An MIQP-based heuristic for Optimal Coordination of Vehicles at Intersections

Robert Hult, Mario Zanon, Sébastien Gros and Paolo Falcone

***Abstract*— The problem of coordinating automated vehicles at intersections can be formulated as an optimal control problem which is inherently difficult to solve, due to its combinatorial nature. In this paper, we propose a two-stage approximation algorithm based on a previously presented decomposition. The procedure (a) first solves a Mixed Integer Quadratic Program (MIQP) to compute an approximate solution to the combinatorial part of the problem, i.e. the order in which the vehicles cross the intersection; then (b), solves a Nonlinear Program (NLP) for the optimal state and control trajectories. We demonstrate the performance of the algorithm through extensive simulation, and show that it greatly outperforms the natural First-Come-First-Served heuristic.**

I. Introduction

The problem of coordinating autonomous vehicles at intersections have received increasing attention during the last years [2],[20]. The main idea is to use vehicle-to-vehicle/infrastructure (V2X) communication and coordination algorithms instead of traffic lights, stop-signs and right-of-way rules. This is commonly projected to enable an increase in energy efficiency, mitigate congestion and increase the capacity of existing infrastructure [8].

In a number of recent contributions, the coordination problem has been formulated within the optimal control (OC) framework [15], [3], [7], [16], [22], [14]. In general, OC formulations has a number of advantages, e.g. the ability to explicitly include complex dynamical models and constraints, efficient numerical algorithms and a mature theory for closed-loop application via Model Predictive Control (MPC). In particular, OC formulations enables the optimization of explicitly stated performance metrics, such as energy-minimization or throughput maximization.

However, solving the OC-coordination problem involves finding the optimal order in which the vehicles cross the intersection. The problem is thus combinatorial, and for all but small, simple instances, it is therefore prohibitively hard to solve to optimality. Many of the existing OC-coordination schemes instead suggest heuristics; commonly variations of First-Come-First-Served (FCFS) policies, to find a crossing order, and to find the vehicle trajectories given this order, see e.g. [4], [15], [3], [22]. While such are attractive due to their simplicity, there is a risk that the crossing order has a large impact on the quality of the solution. For instance, FCFS-schemes can easily produce crossing orders in which large number of heavy vehicles must slow down to favor lighter ones, which would be sub-optimal in an energy minimization context. It is thus desirable to find a crossing order heuristic which takes the specified performance metric into account.

In this paper, we propose two such heuristics. We utilize the primal decomposition of the OC-coordination problem presented in [7], [11], where the problem is separated into one Mixed-Integer Nonlinear Program (MINLP), which schedules intersection occupancy time-slots, and one OC problem per involved vehicle, which gives the vehicle trajectories for a given time-slot. However, rather than solving the difficult MINLP, we solve an MIQP where the objective and constraints are quadratic and linear approximations of the MINLP counterparts respectively. While the complexity of MIQPs does not remove the combinatorial explosion of the solution space, it enables the treatment of practically relevant problem sizes due to the availability of efficient MIQP solvers. We develop two Algorithms in which the MIQP heuristic is used to first find the crossing order, and the vehicle trajectories thereafter are found as the solution to a NLP. We investigate the performance of the algorithms through extensive simulation, and compare them to an algorithm where a FCFS heuristic is used.

The remainder of the paper is organized as follows: In Section II we introduce the assumptions and scenario modelling used, and formulate the OC-coordination problem. In Section III we develop the MIQP based heuristics and present Algorithms for the approximate solution to the OC-Coordination problem. In Section IV we describe a simulation setup and present numerical results. Finally, the paper is concluded in Section V.

II. Problem Statement

In this paper we consider scenarios such as that shown in Fig. 1a, where N_a vehicles needs to be coordinated through an intersection efficiently and without collisions. We assume that all vehicles are automated, has communication capabilities and participates in the coordination, and that no other entities, e.g., bicyclists or pedestrians, are present. We consider only simple intersection geometries without turns, and assume that no vehicle is allowed to overtake or change lane during the coordination process.

A. Vehicle Modeling

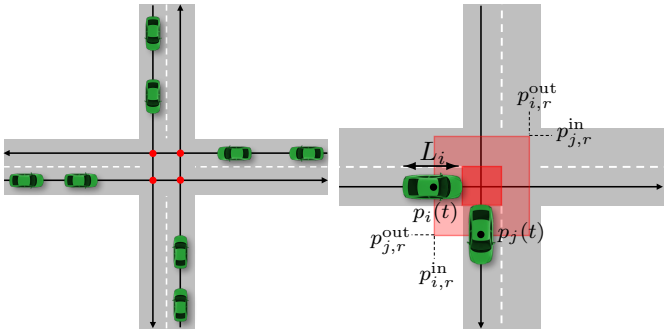
We assume that the vehicles move along fixed paths and let their motion along the paths be described by

$$\dot{x}_i(t) = f_i(x_i(t), u_i(t)), \tag{1}$$

$$0 \geq h_i(x_i(t), u_i(t)), \tag{2}$$

R. Hult, S. Gros and P. Falcone are with the department of Electrical Engineering at Chalmers Unicersity of Technology, Gothenburg, Sweden. {hultr, grosse, pfalc} @chalmers.se

M. Zanon is with IMT School for Advanced Studies, Lucca, Italy mario.zanon@imtlucca.it



(a) Illustration of the scenarios considered in this paper. The conflict points are marked in red.

(b) Illustration of the conflict zone definitions.

Fig. 1: Modeling illustrations

where $u_i(t) \in \mathbb{R}^m$ and $x_i(t) \in \mathbb{R}^n$ are the control an state respectively, and $f_i(\cdot)$ and $h_i(\cdot)$ are smooth functions. In particular, the state vector is such that $x_i(t) = (p_i(t), z_i(t)) \in \mathbb{R}^n$ where $p_i(t)$ is the scalar position of the vehicle center on a coordinate along its path, and $z_i(t) \in \mathbb{R}^{n-1}$ collects all the non-position states. Finally, we assume that (1),(2) are such that $\dot{p}_i(t) > 0$.

B. Intersection Modeling and Collision Avoidance Conditions

In the considered scenarios, collisions can occur both between vehicles on crossing lanes and between vehicles on the same lane. Side collisions between vehicles on crossing lanes are possible in proximity to the *collision points* (CP), which are indicated in red in Fig. 1a. We denote the area around a CP a *conflict zone* (CZ), and denote the entry and exit positions of CZ r on the road coordinate of vehicle i as $p^{\text{in}}_{i,r}$ and $p^{\text{in}}_{i,r}$, respectively. Furthermore, we let the time of entry and exit of CZ r by vehicle i be defined as

$$p_i(t^{\text{in}}_{i,r}) = p^{\text{in}}_{i,r} \quad \text{and} \quad p_i(t^{\text{out}}_{i,r}) = p^{\text{out}}_{i,r}. \tag{3}$$

We define the CZ as in Fig. 1b, taking the vehicle length L_i into account, whereby a sufficient condition for collision avoidance between vehicles i and j thereby is $t^{\text{out}}_{i,r} \leq t^{\text{in}}_{j,r}$, when vehicle i crosses CZ r before vehicle j. We thus require occupancy of a CZ to be mutually exclusive.

