2018 European Control Conference (ECC)
June 12-15, 2018. Limassol, Cyprus

Experimental Validation of Distributed Optimal Vehicle Coordination

Mario Zanon, Robert Hult, Sébastien Gros and Paolo Falcone

***Abstract*— In this paper we solve the problem of coordinating autonomous vehicles approaching an intersection in experiments. We cast the problem in the distributed optimisation framework and use the algorithm proposed in [10], [15] to solve it in real time. We compare two variants of the algorithm in simulations and test our algorithm in experiments using real cars on a test track. The experimental results demonstrate the applicability and real-time feasibility of the algorithm and show that the underlying assumptions are justified.**

I. Introduction

Autonomous driving is mostly motivated by: (a) aiming for a zero-accident era, (b) reducing pollution and energy consumption and (c) increasing the capacity of the infrastructure. These objectives can arguably be attained using cooperative control, as recently studied in e.g. [3], [5], [12]. In this paper, we focus on intersection crossing scenarios.

In order to compute a coordinated policy, communication between the vehicles becomes necessary. The problem can then be framed as a mixed-integer optimal control problem. While computing the global optimum is NP-hard [4], the development of tailored algorithms will make it possible to compute approximated solutions in real time. An approach for finding a feasible solution was proposed and tested in [2] where, however, optimality was not considered. In this paper, we assume that a prescribed crossing order is computed by e.g. a heuristic, and focus on the solution of the distributed optimal control problem. Our algorithm (a) minimises the cost for a given crossing order and (b) is an essential building block towards algorithms optimising the crossing order.

We implement and test for the first time in experiments the optimisation framework based on the crossing times of each agent, proposed in [10], [15]. As opposed to standard optimisation problems, feasibility of the crossing times for each agent is required at every iterate of the algorithm (see Section III-D) while the precedence constraint (enforcing collision avoidance) is satisfied only upon convergence. We enforce feasibility by implementing both the projection proposed in [15] and a constraint-relaxation approach. We compare the two in simulations to validate the intuition that the first approach is superior. Finally, we test the algorithm in experiments with three vehicles.

This paper is structured as follows. In Section II we introduce the formulation of the coordination problem. In Section III we present the implemented distributed algorithm.

This work was supported by Vinnova (grants: 2015-04849 Copplar, and 2015-03075 AstaZero program), the Swedish Research Council (VR, grant 2012-4038) and the EU 7th Framework (grant 610428 AdaptIVe project). The authors are with the Department of Electrical Engineering, Chalmers University of Technology, Göteborg, Sweden. e-mail: {name.surname@chalmers.se}

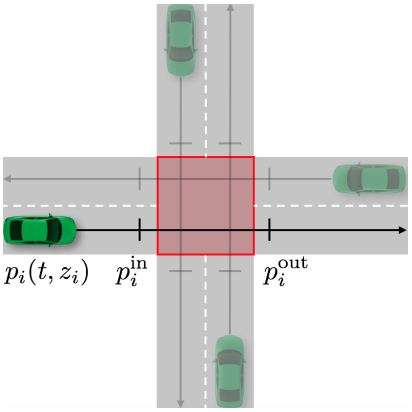

Fig. 1. Illustration of the considered intersection scenario.

In Section IV we discuss the simulation and experimental results. We present the concluding remarks in Section V.

II. Problem Formulation

We consider N_a vehicles approaching an intersection, as illustrated in Figure 1. For each vehicle i, we define the intersection start and end positions p_i^{in}, p_i^{out}, and the in- and out-times, i.e. the times at which the vehicle enters and exits the intersection, as t_i^{in} and t_i^{out}. We lump them together in vectors $t_i := (t_i^{\text{in}}, t_i^{\text{out}})$ and define $t := (t_1, \dots, t_{N_\text{a}})$.

The cost $V_i(t_i)$ associated with each vehicle is defined by a Model Predictive Control (MPC) problem. We define the states and controls of vehicle i at time k as $x_{i,k}$ and $u_{i,k}$ respectively. Moreover, we define vector $z_i := (z_{i,0}, \dots, z_{i,N-1}, x_{i,N})$ with $z_{i,k} := (x_{i,k}, u_{i,k})$ and the prediction horizon $N \in \mathbb{N}$. For simplicity, we assume linear dynamics and affine path constraints. Finally, we denote the set of all integers in a given interval as $\mathbb{I}_{[a,b]} = \{a, a+1, \dots, b\}$. The MPC problem of vehicle i is then given by

