2018 21st International Conference on Intelligent Transportation Systems (ITSC)
Maui, Hawaii, USA, November 4-7, 2018

Parallel Optimal Control for Cooperative Automation of Large-scale Connected Vehicles via ADMM

Zhitao Wang, Yang Zheng, Shengbo Eben Li*, Keyou You, and Keqiang Li

Abstract— **This paper proposes a parallel optimization algorithm for cooperative automation of large-scale connected vehicles. The task of cooperative automation is formulated as a centralized optimization problem taking the whole decision space of all vehicles into account. Considering the uncertainty of the environment, the problem is solved in a receding horizon fashion. Then, we employ the alternating direction method of multipliers (ADMM) to solve the centralized optimization in a parallel way, which scales more favorably to large-scale instances. Also, Taylor series is used to linearize nonconvex constraints caused by coupling collision avoidance constraints among interactive vehicles. Simulations with two typical traffic scenes for multiple vehicles demonstrate the effectiveness and efficiency of our method.**

Index Terms— **Cooperative automation, connected vehicles, ADMM, optimal control.**

I. Introduction

With the rapid development of communication, computation, and automation technologies, connected vehicles become one key component in future transportation systems. Cooperative automation of connected vehicles is expected to be realized using powerful computing ability (*e.g.*, cloud computation) and advanced communication technology (*e.g.*, V2V, V2I) [1]. In recent years, coordinating multiple connected vehicles has received considerable research attention due to its potential to improve traffic efficiency and safety [2], [3]. One main challenge in cooperative automation is to ensure that coupling constraints among connected vehicles are satisfied and the computation is efficient [4], [5].

To coordinate multiple connected vehicles, one natural and promising approach is to use a distributed framework where each vehicle optimizes its own control variable based on local information. Recently, some distributed control laws for platooning of connected vehicles have been proposed, which employ matrix decomposition to decouple coupling constraints in the system. For example, Zheng *et al.* introduced a distributed model predictive controller for heterogeneous platoons with unidirectional topologies in which closed-loop stability and robustness were analyzed [6]. Wu *et al.* proposed a distributed sliding mode algorithm based on a topologically structured function [7]. To deal with coupling constraints in planning problems, an effective solution is to use the priority approach, which has been widely applied in the literature [8]. In principle, planning problems are solved sequentially according to some prioritization to decouple the coupling constraints, thus achieving a distributed solution. For example, Kuwata *et al.* introduced a cooperative distributed robust safe optimization approach for systems with coupled objectives and constraints, where the optimization was carried out in a sequential way [5]. Sycara *et al.* applied a sparse method to improve the convergence performance, where constraints were discovered probabilistically [8]. Other techniques, such as graph searching, are also proposed for solving this problem [9]. LaValle *et al.* established a solution mapping between the planning problem and network-flow, and then applied integer linear programming to optimize the objective [10]. In short, to deal with coupling constraints, the distributed formation of the problem needs to be assigned appropriately, *i.e.*, the topology of platoons and the priorities of the priority approach [3], [8], [11].

Another intuitive way of executing a cooperative task is to establish a centralized system to optimize the global decision space of all vehicles. However, the centralized formulation is computationally expensive when the system size increases [4]. One way to mitigate the scalability issue is to develop powerful numerical solvers to handle large-scale optimization problems [12], [13], [14]. For example, Borrelli *et al.* used IPOPT to solve optimization problems in trajectory planning tasks of multiple UAVs [13]. A scalable first-order solver has recently been developed for sparse conic problems [14]. These solvers could handle the networks of small size efficiently, but may still scale poorly to large-scale and dense instances. Some researches take advantage of special structures of multi-agent formations to improve computation performance. For instance, Alonso-Mora *et al.* studied the formation of multiple vehicles and object transport by dividing the task into obstacle-free region computation and formation parameters optimization [15]. Urcola *et al.* proposed a hybrid centralized-distributed architecture for the path planning for robot formations, where the leader centralizes the information and executes the global process and the followers execute the navigation in a distributed way [16]. Zheng *et al.* introduced an efficient sequential algorithm for scalable design of structured controllers by exploiting underlying sparsity properties [17]. While efficient computation could be obtained through these approaches, the

This study is supported by NSF China with 51575293 and 51622504, National Key R&D Program of China with 2016YFB0100906, and International Sci&Tech Cooperation Program of China under 2016YFE0102200. All correspondence should be sent to S. Li.

Zhitao Wang, Shengbo Eben Li and Keqiang Li are with the Department of Automotive Engineering, Tsinghua University, Beijing, China. (`wangzt16@mails.tsinghua.edu.cn`, `{lishbo, likq}@tsinghua.edu.cn`).

Yang Zheng was with the Department of Automotive Engineering, Tsinghua University, Beijing, China. He is now with the Department of Engineering Science, University of Oxford, UK. (`zhengy093@gmail.com`).

Keyou You is with the Department of Automation, Tsinghua University, Beijing, China. (`youky@tsinghua.edu.cn`).

special nature of the tasks prevents the application of the algorithms to general connected vehicle networks.