Rear-end collisions can occur between vehicles on the same lane, and a necessary condition for rear end collision avoidance is

$$p_j(t) + L_j/2 \leq p_i(t) - L_i/2, \tag{4}$$

when vehicle i is directly in front of vehicle j. While (4) is the least restrictive rear-end collision avoidance condition, we remark that other conditions are possible. For instance speed dependent spacing policies, such as those commonly used in platooning applications could be included [9].

C. Optimal Control Formulation

We let the number of conflict zones be N_{CZ} and the indices of the vehicles that cross CZ r be $\mathcal{I}_r$. We denote the order in which vehicles cross CZ r as $S_r = \{s_{r,1}, \ldots, s_{r,|\mathcal{I}_r|}\}$, where $s_{r,1}$ is the index of the first vehicle to cross, $s_{r,2}$ the second, and so on. Note that $s_{r,i} \in \mathcal{I}_r$ and S_r is a permutation of $\mathcal{I}_r$. We also let the number of lanes be N_l, and let $\mathcal{R}_l$ be all vehicle pairs (i, j) on lane l where i is immediately in front of j. Furthermore, we let the conflict zones crossed by vehicle i be $\mathcal{J}_i$ and the associated timeslot T_i be the set of entry and exit times for vehicle i. Finally, we collect $\mathcal{S} = (S_1, \ldots, S_{N_{\text{CZ}}})$ and $\boldsymbol{T} = (T_i, \ldots, T_{N_a})$.

Finding the optimal state and control trajectories for all vehicles for a given objective thus consists in simultaneously finding the crossing orders $\mathcal{S}$ and the timeslot schedule $\boldsymbol{T}$.

For practical reasons we state the optimal control problem in a discrete-time setting. In particular, we consider piecewise constant inputs with sampling time t_{s} and a multiple-shooting discretization of the dynamics (1). Rather than functions $x_i(t)$, $u_i(t)$ we therefore treat the state and control sequences $x_i = (x_{i,0}, \ldots, x_{i,N})$ and $u_i = (u_{i,0}, \ldots, u_{i,N-1})$, where $t_f = Nt_{\text{s}}$ is a fixed time horizon. The vehicle dynamics (1) and constraints (2), (4) are translated to

$$x_{i,0} = \hat{x}_{i,0} \tag{5}$$
$$x_{i,k+1} = F_i(x_{i,k}, u_{i,k}, t_{\text{s}}), \qquad k \in \mathbb{I}_{[0,N-1]}, \tag{6}$$
$$0 \geq h_i(x_{i,k}, u_{i,k}), \qquad k \in \mathbb{I}_{[0,N-1]}, \tag{7}$$
$$p_{j,k} + L_j/2 \leq p_{i,k} - L_i/2, \qquad k \in \mathbb{I}_{[0,N]}, \tag{8}$$

where $x_{i,k} = (p_{i,k}, z_{i,k})$, $\hat{x}_{i,0} = (\hat{p}_{i,0}, \hat{z}_{i,0})$ denotes the initial state of vehicle i and $F_i(x_{i,k}, u_{i,k}, t_{\text{s}})$ is the solution to (1) at $t = (k+1)t_{\text{s}}$, with $x_i(kt_{\text{s}}) = x_{i,k}$ and $u_i(t) = u_{i,k}$.

Note that, in the discrete time form, the state is $x_{i,k} = (p_{i,k}, z_{i,k})$, i.e., the position is only defined at integer multiples of t_{s}. Due to (3), $t^{\text{in}}_{i,r}$ and $t^{\text{out}}_{i,r}$ can thereby only attain values that are integer multiples of t_{s}, which restricts the set of possible timeslots $\boldsymbol{T}$ to a discrete set. To allow real-valued $\boldsymbol{T}$ with a discrete-time state and control sequence, we instead use $p^d_i(t, w_i) = [1, 0, \ldots]F_i(x_{i,k}, u_{i,k}, \delta t)$, $\delta t = t - k\Delta t$, $k = \lfloor t/\Delta t \rfloor$ and

$$p^d_i(t^{\text{in}}_{i,r}, w_i) - p^{\text{in}}_{i,r} = 0, \qquad r \in \mathcal{J}_i, \tag{9}$$
$$p^d_i(t^{\text{out}}_{i,r}, w_i) - p^{\text{out}}_{i,r} = 0, \qquad r \in \mathcal{J}_i, \tag{10}$$

where we have collected $w_i = (x_i, u_i)$. The position $p^d_i(t, w_i)$ is thereby defined for all t, whereby $\boldsymbol{T}$ can assume real values. Further details on the discretization of the position function can be found in [11]. With $W = (w_1, \ldots, w_{N_a})$ we state the optimal coordination problem as

$$\min_{\mathcal{S}, \boldsymbol{T}, W} \quad J(W) \tag{11a}$$
$$\text{s.t.} \quad (5), (6), (7), (9), (10), i \in \mathbb{I}_{[1,N_{\text{a}}]}, \tag{11b}$$
$$(8), \quad (i,j) \in \mathcal{R}_l, \quad \forall l \in \mathbb{I}_{[1,N_l]}, \tag{11c}$$
$$t^{\text{out}}_{s_{r,i}} \leq t^{\text{in}}_{s_{r,i+1}}, i \in \mathbb{I}_{[1,|\mathcal{I}_r|-1]}, r \in \mathbb{I}_{[1,N_{\text{CZ}}]}, \tag{11d}$$
$$S_r \in \text{perm}(\mathcal{I}_r), \quad \forall r \in \mathbb{I}_{[1,N_{\text{CZ}}]}, \tag{11e}$$

where $J(W) = \sum_{i=1}^{N_a} J_i(w_i)$ and

$$J_i(w_i) = V_i(x_{i,N}) + \sum_{k=0}^{N-1} \ell_i(x_{i,k}, u_{i,k}). \tag{12}$$

III. A Heuristic Solution Method

Problem (11) is a Mixed Integer Nonlinear Program (MINLP), with non-convex constraints and possibly non-convex objective. Since such problems are notoriously difficult, the solution to (11) can only be obtained in reasonable time on small problem instances. In this section we introduce a heuristic which can be used to find approximate solutions to (11) in reasonable time for relevant problem instances sizes. The main idea is to form the approximation in two stages: first, find a crossing order $\hat{\mathcal{S}}$ by solving a low dimensional Mixed Integer Quadratic Program (MIQP), thereafter find the optimal state and control trajectories *given* this order through the fixed-order coordination problem NLP

$$\min_{\mathcal{S},\boldsymbol{T},W} \quad J(W) \qquad \text{s.t.} \quad (11\text{b}) - (11\text{d}), \quad \mathcal{S} = \hat{\mathcal{S}}. \tag{13}$$

The use of the MIQP is motivated by the existence of efficient solvers for such problems and the primal decomposition of the optimal coordination problem first presented in [7]. In this decomposition, a coordination problem similar to (11) is equivalently rewritten as a time-slot scheduling MINLP in $\boldsymbol{T}$, coupled to vehicle specific optimal control problems in w_i. This formulation allows the combinatorial part to be solved in the $\boldsymbol{T}$ space, rather than in the $(\boldsymbol{T}, W)$ space of (11). We construct the MIQP as an approximation of the time-slot scheduling MINLP, and solve it to obtain an approximately optimal timeslot schedule $\boldsymbol{T}$ and thereby the order $\hat{\mathcal{S}}$. The time-slot scheduling MINlP and the approximating MIQP is detailed next.

