Alternating Direction Method of Multipliers for Constrained Iterative LQR in Autonomous Driving

Jun Ma, Zilong Cheng, Xiaoxue Zhang, and Masayoshi Tomizuka

Abstract—In the context of autonomous driving, the iterative linear quadratic regulator (iLQR) is known to be an efficient approach to deal with the nonlinear vehicle model in motion planning problems. Particularly, the constrained iLQR algorithm has shown noteworthy advantageous outcomes of computation efficiency in achieving motion planning tasks under general constraints of different types. However, the constrained iLQR methodology requires a feasible trajectory at the first iteration as a prerequisite when the logarithmic barrier function is used. Also, the methodology leaves open the possibility for incorporation of fast, efficient, and effective optimization methods (i.e., fast-solvers) to further speed up the optimization process such that the requirements of real-time implementation can be successfully fulfilled. In this paper, a well-defined and commonly-encountered motion planning problem is formulated under nonlinear vehicle dynamics and various constraints, and the alternating direction method of multipliers (ADMM) is utilized to determine the optimal control actions leveraging the iLQR. With this development, the approach is able to circumvent the feasibility requirement of the trajectory at the first iteration. An illustrative example of motion planning for autonomous vehicles is then investigated with different driving scenarios taken into consideration, and a noteworthy achievement high computation efficiency is attained with the proposed development. Comparing with the constrained iLQR algorithm based on the logarithmic barrier function, our proposed method reduces the average computation time by 31.93%, 38.52%, and 44.57% in the three scenarios; compared with the optimization solver IPOPT, our proposed method reduces the average computation time by 46.02%, 53.26%, and 88.43% in the three scenarios. As a result, real-time computation and implementation can be achieved through our proposed framework, and thus it provides additional safety to the on-road driving tasks.

Index Terms—Autonomous driving, iterative linear quadratic regulator (iLQR), differential dynamic programming (DDP), alternating direction method of multipliers (ADMM), motion planning, model predictive control (MPC), nonlinear system, non-convex optimization.

I. INTRODUCTION

Due to the rapid increase in traffic density, vehicle safety has become a primary concern in modern intelligent transportation systems. As one of the promising approaches to relieve traffic problems and enhance driving safety, autonomous vehicles have demonstrated great potentials in the current vehicle technology [1], [2]. Consequently, substantial efforts have been devoted to research on autonomous driving in recent years [3]–[7]. Particularly, as one of the core areas in autonomous driving, motion planning has attracted the attention of several disciplines, where it aims to plan a feasible trajectory that conforms to the requirements in terms of obstacle avoidance, energy consumption, traveling time, etc. In fact, there are a number of traditionally challenging dynamic motion planning problems in transportation, which have long threads of research literature [8], [9]. However, they have readily stood to benefit tremendously from the recent advances in optimization and artificial intelligence [10]–[12]. Along with this line, the planning methods in the continuous space generally include the learning-based method and the optimization-based method. For the learning-based method, autonomous vehicles can continuously improve their proficiency from the outcomes of navigational decisions [13], [14]. Nonetheless, the lack of a theoretical guarantee causes potential safety concerns. On the other hand, the optimization-based method relies on the formulation of a mathematical optimization problem, where system requirements can be explicitly expressed as equality and inequality constraints as part of the model predictive control (MPC) synthesis. To be more specific, typical constraints involved in the optimization problem include the dynamic constraint, environmental constraint (such as the obstacle avoidance constraint), physical limit constraint (such as the constraint on the steering angle, acceleration of the engine, and deceleration of the brake), etc. For the optimization-based method, the MPC has been widely deployed to solve the sequential decision-making problems with certain cumulative objectives considered over a certain horizon [15], [16]. In view of its appropriate applicability, various research works present the use of MPC-based methods to generate a feasible solution in the trajectory generation...
problem [17]. However, most of these MPC-based works focus on simple motion tasks with simplified vehicle models. In more realistic and complex situations, the MPC is no longer effective due to the nonlinearity of the vehicle model. Moreover, the generated trajectories need to deal with highly dynamic and complex driving scenarios (such as lane change, maneuvering, turning, overtaking, etc.) in a spatiotemporal domain. As a result, the high complexity and non-convexity of these constraints incurred add additional burdens to the computation efficiency of the MPC [18], [19].

Due to the limitation of the MPC in handling the nonlinear systems, the differential dynamic programming (DDP) is developed, which is known as a second-order shooting method admitting quadratic convergence [20], [21]. Classical DDP needs the second-order derivatives of the dynamics, yet the determination of the second-order derivatives aggravates the computation burden. Under a special condition that only the first-order derivative is used, it is reduced to the iterative linear quadratic regulator (iLQR) method with the Gauss-Newton approximation [22]. Basically, the iLQR expands the idea of the conventional LQR from linear systems to nonlinear systems, and it is known as an optimization-based method for nonlinear systems. Compared with the LQR, the iLQR optimizes a whole control sequence instead of just the control action at the current time. In the iLQR architecture, an initial trajectory is defined and then the iLQR is executed to refine the trajectory in an iterative framework, such that the optimal solution can be obtained efficiently. It is pertinent to note that the conventional iLQR only takes the system dynamic constraint into account in the optimization process, and it is the major shortcoming of the iLQR that it cannot handle the general inequality constraints appropriately. Therefore, to overcome this rather significant impediment, control-limited differential dynamic programming has been used to cater to the box constraints imposed on the control input [23]. In addition, an innovative variant of the iLQR so-called the constrained iLQR has been developed, which offers the inclusion of various general constraints [24], [25]. Rather importantly, the constrained iLQR demonstrates noteworthy results in terms of performance and computation efficiency, despite the existence of very involved nonlinear characteristics of the vehicle model and non-convexity of the obstacle avoidance constraint. Notably, the barrier function is adopted in [25] such that the general constraints are incorporated into the objective function appropriately. It is remarkable that it has shown significant improvement in computation efficiency as compared with the sequential quadratic programming (SQP) solver. However, the use of the logarithmic barrier function and the outer-inner loop framework invokes additional iterations. Also, it requires a feasible trajectory at the first iteration by using the iLQR with the logarithmic barrier function. Moreover, the optimization solver named IPOPT is widely deployed to solve such optimization problems. With this method, the optimization problem is solved directly with all predefined constraints coped simultaneously, and it usually takes longer time to find a satisfying solution. Hence, it remains an open problem for further improvement of the computation efficiency to meet the need for real-time implementation on top of that; and to this extent, the autonomous vehicle can better respond to emergencies in practical situations.

Nowadays, the development of numerical optimization tools has progressed leaps and bounds. As an emerging technique with excellent scalability, the alternating direction method of multipliers (ADMM) [26] has been deployed successfully in various domains, including optimal control, distributed computation, machine learning, and so on [27]–[30]. Essentially, the ADMM extends the method of multipliers and splits the optimization variable into two parts, and then the invoked sub-problems resulting from the splitting schemes can be attempted and solved in a separate framework. This approach relieves the typical computation burden arising from the growth of system dimensions. Recent advances in the ADMM enable an effective determination of the global optimum of a convex optimization problem, and these advances also motivate the researchers to investigate the behavior of the ADMM in non-convex problems. For example, [31] gives an empirical study of the ADMM for non-convex problems, and [32] performs the convergence analysis of the ADMM for a family of non-convex problems. Furthermore, the convergence behavior of the ADMM for problems with nonlinear equality constraints is investigated in [33]. Also, the convergence conditions for a coupled objective function with non-convex and non-smooth characteristics are studied in [34]. In these past research works, the ADMM has been reasonably established at the theoretical level. Certainly, these advanced optimization techniques bring promising prospects to the area of autonomous driving.

