Combined Scheduling and Control Design for the Coordination of Automated Vehicles at Intersections

Maximilian Kneissl [*,**] **Adam Molin** [*] **Sebastian Kehr** [*]
Hasan Esen [*] **Sandra Hirche** [**]

[*] *Corporate R&D, DENSO Automotive Deutschland GmbH, Freisinger Str. 21-23, 85386 Eching, Germany (e-mail: {m.kneissl,a.molin,h.esen}@denso-auto.de)*
[**] *Institute for Information-oriented Control, Technische Universität München, Arcisstr. 21, 80290 München, Germany (e-mail: hirche@tum.de)*

Abstract: Solving the problem of intersection crossing for autonomous vehicles is a challenging task due to combined combinatoric and dynamical control decisions. To reduce the complexity of the computations and distribute the resulting global optimization problem, we propose a combined scheduling-control method. Thereby, in this paper, we focus on the formulation of a resource-constrained-project-scheduling problem (RCPSP) to solve the combinatoric decision, i.e. the order in which vehicles cross an intersection area in a central coordination unit. This problem considers control decisions from the vehicles, which are computed using model predictive control (MPC) laws. In turn, the resulting scheduling solution is incorporated again in local vehicle MPC problems, which negotiate among each other to find a dynamically feasible solution. This seamless combination of scheduling and control results in efficient solutions, which is illustrated using numerical simulation and the results are compared with a first-come-first-served (FCFS) strategy.

Keywords: Interactive vehicle control; Autonomous vehicles; Intersection crossing; Distributed Model Predictive Control; Scheduling algorithms

1. INTRODUCTION

The increase in vehicle automation reveals great potential to improve ground traffic coordination. In particular, Vehicle-to-Vehicle and Vehicle-to-Infrastructure (V2X) communication capabilities of cars enables the implementation of scalable and intelligent vehicle coordination procedures. Coordinating vehicles optimally through intersections becomes a bottleneck in cooperative driving scenarios; this is because it requires combined combinatorial and dynamically optimized decisions. This problem has gained high interest in recent years.

A common approach to compute coordinated vehicle trajectories is to use optimal control (OC) methods and, in particular, model predictive control (MPC), see Zhang et al. (2016); Tedesco et al. (2010); Campos et al. (2014); Kneissl et al. (2018); Katriniok et al. (2017).These enable the consideration of complex vehicle dynamics and dynamics constraints as well as environmental constraints, and provide a predictive solution. The combinatorial decision describes the order in which vehicles cross the intersection area. It induces non-convexity into OC problems and makes them prohibitively difficult to solve to optimality when real-time requirements are given. Therefore, simple heuristics, such as first-come-first-served (FCFS), can be applied as by Zhang et al. (2016) and an improved FCFS rule by Kim and Kumar (2014). However, heuristics that are too simple risk leading to an overall suboptimal solution to the intersection coordination problem, e.g. if the dynamic differences between vehicles are not considered. Hult et al. (2018) provide a mixed-integer-quadratic-programming (MIQP) heuristic to overcome the aforementioned issue.

Alternative approaches deploy scheduling theory to solve the sequencing problem at automated intersections. These methods often use high-level traffic dynamics or do not consider vehicle dynamics at all, as proposed by Li and Zhou (2017) and Yan et al. (2011) using machine scheduling. In most cases, the objective of scheduling approaches is to find a solution that minimizes the problem's makespan, i.e. to minimize the overall time taken for all vehicles to cross the intersection, as proposed e.g. by Wu et al. (2012) and Vial et al. (2016). On the contrary, Colombo and Del Vecchio (2014) and Ahn and Del Vecchio (2016) design a scheduler that incorporates simple linear vehicle dynamics, which acts as a supervisor. It intervenes with the crossing vehicles in a least restrictive manner if

⋆ This work has been conducted within the ENABLE-S3 project that has received funding from the ECSEL Joint Undertaking under grant agreement No 692455. This joint undertaking receives support from the European Union's HORIZON 2020 research and innovation programme and Austria, Denmark, Germany, Finland, Czech Republic, Italy, Spain, Portugal, Poland, Ireland, Belgium, France, Netherlands, United Kingdom, Slovakia, Norway.

they would leave a safe set. Its implementation as a job-shop scheduling problem can be solved for 25 vehicles at an intersection within $100ms$. Scheduling problems can be solved by converting them to a mixed-integer-linear-programming (MILP) optimization problem. Thus, they can also be formulated directly as MILP problems as proposed by Fayazi and Vahidi (2018), in which the vehicles are coordinated through the intersection by receiving target arrival times from the central scheduler.

A main drawback of most scheduling approaches is that they lack the guarantee of dynamic feasibility with respect to local vehicle control systems. In contrast, formulating the intersection crossing task as a scheduling problem enables a seamless integration of precedence constraints. This, for example, excludes solutions that would lead to rear-end collisions, which is often not considered in OC formulations during the approaching phase at intersections. See Zanon et al. (2017); Kneissl et al. (2018).

Therefore, in this paper, we propose a combined scheduling-control design. It exploits the strength of scheduling to determine a safe and deadlock free crossing order and uses a distributed MPC framework, which guarantees feasibility of constraint dynamic vehicle systems. The intersection scenario is formulated as a resource-constrained-project-scheduling problem (RCPSP) and solved by converting it to an MILP problem. It receives a heuristic approximation for the vehicles' crossing duration, which makes the exchange of vehicle model data unnecessary. After extracting the pure order information and, thus, neglecting the timing information from the resulting schedule, the distributed local MPC problems compute timed-trajectories given this crossing order. The trajectories represent a guaranteed dynamically feasible and safe solution by incorporating local dynamic constraints and coupled inter-vehicle safety constraints for collision avoidance. The distributed MPC method is an iterative scheme with any-time feasible inter-sampling iterations and is discussed in detail in Kneissl et al. (2019).

The remainder of this paper is organized as follows. In Section 2, we introduce the model of an intersection and state the overall optimization problem as a centralized MPC formulation. Section 3 defines the scheduling problem and explains the link with local vehicle control problems, which are introduced in Section 4. Numerical results that apply the proposed approach are presented in Section 5, while Section 6 contains concluding remarks and comments about future work.