$$
\begin{aligned}
V_i(t_i) := \\
\min_{z_i} \quad & J_i(z_i) && \text{(1a)}\\
\text{s.t.} \quad & x_{i,0} = \hat{x}_{i,0} && \text{(1b)}\\
& x_{i,k+1} = A_i x_{i,k} + B_i u_{i,k}, \quad k \in \mathbb{I}_{[0,N-1]}, && \text{(1c)}\\
& D_{i,k} x_{i,k} + E_{i,k} u_{i,k} \leq e_{i,k}, \quad k \in \mathbb{I}_{[0,N-1]}, && \text{(1d)}\\
& p_i(t_i^{\text{in}}, z_i) - p_i^{\text{in}} = 0, && \text{(1e)}\\
& p_i(t_i^{\text{out}}, z_i) - p_i^{\text{out}} = 0, && \text{(1f)}
\end{aligned}
$$

where (1b) imposes the initial state, (1c) are the system dynamics, (1d) are path constratints including e.g. actuator limitations, and (1e)-(1f) make use of function p_i describing the position of vehicle i on its path along the road, in order

to force the vehicle to enter and exit the intersection at the prescribed times, see [10]. The vehicle-specific cost $J_i(z_i)$ is assumed to be quadratic in this paper. The use of different costs and a justification for the use of a quadratic cost is provided in [11]. Because both the positions p_i^{in} p_i^{out} and the times t_i are fixed, Problem (1) is a Quadratic Program (QP). The set of feasible in- and out-times is then the domain of the value function V_i of MPC Problem (1), i.e.

$$\mathcal{T}_i := \text{dom}(V_i(t_i)). \tag{2}$$

The coordination problem can then be formulated as

$$\min_{t} \quad \sum_{i=1}^{N_\text{a}} V_i(t_i) \tag{3a}$$

$$\text{s.t.} \quad t_i \in \mathcal{T}_i, \qquad i \in \mathbb{I}_{[1,N_\text{a}]}, \tag{3b}$$

$$t_{i-1}^{\text{out}} \leq t_i^{\text{in}}, \qquad i \in \mathbb{I}_{[2,N_\text{a}]}, \tag{3c}$$

such that the sum of each vehicle's individual cost is minimised, subject to having at most one vehicle in the intersection at any time (3c). Note that, because constraints (1e)-(1f) are generally nonlinear in t_i, Problem (3) is nonconvex. Since we solve the coordination problem (3) only once, the dependence of the cost on the initial state of each vehicle $\hat{x}_{i,0}$ is not explicitly stated.

We assume that the technical assumption [10, Assumption 1] holds (essentially consisting in the notion of the existence of a maximum and minimum control) and $p_i(t, z_i)$ is monotonically increasing in t and differentiable. Then, set $\mathcal{T}_i$ can be described by two bounds $t_i^{\text{in,lb}} \leq t_i^{\text{in}} \leq t_i^{\text{in,ub}}$, and two nonlinear constraints $t_i^{\text{out,lb}}\left(t_i^{\text{in}}\right) \leq t_i^{\text{out}} \leq t_i^{\text{out,ub}}\left(t_i^{\text{in}}\right)$. These quantities can be defined implicitly by the optimal trajectories associated with the linear programs (LPs):

$$(\max)\min_{z_i} \; x_N \text{ s.t. (1b)} - \text{(1d)}, \quad \text{for} \quad t_i^{\text{in,ub}},\ t_i^{\text{in,lb}}, \tag{4a}$$

$$(\max)\min_{z_i} \; x_N \text{ s.t. (1b)} - \text{(1e)}, \quad \text{for} \quad t_i^{\text{out,ub}},\ t_i^{\text{out,lb}}. \tag{4b}$$

For all details concerning this formulation we refer to [10].

III. Numerical Solution of the NLP

In this section, we present an adaptation of sequential quadratic programming (SQP) in a distributed setting for solving Problem (3). We remark that, by definition, both V_i, $t_i^{\text{out,ub}}$, $t_i^{\text{out,lb}}$ are non-smooth functions. A thorough discussion on the continuity and differentiability properties of V_i, $t_i^{\text{out,ub}}$, $t_i^{\text{out,lb}}$ has been presented in [10]. Note however that, provided that an interior-point QP solver is used in the local problems (1) and (4), one obtains a smooth approximation of the non-smooth functions V_i, $t_i^{\text{out,ub}}$, $t_i^{\text{out,lb}}$. Unfortunately, slow convergence can in principle not be excluded, as the smoothly approximated functions become highly nonlinear at the points of non-smoothness of the original problem.

The setup described in Section II perfectly fits the primal decomposition framework, which has the advantage of quickly finding a feasible solution and subsequently improving optimality. This is very desirable in view of robustness versus lossy communication channels, since by reducing the communication needs we increase the rate of packets which can be lost while still delivering the data rate necessary for feasibility (optimality). The pseudo-code of the algorithm's computations and data exchanges can be found in [15].