A. Formulation of a time-slot scheduling MINLP

This decomposition proposed in [7] can be performed when all inter-vehicle couplings are formulated using the timeslots $\boldsymbol{T}$. However, in (11), the rear-end collision avoidance (8) introduce additional couplings between the position at every stage of neighboring vehicles on the same lane. Due to this, the decomposition technique from [7] cannot be used directly. The issue is resolved by introducing a relaxation of the rear end collision avoidance conditions, where (4) is replaced with

$$t^{\text{out}}_{i,r} \leq t^{\text{in}}_{j,r} \quad r \in \mathcal{J}_i, \quad (i,j) \in \mathcal{R}_l, \quad \forall l \in \mathbb{I}_{[1,N_l]}. \tag{14}$$

With this relaxation, rear-end collision avoidance is enforced within the intersection, but disregarded outside, and the following time-slot scheduling problem approximates (11):

$$\min_{\mathcal{S},\boldsymbol{T}} \quad \sum_{i=1}^{N_\text{a}} V_i(T_i) \tag{15a}$$

$$\text{s.t.} \quad T_i \in \text{domain}(V_i(T_i)), \qquad i \in \mathbb{I}_{[1,N_\text{a}]}, \tag{15b}$$

$$(11\text{d}), (11\text{e}), (14) \tag{15c}$$

where $V_i(T_i)$ is given by the vehicle OC problems

$$V_i(T_i) = \min_{w_i} \quad J_i(w_i) \text{ s.t. } (5), (6), (7), (9), (10). \tag{16}$$

The constraint set $\text{domain}(V_i(T_i))$ is the set of T_i for which (16) is feasible, and it was shown in [7], [10] that $T_i \in \text{domain}(V_i(T_i))$ can be expressed as inequalities $c_i(T_i) \leq 0$.

B. Formulation of a MIQP approximation

While the time-slot scheduling problem (15) is a MINLP, an MIQP approximation can be formulated using local approximations of the objective and constraints, similar to how the QP sub-problems are constructed in Sequential Quadratic Programming (SQP) [18]. We approximate the objective around $\boldsymbol{T}^{(0)} = (T_1^{(0)}, \ldots, T_{N_a}^{(0)})$ by

$$V(\boldsymbol{T}, \boldsymbol{T}^{(0)}) = \sum_{i=1}^{N_a} \frac{1}{2} T_i^\top \nabla^2 V_i T_i + (\nabla V_i - \nabla^2 V_i T_i^{(0)})^\top T_i, \tag{17}$$

where all derivatives are evaluated at $T_i^{(0)}$. The constraints are similarly approximated as

$$\hat{c}_i(T_i, T_i^{(0)}) = c_i(T_i^{(0)}) + \nabla c_i (T_i^{(0)} - T_i). \tag{18}$$

Using (17) and (18), the MIQP approximation to (15) is

$$\min_{\mathcal{S},\boldsymbol{T}} \quad V(\boldsymbol{T}, \boldsymbol{T}^{(0)}) \tag{19a}$$

$$\text{s.t.} \quad \hat{c}_i(T_i, T_i^{(0)}) \leq 0, \qquad i \in \mathbb{I}_{[1,N_\text{a}]}, \tag{19b}$$

$$(11\text{d}), (11\text{e}), (14), \tag{19c}$$

Constraint set properties: It is shown in [7] that the constraint set $c_i(T_i) \leq 0$ can be written as

$$c_{i,r}(T_i) = \begin{bmatrix} t^{\text{in}}_{i,r} - u^{\text{in}}_{i,r} \\ l^{\text{in}}_{i,r} - t^{\text{in}}_{i,r} \\ t^{\text{out}}_{i,r} - u^{\text{out}}_{i,r} \\ l^{\text{out}}_{i,r} - t^{\text{out}}_{i,r} \end{bmatrix} \leq 0, \ r \in \mathcal{J}_i, \tag{20}$$

where $l^{\text{in}}_{i,r}$ is given by

$$l^{\text{in}}_{i,r}(T_i) = \min_{w_i,t} \quad t \tag{21a}$$

$$\text{s.t.} \quad (5), (6), (7), (10), \tag{21b}$$

$$p_i^d(t, w_i) = p^{\text{in}}_{i,r}, \tag{21c}$$

$$p_i^d(t^{\text{out}}_{i,q}, w_i) = p^{\text{out}}_{i,q}, \quad q \in \mathcal{J}_i, \tag{21d}$$

$$p_i^d(t^{\text{in}}_{i,q}, w_i) = p^{\text{in}}_{i,q}, \quad q \in \mathcal{J}_i \setminus \{r\}, \tag{21e}$$

$u^{\text{in}}_{i,r}$ is given by

$$u^{\text{in}}_{i,q}(T_i) = \max_{w_i,t} \quad t \qquad \text{s.t.} \qquad (21\text{d}) - (21\text{e}) \tag{22}$$

and $l^{\text{out}}_{i,r}$,$u^{\text{out}}_{i,r}$ are defined similarly.

Computation of Derivatives: The constraints and objective of (15) are the optimal value functions of the parametric NLPs (16), (21) and (22). The derivative of the solution to an optimization problem with respect to a parameter can be obtained using parametric sensitivity analysis [5]. For NLP (16), it holds that

$$\nabla_{T_i} V_i(T_i) = \nabla_{T_i} P_i(T_i, w_i^*) \nu_i^*, \tag{23}$$

where T_i, $P_i(T_i, w_i^*) = 0$ collects constraints (9),(10) and ν_i^* are the corresponding optimal Lagrange multipliers and w_i^* is the optimal primal solution to (16) [1].

The second-order derivative $\nabla^2_{T_i} V_i(T_i)$ can be obtained by applying the chain rule to (23). Note that the resulting

Algorithm 1 Basic approximation algorithm.

1: $\forall i$: Solve NLP (16) w.o. constraints (9),(10) for minimizer $w_i^{(0)}$. Obtain $T_i^{(0)}$ from $w_i^{(0)}$ through (9),(10).
2: $\forall i$: Solve NLP (16) with $T_i^{(i)}$ for $\nabla_{T_i} V_i(T_i^{(i)})$and $\nabla^2_{T_i} V_i(T_i^{(i)})$
3: $\forall i, \forall r \in \mathcal{J}_i$ solve NLPs (21),(22) with $T_i^{(i)}$ for $c_{i,r}(T_i)$ and $\nabla_{T_i} c_{i,r}(T_i)$.
4: Assemble and solve MIQP (19) for the order $\hat{\mathcal{S}}$.
5: Solve NLP (13) using $\hat{\mathcal{S}}$ for $\boldsymbol{T}^*$ and W^*

expression will include the first-order sensitivities of the primal-dual solution to (16) w.r.t. T_i; $\frac{\mathrm{d}w_i^*}{\mathrm{d}T_i}$ and $\frac{\mathrm{d}\nu_i^*}{\mathrm{d}T_i}$. As discussed in [11], these are available at low additional cost when a second-order optimization method, e.g. a primal-dual interior point algorithm, is used to solve (16). An identical reasoning holds for NLPs (21) and (22).