This paper presents an innovative and effective ADMM-based constrained iLQR approach for motion planning. In this work, the nonlinearity of the vehicle model is considered in the problem formulation. Also, various constraints are suitably addressed in this work, including the constraint on the acceleration due to the engine force limit and the braking force limit, the constraint on the steering angle due to the mechanical limitation of the vehicle, and the obstacle avoidance constraint for safety concern in driving scenarios. The main contributions of this work are:

- A novel constrained iLQR approach is presented to deal with the aforementioned constraints incurred in autonomous driving applications, and the nonlinearity in the vehicle system dynamics is addressed effectively.
- The ADMM algorithm is utilized to split the optimization problem into several manageable sub-problems. Therefore, the computation burden resulting from the nonlinearity and non-convexity of the motion planning problem is alleviated. The efficient computation makes the real-time implementation possible, and thus extra assurance of driving safety is provided.
- The proposed algorithm also avoids the necessity of searching for a feasible trajectory at the first iteration, which allows for more flexibility to set the initial trajectory.

The remainder of this paper is organized as follows. Section II presents the vehicle model, objective function, and constraints in the autonomous driving task. Section III presents
the ADMM-based constrained iLQR approach for motion planning. In Section IV, an illustrative example in autonomous driving is given to show the effectiveness of the proposed methodology. At last, the conclusions of this work are given in Section V.

II. PROBLEM STATEMENT

A. Vehicle Model

As shown in Fig. 1, the bicycle model used in [35] is borrowed to represent the vehicle dynamics. In this model, the aerodynamic influences, slip phenomena, and suspension movements are neglected. In terms of the vehicle model used in this work, the state vector is defined as

\[
\mathbf{x} = \begin{bmatrix} p_x, p_y, v_x, v_y, \phi, \omega \end{bmatrix}^T
\]

where \(p_x\) and \(p_y\) represent the X-coordinate and Y-coordinate of the center point of the vehicle, respectively; \(\phi\) denotes the heading angle; \(v_x\) and \(v_y\) denote the longitudinal and lateral velocity of the vehicle; \(\omega\) is the yaw rate. Also, the control input vector is defined as

\[
\mathbf{u} = \begin{bmatrix} a, \delta \end{bmatrix}^T,
\]

where \(a\) and \(\delta\) denote the acceleration and the steering angle, respectively. Here, we also define \(m\) as the mass of the vehicle, \(l_f\) and \(l_r\) as the distance from the center of mass to the front and rear axle, \(k_f\) and \(k_r\) as the cornering stiffness of the front and rear wheels, and \(I_x\) as the polar moment of inertia. For simplicity, we further define \(L_k = l_f k_f - l_r k_r\).

Given the time step \(T_s\), the nonlinear discrete model of the vehicle is expressed as

\[
\mathbf{x}(\tau+1) = f(x(\tau), u(\tau)) = \begin{bmatrix} p_x(\tau+1) \\ p_y(\tau+1) \\ \phi(\tau+1) \\ v_x(\tau+1) \\ v_y(\tau+1) \\ \omega(\tau+1) \end{bmatrix} = \begin{bmatrix} p_x(\tau) + T_s (v_x(\tau) \cos \phi(\tau) - v_y(\tau) \sin \phi(\tau)) \\ p_y(\tau) + T_s (v_x(\tau) \cos \phi(\tau) + v_y(\tau) \sin \phi(\tau)) \\ v_x(\tau) + T_s a(\tau) \\ v_y(\tau) + T_s \omega(\tau) + \frac{m v_x(\tau) v_y(\tau) + T_s \omega(\tau) (I_x + k_r) \phi(\tau) - T_s I_y \omega(\tau) k_f \phi(\tau) - I_x \omega(\tau) + T_s \omega(\tau) k_r \phi(\tau)}{I_x v_x(\tau) + T_s v_x(\tau) k_f \phi(\tau) - I_y v_y(\tau) + T_s v_y(\tau) k_r \phi(\tau)} \\ \frac{m v_x(\tau) v_y(\tau) + T_s \omega(\tau) (I_x + k_r) \phi(\tau) - T_s I_y \omega(\tau) k_f \phi(\tau) - I_x \omega(\tau) + T_s \omega(\tau) k_r \phi(\tau)}{I_y v_y(\tau) + T_s v_y(\tau) k_f \phi(\tau) - I_x v_x(\tau) + T_s v_x(\tau) k_r \phi(\tau)} \end{bmatrix}
\]

(1)

B. Objective Function

In the objective function, the position tracking error and longitudinal velocity tracking error are considered separately. Also, the steering wheel input is considered because it is closely related to the safety and comfort of passengers in on-road driving scenarios. Moreover, the acceleration is taken into account, as it reflects the fuel consumption as well as the comfort of passengers. With the above descriptions, the following objective function in each time stamp is constructed:

\[
\ell_c(x(\tau), u(\tau)) = d_{q_1, q_2}(x(\tau), l'(\tau)) + q_3 \left\| M_v x(\tau) - v'_r(\tau) \right\|^2 + r_1 u(\tau)^T M_2 u(\tau) + r_2 u(\tau)^T M_3 u(\tau),
\]

(2)

where \(\| \cdot \|\) denotes the Euclidean norm operator; \(q_1, q_2, q_3, r_1,\) and \(r_2\) represent the weighting parameters of the X-coordinate position tracking error, Y-coordinate position tracking error, longitudinal velocity tracking error, steering wheel input, and acceleration, respectively; \(l'\) denotes the polyline of the reference trajectory; \(d_{q_1, q_2}\) denotes the weighted Euclidean distance between the vehicle and the corresponding polyline; \(v'_r\) denotes the reference of longitudinal velocity; \(M_v\) is the matrix to extract the longitudinal velocity of the vehicle from the state vector, i.e., \(M_v = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}\); \(M_2\) and \(M_3\) are two matrices given by \(M_2 = \begin{bmatrix} 0 & 0 \end{bmatrix}\) and \(M_3 = \begin{bmatrix} 0 & 0 \end{bmatrix}\).

In this work, to facilitate the use of the iLQR method, the objective function at each time stamp is transformed into the following form:

\[
\ell_c(x(\tau), u(\tau)) = \begin{bmatrix} x(\tau) \\ u(\tau) \end{bmatrix}^T C \begin{bmatrix} x(\tau) \\ u(\tau) \end{bmatrix} - 2 \begin{bmatrix} x(\tau) \\ u(\tau) \end{bmatrix}^T Cr + \text{const},
\]

(3)

where

\[
C = \text{diag}(q_1, q_2, 0, q_3, 0, 0, r_1, r_2),
\]

\[
r = \begin{bmatrix} p'_{x_r} & p'_{y_r} & 0 & v'_{r_x} & 0 & 0 & 0 & 0 \end{bmatrix}^T,
\]

(4)

where \(p'_{x_r}, p'_{y_r}, v'_{r_x}\) denote the X-coordinate reference position, Y-coordinate reference position, and reference longitudinal velocity, respectively.