In this paper, we formulate the cooperative automation of multiple connected vehicles as a centralized optimal control problem. The receding horizon framework is applied to ensure its robustness in dynamic environment and disturbances. To extend the problem solution to large-scale cases, we use the alternating direction method of multipliers (ADMM) [18] to decompose the centralized optimization problem. One notable feature is that parallel computation can be realized after the decomposition using ADMM. Taylor series is utilized to linearize the nonconvex constraints in the problem to facilitate the application of ADMM. Note that ADMM is a simple yet powerful algorithm for distributed optimization, which has been applied in a wide range of fields [18], [19]. The contributions of this paper are 1) we design an optimal control framework for coordinating connected vehicles, in which the cooperative automation task is formulated as a centralized optimization problem in a receding horizon fashion; 2) we propose a parallel optimization algorithm to solve the centralized optimization problem via ADMM, and the algorithm can be implemented on a cloud platform to deal with large-scale vehicle networks.

The rest of this paper is organized as follows. Section II formulates the centralized optimal control problem for the cooperative automation of connected vehicles; Section III introduces the parallel computation algorithm via ADMM; Section IV demonstrates the effectiveness of the proposed algorithm, and we conclude this paper in Section V.

II. Cooperative Automation of Connected Vehicles

Connected vehicles equipped with advanced sensing and communication devices can sense the surroundings and upload information to a cloud platform. Then, the cloud platform computes the decision action and transmits instructions to each vehicle for coordination.

A. *Modelling of Computation Networks*

An undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is used to describe the constraint topology among interactive vehicles, where $\mathcal{V} = \{1, ..., N\}$ denotes a set of connected vehicles in the system and N is the number of vehicles. Since only the vehicles in the proximity of another one have the collision possibility, it is not necessary to consider the constraints with all the other vehicles. $\mathcal{E} = \{1, ..., M\}$ is the set of edges which represent the coupling constraint between two interactive vehicles, defined as

$$\begin{cases} (i,j) \in \mathcal{E}(t) & \|p_i - p_j\| \leqslant d_{\text{perc}} \\ (i,j) \notin \mathcal{E}(t) & \text{otherwise}, \end{cases} \tag{1}$$

where M is the number of coupling constraints in the vehicle network, $p_i = [p_{i,x}, p_{i,y}]^T$ is position of vehicle i in a global coordinate system, d_{perc} is the distance which ensures collision avoidance between two vehicles under the constraint of vehicle kinematics. The perception distance of each vehicle should be not less than d_{perc}, which guarantees

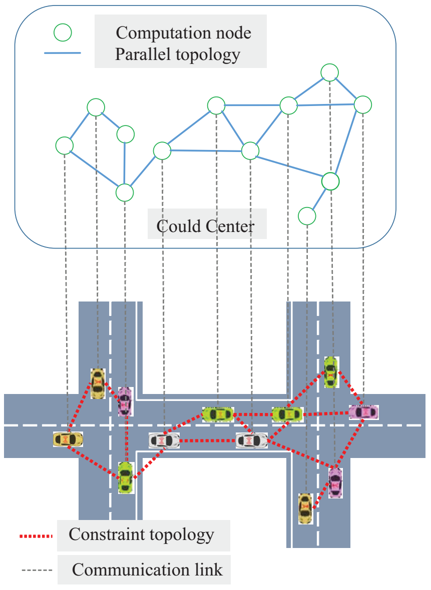

Fig. 1. An illustration of vehicle network and computation network of the cloud platform.

that the vehicles linked by (i, j) could take action in time as soon as sensing the information of the coupling vehicles. We define the neighbor nodes of each vehicle as

$$\mathcal{N}_i = \{j | (i,j) \in \mathcal{E}, \forall j \in \mathcal{V}\}. \tag{2}$$

In addition to the constraint topology among vehicles, the computation network of the cloud platform can be described by $\mathcal{G}$ as well. The cloud platform is composed of computer clusters which allows high-performance parallel computation. The computation nodes are the mapping of vehicle nodes in the constraint topology. The relation between the constraint topology and the computation network is shown in Fig. 1.

***Remark* 1:** Since the vehicles move over time, $\mathcal{G}$ is time-varying and the links between computation nodes $\mathcal{V}$ change as well. To ensure efficient computation, the link and the information transmission between computation nodes should be managed by an effective coordination mechanism. The computation resource scheduling and management are beyond the scope of this paper.

B. *A Centralized Large-scale Optimal Control Problem*

Each connected vehicle in $\mathcal{V}$ has its own origin and target position, which needs to finish its own guidance task in a coordinating fashion with other vehicles. The path of each vehicle can be planned by any efficient planning algorithms, *e.g.*, RRT and A* algorithm [20]. Then, the cooperative automation task can be formulated as an optimization problem, in which each vehicle is scheduled to plan its path and follow the planned path under the constraints of physical environments, *i.e.*, collision avoidance constraints and other physical limitations.

