An Interior Point Algorithm for Optimal Coordination of Automated Vehicles at Intersections

Robert Hult, Mario Zanon, Sébastien Gros, Paolo Falcone

Abstract—In this paper, we consider the optimal coordination of automated vehicles at intersections under fixed crossing-orders. We state the problem as a Direct Optimal Control problem, and propose a line-search Primal-Dual Interior Point algorithm with which it can be solved. We show that the problem structure is such that most computations required to construct the search-direction and step-size can be performed in parallel on-board the vehicles. This is realized through the Schur complement of blocks in the KKT-matrix in two steps and a merit-function with separable components. We analyze the communication requirements of the algorithm, and propose a conservative approximation scheme which can reduce the data exchange. We demonstrate that in hard but realistic scenarios, reductions of almost 99% are achieved, at the expense of less than 1% sub-optimality.

I. INTRODUCTION

The last decade has seen rapid development of technologies for Automated Vehicles (AV). Simultaneously, several standards have been adopted for vehicle-to-vehicle communication, and use of next generation cellular communication in automotive applications is under investigation. Due to this, the interest in applications where the AVs share information and cooperate is increasing, and it is commonly held that such Cooperative Automated Vehicles (CAV) would have positive effects on the traffic system.

One such application is the coordination of CAVs at intersections. The idea is to let the CAVs jointly decide how to cross the intersection safely and efficiently, rather than relying on traffic-lights, road signs and traffic rules. In this paper, we study a numerical algorithm for the optimal control formulation of such scenarios.

The literature on algorithms for coordination of CAVs at intersections was surveyed in [1]–[3], and even though most work is recent, the number of publications is growing rapidly. While a substantial part of the literature rely completely on heuristics [4]–[6], a number of contributions that employ Optimal Control (OC) tools [7]–[14] have been proposed recently. However, most OC-based algorithms rely on heuristics to some extent. This is largely due to the difficult combinatorial nature of the problem, which stems from the need to determine the order in which the vehicles cross the intersection. In a number of contributions the problem is solved in two stages where 1) the crossing order is found through a heuristic (typically variations of “first-come-first-served”) and 2) the control commands are found using OC-tools [11]–[13]. The algorithm in this paper is intended for such applications, and deals with the problem of finding the optimal control commands for a fixed crossing order.

We have studied the intersection problem in earlier work. In [15], we introduced a Sequential Quadratic Programming (SQP) algorithm based on a primal decomposition of the fixed-order coordination problem, where most computations are parallelized and performed on-board the vehicles. We considered the receding horizon application of the SQP algorithm in [16], and presented experimental results. We considered the extension to non-linear motion models and economic objective functions in [17] and to scenarios with turning vehicles in [18]. In [19], we proposed an OC-based heuristic for crossing order selection, and compared the performance of to traffic-lights and other algorithms in [20].

However, the algorithm in [15] did not account for rear-end collisions between vehicles on the same lane, and required the solution of a non-smooth Nonlinear Program (NLP). In this paper, we solve the fixed-order coordination problem with a Primal-Dual Interior Point (PDIP) algorithm which resolves both these issues. The algorithm relies on distributed computation of both the search-direction and step-size. As in [15], [16], this approach is partly centralized, and relies on central units for some computations. In particular, the algorithm uses one intersection-wide central unit and one central unit for each lane, with communication flows as illustrated in Fig. 1.

A. Contributions

The contributions in this paper are 1) The application of a distributed PDIP algorithm to the intersection problem, 2) the analysis of the communication requirements in a practical setting 3) a method with which the communication requirements can be reduced at the cost of sub-optimality. While similar distribution schemes can be found in the literature (see e.g. [21], [22]), the application to the intersection problem is novel.
Fig. 2: Illustration of the scenarios considered, Assumption 2 (black lines) and the Conflict Zones (red boxes).

B. Outline

The remainder of the Paper is organized as follows. In Section II we model and state the intersection problem using an Optimal Control formalism. In Section III we review Primal-Dual Interior Point methods and outline how the computations involved can be parallelized. In Section IV we show how the KKT system can be solved with computation at the vehicle, lane and intersection-centers. In Section V we show how to select the step-size in a distributed fashion. In Section VI we state a rudimentary practical algorithm and provide a numerical example. In Section VII we analyze the communication requirements and propose an approximate representation of the RECA constraints, using which the data sent per iterate can be reduced. The paper is concluded in Section VIII.

II. OPTIMAL COORDINATION AT INTERSECTIONS

We consider intersection scenarios as shown in Fig. 2, where N vehicles approach an intersection with L lanes, and make the following assumptions:

Assumption 1 (Full automation and cooperation). There are no non-cooperative entities present in the scenario.

Assumption 2 (Vehicles on rails). The vehicles do not change lanes and move along fixed and known paths along the road. All vehicles on the same lane use the same path.

Assumption 1 means that we do not consider scenarios with, e.g. legacy vehicles, pedestrians or bicyclists. The assumption is restrictive and limits the applicability to traffic scenarios e.g. legacy vehicles, pedestrians or bicyclists. The assumption is restrictive and limits the applicability to traffic scenarios in a distant future. Assumption 2, however, is not restrictive, since vehicles at intersections in general follow the centerline of the lane that they are on. Both assumptions are standard in the literature (see e.g. [4]–[6], [10]).

A. Motion Models

Assumption 2 enables simple motion models that describe the one-dimensional movement of vehicles along their paths. We consider constrained ODE motion-models such that

\[ \dot{x}_i(t) = f_i(x_i(t), u_i(t)), \quad 0 \geq c_i(x_i(t), u_i(t)), \]

where \( i \) is the vehicle index, \( x_i(t) \in \mathbb{R}^{n_i} \) and \( u_i(t) \in \mathbb{R}^{m_i} \) are the vehicle state and control. In particular, \( x_i(t) = (p_i(t), v_i(t), \tilde{x}_i(t)) \), where \( p_i(t) \) is the position of the vehicle's geometrical center on its path, \( v_i(t) \) is the velocity along the path and, if applicable, \( \tilde{x}_i(t) \) collects all remaining states (e.g. acceleration and/or internal states of the power-train). Both \( f_i \) and \( c_i \) are continuously differentiable.

B. Side Collision Avoidance (SICA)

Side collisions can only occur between vehicles on different lanes, when these are inside an area around the points where the vehicles’ paths intersect. We denote these areas Conflict Zones (CZ), and note that more than one pair \((i, j)\) can have potential collisions at a particular CZ. Collision avoidance consequently amounts to ensuring that vehicles on different lanes occupy each CZ in a mutually exclusive fashion. We enforce this through auxiliary variables for the time of entry and departure of each CZ, implicitly defined through

\[ p_i(t^{\text{in}}_{i,r}) = p^{\text{in}}_{r,i}, \quad \forall r \in Z_i \]
\[ p_i(t^{\text{out}}_{i,r}) = p^{\text{out}}_{r,i}, \quad \forall r \in Z_i \]

where \( Z_i \) collects the indices of the CZ crossed by vehicle \( i \), and \( p^{\text{in}}_{i,r} \) and \( p^{\text{out}}_{i,r} \) are defined as shown in Fig. 3.

A condition for Side Collision Avoidance (SICA) is then

\[ t^{\text{out}}_{r,j} \leq t^{\text{in}}_{r,j}, \quad (i, j, r) \in \mathcal{S}, \]

where \( \mathcal{S} \) collects all vehicle pairs \((i, j)\) and CZ \( r \) where side collisions can occur, and encodes the crossing order.