A. Sequential Quadratic Programming

For notational simplicity, we rewrite NLP (3) as

$$\min_{t} \quad f(t) \qquad \text{s.t.} \quad h(t) \geq 0, \tag{5}$$

where $f(t) = \sum_{i=1}^{N_\text{a}} V_i(t_i)$ and we lump Constraints (3b)-(3c) in function h. We define the associated Lagrangian as $\mathcal{L}(t, \mu) := f(t) - \mu^\top h(t)$. Starting from an initial guess $v^{(0)} = (t^{(0)},\ \mu^{(0)})$, SQP iteratively computes $v^{(j)}$ using

$$v^{(j+1)} = v^{(j)} + \alpha^{(j)} \Delta v^{(j)}, \tag{6}$$

with $\alpha^{(j)} \in (0, 1]$ and $\Delta v^{(j)} = (\Delta t^{(j)},\ \tilde{\mu}^{(j)} - \mu^{(j)})$, obtained as the primal-dual solution $(\Delta t^{(j)},\ \tilde{\mu}^{(j)})$ of the quadratic programming (QP) subproblem

$$\min_{\Delta t} \quad \frac{1}{2}\Delta t^\top H^{(j)} \Delta t + \nabla f(t^{(j)})^\top \Delta t \tag{7a}$$

$$\text{s.t.} \quad h(t^{(j)}) + \nabla h(t^{(j)})^\top \Delta t \geq 0. \tag{7b}$$

The iterations are stopped when the KKT residual r satisfies

$$r^{(j)} := \left\| \begin{array}{c} \nabla f(t^{(j)}) - \nabla h(t^{(j)})\mu^{(j)} \\ \min(0, h(t^{(j)})) \end{array} \right\|_\infty \leq \epsilon. \tag{8}$$

Variants of SQP differ in the computations of $\alpha^{(j)}$ and the so-called Hessian approximation $H^{(j)}$. If exact Hessian is used, i.e. $H^{(j)} = \nabla^2_{tt}\mathcal{L}(t^{(j)}, \mu^{(j)})$, the KKT conditions of each QP subproblem (7) coincide with a special form of linearisation of the KKT conditions of NLP (5), evaluated at $t^{(j)}$. For more details on SQP, we refer to e.g. [13].

The QP matrices $\nabla f(t^{(j)})$, $\nabla h(v^{(j)})$ and the Hessian of the Lagrangian $\nabla^2_{tt}\mathcal{L}(t^{(j)}, \mu^{(j)})$ are called first and second-order sensitivities respectively, whose computation is presented in detail in [10]. Here we recall that the evaluations of f, h require the solution of 2 LPs and 1 QP, while the first and second order sensitivities are obtained at a marginal additional cost with respect to the function evaluations.

While our formulation seems rather standard, we stress that in our distributed SQP problem the cost function and most constraints are separable and related with the solution of linear MPC problems that are local to each agent. The only source of coupling are the precedence constraints (3c). Moreover, the use of the in- and out-times largely reduces the size of the central problem and, therefore, the amount of information that needs to be communicated: at every iterate each agent needs to communicate f, h and the relative first and second order sensitivities, i.e. 13 doubles. Finally, our algorithm can be viewed as a variant of SQP and it therefore inherits the standard convergence properties of SQP.

B. Hessian Regularisation

We enforce positive-definiteness of the reduced Hessian by applying the simple strategy of removing all directions of negative curvature present in the Hessian by adding the minimal regularisation needed. We do this by exploiting the

block-diagonal structure of the Hessian to perform an eigenvalue decomposition of each 2-by-2 block and saturate all eigenvalues to a predefined minimum positive value. Given the small size of the blocks, the eigenvalue decomposition can be made very fast. Note that, since functions V_i, $t_i^{\mathrm{out,ub}}$, and $t_i^{\mathrm{out,lb}}$ are smooth (due to the use of an interior-point QP solver) the Hessian is well defined at any feasible point.

C. Globalisation and Merit Function

In order to guarantee global convergence of the algorithm, we implement a linesearch based on the so-called ℓ_1 merit function, defined as $M(t) = f(t)+\sigma\|\min(h(t),0)\|_1$, where σ is a parameter which must be chosen such that $\sigma > \|\mu\|_\infty$.

The first-order term of the Taylor expansion of M in the SQP direction y can be computed as [13]:

$$\mathrm{D}M(t)[y]y = \nabla f(t)y - \sigma\|\min(h(t),0)\|_1, \qquad (9)$$

where we define the directional derivative of M in direction y evaluated at t as $\mathrm{D}M(t)[y]$.

Once the QP solution $\Delta v^{(j)}$ is obtained, linesearch globalisation techniques choose the step size so as to enforce a decrease in the merit function based on the Armijo condition

$$M(t+\alpha\Delta t) \le M(t) + \gamma\mathrm{D}M(t)[\Delta t]\alpha\Delta t, \qquad (10)$$

where $\gamma \in (0, 0.5]$ is a fixed constant. Backtracking linesearch starts with $\alpha = 1$ and iteratively reduces it using $\alpha \leftarrow \beta\alpha$ with $\beta \in (0,1)$ until condition (10) is satisfied.

