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

Jun Ma, Zilong Cheng, Xiaoxue Zhang, Masayoshi Tomizuka, *Life Fellow, IEEE,* and Tong Heng Lee

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 models 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. 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 an alternating direction method of multipliers (ADMM) is developed to determine the optimal control actions. 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 in autonomous vehicles is then investigated with different driving scenarios taken into consideration. As clearly observed from the simulation results, the significance of this work in terms of obstacle avoidance is demonstrated. Furthermore, a noteworthy achievement of high computation efficiency is attained; and 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.

Index Terms—Autonomous driving, iterative linear quadratic regulator (iLQR), alternating direction method of multipliers (ADMM), motion planning, nonlinear system, non-convex constraint.

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. 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 collision 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 [3]–[5]. However, they have readily stood to benefit tremendously from the recent advances in optimization and artificial intelligence [6], [7].

Along with this line, the planning methods in the continuous space are generally divided into two categories, which are 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 [8]–[10]. 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 MPC synthesis. To be more specific, typical constraints involved in the optimization problem include the dynamic constraint, environmental constraint (such as the collision avoidance constraint), and physical limit constraint (such as the constraint on the steering angle, acceleration of the engine, deceleration of the brake), etc. For the optimization-based method, the model predictive control (MPC) has been widely deployed to solve the sequential decision-making problems with certain cumulative objectives considered over a certain horizon [11]–[14]. 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 [15], [16]. 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 [17]. 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 so-called

J. Ma and M. Tomizuka are with the Department of Mechanical Engineering, University of California, Berkeley, CA 94720 USA (e-mail: jun.ma@berkeley.edu; tomizuka@berkeley.edu).

Z. Cheng, X. Zhang, and T. H. Lee are with the NUS Graduate School for Integrative Sciences and Engineering, National University of Singapore, 119077 (e-mail: zilongcheng@u.nus.edu; xiaoxuezhang@u.nus.edu; eleleeth@nus.edu.sg).

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. The iLQR can be solved by dynamic programming, and eventually, the optimal solution can be obtained efficiently. It is pertinent to note that the conventional iLQR does not take the constraints into consideration. Therefore, to overcome this rather significant impediment, control-limited differential dynamic programming (CLDDP) 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 constrained iLQR has been developed, which offers the inclusion of various general constraints [24]–[26]. Rather importantly, the constrained iLQR demonstrates noteworthy performance in terms of performance and computation efficiency, despite the existence of very involved nonlinear characteristics of the vehicle model and non-convexity of the collision avoidance constraint. In [25], the barrier function is adopted 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. 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. Also, it renders difficult to derive a feasible trajectory at the first iteration by using the iLQR with the logarithmic barrier function.

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) has been deployed successfully in various domains, including optimal control, distributed computation, machine learning, etc [27]–[29]. Essentially, the ADMM extends the method of multipliers and splits the optimization variable into two parts, and then the invoked sub-problems resulted 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, [30] gives an empirical study of the ADMM for non-convex problems, and [31] 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 [32]. Also, the convergence conditions for a coupled objective function with non-convex and non-smooth

characteristics are studied in [33]. 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 approach for solving the iLQR problem in motion planning tasks. 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 acceleration constraint due to the engine force limit and the braking force limit, steering angle constraint due to the mechanical limitation of the vehicle, and collision avoidance constraint for safety concern in driving scenarios. Certainly, these constraints lead to a non-convex constrained optimization problem. To solve this problem, the constrained iLQR approach is presented. Moreover, to speed up the determination of the optimal results, the ADMM algorithm is developed, which splits the optimization problem into several sub-problems. Compared with the existing works in the literature, this work attains the key advantage of alleviating the computation burden resulted from the nonlinearity and non-convexity of the motion planning problem, and also avoids the necessity of planning a feasible trajectory at the first iteration. The efficient computation makes the real-time implementation possible, and thus extra assurance of driving safety is provided.

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 proposes the ADMM approach for solving the iLQR-based motion planning problem. 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

For the vehicle model used in this work, the state vector is defined as $x = \begin{bmatrix} p_x & p_y & \theta & v \end{bmatrix}^T$, where p_x and p_y represent the X-coordinate and Y-coordinate of the center point between the back wheels, respectively; θ denotes the heading of the vehicle; v represents the velocity of the front wheels. Also, the control input vector is defined as $u = \begin{bmatrix} w & a \end{bmatrix}^T$, where w and a denote the steering angle and the acceleration of the front wheels, respectively. Therefore, given the time step h, the rolling distance of the front wheels and the back wheels are given by

$$f(v) = hv$$
$$b(v, w) = d + f(v)\cos(w) - \sqrt{d^2 - f(v)^2\sin^2(w)}, \quad (1)$$

respectively. Then, denote the the distance between the front axle and the back axle of the vehicle by d, the h-step dynamics of the vehicle are expressed as

$$p_x(\tau + 1) = p_x(\tau) + b\big(v(\tau), w(\tau)\big)\cos\big(\theta(\tau)\big)$$
$$p_y(\tau + 1) = p_y(\tau) + b\big(v(\tau), w(\tau)\big)\sin\big(\theta(\tau)\big)$$
$$\theta(\tau + 1) = \theta(\tau) + \sin^{-1}\Big(f\sin\big(w(\tau)\big)/d\Big)$$
$$v(\tau + 1) = v(\tau) + ha(\tau). \quad (2)$$

Thus, the vehicle dynamic equation is given by

$$\begin{bmatrix} p_x(\tau+1) \\ p_y(\tau+1) \\ \theta(\tau+1) \\ v(\tau+1) \end{bmatrix} = f\left(\begin{bmatrix} p_x(\tau) \\ p_y(\tau) \\ \theta(\tau) \\ v(\tau) \end{bmatrix}, \begin{bmatrix} w(\tau) \\ a(\tau) \end{bmatrix} \right) \triangleq f\big(x(\tau), u(\tau)\big). \text{(3)}$$