C. An approximation algorithm

A simple two-stage procedure which constructs an approximate solution to MINLP (11) follows naturally: obtain first the order $\hat{\mathcal{S}}$ by solving MIQP (19), thereafter solve NLP (13) with $\hat{\mathcal{S}}$ to obtain the state- and control sequences. The algorithm is summarized in Algorithm 1. Note that here, the MIQP (19) is constructed using an expansion around the optimal solution for each vehicle where all inter-vehicle couplings have been removed.

Remark 1: Due to the use of approximations (17) and (18), MIQP (19) can be infeasible even though a feasible solution exists to (11). As a consequence, Algorithm 1 can return false negatives. On the other hand, by construction, the algorithm cannot give false positives since no solution will exist for (13), regardless of $\hat{\mathcal{S}}$, if no solution exists to NLP (11). Algorithm 1 can therefore be conservative.

Remark 2: While the relaxed statement (14) of the rear end collision avoidance constraints (4) is used to find $\hat{\mathcal{S}}$, (4) is used in the solution of (13). This has two consequences: 1) The MIQP can produce an order that is infeasible in (13) due to (4), causing Algorithm 1 to fail. 2) The state sequences W returned by Algorithm 1 will always be feasible in (11). We note that in a practical setting a control system could execute a fail-safe maneuver, e.g. come to a stop, if 1) occurs.

D. A simplified approximation algorithm

There are three computationally demanding parts of Algorithm 1: the solution of MIQP (19), the solutions of NLPs (16), (21),(22) and the solution of the NLP (13). While the MIQP is expected to dominate the time required by Algorithm 1 for problem instances above a certain size, the NLPs are expected to constitute the major burden for smaller instances.

Focusing on the NLPs of lines 2 and 3 of Algorithm 1, we note that 4 problems need be solved for each vehicle and each CZ due to (20) and one problem due to (16). While the problems are independent and could be solved in parallel, complete parallelization might not be possible for practical reasons. In the worst case, $\sum_{i=1}^{N_a} 1 + 4|\mathcal{J}_i|$ NLPs must be solved in sequence.

We propose to reduce the number of NLPs to solve by fixing the relationship between the times of entry and exit of all CZ for a vehicle. The purpose is to reduce the degrees of freedom in the MIQP in such a way that the constraint (20) can be removed. To this end, we consider only the time of entry for the first CZ of each vehicle, which we denote t_i^{in}, as a free variable, and define the remaining $t_{i,r}^{\mathrm{in}}$ and $t_{i,r}^{\mathrm{out}}$ to be the optimal times given t_i^{in}. More precisely, we let

$$V_i^R(t_i^{\mathrm{in}}) = \min_{x_i, u_i} \quad J_i(w_i) \tag{24a}$$
$$\text{s.t.} \quad (5), (6), (7), \tag{24b}$$
$$p_i^d(t_i^{\mathrm{in}}, w_i) - p_{i,r}^{\mathrm{in}} = 0, \tag{24c}$$

denote its minimizer $w_i^*(t_i^{\mathrm{in}})$, and let

$$d_{i,r}^{\mathrm{x}}(t_i^{\mathrm{in}}) = t \text{ s.t. } p_i^d(t, w_i^*(t_i^{\mathrm{in}})) - p_{i,r}^{\mathrm{x}} = 0.^1 \tag{25}$$

Moreover, we let $t_{i,r}^{\mathrm{in}} - d_{i,r}^{\mathrm{in}}(t_i^{\mathrm{in}}) = 0$ and $t_{i,r}^{\mathrm{out}} - d_{i,r}^{\mathrm{out}}(t_i^{\mathrm{in}}) = 0$ for each vehicle and each CZ, and collect all such constraints in $g_i(T_i) = 0$. Furthermore, it was shown in [7] that problem (24) has a solution whenever $t_i^{\mathrm{in}} \in [t_i^L, t_i^U]$, where t_i^L and t_i^U are defined by the solutions to NLPs (21) and (22) with constraints (21d) and (21d) removed. Since $g_i(T_i) = 0$ implies $c_i(T_i) \leq 0$ by construction, the solution to the following MINLP is a feasible (but possibly suboptimal) solution to MINLP (15)

$$\min_{\mathcal{S}, \boldsymbol{T}} \quad \sum_{i=1}^{N_{\mathrm{a}}} V_i^R(t_i^{\mathrm{in}}) \tag{26a}$$
$$\text{s.t.} \quad t_i^{\mathrm{in}} \in [t_i^L, t_i^U], \qquad i \in \mathbb{I}_{[1,N_{\mathrm{a}}]}, \tag{26b}$$
$$(11\text{d}), (11\text{e}), (14), \tag{26c}$$
$$g_i(T_i) = 0, \qquad i \in \mathbb{I}_{[1,N_{\mathrm{a}}]}, \tag{26d}$$

A simplified MIQP formulation: Similarly to Algorithm 1, we form an MIQP which approximates MINLP (26) as

$$\min_{\mathcal{S}, \boldsymbol{T}} \quad V_i^R(t_i^{\mathrm{in}}, t_i^{\mathrm{in}(0)}) \tag{27a}$$
$$\text{s.t.} \quad t_i^{\mathrm{in}} \in [t_i^L, t_i^U], \qquad i \in \mathbb{I}_{[1,N_{\mathrm{a}}]}, \tag{27b}$$
$$(11\text{d}), (11\text{e}), (14), \tag{27c}$$
$$d_i(T_i, t_i^{\mathrm{in}(0)}) = 0, \qquad i \in \mathbb{I}_{[1,N_{\mathrm{a}}]}, \tag{27d}$$

Here, V_i^R is the second-order Taylor expansion of $\sum_{i=1}^{N_a} V_i^R(t_i^{\mathrm{in}})$ around $t_i^{\mathrm{in}(0)}$ (c.f. (17)) and where $d_i(T_i, t_i^{\mathrm{in}(0)})$ is the linearization of $g_i(T_i)$ around $t_i^{\mathrm{in}(0)}$ (c.f. (18)). With a few modifications, the MIQP formulation (27) can thereafter be used in the approximation scheme detailed in Section III-C. Note since the constraint $h_i(T_i) \leq 0$ is absent from (26), the simplification only requires $[t_i^L, t_i^U]$ and the solution of (24) for each vehicle to construct the MIQP (27). In particular, we emphasize that the equality constraints $g_i(T_i) = 0$ are also defined through (24), and do not require the solution of separate optimization problems. The simplified procedure is summarized in Algorithm 2.

[1]Note that the assumption that the vehicle starts before the intersection ensures that t exist, and that $\dot{p}(t) > 0$ ensures that t is unique

Algorithm 2 Simplified approximation algorithm.