In this paper, we formulate the automation problem in a centralized fashion based on the receding horizon optimization framework as

$$\begin{aligned} \min_{u_i} \quad & J(u_i) = \sum_{i \in \mathcal{V}} \Big[\int_t^{t+T} h_i(x_i, u_i) d\tau \Big] \\ \text{subject to} \quad & \dot{x}_i = f_i(x_i, u_i) \\ & (x_i, u_i) \in \mathbb{C}_{\mathrm{s}} \\ & (x_i, u_i, x_j, u_j) \in \mathbb{C}_{\mathrm{c}}, j \in \mathcal{N}_i, \end{aligned} \tag{3}$$

where $x_i = [p_i, \theta_i]$ and u_i denote the state and control variable of vehicle i, respectively; $\dot{x}_i = f(x_i, u_i)$ denotes the kinematics of the vehicle; T is the predicted time horizon; $\mathbb{C}_{\mathrm{s}}$ denotes the set of constraints of the vehicle i; and $\mathbb{C}_{\mathrm{c}}$ is the set of coupling constraints of a pairwise of connected vehicles $(i, j), j \in \mathcal{N}_i$, the function $h_i(x_i, u_i) = Q\|x_i - x_{\mathrm{ref},i}\|_2^2$ describes the deviation of the predicted planning trajectory from the reference trajectory $x_{\mathrm{ref},i}$.

Specifically, $\mathbb{C}_{\mathrm{s}}$ represents the constraints of the physical environment for a single vehicle, *e.g.*, the bound of the steering angle and the vehicle position. These can be described as

$$\mathbb{C}_{\mathrm{s}} = \{(x, u) | x_{\mathrm{low}} \leqslant x \leqslant x_{\mathrm{up}}, u_{\mathrm{low}} \leqslant u \leqslant u_{\mathrm{up}}\}, \tag{4}$$

where $\mathbb{C}_{\mathrm{c}}$ represents the collision avoidance constraints between pairwise interactive vehicles. Based on the constraint topology, this is imposed as

$$\mathbb{C}_{\mathrm{c}} = \{(x_i, u_i, x_j, u_j) | \|p_i - p_j\| \geqslant d_{\mathrm{safe}}, j \in \mathcal{N}_i\}, \tag{5}$$

where d_{safe} is a predefined safe separation distance between vehicles.

C. The Receding Horizon Optimization

To cope with the dynamic environment and disturbance, receding horizon optimization is carried out to solve the optimal control problem. It solves the optimization problem (3) to generate control inputs for a finite horizon starting from the latest states, where only the first step of the control inputs is implemented.

It is assumed that the network $\mathcal{G}$ is time-invariant in the predicted time horizon $\tau \in [t, t+T]$ for convenience of computations, *i.e.*, $\mathcal{G}(\tau) = \mathcal{G}(t)$. This assumption makes the coupling constraints time-invariant. The selection of predicted time horizon T should consider the kinematic of the bicycle model and ensure the pairwise vehicles keep safe distance once they sense each other. Every vehicle in the centralized formulation is coupled due to the pairwise collision avoidance constraints. A decentralized solution is proposed to solve this centralized formulation in the following section.

***Remark* 2:** The centralized problem formulation makes it easy to ensure the stability of model predictive control compared with Kuwata's work [5]. Also, this formulation does not need to design any coordination mechanism compared to the priority approach and related distributed control [8].

III. Parallel Optimization Algorithm

The centralized optimal control (3) can be reformulated as a quadratic programming (QP) problem in a discrete time domain. While interior-point methods are efficient to solve (3) when the problem is of small size, the computational demand increases as the number of vehicles grows, which scales poorly to large-scale instances. Decentralized optimization algorithms based on first-order algorithms that scale well to large-scale instances have attracted considerable research attention in recent years [18]. ADMM is one efficient first-order algorithm to distribute the computation of a large-scale problem to a network of computing nodes, which has been applied in many fields, *e.g.*, statistical learning [18], distributed control [21], power system management [22] and sparse semidefinite programs [23]. Motivated by the wide applications, we employ ADMM to solve our large-scale connected automation problem in a parallel way.

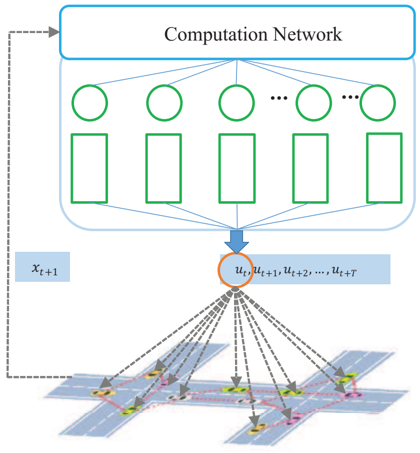

Fig. 2. The receding horizon optimization framework.

In this section, we first transform the centralized optimization problem into a consensus optimization problem. Then, ADMM is applied to decompose the consensus problem, and parallel computation can be realized based on the decomposition through the cloud computation network.

A. The ADMM Algorithm

The ADMM algorithm aims to solve an optimization problem in the following form [18]

$$\begin{aligned} \min \quad & F(a) + G(b) \\ \text{subject to} \quad & Aa + Bb = d, \end{aligned}$$

where F and G are convex function, $a \in \mathbb{R}^{n_a}, b \in \mathbb{R}^{n_b}, d \in \mathbb{R}^{n_d}, A \in \mathbb{R}^{n_d \times n_a}$ and $B \in \mathbb{R}^{n_d \times n_b}$. Given a penalty parameter $\rho > 0$ and a scaled dual multiplier $z \in \mathbb{R}^{n_d}$, the (scaled) ADMM algorithm minimizes the augmented Lagrangian

$$L_\rho(a, b, z) = F(a) + G(b) + \frac{\rho}{2}\|Aa + Bb - d + z\|^2,$$

with respect to the variables a and b separately, followed by a dual variable update:

$$\begin{aligned} a^{k+1} &:= \underset{a}{\operatorname{argmin}} \, L_\rho(a, b^k, z^k), \\ b^{k+1} &:= \underset{b}{\operatorname{argmin}} \, L_\rho(a^{k+1}, b, z^k), \\ z^{k+1} &:= z^k + (Aa^{k+1} + Bb^{k+1} - d). \end{aligned} \tag{6}$$

