\eta}_t < \hat{J}_t + \hat{\eta}_t$, which implies $\bar{J}_t < +\infty$, i.e., the constrained local MPC problem is feasible at t.

Let us now consider two cases for the control at time $t = T-1$: (a) If $u_{T-1} = \hat{u}_{T-1}$, according to (34), it must hold that $\|\hat{x}_{T-1} - x_{T-1}\| \leq \delta_{T-1}$. In this case, according to Lemma 1(II), we must have $x_T \in \mathcal{X}_T$. (b) If $u_{T-1} = \bar{u}_{T-1|T-1}$, according to the analysis in the first paragraph of this proof, there must exist some τ, $0 \leq \tau \leq T-1$, such that τ is the last time instant where the constrained local MPC problem is feasible. Moreover, according to the fail-safe policy (32), we must have $u_\tau = \bar{u}_{\tau|\tau}$, $u_{\tau+1} = \bar{u}_{\tau+1|\tau+1} = \bar{u}_{\tau+1|\tau}, \ldots, u_{T-1} = \bar{u}_{T-1|T-1} = \bar{u}_{T-1|\tau}$. To see this more clearly, if the constrained local MPC problem is feasible at $T-1$, then $\tau = T-1$ and the above statement holds true. In the case where the constrained local MPC problem is infeasible at $T-1$, the analysis in the first paragraph of this proof says that u_{T-2} cannot take the cloud MPC control $\hat{u}_{T-2}$ (because in that case the control would not switch to local MPC control at $t = T-1$ when the local MPC problem is infeasible at $T-1$). Therefore, in this case we must have $u_{T-2} = \bar{u}_{T-2|T-2}$, and, according to the fail-safe policy (32), $u_{T-1} = \bar{u}_{T-1|T-1} = \bar{u}_{T-1|T-2}$. By continuing this analysis, we can show the statement above. Then, according to Lemma 2, we must have $x_T \in \mathcal{X}_T$. This completes the proof. ■

Remark 3: All of our above developments and theoretical results, including Propositions 1–3, apply to the cases with or without control constraints $u_t \in \mathcal{U}$. Such control constraints can be handled by imposing them directly in the cloud and local MPC optimization problems without extra treatments.

Remark 4 (Closed-loop stability-type property): The robust constraint enforcement approaches based on constraint tightening developed in Sections III-A, III-B, and IV-B can be used to establish closed-loop stability-type properties of the system in continued operation. Specifically, one can design a local controller, $u_t = \pi(x_t)$, together with a set, $\mathcal{X}_f$, such that the controller π stabilizes the system for states within the set $\mathcal{X}_f$. Note that due to consistent perturbation by the disturbance signal w_t, it is in general not easy to achieve stability (in the sense of Lyapunov), in particular, $x_t \to 0$, by a state-feedback controller $u_t = \pi(x_t)$. Therefore, by stabilization we mean regional input-to-state stability [34] or ultimate boundedness, i.e., $\limsup_{t\to\infty} \|x_t\| \leq \rho$, where the bound ρ depends on the disturbance set W and the controller. Then, one can impose $\mathcal{X}_f$ as a terminal constraint set and apply the approaches developed above to robustly enforce $x_N \in \mathcal{X}_f$. Proposition 3 shows that, under the assumption of initial feasibility, the system state x_t necessarily enters $\mathcal{X}_f$ at $t = N$. After x_t enters $\mathcal{X}_f$, the local controller π can be activated to take over the control and stabilize the system. For more detailed discussion about such an approach to establishing closed-loop stability-type properties through a local stabilizing controller and a terminal set, the readers are referred to the MPC survey article [35].

Remark 5: Our approach and associated theoretical results, including the error bounds in Propositions 1 and 2 and the robust constraint satisfaction property in Proposition 3, can be extended and used to treat time-varying systems of the form

$$x_{t+1} = A_t x_t + B_t u_t + f_t(x_t, u_t) + w_t \tag{38}$$

by defining $a = \max_{t=0,\ldots,N-1} \|A_t\|$, $L_f = \max_{t=0,\ldots,N-1} L_{f,t}$ and $M_f = \max_{t=0,\ldots,N-1} M_{f,t}$ with $L_{f,t}$ and $M_{f,t}$ being the Lipschitz constants of f_t. Such a treatment based on over-bounding may lead to conservative control performance and is thus not a focus of this article. However, this fact can be useful in some applications.