C. Constraints

First of all, the steering angle of the vehicle is bounded due to its mechanical limitations. Therefore, for all \(\tau = 0, 1, \cdots, T - 1\), we have

\[
-\delta_{\text{max}} \leq V_\delta u(\tau) \leq \delta_{\text{max}},
\]

(5)

with \(V_\delta = \begin{bmatrix} 0 & 1 \end{bmatrix}\), and \(\delta_{\text{max}}\) represents the largest possible steering angle that the vehicle attains.

Then, due to the engine force limit and braking force limit of the vehicle, the acceleration is also bounded, that is for all \(\tau = 0, 1, \cdots, T - 1\),

\[
a_{\text{max dec}} \leq V_a u(\tau) \leq a_{\text{max acc}},
\]

(6)
where \( V_a = [1, 0] \), and \( a_{\text{max,dec}} \) and \( a_{\text{max,acc}} \) are the maximum values of the acceleration of the engine and the deceleration of the brake, respectively, within the capability of the vehicle.

Next, the obstacle avoidance constraint is considered. In this work, the ego vehicle is formulated by a rectangle, and the obstacles (e.g., other vehicles) are formulated as ellipses. As the safe distance at the front and rear of the vehicle is larger, while the safe distance at the sides of the vehicle is smaller, the collision regions can be appropriately represented by ellipses. In this sense, with the the time stamp \( \tau \) and the heading angle of the obstacle \( \theta_0 \), the rotational matrix is given by

\[
R(\tau) = \begin{bmatrix}
\cos\theta_0(\tau) & -\sin\theta_0(\tau) \\
\sin\theta_0(\tau) & \cos\theta_0(\tau)
\end{bmatrix}.
\]

(7)

Then, denote \( e_a \) and \( e_p \) as the length of semi-major and semi-minor axes corresponding to the ellipse, and \( d_{xy}(\tau) \) by ellipses. In this sense, with the time stamp \( \tau \) and the system state vector in the last time stamp.

III. ADMM-BASED CONSTRAINED iLQR ALGORITHM FOR MOTION PLANNING

In this section, we present a general approach for the motion planning problem. In this following text, we propose a feasible nominal trajectory \( \{\hat{x}(\tau), \hat{\dot{x}}(\tau)\}_{\tau=0}^{T} \) is generated by passing the nominal input trajectory to the nonlinear system with the system initial state variables. Then, the second step is the backward pass. With the value function at the final time stamp, i.e., \( V_T(x(T)) = \phi(x(T)) \), the iteration proceeds backwards from the time stamp \( T-1 \) to the first time stamp.

Consider the right-hand side of (11), and we denote the perturbed Q-function as \( Q_{\tau}(\delta x(\tau), \delta u(\tau)) \), where \( \delta x(\tau) \) and \( \delta u(\tau) \) represent the amount of change in terms of the system state vector and control input vector at the time stamp \( \tau \) in the two adjacent iterations. The perturbed Q-function can be represented by

\[
Q_{\tau}(\delta x(\tau), \delta u(\tau)) = \ell_{\tau}(x(\tau) + \delta x(\tau), u(\tau) + \delta u(\tau)) - \ell_{\tau}(x(\tau), u(\tau)) + V_{\tau+1} \left( f(x(\tau) + \delta x(\tau), u(\tau) + \delta u(\tau)) \right) - V_{\tau+1} \left( f(x(\tau), u(\tau)) \right).
\]

(12)

Furthermore, by performing second-order Taylor expansion, we approximate the perturbation term by the following matrix equation:

\[
Q_{\tau}(\delta x(\tau), \delta u(\tau)) \approx \frac{1}{2} \begin{bmatrix}
1 & \delta x(\tau) \\delta u(\tau)
\end{bmatrix} \begin{bmatrix}
0 & (Q_{\tau})_x & (Q_{\tau})_{ux} \\
(Q_{\tau})_x & (Q_{\tau})_{xx} & (Q_{\tau})_{ux} \\
(Q_{\tau})_{ux} & (Q_{\tau})_{ux} & (Q_{\tau})_{uu}
\end{bmatrix} \begin{bmatrix}
1 \\
\delta x(\tau) \\
\delta u(\tau)
\end{bmatrix},
\]

(13)

with

\[
\begin{align*}
(Q_{\tau})_x &= \ell_{\tau} + f_x \cdot (V_{\tau+1})_x \\
(Q_{\tau})_u &= \ell_{\tau} + f_u \cdot (V_{\tau+1})_u \\
(Q_{\tau})_{xx} &= \ell_{\tau,xx} + f_{xx} \cdot (V_{\tau+1})_{xx} + (V_{\tau+1})_x \cdot f_{xx} \\
(Q_{\tau})_{ux} &= \ell_{\tau,ux} + f_{ux} \cdot (V_{\tau+1})_{ux} + (V_{\tau+1})_x \cdot f_{ux} \\
(Q_{\tau})_{uu} &= \ell_{\tau,uu} + f_{uu} \cdot (V_{\tau+1})_{uu} + (V_{\tau+1})_x \cdot f_{uu},
\end{align*}
\]

(14)

where the operator \( \cdot \) in the last three sub-equations denotes the contraction of a vector with a tensor and the subscript in each matrix function denotes the partial derivative of that function with respect to the notation in the subscript. Recall that the iLQR is a special case of the DDP, and the major difference is that the iLQR uses only the first-order derivative of the dynamic function \( f \), so the terms related to \( f_{xx} \), \( f_{ux} \) and \( f_{uu} \) in (14) are ignored in our problem. It is also worthwhile to mention that both the iLQR and the DDP use the second-order derivative of the value function.

Hence, to further proceed the backward iterations, the optimal control action for the perturbed Q-function is given by

\[
\delta u(\tau) = \arg \min_{\delta u(\tau)} Q_{\tau}(\delta x(\tau), \delta u(\tau)),
\]

(15)

and we have

\[
\delta u(\tau) = k(\tau) + K(\tau)\delta x(\tau),
\]

(16)
where \( k(\tau) \) and \( K(\tau) \) are considered as the feedforward vector and feedback matrix at the time stamp \( \tau \). Subsequently, it yields that

\[
\begin{align*}
  k(\tau) &= -(Q_{\tau})^{-1}_{uu} (Q_{\tau})_{u} \\
  K(\tau) &= -(Q_{\tau})^{-1}_{uu} (Q_{\tau})_{ux}.
\end{align*}
\]

(17a) and (17b)

Substitute (17a) and (17b) to the Taylor expansion of perturbed Q-function, and then, the difference, gradient, and Hessian of the value function are given by

\[
\Delta V = -\frac{1}{2} k(\tau)^\top (Q_{\tau})_{uu} k(\tau)
\]

(18a)

\[
(V_t)_x = (Q_{\tau})_x - K(\tau)^\top (Q_{\tau})_{uu} k(\tau)
\]

(18b)

\[
(V_t)_xx = (Q_{\tau})_{xx} - K(\tau)^\top (Q_{\tau})_{uu} K(\tau).
\]

(18c)

As follows, the backward pass is recursively conducted by calculating (17a)-(17b) and (18b)-(18c), until the first time stamp is reached.

