Hindawi
Journal of Chemistry
Volume 2019, Article ID 8035204, 12 pages
https://doi.org/10.1155/2019/8035204

Research Article

Fast Model Predictive Control Based on Adaptive Alternating Direction Method of Multipliers

Yu Li[1], **Qiming Zou**[2], **Xiaoru Ji**[1], **Chanyuan Zhang**[1] and **Ke Lu**[1,3]

[1]*School of Management Science and Engineering, Anhui University of Technology, Ma'anshan, China*
[2]*School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China*
[3]*School of Automation, Southeast University, Nanjing, China*

Correspondence should be addressed to Qiming Zou; qimingzou@aliyun.com and Ke Lu; luke.airoot@gmail.com

Received 29 July 2019; Accepted 6 September 2019; Published 30 September 2019

Guest Editor: Jia-Bao Liu

Model Predictive Control (MPC) can effectively handle control problem with disturbances, multicontrol variables, and complex constraints and is widely used in various control systems. In MPC, the control input at each time step is obtained by solving an online optimization problem, which will cause a time delay in real time on embedded computers with limited computational resources. In this paper, we utilize adaptive Alternating Direction Method of Multipliers (a-ADMM) to accelerate the solution of MPC. This method adaptively adjusts penalty parameter to balance the value of primal residual and dual residual. The performance of this approach is profiled via the control of a quadcopter with 12 states and 4 controls and prediction horizon ranging from 10 to 40. The simulation results demonstrate that the MPC based on a-ADMM has a significant improvement in real-time and convergence performance and thus is more suitable for solving large-scale optimal control problems.

1. Introduction

Model Predictive Control (MPC) [1, 2], a.k.a. receding horizon control, is a classical optimization-based control strategy for multivariable constrained systems. Since its appearance in the 1970s, MPC has been widely applied in a variety of areas, such as the transportation system [3, 4], grid system [5], and water distribution system [6]. MPC only utilizes the first element of the control input sequence obtained by solving an optimization problem. On the one hand, although receding horizon way makes MPC robust to model bias and external noises, MPC requires solving the optimization problem in each control cycle, which may cause a large time delay. On the other hand, low-complexity optimization algorithms have been motivated by the fact that several of these recent applications exploit microcontroller that has limited computing power [7–9].

For linear systems with a quadratic cost function and multiple linear constraints, MPC can be easily transformed into a standard quadratic programming (QP) form [10]. Therefore, off-the-shelf QP solvers can be used in this case. Most of them can be categorized into one of the two main classes, i.e., active-set and interior-point methods. Interior-point methods have become common choice for implementation because of its good convergence performance [11, 12]. However, it tends to demand significant online computation effort to compute the input. Recently, some modified methods have been proposed to reduce the computational complexity [13]. For example, Domahidi et al. decreased the computational burden by introducing a low-rank matrix forward substitution scheme and extended the interior-point methods to applications on low-cost embedded hardware. Compared to interior-point methods, active-set methods need on average substantially more iterations, but each iteration is computationally much cheaper and can be ward- or hot-start. For example, a parametric active-set algorithm for quadratic programming (qpOASES) is another commonly used QP solver which is applicable when numerous related QPs are solved sequentially, by exploiting the geometric property of state space [14].

Unfortunately, since qpOASES is an active-set-based method, it is limited to small- to medium-scale QP problem and sensitive to the external disturbance [14].

Aforementioned methods are centralized optimization algorithms, which are not appropriate for real-time large-scale applications on low-cost embedded computers. On the contrary, due to its success in solving large-scale optimization problems and pretty low computational complexity, distributed optimization algorithms have drawn increasing attention [15]. As a distributed optimization algorithm, Alternating Direction Method of Multipliers (ADMM) is a strong algorithm for solving convex optimization problems [16, 17]. ADMM was first introduced in the mid-1970s for the numerical solution of partial differential equations [18, 19] and particularly useful for solving optimization problems that are too large to be handled by generic optimization solvers. Because of its fast processing speed and good convergence performance, the method has been used in a large number of areas, such as compressed sensing [20], neural network [21], and image processing [22]. More details can be found in [17]. This wide range of applications has triggered a great interest in developing a better and more comprehensive understanding of the theoretical properties of ADMM.

ADMM has been proved to be an excellent fit for QP problems, since it can achieve linear convergence with strong convexity assumptions [23]. However, a drawback of ADMM is that the number of iterations required to converge is highly dependent on the step-size parameters. Unfortunately, analytic results for the optimal choice of penalty parameter are not available except for very special problems [24, 25]. In the special case of a strictly convex quadratic problem, criteria for choosing an optimal constant penalty have been recently proposed in [24, 26]. Although the optimal penalty parameter can significantly improve practical convergence, these methods make strong assumptions about the objective and constraint matrix. By introducing relaxation parameter, a method called relaxed ADMM uses past iterations to compute the current one. Instead of the fixed penalty parameter, Aybat and Iyengar [27] suggested using a penalty parameter sequence. Another different approach to the above methods is to accelerate ADMM by adaptively modifying the penalty parameter. As one of the adaptive ADMM methods, residual balancing (RB) [28] tunes penalty parameter to keep both residuals of similar magnitude. We borrow the idea from this method to further improve the convergence of ADMM combined with MPC.

In this paper, we utilize a-ADMM to efficiently solve the MPC problem, and the performance is verified by multiple simulation experiments. To be more specific, we transform the standard MPC problem into a distributed formulation and then take advantage of a-ADMM to solve this problem. We verified the method on an Unmanned Aerial Vehicle (UAV) control problem, and this mechanical control system includes 12 states and 4 control inputs [29]. We tested our method on different settings, i.e., the prediction horizon N ranges from 10 to 40. For each setting, MPC requires the solution of a QP with $(12+4)\times N$ variables and $(12+4)\times 2\times N$ inequality constraints. In this complex UAV control problem, a-ADMM algorithm achieves an absolute improvement of real-time response ability over conventional interior-point method and convergence performance over other ADMM variations.

The paper is organized as follows. Section 2 briefly introduces MPC, ADMM, and some ADMM variations. Section 3 describes the main procedures of the transformation from MPC to the distributed optimization problem and the application of a-ADMM in this specific problem. In Section 4, we compared the method with some baselines in the UAV control task. Finally, the conclusion is drawn and our future work is declared in Section 5. Appendix A describes how to formulate the QP problem of MPC in some details.

2. Backgrounds

2.1. Model Predictive Control. MPC is an optimal control approach which can handle multiple constraints on states and control inputs. Using a receding horizon control framework, MPC is robust to the model bias and external noises, which makes MPC fit well for highly complex multivariable industry processes.

In this paper, the following discrete-time linear time-invariant system is considered:

$$\mathcal{X}_{t+1} = A\mathcal{X}_t + B\mathcal{U}_t, \tag{1}$$

where $\mathcal{X}_t \in \mathbb{R}^n$ and $\mathcal{U}_t \in \mathbb{R}^m$ denote the state and control input, respectively, at time step t. $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ are the system and control matrices, respectively. For the regulation problem, at each time step t, MPC solves the finite horizon optimal control problem:

$$\min_{\mathcal{U}} \ \mathcal{X}_N^T P \mathcal{X}_N + \sum_{t=1}^{N-1} \left[\mathcal{X}_t^T Q \mathcal{X}_t + \mathcal{U}_t^T R \mathcal{U}_t\right], \tag{2a}$$

$$\text{s.t.} \quad \mathcal{X}_{t+k+1|t} = A\mathcal{X}_{t+k|t} + B\mathcal{U}_{t+k|t}, \quad k = 0, \ldots, N-1, \tag{2b}$$

$$\mathcal{X}_{\min} \le \mathcal{X}_{t+k|t} \le \mathcal{X}_{\max}, \quad k = 1, \ldots, N, \tag{2c}$$

$$\mathcal{U}_{\min} \le \mathcal{U}_{t+k|t} \le \mathcal{U}_{\max}, \quad k = 1, \ldots, N, \tag{2d}$$

$$\mathcal{X}_0 = \mathcal{X}_t, \tag{2e}$$

where N is the prediction horizon. $\mathcal{X}_{t+k|t}$ denotes the predicted state vector at time $t+k$, obtained by applying the input sequence $u_t, \ldots, u_{t+k-1}$. $\mathcal{X}_{\max} \in \mathbb{R}^n$ and $\mathcal{X}_{\min} \in \mathbb{R}^n$ define constraints on state variable $\mathcal{X}$, $\mathcal{U}_{\max} \in \mathbb{R}^m$ and $\mathcal{U}_{\min} \in \mathbb{R}^m$ define constraints on input variable $\mathcal{U}$. $P \in \mathbb{R}^{n\times n}$ and $Q \in \mathbb{R}^{n\times n}$ are stage and terminal matrices which penalize the state deviation at the end of the prediction horizon N and over the entire horizon, respectively. $R \in \mathbb{R}^{m\times m}$ is the cost matrix for the control inputs. Typically, $P \succcurlyeq 0$, $Q \succcurlyeq 0$, and $R \succ 0$.