C. Rear-End Collision Avoidance (RECA)

Due to Assumption 2, rear-end collisions can only occur between two adjacent vehicles on the same lane. We collect all vehicle pairs \((i, j)\) such that \( i \) is immediately behind \( j \) on lane \( j \) in \( \mathcal{R}_j \), and state the necessary condition for Rear-End Collision Avoidance (RECA)

\[ p_i(t) + \delta_{ij} \leq p_q(t), \quad (i, q) \in \mathcal{R}_j, \]

where \( \delta_{ij} = d_i/2 + d_j/2 \). All but the first and last vehicle on a lane has a forward and rearward-facing RECA constraint.

\[^1\]In the event that \( v_i(t^{\text{in}}_{i,r}) = 0 \), \( t^{\text{in}}_{i,r} \) is not uniquely defined by \( p_i(t^{\text{in}}_{i,r}) = p^{\text{in}}_{r,i} \). A practical remedy is to instead use the slightly more complex definition \( t^{\text{in}}_{r,i,j} = \min \{ t | p_i(t^{\text{in}}_{i,r}) = p^{\text{in}}_{r,i} \} \). Since \( \dot{p}(t^{\text{in}}_{i,r}) = 0 \) rarely will be encountered in practice, this is avoided for ease of presentation.
D. Discretization

We employ a direct formulation of the optimal coordination problem, assuming a piece-wise constant parameterization of the inputs \( u_i(t) \). That is, \( u_i(t) = u_{i,k}, \ u_{i,k} \in \mathbb{R}^{m_i}, \ t \in [t_k, t_{k+1}), k = 1, \ldots, K-1, K \in \mathbb{N} \) where \( t_k = k \Delta t \) is the time-discretization size. We introduce the vectors \( x_i = (x_{i,0}, \ldots, x_{i,K}), \ x_{i,k} \in \mathbb{R}^{n_i} \) and consider a multiple shooting discretization of the dynamics (1a), enforcing

\[
x_{i,0}^* = \hat{x}_{i,0},
\]

\[
x_{i,k+1} = F_i(x_{i,k}, u_{i,k}, \Delta t), \quad k = 0, \ldots, K-1,
\]

where \( \hat{x}_{i,0} \) is the initial state of vehicle \( i \). Here, \( F_i(x_{i,k}, u_{i,k}, \Delta t), F_i : \mathbb{R}^{n_i} \times \mathbb{R}^{m_i} \times \mathbb{R} \rightarrow \mathbb{R}^{n_i} \) denotes the solution to (1a) at \( t = t_k + \Delta t \), when \( x_i(t_k) = x_{i,k} \) and \( u_i(t) = u_{i,k} \). The state and control trajectories \( x_i(t) \) and \( u_i(t) \) are thereby described by \( x_i \) and \( u_i = (u_{i,0}, \ldots, u_{i,K}) \), which we collect as \( w_i = (x_i, u_i) \). Moreover, we express the position \( p_i(t) \) at time \( t \) as a function of \( w_i \)

\[
p_i(t, w_{i,v}) = F_{i,p}(x_{i,k}, u_{i,k}, t - t_k), \quad k = \lfloor t/\Delta t \rfloor,
\]

where \( F_{i,p} : \mathbb{R}^{n_i} \times \mathbb{R}^{m_i} \times \mathbb{R} \rightarrow \mathbb{R} \) denotes the position component of \( F_i \). Consequently, \( x_{i,r}^{\text{in}}, t_{i,r}^{\text{in}} \) are well-defined, continuous functions of \( w_i \) through

\[
p_{i_r}^{\text{in}}(t_{i_r}^{\text{in}}, w_{i,v}) = p_{i_r}^{\text{in}}, \quad \forall r \in Z_i,
\]

\[
p_{i_r}^{\text{out}}(t_{i_r}^{\text{out}}, w_{i,v}) = p_{i_r}^{\text{out}}, \quad \forall r \in Z_i,
\]

when (6) holds.

Finally, we only consider enforcement of constraints (1b) and the RECA constraints (5) at times \( t_k \) so that

\[
c_i(x_{i,k}, u_{i,k}) \leq 0, \quad k = 0, \ldots, K,
\]

\[
p_i + \delta_{i,q} \leq p_{i,q}, \quad k = 0, \ldots, K, \quad (i, q) \in R_j.
\]

E. Optimal Control Formulation

We define

- \( \mathcal{N} = \{1, \ldots, N\} \) The set of all vehicles
- \( \mathcal{N}_L = \{1, \ldots, L\} \) The set of all lanes
- \( \mathcal{N}_j \) The set of all vehicles on lane \( j \)

and let

- \( T_{v_i} \) collect \( T_{r,i} = (t_{i,r}^{\text{in}}, t_{i,r}^{\text{out}}), \forall r \in Z_i \)
- \( T = (T_{v_1}, \ldots, T_{v_N}) \)
- \( y_{v_i} = (w_{v_i}, T_{v_i}) \)
- \( y = (y_{v_1}, \ldots, y_{v_N}) \)
- \( p_{v_i} = (p_{v_1}, \ldots, p_{v_i,K}) \)
- \( p_{l_j} \) collect \( p_{v_i}, \forall i \in \mathcal{N}_j \)
- \( p \) collect \( p_{v_i}, \forall i \in \mathcal{N} \)
- \( g(v_i) = 0 \) collect constraints (6), (9)
- \( h(v_i)(y_i) \leq 0 \) collect constraints (11)
- \( h_{i_j}(p_{i_j}) \leq 0 \) collect constraints (12)
- \( h_T(T) \leq 0 \) collect constraints (4).

The objective functions consider are on the form

\[
J_{v_i}(y_{v_i}) = V_i^T(x_{i,N}) + \sum_{k=0}^{K-1} \ell_k(x_{i,k}, u_{i,k}),
\]

with the continuously differentiable terminal cost \( V_i^T : \mathbb{R}^{n_i} \rightarrow \mathbb{R} \) and stage cost \( \ell_k : \mathbb{R}^{n_i} \times \mathbb{R}^{m_i} \rightarrow \mathbb{R} \).

The problem of optimal intersection coordination is then

\[
\min_y \sum_{i=1}^{N} J_{v_i}(y_{v_i}) \tag{14a}
\]

s.t.

\[
g_{v_i}(y_i) = 0, \quad i \in \mathcal{N} \tag{14b}
\]

\[
h_{v_i}(y_i) = 0, \quad i \in \mathcal{N} \tag{14c}
\]

\[
h_{i_j}(p_{i_j}) \leq 0, \quad j \in \mathcal{N}_L \tag{14d}
\]

\[
h_T(T) \leq 0. \tag{14e}
\]

Note that in its “full” form, problem (14) includes selection of the crossing order \( S \). This is a difficult combinatorial problem, and finding exact solutions is in general intractable. In this paper, we assume that the crossing order \( S \) is provided by a heuristic. Problem (14) is consequently denoted the fixed-order problem, and is a continuous Nonlinear Program (NLP).

**Remark 1.** We present Problem (14) without turning vehicles for simplicity, but note that the same formalism \[18\]. Moreover, both (4) and (5) would in practice be defined with state-dependent margins. Such details are omitted for brevity.

III. SOLUTION WITH AN INTERIOR POINT ALGORITHM

The primal-dual interior point (PDIP) algorithms are iterative procedures devised to find (local) minimizers of inequality constrained NLPs. They operate by taking Newton-type steps in the primal-dual variables on the perturbed first-order necessary conditions for optimality (FONC). By simultaneously driving the perturbation to zero, a sequence of primal-dual solution candidates results. This sequence converge to local a minimum of the NLP under some conditions \[23\].