Then, after the backward pass is completed, the third step is carried out, which is the forward pass. In the forward pass, the actual trajectory will be generated using the system dynamics and the control action pre-determined by (17a) and (17b).

Here, we have

\[
\begin{align*}
  u(\tau) &= \hat{u}(\tau) + k(\tau) + K(\tau)(x(\tau) - \hat{x}(\tau)) \\
  x(\tau + 1) &= f(x(\tau), u(\tau)).
\end{align*}
\]

(19)

Remarkably, the basic idea of the iLQR is to find the optimal control action \( \delta u(\tau)^* \) in the backward pass to drive the system, such that the nominal trajectory \( (\hat{x}, \hat{u}) \) can be updated to a new feasible trajectory in the forward pass. In the end, it will converge to the optimal trajectory asymptotically.

### B. ADMM Algorithm for Constrained iLQR

If inequality constraints are considered in the motion planning problem, the constrained optimization problem is given by

\[
\begin{align*}
  \text{minimize} & \quad \phi(x(T)) + \sum_{\tau=0}^{T-1} \ell_\tau(x(\tau), u(\tau)) \\
  \text{subject to} & \quad x(\tau + 1) = f(x(\tau), u(\tau)) \\
  & \quad g(y) \geq 0 \\
  & \quad \tau = 0, 1, \cdots, T - 1,
\end{align*}
\]

(20)

where the vector \( y \) is defined as

\[
y = (u(0), x(1), u(1), \cdots, x(T)) \in \mathbb{R}^{(n+m)T},
\]

and the function \( g \) denotes the inequality constraints included in the optimization problem. Notably, the symbol \( \geq \) represents the generalized “greater than or equal to” in this work. For instance, in the vector case, the symbol \( \geq \) denotes the element-wise “greater than or equal to”; in the matrix case, the symbol \( \geq \) represents the positive semi-definiteness.

To further simplify the optimization problem, we introduce the definition of the indicator function.

**Definition 1**: The indicator function with respect to a set \( \mathcal{B} \) is defined as

\[
\delta_{\mathcal{B}}(B) = \begin{cases} 
0 & \text{if } B \in \mathcal{B} \\
\infty & \text{otherwise}.
\end{cases}
\]

(21)

We further define the sets

\[
\mathcal{F} = \left\{ y = (u(0), x(1), u(1), \cdots, x(T)) \mid x(\tau + 1) = f(x(\tau), u(\tau)), \forall \tau = 0, 1, \cdots, T - 1 \right\}
\]

(22)

\[
\mathcal{G} = \left\{ y \in \mathbb{R}^{(n+m)T} \mid g(y) \geq 0 \right\}.
\]

(22)

Then the optimization problem can be equivalently denoted by

\[
\begin{align*}
  \text{minimize} & \quad \phi(x(T)) + \sum_{\tau=0}^{T-1} \ell_\tau(x(\tau), u(\tau)) + \delta_{\mathcal{F}}(y) + \delta_{\mathcal{G}}(y).
\end{align*}
\]

(23)

By introducing the consensus variable \( \bar{z} \), (23) can be further expressed as

\[
\begin{align*}
  \text{minimize} & \quad \phi(x(T)) + \sum_{\tau=0}^{T-1} \ell_\tau(x(\tau), u(\tau)) \quad \text{subject to} \quad y - \bar{z} = 0.
\end{align*}
\]

(24)

Because the optimization variables in the vector \( \bar{z} \) in terms of the inequality constraints are not exactly related to each variable in the vector \( y \), we introduce the operator \( A : \mathbb{R}^{(n+m)T} \to \mathbb{R}^p \) to extract the variable that is related to the inequality constraints to accelerate the convergence.

Then the optimization problem is converted to

\[
\begin{align*}
  \text{minimize} & \quad \phi(x(T)) + \sum_{\tau=0}^{T-1} \ell_\tau(x(\tau), u(\tau)) + \delta_{\mathcal{F}}(y) + \delta_{\mathcal{G}}(z) \quad \text{subject to} \quad Ay - z = 0,
\end{align*}
\]

(25)

where \( z \in \mathbb{R}^p \) is the new variable that is related to the inequality constraint; \( p \) is the number of optimization variables related to the inequality constraints; the new set \( \mathcal{G} \) is represented by

\[
\mathcal{G} = \left\{ z = A\bar{z} \mid g(\bar{z}) \geq 0 \right\}.
\]

(26)

where \( g \) is a function of \( \bar{z} \) that is used for constructing the inequality constraint in the optimization problem.

Define the augmented Lagrangian function of the optimization problem as

\[
\begin{align*}
  \mathcal{L}_\sigma(y, z; \lambda) &= \phi(x(T)) + \sum_{\tau=0}^{T-1} \ell_\tau(x(\tau), u(\tau)) + \delta_{\mathcal{F}}(y) \\
  &\quad + \delta_{\mathcal{G}}(z) + \frac{\sigma}{2} \| Ay - z + \sigma^{-1} \lambda \|^2 - \frac{1}{2\sigma} \| \dot{\lambda} \|^2,
\end{align*}
\]

(27)
where $\sigma \in \mathbb{R}$ is the penalty parameter of the augmented Lagrangian function, and $\lambda \in \mathbb{R}^p$ is the Lagrange multiplier. Following [26], the ADMM iterations are listed as

$$
\begin{align*}
y^{k+1} &= \arg\min_y \mathcal{L}_\sigma(y, z^k; \lambda^k) \\
z^{k+1} &= \arg\min_z \mathcal{L}_\sigma(y^{k+1}, z; \lambda^k) \\
\lambda^{k+1} &= \lambda^k + \sigma(Ay^{k+1} - z^{k+1}), \quad (28)
\end{align*}
$$

where the superscript $(\cdot)^k$ denotes the iteration number in the ADMM algorithm.

1) First ADMM Iteration: In the first ADMM iteration, it aims at solving the following sub-problem:

$$
\begin{align*}
&\text{minimize} & & \phi(x(T)) + \sum_{\tau=0}^{T-1} \ell_\tau(x(\tau), u(\tau)) \\
&\text{subject to} & & x(\tau + 1) = f(x(\tau), u(\tau)) \\
& & & \tau = 0, 1, \cdots, T - 1, \quad (29)
\end{align*}
$$

Furthermore, to facilitate the deployment of the iLQR method, the optimization problem (29) can be equivalently expressed as

$$
\begin{align*}
&\text{minimize} & & \phi(x(T)) + \sum_{\tau=0}^{T-1} \ell_\tau(x(\tau), u(\tau)) \\
& & & + \frac{\sigma}{2} \|Ay - z^k + \sigma^{-1}\lambda^k\|^2 \\
&\text{subject to} & & x(\tau + 1) = f(x(\tau), u(\tau)) \\
& & & \tau = 0, 1, \cdots, T - 1, \quad (30)
\end{align*}
$$

where the operator $\hat{A}(\tau) : \mathbb{R}^{n+m} \rightarrow \mathbb{R}^p$ extracts the optimization variable that is related to the inequality constraints in the time stamp $\tau$.