Notation: Throughout this paper $x(k|t)$ indicates a prediction of state x for time k computed at time t, while all prediction values at time t are $x(:\,|t)$. The set of integers $\mathbb{I}_{a:b}$ defines $\{a, a+1, ..., b\}$. The weighted 2-norm is denoted by $\|x-\hat{x}\|_Q^2 = (x-\hat{x})^\mathsf{T} Q (x-\hat{x})$ with appropriate dimensions of vectors $x, \hat{x}$ and matrix Q. The relation $a \overset{T}{\succ} b$ indicates that element a appears before element b in a tuple T. The cardinality of a set S is given by $|S|$.

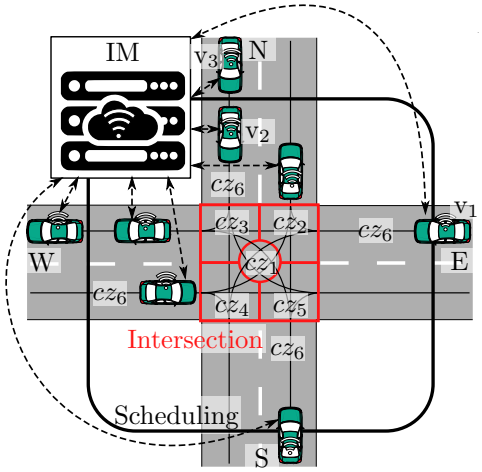

Fig. 1. Intersection crossing scenario.

2. PROBLEM STATEMENT

2.1 Intersection Framework

We model the intersection scenario consisting of a set of N_v connected automated vehicles (CAV) v_i, $i \in \mathbb{I}_{1:N_\mathrm{v}}$, with local control units and an intersection management (IM) unit with which the CAVs exchange information via vehicle-to-infrastructure (V2I) communication. Fig. 1 illustrates the introduced intersection setup. We distinguish between a scheduling area and an intersection area. Vehicles approaching the intersection enter the scheduling area in which, based on their local control computations, the IM determines a crossing order for the inner intersection area and shares this information with the vehicles. We introduce *conflict zones*, cz_j, $j \in \mathbb{I}_{1:6}$, dividing the intersection area into zones for potential rear-end, front and side collisions $(cz_1, ..., cz_5)$ and rear-end collisions on approaching lanes (cz_6). Note that, in general, the concept of conflict zones and the negotiation process presented in this paper is applicable to arbitrary scenarios in which several vehicles share a common area and where collision conflicts can occur, e.g. obstacle avoidance scenarios.

Finally, we define a multi-graph $\mathcal{G}_{route} = (V_{route}, E_{route})$ describing the given vehicles' routes through the intersection. The set of vertices $V_{route} = \{cz_1, ..., cz_6\}$ contains all conflict zones and $V^i_{route} \subset V_{route}$ all conflict zones that vehicle v_i passes. Directed edges $e_{jk} \in E_{route}$ are connections between consecutive conflict zones, which indicates that a vehicle crosses these zones and the driving direction, i.e. $e_{jk} = (cz_j \to cz_k)$, $j, k \in \mathbb{I}_{1:6}$.

2.2 Control Problem Setting

Next, we state the coordination problem of N_v vehicles through the intersection area as a centralized MPC problem, while we refer to its solution as the optimal solution. The central problem serves as a reference for the distributed implementation presented in this paper. It is defined as

$$V^* = \min_{Z} \; V\left(x(t), u(t)\right) \tag{1a}$$

subject to

$$x(k+1|t) = Ax(k|t) + Bu(k|t) \qquad k \in \mathbb{I}_{t:t+M-1} \tag{1b}$$

$$x(k|t) \in \mathbb{X} \qquad k \in \mathbb{I}_{t+1:t+M} \tag{1c}$$

$$u(k|t) \in \mathbb{U} \qquad k \in \mathbb{I}_{t:t+M-1} \tag{1d}$$

$$d_{\mathcal{N}}(d_i(k|t), d_j(k|t)) \geq d_s \qquad \begin{cases} k \in \mathbb{I}_{t+1:t+M} \\ (\mathrm{v}_i, \mathrm{v}_j) \in \mathcal{N} \end{cases} \tag{1e}$$

$$x(t|t) = x(t), \tag{1f}$$

where M is the prediction horizon. The time-discrete local vehicle dynamics models,

$$\underbrace{\begin{pmatrix} d_i \\ v_i \end{pmatrix}^+}_{x_i(t+1)} = \underbrace{\begin{pmatrix} 1 & -T_s \\ 0 & 1 \end{pmatrix}}_{A_i \in \mathbb{R}^{2\times 2}} \underbrace{\begin{pmatrix} d_i \\ v_i \end{pmatrix}}_{x_i(t)} + \underbrace{\begin{pmatrix} -T_s^2 \\ T_s \end{pmatrix}}_{B_i \in \mathbb{R}^{2\times 1}} \underbrace{a_i}_{u_i(t)}, \tag{2}$$

are combined to the central linear time invariant (LTI) model. Above sampling time T_s and d_i, v_i, a_i describe vehicle v_i's distance state, velocity state, and acceleration input, respectively. The distance state d_i will relate to the vehicle's distance to critical zones in the intersection area. For simplicity we neglect in the above notation the dependency of time $t = \{0, 1, 2, ...\}$. The central model (1b) consists of states and inputs

$$x(t) = \left(x_1(t)^{\mathsf{T}}, ..., x_{N_\mathrm{v}}(t)^{\mathsf{T}}\right)^{\mathsf{T}} \in \mathbb{R}^{2N_\mathrm{v}}, \tag{3}$$

and

$$u(t) = \left(u_1(t)^{\mathsf{T}}, ..., u_{N_\mathrm{v}}(t)^{\mathsf{T}}\right)^{\mathsf{T}} \in \mathbb{R}^{N_\mathrm{v}}, \tag{4}$$

respectively, a central system and input matrix definition

$$A = \mathrm{diag}(A_1, A_2, ..., A_{N_\mathrm{v}}) \in \mathbb{R}^{2N_\mathrm{v}\times 2N_\mathrm{v}} \tag{5}$$

and

$$B = \mathrm{diag}(B_1, B_2, ..., B_{N_\mathrm{v}}) \in \mathbb{R}^{2N_\mathrm{v}\times N_\mathrm{v}}. \tag{6}$$