A. A PDIP formulation of the fixed order problem

Collecting (14a) in \( J(w) \), (14b) in \( g(y) \) and (14c)-(14e) in \( h(y) \), the perturbed FONC of Problem (14) are

\[
\nabla_y L = 0, \tag{15a}
\]

\[
g(y) = 0, \tag{15b}
\]

\[
h(y) + s \leq 0, \tag{15c}
\]

\[
D(s) \mu - 1 \tau = 0, \tag{15d}
\]

\[
\mu \geq 0, \tag{15e}
\]

\[
s \geq 0. \tag{15f}
\]

Here, \( s \) is the slack variable associated with \( h \), \( D(s) \) is the matrix with the elements of \( s \) on the main diagonal, \( \tau \in \mathbb{R}_+ \) is the barrier parameter and \( \nabla_y L \) is the gradient of the Lagrange function

\[
L(y, \lambda, \mu) = J(w) + \lambda^T g(y) + \mu^T h(y), \tag{16}
\]

where \( \lambda \) and \( \mu \) are the Lagrange multipliers associated with constraints \( g \) and \( h \) respectively. Collecting \( y, \lambda, \mu, s \) in \( z \), we write (15a)-(15d) as \( r(z) = 0 \).

Starting from \( z^{[0]} \) strictly satisfying (15e)-(15f), the sequence of primal-dual solution candidates is generated through the Newton-iteration

\[
z^{[k+1]} = z^{[k]} + \alpha^{[k]} \Delta z^{[k]}, \tag{17}
\]
where \( k \) is the iteration index, \( \alpha^{[k]} \) the step size and \( \Delta z^{[k]} \) the search-direction. The latter is obtained as the solution of the KKT-system
\[
M(z^{[k]}) \Delta z^{[k]} = -r_{\tau|k}(z^{[k]}),
\]
where the matrix \( M(z) \) is known as the KKT-matrix, which is constructed from \( \frac{\partial L}{\partial z} \), evaluated at \( z^{[k]} \). The step-size is selected such that the updated solution candidate \( z^{[k+1]} \) strictly satisfies (15e), (15f) and provides sufficient decrease on the merit function \( \phi(z) \) and strictly satisfies (15e),(15f). Finally, as the PDIP algorithm progresses, the barrier parameter is \( \tau^{[k]} \) is updated, i.e.,
\[
\tau^{[k+1]} = \xi(\tau^{[k]}, z^{[k]}),
\]
using some update strategy \( \xi \), so that eventually \( \tau^{[k]} \to 0 \).

B. Distribution Strategy

If the PDIP algorithm is applied in a fully centralized setting, the linear system (18) is solved using a standard linear algebra routine. The information needed to assemble \( M(z) \) and \( r_{\tau|k}(z^{[k]}) \) must thus be made available centrally before the search-direction \( \Delta z^{[k]} \) can be found. However, the structure of Problem (14) admits (18) to be solved with most computations performed separately for each vehicle and for each lane. Due to this, \( \Delta z^{[k]} \) can be computed without constructing and solving the full KKT-system centrally. Moreover, the evaluation of the merit function can be split along the same lines, whereby the step size \( \alpha^{[k]} \) can be selected in a distributed fashion. In the following sections, we detail how these distributions are made and the information-exchange required between the vehicles, lane-centers and intersection-center.

IV. SOLVING THE KKT-SYSTEM

In this section, we construct \( r_{\tau|k}(z^{[k]}) \) and \( M(z^{[k]}) \) for Problem (14). For brevity, we omit the iteration index and dependence on \( \tau \) in what follows, and only include the arguments of functions to highlight specific dependencies.

We first note that the program consists of three levels.
1) The Vehicle-Level, consisting of the objective function (14a) and constraints (14b), (14c). The functions can be evaluated independently for each vehicle.
2) The Lane-Level: The RECA constraints (14d) couple vehicles on the same lane. The function \( g_{l_j}(p_{l_j}) \) can be evaluated independently for each lane.
3) The Intersection-Level: The SICA constraints (14e) couple vehicles on different lanes. The function \( h_T(T) \) require information from all vehicles in the scenario.

In the sequel, the use of “Vehicle-level”, “Lane-level” and “Intersection-level” refers to the functions as discussed here.

A. Variable Definitions

We use the following notation:
- \( \lambda_{v_i} \) multiplier for constraint (14b) of vehicle \( i \)
- \( \mu_{v_i} \) multiplier for constraint (14c) of vehicle \( i \)
- \( s_{v_i} \) slack variable for constraint (14c) of vehicle \( i \)
- \( \mu_{v_i}^{L} \) multiplier for forward RECA constraint of vehicle \( i \)
- \( \mu_{T(i)} \) multiplier for SICA constraints of vehicle \( i \)
- \( \mu_{l_j}^{s} \) multiplier for constraint (14d) on lane \( j \)
  Collects \( \mu_{l_j}^{s} \), \( \mu_{v_i}^{L} \) \( \forall i \in \mathcal{N}_{l_j} \)
- \( s_{l_j} \) slack variable for constraint (14d) on lane \( j \)
- \( s_t = (s_{l_1}, \ldots, s_{L_f}) \)
- \( \mu_i = (\mu_{s_1}, \ldots, \mu_{s_L}) \)
- \( \mu_T \) multiplier for constraint (14e) on lane \( j \) Collects \( \mu_{T(i)} \), \( \forall i \in \mathcal{N} \)
- \( s_T \) slack variable for constraint (14e) on lane \( j \)

and collect the primal-dual variables as
\[
\begin{align*}
z_{v_i} &= (y_{v_i}, \lambda_{v_i}, \mu_{v_i}, s_{v_i}) \\
z_o &= (z_{v_1}, \ldots, z_{v_N}) \\
z_{l_j} &= (\mu_{s_1}, s_{l_j}) \\
z_l &= (z_{l_1}, \ldots, z_{l_L}) \\
z_T &= (\mu_T, s_T) \\
z &= (z_{v_1}, z_{l_1})
\end{align*}
\]

Note that since the vehicles are coupled through the SICA constraints (14e), each element in \( \mu_T \) is found in both \( \mu_{T(i)} \) and \( \mu_{T(j)} \) when \( (i, j, r) \in \mathcal{S} \). Similarly, due to the RECA constraints (14d), \( \mu_{l_j}^{s} = -\mu_{v_i}^{L} \) when \( (i, q) \in \mathcal{R} \).

Additionally, we use the notion \( z_{v_i}(j) \) to collect \( z_{v_i} \) \( \forall i \in \mathcal{N}_{l_j} \), and denote the lane of vehicle \( i \) as \( l(i) \). For ease of presentation, we also assume that the vehicles are ordered as they appear on each lane, so that \( z_{v} = (z_{v(1)}, \ldots, z_{v(L)}) \).