1: $\forall i$: Solve NLP (24) w.o. constraints (24c) for minimizer $w_i^{(0)}$. Obtain $t_i^{\mathrm{in}(0)}$ from $w_i^{(0)}$ through (9).
2: $\forall i$: Solve NLPs (21),(21) w.o. constraints (21d), (21d) for $[t_i^L, t_i^U]$
3: $\forall i$: Solve NLP (16) with $t_i^{\mathrm{in}(i)}$ for $\nabla_{t_i^{\mathrm{in}}} V_i$, $\nabla^2_{t_i^{\mathrm{in}}} V_i$, w_i^* and $\frac{\mathrm{d}w_i^*}{\mathrm{d}t_i^{\mathrm{in}}}$.
4: Assemble and solve MIQP (27) for the order $\hat{\mathcal{S}}$.
5: Solve NLP (13) using $\hat{\mathcal{S}}$ for $\boldsymbol{T}^*$ and W^*

Algorithm 3 A basic First-Come-First-Served algorithm.

1: $\forall i$: Solve NLP (16) w.o. constraints (9),(10) for minimizer $w_i^{(0)}$. Obtain $T_i^{(0)}$ from $w_i^{(0)}$ through (9),(10).
2: Set $\mathcal{I} = \{1, \dots, N_a\}$, set $\mathcal{S}_r = \emptyset$, $\forall r = 1, \dots, N_{\mathrm{CZ}}$.
3: **while** $\mathcal{I} \neq \emptyset$ **do**
4: Select $j \in \arg\min_{i \in \mathcal{I}} t_i^{\mathrm{in}}$
5: Add j as the last vehicle in $\mathcal{S}_r$, $\forall r \in \mathcal{J}_j$
6: $\mathcal{I} \leftarrow \mathcal{I} \setminus \{r\}$
7: **end while**
8: Assemble $\hat{\mathcal{S}}$ from $\mathcal{S}_r$, $\forall r = 1, \dots, N_{\mathrm{CZ}}$.
9: Solve NLP (13) using $\hat{\mathcal{S}}$ for $\boldsymbol{T}^*$ and W^*.

Derivatives of $d_i(T_i, t_i^{\mathrm{in}(0)})$: The linearization of $g_i(T_i)$ requires the evaluation of $\frac{\mathrm{d}t_{i,r}^x}{\mathrm{d}t_i^{\mathrm{in}}}$ at some point $t_i^{\mathrm{in}(0)}$ through $\nabla_{T_i} g_i(T_i)$. Noting that the parametric solution $w_i^*(t_i^{\mathrm{in}})$ to (24) must satisfy $\frac{\mathrm{d}}{\mathrm{d}t_i^{\mathrm{in}}} p_i^d(t, w_i^*(t_i^{\mathrm{in}})) = 0$, we have that

$$\frac{\mathrm{d}t_{i,r}^x}{\mathrm{d}t_i^{\mathrm{in}}} = -\left(\frac{\partial p_i^{\mathrm{d}}}{\partial t_{i,r}^x}\right)^{-1} \frac{\partial p_i^{\mathrm{d}}}{\partial w_i^*} \frac{\mathrm{d}w_i^*}{\mathrm{d}t_i^{\mathrm{in}}}, \tag{28}$$

where the arguments have been dropped for brevity. The expression contains the first-order sensitivity to parameter variations of the solution to w_i^*. As discussed above, these are readily available, provided that a second-order optimization method was used to solve (24). Consequently, $\nabla_{T_i} g_i(T_i)$ comes at a small additional expense when (24) is solved.

Remark 3: While the simplified algorithm reduced the degrees of freedom for each car from $2|\mathcal{J}_i|$ to 1 to reduce the computational load, intermediate formulations are possible. At the cost of re-introducing constraints like (20), the free variables could be any selection of the components of T_i, e.g., the in-time of the first CZ and the out-time of the last CZ, the in-times of all CZs etc. Each additional free variable will require the solution of variations of (21) and (22).

IV. Numerical Results

In this section we present results from the application of the proposed approximation schemes to scenarios with electric vehicles. We demonstrate the performance of the two proposed algorithms through example scenarios and a randomized simulation study. We compare Algorithms 1,2 and the simple FCFS policy summarized in Algorithm 3. In particular, to demonstrate the flexibility of the proposed scheme, we study the performance of Algorithms 1-3 for two different objective functions for scenarios involving vehicles of different types.

A. Simulation Setup

We consider the longitudinal dynamics of an electric vehicle (see, e.g. [6]), where $\dot{p}_i(t) = v_i(t)$ and

$$\dot{v}_i(t) = \frac{1}{m_i}\left(F_{d,i}(t) - F_{b,i}(t) - F_{a,i}(t) - F_{r,i}(t)\right). \tag{29}$$

Here, m_i is the vehicle mass, $F_{d,i}$ is the force resulting from the electric motor torque, $F_{b,i}$ is the retarding force from the friction brakes, $F_{a,i} = \frac{1}{2}\rho A_i C_{d,i} v_i^2(t)$ is the aerodynamic drag, $F_{r,i} = m_i g C_{r,i}$ is the rolling resistance, ρ is the air density, g the gravitational acceleration, A_i the projected frontal area of the vehicle, $C_{d,i}$ the aerodynamic drag coefficient and $C_{r,i}$ the rolling resistance coefficient. For simplicity, we assume that the transmission losses are negligible and that the propulsive force is proportional to the engine torque $T_{m,i}(t)$ through $F_{d,i}(t) = \frac{M_i}{r_{w,i}} T_{m,i}(t)$, where $r_{w,i}$ is the wheel radius and M_i is the transmission gear ratio. The electric motor is subject to the following limitations

$$0 \leq T_{m,i}(t) \leq \min(T_{m,i}^{\max}, P_i^{\max}/\omega_{m,i}(t)), \tag{30a}$$
$$0 \leq \omega_{m,i}(t) \leq \omega_{m,i}^{\max}, \tag{30b}$$

where $\omega_{m,i}(t) = \frac{M_i}{r_{w,i}} v_i(t)$ is the motor speed, $\omega_m^{\max}$ is the maximum motor speed, $T_i^{\max}$ the maximum torque and $P^{\max}$ is the maximum power that can be supplied continuously. Finally, we also use $0 \leq F_{b,i}(t) \leq F_{b,i}^{\max}$, where $F_{b,i}^{\max}$ is the maximum friction brake force. We thereby have $x_i(t) = (p_i(t), v_i(t))$ and $u_i(t) = (T_{m,i}(t), F_{b,i}(t))$.