V. Simulation Examples

In this section, three simulation examples are presented to demonstrate the effectiveness of our proposed cloud-assisted nonlinear MPC approach. The simulations are implemented in MATLAB 2019b. All computations are performed on a laptop with an Intel i7-10710 U CPU with six cores, 1.6 GHz clock rate and 16 GB RAM. The cloud-assisted MPC framework is designed for finite-duration control tasks, in which the cloud MPC problem is solved only once at the initial time while the shrinking-horizon local MPC problem is solved at each sample time instant. MATLAB MPC toolbox is used to facilitate the MPC formulation, and the induced optimization problems are solved with MATLAB *fmincon* function. We note that in real implementation, the cloud MPC problem can be handled in the cloud by standard nonlinear programming solvers (such as derivatives-based algorithms [36] or evolutionary algorithms [37]). The best solver choice may depend on specific problem and cloud computing architecture, the investigation of which is left to future research.

A. Example 1: First-Order System

As the first example, we consider a first-order system in the form of (1) with the following parameters:

$$A = 0.75, \quad B = 1, \quad f(x_t, u_t) = 0.1x_t - \sin(0.1x_t)$$
$$|w_t| \leq 0.02, \quad x_0 = -10, \quad \varepsilon_0 = \hat{x}_0 - x_0 = -0.5. \tag{39}$$

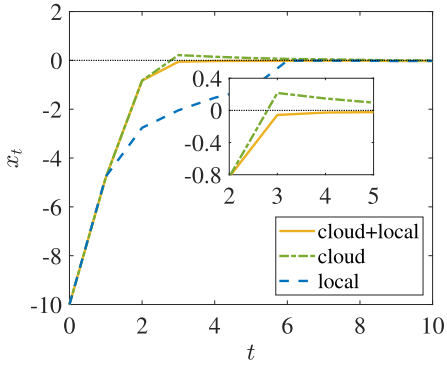


Fig. 3. Simulation results of Example 1: State trajectories.

We consider a control task defined by the following cost function to minimize:

$$J = \sum_{t=0}^{N-1} \left(|x_t| + \sqrt{5}|u_t| \right) + \sqrt{2}|x_N| \tag{40}$$

where $N = 10$ corresponds to the end time of the control task. Note that we consider such a cost function because it is globally Lipschitz, i.e., satisfying our assumption **A3** globally, and thereby facilitates the implementation of our approach and the validation of our theoretical results. Moreover, we assume the control input, u_t, and the system state at the terminal time, x_N, are subjected to the following constraints:

$$\begin{aligned} u_t \in \mathcal{U} &= \{u \in \mathbb{R} : |u| \leq 3\}, \ t = 0, 1, \ldots, N-1 \\ x_N \in \mathcal{X}_N &= \{x \in \mathbb{R} : |x| \leq 2.5\}. \end{aligned} \tag{41}$$

From (39), it can be obtained that $\left|\frac{\partial f}{\partial x}\right| = |0.1 - 0.1\cos(0.1x)| \leq 0.2$ and $\frac{\partial f}{\partial u} = 0$, indicating that $L_f = 0.2$ and $M_f = 0$. The cost function in (40) is Lipschitz continuous with $L_\phi = 1$ and $L_\psi = \sqrt{2}$. Based on these Lipschitz constants, the constraint handling techniques presented in Section III can be used to robustly enforce the constraints in (41). For instance, at the initial time $t = 0$, the tightened constraint sets defined in (17) and (30) are $\mathcal{X}_N \sim \mathcal{B}_{\delta_N} = \{x \in \mathbb{R} : |x| \leq 2.0401\}$ and $\mathcal{X}_N \sim \mathcal{B}_{\xi_{N|0}} = \{x \in \mathbb{R} : |x| \leq 0.3510\}$, respectively. To clearly demonstrate that our proposed strategy of fusing the cloud and local MPC controls using the switching policy (34) can effectively improve the overall control performance, we also implement "sole cloud MPC" and "sole local MPC" (i.e., the cloud/local MPC solutions are applied over the entire control task duration without switching) for comparison.

Fig. 3 shows the simulated trajectories of system state x_t corresponding to different MPC schemes. It can be seen that our proposed approach of combining cloud and local MPC controls leads to the best transient response. The mean regulation error (MRE $= \frac{1}{N+1}\sum_{t=0}^{N}|x_t|$) and the resulting actual cost value of each method are summarized in Table I, where it can be seen that our cloud and local MPC fusion strategy achieves the smallest MRE and cost.