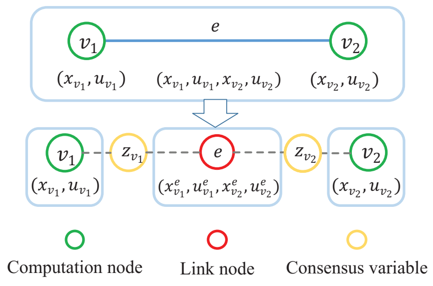

Fig. 3. Formulation of the consensus optimization: consensus constraints are applied to decompose the coupling.

The superscript k indicates that a variable is fixed to its value at the k-th iteration. ADMM is particularly suitable when the minimizations with respect to each of the variables a and b in (6) can be carried out efficiently. We refer the interested readers to [18] for more details.

B. Formulation of the Consensus Optimization

To deal with $\mathbb{C}_s$ and $\mathbb{C}_c$ in (3), two indicator functions $\mathbb{I}_s(\cdot)$ and $\mathbb{I}_c(\cdot)$ are defined to describe the constraints of the vehicle i and pairwise interactive vehicles (i, j) respectively:

$$\begin{aligned}&\mathbb{I}_c(x_i, u_i, x_j, u_j) := \\ &\begin{cases}0 & (x_i, u_i, x_j, u_j) \in \mathbb{C}_c, j \in \mathcal{N}_i \text{ and } \dot{x}_i = f_i(x_i, u_i) \\ \infty & \text{otherwise},\end{cases} \\ &\mathbb{I}_s(x_i, u_i) := \\ &\begin{cases}0 & (x_i, u_i) \in \mathbb{C}_s \text{ and } \dot{x}_i = f_i(x_i, u_i) \\ \infty & \text{otherwise}.\end{cases}\end{aligned} \tag{7}$$

Then, the centralized problem (3) can be equivalently rewritten as

$$\begin{aligned}\min_{u_i} \hat{J}(u_i) = \sum_{i\in\mathcal{V}} \Big[\int_t^{t+T} h_i(x_i, u_i) d\tau + \mathbb{I}_s(x_i, u_i)\Big] + \\ \sum_{(i,j)\in\mathcal{E}} \mathbb{I}_c(x_i, u_i, x_j, u_j).\end{aligned} \tag{8}$$

The optimization problem (8) is coupled among $\mathcal{V}$ due to the coupling constraint $\mathbb{I}_c(\cdot)$. By introducing a set of consensus constraints, problem (8) could be equivalently rewritten into the standard ADMM form

$$\begin{aligned}\min_{u_v, u_v^e} \sum_{v\in\mathcal{V}} \Big[\int_t^{t+T} h_v(x_v, u_v) d\tau + \mathbb{I}_s(x_v, u_v)\Big] + \\ \sum_{e\in\mathcal{E}} \mathbb{I}_c(\{x_v^e, u_v^e\}_{v\in\mathcal{V}(e)})\end{aligned} \tag{9}$$

$$\text{Subject to:} \quad u_v = z_v,\ u_v^e = z_v, \quad v \in \mathcal{V}, e \in \mathcal{E}(v),$$

where $\mathcal{V}(e)$ represents the set of computation nodes linked with edge $e \in \mathcal{E}$, which called local nodes, $\mathcal{E}(v)$ represents the set of link nodes linked with node $v \in \mathcal{V}$, and $\{x_v^e, u_v^e\}_{v\in\mathcal{V}(e)}$ denotes a vector indexed by a parameter $v \in \mathcal{V}(e)$, z_v is the consensus variable.

For the purpose of parallel computation, we assign link node to each pairwise interactive local node and arrange the single vehicle objective function and coupled objective function in (9) to local and link node respectively with consensus constraint exist during local and link nodes. Next, the parallel algorithm will be realized based on ADMM. Fig. 3 illustrates the formulation of the consensus optimization.

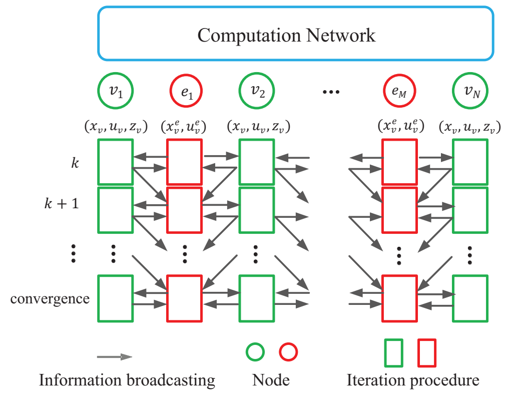

Fig. 4. An illustration of the parallel algorithm: the local and link nodes run their own task and change information with neighbors.