D. Local Feasibility Issues

NLP solvers typically only guarantee feasibility of the constraints at convergence and not throughout the iterations. Because in the case of Problem (3), the cost terms $V_i(t_i)$ are not defined for infeasible times $t_i \notin \mathcal{T}_i$, local feasibility must be enforced at every iterate.

We compare two methods based on: (a) introducing soft constraints using slack variables and penalising the ℓ_1-norm of the constraint violation with a sufficiently high weight and (b) using a feasibility-restoring projection of each iterate, as proposed in [15]. In the following we will call approach (a) *relaxation* and approach (b) *projection*. In [15] the use of projection is preferred to the use of relaxation. However, this choice was not motivated formally and no comparison between the two approaches was made.

While appealing for its simplicity, approach (a) suffers from one main drawback: the local quadratic approximation evaluated at an infeasible point is dominated by the cost associated with the constraint violation and looses its validity once the problem becomes feasible. This approach can be implemented by replacing (1f) in each agent's problem by

$$p_i(t_i^{\mathrm{out}}, z_i) - p_i^{\mathrm{out}} + s_1 - s_2 = 0, \quad s_1 \ge 0, \quad s_2 \ge 0. \quad (11)$$

Propositon 1: Reformulate constraints (1f) as (11) and add the term $\rho s_1 + \rho s_2$ to the cost of each local Problem (1). Choose $\rho > \lambda_{\max}$, i.e. the largest Lagrange multiplier associated with any constraint of the local Problem (1) for all feasible in-out times t_i. Then, the SQP algorithm with linesearch will converge.

Proof: The condition $\rho > \lambda_{\max}$ is necessary in order to make sure that the relaxed problem yields the solution to the original problem whenever the latter has a feasible solution [7, Theorem 14.3.1]. The relaxation of the constraints ensures that the local problems are feasible for any in- out-time pair and linesearch ensures a decrease in the merit function. Therefore, for all feasible t, the relaxed formulation coincides with the original one, while for all infeasible t, linesearch ensures that progress is made towards both feasibility and optimality. ■

In order to efficiently implement the projection approach (b), we observe that (i) the constraints (3b) are simple bounds on the in times t_i^{in} which are always satisfied, and (ii) the constraints on the out times are given by $t_i^{\mathrm{out,lb}}\left(t_i^{\mathrm{in}}\right) \le t_i^{\mathrm{out}} \le t_i^{\mathrm{out,ub}}\left(t_i^{\mathrm{in}}\right)$. Therefore, the in times t_i^{in} are always feasible and all out times t_i^{out} which become infeasible at iterate k can be projected back onto the set of feasible times:

$$\mathcal{P}_i\left(t_i^{\mathrm{out}(j)}\right) := \left[t_i^{\mathrm{out}(j)}\right]_{t_i^{\mathrm{out,lb}}\left(t_i^{\mathrm{in}(j)}\right)}^{t_i^{\mathrm{out,ub}}\left(t_i^{\mathrm{in}(j)}\right)},$$

with $[a]_b^c := \max(\min(a,c),b)$. The projection can be done easily without extra computations, as the evaluation of $t_i^{\mathrm{out,ub}}\left(t_i^{\mathrm{in}(j)}\right)$, $t_i^{\mathrm{out,lb}}\left(t_i^{\mathrm{in}(j)}\right)$ is needed for each iterate $j+1$ of the SQP algorithm. An illustration of the projection procedure is given in [15]. The main drawback of this approach is the partial loss of parallelisability at the agent level, i.e. the two LPs for the computation of the linearisation of constraint (3b) can be solved in parallel, but must be solved before QP (1). In the relaxation approach instead, the LPs and the QP can all be solved in parallel.

While the projection is simple to perform, the linesearch procedure has to be modified in order to guarantee that the step is a descent direction. We therefore define a modification of the Armijo condition (10) using the projection as follows

$$M(\mathcal{P}(t+\alpha\Delta t)) \le M(t) + \gamma\mathrm{D}M(t)[\Delta t]\alpha\Delta t, \qquad (12)$$

with Δt computed by (7). As opposed to standard linesearch, in (12) we do not backtrack from the projected Newton step. We use the non-projected Newton step to obtain the step candidate and project each step candidate.