Fig. 4 illustrates the actual cost-to-go values J_t^l, J_t^c, their worst-case estimates $\bar{J}_t + \bar{\eta}_t, \hat{J}_t + \hat{\eta}_t$, and the cloud-local switching sequence. Recall that in the constrained version of our switching policy, (34), in which control to apply is determined by two conditions. We have found that the second condition, $|\hat{x}_t - x_t| \leq \delta_t$, is always satisfied in this example, and in this case switching is determined by the sign of $(\bar{J}_t + \bar{\eta}_t) - (\hat{J}_t + \hat{\eta}_t)$. In particular, $\text{sign}(\bar{J}_t + \bar{\eta}_t - \hat{J}_t - \hat{\eta}_t) = 1$ indicates that the cloud MPC control $\hat{u}_t$ is applied and $\text{sign}(\bar{J}_t + \bar{\eta}_t - \hat{J}_t - \hat{\eta}_t) = -1$ indicates that the local MPC control $\bar{u}_{t|t}$ is applied. As discussed in the last paragragh of Section IV-A, our switching policy based on the worst-case cost-to-go estimates $\bar{J}_t + \bar{\eta}_t$ and $\hat{J}_t + \hat{\eta}_t$ may not be the optimal policy. The optimal policy that minimizes the actual cost-to-go can be determined by the sign of $J_t^l - J_t^c$. Nonetheless, the switching sequence determined by our policy matches the optimal switching sequence for 80% of the time instants, indicating that our policy is a reasonably good approximation of the optimal policy. Note that J_t^l and J_t^c, which determines the optimal policy, cannot be precomputed but can only be computed after the actual state trajectories from current time t to terminal time N corresponding to cloud and local MPC implementations have been revealed.

TABLE I
MREs AND ACTUAL COST VALUES OF DIFFERENT MPC APPROACHES

	cloud + local	cloud	local
MRE	1.5822	1.6219	2.2344
cost	29.9165	30.7720	32.8074

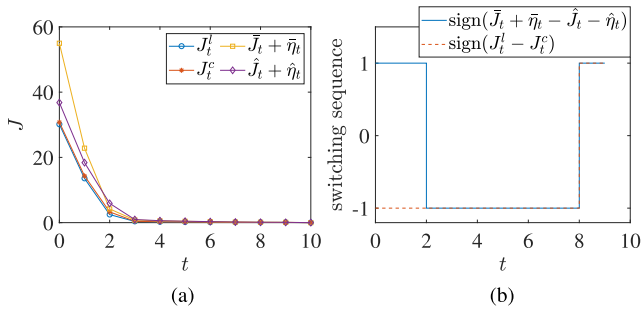


Fig. 4. Simulation results of Example 1: (a) Cost-to-go values and their estimates. (b) Switching sequences.

B. Example 2: Inverted Pendulum on a Cart

As shown in Fig. 5, this example considers an inverted pendulum mounted to a cart. The system state is defined as $x = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \end{bmatrix}^\top = \begin{bmatrix} z & \dot{z} & \theta & \dot{\theta} \end{bmatrix}^\top$, where z is the cart position and θ is the pendulum angle. The continuous-time model of the inverted pendulum system is given by [38]

$$\dot{x} = \begin{bmatrix} x_2 \\ \frac{F - K_d x_2 - m_{\text{pend}} L x_4^2 \sin(x_3) + m_{\text{pend}} g \sin(x_3)\cos(x_3)}{m_{\text{cart}} + m_{\text{pend}} \sin^2(x_3)} \\ x_4 \\ \frac{\dot{x}_2 \cos(x_3) + g \sin(x_3)}{L} \end{bmatrix} + w \tag{42}$$

where $m_{\text{cart}} = 1$ kg is the cart mass, $m_{\text{pend}} = 1$ kg is the pendulum mass, $L = 0.5$ m is the length of the pendulum,

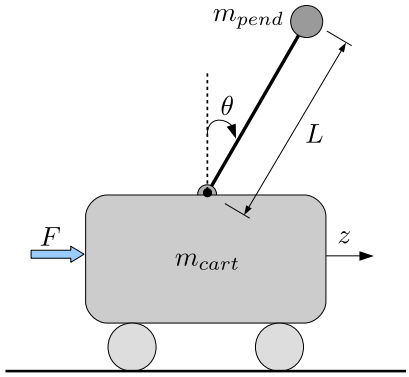


Fig. 5. Inverted pendulum on a cart.

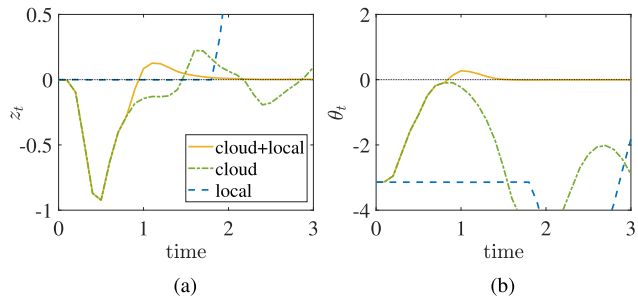


Fig. 6. Simulation results of Example 2: State trajectories.