Remark 1: Since only dynamic constraints are included in the optimization problem, the optimization problem (30) can be solved by the iLQR algorithm efficiently.

2) Second ADMM Iteration: In the second iteration, it aims at solving the following sub-problem:

$$
\begin{align*}
&\text{minimize} & & \|A\hat{y}^{k+1} + \sigma^{-1}\lambda^k - z\|^2 \\
&\text{subject to} & & z \in \mathcal{C}. \quad (31)
\end{align*}
$$

Then this problem can be separated into $T + 1$ sub-problems, which can be solved in a parallel manner. That is, for all $\tau = 0, 1, \cdots, T - 1$,

$$
\begin{align*}
&\text{minimize} & & \|\hat{A}(\tau) \begin{bmatrix} x^{k+1}(\tau) \\ u^{k+1}(\tau) \end{bmatrix} + \sigma^{-1}\lambda^k(\tau) - z(\tau)\|^2 \\
&\text{subject to} & & g_\tau(z(\tau)) \geq 0, \quad (32)
\end{align*}
$$

where

$$
\begin{align*}
z &= (z(0), z(1), \cdots, z(T - 1)) \\
\lambda &= (\lambda(0), \lambda(1), \cdots, \lambda(T - 1)). \quad (33)
\end{align*}
$$

Fig. 2. Schematic illustration of the proposed algorithm.

Remark 2: In each time stamp, the sub-problem (32) represents a non-convex projection optimization problem. This is because that, for the Euclidean norm constraint, the projection inwards the ellipse is convex, while the projection outwards the ellipse is non-convex. In other words, $g_\tau(z(\tau)) \leq 0$ is convex, while $g_\tau(z(\tau)) \geq 0$ is non-convex. The surrounding vehicles in this work are formulated as ellipses, and the projection problem onto an ellipse can be easily solved with many existing methods in the literature, so the obstacle avoidance constraint can be addressed appropriately even though it is non-convex. It is also worthwhile to mention that the box constraints are convex, but it can still be addressed with a similar projection method in the second ADMM iteration.

3) Lagrange Multiplier Update: Next, the Lagrange multiplier can be updated with the following rule:

$$
\lambda^{k+1} = \lambda^k + \sigma(Ay^{k+1} - z^{k+1}). \quad (34)
$$

Remark 3: Due to the properties of the ADMM, it circumvents the feasibility requirement of the trajectory at the first iteration.

To summarize the above discussions, the schematic illustration of the proposed algorithm shown in Fig. 2, and the pseudocode is summarized in Algorithm 1.

C. Convergence Analysis and Stopping Criterion

The ADMM algorithm applied in a convex optimization problem can realize the first-order convergence. However, for non-convex optimization problems, the convergence of the algorithm cannot be guaranteed, yet there are many successful applications of the non-convex ADMM algorithms available in the literature. Besides, the convergence of the non-convex ADMM in a variety of scenarios is also well-established. It is noteworthy to mention that the ADMM algorithm usually shows satisfying convergence in the consensus structure.

Furthermore, the stopping criterion is essential for the algorithm because it will significantly influence the computation efficiency and performance. The stopping criterion for the
Algorithm 1 ADMM-Based Constrained iLQR for Motion Planning

Require: System dynamic model \( f \);
objective function \( \phi \) and \( \ell \);
inequality constraints \( g \);
initial system state vector \( x_0 \);
penalty parameter \( \sigma \);
the maximum iteration number of ADMM \( N_{\text{ADMM}} \);
the maximum iteration number of iLQR \( N_{\text{iLQR}} \).

1: Generate the initial trajectory \( (\dot{x}(\tau), \dot{u}(\tau)) \).
2: Formulate the optimization problem into the iLQR solvable form by (30).
   for \( i = 1, 2, \ldots, N_{\text{ADMM}} \) do
     (ADMM First Iteration)
     for \( j = 1, 2, \ldots, N_{\text{iLQR}} \) do
       6: Perform the iLQR backward pass by (17a)-(17b) and (18b)-(18c).
       7: Perform the iLQR forward pass by (19).
       if iLQR stopping criterion is satisfied then break
       end if
     end for
   (ADMM Second Iteration)
   13: Solve the optimization problem (32).
   (Lagrange Multiplier Update)
   15: Update the Lagrange multiplier by (34).
   if ADMM stopping criterion is satisfied then break
   end if
  end for

iLQR iterations is usually chosen as the amount of change of the value function. There are numerous choices of the stopping criterion in terms of the ADMM algorithm, such as the primal-dual residual error, primal residual error, etc. In our case, to simplify the whole optimization process, we use the ADMM iteration number as the stopping criterion.

IV. ILLUSTRATIVE EXAMPLE

Next, an illustrative example of autonomous driving is presented to validate the effectiveness of the proposed development. Three scenarios are considered here, including one static obstacle avoidance problem and two dynamic obstacle avoidance problems involving lane change and overtaking tasks. The optimization algorithm is implemented in a desktop platform with Intel(R) Xeon(R) W-2225 CPU @ 4.10GHz, the optimization is conducted in Python 3.7 with the Numba package, and the figures are plotted in MATLAB 2020b.

In the simulation, the solid thick lines in black color represent the road boundaries, the solid thin lines in black color represent the line that separates the road, the dash lines in black color represent the reference of the ego vehicle. Also, the ego vehicle is represented by the blue rectangle, the surrounding vehicles are represented by the yellow rectangles, and collision regions are represented by the ellipses with dash lines. The mass of the vehicle \( m = 1412 \text{ kg} \). The distance from the center of mass to the front and rear axle are \( l_f = 1.06 \text{ m} \) and \( l_r = 1.85 \text{ m} \), respectively. The cornering stiffness of the front and rear wheels are \( k_f = -128916 \text{ N/ rad} \) and \( k_r = -85944 \text{ N/ rad} \), respectively. The polar moment of inertia is \( I_z = 1536.7 \text{ kg·m}^2 \). The sampling time is \( T_s = 0.1 \text{ s} \). In all scenarios, the upper bound and lower bound for the steering angle are set as 0.6 rad and -0.6 rad, respectively. For the acceleration, the upper bound and lower bound are given by 1.5 m/s² and -3 m/s², respectively. The length and width of both the ego vehicle and the surrounding vehicles are 3 m and 2 m, respectively. In all scenarios, the weighting parameters are set as \( q_1 = 0, q_2 = 1, q_3 = 1, r_1 = 10, r_2 = 1 \) (X-coordinate position tracking error is not penalized in these driving tasks).

Fig. 3. Primal residual error during ADMM iterations in the static obstacle avoidance scenario.

A. Scenario 1: Static Obstacle Avoidance

In the first scenario, a static obstacle avoidance problem is investigated, where there is one vehicle parked on the street with a coordinate of \( (15 \text{ m}, -1 \text{ m}) \). For the ego vehicle, the initial position is set as \( (0 \text{ m}, 0 \text{ m}) \), the initial steering angle is 0 rad, and the initial longitudinal velocity is \( 5 \text{ m/s} \). The ego vehicle aims to drive along the street and avoid the collision with the vehicle parked on the street. The reference for the position is set as \( p_{x}^f = 0 \text{ m} \), and the reference for the longitudinal velocity is \( v_{x}^f = 8 \text{ m/s} \).