B. KKT-System

Note that all primal-variables \( y \) are at the vehicle level of the problem. The elements of (15a) relating to the primal variables of vehicle \( i, y_{v_i} \) are
\[
\begin{align*}
\nabla_{y_{v_i}} L &= \nabla y_{v_i} J_{v_i} + \nabla y_{v_i} g_{v_i} \lambda_{v_i} + \nabla y_{v_i} g_{v_i} \mu_{v_i} + \\
&\quad \frac{\partial g_{v_i}}{\partial y_{v_i}} (\mu_{v_i}^{L} - \mu_{v_i}^{T}) + \frac{\partial y_{v_i}}{\partial \mu_{v_i}} \frac{\partial h_{v_i}}{\partial \mu_{v_i}} \mu_{T(i)}.
\end{align*}
\]

Here, \( \frac{\partial h_{v_i}}{\partial \mu_{v_i}} \) is a square matrix with 1 and -1 on the main diagonal where the sign of a particular element is determined by \( \mathcal{S} \) through (4). We define
\[
\begin{align*}
r_{v_i}(z_{v_i}, \mu_{v_i}^{L}, \mu_{v_i}^{T}, \mu_{T(i)}) &= \begin{bmatrix}
\nabla_{y_{v_i}} L(y_{v_i}, \mu_{v_i}^{L}, \mu_{v_i}^{T}, \mu_{T(i)}) \\
\nabla y_{v_i} g_{v_i} \\
\nabla y_{v_i} h_{v_i} \\
\n\mu_{v_i} + D(s_{v_i})^{-1} 1_{\tau}
\end{bmatrix},
\end{align*}
\]

\[
\begin{align*}
r_{l_j} (p_{l_j}, z_{l_j}) &= \begin{bmatrix}
\nabla h_{l_j}(p_{l_j}) + s_{l_j}
\n\mu_{v_i} + D(s_{v_i})^{-1} 1_{\tau}
\end{bmatrix},
\end{align*}
\]

\[
\begin{align*}
r_{l}(p, z_{l}) &= \begin{bmatrix}
\nabla h_{l}(p) \\
\n\mu_{l_j} + D(s_{l_j})^{-1} 1_{\tau}
\end{bmatrix},
\end{align*}
\]

\[
\begin{align*}
r_{T}(T, z_{T}) &= \begin{bmatrix}
\nabla h_{T}(T) + s_{T}
\n\mu_{T} + D(s_{T})^{-1} 1_{\tau}
\end{bmatrix},
\end{align*}
\]

\[
\begin{align*}
M_{v_i}(z_{v_i}) &= \begin{bmatrix}
\nabla^2_{y_{v_i}} L(y_{v_i}) \\
\nabla y_{v_i} g_{v_i} \\
\nabla y_{v_i} h_{v_i} \\
\n\mu_{v_i} + D(s_{v_i})^{-1} D(\mu_{v_i})
\end{bmatrix},
\end{align*}
\]

(25)
where

\[
\Theta = \left( M_T - \sum_{i=1}^{N} M_{Tv_i} M_{v_i}^{-1} M_{v_i}^{T} - \sum_{j=1}^{L} \Lambda_j \Gamma_j^{-1} \Lambda_j^\top \right),
\]

\[
\theta = -r_T + \sum_{i=1}^{N} M_{Tv_i} M_{v_i}^{-1} r_{v_i} - \sum_{j=1}^{L} \Lambda_j \gamma_j,
\]

\[
\Gamma_j = M_{ij} - M_{ijv(j)}^{-1} M_{v(j)} v_{ij},
\]

\[
\gamma_j = -r_{ij} + M_{ijv(j)} M_{v(j)}^{-1} r_{v(j)},
\]

\[
\Lambda_j = -M_{Tv(j)} M_{v(j)}^{-1} M_{ijv(j)},
\]

\[\Delta z_{v} = r_{v} - [M_{vl} M_{vT}] \begin{bmatrix} \Delta z_l \\ \Delta z_T \end{bmatrix}, \]

\[\Theta \Delta z_{T} = \theta, \quad \Gamma \Delta z_{l} = \gamma - \Lambda^\top \Delta z_{T},\]

\[\theta = \psi - \Lambda \Gamma^{-1} \gamma,\]

Due to (32) and (28) we have that

\[
\Gamma = \text{blockdiag}(\Gamma_1, \ldots, \Gamma_L),
\]

\[
\gamma_j = \langle \gamma_1, \ldots, \gamma_L \rangle,
\]

\[
\gamma_j = -r_{ij} + M_{ijv(j)} M_{v(j)}^{-1} r_{v(j)},
\]

which gives

\[
\Lambda = \begin{bmatrix} \Lambda_1, \ldots, \Lambda_L \end{bmatrix},
\]

\[
\Lambda_j = -M_{Tv(j)} M_{v(j)}^{-1} M_{ijv(j)},
\]

so that

\[
\Lambda \Gamma^{-1} \Lambda^\top = \sum_{j=1}^{L} \Lambda_j \Gamma_j^{-1} \Lambda_j^\top, \quad \Lambda \Gamma^{-1} = \sum_{j=1}^{L} \Lambda_j \Gamma_j^{-1} \gamma_j^\top.
\]

and (43) can be solved as the \( L \) separate sets of equations

\[
\Gamma_j \Delta z_{l} = \gamma_j - \Lambda_j^\top \Delta z_{T}, \quad \forall j \in N_L.
\]
Algorithm 1 Distributed solution of KKT system. Here, $C$ denotes the central unit for the intersection, $l_j$ denotes the central unit for lane $j$, and $v_i$ denotes vehicle $i$. All vehicles are assumed to hold copies of $\mu_{v_i}^1$, $\mu_{v_i}^2$, and $\mu_{T(i)}$.

1: \textbf{procedure} SearchDirection(\(z, \tau\))
2: \hspace{1em} \forall v_i: Compute \(D_{v_i\rightarrow C}, D_{v_i\rightarrow l(i)}\), and pass to $C$ and $l(i)$
3: \hspace{1em} \forall j: Compute $D_{j\rightarrow C}$ and pass to $C$
4: \hspace{1em} \hspace{1em} $C':$ Assemble and solve (33a), pass the appropriate parts of $\Delta \mu_T$ to all $l_j$, and all $v_i$
5: \hspace{1em} \hspace{1em} Solve (33b) for $\Delta z_{l_j}$, using the received components of $\Delta \mu_T$, pass the appropriate parts of $\Delta \mu_{ij}$ to all $v_i$ on lane $j$
6: \hspace{1em} \forall v_i: Solve (33c) for $\Delta z_{v_i}$, using the received components of $\Delta \mu_T$ and $\Delta \mu_{ij}$
7: \textbf{end procedure}

Similarly, due to (26)

$$M_{Tv}M_{v}^{-1}M_{Tv} = \sum_{i=1}^{N} M_{Tv} M_{v_i}^{-1} M_{Tv}, \quad (54)$$

and (36) can be solved as the $N$ sets of equations

$$M_{Tv} \Delta z_{v_i} = r_{v_i} - [M_{v_i} M_{Tv}] \begin{bmatrix} \Delta z_{l_j} \\ \Delta z_{T} \end{bmatrix}, \quad \forall i \in \mathcal{N}. \quad (56)$$

Proposition 1 gives that (31) can be solved in a dynamic programming-like fashion, with an upwards and a downwards pass through the problem levels. Algorithm 1 summarizes this procedure in a setting where the vehicle-level computations are performed on the vehicles, the lane-level computations are performed on lane-centers and the intersection-level computations are performed at the intersection-center (c.f. Fig. 1). Here, $D_{v_i\rightarrow C}, D_{v_i\rightarrow l(i)}$ collects the data sent from the vehicle to the lane and intersection-centers, and $D_{l_j\rightarrow C}$ the data sent from a lane-center to the intersection-center, to be detailed in the next sub-section.

Note that this enables parallelization, since Lines 2 and 3 are separable between the vehicles, and Lines 3 and 5 are separable between the lanes. Note also that the factors of matrices $M_{v_i}, \forall i \in \mathcal{N}$ and $\Gamma_j \forall j \in \mathcal{N}_j$ computed on Lines 2 and 3 can be stored and reused on Lines 6 and 5, respectively.

D. Message-passing between the problem levels

The vehicles need to supply $D_{v_i\rightarrow l(i)}$ to the lane-centers and $D_{v_i\rightarrow C}$ to the intersection-center to enable construction of (33b) and (33a). Similarly, the lane-centers need to supply $D_{l_j\rightarrow C}$ to the intersection-center. These matrices are sparse and contain the following data.