B. Objective Function

In the objective function, the position tracking error and velocity tracking error are considered. Note that the position tracking and the velocity tracking are separated to avoid the pulling effect as detailed in [25]. 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:

$$\ell_\tau\big(x(\tau), u(\tau)\big) = q_1 d^r\big(x(\tau), l^r_\tau\big) + q_2 \big\| M_v x(\tau) - v^r_\tau \big\|^2$$
$$+ r_1 u(\tau)^T M_w u(\tau) + r_2 u(\tau)^T M_a u(\tau), \text{ (4)}$$

where $\|\cdot\|$ denotes the Euclidean norm operator; q_1, q_2, r_1, and r_2 represent the weighting parameters of the position tracking error, velocity tracking error, steering wheel input, and acceleration, respectively; l^r denotes the polyline of the reference trajectory and $d^r(x_k, l^r)$ denotes the Euclidean distance between the vehicle and the corresponding polyline; M_v is the matrix to extract the velocity of the vehicle from the state vector, i.e., $M_v = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix}$; v^r is the reference velocity; M_w and M_a are two matrices given by $M_w = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$ and $M_a = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$.

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

$$\ell_\tau\big(x(\tau), u(\tau)\big) = \begin{bmatrix} x(\tau) \\ u(\tau) \end{bmatrix}^T C \begin{bmatrix} x(\tau) \\ u(\tau) \end{bmatrix} - 2 \begin{bmatrix} x(\tau) \\ u(\tau) \end{bmatrix}^T Cr. \text{ (5)}$$

where

$$C = \text{diag}\{0, q_1, 0, q_2, r_1, r_2\}$$
$$r = \begin{bmatrix} 0 & 0 & 0 & v_r & 0 & 0 \end{bmatrix}^T. \text{ (6)}$$

C. Constraints

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

$$-w_{max} \le u(\tau)^T V_w \le w_{max}, \text{ (7)}$$

with $V_w = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, and w_{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 $\tau = 0, 1, \cdots, T$,

$$a_{maxdec} \le u(\tau)^T V_a \le a_{maxacc}, \text{ (8)}$$

where $V_a = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$, and a_{maxdec} and a_{maxacc} 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 head angle at the time stamp τ, the rotational matrix is given by

$$R(\tau) = \begin{bmatrix} \cos\theta(\tau) & -\sin\theta(\tau) \\ \sin\theta(\tau) & \cos\theta(\tau) \end{bmatrix}. \text{ (9)}$$

Then, denote e_a and e_b as the length of semi-major and semi-minor axes corresponding to the ellipse, and $d_{xy}(\tau)$ as the position difference between the ego vehicle and the obstacle, and then the obstacle avoidance constraint is formulated as

$$h(x(\tau)) = 1 - d_{xy}(\tau)^T A(\tau) d_{xy}(\tau) \le 0, \text{ (10)}$$

with

$$A(\tau) = R(\tau) \begin{bmatrix} e_a^2 & 0 \\ 0 & e_b^2 \end{bmatrix} R(\tau)^T. \text{ (11)}$$

III. ADMM ALGORITHM FOR iLQR-BASED MOTION PLANNING PROBLEM

In this section, we present a general approach for the motion planning problem. In this following text, we propose the use of a nonlinear system model with n system state variables and m control input variables. Also, for the sake of simplicity, we utilize the vector concatenation symbol, i.e., the notation $a = (a_1, a_2, \cdots, a_N)$ is equivalent to the notation $a = [a_1^T, a_2^T, \cdots, a_N^T]^T$.

A. Motion Planning with iLQR

Generally, a motion planning problem in autonomous driving can be formulated as the standard iLQR problem, where the optimization problem is expressed as

$$\underset{\big(x(\tau), u(\tau)\big) \in \mathbb{R}^n \times \mathbb{R}^m}{\text{minimize}} \quad \phi(x(T)) + \sum_{\tau=0}^{T-1} \ell_\tau\big(x(\tau), u(\tau)\big)$$
$$\text{subject to} \quad x(\tau+1) = f\big(x(\tau), u(\tau)\big)$$
$$\tau = 0, 1, \cdots, T-1$$
$$x(0) = x_0, \text{ (12)}$$

where $\phi(x(T))$ denotes the terminal cost function to the system state vector in the last time stamp.

The principle of dynamic programming reduces the minimization over a sequence of control actions to a sequence of minimization over one single control action, which is summarized as the Bellman equation characterized by

$$V_\tau\big(x(\tau)\big) = \min_{u(\tau)} \Big\{ \ell_\tau\big(x(\tau), u(\tau)\big) + V_{\tau+1}\Big(f\big(x(\tau), u(\tau)\big)\Big) \Big\},$$
$$\text{(13)}$$

3

with $V_\tau(x(\tau))$ and $V_{\tau+1}(f(x(\tau), u(\tau)))$ denoting the minimum cost-to-go that starts from τ and $\tau + 1$, respectively.

For the first step of iLQR, a feasible nominal trajectory $\{\hat{u}(\tau), \hat{x}(\tau)\}_{\tau=0}^T$ is generated by passing the nominal input trajectory to the nonlinear system with the system initial state variables. Then, the second step is the so-called backward pass. With the value function at the final step, 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 (13), 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 τ in the two adjacent iterations. The perturbed Q-function can be represented by

$$
\begin{aligned}
&Q_\tau\big(\delta x(\tau), \delta u(\tau)\big) \\
&= \ell_\tau\big(x(\tau) + \delta x(\tau), u(\tau) + \delta u(\tau)\big) - \ell_\tau\big(x(\tau), u(\tau)\big) \\
&+ V_{\tau+1}\Big(f\big(x(\tau) + \delta x(\tau), u(\tau) + \delta u(\tau)\big)\Big) \\
&- V_{\tau+1}\Big(f\big(x(\tau), u(\tau)\big)\Big).
\end{aligned} \tag{14}
$$