In MPC, control inputs are obtained by solving open-loop optimal control problem (2a)–(2e) [10]. To be more specific, at each time step t, the optimization problem (2a)–(2e) is solved by taking the current state $\mathcal{X}_t$ as initial

state of the problem and generates a sequence of optimal control input. However, according to receding horizon framework of MPC, only the first element of the control input sequence is applied to the plant. The remaining solutions are culled, and optimal control problem with new initial state $\mathcal{X}_{t+1}$ will be solved again at the next time step $t+1$.

2.2. Alternating Direction Method of Multipliers. ADMM is a strong algorithm which can efficiently solve convex optimization problems, and it can decompose the global problem into several smaller and easy-to-solve local subproblems [30]. Because of its fast processing speed and good convergence performance, ADMM has received tremendous interest and approval for solving numerous problems in machine learning, statistics, and signal processing [17, 31–33].

ADMM can efficiently solve the optimization problems of the following standard form:

$$\min_{\mathcal{U}} \quad f(\mathcal{U}), \tag{3a}$$

$$\text{s.t.} \quad \mathcal{U} \in C, \tag{3b}$$

where f and C are convex. This problem can be rewritten in another form as

$$\min_{\mathcal{U},\mathcal{Z}} \quad f(\mathcal{U}) + g(\mathcal{Z}), \tag{4a}$$

$$\text{s.t.} \quad \mathcal{U} - \mathcal{Z} = 0, \tag{4b}$$

where g is the indicator function of C, $\mathcal{U}$ is primal variable, and $\mathcal{Z}$ is separate variable. The augmented Lagrangian is defined as

$$L_\rho(\mathcal{U}, \mathcal{Z}, c) = f(\mathcal{U}) + g(\mathcal{Z}) + \left(\frac{\rho}{2}\right)\left\|\mathcal{U} - \mathcal{Z} + c\right\|_2^2, \quad \rho > 0, \tag{5}$$

where ρ is the step size (or penalty parameter) and c is the scaled dual variable. The general augmented Lagrange multiplier method is to minimize L_ρ, w.r.t. $\mathcal{U}$ and $\mathcal{Z}$ at the same time. This is in difficulty and does not exploit the fact that the objective function is separable. To remedy this issue, ADMM decomposes the global problem (4) into two subproblems and minimizes $\mathcal{U}$ and $\mathcal{Z}$, respectively, with the following three steps:

$$\mathcal{U}^{k+1} := \arg\min\left(f(\mathcal{U}) + \frac{\rho}{2}\left\|\mathcal{U} - \mathcal{Z}^k + c^k\right\|_2^2\right), \tag{6a}$$

$$\mathcal{Z}^{k+1} := \arg\min\left(g(\mathcal{Z}) + \frac{\rho}{2}\left\|\mathcal{U}^{k+1} - \mathcal{Z} + c^k\right\|_2^2\right), \tag{6b}$$

$$c^{k+1} := c^k + \mathcal{U}^{k+1} - \mathcal{Z}^{k+1}. \tag{6c}$$

Under rather mild conditions, ADMM converges for any constant penalty parameter [34]. The convergence of ADMM is based on the primal and dual residuals:

$$\begin{aligned} r^k &= \mathcal{U}^k - \mathcal{Z}^k, \\ s^k &= -\rho\left(\mathcal{Z}^k - \mathcal{Z}^{k-1}\right). \end{aligned} \tag{7}$$

It has been observed that the algorithm approaches to the optimal solution when the primal and dual residuals approach to zero. Generally, the iterative process is terminated if

$$\left\|r^k\right\|_2 \le \sqrt{p}\varepsilon^{\text{abs}} + \varepsilon^{\text{rel}} \max\left\{\left\|\mathcal{U}^k\right\|_2, \left\|\mathcal{Z}^k\right\|_2\right\}, \tag{8a}$$

$$\left\|s^k\right\|_2 \le \sqrt{n}\varepsilon^{\text{abs}} + \varepsilon^{\text{rel}}\left\|(c\rho)^k\right\|_2, \tag{8b}$$

where $\varepsilon^{\text{abs}} > 0$ is absolute tolerance and $\varepsilon^{\text{rel}} > 0$ is relative tolerance. However, ρ has an immediate effect on the convergence element of the algorithm, and inappropriate tuning of this algorithm parameter may render the method moderative.

One way of accelerating the convergence properties of the algorithm is to utilize previous iterates when computing the following ones. This practical method is called relaxed Alternating Direction Method of Multipliers (r-ADMM) [35] that replaces $\mathcal{U}_{k+1}$ with $\alpha^k\mathcal{U}_{k+1} - (1-\alpha^k)\mathcal{Z}^k$ in the $\mathcal{Z}-$ and $c-$ updates (6a)–(6c), yielding

$$\mathcal{U}^{k+1} := \arg\min\left(f(\mathcal{U}) + \frac{\rho}{2}\left\|\mathcal{U} - \mathcal{Z}^k + c^k\right\|_2^2\right), \tag{9a}$$

$$\widetilde{\mathcal{U}}^{k+1} := \alpha^k\mathcal{U}^{k+1} - \left(1-\alpha^k\right)\mathcal{Z}^k, \tag{9b}$$

$$\mathcal{Z}^{k+1} := \arg\min\left(g(\mathcal{Z}) + \frac{\rho}{2}\left\|\mathcal{U}^{k+1} - \mathcal{Z} + c^k\right\|_2^2\right), \tag{9c}$$

$$c^{k+1} := c^k + \mathcal{U}^{k+1} - \mathcal{Z}^{k+1}. \tag{9d}$$

The parameter $\alpha^k \in (0, 2)$ is called the relaxation parameter. Note that letting $\alpha^k = 1$ for all k recovers the original ADMM iterations. Empirical studies have illustrated that overrelaxation, i.e., letting $\alpha^k > 1$, is often beneficial and the guideline $\alpha^k \in [1.5, 1.8]$ has been proposed [36].

Another approach for improving the convergence properties of the algorithm is to utilize different penalty parameters for each iteration. Adaptive Alternating Direction Method of Multipliers (a-ADMM) [28] is based on the following observation: increasing ρ^k strengthens the penalty term and leads to smaller primal residuals but larger dual ones; on the contrary, decreasing ρ^k results in smaller dual residuals and larger primal residuals. As both residuals must be small at convergence, it requires adjusting ρ to keep both primal residuals and dual residuals of the same magnitude. A simple and effective scheme for this purpose is as follows:

$$\rho^{k+1} := \begin{cases} \tau^{\text{incr}}\rho^k, & \text{if } \left\|r^k\right\|_2 > \mu\left\|s^k\right\|_2, \\ \dfrac{\rho^k}{\tau^{\text{decr}}}, & \text{if } \left\|s^k\right\|_2 > \mu\left\|r^k\right\|_2, \\ \rho^k, & \text{otherwise}, \end{cases} \tag{10}$$

where $\mu > 1$, $\tau^{\text{incr}} > 1$, and $\tau^{\text{decr}} > 1$. Using different penalty parameter ρ^k for each iteration k, a-ADMM can converge quickly and be insensitive to the initial penalty parameter. However, without a careful choice of μ, τ^{incr}, and τ^{decr}, this algorithm may fail to converge [28].