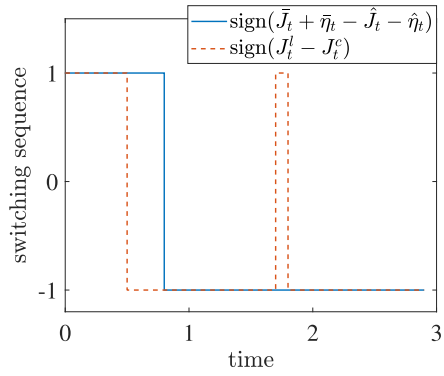


Fig. 7. Simulation results of Example 2: Switching sequences.

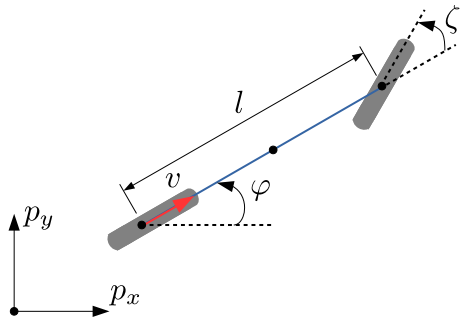


Fig. 8. Kinematic bicycle model.

$K_d = 10$ Ns/m is the damping parameter, $g = 9.81$ m/s^2 is the gravity acceleration, and w is the external disturbance. The system is controlled by a variable force F. We discretize the continuous-time model (42) with a sampling period of $\Delta T = 0.1$ s, and the derived discrete-time model is used by the cloud MPC to compute the control sequence. The lower-fidelity model used by the local MPC is obtained by linearizing (42) around the equilibrium $x = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}^\top$. The system is subjected to random disturbance w_t which is bounded within $W = \{w \in \mathbb{R}^4 : -0.001 I_4 \leq w \leq 0.001 I_4\}$, where $I_4 \in \mathbb{R}^4$ is a vector with all entries being one. It can be obtained that w_t is bounded by $\|w_t\|_2 \leq 0.002$, where $\|\cdot\|_2$ denotes the Euclidean norm. The upper bound of the error between the state estimate $\hat{x}_0$ and the actual state x_0 is chosen as $\delta_0 = 0.04$. The cost function is selected as

$$J = \sum_{t=0}^{N-1} \left(\|x_t\|_Q + \|u_t\|_R\right) + \|x_N\|_Q \tag{43}$$

where $x_t = \begin{bmatrix} z_t & \dot{z}_t & \theta_t & \dot{\theta}_t \end{bmatrix}^\top$, $u_t = F_t$, $Q = \text{diag}(3, 0.4, 3, 0.4)$, and $R = 1 \times 10^{-5}$. Based on the discrete-time model of the inverted pendulum system, it can be derived that $L_f = 4.6291$ and $M_f = 0.4031$. The cost function defined in (43) implies that $L_\phi = L_\psi = \sqrt{3}$.

The system state trajectories corresponding to different MPC schemes are presented in Fig. 6, and it can be seen that only our approach of combined cloud + local MPC achieves satisfactory convergence. Recall that the local MPC relies on a linearized model, which is accurate only for states within a small neighborhood of the linearization point. Because the initial condition x_0 is far from the linearization point, the local MPC fails to stabilize the inverted pendulum. Similarly, the cloud MPC also fails to stabilize the inverted pendulum due to an accumulation of errors over time. Recall that the cloud MPC, with its control trajectory computed only at the initial time $t = 0$, is essentially an open-loop control scheme, and therefore cannot counteract such errors through feedback. Fig. 7 shows the switching sequences of $(\bar{J}_t + \bar{\eta}_t) - (\hat{J}_t + \hat{\eta}_t)$ and $J_t^l - J_t^c$ with a 86.67% match.

C. Example 3: Autonomous Vehicle Path Following

This example considers the path following of an autonomous vehicle as presented in Fig. 8. The kinematic model of the vehicle is described by [39]

$$\begin{bmatrix} \dot{p}_x \\ \dot{p}_y \\ \dot{\varphi} \end{bmatrix} = \begin{bmatrix} v\cos(\varphi) \\ v\sin(\varphi) \\ \frac{v}{l}\tan(\zeta) \end{bmatrix} + w \tag{44}$$

where (p_x, p_y) denotes the position of the vehicle, φ is the yaw angle, l is the wheelbase, and w is the external disturbance. The speed v and steering angle ζ are the control variables for the vehicle. Denote $x^{\text{ref}} = \begin{bmatrix} p_x^{\text{ref}} & p_y^{\text{ref}} & \varphi^{\text{ref}} \end{bmatrix}^\top$ and $u^{\text{ref}} = \begin{bmatrix} v^{\text{ref}} & \zeta^{\text{ref}} \end{bmatrix}^\top$ as the reference path to follow and its corresponding control variables. Then, by letting $\gamma = \tan(\zeta)$ ($\gamma^{\text{ref}} = \tan(\zeta^{\text{ref}})$), $\tilde{x} =$