States and inputs are constrained by closed and polytopic sets $\mathbb{X}$ and $\mathbb{U}$. The coupling constraint (1e) ensures inter-vehicle distances between vehicles v_i and v_j to be greater or equal to a defined safety distance d_s for all possible neighbor relations, $\mathcal{N}$, w.r.t. a common critical zone. Thereby, $\mathcal{N}$ is a set of all feasibly possible $(\mathrm{v}_i, \mathrm{v}_j)-$tuple, i.e.

$$\mathcal{N} \subseteq \{(\mathrm{v}_i, \mathrm{v}_j) | (i,j) \in \mathbb{I}_{1:N_\mathrm{v}} \times \mathbb{I}_{1:N_\mathrm{v}}, i \neq j\}. \tag{7}$$

The initial state at the current time step t is defined by (1f). Finally,

$$\begin{aligned} Z = (&x(t+1|t)^{\mathsf{T}}, ..., x(t+M|t)^{\mathsf{T}}, \\ &u(t|t), ..., u(t+M-1|t)), \end{aligned} \tag{8}$$

defines the optimization vector and

$$V\left(x(t), u(t)\right) = \sum_{i=1}^{N_\mathrm{v}} V_i\left(x_i(t), u_i(t)\right) =$$

$$\sum_{i=1}^{N_\mathrm{v}} \left(\sum_{k=t+1}^{t+M} \|x_i(k|t) - x_i^r\|^2_{Q_i(k)} + \sum_{k=t}^{t+M-1} \|u_i(k|t)\|^2_{R_i(k)} \right) \tag{9}$$

the global objective function, which is the sum of all individual vehicle objectives. Thereby, x_i^r is a given state reference, constant for one optimization step, and $Q_i(k)$ and $R_i(k)$ are positive semi-definite matrices.

One possibility is to solve problem (1) centrally in the IM infrastructure unit and share the computed system inputs with the respective vehicles. This, however, has several drawback. First, (1) grows with the number of vehicles what consequently results in a growing computation time. Second, constraint (1e) induces non-convexity to the problem which makes it hard to solve. Due to this combinatorial variability, (1) is a mixed integer quadratic program (MIQP). Third, the local vehicle models (2) have to be known by the central IM unit and cannot be kept locally in the vehicles.

To overcome the above discussed drawbacks, we propose a distributed coordination strategy. In the central IM unit we solve a scheduling problem which will be used to compute the combinatorial decision in (1e) and remove the non-convexity (Section 3). The remaining central MPC problem, which now is reduced to a quadratic problem (QP), is distributed between the vehicles (Section 4). Each vehicle solves the problem related to its own dynamics with shared trajectory information from neighboring vehicles. This gives the benefit of computationally distributing the problem and thus provides a scalable solution. Furthermore, dynamic models can be kept locally in each vehicle. Finally, from a safety perspective it is preferred to make control decisions inside a vehicle rather than relay on actuation commands coming from a central unit via a wireless V2I communication channel.

3. CROSSING ORDER SCHEDULING

The problem of automated intersection crossing, using the framework in Section 2.1, can be seamlessly formulated as a scheduling problem. Therefore, we first introduce the standard notation of a resource-constrained-project-scheduling problem (RCPSP) in the beginning of Section 3.1 as classified in Brucker et al. (1999). Following this, we map this notation to the intersection problem. Finally, we solve it by formulating a mixed integer linear problem (MILP) in Section 3.2.

3.1 Formulation of the scheduling problem

Let us define the RCPSP by the tuple

$$\left(\mathcal{V}, \delta, \mathcal{E}, \mathcal{R}, \alpha, \mathcal{B}\right), \tag{10}$$

where $\mathcal{V} = \{A_0, ..., A_{n+1}\}$ is a set of activities and the subset $\mathcal{A} = \{A_1, ..., A_n\} \subset \mathcal{V}$ with n non-dummy activities; $\delta \in \mathbb{N}^{n+2}$ is a vector describing the duration of each activity and we set the dummy activities' duration to $\delta_0 = \delta_{n+1} = 0$; matrix $\mathcal{E} \in \mathbb{N}^{l\times 2}$ contains $l \in \mathbb{N}$ precedence relations where each row in $\mathcal{E}$ with elements $(A_i, A_j) \in \mathcal{A}$, $i \neq j$, means that activity A_i precedes activity A_j; $\mathcal{R} = \{\rho_1, ..., \rho_m\}$ is the set of renewable resources with $m \in \mathbb{N}$; $\alpha \in \mathbb{N}^m$ is a vector describing the amount of available resources with the respective identifier; finally the matrix of demands is given by $\mathcal{B} = (\beta_{ir}) \in \mathbb{N}^{n+2\times m}$ with elements describing the amount of consumed resources $\rho_r \in \mathcal{R}$ for each activity $A_i \in \mathcal{V}$.

The mapping into the intersection framework is proposed as follows. Each non-dummy activity $A_i \in \mathcal{A}$, $i \in \mathbb{I}_{1:n}$, indicates a vehicle's route through the scheduling and intersection zone, i.e.

$$A_i = (e_{jk}, ..., e_{lm}), \text{where } e_{jk}, e_{lm} \in E_{route}. \tag{11}$$

We distinguish two types of activities. The first one, *drive to*, models the vehicle driving in the scheduling zone

towards the beginning of the intersection. The second one, *cross*, passing through the intersection. For clarification, consider the following demonstrative example.

Example 1. Assume vehicle v_1 in Fig. 1 driving from E to S, then its route is $cz_6 \rightarrow cz_2 \rightarrow cz_1 \rightarrow cz_4$. Vehicle v_1's *drive to* activity would be "driving in cz_6" and its *cross* activity would be "driving through cz_2, cz_1, and cz_4".

Duration δ_i of an activity $A_i \in \mathcal{A}$ models the expected time a vehicle consumes to perform the respective activity, i.e. to drive to the intersection or to cross it. The activity duration vector δ represents the interface to the local vehicle control problems, discussed in Section 4. Here several candidates for suitable heuristics exist. We propose an MPC-related duration measure in order to achieve a close link to the local vehicles' control decisions. Therefore, let us define

$$\delta_i = f_j(\hat{Z}_j), \tag{12}$$

with activity A_i's duration δ_i given by vehicle v_j's MPC optimization, $\hat{Z}_j$ (see Section 4). We introduce a precedence relation in $\mathcal{E}$ for the routes of a vehicle, i.e. the *drive to* activity precedes the *cross* activity. Furthermore, a precedence relation is added if we a priori know a certain crossing order, e.g. if two vehicles approaching the intersection on the same lane and the first cannot been taken over by its follower (rear-end collision avoidance) such as illustrated in the following example.