C. A Parallel Algorithm based on ADMM

Here, we apply the ADMM to decompose the centralized consensus optimization problem (9), leading to a parallel algorithm to solve the problem. First, the augmented Lagrangian function for the problem (9) is

$$\begin{aligned}L_\rho(u_v, u_v^e, z_v, \lambda_v^e) = \sum_{v\in\mathcal{V}} \Big[\int_t^{t+T} h_v(x_v, u_v) d\tau + \mathbb{I}_s(x_v, u_v)\Big] \\ + \sum_{e\in\mathcal{E}} \mathbb{I}_c(\{x_v^e, u_v^e\}_{v\in\mathcal{V}(e)}) + \sum_{v\in\mathcal{V}(e)} \frac{\rho}{2}\|u_v - z_v + \lambda_v\|_2^2 \\ + \sum_{e\in\mathcal{E}} \sum_{v\in\mathcal{V}(e)} \frac{\rho}{2}\|u_v^e - z_v + \lambda_v^e\|_2^2,\end{aligned} \tag{10}$$

where λ_v and λ_v^e are scaled dual variables corresponding to u_v and $\{u_v^e\}_{v\in\mathcal{V}(e)}$ respectively, and ρ is the augmented Lagrangian parameter.

According to (6), the ADMM algorithm minimizes the augmented Lagrangian (10) iteratively. At each iteration k, the algorithm consists of the following three steps.

Step 1: Each local node $v \in \mathcal{V}$ solves the following local problem in parallel to update u_v:

$$\begin{aligned}x_v^{k+1}, u_v^{k+1} := \operatorname*{argmin}_{u_v} \Big\{\int_t^{t+T} h(x_v, u_v) d\tau + \mathbb{I}_s(x_v, u_v) + \\ \sum_{e\in\mathcal{E}(v)} \Big[\frac{\rho}{2}\|u_v - z_v^k + \lambda_v^k\|_2^2\Big]\Big\}.\end{aligned} \tag{11}$$

Each link node solves the following coupled optimization problem locally in a parallel way to update u_v^e:

$$\begin{aligned}\{x_v^e, u_v^e\}_{v\in\mathcal{V}(e)}^{k+1} := \operatorname*{argmin}_{u_v^e} \Big\{\mathbb{I}_c(\{x_v^e, u_v^e\}_{v\in\mathcal{V}(e)}) + \\ \sum_{v\in\mathcal{V}(e)} \Big[\frac{\rho}{2}\|u_v^e - z_v^k + \lambda_v^{e\,k}\|_2^2\Big]\Big\}.\end{aligned} \tag{12}$$

Step 2: Each local node updates the consensus variable z_v in the following form:

$$z_v^{k+1} := \frac{\left(u_v^{k+1} + \frac{\lambda_v^k}{\rho}\right) + \sum_{e\in\mathcal{E}(v)} \left(u_v^{e\,k+1} + \frac{\lambda_v^{e\,k}}{\rho}\right)}{1 + \sum_{e\in\mathcal{E}(v)} 1}. \tag{13}$$

Step 3: The scaled dual variable λ is updated as (14) and (15), which could be computed on each local node and link node respectively:

$$\lambda_v^{k+1} := \lambda_v^k + (u_v^{k+1} - z_v^{k+1}), \tag{14}$$
$$\lambda_v^{e\,k+1} := \lambda_v^{e\,k} + (u_v^{e\,k+1} - z_v^{k+1}). \tag{15}$$

The steps (11)-(15) iterate on the computation nodes in parallel until convergence. In each step, the computation procedure is entirely decentralized to each node in the cloud network. By broadcasting information with neighbor computation nodes, the resulting algorithm can be carried out in a parallel way. Fig. 4 shows the process and information broadcasting during each iteration of the ADMM algorithm.

***Remark* 3:** Compared with [21], [24], our algorithm is more suitable for parallel computation since the single and coupled objective functions are solved in parallel by introducing consensus variables rather than in sequence.

D. Stopping Conditions

According to [18], the stopping criterion can be described with the primal and dual residuals as follows:

$$\begin{aligned} r^{k+1} &= \|u^{k+1} - z^{k+1}\|_2 \leqslant \epsilon^{\text{pri}}, \\ s^{k+1} &= \rho\|z^{k+1} - z^k\|_2 \leqslant \epsilon^{\text{dual}}, \end{aligned} \tag{16}$$

where ϵ^{pri} and ϵ^{dual} are primal and dual feasibility tolerances respectively:

$$\begin{aligned} \epsilon^{\text{pri}} &= \epsilon^{\text{abs}}\sqrt{(N+2M)N_p} + \epsilon^{\text{rel}}\max\{\|u^k\|_2, \|z^k\|_2\}, \\ \epsilon^{\text{dual}} &= \epsilon^{\text{abs}}\sqrt{(N+2M)N_p} + \epsilon^{\text{rel}}\frac{\|\lambda^k\|_2}{\rho}, \end{aligned} \tag{17}$$

where ϵ^{abs} and ϵ^{rel} are the absolute and relative criterion, M is the size of link computation node group and N_p is the number of predictive step.

While the ADMM algorithm converges independently of the choice of penalty parameter ρ, in practice this value strongly influences the number of iterations required for convergence. Unfortunately, analytic results for the optimal choice of ρ are not available except for some special problems [14]. As suggested in [18], in order to improve the practical convergence performance and make performance less dependent on the choice of ρ, our algorithm employs the following dynamic adaptive rule:

$$\rho^{k+1} = \begin{cases} \tau^{\text{incr}}\rho^k & \|r^k\|_2 > \mu\|s^k\|_2, \\ \dfrac{\rho^k}{\tau^{\text{decr}}} & \|s^k\|_2 > \mu\|r^k\|_2, \\ \rho^k & \text{otherwise}, \end{cases} \tag{18}$$

where τ^{incr}, τ^{decr} and μ are set to 2, 2 and 5 in our algorithm.