The primal residual error during the ADMM iterations is illustrated in Fig. 3. With the proposed approach, the change of trajectory for the ego vehicle during ADMM iterations is plotted in Fig. 4. To visualize the change clearly, only the trajectories of the ego vehicle in iteration 1, iteration 2, and iteration 20 are given. From this figure, it shows that a feasible trajectory is obtained finally. With the planned trajectory, the simulation is carried out, and Fig. 5 illustrates the trajectory and longitudinal velocity of the ego vehicle. Note that in this scenario, the position of the ego vehicle is plotted every 1 s. As indicated from this figure, the ego vehicle follows the planned trajectory (represented by the solid yellow line in the
background) closely, and it successfully avoids the obstacle (the vehicle parked on the street). From the longitudinal velocity profile, it is clear that to avoid this static obstacle, the ego vehicle accelerates at the beginning, and constantly converges to its desired value (represented by the dash yellow line).

Moreover, to validate the constraint satisfaction conditions, Fig. 6 is given, where the steering angle and acceleration of the ego vehicle are depicted. Also, the distance from the ego vehicle to the obstacle (denoted by $d_{xy}$) is presented in this figure. It can be seen that the steering angle and the acceleration are all bounded based on our settings. Besides, the distance between the ego vehicle and the obstacles indicates that no collision occurs in this scenario.

**B. Scenario 2: Dynamic Obstacle Avoidance–Lane Change**

In the second scenario, a dynamic obstacle avoidance problem is considered, where a lane change task in autonomous driving is discussed. Under this circumstance, it is assumed that there are two surrounding vehicles. One of them is in front of the ego vehicle, which has an initial position of $(20 \text{ m}, 0 \text{ m})$ and a constant longitudinal velocity of $3 \text{ m/s}$. Also, there is another vehicle on the adjacent lane (the target lane after the lane change). For the vehicle on the target lane, it is initially located at $(0 \text{ m}, 4 \text{ m})$ and it moves along the street with a longitudinal velocity of $6 \text{ m/s}$. Note that the initial position of the ego vehicle is $(0 \text{ m}, 0 \text{ m})$, the initial steering angle is $0 \text{ rad}$, and the initial longitudinal velocity is $8 \text{ m/s}$. The ego vehicle aims to change the lane while avoiding the collisions with the two vehicles driving on the street. The references for the position and the longitudinal velocity are set as $p_y = 4 \text{ m}$ and $v_x = 8 \text{ m/s}$.

Similarly, the primal residual error during the ADMM iterations is illustrated in Fig. 7. The change of trajectory for the ego vehicle during ADMM iterations is depicted in Fig. 8, where its trajectories in iteration 1, iteration 2, and iteration 20 are plotted. From this figure, the asymptotic convergence of the trajectory is validated. Subsequently, the trajectory of all the vehicles and the longitudinal velocity of the ego vehicle is given in Fig. 9. In this case, the position of all the vehicles is plotted every 1 s. It can be observed from this figure that the ego vehicle follows the planned trajectory (represented by the solid purple line in the background) closely, and no collision with the surrounding vehicles happens. It is worthwhile to mention that, though there are some interactions between the ego vehicle and the collision regions (represented by the ellipses), actually they do not collide at the same time stamp.
From the longitudinal velocity profile, it can be seen that the ego vehicle accelerates at the beginning, and converges to around 8 m/s.

Additionally, Fig. 10 is shown to validate the constraint satisfaction conditions. As indicated from this figure, the steering angle and acceleration of the ego vehicle are all constrained within the predefined bounds. In terms of the last subplot in Fig. 10, the curves in the red color and the blue color represent the distance from the ego vehicle to the obstacle on $p_y = 0$ m and $p_y = 4$ m, respectively. The results clearly illustrate that the ego vehicle avoids the surrounding vehicles successfully in this scenario.

C. Scenario 3: Dynamic Obstacle Avoidance–Overtaking

In the third scenario, an overtaking task is investigated considering two surrounding vehicles. One of them is on the adjacent lane, which has an initial position of (10 m, 4 m) and an longitudinal velocity of 10 m/s. Also, there is another vehicle in the front of the ego vehicle, which is initially located at (30 m, 0 m) and moves along the street with an initial velocity of 3 m/s, and the its velocity firstly increases to 8 m/s and then decreases to 3 m/s. Note that the initial position of the ego vehicle is (0 m, 0 m), the initial steering angle is 0 rad, and the initial longitudinal velocity is 15 m/s. The ego vehicle aims to perform the overtaking task and change to the original lane in the end. The reference for the position is set as $p_x^r = 0$ m, and the reference for the longitudinal velocity is $v_x^r = 15$ m/s.

The primal residual error and the change of trajectory for the ego vehicle during the ADMM iterations are illustrated in Fig. 11 and Fig. 12, respectively. Note that the asymptotic convergence of the trajectory is validated. Also, the trajectories of all the vehicles and the longitudinal velocity of the ego vehicle are given in Fig. 13. Similarly, the position of all the vehicles is plotted every 1 s. The different line types and line colors have the same meaning in Scenario 2. It can be seen that no collision with the surrounding vehicles happens. Also, from the longitudinal velocity profile, it can be seen that the ego vehicle accelerates at the beginning, and converges to around 15 m/s. Similarly, Fig. 14 shows that all the constraints are satisfied appropriately, including the bound of the steering angle and acceleration, as well as the distance from the ego vehicle to the surrounding vehicles.

D. Discussion on Computation Time

For the optimization in all scenarios, 5 trials with a prediction horizon of $T = 60$ are performed, and the computation
TABLE I

COMPUTATION TIME IN THE ALL THREE DRIVING SCENARIOS

|                  | Computation Time in Scenario 1 | Computation Time in Scenario 2 | Computation Time in Scenario 3 |
|------------------|--------------------------------|--------------------------------|--------------------------------|
|                  | Method 1 | Method 2 | Method 3 | Method 1 | Method 2 | Method 3 | Method 1 | Method 2 | Method 3 |
| Trial 1          | 0.0591 s | 0.0910 s | 0.1227 s | 0.1510 s | 0.2550 s | 0.3399 s | 0.0768 s | 0.1334 s | 0.6657 s |
| Trial 2          | 0.0549 s | 0.1080 s | 0.1238 s | 0.1786 s | 0.2614 s | 0.3457 s | 0.0686 s | 0.1496 s | 0.6653 s |
| Trial 3          | 0.0836 s | 0.1074 s | 0.1285 s | 0.1548 s | 0.2774 s | 0.3413 s | 0.0801 s | 0.1420 s | 0.6645 s |
| Trial 4          | 0.0824 s | 0.1102 s | 0.1299 s | 0.1512 s | 0.2516 s | 0.3368 s | 0.0837 s | 0.1358 s | 0.6712 s |
| Trial 5          | 0.0901 s | 0.0813 s | 0.1235 s | 0.1558 s | 0.2619 s | 0.3399 s | 0.0763 s | 0.1346 s | 0.6640 s |

Fig. 12. Change of trajectory during i LQR iterations in the overtaking scenario.