Example 2. Assume vehicle v_2 in Fig. 1 driving from N to S and vehicle v_3 from N to E. Furthermore, let $A_i = (cz_6 \rightarrow cz_3, cz_3 \rightarrow cz_4)$ be the intersection crossing activity of vehicle v_2, and $A_j = (cz_6 \rightarrow cz_3, cz_3 \rightarrow cz_1, cz_1 \rightarrow cz_5)$ of vehicle v_3. Then a pair (A_i, A_j) in $\mathcal{E}$ ensures that v_2 enters the intersection before v_3.

The resources represent the set of conflict zones in the scenario, i.e.

$$\mathcal{R} = \{cz_1, ..., cz_6\} \tag{13}$$

in our intersection setup. The availability of the respective resources at a certain time-instant is defined by

$$\alpha = (1, 1, 1, 1, 1, N_\mathrm{v})^\mathsf{T} \tag{14}$$

and elements of the demand matrix $\mathcal{B}$ are

$$\beta_{ir} = \begin{cases} 1 & \text{if } cz_r \in A_i \\ 0 & \text{else} \end{cases}. \tag{15}$$

Fig. 2 illustrates the construction of a precedence graph according to the definition of $\mathcal{E}$ and $\mathcal{R}$, while Table 1 summarizes the scheduling taxonomy related to the intersection model.

3.2 Solution of the RCPSP

Problem (10) can be solved by formulating it as MILP. We suggest a modified version of the discrete-time MILP formulation introduced by Pritsker et al. (1969).

Let $b_{i,\tau}$ be a binary decision variable with $b_{i,\tau} = 1$ if activity i starts at time τ and $b_{i,\tau} = 0$ otherwise, then (10) can be formulated as the following MILP:

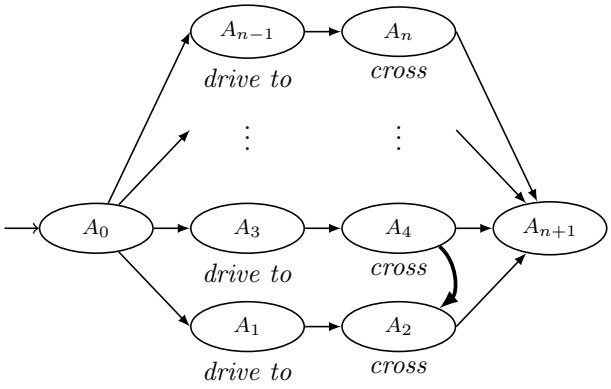

Fig. 2. Precedence graph representing the vehicles' routes through the intersection by distinguishing the activity types *drive to* and *cross*, as well as a priory known inter-vehicle relations as illustrated between A_4 and A_2.

Table 1. Intersection Scheduling Taxonomy

Scheduling meaning	Param.	Model Param.	Intersection meaning
non-dummy activity	A_i	$= (e_j, ..., e_k)$ $e_{jk}, e_{lm} \in E_{route}$	route through intersection
duration	δ_i	$= f_j(z_j)$	crossing duration prediction
precedence relations	$\epsilon_{k,:}$	$= (A_i, A_j)$	vehicles on same lane
resources	$\mathcal{R}$	$= \{cz_1, ..., cz_6\}$	conflict zones
availabilities	α	$= (1, ..., 1, N_\mathrm{v})^T$	# respective conflict zones
demands	β_{ir}	$= \begin{cases} 1 \text{ if } cz_r \in A_i \\ 0 \text{ else} \end{cases}$	passed czs

$$b^* = \underset{b}{\operatorname{argmin}} \sum_{i \in \mathbb{I}_{1:|\mathcal{V}|}} \sum_{\tau \in H} \tau b_{i,\tau} \tag{16a}$$

subject to

$$\sum_{\tau \in H} \tau b_{j,\tau} \geq \sum_{\tau \in H} \tau b_{i,\tau} + \delta_i \qquad (i,j) \in \mathcal{E} \tag{16b}$$

$$\sum_{i=1}^{n} \left(\beta_{i,k} \sum_{m=\tau-\delta_i+1}^{\tau} x_{i,m} \right) \leq \mathcal{B}_k \qquad \tau \in H, k \in \mathcal{R} \tag{16c}$$

$$\sum_{\tau \in H} b_{i,\tau} = 1 \qquad i \in \mathcal{V} \tag{16d}$$

$$b_{i,\tau} \in \{0,1\} \qquad i \in \mathcal{V}, \tau \in H, \tag{16e}$$

with $H = \{0, 1, ..., T_{sched}\}$ representing the set of scheduling time-steps upto a scheduling horizon T_{sched}, with $T_{sched} > MT_s$, and the optimization variable $b = (b_{1,1}, ..., b_{i,\tau}, ..., b_{n+2,T_{sched}})$, $i \in \mathbb{I}_{1:|\mathcal{V}|}, \tau \in H$. Note that (16a) is formulated such that it minimizes the problem's makespan as well as each activity's makespan, i.e. finds the solution with the shortest overall and individual vehicles' time consumption. This objective is chosen because the infrastructure's goal is to maximize the vehicle throughput in the intersection area.

The scheduling result can be conveniently illustrated by a Gantt-chart as exemplary shown in Fig. 3.

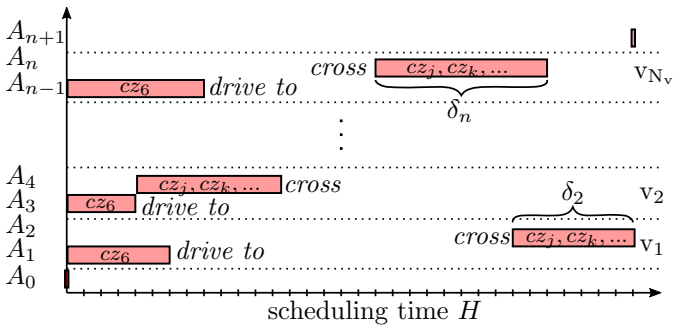

Fig. 3. Exemplary scheduling result indicating time and duration of execution for each activity, where for vehicles v_i activities *drive to* the intersection area and *cross* the intersection are distinguished. Furthermore, for each activity the consumed resources are illustrated with $j, k \in \{1, ..., 5\}$.