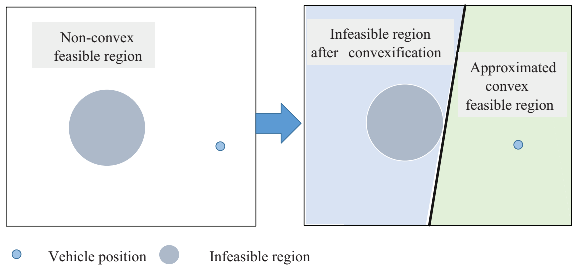
Non-convex feasible region
Infeasible region after convexification
Approximated convex feasible region
Vehicle position
Infeasible region

Fig. 5. Approximation of the nonconvex constraint: by linearization, a convex while sub-optimal decision space is got.

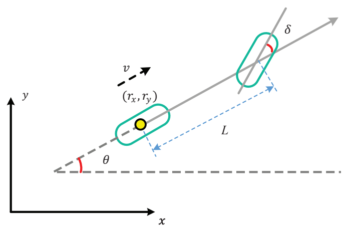

Fig. 6. The bicycle model.

E. Convexification of the Problem

The convergence of ADMM is typically guaranteed for convex problems [18]. Here, we introduce a procedure to linearize the non-convex constraints due to the collision avoidance (5). Based on the properties of receding horizon optimization, we generate a seed trajectory r^0 by moving the optimal control input calculated in previous cycle one step forward and repeating the last prediction step [24].

To linearize the non-convex constraints, the Taylor's series is applied to the constraints (5) at p^0:

$$2(p_i^0 - p_j^0)^T(p_i - p_j) - \|p_i^0 - p_j^0\|_2^2 \geqslant d_{\text{safe}}^2, j \in \mathcal{N}_i. \tag{19}$$

By linearization, we approximate the original nonconvex constraint with a convex linear constraint, which returns a sub-optimal planning path in the prediction time horizon for each vehicle in the network. Fig. 5 illustrates the approximation of the nonconvex constraint.

IV. Numerical Simulations and Discussion

In our simulation, the kinematics of each vehicle ignoring tire slip angle are described as follows:

$$\begin{aligned} \dot{r}_x &= v\cos\theta, \\ \dot{r}_y &= v\sin\theta, \\ \dot{\theta} &= \frac{v}{L}\tan\delta, \end{aligned} \tag{20}$$

where (r_x, r_y) denotes the rear wheel axle center coordinates of the vehicle, θ denotes the vehicle heading angle with respect to the global x-axis (positive counter-clockwise), v denotes the speed of rear wheel axle center, δ denotes the steer angle (positive counter-clockwise), and L is the vehicle wheelbase. The position (p_x, p_y) of the vehicle could be derived from these variables. See Fig. 6 for an illustration.

Note that the kinematic bicycle model, described as (20), is non-linear and coupled between lateral and longitudinal kinematics. To simplify the formulation and facilitate solution to

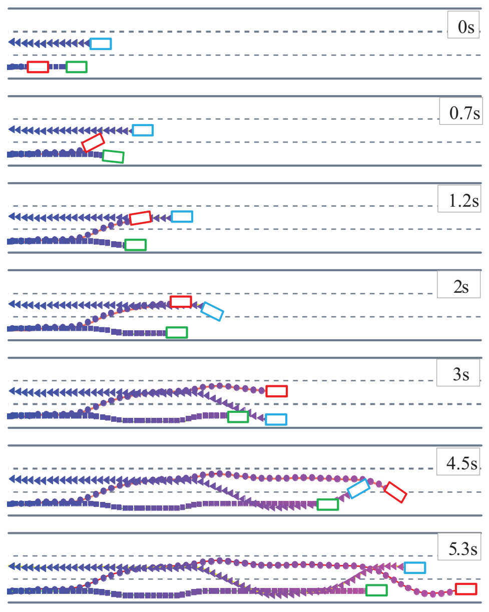

Fig. 7. Snapshots of cooperative overtaking of three vehicles.

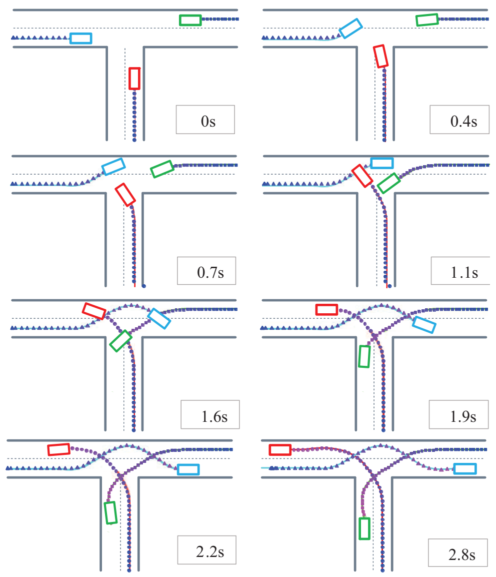

Fig. 8. Snapshots of coordination in the intersection of three vehicles.