3. Fast MPC Based on Adaptive ADMM

3.1. Adaptive ADMM MPC. We focus on improving MPC by solving its QP problem with a-ADMM. In this work, the standard MPC problem is transformed into an equivalent QP problem which could be easily decomposed into several smaller local subproblems. And a-ADMM could be finally adopted to efficiently solve the resulting problem.

MPC problem ((2a)–(2e)) can be expressed as a QP problem when the system dynamic and constraints are linear and the cost function is quadratic [10]. Specifically, by substituting $\mathcal{X}_{t+k|t} = A^k \mathcal{X}_t + \sum_{j=0}^{k-1} A^j B \mathcal{U}_{t+k-1-j}$ into the cost function, the optimization problem equations (2a)–(2e) can be formulated as

$$\min_{\mathcal{U}} \quad \mathcal{U}^T H \mathcal{U} + \mathcal{X}_t^T C^T \mathcal{U} + \mathcal{X}_t^T Y \mathcal{X}_t, \tag{11a}$$

$$\text{s.t.} \quad G\mathcal{U} \le W + S\mathcal{X}_t, \tag{11b}$$

where $\mathcal{U} = [\mathcal{U}_t^T, \mathcal{U}_{t+1}^T, \ldots, \mathcal{U}_{t+N-1}^T]^T$ is the optimization vector and H, C, Y, G, W, and S are constant matrices that can be easily obtained offline. Note that H has to be positive definite for the convexity of this problem. The detailed process of this transformation is shown in Appendix A.

As the problem is determined by the current state $\mathcal{X}_t$, the implementation of MPC requires the online solution of problem ((11a) and (11b)) at each time step. Although off-the-shelf QP solvers, e.g., active-set methods [37] and interior-point methods [11], are available, it may demand significant computation effort in order to compute the input $\mathcal{U}_t$ online.

Fortunately, the QP problem (11a) and (11b) can be further transformed into ADMM standard form by introducing a split vector $\mathcal{Z}$ named slack vector (see [17]):

$$\min_{\mathcal{U},\mathcal{Z}} \quad f(\mathcal{U}) + g(\mathcal{Z}), \tag{12a}$$

$$\text{s.t.} \quad \mathcal{U} - \mathcal{Z} = 0, \tag{12b}$$

where $f(\mathcal{U}) = \mathcal{U}^T H \mathcal{U} + \mathcal{X}_t^T C^T \mathcal{U} + \mathcal{X}_t^T Y \mathcal{X}_t$ is the initial objective function with qualified domain and $g(\mathcal{Z}) = I(\mathcal{Z})$ is defined as the indicator function:

$$I(\mathcal{Z}) \coloneqq \begin{cases} 0, & \text{if } G\mathcal{Z} \le W + S\mathcal{X}_t, \\ +\infty, & \text{otherwise.} \end{cases} \tag{13}$$

The augmented Lagrangian is defined as

$$L_\rho(\mathcal{U}, \mathcal{Z}, c) = f(\mathcal{U}) + g(\mathcal{Z}) + \left(\frac{\rho}{2}\right) \left\| \mathcal{U} - \mathcal{Z} + c \right\|_2^2, \quad \rho > 0, \tag{14}$$

where ρ is the penalty parameter and c is the dual parameter. We propose to solve problem (11a) and (11b) with a-ADMM as follows:

$$\mathcal{U}^{k+1} \coloneqq \arg\min \left(\mathcal{U}^T H \mathcal{U} + \mathcal{X}_t^T C^T \mathcal{U} + \frac{\rho}{2} \left\| \mathcal{U} - \mathcal{Z}^k + \mathcal{C}^k \right\|_2^2 \right), \tag{15a}$$

$$\mathcal{Z}^{k+1} \coloneqq \arg\min \left(I(\mathcal{Z}) + \frac{\rho}{2} \left\| \mathcal{U}^{k+1} - \mathcal{Z} + \mathcal{C}^k \right\|_2^2 \right), \tag{15b}$$

$$\mathcal{C}^{k+1} \coloneqq \mathcal{C}^k + \mathcal{U}^{k+1} - \mathcal{Z}^{k+1}, \tag{15c}$$

where the penalty parameter ρ is a piecewise linear function as equation (10). As the optimizer $\mathcal{U}$ is only needed, the term involving Y is usually removed from problem (11a) and (11b). Since both residuals must be small at convergence, ρ is tuned to keep both residuals of similar magnitude.

3.2. Convergence Analysis of Adaptive ADMM-MPC. He et al. [28] proved that convergence is guaranteed for a-ADMM when either of the two following conditions is satisfied:

Condition 1 (bounded increasing):

$$\sum_{k=1}^{\infty} \left(\eta^k\right)^2 < \infty, \tag{16}$$

where $\eta^k = \sqrt{\max\{(\rho^k/\rho^{k-1}), 1\}} - 1$.

Condition 2 (bounded decreasing):

$$\sum_{k=1}^{\infty} \left(\theta^k\right)^2 < \infty, \tag{17}$$

where $\theta^k = \sqrt{\max\{(\rho^k - 1/\rho^k), 1\}} - 1$.

Condition 1 (Condition 2) suggests that the increasement (decreasement) of adaptive penalty parameter is bounded.

Based on the above analysis, the framework of a-ADMM algorithm is portrayed in Algorithm 1.

4. Simulation Experiment

We evaluated our approach on an UAV control task. The dynamic model of UAV used in this experiment is from [29] and can be expressed as follows:

$$m\ddot{x} = -u \sin\theta - \beta\dot{x}, \tag{18a}$$

$$m\ddot{y} = u \cos\theta \sin\phi - \beta\dot{y}, \tag{18b}$$

$$m\ddot{z} = u \cos\theta \cos\phi - mg - \beta\dot{z}, \tag{18c}$$

$$\ddot{\theta} = \tilde{t}_\theta, \tag{18d}$$

$$\ddot{\phi} = \tilde{t}_\phi, \tag{18e}$$

$$\ddot{\psi} = \tilde{t}_\psi, \tag{18f}$$

where x, y, and z denote the position of the UAV and θ, ϕ, and ψ denote the rotation angle of the UAV around the Cartesian coordinate axis. The damping coefficient β takes into account the actual friction effect. This dynamic model has 12 states $\mathcal{X} \in \mathbb{R}^{12}$ and 4 control inputs $\mathcal{U} \in \mathbb{R}^4$. For the purpose of designing a appropriate MPC controller for UAV, the nonlinear dynamic model ((18a)–(18f)) is linearized at the equilibrium point (see [29]):

$$\begin{cases} \mathcal{X}_{t+1} = A\mathcal{X}_t + B\mathcal{U}_t, \\ \mathcal{Y}_t = C\mathcal{X}_t, \end{cases} \tag{19}$$

where $\mathcal{X}_t = [\theta, \phi, \psi, x, y, z, \dot{\theta}, \dot{\phi}, \dot{\psi}, \dot{x}, \dot{y}, \dot{z}]' \in \mathbb{R}^{12}$ is the state vector, $\mathcal{U}_t = [u, \tilde{t}_\theta, \tilde{t}_\phi, \tilde{t}_\psi] \in \mathbb{R}^4$ is the input vector, and $\mathcal{Y}_t \in \mathbb{R}^{12}$ is the output vector. Assuming that the UAV system is completely measurable, and the observation matrix C is a 12×12 identity matrix. The coefficient matrices A and B are as follows:

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0.1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0.1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0.1 & 0 & 0 & 0 \\ 0.0488 & 0 & 0 & 1 & 0 & 0 & 0.0016 & 0 & 0 & 0.0992 & 0 & 0 \\ 0 & -0.0488 & 0 & 0 & 1 & 0 & 0 & -0.0016 & 0 & 0 & 0.0992 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0.0992 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0.9734 & 0 & 0 & 0 & 0 & 0 & 0.0488 & 0 & 0 & 0.9846 & 0 & 0 \\ 0 & -0.9734 & 0 & 0 & 0 & 0 & 0 & -0.0488 & 0 & 0 & 0.9846 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.9846 \end{bmatrix},$$

$$B = \begin{bmatrix} 0 & -0.0726 & 0 & 0.0726 \\ -0.0726 & 0 & 0.0726 & 0 \\ -0.0152 & 0.0152 & -0.0152 & 0.0152 \\ 0 & -0.0006 & -0.0000 & 0.0006 \\ 0.0006 & 0 & -0.0006 & 0 \\ 0.0106 & 0.0106 & 0.0106 & 0.0106 \\ 0 & -1.4512 & 0 & 1.4512 \\ -1.4512 & 0 & 1.4512 & 0 \\ -0.3049 & 0.3049 & -0.3049 & 0.3049 \\ 0 & -0.0236 & 0 & 0.0236 \\ 0.0236 & 0 & -0.0236 & 0 \\ 0.2107 & 0.2107 & 0.2107 & 0.2107 \end{bmatrix}. \tag{20}$$

The baseline methods we compared in our experiments are as follows.