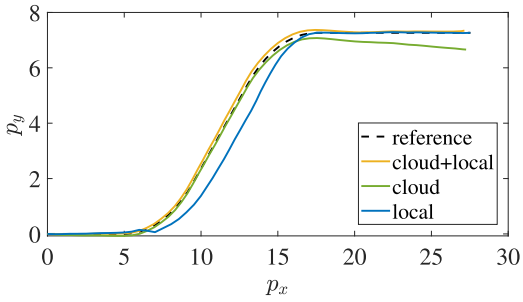


Fig. 9. Simulation results of Example 3: Position trajectories.

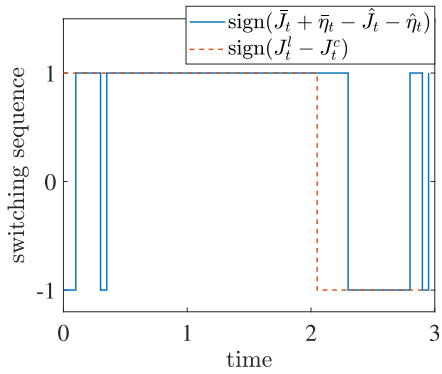


Fig. 10. Simulation results of Example 3: Switching sequences.

$\begin{bmatrix}\tilde{p}_x & \tilde{p}_y & \tilde{\varphi}\end{bmatrix}^\top = \begin{bmatrix}p_x - p_x^{\mathrm{ref}} & p_y - p_y^{\mathrm{ref}} & \varphi - \varphi^{\mathrm{ref}}\end{bmatrix}^\top$, and $\tilde{u} = \begin{bmatrix}\tilde{v} & \tilde{\gamma}\end{bmatrix}^\top = \begin{bmatrix}v - v^{\mathrm{ref}} & \gamma - \gamma^{\mathrm{ref}}\end{bmatrix}^\top$, the model (44) can be written as

$$\dot{\tilde{x}} = A\tilde{x} + B\tilde{u} + f + w \tag{45}$$

where

$$A = \begin{bmatrix}0 & 0 & -v^{\mathrm{ref}}\sin(\varphi^{\mathrm{ref}}) \\ 0 & 0 & v^{\mathrm{ref}}\cos(\varphi^{\mathrm{ref}}) \\ 0 & 0 & 0\end{bmatrix}, B = \begin{bmatrix}\cos(\varphi^{\mathrm{ref}}) & 0 \\ \sin(\varphi^{\mathrm{ref}}) & 0 \\ \frac{\gamma^{\mathrm{ref}}}{l} & \frac{v^{\mathrm{ref}}}{l}\end{bmatrix}$$

$$f = \begin{bmatrix}\left(\cos(\tilde{\varphi}+\varphi^{\mathrm{ref}}) - \cos(\varphi^{\mathrm{ref}})\right)(\tilde{v}+v^{\mathrm{ref}}) + v^{\mathrm{ref}}\sin(\varphi^{\mathrm{ref}})\tilde{\varphi} \\ \left(\sin(\tilde{\varphi}+\varphi^{\mathrm{ref}}) - \sin(\varphi^{\mathrm{ref}})\right)(\tilde{v}+v^{\mathrm{ref}}) - v^{\mathrm{ref}}\cos(\varphi^{\mathrm{ref}})\tilde{\varphi} \\ \frac{\tilde{v}\tilde{\gamma}}{l}\end{bmatrix}.$$

The continuous-time model (45) is discretized with a sampling period of $\Delta T = 0.05$ s as the higher-fidelity model for cloud MPC, while the linear model used by the local MPC is obtained by neglecting the nonlinear term f. The disturbance w_t acting on the system is bounded within $W = \{w \in \mathbb{R}^3 : -0.01I_3 \leq w \leq 0.01I_3\}$, indicating that $\|w_t\|_2 \leq 0.0173$. The upper bound of the error between the state estimate $\hat{\tilde{x}}_0$ and the actual state $\tilde{x}_0$ is set as $\delta_0 = 0.0346$. Furthermore, for tracking the reference path, the cost function is design as

$$J = \sum_{t=0}^{N-1}\left(\|\tilde{x}_t\|_Q + \|\tilde{u}_t\|_R\right) + \|\tilde{x}_N\|_Q \tag{46}$$

where $Q = \mathrm{diag}(3, 3, 0.01)$ and $R = 0.001I_{2\times 2}$. Note that the system parameters A, B, and f are related to the time-varying signal u^{ref}. In this case, we derive the constants L_f and M_f for cost estimation according to Remark 5, where $L_f = 0.5$ and $M_f = 0.3357$. The cost function defined in (46) is Lipschitz continuous with $L_\phi = L_\psi = \sqrt{3}$.