Given the result of (16) and the respective consumed resources of each activity $A_j \in \mathcal{A}$, we are able to extract a crossing order, o_i, for each critical zone $cz_i,\ i \in \mathbb{I}_{1:5}$. This is achieved by neglecting the actual scheduling time H and solely extract the order in which vehicles $\mathrm{v}_j,\ j \in \mathbb{I}_{1:N_\mathrm{v}}$ are scheduled to cross a certain conflict zone cz_i. Consequently, we achieve a set of orders

$$\mathcal{O} = \{o_i = (\mathrm{v}_j, \mathrm{v}_k, ...) \mid i \in \mathbb{I}_{1:5}; j, k \in \mathbb{I}_{1:N_\mathrm{v}}\}. \tag{17}$$

Remark 1. Note that modeling the scheduling such that activities reserve the complete intersection area (*cross*), does not mean that in the end only a single vehicle can be in the intersection, as we only extract the order decision and pass this information to the local control units, as discussed in the following section.

4. DISTRIBUTED COORDINATION CONTROL

In this section, we first discuss the distributed control law and its connection to the scheduling decision. Thereafter, we argue why the proposed RCPSP results in a deadlock free decision and thus enables a guaranteed feasible control negotiation.

4.1 Distributed MPC

Given the orders (17), computed with the scheduling law, the local vehicles can receive a set of neighbors. Based on this information the vehicle trajectories will be computed using distributed MPC laws with respective neighbor predictions. After formulating this distributed MPC setup we introduce how to predict the vehicles' activity duration (12).

The MPC laws, solved locally in each vehicle $\mathrm{v}_i, i \in \mathbb{I}_{1:N_\mathrm{v}}$, are given by:

$$Z_i^* = \underset{Z_i}{\operatorname{argmin}}\ V_i\left(k, x(t), u(t)\right) \tag{18a}$$

subject to

$$x_i(k+1|t) = A_i x_i(k|t) + B_i u_i(k|t) \quad k \in \mathbb{I}_{t:t+M-1} \tag{18b}$$
$$x_i(k|t) \in \mathbb{X}_i \quad k \in \mathbb{I}_{t+1:t+M} \tag{18c}$$
$$u_i(k|t) \in \mathbb{U}_i \quad k \in \mathbb{I}_{t:t+M-1} \tag{18d}$$
$$d_{\mathcal{P}_i}(k|t) \geq d_s \quad k \in \mathbb{I}_{t+1:t+M} \tag{18e}$$
$$d_{\mathcal{S}_i}(k|t) \geq d_s \quad k \in \mathbb{I}_{t+1:t+M} \tag{18f}$$
$$x_i(t|t) = x_i(t) \tag{18g}$$
$$x_i(t+M|t) \in \{x_i(t) | \mathrm{v}_i(t) = 0\} \tag{18h}$$
$$u_i(t+M-1|t) = 0. \tag{18i}$$

Thereby, Z_i is the complement of (8) containing only local information from vehicle v_i. Similar, (18b) - (18d) and (18g) are as defined in (1) with local information only. The application of terminal constraints (18h) and (18i) contribute to the recursive feasibility guarantee of the distributed computations (Kneissl et al. (2019)). For a given scheduling decision (17) we are able to formulate the distance constraints (18e) and (18f). Note that all feasible combinations $\mathcal{N}$ in (1e) are replaced by these local constraints for a single combination. This reduces the computational effort. The set of predecessors,

$$\mathcal{P}_i = \{\ \mathrm{v}_j \mid j \in \mathbb{I}_{1:N_\mathrm{v}} \wedge \mathrm{v}_j \overset{o_k}{\succ} \mathrm{v}_i, cz_k \in V_{route}^i \setminus cz_6\}, \tag{19}$$

contains all vehicles crossing before vehicle v_i on its route through the intersection. Then we derive $d_{\mathcal{P}_i}(k|t) = \tilde{d}_i^{cz_m}(k|t) - \tilde{d}_{j_k^*}^{cz_m}(k|t-1)$, with

$$j_k^* = \underset{j \in \mathcal{P}_i}{\operatorname{argmax}}\ \tilde{d}_j^{cz_m}(k|t-1), \tag{20}$$

and notation $\tilde{d}_i^{cz_m}$ indicating a transformation of $\mathrm{v}_i's$ distance state d_i to the beginning of the critical zone cz_m, which v_i and v_j have in common.

Similar for the set of successors,

$$\mathcal{S}_i = \{\ \mathrm{v}_j \mid j \in \mathbb{I}_{1:N_\mathrm{v}} \wedge \mathrm{v}_j \overset{o_k}{\prec} \mathrm{v}_i, cz_k \in V_{route}^i \setminus cz_6\}, \tag{21}$$

we construct $d_{\mathcal{S}_i}(k|t) = \tilde{d}_{j_k^*}^{cz_m}(k|t-1) - \tilde{d}_i^{cz_m}(k|t)$, while substituting $\mathcal{P}_i$ with $\mathcal{S}_i$ and argmax with argmin in (20).

We find that local problems (18) are convex as (18a) is quadratic and (18b) - (18i) are linear constraints. These problems are thus QPs and can be solved efficiently.

Above we described how the global scheduling decisions are incorporated in the local control problems. Now, it remains to discuss the reverse link between local control decisions and the scheduling problem. This link is represented by (12). Solving (18) is conducted in a distributed and iterative manner where neighbor intentions ((18e) - (18f)) are shared in each iteration step. Details on the distributed algorithm, which guarantees any-time feasible solutions, are presented in Kneissl et al. (2019). During iterations in the procedure each vehicle computes a nominal trajectory $\hat{Z}_i$, neglecting (18h) and (18i) when solving (18), and Z_i^* used to provide the feasibility guarantee. In what follows we will refer to the states from the nominal optimization vector $\hat{Z}_i$.