4.1. Interior-Point Method (IPM). IPM is the mostly used QP solver modeling the problem constraints as parametrized penalty functions which also referred to as barrier functions. At each iteration, an unconstrained optimization problem is solved for varying barrier function until the optimum is achieved.

4.2. qpOASES: a Parametric Active-Set Algorithm for Quadratic Programming. qpOASES is a very used QP solver which is applicable when numerous related QPs are solved sequentially, by exploiting the geometric property of state space [14].

4.3. Alternating Direction Method of Multipliers (ADMM). ADMM is a strong algorithm for solving convex optimization problems [16, 17] which decomposes the global

a-ADMM algorithm
Initialization: optimization variables $\mathcal{U}^0$, $\mathcal{Z}^0$, $\mathcal{C}^0$
While not converge by stopping criteria ((8a) and (8b)) and k¡maxiter **do**
$\mathcal{U}^{k+1} \coloneqq \arg\min(\mathcal{U}^T H \mathcal{U} + \mathcal{X}_t^T C^T \mathcal{U} + \rho/2 \| \mathcal{U} - \mathcal{Z}^k + \mathcal{C}^k \|_2^2)$
$\mathcal{Z}^{k+1} \coloneqq \arg\min(I(\mathcal{Z}) + \rho/2 \| \mathcal{U}^{k+1} - \mathcal{Z} + \mathcal{C}^k \|_2^2)$
$\mathcal{C}^{k+1} \coloneqq \mathcal{C}^k + \mathcal{U}^{k+1} - \mathcal{Z}^{k+1}$
if $\|r^k\|_2 > \|s^k\|_2$
$\rho^{k+1} = \tau^{\text{incr}} \rho^k$
else if $\|s^k\|_2 > \|r^k\|_2$
$\rho^{k+1} = \rho^k / \tau^{\text{decr}}$
else $\rho^{k+1} = \rho^k$
end if
$k = k + 1$
end while
Output: optimal control input $\mathcal{U}^*$

Algorithm 1: a-ADMM algorithm framework.

problem into several smaller and easy-to-solve local subproblems [30].

4.4. Relaxed Alternating Direction Method of Multipliers (r-ADMM). r-ADMM is one of the variants of ADMM which introduces relaxation parameter α to relax the update of the optimal solution. Empirical studies have illustrated that overrelaxation is beneficial to the convergence [36].

We first fixed the prediction horizon $N = 10$ to find the optimal setting of ADMM and r-ADMM: the penalty parameter ρ and the relaxation parameter α. Figure 1 gives the results of different values of parameters. As can be seen, the average number of iterations depends significantly on the penalty parameter ρ and the relaxation parameter α. Based on this result, we simply set $\rho = 0.7$ and $\alpha = 1.5$ in the following experiments.

Similarly, we used the same control task to find the optimal setting of a-ADMM. According to the theory of a-ADMM, the relaxation parameter α is fixed and can be set $\alpha = 1.5$ based on previous results. However, different from ADMM, the penalty parameter ρ of a-ADMM is a piecewise linear function as equation (10) influenced by three other parameters, i.e., μ, τ^{incr}, and τ^{decr}. Figure 2 gives the results of different values of μ, τ^{incr}, and τ^{decr} on the UAV control task when the prediction horizon $N = 10$. Based on the lowest average iterations of each setting, we set $\mu = 10$, $\tau^{\text{incr}} = 2$, and $\tau^{\text{decr}} = 10$ in the following experiments, and the optimization variables $\mathcal{U}^0$, $\mathcal{Z}^0$, and $\mathcal{C}^0$ are all initialized to a zero vector, so that the initial strategy of optimization variables does not affect the performance comparison of these algorithms (IPM, qpOASES, ADMM, r-ADMM, and a-ADMM). The parameters of these algorithms are listed in Table 1.

The feedback loop controls the quadcopter to land on a platform located at coordinate position $(0, 0, 0)$. As shown in Figure 3, the closed-loop performance (including the feasibility and stability) of ADMM and a-ADMM is exactly the same. The reason is that, with the same absolute tolerance ε^{abs} and relative tolerance ε^{rel}, ADMM and a-ADMM converge to the same optimal solution leading to the same closed-loop performance.

Figure 4 gives our real-time performance of each method on the UAV control task. This figure shows the cumulative running time of our method and other baselines with the prediction horizon N ranging from 10 to 40. From the simulation results, it can be seen that, combined with MPC, ADMM and its variants significantly outperform custom QP solver, IPM, and qpOASES. With the increment of the task complexity, the running time of IPM and qpOASES increases obviously, while that of ADMM, r-ADMM, and a-ADMM basically remains unchanged.

As shown in Figure 4, there is not much difference in the cumulative running time of several variants of the ADMM algorithm. In this part, we aim to further investigate the difference in the cumulative number of iterations between them. Figure 5 portrays the cumulative iteration number of each method when the prediction horizon N ranging from 10 to 40. Apparently, although ADMM and its variations have the similar real-time performance, a-ADMM significantly outperforms all other baselines in terms of convergence performance under all the task settings. qpOASES proposes to move on a straight line in the state space when transitioning from the "old" QP problem to the "new" one. The reason why the cumulative iteration number of qpOASES stays stable after several time steps is mainly because the state of quadcopter has converged to the objective region, and the adjacent QP problem is quite similar.

5. Conclusions

In this paper, we utilize a-ADMM to accelerate the solution of MPC. We firstly transform the standard MPC problem into a distributed formulation and then take advantage of a-ADMM to solve this problem. We verified this method on an UAV control problem and compared a-ADMM with IPM, ADMM, and r-ADMM under different control settings, i.e., the prediction horizon N ranging from 10 to 40. The experimental results have shown that a-ADMM can greatly improve the effectiveness of MPC. As a future and

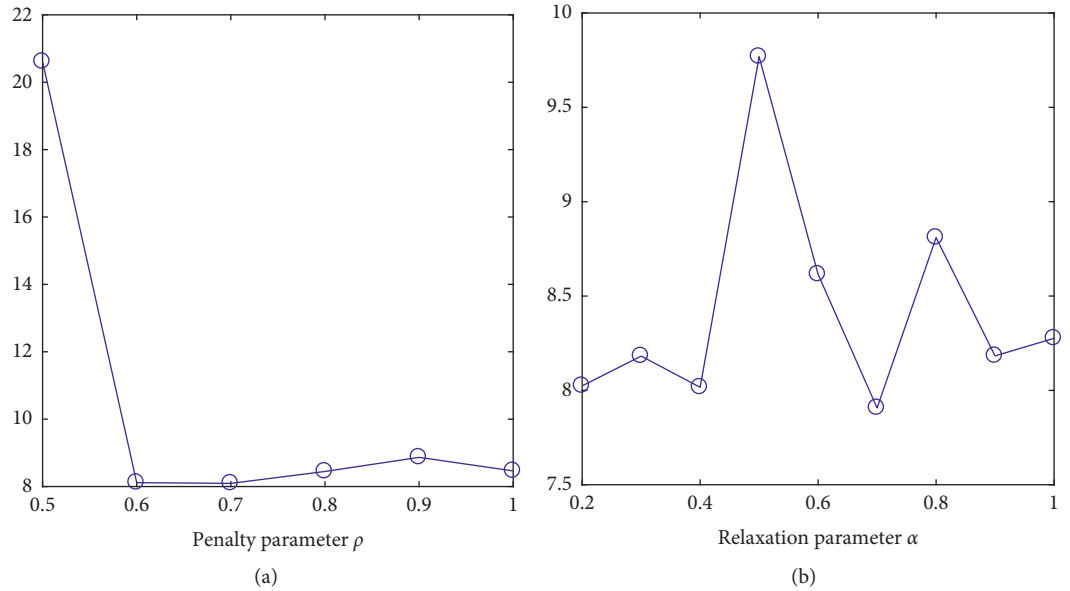

Figure 1: Effect of penalty parameter ρ of ADMM (a) and relaxation parameter α of r-ADMM (b) in terms of average number of iterations taken to converge. For each setting, algorithms are performed 50 times on the UAV control task ($N = 10$) with random initial states.