Content of $D_{v_i\rightarrow l(i)}$: The lane-centers need to send (34c)-(34e) to evaluate $\Lambda_j \Gamma_j^{-1} \Lambda_j^T$ and $\Lambda_j \Gamma_j^{-1} \gamma_j$. Since the RECA and SICA constraints are separable between the vehicles, we have that

$$M_{l_j(v)(j)} M_{v_i}^{-1} M_{l_j(v)(j)}^T = \sum_{i \in \mathcal{N}_j} M_{l_j(v)(j)} M_{v_i}^{-1} M_{l_j(v)(j)}$$

$M_{l_j(v)(j)} M_{v_i}^{-1} M_{l_j(v)(j)}^T = \sum_{i \in \mathcal{N}_j} M_{l_j(v)(j)} M_{v_i}^{-1} r_{v_i}$

$$M_{Tv(j)} M_{v_i}^{-1} r_{Tv(j)} = \sum_{i \in \mathcal{N}_j} M_{Tv(j)} M_{v_i}^{-1} r_{Tv(i)}$$

The information sent from vehicle $i$ to its lane-center is therefore $M_{Tv(i)} M_{v_i}^{-1} M_{Tv(i)}$ (to build $\Gamma_j$), $M_{Tv(i)} M_{v_i}^{-1} r_{Tv(i)}$ and $p_{Tv(i)}$ (to build $\gamma_j$), $M_{Tv(i)} M_{v_i}^{-1} M_{Tv(i)}$ (to build $\Lambda_j$). Due to (12) and (4), the non-zero elements of $M_{Tv(i)}$ are in the columns corresponding to $p_{Tv(i)}$, so that $D_{v_i\rightarrow l(i)}$ contains

$$\frac{\partial z_{v_i}^T}{\partial p_{Tv}} M_{v_i}^{-1} \frac{\partial z_{v_i}}{\partial p_{Tv}}, \quad \frac{\partial z_{v_i}^T}{\partial r_{Tv}} M_{v_i}^{-1} \frac{\partial z_{v_i}}{\partial r_{Tv}}, \quad \frac{\partial z_{v_i}^T}{\partial r_{Tv}} M_{v_i}^{-1} r_{Tv}, \quad p_{Tv}.$$ \hfill (60)

Content of $D_{v_i\rightarrow C}$: The information sent from vehicle $i$ to the intersection-center is $M_{Tv(i)} M_{v_i}^{-1} r_{Tv(i)}$ (to build $\Theta$), $M_{Tv(i)} M_{v_i}^{-1} M_{Tv(i)}$ and $p_{Tv(i)}$ (to build $\theta$). Due to (4) the non-zero elements of $M_{Tv(i)}$ are in the columns corresponding to $T_{Tv(i)}$, so that $D_{v_i\rightarrow C}$ contains

$$\frac{\partial z_{v_i}^T}{\partial T_{Tv}} M_{v_i}^{-1} \frac{\partial z_{v_i}}{\partial T_{Tv}}, \quad \frac{\partial z_{v_i}^T}{\partial T_{Tv}} M_{v_i}^{-1} r_{Tv}, \quad T_{Tv}. \quad (62)$$

Communication from $C$ to $l_j$: The intersection-center needs to pass $\Delta \mu_{T(i)} \forall i \in \mathcal{N}_j$ to construct the right-hand side of (33b) for lane $j$.

Communication from $l_j$ and $C$ to $v_i$: The dependencies on $\Delta \mu_T$ and $\Delta \mu_{ij}$ in (33c) reads as

$$[M_{v_i}, M_{Tv}] \begin{bmatrix} \Delta z_{l} \\ \Delta z_{T} \end{bmatrix} = \frac{\partial z_{v_i}}{\partial p_{Tv}} (\Delta \mu_{v_i} - \Delta \mu_{v_i}^T) + \frac{\partial z_{v_i}}{\partial T_{Tv}} \frac{\partial h}{\partial T_{Tv}} \Delta \mu_{T(i)}. \quad (63)$$

That is, the intersection-center must pass $\Delta \mu_{T(i)}$, and the lane-center for vehicle $i$ need to send $\Delta \mu_{v_i} - \Delta \mu_{v_i}^T$ to construct the right-hand side of (33c) for vehicle $i$.

V. DISTRIBUTED COMPUTATION OF THE STEP-SIZE

In this section, we discuss selection of the step size $\alpha$ through a back-tracking line-search on a merit function where most computations can be separated between, and computed in parallel within, the problem levels.

A. Feasibility enforcing step-size selection

To ensure that $\alpha$ is chosen so that $s[k+1] > 0$, $\mu[k+1] > 0$, we employ the fraction from the boundary rule

$$s + \alpha^\text{max} \Delta s \geq \kappa s,$$

$$\mu + \alpha^\text{max} \Delta \mu \geq \kappa s.$$ \hfill (64a)

$$\mu + \alpha^\text{max} \Delta \mu \geq \kappa s.$$ \hfill (64b)
where $\kappa > 0$ is a parameter [23]. Due to the problem structure, (64) can be evaluated separately for the vehicles, giving $\alpha_{vi}^{\max}$, $\forall i \in \mathcal{N}$, for the RECA constraints on a lane, giving $\alpha_{ji}^{\max}$, $\forall j \in \mathcal{N}_L$ and for the SICCA constraints $\alpha_T^{\max}$. The maximum allowed step size for the search direction $\Delta z$ is thereby

$$\alpha^{\max} = \min(\alpha_{vi}^{\max}, \ldots, \alpha_{vn}^{\max}, \alpha_{ti}^{\max}, \ldots, \alpha_{tn}^{\max}, \alpha_T^{\max}).$$ (65)

**B. Solution-improving step-size selection**

We find $\alpha \leq \alpha^{\max}$ which improves the solution by back-tracking on the merit function suggested in [23], which reads

$$\phi(y, s) = \sum_{i=1}^{N} \phi_{vi}(y_{vi}, s_{vi}) + \sum_{j=1}^{L} \phi_{Tj}(p_{ij}, s_{ij}) + \phi_T(T, s_T),$$

where

$$\phi_{vi}(y_{vi}, s_{vi}) = J_{vi}(y_{vi}) + \nu(||g_{vi}(y_{vi})||_1 + ||h_{vi}(y_{vi}) + s_{vi}||_1) - \tau_T^1 \log(s_{vi}),$$

$$\phi_{Tj}(p_{ij}, s_{ij}) = \nu||h_{Tj}(p_{ij}) + s_{ij}||_1 - \tau_T^1 \log(s_{ij}),$$

$$\phi_T(T, s_T) = \nu||h_{T}(T) + s_T||_1 - \tau_T^1 \log(s_T),$$

with the logarithm taken element-wise and parameter $\nu$. We use the Armijo condition, and accept a step $\alpha$ when

$$\phi(y + \alpha \Delta y, s + \alpha \Delta s) \leq \phi(y, s) + \zeta \phi'(y, s)\alpha,$$ (67)

where $\zeta \in [0, 0.5]$ is a parameter, and

$$\phi'(y, s) = \frac{\partial \phi(y + \alpha \Delta y, s + \alpha \Delta s)}{\partial \alpha}\bigg|_{\alpha=0} = \frac{\partial \phi}{\partial y} \Delta y + \frac{\partial \phi}{\partial s} \Delta s$$

$$= \sum_{i=1}^{N} \phi_{vi}'(y_{vi}, s_{vi}) + \sum_{j=1}^{L} \phi_{Tj}'(p_{ij}, s_{ij}) + \phi_T'(T, s_T).$$ (68)

Evaluation of $\phi(y, s), \phi'(y, s)$ can be separated between the vehicles ($\phi_{vi}(y_{vi}, s_{vi}), \phi_{vi}(y_{vi}, s_{vi})$), lanes ($\phi_{Tj}(p_{ij}, s_{ij}), \phi_{Tj}(p_{ij}, s_{ij})$) and the intersection ($\phi_T(T, s_T), \phi_T'(T, s_T)$).