Objective 1: Economic: The first objective function is taken from [12], and constitutes a trade-off between the minimization of energy-consumption and travel time delay. In particular, we use the stage cost

$$\ell_i(x_{i,k}, u_{i,k}) = \int_{k\Delta t}^{(k+1)\Delta t} P_i(x_i(t), u_{i,k}) - \alpha_i \frac{v_i(t)}{\Delta t} \mathrm{d}t, \tag{31}$$

and the terminal cost is $V_{f,i}(v_{i,N}) = \frac{1}{2}(v_{i,N} - v_i^{\mathrm{r}})^2 q_i + \beta_i(v_{i,N} - v_i^{\mathrm{r}})$. Here, $P_i(x_i(t), u_i(t)) = T_{m,i}\omega_{m,i}/\eta_i(T_{m,i}\omega_{m,i})$ is the power supplied to the motor and $\eta_i(T_{m,i}\omega_{m,i})$ is the motor efficiency map. We use the efficiency map of [17], where $\eta(T_m, \omega_m) = \frac{T_m \omega_m}{T_m \omega_m + P_{\mathrm{loss}}(T_m, \omega_m)}$, with $P_{\mathrm{loss}}(T_m, \omega_m) = c_0 + c_1\omega_m + c_2\omega_m T_m + c_3\omega_m^2$, and coefficients c_i obtained from a fit to empirical data. We choose the parameters α_i, q_i and β_i with the procedure proposed in [12], so that if unhindered, the vehicle converges to the specified speed v_i^{r}.

Objective 2: Tracking: The second objective is a "standard" OC objective, consisting of the sum of squared tracking errors:

$$\ell_i(x_{i,k}, u_{i,k}) = (v_{i,k} - v_i^{\mathrm{r}})^2 Q_{v,i} + (T_{m,i,k} - T_{m,i}^{\mathrm{r}})^2 R_{T,i} + F_{b,i,k}^2 R_{F,i} \tag{32}$$

where $Q_{v,i}, R_{T,i}, R_{F,i}$ are positive constants and $T_{m,i}^{\mathrm{r}}$ is the motor torque required to maintain the set speed v_i^{r}. This objective is in its form similar to that used in [7],[3],[13].

Parameters: We consider two different types of electric vehicles, one *"Light"* and one *"Heavy"*, where the differences in parameters are summarized in Table I. The remaining parameters are for both types set to $r_w = 0.32$ m, $C_{rr} = 0.015$, $\omega_{m,i}^{\max} = 10^4$ RPM, $v_i^{\mathrm{r}} = 70$ km/h and to ease visualization, the length of both vehicle types are $L_i = 4.8$ m.

	m	A	C_d	$P^{\max}$	$T^{\max}$	$F_b^{\max}$	M
Light	1500	2.3	0.32	80	250	10	7.9
Heavy	15000	4	0.7	400	800	40	15

TABLE I: Vehicle parameters. Units are kg, m^2, $(\cdot)$, kW, Nm, kN, $(\cdot)$, respectively.

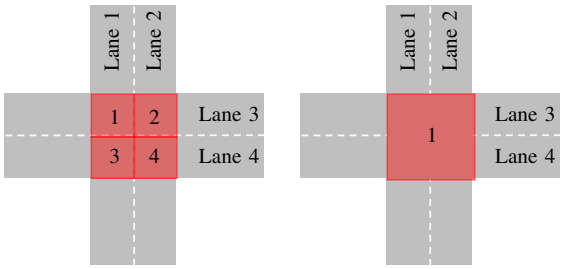
Finally, for the Economic Objective, we use the same motor efficiency map for both types, but scale it using $P_i^{\max}$ and $\omega_{m,i}^{\max}$, and for the Tracking Objective, we use $Q_v^L = (1/v_i^{\mathrm{r}})^2$,$R_T^L = (1/T_{m,i}^L)^2$, $R_F^L = (1/F_m^L)^2$ for the light vehicle and $Q_v^H = 100 Q_v^L$, $R_T^H = (10/T_{m,i}^H)^2$, $R_F^H = (10/F_m^H)^2$ for the heavy vehicle.

Simulator: We use Matlab's general purpose NLP solver fmincon to solve (13), (16), (21) and (22), and CPLEX to solve the MIQPs (19) and (27). For all cases, the dynamics are integrated using the Explicit Runge-Kutta-4 integrators (ERK4) from the ACADO toolkit [19], using the integration size $t_{\mathrm{s}} = 0.2$. Finally, we let the horizon be $N = 100$.

B. Simulation Results

Results from an example instance is given in Fig. 3 and Fig. 4. The scenario consists of 3 light vehicles on each lane in an intersection such as shown in Fig. 2a. The vehicles are generated randomly between -70 and -200 m, such that the inter-vehicle spacing is greater than 15 m, and all traveling at $\hat{v}_{i,0} = v^{\mathrm{r}} = 70$ km/h. The coordination is performed with the Economic Objective, and for this initial configuration, Algorithm 1 and Algorithm 2 gave the same results, while the result from Algorithm 3 (FCFS) differed significantly. Note in particular the large deviations in speed from v^{r} caused by the FCFS algorithm, shown in Fig. 3. The results demonstrate the ability of the proposed heuristics to take the objective function and the restrictions imposed by the vehicle dynamics and path constraints into account when selecting the crossing order. It also shows the large impact the crossing order can have on the resulting trajectories.

Heterogeneous scenarios: The approximation algorithms' ability to account for heterogeneity among the involved vehicles is illustrated in Fig. 5 and Fig. 6. The figures show results from a scenario with one car per lane and a single CZ, as shown in Fig. 2b, where the vehicles start at positions $\hat{p}_{1,0} = -150$ m, $\hat{p}_{2,0} = -155$ m, $\hat{p}_{3,0} = -160$ m and $\hat{p}_{4,0} = -165$ m, and all with $\hat{v}_{i,0} = v^{\mathrm{r}} = 70$ km/h. In particular, it both shows the solution when all vehicles are of the light type, and the solution when Vehicle 4 is of type heavy. With all light vehicles, Algorithms 1-3 give the same solution, having the crossing order $(1, 2, 3, 4)$. However, with Vehicle 4 heavy, Algorithms 1 and 2 returns crossing order $(1, 2, 4, 3)$. The choice of order reflects that for the given



(a) Multiple CZ. (b) Single CZ.

Fig. 2: Lane and CZ numbering

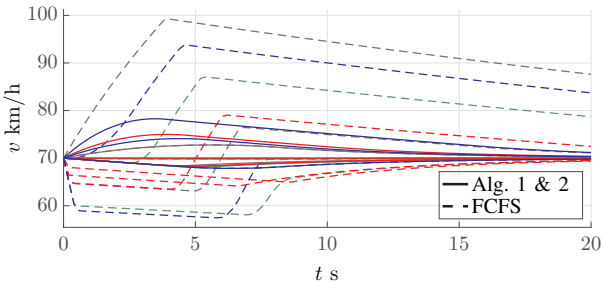


Fig. 3: Velocities corresponding to the position trajectories in Fig. 4.

objective and scenario configuration, it incurs a lower cost to slow down the light Vehicle 3 and let it pass after the heavy Vehicle 4, even though the former starts closer to the intersection than the latter. The corresponding velocity profiles are given in Fig. 6, where it is clear that virtually all control effort is spent on the light vehicles when the heavy vehicle is present.