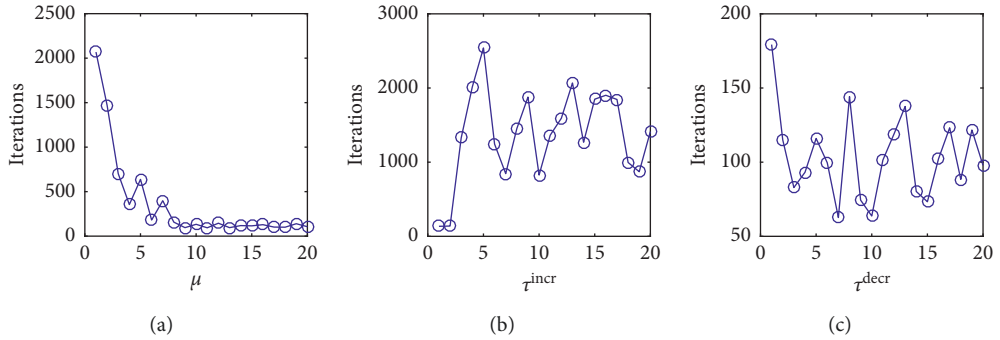

Figure 2: Effect of μ (a), τ^{incr}, (b) and τ^{decr} (c) in terms of average number of iterations taken to converge. For each setting, a-ADMM is performed 50 times on the UAV control task ($N = 10$) with random initial states.

Table 1: List of algorithmic parameters.

Parameters	IPM	qpOASES	ADMM	r-ADMM	a-ADMM
ρ	—	—	0.7	0.7	Adaptive
$\mathcal{U}^0$	0	0	0	0	0
$\mathcal{Z}^0$	0	0	0	0	0
$\mathcal{C}^0$	0	0	0	0	0
α	—	—	1	1.5	1
ϵ^{abs}	—	—	$1e-2$	$1e-2$	$1e-2$
ϵ^{rel}	—	—	$1e-4$	$1e-4$	$1e-4$
μ	—	—	—	—	10
τ^{incr}	—	—	—	—	2
τ^{decr}	—	—	—	—	10
Maxiter	1000	1000	1000	1000	1000

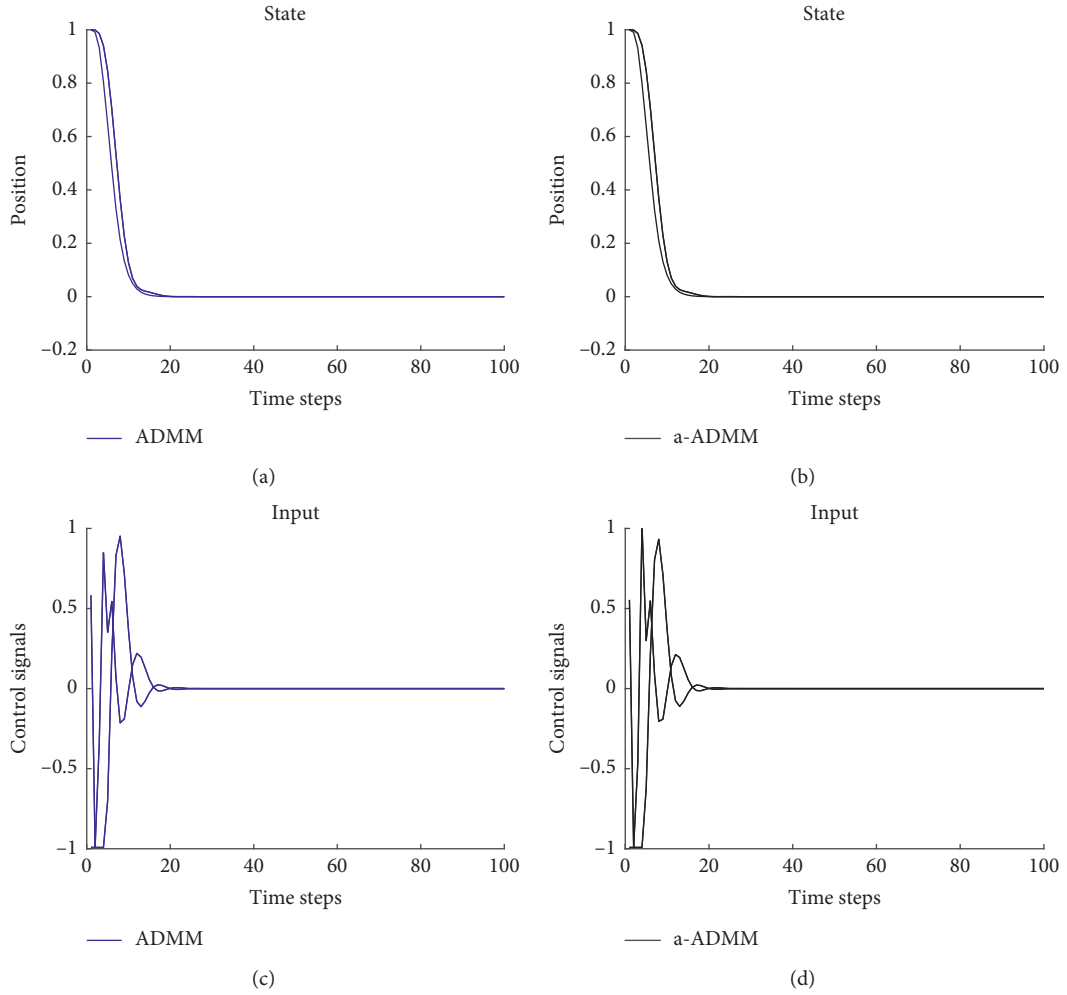

Figure 3: Closed-loop performance (running time on y-axis) versus number of time steps (time steps on x-axis) of ADMM and a-ADMM on UAV control task with $N = 10$.

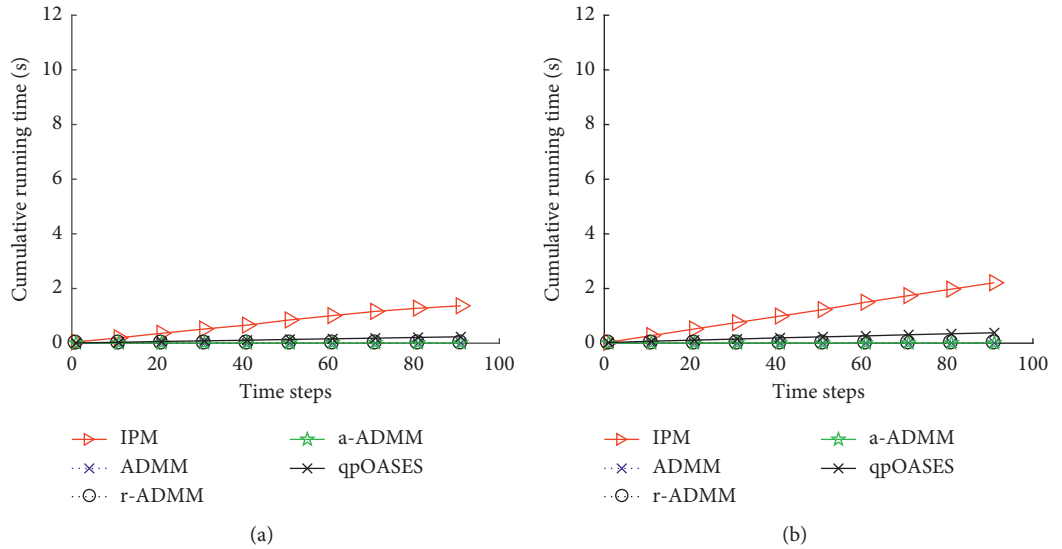

Figure 4: Continued.