For a given vehicle $\mathrm{v}_i's$ activity A_j we extract $cz_s \in V_{route}$ and $cz_e \in V_{route}$ which are the first zone in the intersection of $\mathrm{v}_i's$ route and the zone after leaving the intersection area, respectively. We estimate the duration of activity A_j by

$$\delta_j = \lfloor t_{end} - t_{start} \rfloor, \tag{22}$$

which is computed distinguishing following cases:

$$t_{end} = \begin{cases} \min\left((t+M)T_s + \frac{\tilde{d}_i^{cz_e}(t+M|t)}{\mathrm{v}_i(t+M|t)}, tT_s + T_{sched}\right) \\ \qquad\qquad \text{if } \tilde{d}_i^{cz_e}(t+M|t) \geq 0 \\ k_{end}T_s, \qquad\qquad \text{if } \tilde{d}_i^{cz_e}(t+M|t) < 0, \end{cases} \tag{23}$$

with

$$k_{end} = \underset{k}{\operatorname{argmin}} \ |\tilde{d}_i^{cz_e}(k|t)| \quad \text{s.t.} \quad k \in \mathbb{I}_{t:t+M}. \tag{24}$$

And similar for t_{start} by substituting k_{end} with k_{start} and $\tilde{d}_i^{cz_e}$ with $\tilde{d}_i^{cz_s}$ in (23) and (24).

Finally, Algorithm 1 gives a summary of the combined scheduling-control coordination procedure.

Algorithm 1 Combined Scheduling-Control Procedure

1: ***Vehicles:***
2: compute distributed MPC problems (18) without (18e),(18f),(18h),(18i)
3: ***IM:***
4: compute activity duration estimation (22)
5: solve (16) and determine (17)
6: ***Vehicles:***
7: negotiate distributed intersection crossing using (18) according to Kneissl et al. (2019)

4.2 Feasibility Discussion

This section discusses on an example the deadlock free scheduling solution and how it relates to dynamic feasibility of local control decisions. Proving generality of this result is subject of future work.

Assume two vehicles, v_1 and v_2, conduct a left turn from opposite directions. Then, in the $d_1(t)$-$d_2(t)$-state-space there are unfeasible areas due to commonly passed *czs*, as illustrated by the red boxes in Fig. 4. As the *cross*-activity of the RCPSP (10) groups all passed *cz*s of a vehicle, a consistent order solution is computed. Compare the blue shaded area in the left plot of Fig. 4, which indicates the possible $d_1(t)$-$d_2(t)$-trajectory space if v_1 has to cross before v_2. This space is a connected set and thus there exists a homotopy class of $d_1(t)$-$d_2(t)$-trajectories (Gregoire et al. (2014)). On the contrary, the right plot in Fig. 4 shows a non-consistent order decision where the trajectory ends in deadlock configuration D and thus the goal configuration G cannot be reached.

Now, it remains to discuss the relation to dynamic feasibility of the local MPC problems. In Kneissl et al. (2019) we discuss feasibility for the area $d_1^e < d_2^s$, which is considered via constraints (18e) and (18f). Moving the trajectory there, i.e. from S to A in Fig. 4 is out of scope of this paper, but we have a simple guarantee to reach it, if a vehicle is able to stop before the entrance of an intersection. After v_1 has left the intersection area the constraint can be unbound (B in Fig. 4). Thus, we conclude that a deadlock free schedule enables a dynamically feasible vehicle coordination.

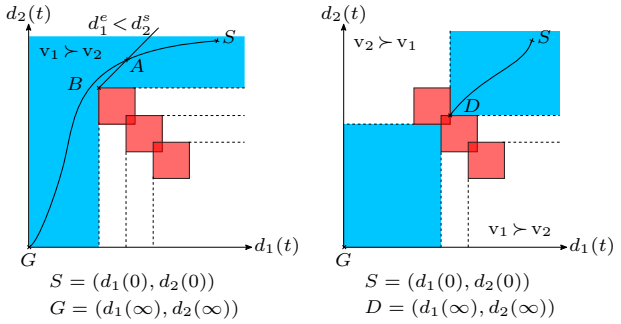

Fig. 4. Feasible trajectory space (blue shaded area) in the d_1-d_2-space for two vehicles on a left turn from opposite directions. Left: consistent scheduling decision with $\mathrm{v}_1 \succ \mathrm{v}_2$ for all commonly passed *cz*s; Right: non-consistent scheduling decision ($\mathrm{v}_1 \succ \mathrm{v}_2$ and $\mathrm{v}_1 \prec \mathrm{v}_2$) leading to a deadlock situation.

Table 2. Simulation parameters.

Name	**Parameter**	**Value**
# simulated vehicles	N_v	6
sampling time	T_s	$0.1s$
MPC horizon length	M	50
state weights	Q_i^n	$\mathrm{diag}(0,5)$
input weight	R_i^n	12
velocity constraints	$[v_{i,min}, v_{i,max}]$	$[0, 9m/s]$
acceleration constraints	$[a_{i,min}, a_{i,max}]$	$[-7m/s^2, 4m/s^2]$

Scenario	vehicle velocity references (m/s)					
	$x_{1,v}^r$	$x_{2,v}^r$	$x_{3,v}^r$	$x_{4,v}^r$	$x_{5,v}^r$	$x_{6,v}^r$
1	5	6	5	7	8	9
2	5	5	5	5	5	5
3	5	5	9	5	5	5

5. NUMERICAL EVALUATION

In this section we illustrate the functionality of the introduced RCPSP method in combination with the distributed MPC laws. Furthermore, we discuss its benefit in comparison to a first-come-first-served (FCFS) strategy on example scenarios. Using the FCFS law, the vehicle's crossing order in the intersection area is determined according to their distance to the entrance of the intersection. That means, the closest vehicle reserves the critical zones it passes first and similar for following vehicles organized by ascending distance.

Figure 5 introduces the vehicle setups of the simulated scenarios. *Scenario* 1 is presented in the top plot and *Scenarios* 2 and 3 at the bottom. The plots show the vehicle IDs, $\mathrm{v}_i, i \in \mathbb{I}_{1:6}$ and their respective maneuvers in the intersection, with right turn r, left turn l, and straight crossing s. Table 2 lists the simulation parameters. It distinguishes the reference values $x_{i,v}^r$ of the velocity state for vehicles $\mathrm{v}_i, i \in \mathbb{I}_{1:6}$ with respect to Scenarios $1-3$. Furthermore, the inter-vehicle safety distance $d_s = d_{cz} + l_\mathrm{v}$ is computed by considering the length of a route through a critical zone, d_{cz}, two vehicles have in common and the length of a vehicle, l_v, where we assume for simplicity that all vehicles have same dimensions.

