# A One-Step Feasible Negotiation Algorithm for Distributed Trajectory Generation of Autonomous Vehicles

Maximilian Kneissl[1,2], Adam Molin[1], Hasan Esen[1], and Sandra Hirche[2]

*Abstract*— **We propose a distributed trajectory generation method for connected autonomous vehicles. It is integrated in an intersection crossing scenario where we assume a given vehicle order provided by a high-level scheduling unit. The multi-vehicle framework is modeled by local independent vehicle dynamics with coupling constraints between neighboring vehicles. Each vehicle in the framework computes in parallel a local model predictive control (MPC) decision, which is shared with its neighbors after conducting a convex Jacobi update step. The procedure can be iteratively repeated within a sampling time-step to improve the overall coordination decisions of the multi-vehicle setup. However, iterations can be stopped after each inter-sampling step with a guaranteed feasible solution which satisfies local and coupling constraints. We construct feasible initial trajectory candidates and propose a method to emulate the centralized solution. This makes the Jacobi algorithm suitable for distributed trajectory generation of autonomous vehicles in low and medium speed driving. Simulation results compare the performance of the distributed Jacobi MPC scheme with the centralized solution and illustrate the feasibility guarantee in an intersection scenario with unforeseen events.**

## I. INTRODUCTION

Autonomous cars bring the capability of making traveling on roads safer and also more efficient. The ability to share intention-based data with a vehicle's surrounding is a significant benefit of this technology and obtains much attention in research and development of automated driving. The control decisions, vehicles make, based on other vehicles' intentions often underlie optimization problems. When sharing this optimized intentions with surrounding vehicles via a potentially unreliable communication channel, it is an important requirement to provide a feasible and thus safe solution of the distributed optimization problem for each communication step.

In this paper we consider an automated intersection crossing scenario, where vehicles enter the intersection zone and receive a crossing sequence with respect to other vehicles in the zone from an intersection management (IM) infrastructure computer. Next, an inter-vehicle negotiation is accomplished to agree on a safe and efficient crossing trajectory based on other vehicles' decisions. The local vehicle decisions are computed via a model predictive control (MPC) law, whilst using independent plant models with coupling constraints between related vehicles, as e.g. proposed in [1].

In [2] an intersection problem, formulated as an MPC problem subject to system constraints, is solved analytically using Hamiltonian analysis. Collision avoidance between vehicles is guaranteed for the time vehicles leave the intersection area. A guarantee for the complete crossing phase is, however, not provided. For a given crossing priority, [3] discusses the solution of sequentially solving the vehicles' MPC problems and also extensions to approximative parallel implementations. [4] proposes an iterative procedure for safe intersection crossing where uncertainties, such as an emergency braking case, of vehicles during an approaching phase are considered. Authors of [5], [6] solve the intersection crossing problem by computing in- and out-times of a vehicle in a critical area iteratively while enabling asynchronous communication updates. These works are expanded with a robust solution and real-world experiments in [7]. A solution for parallel computation, without nested iterations, is presented in [8]. However, a lower-bound on the velocity, larger than zero, is required.

Primal decomposition methods are potential candidates to provide safety guarantees and the ability to dynamically react during crossing a critical intersection zone, while enabling parallel computations. In [9] a Jacobi decomposition is proposed for a distributed MPC setup with coupled system dynamics. An extension to coupling constraints is also considered, where resources leading to this coupling can be accessed mutually exclusive. Authors from [10] extend this for a general formulation of QP problems including shared constraints. It remains an open task to make this method suitable for trajectory generation in a distributed vehicle setup where reference values and constraints may change during run-time.