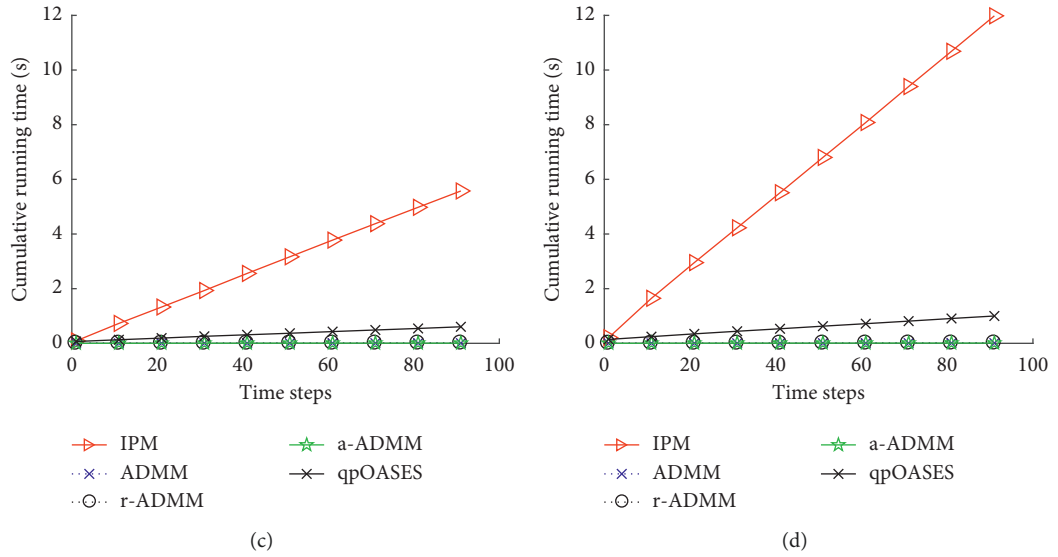

Figure 4: Real-time performance (running time on y-axis) versus number of time steps (time steps on x-axis) of IPM, qpOASES, ADMM, r-ADMM, and a-ADMM on UAV control task with N ranging from 10 to 40: (a) $N = 10$, (b) $N = 20$, (c) $N = 30$, and (d) $N = 40$.

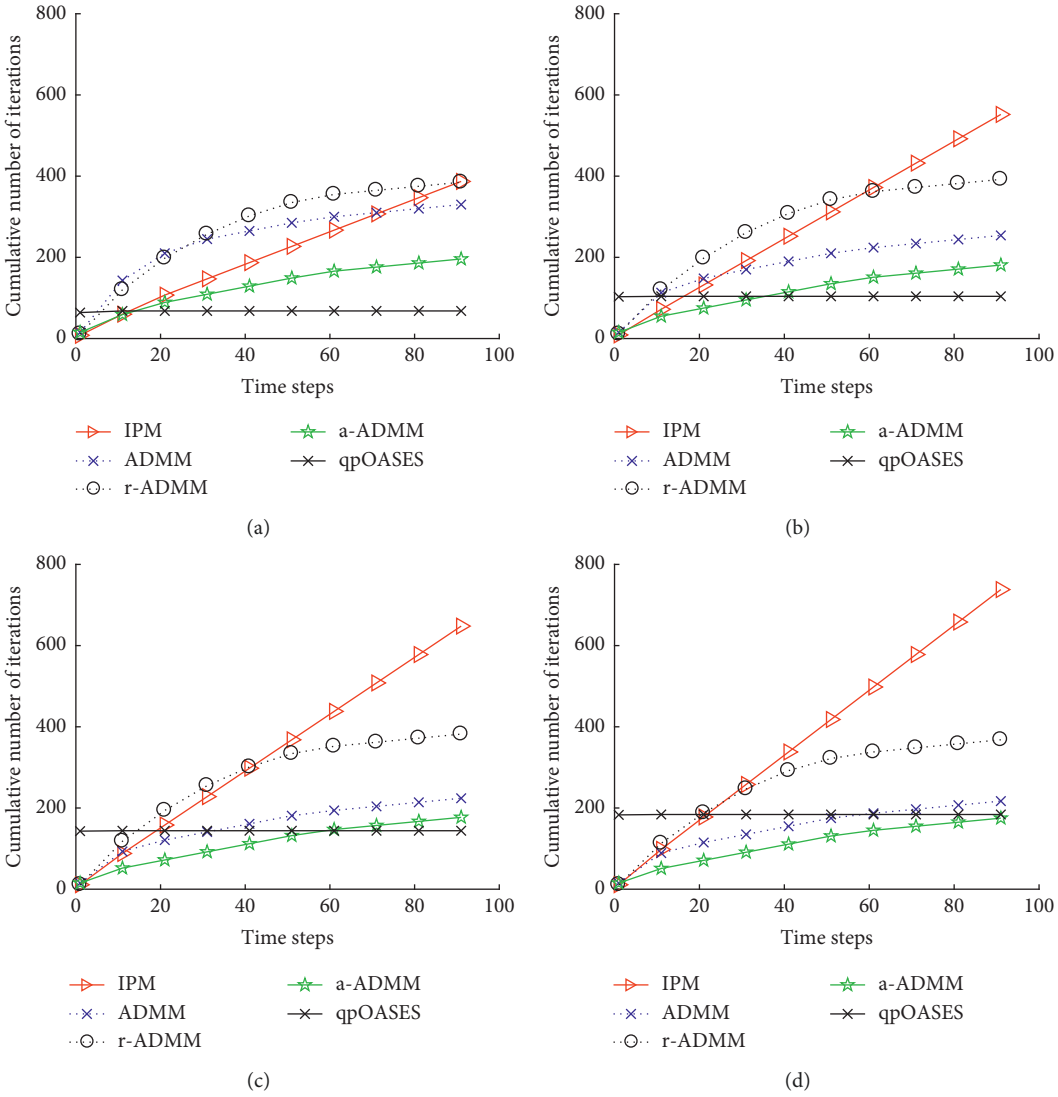

Figure 5: Convergence performance (cumulative number of iterations on y-axis) versus number of time steps (time steps on x-axis) of IPM, qpOASES, ADMM, r-ADMM, and a-ADMM on UAV control task with N ranging from 10 to 40: (a) $N = 10$, (b) $N = 20$, (c) $N = 30$, and (d) $N = 40$.

noteworthy research direction, one might provide better scheme for adaptively adjusting penalty parameter considering more complex constraints.

Appendix

A. Equivalent Representation of $\mathcal{X}_{t+k}$

The linear MPC problem can be expressed as QP problem by substituting $X_{t+k} = A^k\mathcal{X}_t + \sum_{j=0}^{k-1} A^j B\mathcal{U}_{t+k-1-j}$ into cost function. Firstly, we verified that $\mathcal{X}_{t+k}$ has an equivalent representation based on the following transformation:

$$\begin{aligned}
\mathcal{X}_{t+1} &= A\mathcal{X}_t + B\mathcal{U}_t,\\
\mathcal{X}_{t+2} &= A\mathcal{X}_{t+1} + B\mathcal{U}_{t+1}\\
&= A\left(A\mathcal{X}_t + B\mathcal{U}_t\right) + B\mathcal{U}_{t+1}\\
&= A^2\mathcal{X}_t + AB\mathcal{U}_t + B\mathcal{U}_{t+1},\\
\mathcal{X}_{t+3} &= A\mathcal{X}_{t+2} + B\mathcal{U}_{t+2}\\
&= A\left(A^2\mathcal{X}_t + AB\mathcal{U}_t + B\mathcal{U}_{t+1}\right) + B\mathcal{U}_{t+2}\\
&= A^3\mathcal{X}_t + A^2B\mathcal{U}_t + AB\mathcal{U}_{t+1} + B\mathcal{U}_{t+2}\\
&\cdots\\
&\cdots\\
\mathcal{X}_{t+k} &= A^k\mathcal{X}_t + \sum_{j=0}^{k-1} A^j B\mathcal{U}_{t+k-1-j}.
\end{aligned} \tag{A.1}$$

The conclusion can be deduced, $\mathcal{X}_{t+k} = A^k\mathcal{X}_t + \sum_{j=0}^{k-1} A^j B\mathcal{U}_{t+k-1-j}$. Having disposed of this preliminary step, we begin to prove the equivalent form of the objective function in Appendix B.