**Algorithm 2 Distributed selection of step-size $\alpha$, first level.**

1. **procedure** STEP SIZE SELECTION ($z, \Delta z, \tau$)
2. $\forall i$: Find $\alpha_{vi}^{\max}$, assemble $\phi_{vi}(y_{vi}, s_{vi})$ and $\phi_{vi}^\prime(y_{vi}, s_{vi})$, pass to $C$ together with $\Delta T_{vi}$, pass $\Delta P_{vi}$ to $l(i)$.
3. $\forall j$: Find $\alpha_{Tj}^{\max}$, assemble $\phi_{Tj}(p_{ij}, s_{ij})$ and $\phi_{Tj}^\prime(p_{ij}, s_{ij})$.
4. $C$: Find $\alpha_T^{\max}$ and determine $\alpha_T^{\max}$ with (65).
5. $C$: Find $\phi_T(T, s_T), \phi_T'(T, s_T)$ ass. $\phi(y, s), \phi'(y, s)$
6. $C$: Set $\alpha = \alpha_T^{\max}$
7. **loop**
8. $\forall i$: Pass $\phi_{vi}(y_{vi} + \alpha \Delta y_{vi}, s_{vi} + \alpha \Delta s_{vi})$ to $C$
9. $\forall j$: Pass $\phi_{Tj}(p_{ij} + \alpha \Delta p_{ij}, s_{ij} + \alpha \Delta s_{ij})$ to $C$
10. $C$: Compute $\phi_T(T + \alpha \Delta T, s_T + \alpha \Delta s_T)$, assemble
11. $\phi(y + \alpha \Delta y, s + \alpha \Delta s)$ through (66)
12. **if** $\phi(y + \alpha \Delta y, s + \alpha \Delta s) < \phi(y, s) + \alpha \gamma \phi'(y, s)$
13. **return** $\alpha, \alpha_T^{\max}$ and accept-notice to all $v_i, l_j$
14. **else**
15. $\alpha \leftarrow \beta \alpha$
16. Pass $\alpha$ to all $v_i, l_j$
17. **end if**
18. **end loop**

**1) Termination Criteria:** We use the norm of $r$ as termination criteria, such algorithm terminates when

$$||r_{\tau[k]}|| < \varepsilon \quad \text{and} \quad \tau^{[k]} < \varepsilon,$$ (70)

for some tolerance $\varepsilon$. Termination must thus be decided centrally, and all $r_{vi}, r_{ij}$ must be sent to the intersection-center.

**2) Barrier Parameter Update:** While elaborate schemes are possible for updates of $\tau$, we employ the Fiacco-McCormick rule for simplicity. In particular, we update $\tau^{[k+1]} \leftarrow \eta \tau^{[k]}$, where the parameter $\eta \in [0, 1]$, when $||r_{\tau[k]}|| < \tau^{[k]}$. The barrier parameter must thus be decided centrally.

**A. Example**

As an example, we consider a scenario with three vehicles on each lane. Assuming that all vehicles are electric, their motion is described by

$$\dot{p}_i(t) = v_i(t),$$ (71a)

$$\dot{v}_i(t) = \frac{1}{m_i}(c_{\text{Torque}}M_i(t) - F_i^{\text{br}} - c_{\text{drag}}v_i(t)^2 - c_T),$$ (71b)

$$M(t) \leq \min(M_{\text{max}}, P_{\text{max}}/\omega_i(t)),$$ (71c)

$$0 \leq \omega_i(t) \leq \omega_{i}^{\max},$$ (71d)

where $M_i(t)$ is the motor torque, $F_i^{\text{br}}(t)$ the friction brake force, $\omega_i(t) = c''v_i(t)$ the motor speed and $x_i(t) = (p_i(t), v_i(t)), u_i(t) = (M_i(t), F_i^{\text{br}}(t))$. The parameters $c_{\text{Torque}}, c_{\omega}, c_{\text{drag}}, c_T, \omega_i^{\max}, M_{\text{max}}$ and are selected as in $P_{\text{max}}$ [17], and we use $K = 100$ and Explicit 4th order
Algorithm 3 A Basic Distributed Primal-Dual Interior Point algorithm for the fixed order intersection problem.
1: procedure FIXEDORDERPDIP($\tau^{[0]}$)
2: \[ k \leftarrow 0 \]
3: \[ C: \text{Initialize } z^{[0]}_i \text{ and send } p^{[0]}_i \text{ to all } v_i \text{ and } l_j \]
4: \[ \forall l_j: \text{Initialize } z^{[0]}_j \text{ and send } p^{[0]}_j \text{ to } v(j) \]
5: \[ \forall v_i: \text{Initialize } z^{[0]}_i \text{ and send } T^{[0]}_i \text{ to } C, p^{[0]}_i \text{ to } l_j \]
6: loop
7: \[ k \leftarrow k + 1 \]
8: \[ C: \text{Send iteration start and } \tau^{[k]} \text{ to all } v_i, l_j \]
9: \[ \forall v_i: \text{Compute } M_{v_i}, r_{v_i}, \text{ modify if necessary} \]
10: \[ \nabla^2_{z^{[k]}_i} \text{L.} \]
11: \[ \alpha^{[k]} \leftarrow \text{STEPSIZESELECTION}_{\Delta z} \]
12: \[ C: \text{Update } z^{[k+1]}_i \leftarrow z^{[k]}_i + \alpha^{[k]} \Delta z^{[k]}_i \]
13: \[ \forall l_j: \text{Update } z^{[k+1]}_j \leftarrow z^{[k]}_j + \alpha^{[k]} \Delta z^{[k]}_j \]
14: \[ \forall v_i: \text{Update } z^{[k+1]}_i \leftarrow z^{[k]}_i + \alpha^{[k]} \Delta z^{[k]}_i \]
15: if TERMINATE$(z^{[k+1]}_i, \Delta z^{[k]}_i, \tau^{[k]})$ then
16: \[ \text{return Solution found} \]
17: else
18: \[ \tau^{[k+1]} \leftarrow \text{UPDATEBARRIERPARAMETER()} \]
19: end if
20: end loop
21: end procedure

The objective function is
\[
J_{v_i}(y_{v_i}) = Q_i^T(v_{i,K} - v_r)^2 + \sum_{k=0}^{K} Q_i(v_{i,k} - v_r)^2 + (u_{i,k} - u_r^i)\top R_i(u_{i,k} - u_r^i),
\]
where $v_r$ is a reference speed, and $u_r^i$ is an input which maintains $v_r$. Here, the weights are selected as $Q_i = 1/(v_{r}^i)^2$ and $R_i = \text{diag}(1/F_{\text{max}}^i, 1/F_{\text{max}}^i)$, and $Q_i^T$ is the cost-to-go associated with the LQR controller computed with $Q_i, R_i$ and the linearization of (71b) around $v_r^i$.

The vehicles are initialized randomly between 80 and 120 meters before the intersection, with $v_{i,0} = v_r = 70$ km/h. The initial solution candidate $w^{[0]}_i, T^{[0]}_i$ is that where all vehicles drive at $v_r$ for $k = 0, \ldots, K, \lambda^{[0]}_i = 0, \mu^{[0]}_i = s^{[0]}_i = 1$ and $\tau^{[0]} = 1$.