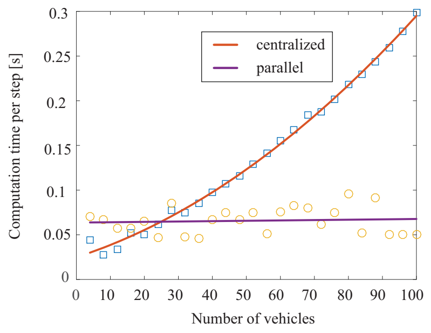

Fig. 9. Computation performance of the centralized and parallel algorithms: the proposed algorithm outperform the centralized one for large scale instances.

the control problem, a common method is to assume that the vehicle speed is preset and constant in a short predictive horizon [24]. Then, we apply Taylor series to the kinematic model to transform the vehicle kinematics to a linear state space model [21].

Two typical traffic scenes with multiple connected vehicles are simulated to test the effectiveness of the proposed algorithm. The vehicle speeds are from 40km/h to 50km/h, and the wheelbase of vehicle is set as L=2.4m. Considering the tradeoff between the computation demanding and tracking performance, we set the sampling time T_s=0.1s and the prediction horizon N_p=15 in the simulation. The parameters of ADMM algorithm are set as following ϵ^{abs}=0.01, and ϵ^{rel}=0.01. The computation is implemented on a system with Intel i7 processor at 3.4 GHz. The ADMM algorithm is implemented in MATLAB environment with the quadratic programs solved by the interior-point solver Gurobi [25].

A. Numerical Simulation

The first traffic task is the cooperative overtaking of multiple vehicles in three lanes, a normal scenario in highway driving. In our simulation, three vehicles have different reference paths and desired velocities. The trajectories generated form the algorithm are shown in Fig. 7. When overtaking, the overtaken vehicles (the green and blue vehicles) will give way to the overtaking vehicle (the red vehicle), which could improve the safety and efficiency of the traffic flow. After the overtaking task, every vehicle returns to its original reference path.

Our second scenario is vehicle motion control at intersection without traffic lights. As shown in Fig. 8, the vehicles will deviate from the reference trajectory to keep a safe distance from surrounding vehicles when the conflicting may happen, *i.e.*, the blue vehicle will change lane to avoid collision with the red vehicle and the green vehicle turns left in advance to keep a safe distance with the blue vehicle. This scenario tests the effectiveness of the proposed algorithm in a more complicated traffic condition, and the results are promising and meaningful for improving traffic efficiency.

The authors in [26] adopted the priority approach to deal with the scenario, where a similar effect was obtained. As mentioned in the previous section, one major advantage of our algorithm is the ability of parallel computation, which is able to scale to large-scale instances. We demonstrate this fact in the next subsection.

B. Discussion of the Computation Performance

In the discretized setting, the centralized optimal control problem amounts to solve a quadratic programming (QP) problem, which can be solved in polynomial time. In our problem, the dimension of the QP program is $N \times N_p$, where N is the number of vehicles. The required number of iterations is $k = \mathcal{O}(\frac{1}{\epsilon})$ for ADMM, where ϵ is the prescribed precision. In principal, the computation complexity of the centralized and parallel solutions are $\mathcal{O}\big((N \times N_p)^m\big)$ and $\mathcal{O}\big(k(N_p)^m\big)$ respectively, which indicates the parallel algorithm is scalable to large-scale problems.

Here, we conducted numerical experiments to verify the performance of our algorithm with different vehicle size, ranging from 4 to 100. Fig. 9 shows the computational time in one sampling step. The time used for communication is negligible and the accumulation of the maximum time consumption for each computation node in every iteration is used as the computation for the parallel algorithm. As the number of vehicles increases, the optimization time on each computation node is almost invariability, while the computation time for the centralized optimization increases. This is in accordance to our expectation, since the computation in the ADMM algorithm can be carried out in a parallel fashion, where the dimension of each subproblem is independent to the network size. When the vehicle number is small, the centralized optimization algorithm seems faster in our simulations. However, the computation time of the centralized method grows quickly as the number of vehicles increases, thus the centralized algorithm is not suitable for large-scale instances. In contrast, the parallel algorithm is scalable to large-scale cases.

V. Conclusions

This paper has proposed a parallel optimization algorithm using ADMM, which is promising for cooperative automation of large-scale connected vehicles. In particular, we formulated the cooperative automation task as a centralized optimization problem in a receding horizon fashion. Taylor expansion was applied to the centralized problem?The resulting algorithm is suitable for parallel implementations. Numerical experiments based on typical traffic scenarios demonstrated the effectiveness and efficiency of the proposed algorithm. One future work is to address the convergence of the parallel algorithm under time delay of information broadcasting. Also, validations on real world experiments would be interesting to test the performance of our algorithm.

Acknowledgement

The authors would like to thank Prof. Guangyu Tian from Tsinghua University for his valuable comments on this work.

References

[1] N. Lu, N. Cheng, N. Zhang, X. Shen, and J. W. Mark, “Connected vehicles: Solutions and challenges,” *IEEE Internet Things J.*, vol. 1, no. 4, pp. 289–299, 2014.

[2] J. Lee and B. Park, “Development and evaluation of a cooperative vehicle intersection control algorithm under the connected vehicles environment,” *IEEE Trans. Intell. Transp. Syst.*, vol. 13, pp. 81–90, March 2012.