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

$$
\begin{aligned}
&Q_\tau\big(\delta x(\tau), \delta u(\tau)\big) \\
&\approx \frac{1}{2}
\begin{bmatrix} 1 \\ \delta x(\tau) \\ \delta u(\tau) \end{bmatrix}^T
\begin{bmatrix}
0 & (Q_\tau^T)_x & (Q_\tau^T)_u \\
(Q_\tau)_x & (Q_\tau)_{xx} & (Q_\tau)_{xu} \\
(Q_\tau)_u & (Q_\tau)_{ux} & (Q_\tau)_{uu}
\end{bmatrix}
\begin{bmatrix} 1 \\ \delta x(\tau) \\ \delta u(\tau) \end{bmatrix},
\end{aligned} \tag{15}
$$

with

$$
\begin{aligned}
(Q_\tau)_x &= (\ell_\tau)_x + f_x^T (V_{\tau+1})_x \\
(Q_\tau)_u &= (\ell_\tau)_u + f_u^T (V_{\tau+1})_u \\
(Q_\tau)_{xx} &= (\ell_\tau)_{xx} + f_x^T (V_{\tau+1})_{xx} f_x + (V_{\tau+1})_x \cdot f_{xx} \\
(Q_\tau)_{ux} &= (\ell_\tau)_{ux} + f_u^T (V_{\tau+1})_{xx} f_x + (V_{\tau+1})_x \cdot f_{ux} \\
(Q_\tau)_{uu} &= (\ell_\tau)_{uu} + f_u^T (V_{\tau+1})_{xx} f_u + (V_{\tau+1})_x \cdot f_{uu},
\end{aligned} \tag{16}
$$

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.

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

$$
\delta u(\tau)^* = \underset{\delta u(\tau)}{\arg\min} \quad Q_\tau\big(\delta x(\tau), \delta u(\tau)\big), \tag{17}
$$

and then we further formulate the optimal input perturbation as

$$
\delta u(\tau)^* = k(\tau) + K(\tau)\delta x(\tau), \tag{18}
$$

where $k(\tau)$ and $K(\tau)$ are considered as the feedforward modification and feedback gain at the time stamp τ. Subsequently, it yields that

$$
k(\tau) = -(Q_\tau)_{uu}^{-1} (Q_\tau)_u \tag{19a}
$$

$$
K(\tau) = -(Q_\tau)_{uu}^{-1} (Q_\tau)_{ux}. \tag{19b}
$$

Substitute (19a) and (19b) to the Taylor expansion of perturbed Q-function, and then the difference, gradient, and Hessian of the value function are given by

$$
\Delta V = -\frac{1}{2} k(\tau)^T (Q_\tau)_{uu} k(\tau) \tag{20a}
$$

$$
(V_\tau)_x = (Q_\tau)_x - K(\tau)^T (Q_\tau)_{uu} k(\tau) \tag{20b}
$$

$$
(V_\tau)_{xx} = (Q_\tau)_{xx} - K(\tau)^T (Q_\tau)_{uu} K(\tau). \tag{20c}
$$

As follows, the backward pass is recursively conducted by calculating (19a)-(19b) and (20b)-(20c), until the first time stamp is reached.

Then, after the backward pass is completed, the third step is carried out, which is the so-called forward pass. In the forward pass, the actual trajectory will be generated using the system dynamics and the control action pre-determined by (19a) and (19b). Here, we have

$$
\begin{aligned}
u(\tau) &= \hat{u}(\tau) + k(\tau) + K(\tau)\big(x(\tau) - \hat{x}(\tau)\big) \\
x(\tau + 1) &= f\big(x(\tau), u(\tau)\big).
\end{aligned} \tag{21}
$$

After an iteration for all three steps is completed, the nominal trajectory $(\hat{x}, \hat{u})$ will be updated to a new feasible trajectory. In the end, it will converge to the optimal trajectory asymptotically.

B. ADMM Algorithm for iLQR

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

$$
\begin{aligned}
\underset{(x(\tau), u(\tau)) \in \mathbb{R}^n \times \mathbb{R}^m}{\text{minimize}} \quad & \phi\big(x(T)\big) + \sum_{\tau=0}^{T-1} \ell_\tau\big(x(\tau), u(\tau)\big) \\
\text{subject to} \quad & x(\tau + 1) = f\big(x(\tau), u(\tau)\big) \\
& g(y) \geq 0 \\
& \tau = 0, 1, \cdots, T - 1,
\end{aligned} \tag{22}
$$

where the vector y is defined as

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

and the function g denotes the inequality constraints included in the optimization problem. Notably, the symbol $\geq$ represents the generalized "greater or equal than" in this work. For instance, in the vector case, the symbol $\geq$ denotes the element-wisely "greater or equal than"; 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 $\mathbb{B}$ is defined as

$$
\delta_{\mathbb{B}}(B) = \begin{cases} 0 & \text{if } B \in \mathbb{B} \\ \infty & \text{otherwise.} \end{cases} \tag{23}
$$

We further define the sets

$$
\begin{aligned}
\mathscr{F} &= \Big\{ y = \big(x(0), u(0), x(1), u(1), \cdots, x(T), u(T)\big) \,\Big| \\
& \quad x(\tau + 1) = f\big(x(\tau), u(\tau)\big), \forall \tau = 0, 1, \cdots, T \Big\} \\
\tilde{\mathscr{G}} &= \Big\{ y \in \mathbb{R}^{(n+m)T} \,\Big|\, g(y) \geq 0 \Big\}.
\end{aligned} \tag{24}
$$

Then the optimization problem can be equivalently denoted by

$$
\begin{aligned}
\underset{\left(x(\tau),u(\tau)\right)\in\mathbb{R}^n\times\mathbb{R}^m}{\text{minimize}} \quad & \phi(x(T)) + \sum_{\tau=0}^{T-1} \ell_\tau\big(x(\tau),u(\tau)\big) \\
& + \delta_{\mathscr{F}}\left(y\right) + \delta_{\mathscr{G}}\left(y\right).
\end{aligned} \tag{25}
$$