Now, we demonstrate the simulation results by applying Algorithm 1 for Scenarios $1-3$ in Figures 6,7, and 8, respectively. In these figures the top plot represents the

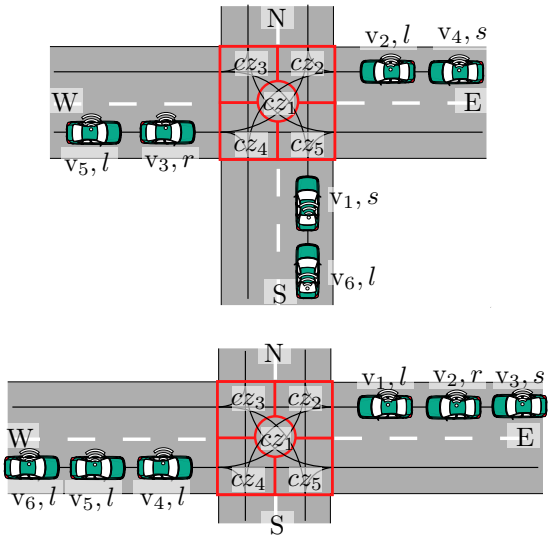

Fig. 5. Vehicle setup in simulation scenarios with respective vehicle maneuvers through the intersection, right turn r, left turn l, and straight cross s. Top plot: Scenario 1, bottom plot: Scenarios 2 and 3.

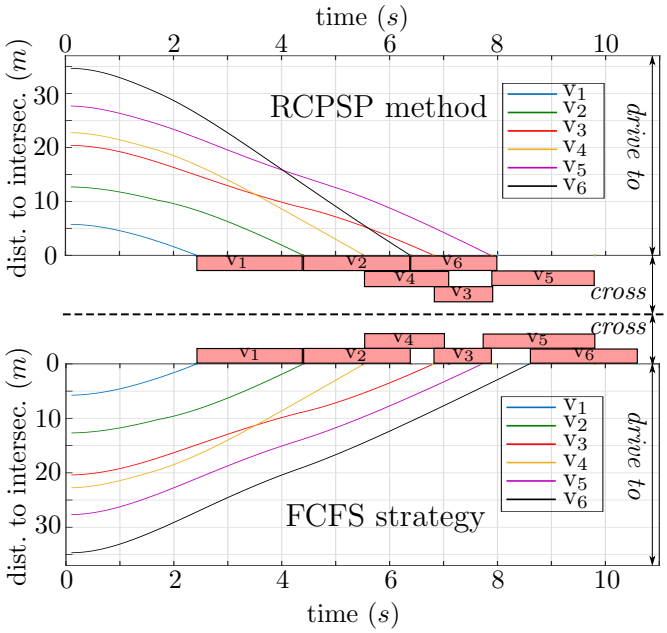

Fig. 6. Scenario 1.

actual *drive to* activities using the distributed MPC laws for a computed crossing order resulting form the RCPSP solution. The lower plot represents the implemented FCFS strategy and the bars in the middle part indicate the actual *cross* duration of the vehicles for RCPSP and FCFS, respectively.

As an example we state for Scenario 1 the estimated duration of activities:

$$\delta = (2.6s, \mathbf{1.4}s, 3.8s, \mathbf{1.4}s, 6.0s, \mathbf{0.6}s, \quad (25)$$
$$5.0s, \mathbf{1.0}s, 5.2s, \mathbf{1.0}s, 5.8s, \mathbf{1.0}s), \quad (26)$$

where the bold values are the estimated *crossing* duration for vehicles $v_1 ... v_6$, respectively, and the other vales are the estimated *drive to* duration. The resulting RCPSP order decision set $\mathcal{O}$ is given by

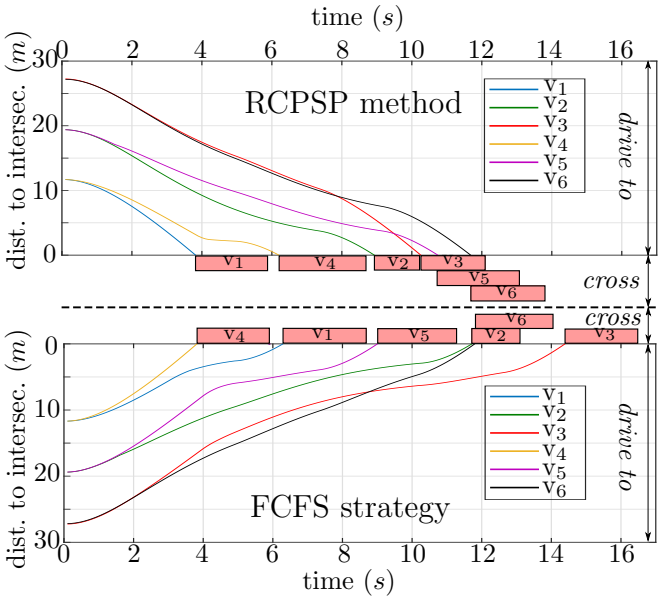

Fig. 7. Scenario 2.

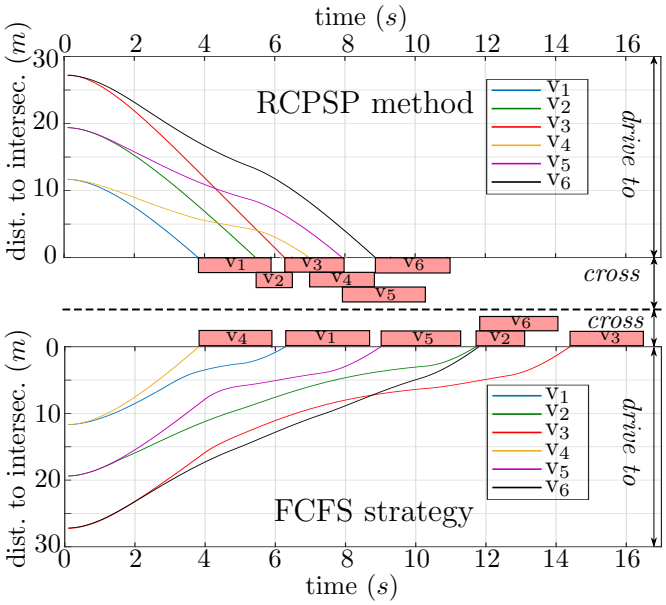

Fig. 8. Scenario 3.

$$\mathcal{O} = \{o_1 = (v_2, v_6, v_5), o_2 = (v_1, v_2, v_4, v_5), o_3 = (v_4, v_6),$$
$$o_4 = (v_2, v_3, v_5), o_5 = (v_1, v_6)\}.$$

Table 3 summarizes the simulation results by comparing the proposed RCPSP method and the FCFS strategy. Time represents the total simulated time until all vehicles have crossed the intersection area, Δ time lists the percentage of saved time by applying the RCPSP instead of FCFS. Cost indicates the sum of all vehicle individual MPC costs over the time of simulation, k_{sim}, i.e. $\sum_{k=1}^{k_{sim}} V(x(t), u(t))$, and Δ cost the saving, similar as above. We find that the RCPSP method outperforms the FCFS strategy. This becomes significantly visible in Scenario 3, where vehicle v_3 has a higher velocity reference value ($x^r_{3,v} = 9$) compared to the other vehicles. The RCPSP method is able to consider the individual vehicle interests (references or weights), while the FCFS strategy is not aware of that.