IV. Results

In order to test the algorithm, we have considered identical vehicles defined by the linear system

$$\dot{x} = A_{\mathrm{c}}x + B_{\mathrm{c}}u, \quad A_{\mathrm{c}} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad B_{\mathrm{c}} = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

where $u \in [u^{\mathrm{lb}}, u^{\mathrm{ub}}]$ $\mathrm{m/s^2}$ is the (scalar) control and $x = (p, v)$ is the state vector, which includes position p and velocity $v \ge 0$. We discretise the system using the zero-order hold method to obtain

$$x_{k+1} = Ax_k + Bu_k, \quad A = \begin{bmatrix} 1 & T_{\mathrm{s}} \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 0.5\,T_{\mathrm{s}}^2 \\ T_{\mathrm{s}} \end{bmatrix},$$

where T_{s} is the sampling time. Note that the technical assumption [10, Assumption 1] holds for this system. Each

TABLE I

SIMULATION PARAMETERS (SI UNITS)

Vehicle #	1	2	3	4	5	6
Q_i	1	1	10	10	1	1
R_i	1	1	1	1	1	1
v_i^{d}	80	80	65	70	70	60
$\hat{p}_0$	-55	-60	-55	-70	-70	-60
$\hat{v}_0$	80	80	65	70	70	60

TABLE II

SCENARIO CROSSING ORDER

# scenario	1	2	3	4	5	6	7
Order	1	1	2	1	1	2	2
	2	2	1	2	2	1	1
	6	4	3	5	3	3	3
	3	3	4	4	4	4	6
	4	5	5	3	6	6	4
	5	6	6	6	5	5	5

vehicle i is assigned the cost

$$J_i(z_i) = \sum_{k=0}^{N} Q_i(v_{i,k} - v_{i,k}^{\mathrm{d}})^2 + R_i u_{i,k}^2 \qquad (13)$$

where Q_i and R_i are vehicle-specific weights and $v_{i,k}^{\mathrm{d}}$ is the desired speed at time instant k.

In the previous sections, we have presented the algorithm to compute the in-out times. While one could formulate the MPC problem of each vehicle based on Probelm (1), we preferred to adopt a slightly modified version, where constraints (1e)-(1f) have been relaxed as $p_i(t_i^{\mathrm{in}}, z_i) \leq p_i^{\mathrm{in}}$, $p_i(t_i^{\mathrm{out}}, z_i) \geq p_i^{\mathrm{out}}$. This relaxation still enforces the safety-critical collision avoidance constraint, but does not try to compensate for safe perturbations. However, the coordination problem has been formulated by relying on (1). Moreover, in order to retain feasibility even in the presence of sensor noise and perturbations, we formulated these constraints as soft constraints with exact penalty [6], [14].

A. Simulations

All simulation parameters are specified in Tables I and II, where the units have been omitted for ease of reading. We used $T_{\mathrm{s}} = 0.1$ s and the intersection was defined by $p_i^{\mathrm{in}} = 0$ m, $p_i^{\mathrm{out}} = 8$ m and the control bounds are $-u^{\mathrm{lb}} = u^{\mathrm{ub}} = 2\ \mathrm{m/s^2}$ for each vehicle i.

In order to compare the two feasibility-enforcing approaches by means of numerical simulations, we have solved 5 scenarios using both the projection and the slack-based feasibility-enforcing techniques. In all simulations we used $\gamma = 0.01$, $\beta = 0.5$, $\epsilon = 10^{-2}$, and a QP accuracy of 10^{-8}.

We stress again that evaluating functions f and h constitutes the major computational cost (solving QP (1) and the two LPs (4)), while the sensitivity information ∇f, ∇h, $\nabla^2_{tt}\mathcal{L}$ is much cheaper to compute. Therefore, every linesearch iteration has about the same computational cost as one SQP iteration, but a lower communication cost, since less information needs to be communicated.

The results are displayed in Table III: in some scenarios relaxation and projection yield comparable results in terms of SQP and linesearch iterates n_{SQP} and n_{ls} respectively.

TABLE III

SQP AND BACKTRACKING LINESEARCH ITERATIONS

# scenario		1	2	3	4	5	6	7
Projection	n_{SQP}	6	10	5	4	7	8	8
	n_{ls}	6	10	5	4	7	8	8
	n_{feas}	2	2	2	3	1	2	2
Relaxation	n_{SQP}	5	7	9	16	9	8	7
	n_{ls}	7	10	15	49	8	16	13
	n_{feas}	3	4	7	15	3	8	6

However, the amount of iterates n_{feas} to obtain feasibility is much lower for projection. Additionally, in some scenarios relaxation requires many more iterates than projection. We moreover remark that relaxation never required Hessian regularisations, while projection added some regularisation once in Scenarios 1 and 4 and twice in Scenario 7. Finally, that the cost of the first feasible solution found by the solver was up to 55% larger than the cost at convergence.