B. The Transformation of the Objective Function

Secondly, we transform the objective function into its equivalent form. As previously mentioned, the objective function can be casted into $\mathcal{U}^T H\mathcal{U} + 2\mathcal{X}_t^T C^T\mathcal{U} + \mathcal{X}_t^T Y\mathcal{X}_t$ by substituting $\mathcal{X}_{t+k} = A^k\mathcal{X}_t + \sum_{j=0}^{k-1} A^j B\mathcal{U}_{t+k-1-j}$, and the formula is derived as follows:

$$\begin{aligned}
\mathcal{X}_{t+k} &= A^k\mathcal{X}_t + \sum_{j=0}^{k-1} A^i B\mathcal{U}_{t+k-1-i}, \quad k = 1, 2, \ldots, N\\
&= A^k\mathcal{X}_t + A^{k-1}B\mathcal{U}_t + A^{k-2}B\mathcal{U}_{t+1} + \cdots + AB\mathcal{U}_{t+N-2}\\
&\quad + B\mathcal{U}_{t+N-1},
\end{aligned} \tag{A.2}$$

Set $T_k = [A^{k-1}B, A^{k-2}B, \ldots, B]$ and $\mathcal{U} = [\mathcal{U}_t, \mathcal{U}_{t+1}, \ldots, \mathcal{U}_{t+N-1}]$, and $A^{k-1}B\mathcal{U}_t + A^{k-2}B\mathcal{U}_{t+1} + \cdots + AB\mathcal{U}_{t+N-2} + B\mathcal{U}_{t+N-1}$ can be expressed as $T_k\mathcal{U}$ consequently,

$$\begin{aligned}
\mathcal{X}_{t+k} &= A^k\mathcal{X}_t + T_k\mathcal{U},\\
\mathcal{X}_{t+k}^T &= \mathcal{X}_t^T\left(A^k\right)^T + \mathcal{U}^T T_k^T,\\
\mathcal{X}_{t+k}^T Q\mathcal{X}_{t+k} &= \left(\mathcal{X}_t^T\left(A^k\right)^T + \mathcal{U}^T T_k^T\right)Q\left(A^k\mathcal{X}_t + T_k\mathcal{U}\right)\\
&= \left(\mathcal{X}_t^T\left(A^k\right)^T Q + \mathcal{U}^T T_k^T Q\right)\left(A^k\mathcal{X}_t + T_k\mathcal{U}\right)\\
&= \mathcal{X}_t^T\left(A^k\right)^T QA^k\mathcal{X}_t + \mathcal{X}_t^T\left(A^k\right)^T QT_k\mathcal{U} + \mathcal{U}^T T_k^T QA^k\mathcal{X}_t + \mathcal{U}^T T_k^T QT_k\mathcal{U}\\
&= \mathcal{X}_t^T\left(A^k\right)^T QA^k\mathcal{X}_t + 2\mathcal{X}_t^T\left(A^k\right)^T QT_k\mathcal{U} + \mathcal{U}^T T_k^T QT_k\mathcal{U}\\
&= \mathcal{X}_t^T y\mathcal{X}_t + 2\mathcal{X}_t^T c\mathcal{U} + U^T h\mathcal{U},
\end{aligned} \tag{A.3}$$

where

$$\begin{cases}
y = \left(A^k\right)^T QA^k,\\
c = \left(A^k\right)^T QT_k,\\
h = T_k^T QT_k,
\end{cases} \tag{A.4}$$

In a similar way, $\mathcal{X}_N^T P\mathcal{X}_N$ can be rewritten and finally obtained as

$$J\left(\mathcal{U}, \mathcal{X}_t\right) = \mathcal{U}^T H\mathcal{U} + 2\mathcal{X}_t^T C^T\mathcal{U} + \mathcal{X}_t^T Y\mathcal{X}_t, \tag{A.5}$$

where

$$\begin{aligned}
Y &= \sum_{k=0}^{N-1}\left(A^k\right)^T QA^k + \left(A^N\right)^T PA^N,\\
C &= \sum_{k=0}^{N-1}\left(A^k\right)^T QT_k + \left(A^N\right)^T PT^N,\\
H &= \sum_{k=0}^{N-1} T_k^T QT_k + T_N^T PT_N + M,\\
\mathbf{M} &= \begin{pmatrix} R & & \\ & \ddots & \\ & & R \end{pmatrix}.
\end{aligned} \tag{A.6}$$

C. The Transformation of Constraints

The last part is about the transformation of constraints, and the formula is derived as follows:

$$\begin{aligned} \mathcal{X}_{\min} \le \mathcal{X}_{t+k|t} \le \mathcal{X}_{\max}, \quad k = 0, \ldots, N-1, \\ \mathcal{U}_{\min} \le \mathcal{U}_{t+k|t} \le \mathcal{U}_{\max}, \quad k = 0, \ldots, N-1. \end{aligned} \tag{A.7}$$

By substituting $\mathcal{X}_{t+k} = A^k \mathcal{X}_t + \sum_{j=0}^{k-1} A^j B \mathcal{U}_{t+k-1-j}$, the constraints are equivalent to $G\mathcal{U} \le W + S\mathcal{X}_t$ through the following process:

$$\begin{cases} T_k = \left[A^{k-1}B, A^{k-2}B, \ldots, B\right], \\ \mathcal{X}_{t+k} = A^k \mathcal{X}_t + T_k \mathcal{U} \\ \mathcal{U}_k = \left[\mathcal{U}_t, \mathcal{U}_{t+1}, \ldots, \mathcal{U}_{t+N-1}\right], \end{cases} \tag{A.8}$$

$$\mathcal{X}_{\min} \le \mathcal{X}_{t+k} \le \mathcal{X}_{\max}. \tag{A.9}$$

By substituting formula (A.8) into formula (A.9), we obtain

$$\mathcal{X}_{\min} \le A^k \mathcal{X}_t + T_k \mathcal{U} \le \mathcal{X}_{\max}. \tag{A.10}$$

The solution of $\mathcal{U}$ is obtained as follows:

$$T_k^{-1}\left(\mathcal{X}_{\min} - A^k \mathcal{X}_t\right) \le \mathcal{U} \le T_k^{-1}\left(\mathcal{X}_{\max} - A^k \mathcal{X}_t\right). \tag{A.11}$$

That is,

$$\begin{cases} \mathcal{U}_k \le T_k^{-1} \mathcal{X}_{\max} - T_k^{-1} A^k \mathcal{X}_t, \\ -\mathcal{U}_k \le -T_k^{-1} \mathcal{X}_{\min} + T_k^{-1} A^k \mathcal{X}_t. \end{cases} \tag{A.12}$$

Formula (A.12) is equivalent to

$$G\mathcal{U} \le W + S\mathcal{X}_t, \tag{A.13}$$

where

$$\begin{cases} G = [1, -1]^T, \\ W = T_k^{-1} \otimes \left[\mathcal{X}_{\max}, \mathcal{X}_{\min}\right]^T, \\ E = \left[T_k^{-1} A^k, T_k^{-1} A^k\right]^T. \end{cases} \tag{A.14}$$

Data Availability

We use simulation data, and our model and related hyperparameter are provided in our paper.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was jointly supported by the National Natural Science Foundation of China (11871077), the Natural Science Foundation of Anhui Province (1808085MA04), and the Natural Science Foundation of Department of Education of Anhui Province (KJ2017A362).