The position trajectories of the vehicle corresponding to different MPC methods are illustrated in Fig. 9, and it is clear that our cloud + local MPC approach achieves the most accurate path following. Furthermore, Fig. 10 presents the switching sequences of $(\bar{J}_t + \bar{\eta}_t) - (\hat{J}_t + \hat{\eta}_t)$ and $J_t^l - J_t^c$ with a 81.67% match.

VI. Conclusion and Future Work

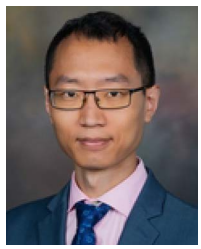
In this article, we developed a cloud-assisted MPC framework for finite-duration control tasks. In this framework, cloud-computed control trajectory based on a higher-fidelity nonlinear model and local shrinking-horizon MPC solutions based on a lower-fidelity linear model are fused by a switching policy to achieve improved control performance in the presence of cloud/local model prediction errors (due to plant-model mismatches and cloud request-response/communication delay effects). We analyzed properties of the proposed cloud-assisted MPC framework and established approaches to robustly handling constraints within this framework in spite of model prediction errors. We then demonstrated the effectiveness of the framework in terms of improving overall control performance using multiple simulation examples, including an automotive control example to illustrate its potential industrial applications.

For future work, we will investigate methods to refine our switching policy as well as investigate other fusion schemes to achieve further improved control performance. We will also extend our cloud-assisted MPC framework from finite-duration control tasks to a broader range of application scenarios.

References

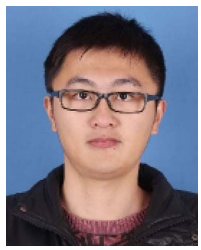
[1] R. L. Grossman, "The case for cloud computing," *IT Professional*, vol. 11, no. 2, pp. 23–27, 2009.
[2] O. Givehchi, H. Trsek, and J. Jasperneite, "Cloud computing for industrial automation systems–A comprehensive overview," in *Proc. IEEE Conf. Emerg. Technol. Factory Automat.*, 2013, pp. 1–4.
[3] T. Hegazy and M. Hefeeda, "Industrial automation as a cloud service," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 10, pp. 2750–2763, Oct. 2015.
[4] O. Givehchi, J. Imtiaz, H. Trsek, and J. Jasperneite, "Control-as-a-service from the cloud: A case study for using virtualized PLCs," in *Proc. IEEE Workshop Factory Commun. Syst.*, 2014, pp. 1–4.
[5] Z. Li, I. Kolmanovsky, E. Atkins, J. Lu, D. Filev, and J. Michelini, "Cloud aided semi-active suspension control," in *Proc. IEEE Symp. Comput. Intell. Veh. Transp. Syst.*, 2014, pp. 76–83.
[6] Z. Li, I. Kolmanovsky, E. Atkins, J. Lu, D. P. Filev, and J. Michelini, "Road risk modeling and cloud-aided safety-based route planning," *IEEE Trans. Cybern.*, vol. 46, no. 11, pp. 2473–2483, Nov. 2016.
[7] Z. Li, I. V. Kolmanovsky, E. M. Atkins, J. Lu, D. P. Filev, and Y. Bai, "Road disturbance estimation and cloud-aided comfort-based route planning," *IEEE Trans. Cybern.*, vol. 47, no. 11, pp. 3879–3891, Nov. 2017.
[8] E. F. Camacho and C. Bordons, *Model Predictive Control*. London, U.K.: Springer, 2007.