Fig. 13. Trajectory and longitudinal velocity of the vehicle in the overtaking scenario.

Notably, with Method 2, the average computation time in these three scenarios is given by 0.0996 s, 0.2575 s, and 0.1391 s, respectively. Taking Method 2 as the baseline, our proposed method reduces the average computation time by 31.93%, 38.52%, and 44.57% in the three scenarios, respectively. With Method 3, the average computation time in these three scenarios is given by 0.1256 s, 0.3387 s, and 0.6661 s, respectively. Taking Method 3 as the baseline, our proposed method reduces the average computation time by 46.02%, 53.26%, and 88.43% in the three scenarios, respectively. Apparently, when using Method 1 and Method 2, Scenarios 1 and 3 take shorter time for computation, while Scenario 2 takes longer time. However, with Method 3, Scenario 3 takes longer time for the computation.

As clearly observed from Table I, the proposed algorithm is highly efficient, which makes it possible to implement in a real-time framework. Compared with the existing works on the constrained iLQR algorithm with the logarithmic barrier function and the optimization solver IPOPT, this work leads to less computation effort. Also, it avoids the feasibility requirement for the trajectory at the first iteration. Notably, the computation efficiency of the algorithm can be further enhanced by choosing a loose stopping criterion. The prediction horizon in the given examples is large enough to realize the on-road driving tasks, and it is also possible to reduce the computation time by choosing a smaller prediction horizon.
V. CONCLUSION

This work investigates the motion planning problem in autonomous driving applications. Considering the nonlinear dynamics of the vehicle model and various pertinent constraints, a constrained optimization problem is suitably formulated on the basis of the iLQR method. Next, we propose the implementation of the ADMM approach to split the optimization problem into several sub-problems, and these sub-problems can be efficiently solved. As a result, real-time computation and implementation can be realized through this framework, and thus it provides additional safety to the on-road driving tasks. Moreover, an illustrative example of autonomous driving is used to validate the performance of the approach in motion planning tasks, where different typical driving tasks have been scheduled as part of the test itinerary. As illustrated from the comparative computation times and simulation results, the significance as claimed in this work is suitably demonstrated.

REFERENCES

[1] S. E. Li et al., “Dynamical modeling and distributed control of connected and automated vehicles: Challenges and opportunities,” IEEE Intell. Transp. Syst. Mag., vol. 9, no. 3, pp. 46–58, Jul. 2017.

[2] J. Zhang, F.-Y. Wang, K. Wang, W.-H. Lin, X. Xu, and C. Chen, “Data-driven intelligent transportation systems: A survey,” IEEE Trans. Intell. Transp. Syst., vol. 12, no. 4, pp. 1624–1639, Dec. 2011.

[3] P. Hang, C. Lv, C. Huang, Y. Xing, and Z. Hu, “Cooperative decision making of connected automated vehicles at multi-lane merging zone: A coalitional game approach,” IEEE Trans. Intell. Transp. Syst., vol. 23, no. 4, pp. 3829–3841, Apr. 2022.

[4] F.-Y. Wang, “Parallel control and management for intelligent transportation systems: Concepts, architectures, and applications,” IEEE Trans. Intell. Transp. Syst., vol. 11, no. 3, pp. 630–638, Sep. 2010.

[5] D. González, J. Pérez, V. Milánés, and F. Nashashibi, “A review of motion planning techniques for automated vehicles,” IEEE Trans. Intell. Transp. Syst., vol. 17, no. 4, pp. 1135–1145, Apr. 2016.

[6] L. Hou, L. Xin, S. E. Li, B. Cheng, and W. Wang, “Interactive trajectory prediction of surrounding road users for autonomous driving using structural-LSTM network,” IEEE Trans. Intell. Transp. Syst., vol. 21, no. 11, pp. 4615–4625, Nov. 2020.

[7] J. Duan, Y. Guan, E. Li, Y. Ren, Q. Sun, and B. Cheng, “Distributional soft actor-critic: Off-policy reinforcement learning for addressing value estimation errors,” IEEE Trans. Neural Netw. Learn. Syst., early access, Jun. 8, 2021, doi: 10.1109/TNNLS.2021.3082568.

[8] J. Chen, S. E. Li, and M. Tomizuka, “Interpretable end-to-end urban autonomous driving with latent deep reinforcement learning,” IEEE Trans. Intell. Transp. Syst., vol. 23, no. 6, pp. 5068–5078, Dec. 2022.

[9] J. Duan, S. E. Li, Y. Guan, Q. Sun, and B. Cheng, “Hierarchical reinforcement learning for self-driving decision-making without reliance on labelled driving data,” IET Intell. Transp. Syst., vol. 14, no. 5, pp. 297–305, 2020.

[10] Y. Shan, B. Zheng, L. Chen, L. Chen, and D. Chen, “A reinforcement learning-based adaptive path tracking approach for autonomous driving,” IEEE Trans. Veh. Technol., vol. 69, no. 10, pp. 10581–10595, Oct. 2020.

[11] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, “Constrained model predictive control: Stability and optimality,” Automatica, vol. 36, no. 6, pp. 789–814, 2000.

[12] C. Liu, S. Lee, S. Varnhagen, and H. E. Tseng, “Path planning for autonomous vehicles using model predictive control,” in Proc. IEEE Intell. Vehicles Symp. (IV), Jun. 2017, pp. 174–179.

[13] J. Ji, A. Khajepour, W. W. Melek, and Y. Huang, “Path planning and tracking for vehicle collision avoidance based on model predictive control with multicostants,” IEEE Trans. Ultrason. Eng., vol. 66, no. 2, pp. 952–964, Feb. 2017.

[14] X. Zhang, J. Ma, S. Huang, Z. Cheng, and T. H. Lee, “Integrated planning and control for collision-free trajectory generation in 3D environment with obstacles,” in Proc. 19th Int. Conf. Control, Autom. Syst. (ICCAS), Oct. 2019, pp. 974–979.

[15] X. Zhang, J. Ma, Z. Cheng, S. Huang, S. S. Ge, and T. H. Lee, “Trajectory generation by chance-constrained nonlinear MPC with probabilistic prediction,” IEEE Trans. Cybern., vol. 51, no. 7, pp. 3616–3629, Jul. 2021.

[16] D. Mayne, “A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems,” Int. J. Control, vol. 3, no. 1, pp. 85–95, Jan. 1966.

[17] Z. Xie, C. K. Liu, and K. Hauser, “Differential dynamic programming with nonlinear constraints,” in Proc. IEEE Int. Conf. Robot. Autom. (ICRA), May 2017, pp. 695–702.

[18] A. Nagarjuna and S. Saripalli, “An iterative LQR controller for off-road and on-road vehicles using a neural network dynamics model,” in Proc. IEEE Intell. Vehicles Symp. (IV), Oct. 2020, pp. 1740–1745.

[19] Y. Tassa, N. Mansard, and E. Todorov, “Control-limited differential dynamic programming,” in Proc. IEEE Int. Conf. Robot. Autom. (ICRA), May 2014, pp. 1168–1175.

[20] Y. Pan, Q. Lin, H. Shah, and J. M. Dolan, “Safe driving for self-driving via adaptive constrained ILQR,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Oct. 2020, pp. 2377–2383.