By introducing the consensus variable $\tilde{z}$, (25) can be further expressed as

$$
\begin{aligned}
\underset{\left(x(\tau),u(\tau)\right)\in\mathbb{R}^n\times\mathbb{R}^m}{\text{minimize}} \quad & \phi\big(x(T)\big) + \sum_{\tau=0}^{T-1} \ell_\tau\big(x(\tau),u(\tau)\big) \\
& + \delta_{\mathscr{F}}\left(y\right) + \delta_{\tilde{\mathscr{G}}}\left(\tilde{z}\right) \\
\text{subject to} \quad & y - \tilde{z} = 0.
\end{aligned} \tag{26}
$$

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

Then the optimization problem is converted to

$$
\begin{aligned}
\underset{\left(x(\tau),u(\tau)\right)\in\mathbb{R}^n\times\mathbb{R}^m}{\text{minimize}} \quad & \phi(x(T)) + \sum_{\tau=0}^{T-1} \ell_\tau\big(x(\tau),u(\tau)\big) \\
& + \delta_{\mathscr{F}}\left(y\right) + \delta_{\mathscr{G}}\left(z\right) \\
\text{subject to} \quad & \mathcal{A}y - z = 0,
\end{aligned} \tag{27}
$$

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 $\mathscr{G}$ is presented by

$$
\mathscr{G} = \{z = \mathcal{A}\tilde{z}\,|\,\mathcal{G}(\tilde{z}) \geq 0\}. \tag{28}
$$

Define the augmented Lagrangian function of the optimization problem as

$$
\begin{aligned}
\mathcal{L}_\sigma(y,z;\lambda) = {} & \phi(x(T)) + \sum_{\tau=0}^{T-1} \ell_\tau\big(x(\tau),u(\tau)\big) + \delta_{\mathscr{F}}\left(y\right) \\
& + \delta_{\mathscr{G}}\left(z\right) + \frac{\sigma}{2}\|\mathcal{A}y - z + \sigma^{-1}\lambda\|^2 - \frac{1}{2\sigma}\|\lambda\|^2,
\end{aligned} \tag{29}
$$

where $\sigma \in \mathbb{R}$ is the penalty parameter of the augmented Lagrangian function, and $\lambda \in \mathbb{R}^p$ is the Lagrangian multiplier. Then the ADMM iterations are listed as

$$
\begin{aligned}
y^{k+1} &= \underset{y}{\arg\min}\ \mathcal{L}_\sigma\big(y, z^k; \lambda^k\big) \\
z^{k+1} &= \underset{z}{\arg\min}\ \mathcal{L}_\sigma\big(y^{k+1}, z; \lambda^k\big) \\
\lambda^{k+1} &= \lambda^k + \sigma\big(\mathcal{A}y^{k+1} - z^{k+1}\big),
\end{aligned} \tag{30}
$$

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:

$$
\begin{aligned}
\underset{\left(x(\tau),u(\tau)\right)\in\mathbb{R}^n\times\mathbb{R}^m}{\text{minimize}} \quad & \phi(x(T)) + \sum_{\tau=0}^{T-1} \ell_\tau\big(x(\tau),u(\tau)\big) \\
& + \frac{\sigma}{2}\|\mathcal{A}y - z^k + \sigma^{-1}\lambda^k\|^2 \\
\text{subject to} \quad & x(\tau+1) = f\big(x(\tau),u(\tau)\big) \\
& \tau = 0,1,\cdots,T-1.
\end{aligned} \tag{31}
$$

Furthermore, to facilitate the deployment of the iLQR method, the optimization problem (31) can be equivalently expressed as

$$
\begin{aligned}
\underset{\left(x(\tau),u(\tau)\right)\in\mathbb{R}^n\times\mathbb{R}^m}{\text{minimize}} \quad & \phi(x(T)) + \sum_{\tau=0}^{T-1}\bigg(\ell_\tau\big(x(\tau),u(\tau)\big) \\
& + \frac{\sigma}{2}\left\| \hat{\mathcal{A}}(\tau)\begin{bmatrix} x(\tau) \\ u(\tau) \end{bmatrix} - \begin{bmatrix} z_x^k(\tau) \\ z_u^k(\tau) \end{bmatrix} + \sigma^{-1}\begin{bmatrix} \lambda_x^k(\tau) \\ \lambda_u^k(\tau) \end{bmatrix} \right\|^2 \bigg) \\
\text{subject to} \quad & x(\tau+1) = f\big(x(\tau),u(\tau)\big) \\
& \tau = 0,1,\cdots,T-1,
\end{aligned} \tag{32}
$$

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

Remark 1. Since only dynamic constraints are included in the optimization problem, the optimization problem (32) can be trivially solved by the iLQR algorithm.

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

$$
\begin{aligned}
\underset{z\in\mathbb{R}^{pT}}{\text{minimize}} \quad & \|\mathcal{A}y^{k+1} + \sigma^{-1}\lambda^k - z\|^2 \\
\text{subject to} \quad & z \in \mathscr{G}.
\end{aligned} \tag{33}
$$

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

$$
\begin{aligned}
\underset{z(\tau)\in\mathbb{R}^p}{\text{minimize}} \quad & \left\| \hat{\mathcal{A}}(\tau)\begin{bmatrix} x^{k+1}(\tau) \\ u^{k+1}(\tau) \end{bmatrix} + \sigma^{-1}\lambda^k(\tau) - z(\tau) \right\|^2 \\
\text{subject to} \quad & g_\tau\big(z(\tau)\big) \geq 0,
\end{aligned} \tag{34}
$$

where

$$
\begin{aligned}
z &= \big(z(0), z(1), \cdots, z(T)\big) \\
\lambda &= \big(\lambda(0), \lambda(1), \cdots, \lambda(T)\big).
\end{aligned} \tag{35}
$$

Remark 2. In each time stamp, the sub-problem (34) represents a non-convex projection optimization problem. Even though finding its global optimum cannot be guaranteed, the constraints generally cover the box constraints and Euclidean norm constraints in motion planning tasks, where the global optimum can be intuitively obtained.