[9] E. Alpaydin, *Introduction to Machine Learning*. Cambridge, MA, USA: MIT Press, 2020.
[10] S. Mubeen, P. Nikolaidis, A. Didic, H. Pei-Breivold, K. Sandström, and M. Behnam, "Delay mitigation in offloaded cloud controllers in industrial IoT," *IEEE Access*, vol. 5, pp. 4418–4430, 2017.
[11] I. Pelle, J. Czentye, J. Dóka, and B. Sonkoly, "Towards latency sensitive cloud native applications: A performance study on AWS," in *Proc. IEEE Int. Conf. Cloud Comput.*, 2019, pp. 272–280.
[12] P. Skarin, J. Eker, M. Kihl, and K.-E. Årzén, "Cloud-assisted model predictive control," in *Proc. IEEE Int. Conf. Edge Comput.*, 2019, pp. 110–112.
[13] P. Skarin, J. Eker, and K.-E. Årzén, "Cloud-based model predictive control with variable horizon," in *Proc. Int. Federation Autom. Control World Congr.*, 2020, pp. 6993–7000.
[14] P. Skarin, J. Eker, and K.-E. Årzén, "A cloud-enabled rate-switching MPC architecture," in *Proc. IEEE Conf. Decis. Control*, 2020, pp. 3151–3158.
[15] M. S. Darup, "Encrypted model predictive control in the cloud," in *Privacy in Dynamical Systems*. Berlin, Germany: Springer, 2020, pp. 231–265.
[16] Y. Chen, Z. Du, and M. Garcia-Acosta, "Robot as a service in cloud computing," in *Proc. IEEE 5th Int. Symp. Service Oriented Syst. Eng.*, 2010, pp. 151–158.
[17] J. Delsing, F. Rosenqvist, O. Carlsson, A. W. Colombo, and T. Bangemann, "Migration of industrial process control systems into service oriented architecture," in *Proc. 38th Annu. Conf. IEEE Ind. Electron. Soc.*, 2012, pp. 5786–5792.
[18] H. Sequeira, P. Carreira, T. Goldschmidt, and P. Vorst, "Energy cloud: Real-time cloud-native energy management system to monitor and analyze energy consumption in multiple industrial sites," in *Proc. IEEE/ACM 7th Int. Conf. Utility Cloud Comput.*, 2014, pp. 529–534.
[19] L. Heilig, R. R. Negenborn, and S. Voß, "Cloud-based intelligent transportation systems using model predictive control," in *Proc. Int. Conf. Comput. Logistics*, 2015, pp. 464–477.
[20] J. P. Hespanha, P. Naghshtabrizi, and Y. Xu, "A survey of recent results in networked control systems," *Proc. IEEE*, vol. 95, no. 1, pp. 138–162, Jan. 2007.
[21] R. A. Gupta and M.-Y. Chow, "Networked control system: Overview and research trends," *IEEE Trans. Ind. Electron.*, vol. 57, no. 7, pp. 2527–2535, Jul. 2010.
[22] C. Hatipoglu, U. Ozguner, and K. A. Redmill, "Automated lane change controller design," *IEEE Trans. Intell. Transp. Syst.*, vol. 4, no. 1, pp. 13–22, Mar. 2003.
[23] T. Haberkorn, P. Martinon, and J. Gergaud, "Low thrust minimum-fuel orbital transfer: A homotopic approach," *J. Guidance, Control, Dyn.*, vol. 27, no. 6, pp. 1046–1060, 2004.
[24] B. Acikmese and S. R. Ploen, "Convex programming approach to powered descent guidance for mars landing," *J. Guidance, Control, Dyn.*, vol. 30, no. 5, pp. 1353–1366, 2007.
[25] A. Vick, J. Guhl, and J. Krüger, "Model predictive control as a service – concept and architecture for use in cloud-based robot control," in *Proc. Int. Conf. Methods Models Automat. Robot.*, 2016, pp. 607–612.
[26] M. L. Darby, M. Nikolaou, J. Jones, and D. Nicholson, "RTO: An overview and assessment of current practice," *J. Process Control*, vol. 21, no. 6, pp. 874–884, 2011.
[27] I. Kolmanovsky and E. G. Gilbert, "Theory and computation of disturbance invariant sets for discrete-time linear systems," *Math. Problems Eng.*, vol. 4, no. 4, pp. 317–367, 1998.
[28] H. H. Sohrab, *Basic Real Analysis*, vol. 231. Berlin, Germany: Springer, 2003.
[29] J. He and X. Yao, "Towards an analytic framework for analysing the computation time of evolutionary algorithms," *Artif. Intell.*, vol. 145, no. 1/2, pp. 59–97, 2003.
[30] A. Richards and J. How, "Robust stable model predictive control with constraint tightening," in *Proc. Amer. Control Conf.*, 2006, pp. 1557–1562.
[31] D. L. Marruedo, T. Alamo, and E. Camacho, "Input-to-state stable MPC for constrained discrete-time nonlinear systems with bounded additive uncertainties," in *Proc. IEEE Conf. Decis. Control*, 2002, pp. 4619–4624.
[32] J. Köhler, M. A. Müller, and F. Allgöwer, "A novel constraint tightening approach for nonlinear robust model predictive control," in *Proc. Amer. Control Conf.*, 2018, pp. 728–734.
[33] U. V. Kalabić and I. V. Kolmanovsky, "A constraint-separation principle in model predictive control," *Automatica*, vol. 121, 2020, Art. no. 109190.
[34] D. Limon, I. Alvarado, A. Ferramosca, T. Alamo, and E. F. Camacho, "Enhanced robust NMPC based on nominal predictions," *IFAC Proc. Volumes*, vol. 43, no. 14, pp. 220–225, 2010.
[35] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.
[36] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty, *Nonlinear Programming: Theory and Algorithms*. Hoboken, NJ, USA: John Wiley & Sons, Inc., 2013.
[37] P. J. Fleming and R. C. Purshouse, "Evolutionary algorithms in control systems engineering: A survey," *Control Eng. Pract.*, vol. 10, no. 11, pp. 1223–1241, 2002.
[38] R. Gurumoorthy and S. Sanders, "Controlling non-minimum phase nonlinear systems–The inverted pendulum on a cart example," in *Proc. Amer. Control Conf.*, 1993, pp. 680–685.
[39] A. De Luca, G. Oriolo, and C. Samson, "Feedback control of a nonholonomic car-like robot," in *Robot Motion Planning and Control*. Berlin, Germany: Springer, 1998, pp. 171–253.