The development of $||r(z')||_2$ and $\tau[k]$ is shown in Fig. 5a, and the step sizes used is shown in Fig. 5b.

To illustrate how the algorithm progresses, the velocity profiles of one vehicle is provided in Fig. 5c. Note that the final 15 iterates are indistinguishable. In a practical context, little would be lost by stopping after iteration 16.

For illustration, the sparsity-pattern of $M$ is given in Fig. 6. The size is $25832 \times 25832$, where $M_{v_i}$ is of size $24176 \times 24176$, $M_{l_j}$ of size $404 \times 404$ and $M_{T}$ of size $40 \times 40$. Besides evaluating the involved functions and their derivatives, the main computational effort is therefore the factorization of the vehicle blocks $M_{v_i}$, roughly sized $2010 \times 2010$, and the lane blocks $M_{l_j}$, sized $404 \times 404$. Since the factors for all $M_{v_i}$’s can be computed in parallel between the vehicles (Line 2 in Algorithm 1), and the $\Gamma_j$’s can be factorized in parallel between the lanes (Line 3), the computational time required to solve the KKT system $t_{\text{direction}}$ will roughly be
\[
t_{\text{direction}} \approx \max_{i \in N}(\text{timeToFactorize}(M_{v_i}))) + \max_{j \in N_L}(\text{timeToFactorize}(\Gamma_j)) \quad (73)
\]

However, computational time will likely not dominate the time it takes to find a solution. Following the lessons learned from the experiments reported in [16], the time required to communicate between the vehicles, lane-centers and intersection-center is likely larger. In the next section, we analyze the communication requirements, and discuss some modifications to the scheme which decrease both the number of transmissions and the amount of data communicated.

VII. COMMUNICATION REQUIREMENTS

In this section, we discuss the communication requirements of Algorithm 3. We first analyze the data flow between the vehicles, lane-centers and intersection-center in Section VII-A, and demonstrate that the proposed algorithm requires an unrealistic data exchange. We discuss how the requirements can be reduced in Sections VII-B and VII-C.

A. Analysis of Communication Requirements

Most data is exchanged during solution of the KKT-system in Algorithm 1 and the selection of the step-size in Algorithm 2. Descriptions of the data involved as well as the number of floats communicated are summarized in Table I.

Most often $K \gg nT_{\text{r}}$, whereby most communication occurs during Line 2 of Algorithm 1 when the vehicle sends $D_{v_i,f}(i)$. Besides the communication between the lane-centers and the intersection-center and an initial round of communication where the initial guess $z^{[0]}_i, z^{[0]}_j$ is sent to the vehicles, the communication required for the remaining parts (i.e., the indication of a new iteration, the current barrier parameter value, termination of line-search or algorithm completion) consists of single floats and logicals. As illustrated in Fig. 7, these can be sent together with the search-direction and step-size results.

Communication in the Example: In the example $K = 100$, whereby each vehicle sends more than 5000 floats per iterate (more than 320000 bits) to their respective lane-center. Even if all vehicles communicate in parallel, the physical transmission will take at least 58.7 ms using the 802.11p protocol; the current standard for vehicular communications. During 33 iterations, at least 1.94 s would be spent communicating to construct $\Delta z$, much too high for practical applications. Next, we discuss how the data exchange can be reduced.

2This implementation only includes the elements of $T_{v_i}$ needed for the SICA constraints. The size of $T_{v_i}$ therefore vary between different $i$.

3The time per bit is computed using the formula $50 + 8\text{ceil}((n_{\text{data}}\text{bits} + 22)/48) \mu s$, reported in [25]. Double precision is assumed.
To this end, we propose to replace all RECA constraints (12) with constraints on the form

\[ p_{i,k} + \delta_{ij}/2 \leq B_{ij}(k, \theta_{ij}), \quad k = 1, \ldots, K \]  
\[ B_{ij}(k, \theta_{ij}) + \delta_{ij}/2 \leq p_{j,k}, \quad k = 1, \ldots, K, \]  

where \( B_{ij}(k, \theta_{ij}) \) is a function of \( k \), parametrized with \( \theta_{ij} \in \mathbb{R}^q \), and introduce \( \theta_{ij} \) as additional decision variables in the fixed order problem (14). Rather than enforcing the RECA constraints directly using \( p_{vi} \) and \( p_{vj} \), (74) requires that \( B_{ij}(k, \theta_{ij}) \) is between \( p_{i,k} \) and \( p_{j,k} \) at all \( k \), whereby RECA is ensured by selection of the coupling parameter \( \theta_{ij} \). This circumvents the \( K^2 \) growth, and enables practical schemes which rely on data exchange that scales as \( q^2 \). The price paid is conservativeness and sub-optimality, as the set of feasible trajectories is reduced when \( q < K \).

The parametrized RECA coupling can be included in the distributed scheme in two different ways. In the following discussion, we use notation as shown in (8), where \( \theta_{ij}^1 \) and \( \theta_{ij}^2 \) are coupling parameters of vehicle \( i \)'s forward (74a) and rearward (74b) facing RECA conditions, respectively. That is, if vehicle \( j \) is in front of vehicle \( i \), we have that \( \theta_{ij}^1 = \theta_{ij}^2 = \theta_{ij} \).

Similarly, we let \( B_{vij}^1(\theta_{ij}^1) \) and \( B_{vij}^2(\theta_{ij}^2) \) collect the function \( B_{ij}(k, \theta_{ij}) \) for \( k = 1, \ldots, K \) in the forward- and rearward-facing constraints for vehicle \( i \), respectively, and denote the corresponding multipliers and slacks \( (\nu_{ij}^t, s_{ij}^t) \) and \( (\nu_{ij}^r, s_{ij}^r) \). We also let \( \theta_{ij}^v = (\theta_{ij}^1, \theta_{ij}^2) \), \( B_{vij}(\theta_{ij}^v) = (B_{vij}^1(\theta_{ij}^1), B_{vij}^2(\theta_{ij}^2)) \), \( \nu_{ij}^v = (\nu_{ij}^1, \nu_{ij}^r) \), and \( s_{ij}^RECA = (s_{ij}^t, s_{ij}^r) \) for each vehicle, collect \( \theta_{ij} \) \( \forall (i, j) \in \mathcal{N}_j \) in \( \theta_{ij} \), and collect \( \theta_{ij} \) \( \forall j \in \mathcal{N}_i \) in \( \theta \).