The main contribution of this paper is a distributed MPC trajectory generation algorithm for automated vehicles, where after each V2X-based (vehicle-to-everything) information exchange a recursive feasible solution is guaranteed. The method is suitable for low and medium speed scenarios such as urban driving and computation of the local vehicle MPC problems are conducted in parallel. The applied iterative Jacobi-based negotiation assures a safe vehicle coordination in each step, while with continuation of iterations, vehicles can improve their performance with respect to a global objective measure. The goal is to not only ensure safety

*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

[1]M. Kneissl, A. Molin, and H. Esen are with the Corporate R&D department of DENSO Automotive Deutschland GmbH, Freisinger Str. 21-23, 85386 Eching, Germany {m.kneissl,a.molin,h.esen}@denso-auto.de

[2]S. Hirche and M.Kneissl are with the Institute for Information- oriented Control, Technische Universität München, Arcisstrae 21, 80290 München, Germany hirche@tum.de

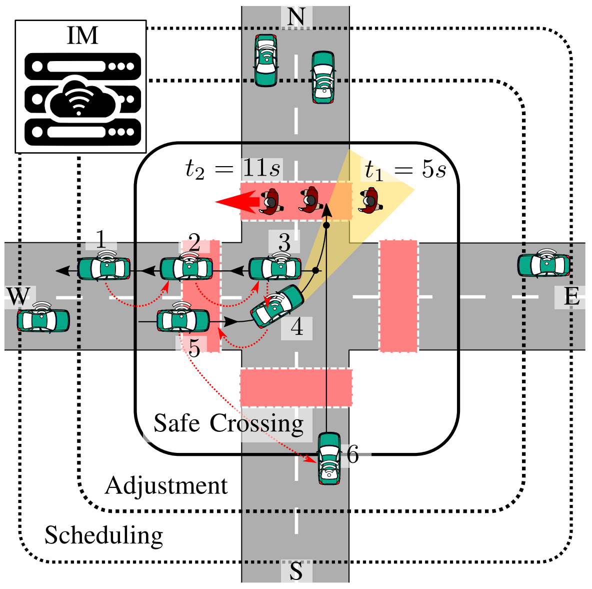

Fig. 1: Intersection crossing scenario.

of the resulting trajectories, but also contain smooth and performance-oriented properties, while enabling changing reference values. Therefore, we propose to apply a method, using a backward reach-set computation, to adjust the local MPC objective weights and thus influence the trajectory behavior. This algorithm is seamlessly integrated into an intersection crossing framework.

The remainder of this paper is organized as follows. In Section II we state the model of the intersection area, present the centralized vehicle coordination model, and decompose it into a distributed vehicle setup. Section III presents the iterative and parallel Jacobi method with proven feasible inter-sampling solutions. Next, we discuss the choice of feasible initial guesses for inter-sampling phases. Furthermore, the backward-reach-set method tailors the presented Jacobi method to trajectory generation of autonomous vehicles. Numerical examples are provided in Section IV, before we state concluding remarks and extension ideas for future work in Section V.

**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 $\mathbb{I}_{a:b}$ is defined by $\{a, a+1, ..., b\}$. The weighted 2-norm is denoted by $\|x-\hat{x}\|_Q^2 = (x-\hat{x})^\top Q(x-\hat{x})$ with appropriate dimensions of vectors $x, \hat{x}$ and matrix $Q$.

## II. Problem Statement

First, we briefly sketch the steps of receiving a vehicle dependency information from the IM in the *scheduling* zone, and the trajectory adaptation in the *adjustment* zone to achieve a safe inter-vehicle distance. Then we present the algorithmic steps conducted in the *safe-crossing* zone. Figure 1 illustrates the intersection setup.

### A. Intersection Model

We model an intersection consisting of three zones and an intersection management (IM) infrastructure unit. Vehicles approaching the intersection share their predicted trajectories with the IM unit when they enter the scheduling zone. Based on this information a centralized scheduling decision is computed, which represents the crossing sequence of vehicles in the intersection area. Formally, we achieve a directed graph

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}), \tag{1}$$

where vertices $\mathcal{V} = \{1, 2, ..., N_v\}$ label vehicles in the intersection area and $N_v$ is the total number of vehicles. Furthermore, a directed edge $e = (i, j) \in \mathcal{E}$, with $i, j \in \mathcal{V}$, indicates that vehicle $i$ is predecessor of vehicle $j$ and consequently crosses a common reference point first. Thus, the neighbor set of a vehicle $i$,

$$\mathcal{N}_i = \mathcal{P}_i \cup \mathcal{S}_i \tag{2}$$

can consist of a predecessor vehicle, $\mathcal{P}_i = \{j|(j,i) \in \mathcal{E}\}$, and a successor vehicle, $\mathcal{S}_i = \{j|(i,j) \in \mathcal{E}\}$.