B. Experiments

In this subsection, we present the results of experiments performed at the AstaZero proving ground next to Gothenburg, Sweden. A video summarising the experimental tests is available at [9]. The coordination algorithm was validated in closed loop using three automated Volvo cars (two Volvo S60 sedans and one Volvo XC90 SUV) and vehicle-to-vehicle (V2V) communication from RENDITS [1]. Each vehicle was further equipped with Real-Time-Kinematic (RTK) GPS receivers, inertial sensors, and a MicroAutobox 2 real-time computer connected to a latop, on which the experiment software was executed. The acceleration requested by MPC is dispatched to a lower-level controller designed by Volvo, given as a black-box to us. Though the the engine dynamics and gear shifts are rather nonlinear, the presence of this controller makes the system easier to control.

The SQP algorithm was applied in a distributed fashion using the V2V communication, where the SQP subproblems (7) computing the primal-dual updates were solved at a central computational node. The central node was placed close to the intersection, but in principle its role could have been taken by any of the vehicles, thus eliminating the need for a dedicated infrastructure. The evaluation of the cost V_i, constraint bounds $t_i^{\mathrm{out,ub}}$, $t_i^{\mathrm{out,lb}}$ and corresponding sensitivities were done on-board the vehicles by solving the QPs and LPs using HPMPC [8] on the laptop connected to the MicroAutobox 2. Due to its higher reliability, the projection approach was chosen for the experimental implementation.

In each experiment the vehicles were controlled from stand-still to a configuration from which a three way collision would occur if the speeds of all vehicles were held constant. The coordination was set to start at a predefined time. Short before, the NLP was solved using the proposed distributed algorithm by relying on wireless communication between the vehicles and a central node.

Over the experimental campaign, we successfully completed over 100 runs with 3 vehicles and over 50 runs with 2 vehicles, and explored various parameter settings. However, for the sake of brevity, we present the results of only one run

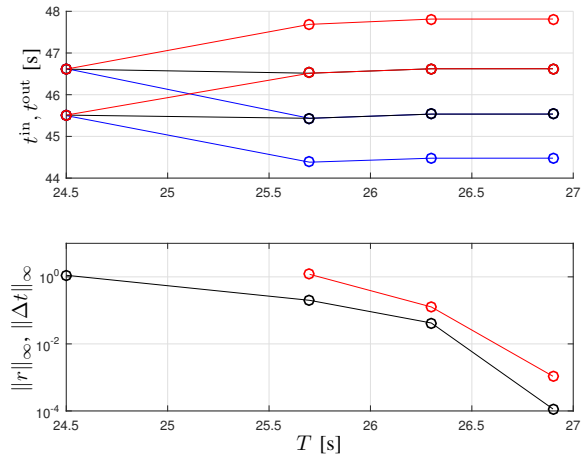

Fig. 2. SQP convergence and optimal times: first car in blue, second in black and third in red. Top graph: in-out times for the three vehicles. Bottom graph: norm of the KKT residual r (black) and step Δt (red).

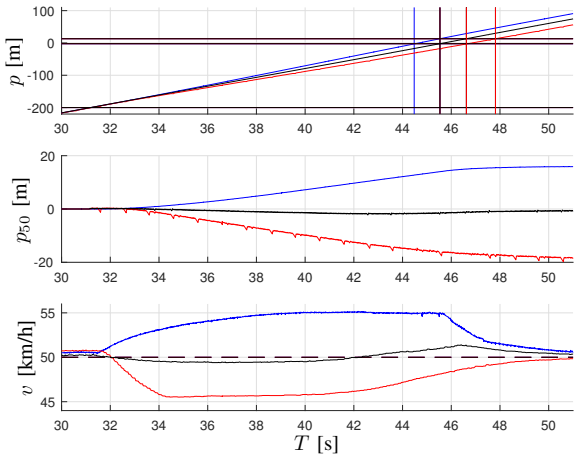

Fig. 3. Position trajectories (top graph, a zoom is given in Figure 4): first car in blue, second in black and third in red. The intersection start and stop is marked by two black lines, the coordination starts when the three vehicles are at $\hat{p}_0 = -200$ m and the vertical lines mark the in- out-times for each vehicle. Deviation of the position with respect to the constant speed scenario (middle graph). Velocity trajectory (bottom graph): reference velocity v^{d} in dashed lines and measured velocities in continuous line.

in this paper. In this run, the vehicles were all controlled to be at $\hat{p}_0 = -200$ m, $\hat{v}_0 = 50$ km/h at time $T_{\text{c}} = 31.3$ s from the beginning of the run, which was also used as the starting time for the coordination. Furthermore, for all vehicles, the control objective was to track the desired velocity $v^{\text{d}} = 50$ km/h, using the weights $Q_i = 1 \frac{\text{s}^2}{\text{m}^2}$ and $R_i = 10 \frac{\text{s}^4}{\text{m}^2}$ and the horizon length $N = 200$. The control bounds were set to $u^{\text{lb}} = -3$ m/s^2, $u^{\text{ub}} = 1.6$ m/s^2. Finally, we used $\gamma = 0.01$, $\beta = 0.5$, $\epsilon = 10^{-2}$, and a QP accuracy of 10^{-8}.