**The “Primal” Approach:** The first alternative is to handle the coupling parameters \( \theta_{ij} \) at the lane-centers, and constraints (74a), (74b) on-board the vehicles. The lane-center variables are in this case \( z_{ij} = \theta_{ij} \), so that \( r_{ij} = \nabla_{\theta_{ij}} \mathcal{L} = \sum_{i \in \mathcal{N}_j} \nabla_{\theta_{ij}} B_{vij} \nu_{ij} \), and the vehicle variables \( z_{ij} \) include \( (\nu_{ij}, s_{ij}^RECA) \). The information sent to the lane-center is
\begin{align*}
\text{TABLE I: Summary of the communication between the vehicles} \quad (v_i), \text{ lane-centers} \quad (l_j) \text{ and intersection-center} \quad (C). \text{ The second column states in which} \\
\text{Algorithm (A) and Line (L) the communication occurs. The numbers under braces denotes the size of the object, where} \\
n_{T_{vi}} \text{ is the size of the object (76).} \\
\text{For all couplings on lane} \quad j \quad \text{and the information sent from vehicle} \quad i \quad \text{on Line 5 of Algorithm 1 is} \\
\text{sent the lane-center to a vehicle on Line 5 of Algorithm 1 consists of} \Delta \theta_{vi} \text{ (2q floats).} \\
\text{Since} \quad p_{vi} \quad \text{and} \quad p_{v_j} \quad \text{are only coupled indirectly, the term} \\
\frac{\partial z_{vi}}{\partial p_{vi}} (\mu^f_{vi} - \mu^l_{vi}) \text{ is not present in } \nabla_{y_i} L \text{ and } \Delta \mu^f_{vi} \text{ or} \\
\Delta \mu^l_{vi} \text{ is not sent by the lane-centers.} \\
\text{The “Dual” Approach: The second alternative is to consider} \\
\theta^u_{vi} \text{ and } \theta^l_{vi} \text{ as separate variables, and couple the vehicles through the constraint} \\
\theta^u_{vi} - \theta^l_{vi} = 0. \quad (76)
\end{align*}
\( \nabla_{y_{i,v}} \mathcal{L} = \nabla_{y_{i,v}} J_{v_i} + \nabla_{y_{i,v}} g_{v_i} \lambda_{v_i} + \nabla_{y_{i,v}} g_{v_i} \mu_{v_i} + \frac{\partial y_{i,v}}{\partial \theta_{v_i}} \pi_{v_i} - \frac{\partial y_{i,v}}{\partial \theta_{v_i}} \pi_{v_i} + \frac{\partial y_{i,v}}{\partial T_{v_i}} \mu_{v_i,T}. \) (78)

Consequently, the lane-center-to-vehicle communication on Line 5 of Algorithm 1 consists of \( \Delta \pi_{v_i}, \Delta \mu_{v_i} \), i.e. 2q floats.

Since both approaches have the same communication footprint, they are both useful alternatives. However, when \( B_{ij}(k, \theta_{ij}) \) is nonlinear in \( \theta_{ij} \) terms appear off the main block-diagonal in \( \nabla_{y_{i,v}}^2 \mathcal{L} \). Due to this, positive definiteness of \( \nabla_{y_{i,v}} \mathcal{L} \) can not be ensured by modifying \( \nabla_{y_{i,v}}^2 \mathcal{L} \) (c.f. the discussion in Section V-C). In Dual approach, all primal variables are at the vehicle level, and block-wise regularization is possible.

Evaluation and Example: By selecting \( \theta_{ij} \) such that \( q \) is small, significant reductions in the amount of data communicated are achieved at the cost of sub-optimality. To evaluate the cost-benefit trade-off we consider the case shown in Fig. 9, where \( B_{ij}(k, \theta_{ij}) \) is the piece-wise linear function

\[
B_{ij}(k, \theta_{ij}) = \begin{cases} 
\theta_{ij}^{(1)} + \frac{\theta_{ij}^{(2)} - \theta_{ij}^{(1)}}{[K/3]} k & k \in \{0, [K/3]\} \\
\theta_{ij}^{(2)} + \frac{\theta_{ij}^{(3)} - \theta_{ij}^{(2)}}{[K/3]} (k - [K/3]) & k \in \{[K/3] + 1, 2[K/3]\} \\
\theta_{ij}^{(3)} + \frac{\theta_{ij}^{(4)} - \theta_{ij}^{(3)}}{[K/3]} (k - 2[K/3]) & k \in \{2[K/3] + 1, K + 1\},
\end{cases}
\]

where the superscript on \( \theta_{ij} \) indicates the element (i.e. \( q = 4 \)). When \( n_{T_{v_i}} = 4 \), no more than 60 floats are sent from a vehicle to the lane-center on Line 5 of Algorithm 1, which theoretically will take at least 0.7 ms (see footnote on Page 8), a reduction of almost 99 \%.

To assess the sub-optimality induced, we evaluated 500 scenarios with 4 vehicles per lane (16 in total), using the models and objective functions of Section VI-A. The vehicles were initialized at randomly drawn distances between 50 and 150 meters from the intersection, and the crossing order was computed with the heuristic of [19]. As Fig. 10 demonstrates, the sub-optimality induced by the parametrized constraints is below 0.1 \% in more than 50\% of the cases. The small impact is illustrated in Fig. 11, which shows the difference in the optimal velocity profiles for the scenario corresponding to the median sub-optimality. Interestingly, the difference between the optimal control commands at \( k = 0 \) in the two solutions is less than 0.013 \% of the input range for all vehicles. This is below the quantization error of many actuators, whereby the difference might not be noticed in practice. Finally, more “flexible” parametrization of \( B_{ij} \) could be used to reduce sub-optimality, e.g. by including additional linear segments or polynomials.

C. Reduction of the number of communication rounds

Algorithm 3 is rudimentary and could be augmented to converge in fewer iterations. For instance, instead of the simple update rule for \( \tau \), adaptive or a Predictor-Corrector strategies could be employed [23].

To reduce the number of iterations, the problem to a sufficiently high accuracy for larger \( \tau \). In this case, all equality constraints are satisfied to the set tolerance, and the inequality constraints are satisfied with a margin. While convergence to is reached in fewer iterations, sub-optimality is introduced. However, as remarked in the discussion on Fig. 5c, practically acceptable solutions can be obtained for \( \tau \) larger than relevant tolerances on \( ||r(z)|| \). It is therefore expected that the practical implications of stopping Algorithm 3 at \( \tau \) significantly larger than \( \varepsilon \) is small. As an example, Fig. 12 shows results from the scenario considered in Section VI-A, where \( \tau \) is prevented from being smaller than \( \tau_{\text{min}} \), for \( \tau_{\text{min}} \) between 1 and \( 10^{-6} \), with \( \varepsilon = 10^{-6} \) for all cases. The sub-optimality induced and the number iterations required to reach \( ||r(z)|| \leq \varepsilon \) is shown in Fig. 12. Note for instance that 23 iterations are required for \( \tau_{\text{min}} = 10^{-2} \), compared to 33 in case of \( \tau_{\text{min}} = 10^{-6} \), at the expense of less than 1 \% sub-optimality. The optimal velocity profiles for one vehicle for the different values of \( \tau_{\text{min}} \) is shown in Fig. 13 (c.f. Fig. 5c ). The difference with
Fig. 12: Sub-optimality (red) and number of iterations (blue) required to reach $||r(x)|| \leq \epsilon$ for different value of $\tau_{\min}$.

Fig. 13: Velocity profiles for a vehicle corresponding to the solution satisfying $||r(x)||_2 \leq 10^{-6}$ for different values of $\tau_{\min}$. For the red trajectory $\tau_{\min} = 1$ and for the green trajectory $\tau_{\min} = 10^{-6}$. Hues between red and green correspond to intermediate values.

respect to the optimal solution is small enough to be practically irrelevant for all but the highest value of $\tau_{\min}$.

VIII. CONCLUSION

In this paper we presented a Primal-Dual Interior Point algorithm for the optimal coordination of automated vehicles at intersections under a fixed crossing order. The algorithm is motivated by deficiencies in earlier work, and enables inclusion of complicating rear-end collision avoidance constraints. We showed that the problem is structured so that the KKT-system can be solved in steps, where most operations are parallelized and solved separately for all vehicles and for all lanes. We demonstrated that step-size selection through backtracking on a merit function can be distributed under the same pattern. To reduce the data exchange, we proposed a parameterized and slightly conservative re-formulation of the rear-end collision avoidance constraints, and demonstrated its merits through randomized evaluation.

We are currently investigating formulations of the coordination problem that allows removal of the restrictive assumption of full CAV penetration. We also aim at extending our approach to scenarios with several connected intersections.

REFERENCES