Let $d_{ji}(:|t)$ indicate the distance of vehicle $i$ to vehicle $j \in \mathcal{P}_i$ with respect to a common reference point in the safe-crossing zone. Then, during driving in the adjustment zone vehicles adjust their trajectories such that the following assumption holds.

*Assumption 1:* When vehicle $i$ enters the safe-crossing zone at time step $t$, $d_{ji}(:|t) \geq d_s$ holds. Furthermore, the dimension of the adjustment zone is such that vehicles could come to a full-stop before leaving it.
The black dots at crossing paths in Figure 1 show examples of common reference points. Moreover, $d_s$ is a given inter-vehicle safety distance and we state an assumption on the central scheduling solution.

*Assumption 2:* The central scheduling decision is such that if $j \in \mathcal{N}_i$ and thus fulfills either $d_{ji}(:|t) \geq d_s$ or $d_{ij}(:|t) \geq d_s$ no other vehicle $k \in \mathcal{V}$, $k \neq i$ violates the safety condition $d_s$ with respect to vehicle $i$.
This ensures that the resulting neighboring description $\mathcal{G}$ represents an overall safe geometrical inter-vehicle relation.

### B. Centralized Coordination Model

First, we state the intersection coordination problem in the safe-crossing zone as a centralized MPC problem and then we discuss the decomposition of it in Section II-C. It is reasonable to decouple lateral and longitudinal vehicle dynamics for low vehicle speeds [11]. Therefore, we solely rely on longitudinal information for coordinating the vehicles.

Let the discrete longitudinal vehicle dynamics model for a single vehicle be of form

$$\underbrace{\begin{pmatrix} d_i \\ v_i \end{pmatrix}^+}_{x_i(t+1)} = \underbrace{\begin{pmatrix} 1 & -T_s \\ 0 & 1 \end{pmatrix}}_{A_i \in \mathbb{R}^{2\times 2}} \underbrace{\begin{pmatrix} d_i \\ v_i \end{pmatrix}}_{x_i(t)} + \underbrace{\begin{pmatrix} -T_s^2 \\ T_s \end{pmatrix}}_{B_i \in \mathbb{R}^{2\times 1}} \underbrace{a_i}_{u_i(t)}, \tag{3}$$

where $T_s \in \mathbb{R}^+$ is the discretization sampling time, the state $d_i(t) \in \mathbb{R}$ is the distance to the point where vehicle $i$ leaves the safe-crossing zone, $v_i(t) \in \mathbb{R}$ the vehicle's velocity, and control input $a_i(t) \in \mathbb{R}$ the acceleration. For simplified representation, we omit the dependency of the discrete time $t = \{0, 1, 2, ...\}$ for the state and input in (3). Let the central state vector be the aggregation of the local representations, such that

$$x(t) = \left(x_1(t)^\mathsf{T}, ..., x_{N_v}(t)^\mathsf{T}\right)^\mathsf{T} \in \mathbb{R}^{2N_v}, \tag{4}$$

and similar for the control input,

$$u(t) = \left(u_1(t)^\mathsf{T}, ..., u_{N_v}(t)^\mathsf{T}\right)^\mathsf{T} \in \mathbb{R}^{N_v}. \tag{5}$$

For a given MPC prediction horizon $M$, we define

$$X = (x(t+1|t)^\mathsf{T}, ..., x(t+M|t)^\mathsf{T}), \tag{6}$$

and

$$U = (u(t|t), ..., u(t+M-1|t)). \tag{7}$$

The central system and input matrices are

$$A = \text{diag}(A_1, A_2, ..., A_{N_v}) \in \mathbb{R}^{2N_v \times 2N_v} \tag{8}$$

and

$$B = \text{diag}(B_1, B_2, ..., B_{N_v}) \in \mathbb{R}^{2N_v \times N_v}, \tag{9}$$

respectively.