References

[1] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3–20, 2002.

[2] Y.-G. Xi, D.-W. Li, and S. Lin, "Model predictive control—status and challenges," *Acta Automatica Sinica*, vol. 39, no. 3, pp. 222–236, 2013.

[3] M. Burger, M. van den Berg, A. Hegyi, B. De Schutter, and J. Hellendoorn, "Considerations for model-based traffic control," *Transportation Research Part C: Emerging Technologies*, vol. 35, pp. 1–19, 2013.

[4] K. Lu, P. P. Du, J. D. Cao, Q. Zou, T. He, and W. Huang, "A novel traffic signal split approach based on explicit model predictive control," *Mathematics and Computers in Simulation*, vol. 155, pp. 105–114, 2019.

[5] K. Samir, C. Patricio, R. Vargas, U. Ammann, and J. Rodriguez, "Model predictive control–a simple and powerful method to control power converters," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 6, pp. 1826–1838, 2009.

[6] L. Zhang, M. Pan, and S. Quan, "Model predictive control of water management in PEMFC," *Journal of Power Sources*, vol. 180, no. 1, pp. 322–329, 2008.

[7] G. Valencia-Palomo, J. A. Rossiter, and F. R. López-Estrada, "Improving the feed-forward compensator in predictive control for setpoint tracking," *ISA Transactions*, vol. 53, no. 3, pp. 755–766, 2014.

[8] G. Valencia-Palomo and J. A. Rossiter, "Efficient suboptimal parametric solutions to predictive control for PLC applications," *Control Engineering Practice*, vol. 19, no. 7, pp. 732–743, 2011.

[9] B. Huyck, J. De Brabanter, B. De Moor, J. F. Van Impe, and F. Logist, "Online model predictive control of industrial processes using low level control hardware: a pilot-scale distillation column case study," *Control Engineering Practice*, vol. 28, pp. 34–48, 2014.

[10] A. Bemporad, "Model predictive control design: new trends and tools," in *Proceedings of the 45th IEEE Conference on Decision and Control*, pp. 6678–6683, San Diego, CA, USA, December 2006.

[11] A. Domahidi, A. U. Zgraggen, M. N. Zeilinger, M. Morari, and C. N. Jones, "Efficient interior point methods for multistage problems arising in receding horizon control," in *Proceedings of the 2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pp. 668–674, Maui, HI, USA, December 2012.

[12] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: an operator splitting solver for quadratic programs," in *Proceedings of the UKACC 12th International Conference on Control*, p. 339, Sheffield, UK, September 2018.

[13] F. A. Potra and S. J. Wright, "Interior-point methods," *Journal of Computational and Applied Mathematics*, vol. 124, no. 1-2, pp. 281–302, 2000.

[14] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "qpOASES: a parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, vol. 6, no. 4, pp. 327–363, 2014.

[15] P. D. Christofides, R. Scattolini, D. Muñoz de la Peña, and J. Liu, "Distributed model predictive control: a tutorial review and future research directions," *Computers & Chemical Engineering*, vol. 51, pp. 21–41, 2013.

[16] V. Sindhwani, R. Roelofs, and M. Kalakrishnan, "Sequential operator splitting for constrained nonlinear optimal control," in *Proceedings of the American Control Conference 2017*, pp. 4864–4871, Seattle, WA, USA, May 2017.

[17] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[18] J. M. Bioucas-Dias and M. A. T. Figueiredo, "Alternating direction algorithms for constrained sparse regression: application to hyperspectral unmixing," in *Proceedings of the 2nd Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing*, pp. 1–4, Reykjavik, Iceland, June 2010.

[19] M. A. T. Figueiredo and J. M. Bioucas-Dias, "Restoration of poissonian images using alternating direction optimization," *IEEE Transactions on Image Processing*, vol. 19, no. 12, pp. 3133–3145, 2010.

[20] C. Zhao, R. Wang, and G. Wen, "Better and faster, when ADMM meets CNN: compressive-sensed image reconstruction," in *Advances in Multimedia Information Processing—PCM 2017*, Springer, Cham, Switzerland, 2017.

[21] A. Samaneh, F. Jiashi, J. Stefanie, and D. Trevor, "Auxiliary image regularization for deep cnns with noisy labels," 2015, http://arxiv.org/abs/1511.07069.

[22] J. H. R. Chang, C. L. Li, B. Poczos, B. V. K. Vijaya Kumar, and A. C. Sankaranarayanan, "One network to solve them all—solving linear inverse problems using deep projection models," in *Proceedings of the 2017 IEEE International Conference on Computer Vision*, Venice, Italy, October 2017.

[23] D. Boley, "Local linear convergence of the alternating direction method of multipliers on quadratic or linear programs," *SIAM Journal on Optimization*, vol. 23, no. 4, pp. 2183–2207, 2013.

[24] A. U. Raghunathan and S. D. Cairano, "Alternating direction method of multipliers for strictly convex quadratic programs: optimal parameter selection," in *Proceedings of the American Control Conference*, pp. 4324–4329, Washington, DC, USA, June 2014.

[25] A. U. Raghunathan and S. D. Cairano, "Optimal step-size selection in alternating direction method of multipliers for convex quadratic programs and model predictive control," in *Proceedings of the Symposium on Mathematical Theory of Networks and Systems*, pp. 807–814, Beer Sheva, Israel, June 2014.

[26] E. Ghadimi, A. Teixeira, I. Shames, and M. Johansson, "Optimal parameter selection for the alternating direction method of multipliers (ADMM): quadratic problems," *IEEE Transactions on Automatic Control*, vol. 60, no. 3, pp. 644–658, 2015.

[27] N. S. Aybat and G. Iyengar, "An alternating direction method with increasing penalty for stable principal component pursuit," *Computational Optimization and Applications*, vol. 61, no. 3, pp. 635–668, 2015.

[28] B. S. He, H. Yang, and S. L. Wang, "Alternating direction method with self-adaptive penalty parameters for monotone variational inequalities," *Journal of Optimization Theory and Applications*, vol. 106, no. 2, pp. 337–356, 2000.

[29] A. Bemporad, C. A. Pascucci, and C. Rocchi, "Hierarchical and hybrid model predictive control of quadcopter air vehicles," *IFAC Proceedings Volumes*, vol. 42, no. 17, pp. 14–19, 2009.

[30] P. Giselsson and S. Boyd, "Linear convergence and metric selection for douglas-rachford splitting and ADMM," *IEEE Transactions on Automatic Control*, vol. 62, no. 2, pp. 532–544, 2017.

[31] N. Liu and J. Wang, "Energy sharing for interconnected microgrids with a battery storage system and renewable energy sources based on the alternating direction method of multipliers," *Applied Sciences*, vol. 8, no. 4, p. 590, 2018.

[32] L. Wang, Y. Chen, F. Lin, Y. Chen, F. Yu, and Z. Cai, "Impulse noise denoising using total variation with overlapping group sparsity and Lp-pseudo-norm shrinkage," *Applied Sciences*, vol. 8, no. 11, p. 2317, 2018.

[33] C. Manss and D. Shutin, "Global-entropy driven exploration with distributed models under sparsity constraints," *Applied Sciences*, vol. 8, no. 10, p. 1722, 2018.

[34] G. Euhanna, T. André, S. Iman, and J. Mikael, "Optimal parameter selection for the alternating direction method of multipliers (ADMM): quadratic problems," *IEEE Transactions on Automatic Control*, vol. 60, no. 3, pp. 644–658, 2015.

[35] J. Eckstein and D. P. Bertsekas, "On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators," *Mathematical Programming*, vol. 55, no. 1–3, pp. 293–318, 1992.

[36] J. Eckstein and M. C. Ferris, "Operator-splitting methods for monotone affine variational inequalities, with a parallel application to optimal control," *INFORMS Journal on Computing*, vol. 10, no. 2, pp. 218–235, 1998.

[37] K. Scheinberg, "An efficient implementation of an active set method for SVMs," *Journal of Machine Learning Research*, vol. 7, pp. 2237–2257, 2006.