[3] 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 Intell. Transp. Syst. Mag.*, vol. 9, no. 3, pp. 46–58, 2017.

[4] B. Li, Y. Zhang, Z. Shao, and N. Jia, “Simultaneous versus joint computing: A case study of multi-vehicle parking motion planning,” *J. Comput. Sci.*, vol. 20, pp. 30–40, 2017.

[5] Y. Kuwata and J. P. How, “Cooperative distributed robust trajectory optimization using receding horizon milp,” *IEEE Trans. Control Syst. Technol*, vol. 19, no. 2, pp. 423–431, 2011.

[6] Y. Zheng, S. E. Li, K. Li, F. Borrelli, and J. K. Hedrick, “Distributed model predictive control for heterogeneous vehicle platoons under unidirectional topologies,” *IEEE Trans. Control Syst. Technol*, vol. 25, no. 3, pp. 899–910, 2017.

[7] Y. Wu, S. E. Li, Y. Zheng, and J. K. Hedrick, “Distributed sliding mode control for multi-vehicle systems with positive definite topologies,” in *IEEE Conf. on Decision and Control*, pp. 5213–5219, 2016.

[8] P. Velagapudi, K. Sycara, and P. Scerri, “Decentralized prioritized planning in large multirobot teams,” in *IEEE Conf. Intell. Robot. and Syst.*, pp. 4603–4609, 2010.

[9] M. G. Plessen, D. Bernardini, H. Esen, and A. Bemporad, “Multi-automated vehicle coordination using decoupled prioritized path planning for multi-lane one-and bi-directional traffic flow control,” in *IEEE Conf. on Decision and Control*, pp. 1582–1588, 2016.

[10] J. Yu and S. M. LaValle, “Optimal multirobot path planning on graphs: Complete algorithms and effective heuristics,” *IEEE Trans. Robot.*, vol. 32, no. 5, pp. 1163–1177, 2016.

[11] Y. Zheng, S. E. Li, J. Wang, D. Cao, and K. Li, “Stability and scalability of homogeneous vehicular platoon: Study on the influence of information flow topologies,” *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 1, pp. 14–26, 2016.

[12] A. Richards and J. P. How, “Aircraft trajectory planning with collision avoidance using mixed integer linear programming,” in *Proc. Amer. Control Conf.*, vol. 3, pp. 1936–1941, 2002.

[13] F. Borrelli, D. Subramanian, A. U. Raghunathan, and L. T. Biegler, “A comparison between milp and nlp techniques for centralized trajectory planning of multiple unmanned air vehicles,” in *Proc. Amer. Control Conf.*, 2006.

[14] Y. Zheng, G. Fantuzzi, A. Papachristodoulou, P. Goulart, and A. Wynn, “Chordal decomposition in operator-splitting methods for sparse semidefinite programs,” *arXiv preprint arXiv:1707.05058*, 2017.

[15] J. Alonso-Mora, S. Baker, and D. Rus, “Multi-robot formation control and object transport in dynamic environments via constrained optimization,” *Int. J. Robot. Res.*, vol. 36, no. 9, pp. 1000–1021, 2017.

[16] P. Urcola, M. T. Lázaro, J. A. Castellanos, and L. Montano, “Cooperative minimum expected length planning for robot formations in stochastic maps,” *Rob. and Auton. Syst.*, vol. 87, pp. 38–50, 2017.

[17] Y. Zheng, R. P. Mason, and A. Papachristodoulou, “Scalable design of structured controllers using chordal decomposition,” *IEEE Trans. Autom. Control*, vol. 63, no. 3, pp. 752–767, 2018.

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

[19] J. Bento, N. Derbinsky, J. Alonso-Mora, and J. S. Yedidia, “A message-passing algorithm for multi-agent trajectory planning,” in *Adv. Neural Inf. Process Syst.*, pp. 521–529, 2013.

[20] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, “A survey of motion planning and control techniques for self-driving urban vehicles,” *IEEE Trans. Intell. Vehi.*, vol. 1, no. 1, pp. 33–55, 2016.

[21] H. Y. Ong and J. C. Gerdes, “Cooperative collision avoidance via proximal message passing,” in *Proc. Amer. Control Conf.*, pp. 4124–4130, 2015.

[22] T. Erseghe, “Distributed optimal power flow using admm,” *IEEE Trans. Power Syst.*, vol. 29, no. 5, pp. 2370–2380, 2014.

[23] Y. Zheng, G. Fantuzzi, A. Papachristodoulou, P. Goulart, and A. Wynn, “Fast ADMM for semidefinite programs with chordal sparsity,” in *Proc. Amer. Control Conf.*, pp. 3335–3340, 2017.

[24] H. Zheng, R. R. Negenborn, and G. Lodewijks, “Fast ADMM for distributed model predictive control of cooperative waterborne agvs,” *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 4, pp. 1406–1413, 2017.

[25] G. Optimization, “Inc.,“gurobi optimizer reference manual,” 2017,” *URL: http://www. gurobi. com*, 2017.

[26] M. Gerdts and B. Martens, “Optimization-based motion planning in virtual driving scenarios with application to communicating autonomous vehicles,” *arXiv preprint arXiv:1801.07612*, 2018.