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

$$
\lambda^{k+1} = \lambda^k + \sigma\big(\mathcal{A}y^{k+1} - z^{k+1}\big).
$$

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 ADMM algorithm for iLQR-based motion planning is given in Algorithm 1.

Algorithm 1 ADMM Algorithm for iLQR-Based Motion Planning

Require: System dynamic model f;
          objective function ϕ and ℓ;
          inequality constraints g;
          initial system state vector x_0;
          penalty parameter σ;
          the maximum iteration number of ADMM $N_{\max}^{\text{ADMM}}$;
          the maximum iteration number of iLQR $N_{\max}^{\text{iLQR}}$.
1: Generate the initial trajectory $(\hat{x}(\tau), \hat{u}(\tau))$.
2: Formulate the optimization problem into the iLQR solvable form by (32).
3: **for** $i = 1, 2, \cdots, N_{\max}^{\text{ADMM}}$ **do**
4:    **(ADMM First Iteration)**
5:    **for** $j = 1, 2, \cdots, N_{\max}^{\text{iLQR}}$ **do**
6:       Perform the iLQR backward pass by (19a)-(19b) and (20b)-(20c).
7:       Perform the iLQR forward pass by (21).
8:       **if** iLQR stopping criterion is satisfied **then**
9:          **break**
10:      **end if**
11:   **end for**
12:   **(ADMM Second Iteration)**
13:   Solve the optimization problem (34).
14:   **(Lagrange Multiplier Update)**
15:   Update the Lagrangian multiplier by (36).
16:   **if** ADMM stopping criterion is satisfied **then**
17:      **break**
18:   **end if**
19: **end for**

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 iLQR iterations is usually chosen as the amount of change of the objective function. There are numerous choices of the stopping criterion in terms of the ADMM algorithm, such as the primal-dual residual error. However, in our case, we only consider the primal residual error in the stopping criterion because the second iteration always returns a feasible trajectory to the constrained iLQR problem.
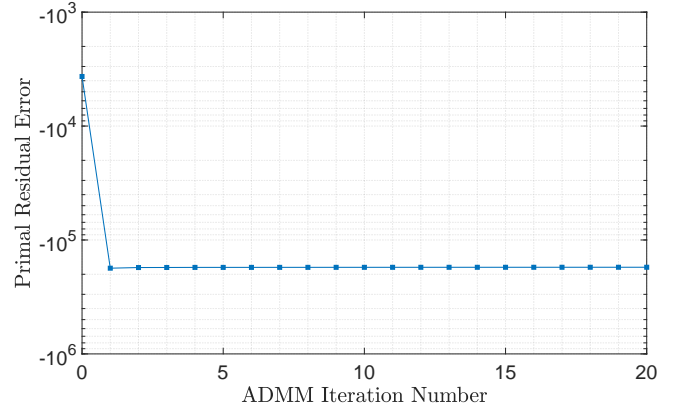

Fig. 1. Primal residual error during ADMM iterations in the static obstacle avoidance scenario.
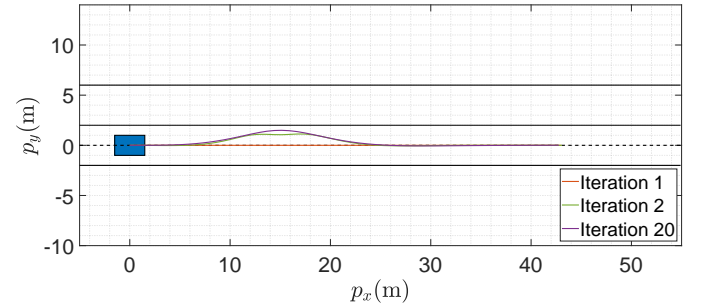

Fig. 2. Change of trajectory during iLQR iterations in the static obstacle avoidance scenario.

IV. ILLUSTRATIVE EXAMPLE

Next, an illustrative example of autonomous driving is presented to validate the effectiveness of the proposed development. Two scenarios are considered here, including one static obstacle avoidance problem and one dynamic obstacle avoidance problem. The optimization algorithm is implemented in a laptop platform with Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz, the optimization is conducted in Python 3.7 with the Numba package, and the figures are plotted in MATLAB 2019b.

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 rectangle, and collision regions are presented by the ellipse with dash lines in yellow color.

In both scenarios, the upper bound and lower bound for the steering angle are set as $0.6\,\text{rad}$ and $-0.6\,\text{rad}$, respectively. For the acceleration/deceleration, the upper bound and lower bound are given by $3\,\text{m/s}^2$ and $-3\,\text{m/s}^2$, respectively. The length and width of both the ego vehicle and the surrounding vehicles are $3\,\text{m}$ and $2\,\text{m}$, respectively. For the ellipse representing the collision region, the length of semi-major and semi-minor axes is $5\,\text{m}$ and $2.5\,\text{m}$. respectively. The penalty parameter σ of the augmented Lagrangian function is set as
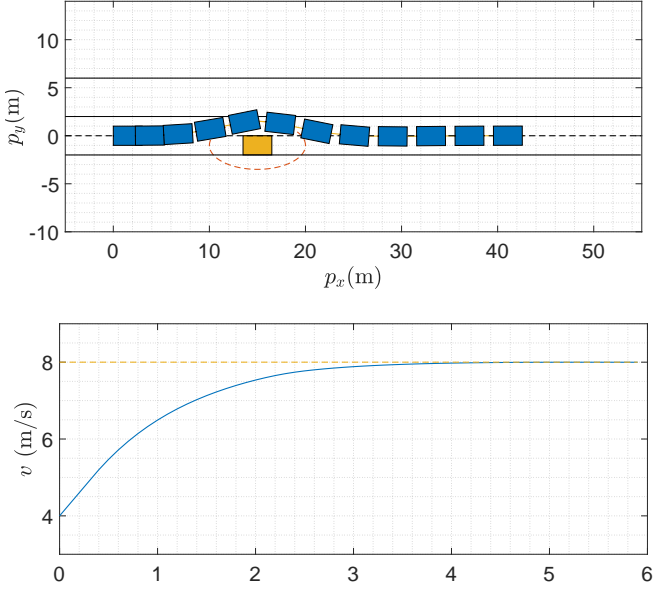
6