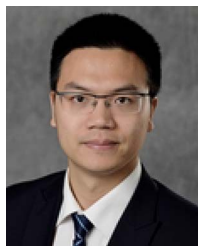
Nan Li (Member, IEEE) received the M.S. degree in mathematics and the Ph.D. degree in aerospace engineering from the University of Michigan, Ann Arbor, MI, USA, in 2020 and 2021, respectively.

He is currently an Assistant Professor with the Department of Aerospace Engineering, Auburn University, Auburn, AL, USA. From 2021 to 2022, prior to joining Auburn University, he was a Postdoctoral Research Fellow with the University of Michigan. His research interests are in stochastic control, learning, multiagent systems, and aerospace and automotive applications.



Kaixiang Zhang received the B.Eng. degree in automation from Xiamen University, Xiamen, China, and the Ph.D. degree in control science and engineering from Zhejiang University, Hangzhou, China, in 2014 and 2019, respectively.

From November 2017 to August 2018, he was a Visiting Scholar with the National Institute for Research in Computer Science and Automation, Rennes, France. He currently holds a Postdoctoral position with the Department of Mechanical Engineering, Michigan State University, East Lansing, MI, USA. His research interests include visual servoing, robotics, and nonlinear control.



Zhaojian Li (Senior Member, IEEE) received the B.Eng. degree from the Nanjing University of Aeronautics and Astronautics, in 2010, and the M.S. and Ph.D. degrees in aerospace engineering (flight dynamics and control) from the University of Michigan, Ann Arbor, MI, USA, in 2013 and 2015, respectively, both in aerospace engineering (flight dynamics and control).

He is currently an Assistant Professor with the Department of Mechanical Engineering, Michigan State University, East Lansing, MI, USA. His research interests include learning-based control, nonlinear and complex systems, and robotics and automated vehicles.

Dr. Li is a recipient of the NSF CAREER Award.

Vaibhav Srivastava (Senior Member, IEEE) received the B.Tech. degree in mechanical engineering from the Indian Institute of Technology Bombay, Mumbai, India, and the M.S. degree in mechanical engineering, in 2007 and 2011, respectively, the M.A. degree in statistics and the Ph.D. degree in mechanical engineering, from the University of California at Santa Barbara, Santa Barbara, CA, USA, both in 2012.

He served as a Lecturer and Associate Research Scholar with the Mechanical and Aerospace Engineering Department, Princeton University, Princeton, NJ, USA. He is currently an Assistant Professor with the Electrical and Computer Engineering Department, Michigan State University, East Lansing, MI, USA. He is also affiliated with mechanical engineering, cognitive science program, and connected and autonomous networked vehicles for active safety (CANVAS) program. His research focuses on cyber physical human systems with emphasis on mixed human–robot systems, networked multiagent systems, aerial robotics, and connected and autonomous vehicles.

Xiang Yin (Member, IEEE) was born in Anhui, China, in 1991. He received the B.Eng. degree from Zhejiang University, and the M.S. and Ph.D. degrees from the University of Michigan, Ann Arbor, MI, USA, in 2012, 2013, and 2017, all in electrical engineering.

Since 2017, he has been with the Department of Automation, Shanghai Jiao Tong University, Shanghai, China, where he is an Associate Professor. His research interests include formal methods, discrete-event systems, and cyber-physical systems.

Dr. Yin also received the IEEE Conference on Decision and Control (CDC) Best Student Paper Award Finalist in 2016. He is the cochair of the IEEE CSS Technical Committee on Discrete Event Systems. He is also a Member of the IEEE CSS Conference Editorial Board.