Randomized evaluation: In the example of Fig. 3, the approximation gives an objective function value which is 68 % of the one obtained through the use of the FCFS policy. To examine the consistency of this behavior, we performed an evaluation over a larger number of scenarios. In particular, we considered 100 randomly generated 12 vehicle scenarios like that shown in Fig. 4, for each of the cases when $0, 1, 2, 3, 4, 5$ and 6 of the 12 vehicles were heavy. We solved each scenario using Algorithms 1-3, for both the Economic and Tracking objectives. The results for the Economic objective are given in Fig. 7 which shows the average value of $r_{1,K} = (J_K^* - J_U^*)/J_U^*$, where J_K^* is the optimal value given by Algorithm K. This is compared to J_U^*, which is the optimal cost of crossing intersection when keeping constant speed and disregarding collision avoidance, i.e., the minimum cost for all vehicles to cross the intersection. From Fig. 7, it is clear that both Algorithms 1 and 2 are superior to the FCFS heuristic regardless the distribution of heavy and light vehicles, with a tendency for larger improvements with more heterogeneity As discussed above, this is due to the fact that the proposed algorithms use (approximate) representation of the objective and the restrictions imposed by the vehicle dynamics when generating the crossing order.

The corresponding results for the Tracking objective is given in Fig. 8. Since the constant speed cost in this case is 0, the average total cost incurred, $r_{2,K} = J_K^*$ for Algorithm K,

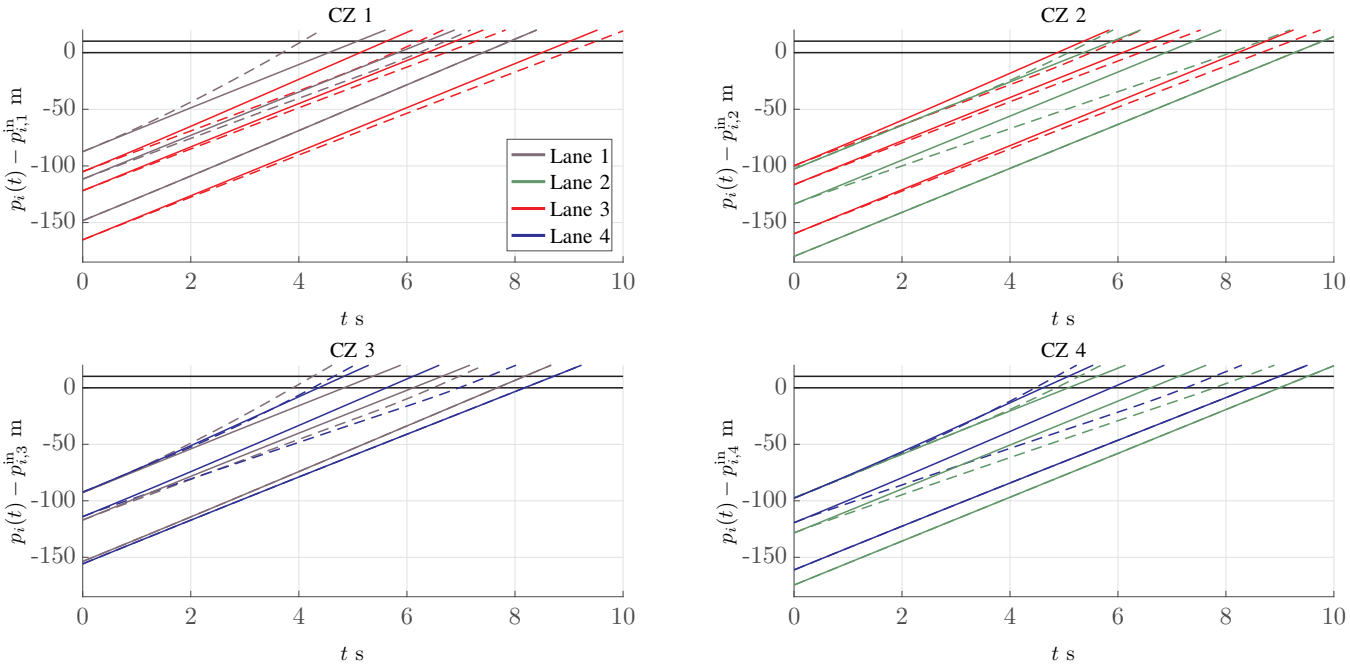


Fig. 4: Position trajectories of the scenario discussed in Section IV-B, with layout as in Fig. 2a. The horizontal lines demarcates the beginning and end for each CZ, and the position trajectories in each subfigure are shifted so the CZ is entered when $p_i(t) = 0$. The colors differentiate lanes, with solid lines from Algorithm 1 and dashed lines from Algorithm 3.

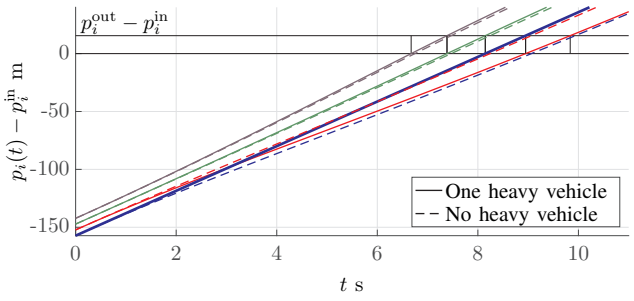


Fig. 5: Position trajectories from the output of Algorithm 1 for a scenario with layout as in Fig. 2b. The lanes are differentiated by color and the trajectory of the heavy vehicle is drawn drawn in bold.

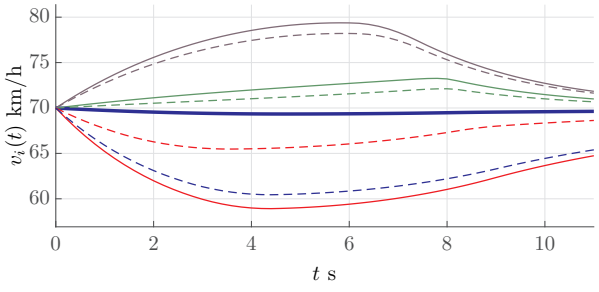


Fig. 6: Velocity trajectories corresponding to Fig. 5. In the case with a heay vehicle (solid lines), the speed of the heavy vehicle (thick line) is kept almost constant and the lighter vehicles adjust their speeds to avoid collisions.

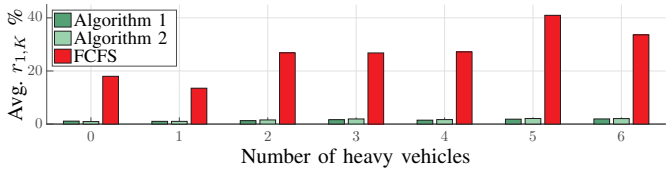


Fig. 7: Average cost increase in the Economic objective case.

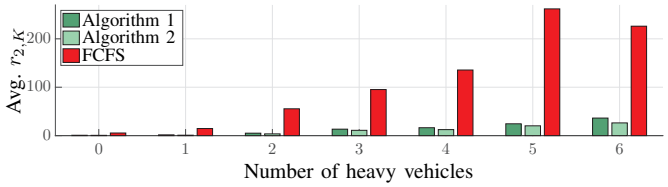


Fig. 8: Average cost in the Tracking objective case.

is shown. As can be seen in the figure, the results are similar to those for the Economic objective, with both the proposed Algorithms outperforming the FCFS benchmark.