Fig. 3. Trajectory and velocity of the vehicle in the static obstacle avoidance scenario.
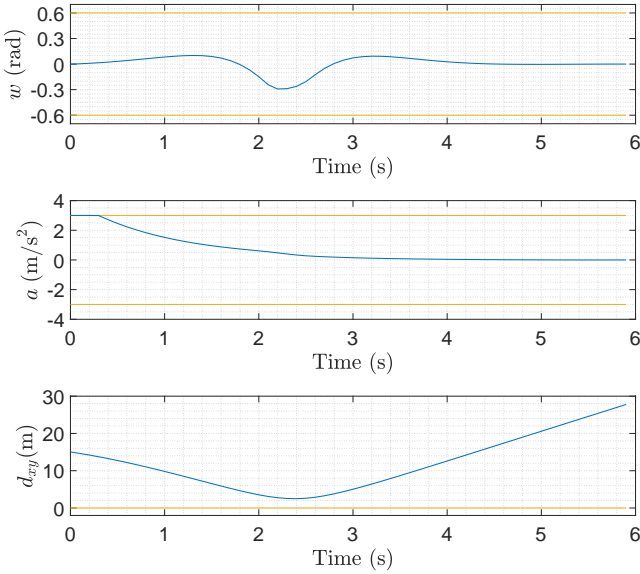

Fig. 4. Steering angle & acceleration/deceleration of the ego vehicle and distance from the ego vehicle to the obstacle in the static obstacle avoidance scenario.

10. Also, the prediction horizon is 60, and samping time is $0.1\,\mathrm{s}$. The maximum iteration numbers of the iLQR algorithm and the ADMM algorithm are set as 100 and 20, respectively.

A. Scenario 1: Static Obstacle Avoidance

In the first scenario, a static obstacle avoidance problem is investigated. Here, we assume there is one vehicle parking on the street with a coordinate of $(15\,\mathrm{m}, -1\,\mathrm{m})$. For the ego vehicle, the initial position is set as $(0\,\mathrm{m}, 0\,\mathrm{m})$, the initial steering angle is $0\,\mathrm{rad}$, and the initial velocity is $4\,\mathrm{m/s}$. The reference for the position is set as $p_y = 0\,\mathrm{m}$, and the reference for the velocity is $8\,\mathrm{m/s}$.

The primal residual error during the ADMM iterations is illustrated in Fig. 1. With the proposed approach, the change of trajectory for the ego vehicle during ADMM iterations is plotted in Fig. 2. To visualize the change clearly, only the trajectory 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. 3 illustrates the trajectory and velocity of the ego vehicle. Note that in this scenario, the position of the ego vehicle is plotted every $0.5\,\mathrm{s}$. 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 road). From the 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. 4 is given, where the steering angle and acceleration/deceleration of the ego vehicle are depicted. Also, the distance from the ego vehicle to the obstacle (denoted by d_{xy}) is presented from this figure. It can be seen that the steering angle and the acceleration/deceleration 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

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 is a vehicle in front of the ego vehicle, which has an initial position of $(20\,\mathrm{m}, 0\,\mathrm{m})$ and a constant velocity of $3\,\mathrm{m/s}$. Also, it is assumed that 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 $(0\,\mathrm{m}, 4\,\mathrm{m})$ and it moves along the road with a velocity of $6\,\mathrm{m/s}$. Note that the initial position of the ego vehicle is $(0\,\mathrm{m}, 0\,\mathrm{m})$, the initial steering angle is $0\,\mathrm{rad}$, and the initial velocity is $8\,\mathrm{m/s}$. The reference for the position is set as $p_y = 4\,\mathrm{m}$. In this case, velocity tracking is not considered in the objective function.

Similarly, the primal residual error during the ADMM iterations is illustrated in Fig. 5. The change of trajectory for the ego vehicle during ADMM iterations is depicted in Fig. 6, where its trajectory in iteration 1, iteration 2, and iteration 20 is plotted. From this figure, the asymptotic convergence of the trajectory is validated. Subsequently, the trajectory of all the vehicles and the velocity of the ego vehicle is given in Fig 3. In this case, the position of all the vehicles is plotted every 0.1s. 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 velocity profile, it can be seen that the ego vehicle accelerates at the beginning, and converges to around $8.08\,\mathrm{m/s}$.
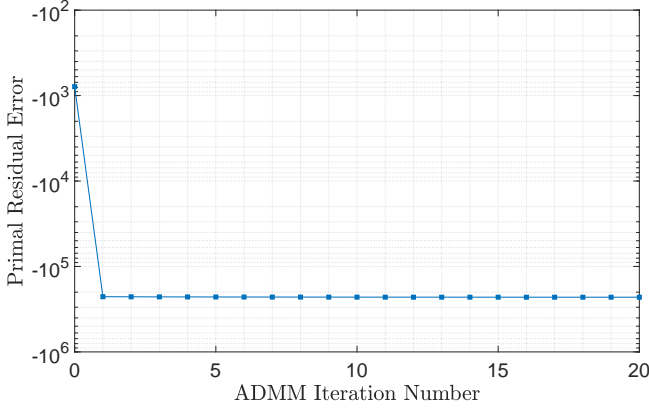
Fig. 5. Primal residual error during ADMM iterations in the dynamic obstacle avoidance scenario.
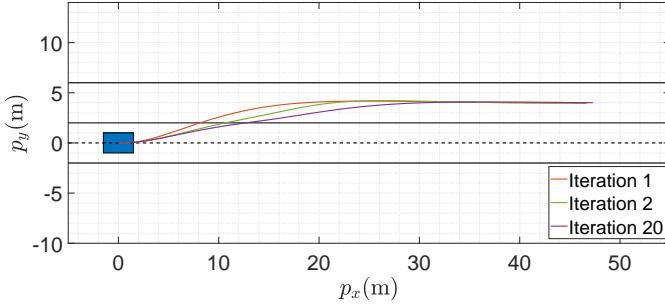

Fig. 6. Change of trajectory during iLQR iterations in the dynamic obstacle avoidance scenario.