The evolution in time of the KKT residual r, the step size Δt and the solution at the given time are displayed in Figure 2. In this run, the solver always takes full steps without any projection, therefore, feasibility is obtained after one step and the successive steps improve optimality. All times are given with respect to the beginning of the scenario, which includes the startup phase to bring the vehicles to the assigned initial configuration from standstill. While the SQP algorithm is initialized at $T = 24.0$ s, the sensitivities for the first iterate become available only at $T = 24.5$ s. To compute each SQP iterate, the central node needs to wait 5, 10, 5 and 5 sampling intervals respectively. These comparatively long iteration times are due to implementation-specific details and could be reduced significantly with a better implementation. In particular, we emphasize that the total computation time, i.e. solving the central SQP subproblems and the vehicle-level QPs and LPs, only took about 3.4% of the total execution time. Moreover, due to the unreliability of the wireless communication channel resulting in packet drops, the total time per iterate varies over the iterations. Therefore, variations would be present even with a more efficient implementation, though with a much smaller magnitude.

The closed-loop trajectories are displayed in Figure 3. As one can expect, at the solution $t_1^{\text{out}} = t_2^{\text{in}}$ and $t_2^{\text{out}} = t_3^{\text{in}}$. The vehicles deviate from their constant-velocity reference in order to meet the crossing times. The second car first brakes and then accelerates to reduce the time spent inside the intersection, thus allowing the other cars to deviate less from their references and, therefore, reduce the overall cost.

Without coordination, all vehicles would proceed from the coordination starting point at a constant velocity $v = 50$ km/h. The difference in position between the coordinated and uncoordinated solutions, here denoted p_{50}, is also displayed Figure 3. Because we do not provide any reference for the position, after crossing the intersection, the first car is ahead of the position it would have in the absence of the other two cars, whereas the last vehicle lags behind.

The positions shown in Figures 3-4 were computed based on readings from GPS receivers, and the quality of the signal differs between three cars. The GPS unit for the third car (red line in the plots) systematically provided unreliable measurements with a frequency of 1 Hz. Due to these errors and the mismatch between the prediction model and the physical vehicles, the constraints enforcing the in- and out-times are slightly violated by the closed-loop system. We define the time violations as δt_i^{in}, δt_i^{out}, where a positive sign means that the actual time was larger than planned, such that $\delta t_i^{\text{in}} \geq 0$ corresponds to a safe configuration (the vehicle entered late), while $\delta t_i^{\text{out}} \geq 0$ corresponds to a dangerous configuration (the vehicle left late). For the position we kept the convention of the constraint definition, such that $p_i(t_i^{\text{in}}, z_i) - p_i^{\text{in}} \leq 0$ is safe, while $p_i(t_i^{\text{out}}, z_i) - p_i^{\text{out}} \leq 0$ is dangerous. The estimated violation of the in-out times for the experiment run is given in Table IV, where the values are taken from the position signal displayed in Figure 4. For vehicles 2-3 this signal is rather noisy and not very reliable, hence the large violations. We remark that the observed constraint violations can be easily accounted for in the proposed setup by constraint tightening, i.e. by enlarging the intersection definition. Because the constraint has the dimension of a distance, the tightening procedure can be directly related to positioning errors.

The analysis of the experimental results suggests that using very simple linear dynamics and affine constraints is accurate

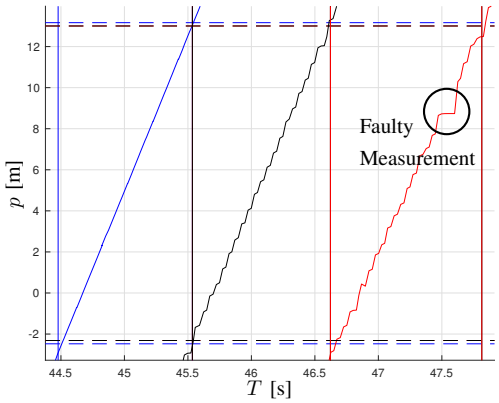

Fig. 4. Zoom of the GPS position measurements from Figure 3 in the intersection area. The horizontal dashed lines mark the beginning and the end of the intersection for each vehicle; the red line is superposed to the black one, as the two corresponding cars are identical. The vertical lines mark the in and out times sent to each car by the central node.

enough to safely control the vehicles and the largest source of uncertainty were unreliable sensor readings from one of the vehicles. The proposed decomposition of the control scheme showed the following desirable properties: (a) it did not need efficient and reliable communication; (b) even if not implemented in the most efficient way, the algorithm converged fast enough for a real-time implementation; (c) after the solution has been computed, communication is not required and the vehicles can be safely controlled in a decoupled fashion; (d) sporadic wrong sensor readings are natively handled by the algorithm.