Now, denote the centralized MPC problem as

$$V^* = \min_{X,U} \quad V(k, X, U) \tag{10a}$$

subject to

$$x(k+1|t) = Ax(k|t) + Bu(k|t) \quad \forall k \in \mathbb{I}_{t:t+M-1} \tag{10b}$$

$$x(k|t) \in \mathbb{X} \quad \forall k \in \mathbb{I}_{t+1:t+M} \tag{10c}$$

$$u(k|t) \in \mathbb{U} \quad \forall k \in \mathbb{I}_{t:t+M-1} \tag{10d}$$

$$d_{ji}(k|t) \geq d_s \quad \begin{cases} \forall k \in \mathbb{I}_{t+1:t+M} \\ \{(i,j) | i \in \mathcal{V}, j \in \mathcal{S}_i\} \end{cases} \tag{10e}$$

$$x(t|t) = x(t). \tag{10f}$$

The objective function (10a) is

$$V(k, X, U) = \sum_{i=1}^{N_v} V_i(k, x_i(t), u_i(t)) =$$
$$\sum_{i=1}^{N_v} \left( \sum_{k=t+1}^{t+M} \|x_i(k|t) - x_i^r\|^2_{Q_i(k)} + \sum_{k=t}^{t+M-1} \|u_i(k|t)\|^2_{R_i(k)} \right) \tag{11}$$

where $x_i^r \in \mathbb{R}^2$ denotes a state reference, assumed constant for one computation step. The weighting matrices are assumed to be positive semi-definite, i.e. $Q_i(k) \succeq 0$ and $R_i(k) \succeq 0$. The model constraint (10b) results form (4), (5), as well as (8), (9), and constraint sets $\mathbb{X}$ and $\mathbb{U}$ are closed and polytopic. The distance constraint (10e) is constructed by $d_{ji}(k|t) = \tilde{d}_i(k|t) - \tilde{d}_j(k|t)$, where $\tilde{d}_j = f_j(d_j)$ is an affine transformation of vehicle $j$'s distance state to a reference point vehicle $i$ and vehicle $j$ have in common, and similar for $\tilde{d}_i$. Finally, (10f) defines the initial state. Note that because (10a) is of quadratic form and (10b) - (10f) are linear constraints, problem (10) is convex. Thus it can be cast in a standard QP form,

$$V^* = \min_{z} \quad V(z) \tag{12a}$$

subject to

$$\mathcal{A}_{eq} z - b_{eq} = 0 \tag{12b}$$

$$\mathcal{A}_{in} z - b_{in} \leq 0, \tag{12c}$$

where $z = (X, U)^\mathsf{T}$, $\mathcal{A}_{eq}$ and $b_{eq}$ result from (10b) and (10f), while $\mathcal{A}_{in}$ and $b_{in}$ result from (10c) - (10e). Certainly, $z$ is a time-dependent trajectory vector of the introduced form $z(k|t)$. We omit the time-dependency of $z$ and similar for its local representation $z_i$ to simplify notation when the context allows it.

## C. Model Decomposition

Now, we decompose the central problem (12) in order to distribute the computation among the vehicles in a sensible way. For each vehicle $i \in \mathcal{V}$ we define a local optimization vector $z_i$ containing elements from $z$. Each element of $z$ has to be contained in at least one resulting $z_i$.

Here, we apply a dynamically *decoupled-decomposition* with coupling constraints where $z_i$ contains only elements related to vehicle $i$'s dynamics. This has the benefit that the vehicles can preserve their local models as private and do not need to share it, or parts of it, with neighboring vehicles. It also keeps the number of optimization variables for each local problem at a minimum. However, we pay the price of a cooperative inter-vehicle behavior, yet feasibility and thus safety is ensured, what is the main focus in this paper.

Let $(X_i, U_i) = z_i^\mathsf{T}$, then the *decoupled-decomposition* results in a local vehicle MPC problem of the form

$$V_i^* = \min_{X_i, U_i} \quad V_i(k, X_i, U_i) \tag{13a}$$

subject to

$$x_i(k+1|t) = A_i x_i(k|t) + B_i u_i(k|t) \quad \forall k \in \mathbb{I}_{t:t+M-1} \tag{13b}$$

$$x_i(k|t) \in \mathbb{X}_i \quad \forall k \in \mathbb{I}_{t+1:t+M} \tag{13c}$$

$$u_i(k|t) \in \mathbb{U}_i \quad \forall k \in \mathbb{I}_{t:t+M-1} \tag{13d}$$

$$d_{ji}(k|t) \geq d_s \quad \begin{cases} \forall k \in \mathbb{I}_{t+1:t+M} \\ j \in \mathcal{P}_i \end{cases} \tag{13e}$$