Finally, we note that the results shown in Fig. 7 and Fig. 8 also indicate that the performance of Algorithms 1 and 2 are very similar for both objective functions. For the economic objective, the average over all simulated cases is $r_{1,1} = 1.48\%$ and $r_{1,2} = 1.62\%$.

Remark 4: Finally, we remark that the MIQP in this case had 48 continuous and 36 binary variables, with solve times in the range of hundreds of a second with CPLEX. The construction of the objective function and constraints for the MIQP required around 30 s for Algorithm 1 and 6 s for Algorithm (3), while NLP (13) required in the range of 10-15 s. All times are reported by Matlab on a standard laptop.

V. Discussion and Conclusion

In this paper we have presented two optimal control based algorithms for coordination of automated vehicles at intersection, which relies on MIQP-based heuristics to compute the intersection crossing order. The algorithms were applied to realistic scenarios involving nonlinear vehicle models, path constraints and objective functions. We demonstrated that the MIQP heuristic approximately account for the specified performance objective and the restrictions imposed by the vehicle dynamics. This enabled superior performance compared to a competing heuristic for two different objective functions and scenarios including vehicles of different types, which indicates that the performance of the proposed schemes is not coupled tightly to specific vehicle dynamics, scenario compositions nor objective functions.

While the times reported in Section IV-B required to build the MIQP data and solve NLP (13) are too large for real time applications, we emphasize that these are clocked on a Matlab implementation. Efficient implementations using structure exploiting NLP solvers and parallelization could likely reduce this with several orders of magnitude. More importantly, we note that the performance of Algorithm 2 is very close to that of Algorithm 1, and that the MIQP data in Algorithm 2 can be constructed much faster, as it involves solving 3 rather than $1 + 4|\mathcal{J}_i|$ NLPs per vehicle. In fact, for most systems, the upper and lower bounds $[t_i^L, t_i^U]$ can be obtained from a forward simulation of the dynamics with maximal accelerating and decelerating input, respectively, rather than solving NLPs (21),(22). In that case the MIQP data can be constructed by solving the NLP (16) for each vehicle. Both Algorithms 1 and 2 could in a practical setting be executed in a semi-distributed fashion: The MIQP data is computed onboard each vehicle and communicated to a central entity, which thereafter assembles and solves MIQP (27) for the crossing order. The central node thereafter initiates the semi-distributed SQP of [11],[21] or the algorithm of [13] to solve (13).

Future work includes large-scale evaluation and further generalizations to the problem formulation, e.g. including vehicles that make turns in the intersection.

References

[1] C. Büskens and H. Maurer. *Online Optimization of Large Scale Systems*, chapter Sensitivity Analysis and Real-Time Optimization of Parametric Nonlinear Programming Problems, pages 3–16. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001.

[2] L. Chen and C. Englund. Cooperative intersection management: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 17(2):570–586, Feb 2016.

[3] G. R. de Campos, P. Falcone, H. Wymeersch, R. Hult, and J. Sjöberg. Cooperative receding horizon conflict resolution at traffic intersections. In *53rd CDC*, pages 2932–2937, Dec 2014.

[4] K. Dresner and P. Stone. A Multiagent Approach to Autonomous Intersection Management. *Journal of Artificial Intelligence Research*, 31(1):591–656, March 2008.

[5] A.V. Fiacco. *Introduction to sensitivity and stability analysis in nonlinear programming*. Academic Press, New York, 1983.

[6] L. Guzzella and A. Sciarretta. *Vehicle Propulsion Systems: Introduction to Modeling and Optimization*. 2005.

[7] R. Hult, G. R. de Campos, P. Falcone, and H. Wymeersch. An approximate solution to the optimal coordination problem for autonomous vehicles at intersections. In *American Control Conference (ACC), 2015*, pages 763–768, July 2015.

[8] R. Hult, G. R. de Campos, E. Steinmetz, L. Hammarstrand, P. Falcone, and H. Wymeersch. Coordination of cooperative autonomous vehicles: Toward safer and more efficient road transportation. *IEEE Signal Processing Magazine*, 33(6):74–84, Nov 2016.

[9] R. Hult, F. E. Sancar, M. Jalalmaab, A. Vijayan, A. Severinson, M. Di Vaio, P. Falcone, B. Fidan, and S. Santini. Design and Experimental Validation of a Cooperative Driving Control Architecture for the Grand Cooperative Driving Challenge 2016. *IEEE Transactions on Intelligent Transportation Systems*, PP(99):1–12, 2018.

[10] R. Hult, M. Zanon, S. Gros, and P. Falcone. Optimal coordination of three cars approaching an intersection. `https://youtu.be/nYSXvnaNRK4`. Accessed: 2017-03-03.

[11] R. Hult, M. Zanon, S. Gros, and P. Falcone. Primal Decomposition of the Optimal Coordination of Vehicles at Traffic Intersections. In *Proceedings of the Conference on Decision and Control*, 2016.

[12] R. Hult, M. Zanon, S. Gros, and P. Falcone. Energy-Optimal Coordination of Autonomous Vehicles at Intersections. In *Proceedings of the European Control Conference*, 2018.

[13] Y. Jiang, M. Zanon, R. Hult, and B. Houska. Distributed algorithm for optimal vehicle coordination at traffic intersections. In *Proceedings of the 20th IFAC World Congress, France*, pages 12082–12087, 2017.

[14] A. Katriniok, P. Kleibaum, and M. Joševski. Distributed Model Predictive Control for Intersection Automation Using a Parallelized Optimization Approach. In *IFAC World Congress 2017*. IFAC, 2017.

[15] K. D. Kim and P. R. Kumar. An mpc-based approach to provable system-wide safety and liveness of autonomous ground traffic. *IEEE Transactions on Automatic Control*, 59(12):3341–3356, Dec 2014.

[16] N. Murgovski, G. R. de Campos, and J. Sjöberg. Convex modeling of conflict resolution at traffic intersections. pages 4708–4713, Dec 2015.

[17] N. Murgovski, L. M. Johannesson, and B. Egardt. Optimal Battery Dimensioning and Control of a CVT PHEV Powertrain. *IEEE Transactions on Vehicular Technology*, 63(5):2151–2161, Jun 2014.

[18] J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer, 2 edition, 2006.

[19] R. Quirynen, M. Vukov, M. Zanon, and M. Diehl. Autogenerating Microsecond Solvers for Nonlinear MPC: a Tutorial Using ACADO Integrators. *Optimal Control Applications and Methods*, 36:685–704, 2014.

[20] J. Rios-Torres and A. A. Malikopoulos. A survey on the coordination of connected and automated vehicles at intersections and merging at highway on-ramps. *IEEE Transactions on Intelligent Transportation Systems*, 18(5):1066–1077, May 2017.

[21] M. Zanon, S. Gros, H. Wymeersch, and P. Falcone. An Asynchronous Algorithm for Optimal Vehicle Coordination at Traffic Intersections. In *20th IFAC World Congress*, 2017.

[22] Y. J. Zhang, A. A. Malikopoulos, and C. G. Cassandras. Optimal control and coordination of connected and automated vehicles at urban traffic intersections. In *2016 American Control Conference (ACC)*.