After a thorough experimental campaign in a safe configuration, i.e. using parallel lanes, we ran some experiments on a real crossing scenario. For safety reasons, we added 5 m to the intersection length. An aerial picture is displayed in Figure 5, with the intersection shown as a red box. In the real crossing experiment, we obtained results comparable to those obtained in the parallel lane configuration.

V. Conclusions and Future Research

We have presented a distributed control scheme for optimal vehicle coordination at intersections based on an ad-hoc primal decomposition algorithm with a feasibility-enforcing projection. Simulation results suggested a lower reliability of the competing relaxation feasibility-enforcing approach.

We have tested our control approach in experiments using real cars on a test track. Though our implementation of the algorithm on the real-time platform was simple and far from optimal, the experimental results have demonstrated the applicability and real-time feasibility of our approach. Moreover, the sporadic faulty sensor readings and the unreliable communication have demonstrated some degree of robustness of our control approach.

Future work will consider improved algorithms that (a) update the in- and out-times in a closed-loop fashion, (b) robustly satisfy the constraints in the presence of positioning inaccuracies, (c) solve the ordering problem, i.e. the full mixed-integer problem.

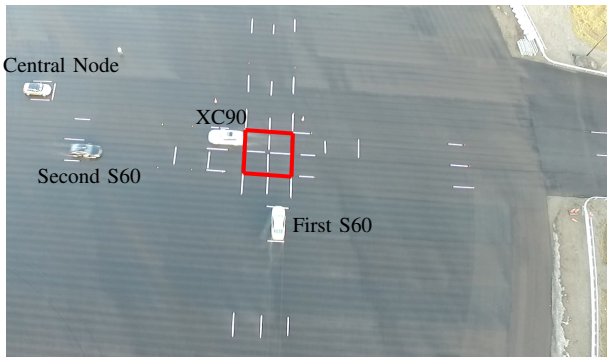

Fig. 5. Aerial picture of the crossing experiment.

TABLE IV

In-out time and position errors (SI units)

Car #	δt_i^{in}	δt_i^{out}	$p_i(t_i^{\text{in}}, z_i) - p_i^{\text{in}}$	$p_i(t_i^{\text{out}}, z_i) - p_i^{\text{out}}$
1	0.0287	0.0078	−0.4363	−0.1213
2	0.0077	−0.0156	−0.2461	0.2914
3	0.0480	0.0257	−0.6610	−0.5217

References

[1] RENDITS. http://www.rendits.com/. Accessed: 2016-11-05.

[2] Javier Alonso, Vicente Milans, Joshu Prez, Enrique Onieva, Carlos Gonzlez, and Teresa de Pedro. Autonomous vehicle control systems for safe crossroads. *Transportation Research Part C: Emerging Technologies*, 19(6):1095 – 1110, 2011.

[3] S.R. Azimi, G. Bhatia, R. Rajkumar, and P. Mudalige. Vehicular networks for collision avoidance at intersections. *SAE International Journal of Passenger Cars - Mechanical Systems*, 4(1):406–416, 2011.

[4] A. Colombo and D. Del Vecchio. Efficient algorithms for collision avoidance at intersections. In *Proceedings of the 15th ACM International Conference on Hybrid Systems: Computation and Control*, pages 145–154, New York, NY, USA, 2012.

[5] G. R. de Campos, P. Falcone, and J. Sjöberg. Autonomous cooperative driving: A velocity-based negotiation approach for intersection crossing. In *Intelligent Transportation Systems - (ITSC), 2013 16th International IEEE Conference on*, pages 1456–1461, 2013.

[6] N.M.C. de Oliveira and L.T. Biegler. Constraint Handling and Stability Properties of Model-Predictive Control. *AIChE Journal*, 40(7):1138–1155, 1994.

[7] R. Fletcher. *Practical Methods of Optimization*. Wiley, Chichester, 2nd edition, 1987.

[8] G. Frison, H.B. Sorensen, B. Dammann, and J.B. Jorgensen. High-performance small-scale solvers for linear model predictive control. In *Proc. of the European Control Conference*, pages 128–133, 2014.

[9] 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.

[10] 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.

[11] 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.

[12] J. Lee and B. Park. Development and Evaluation of a Cooperative Vehicle Intersection Control Algorithm Under the Connected Vehicles Environment. *IEEE Transactions on Intelligent Transportation Systems*, 13(1):81–90, 2012.

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

[14] P.O.M. Scokaert and J.B. Rawlings. Feasibility Issues in Linear Model Predictive Control. *AIChE Journal*, 45(8):1649–1659, 1999.

[15] 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.