Table 3. Overall crossing time and cost evaluation for different intersection scenarios.

Method	Time (s)	Δ time (%)	Cost	Δ cost (%)
Scenario 1				
FCFS	10.9	6.4	3.75e4	16.8
RCPSP method	10.2		3.12e4	
Scenario 2				
FCFS	16.8	15.5	3.87e4	22.2
RCPSP method	14.2		3.01e4	
Scenario 3				
FCFS	16.8	32.7	7.06e4	62.6
RCPSP method	11.3		2.64e4	

This leads to a more efficient crossing order decision with the RCPSP.

6. CONCLUSION

In this paper we present a combined scheduling-control methodology for intersection crossing of automated vehicles. The scheduling problem, formulated as a resource-constrained-project-scheduling problem, receives crossing duration estimations via V2I communication form the local vehicle MPC controllers. Based on its solution, a crossing order in the intersection area can be found. This information is then applied in the distributed MPC negotiation procedure. Thus, this method makes use of the strength of scheduling problems by determining feasible crossing orders and guarantee dynamic feasibility through the locally distributed MPC problems. Furthermore, the distribution between the central intersection management unit (scheduling problem) and the local vehicles (distributed MPC) keeps the problem scalable and avoid the computation of a non-convex optimization problem. Finally, the individual vehicle models can be kept privately in each vehicle unit without the need to share it with other vehicles or coordination units. We show the benefit of the proposed method on numerical examples by a comparison with a first-come-first-served strategy.

Future work shall investigate the effect of re-scheduling during the crossing procedure. Moreover, computation time and the optimality gap with respect to the central solution will be subject of further research.

REFERENCES

Ahn, H. and Del Vecchio, D. (2016). Semi-autonomous intersection collision avoidance through job-shop scheduling. In *Proceedings of the 19th International Conference on Hybrid Systems: Computation and Control*, 185–194. ACM.

Brucker, P., Drexl, A., Möhring, R., Neumann, K., and Pesch, E. (1999). Resource-constrained project scheduling: Notation, classification, models, and methods. *European Journal of Operational Research*, 112(1), 3–41.

Campos, G.R., Falcone, P., Wymeersch, H., Hult, R., and Sjöberg, J. (2014). Cooperative receding horizon conflict resolution at traffic intersections. In *53rd IEEE Conf. Decision and Control (CDC)*, 2932–2937.

Colombo, A. and Del Vecchio, D. (2014). Least restrictive supervisors for intersection collision avoidance: A scheduling approach. *IEEE Trans. on Autom. Control*, 60(6), 1515–1527.

Fayazi, S.A. and Vahidi, A. (2018). Mixed-integer linear programming for optimal scheduling of autonomous vehicle intersection crossing. *IEEE Trans. on Intel. Veh.*, 3(3), 287–299.

Gregoire, J., Bonnabel, S., and De La Fortelle, A. (2014). Priority-based coordination of robots.

Hult, R., Zanon, M., Gras, S., and Falcone, P. (2018). An miqp-based heuristic for optimal coordination of vehicles at intersections. In *57th IEEE Conf. Decision and Control (CDC)*, 2783–2790.

Katriniok, A., Kleibaum, P., and Joševski, M. (2017). Distributed model predictive control for intersection automation using a parallelized optimization approach. *IFAC-PapersOnLine*, 50(1), 5940–5946.

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

Kneissl, M., Molin, A., Esen, H., and Hirche, S. (2018). A feasible mpc-based negotiation algorithm for automated intersection crossing. In *2018 European Control Conference (ECC)*, 1282–1288. IEEE.

Kneissl, M., Molin, A., Esen, H., and Hirche, S. (2019). A one-step feasible negotiation algorithm for distributed trajectory generation of autonomous vehicles. In *58h IEEE Conf. Decision and Control (CDC)*.

Li, P.T. and Zhou, X. (2017). Recasting and optimizing intersection automation as a connected-and-automated-vehicle (cav) scheduling problem: A sequential branch-and-bound search approach in phase-time-traffic hypernetwork. *Transportation Research Part B: Methodological*, 105, 479–506.

Pritsker, A.A.B., Waiters, L.J., and Wolfe, P.M. (1969). Multiproject scheduling with limited resources: A zero-one programming approach. *Management science*, 16(1), 93–108.

Tedesco, F., Raimondo, D.M., Casavola, A., and Lygeros, J. (2010). Distributed collision avoidance for interacting vehicles: a command governor approach. *IFAC Proceedings Volumes*, 43(19), 293–298.

Vial, J.J.B., Devanny, W.E., Eppstein, D., and Goodrich, M.T. (2016). Scheduling autonomous vehicle platoons through an unregulated intersection. *arXiv preprint arXiv:1609.04512*.

Wu, J., Abbas-Turki, A., and El Moudni, A. (2012). Cooperative driving: an ant colony system for autonomous intersection management. *Applied Intelligence*, 37(2), 207–222.

Yan, F., Dridi, M., and Moudni, A. (2011). A scheduling approach for autonomous vehicle sequencing problem at multi-intersections. *International Journal of Operations Research*, 9(1).

Zanon, M., Gros, S., Falcone, P., and Wymeersch, H. (2017). An asynchronous algorithm for optimal vehicle coordination at traffic intersections. In *20th IFAC World Congress*.

Zhang, Y.J., Malikopoulos, A.A., and Cassandras, C.G. (2016). Optimal control and coordination of connected and automated vehicles at urban traffic intersections. In *American Control Conference (ACC)*, 6227–6232.