Additionally, Fig. 4 is shown to validate the constraint satisfaction conditions. As indicated from this figure, the steering angle and acceleration/deceleration of the ego vehicle are all constrained within the predefined upper/lower bound. Also, as the distance from the ego vehicle to the obstacle illustrates, the ego vehicle avoids the surrounding vehicles successfully in this scenario.

C. Discussion on Computation Time

For the optimization in both scenarios, 5 trials with a prediction horizon of $T = 60$ are performed, and the computation time by using our proposed approach (denoted by Method 1) is recorded as shown in Table I. The average time in these two scenarios is calculated, which is given by $0.1306\,\text{s}$ and $0.1219\,\text{s}$, respectively.

Apart from this, a comparison study (denoted by Method 2) is conducted using the constrained iLQR algorithm with the logarithmic barrier function. In Scenario 1, if the initial velocity of the ego vehicle is set as $4\,\text{m/s}$, a feasible trajectory cannot be obtained. Thus, we set the initial velocity as $0\,\text{m/s}$. Also, in Scenario 2, if the initial velocity of the ego vehicle is set as $8\,\text{m/s}$, a feasible trajectory cannot be obtained because it will collide with the vehicle in the front. Hence, we set the initial velocity as $4\,\text{m/s}$ in this case. Subsequently, the computation time is recorded and presented in Table I. Notably, the average time in these two scenarios is given by $0.4427\,\text{s}$ and $0.6820\,\text{s}$, respectively.
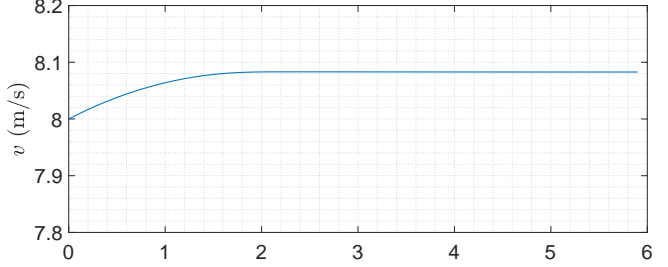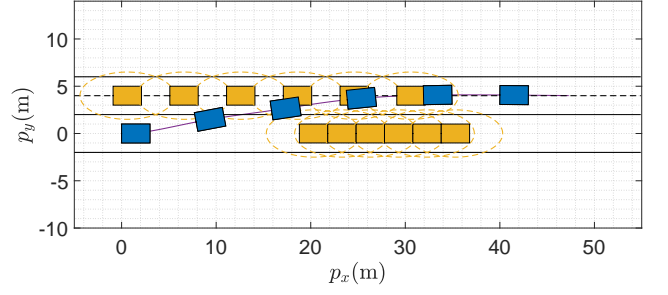

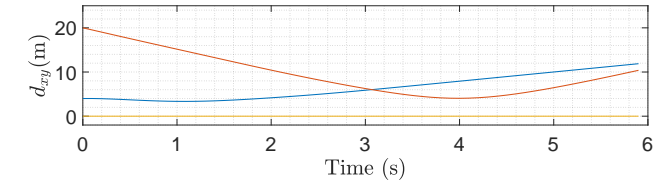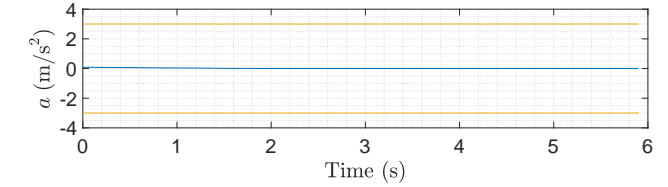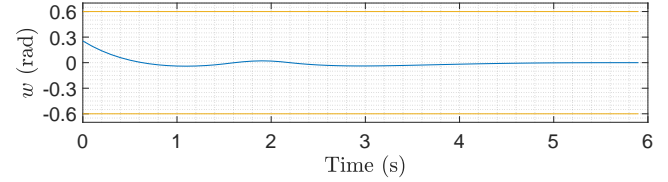Fig. 7. Trajectory and velocity of the vehicle in the dynamic obstacle avoidance scenario.

Fig. 8. Steering angle & acceleration/deceleration of the ego vehicle and distance from the ego vehicle to the obstacles in the dynamic obstacle avoidance scenario.

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, this work leads to less computation effort and avoids the feasibility requirement for the trajectory at the first iteration.

As illustrated from Fig. 1 and Fig. 5, a feasible trajectory is obtained after few ADMM iterations only, and thus the computation efficiency of the algorithm can be further enhanced by choosing a loose stopping criterion. Also, the prediction horizon in the given examples is large enough to realize the on-road driving tasks. It is also possible to reduce the computation time by choosing a smaller prediction horizon.