[21] J. Chen, W. Zhan, and M. Tomizuka, “Autonomous driving motion planning with constrained iterative LQR,” IEEE Trans. Intell. Veh., vol. 4, no. 2, pp. 244–254, Jun. 2019.

[22] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, “Distributed optimization and statistical learning via the alternating direction method of multipliers,” Found. Trends Mach. Learn., vol. 3, no. 1, pp. 1–122, 2011.

[23] J. Ma, Z. Cheng, X. Zhan, M. Tomizuka, and T. H. Lee, “On symmetric Gauss–Seidel ADMM algorithm for $H_{\infty}$ guaranteed cost control with convex parameterization,” IEEE Trans. Syst., Man, Cybern., Syst., early access, Jul. 27, 2022, doi: 10.1109/TSMCC.2021.3185162.

[24] G. Raja, S. Anbalagan, G. Vijayaraghavan, S. Theerthagiri, S. V. Suryanarayan, and X.-W. Wu, “SP-CIDS: Secure and private collaborative IDS for VANETs,” IEEE Trans. Intell. Transp. Syst., vol. 22, no. 7, pp. 4385–4393, Jul. 2021.

[25] J. Chen and T. Ma, X. Zhu, and T. H. Lee, “Semi-proximal ADMM for model predictive control problem with application to a UAV system,” in Proc. 20th Int. Conf. Control, Autom. Syst. (ICCAS), Oct. 2020, pp. 82–87.

[26] J. Ma, Z. Cheng, X. Zhan, M. Tomizuka, and T. H. Lee, “Optimal decentralized control for uncertain systems by symmetric Gauss–Seidel semi-proximal ALM,” IEEE Trans. Autom. Control, vol. 66, no. 11, pp. 5554–5560, Nov. 2021.

[27] Z. Xu, S. De, M. Figueiredo, C. Studer, and T. Goldstein, “An empirical study of ADMM for nonconvex problems,” 2016, arXiv:1612.03439.

[28] M. Hong, Z.-Q. Luo, and M. Razaviyayn, “Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems,” SIAM J. Optim., vol. 26, no. 1, pp. 337–364, 2016.

[29] J. Wang and L. Zhao, “Nonconvex generalization of alternating direction method of multipliers for nonlinear equality constrained problems,” 2017, arXiv:1705.03412.

[30] Y. Wang, W. Yin, and J. Zeng, “Global convergence of ADMM in nonconvex nonsmooth optimization,” J. Sci. Comput., vol. 78, no. 1, pp. 29–63, Jun. 2019.

[31] Q. Ge, Q. Sun, S. E. Li, S. Zheng, W. Wu, and X. Chen, “Numerically stable dynamic bicycle model for discrete-time control,” in Proc. IEEE Intell. Vehicles Symp. Workshops, Jul. 2021, pp. 128–134.

[32] A. Wächter and L. T. Biegler, “On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming,” Math. Program., vol. 106, no. 1, pp. 25–57, May 2006.
Jun Ma received the B.Eng. degree (First Class Hons.) in electrical and electronic engineering from Nanyang Technological University, Singapore, in 2014, and the Ph.D. degree in electrical and computer engineering from the National University of Singapore, Singapore, in 2018. From 2018 to 2021, he held several positions at the National University of Singapore; University College London, London, U.K.; University of California, Berkeley, Berkeley, CA, USA; and Harvard University, Cambridge, MA, USA. He is currently an Assistant Professor with the Robotics and Autonomous Systems Thrust, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou, China; an Affiliate Assistant Professor with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong SAR, China; and an Assistant Professor with the HKUST Shenzhen-Hong Kong Collaborative Innovation Research Institute, Futian, Shenzhen, China. His research interests include control, optimization, and machine learning with application to robotics and autonomous driving.

Zilong Cheng received the B.Eng. degree in automotive engineering from Jilin University, Changchun, China, in 2018. He is currently pursuing the Ph.D. degree in electrical and computer engineering with the National University of Singapore, Singapore. His research interests include reinforcement learning, optimization, optimal control, and robotics. He was a recipient of the NUS Graduate School for Integrative Sciences and Engineering Scholarship.

Xiaoxue Zhang received the B.Eng. and M.Sc. degrees in automotive engineering from China Agricultural University, Beijing, China, in 2015 and 2018, respectively. She is currently pursuing the Ph.D. degree in electrical and computer engineering with the National University of Singapore, Singapore. Her research interests include reinforcement learning, optimization, motion planning, optimal control, and robotics. She was a recipient of the NUS Graduate School for Integrative Sciences and Engineering Scholarship.

Masayoshi Tomizuka (Life Fellow, IEEE) received the B.S. and M.S. degrees in mechanical engineering from Keio University, Tokyo, Japan, in 1968 and 1970, respectively, and the Ph.D. degree in mechanical engineering from the Massachusetts Institute of Technology, Cambridge, MA, USA, in February 1974. From 1974, he has joined the Faculty of the Department of Mechanical Engineering, University of California, Berkeley, Berkeley, CA, USA, where he is currently the Cheryl and John Neerhout, Jr., Distinguished Professorship Chair. His current research interests are optimal and adaptive control, digital control, motion control, and their applications to robotics and vehicles. Dr. Tomizuka is a Life Fellow of ASME and a fellow of the International Federation of Automatic Control (IFAC) and the Society of Manufacturing Engineers. He was a recipient of the Best J-DSMC Best Paper Award in 1995 and 2010, the ASME/DSCD Outstanding Investigator Award in 1996, the Charles Russ Richards Memorial Award (ASME) in 1997, the Rufus Oldenburger Medal (ASME) in 2002, the John R. Ragazzini Award (AACC) in 2006, the Richard Bellman Control Heritage Award (AACC) in 2018, the Honda Medal (ASME) in 2019, and the Nathaniel B. Nichols Medal (IFAC) in 2020. He has served as the Program Director for the Dynamic Systems and Control Program, Civil and Mechanical Systems Division, NSF, from 2002 to 2004. He has served as a Technical Editor for the Journal of Dynamic Systems, Measurement, and Control (ASME), Measurement and Control, and J-DSMC from 1988 to 1993. He has served as the Editor-in-Chief for the IEEE/ASME TRANSACTIONS ON MECHATRONICS from 1997 to 1999.

Tong Heng Lee received the B.A. degree (First Class Hons.) in electrical engineering from the University of Cambridge, Cambridge, U.K., in 1980, the M.Eng. degree in electrical engineering from the National University of Singapore (NUS), Singapore, in 1985, and the Ph.D. degree in electrical engineering from Yale University, New Haven, CT, USA, in 1987. He is currently a Professor with the Department of Electrical and Computer Engineering, NUS, and the NUS Graduate School, NUS NGS. He was the Past Vice-President (research) at NUS. His research interests are in the areas of adaptive systems, knowledge-based control, intelligent mechatronics, and computational intelligence. Dr. Lee holds an Associate Editor appointments in IEEE TRANSACTIONS ON CYBERNETICS, Control Engineering Practice (IFAC), and the International Journal of Systems Science (Taylor & Francis, U.K.). In addition, he is a Past Deputy Editor-in-Chief of Mechatronics (IFAC).