8

TABLE I
COMPUTATION TIME IN THE TWO SCENARIOS

	Computation Time in Scenario 1		Computation Time in Scenario 2	
	Method 1	Method 2	Method 1	Method 2
Trial 1	0.1411 s	0.4448 s	0.1366 s	0.7147 s
Trial 2	0.1386 s	0.4399 s	0.1107 s	0.6728 s
Trial 3	0.1187 s	0.4412 s	0.1227 s	0.6592 s
Trial 4	0.1241 s	0.4468 s	0.1206 s	0.6872 s
Trial 5	0.1305 s	0.4409 s	0.1217 s	0.6762 s

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 with small sizes, and these sub-problems can be efficiently solved in a parallel manner. 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, Y. Zheng, K. Li, Y. Wu, J. K. Hedrick, F. Gao, and H. Zhang, "Dynamical modeling and distributed control of connected and automated vehicles: Challenges and opportunities," *IEEE Intelligent Transportation Systems Magazine*, vol. 9, no. 3, pp. 46–58, 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 Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1624–1639, 2011.

[3] F.-Y. Wang, "Parallel control and management for intelligent transportation systems: Concepts, architectures, and applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 630–638, 2010.

[4] D. González, J. Pérez, V. Milanés, and F. Nashashibi, "A review of motion planning techniques for automated vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1135–1145, 2015.

[5] Z. Wang, Y. Bian, S. E. Shladover, G. Wu, S. E. Li, and M. J. Barth, "A survey on cooperative longitudinal motion control of multiple connected and automated vehicles," *IEEE Intelligent Transportation Systems Magazine*, vol. 12, no. 1, pp. 4–24, 2019.

[6] K. Bergman, O. Ljungqvist, and D. Axehill, "Improved optimization of motion primitives for motion planning in state lattices," in *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2019, pp. 2307–2314.

[7] 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 Transactions on Intelligent Transportation Systems*, 2019.

[8] Y. Guan, Y. Ren, S. E. Li, Q. Sun, L. Luo, and K. Li, "Centralized cooperation for connected and automated vehicles at intersections by proximal policy optimization," *IEEE Transactions on Vehicular Technology*, 2020.

[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 Intelligent Transport Systems*, vol. 14, no. 5, pp. 297–305, 2020.

[10] L. Chen *et al.*, "A reinforcement learning-based adaptive path tracking approach for autonomous driving," *IEEE Transactions on Vehicular Technology*, 2020.

[11] E. F. Camacho and C. B. Alba, *Model Predictive Control*. Berlin: Springer Science & Business Media, 1998.

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

[13] C. Liu, S. Lee, S. Varnhagen, and H. E. Tseng, "Path planning for autonomous vehicles using model predictive control," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 174–179.

[14] K. Li, Y. Bian, S. E. Li, B. Xu, and J. Wang, "Distributed model predictive control of multi-vehicle systems with switching communication topologies," *Transportation Research Part C: Emerging Technologies*, vol. 118, p. 102717, 2020.

[15] J. Ji, A. Khajepour, W. W. Melek, and Y. Huang, "Path planning and tracking for vehicle collision avoidance based on model predictive control with multiconstraints," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 2, pp. 952–964, 2016.

[16] O. Celik, H. Abdulsamad, and J. Peters, "Chance-constrained trajectory optimization for non-linear systems with unknown stochastic dynamics," *arXiv preprint arXiv:1906.11003*, 2019.

[17] C. Hubmann, M. Becker, D. Althoff, D. Lenz, and C. Stiller, "Decision making for autonomous driving considering interaction and uncertain prediction of surrounding vehicles," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 1671–1678.

[18] 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 *Proceedings of International Conference on Control, Automation and Systems*, 2019, pp. 974–979.

[19] 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," *arXiv preprint arXiv:2006.07907*, 2020.

[20] D. Mayne, "A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems," *International Journal of Control*, vol. 3, no. 1, pp. 85–95, 1966.

[21] Z. Xie, C. K. Liu, and K. Hauser, "Differential dynamic programming with nonlinear constraints," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 695–702.

[22] A. Nagariya and S. Saripalli, "An iterative LQR controller for off-road and on-road vehicles using a neural network dynamics model," *arXiv preprint arXiv:2007.14492*, 2020.

[23] Y. Tassa, N. Mansard, and E. Todorov, "Control-limited differential dynamic programming," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 1168–1175.

[24] J. Chen, W. Zhan, and M. Tomizuka, "Constrained iterative LQR for on-road autonomous driving motion planning," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2017, pp. 1–7.

[25] ——, "Autonomous driving motion planning with constrained iterative LQR," *IEEE Transactions on Intelligent Vehicles*, vol. 4, no. 2, pp. 244–254, 2019.

[26] Y. Pan, Q. Lin, H. Shah, and J. M. Dolan, "Safe planning for self-driving via adaptive constrained ILQR," *arXiv preprint arXiv:2003.02757*, 2020.

[27] H. Zheng, R. R. Negenborn, and G. Lodewijks, "Fast admm for distributed model predictive control of cooperative waterborne agvs," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 4, pp. 1406–1413, 2016.

[28] J. Ma, Z. Cheng, X. Zhang, M. Tomizuka, and T. H. Lee, "Optimal decentralized control for uncertain systems by symmetric Gauss-Seidel semi-proximal ALM," *arXiv preprint arXiv:2001.00306*, 2020.

[29] ——, "On symmetric Gauss-Seidel ADMM algorithm for H_∞ guaranteed cost control with convex parameterization," 2020.

[30] Z. Xu, S. De, M. Figueiredo, C. Studer, and T. Goldstein, "An empirical study of ADMM for nonconvex problems," *arXiv preprint arXiv:1612.03349*, 2016.

[31] M. Hong, Z.-Q. Luo, and M. Razaviyayn, "Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems," *SIAM Journal on Optimization*, vol. 26, no. 1, pp. 337–364, 2016.

[32] J. Wang and L. Zhao, "Nonconvex generalization of ADMM for nonlinear equality constrained problems," *arXiv preprint arXiv:1705.03412*, 2017.

[33] Y. Wang, W. Yin, and J. Zeng, "Global convergence of ADMM in nonconvex nonsmooth optimization," *Journal of Scientific Computing*, vol. 78, no. 1, pp. 29–63, 2019.