$$d_{ij}(k|t) \geq d_s \quad \begin{cases} \forall k \in \mathbb{I}_{t+1:t+M} \\ j \in \mathcal{S}_i \end{cases} \tag{13f}$$

$$x_i(t|t) = x_i(t) \tag{13g}$$

$$x_i(t+M|t) \in \{x_i(t) | v_i(t) = 0\} \tag{13h}$$

$$u_i(t+M-1|t) = 0, \tag{13i}$$

here (13b) - (13d) and (13g) are defined with the same properties as in problem (10), but for values of a local system, respectively. We also assume that $\{x_i(t)|v_i(t) = 0\} \subset \mathbb{X}_i$ and that $\mathbb{U}_i$ contains the origin. Note that in a local vehicle we model the distance constraint to both a predecessor vehicle (13e) and a successor vehicle (13f), if either of those exist. These are constructed by receiving trajectory data from neighboring vehicles via a communication connection. As a result we achieve dynamically decoupled systems with shared coupling constraints. The need of the terminal constraints (13h) and (13i) is discussed in the following Section III.

## III. ITERATIVE JACOBI NEGOTIATION ALGORITHM

To guarantee one-step feasible solutions, we apply a Jacobi overrelaxation (JOR) algorithm to the problem of distributed trajectory generation in the intersection area (Chapter 2.4 [12]). Therefore, let us introduce the following notation for a vehicle $i$:

- $z_i^*$, the optimal solution of the local problem (13),
- $z_i^p$, the trajectory of the $p$-th inter-sampling iteration after a JOR update,
- $\hat{z}_i$, a feasible candidate for the local trajectory at the beginning of the JOR iteration procedure.

### A. Jacobi Overrelaxation Algorithm

Algorithm 1 is applied to compute the vehicles' trajectories in a distributed manner.

**Algorithm 1** Jacobi overrelaxation

At the current time step $t$:

*Step 1:* Set $p \leftarrow 0$, each vehicle $i \in \mathcal{V}$ receive trajectory candidates, $\hat{z}_j$, form neighbors $\mathcal{N}_i$, and set $z_j^p = \hat{z}_j$.

*Step 2:* In parallel, compute $z_i^*$, $\forall i \in \mathcal{V}$, by solving (13).

*Step 3:* Determine the next inter-sampling iterate,

$$z_i^{p+1} = \omega_i z_i^* + (1-\omega_i) z_i^p, \ \forall i \in \mathcal{V}, \tag{14}$$

where $\sum_{i=1}^{N_v} \omega_i = 1$ and $\omega_i > 0$. Share this information with vehicles $j \in \mathcal{N}_i$.

*Step 4:* If $V_i(z_i^p) - V_i(z_i^{p-1}) > \epsilon, \ \forall i \in \mathcal{V}$ and $p < p_{max}$, increase $p \leftarrow p+1$ and go to *Step 2*.
Else, apply $u_i(0|t)$, $\forall i \in \mathcal{V}$, to the local vehicle, increase $t \leftarrow t+1$, and go to *Step 1*.

In *Step 4* of Algorithm 1, $\epsilon$ is a specified convergence bound and $p_{max}$ an upper-bound on the number of inter-sampling iterations.

*Remark 1:* Regarding *Step 2*, remember that $d_{ji}(k|t) = \tilde{d}_i(k|t) - \tilde{d}_j(k|t) = f_i(z_i) - f_j(z_j^p)$ and similar for $d_{ij}(k|t)$.

*Remark 2:* The inter-sampling update (14) can be conducted fully distributed, due to the use of decoupled model dynamics.

*Proposition 1:* Given a feasible initial candidate $z_i^0$, $\forall i \in \mathcal{V}$, each consecutive iterate of (14) is again a feasible solution for problem (13).

*Proof:* The optimized trajectory $z_i^*$ is a feasible solution of (13) and the same holds for $z_i^0$, as assumed above. Sets $\mathbb{X}_i$ as well as $\mathbb{U}_i$ are convex polyhedrals, containing (13h) and (13i). Furthermore, $d_i(k|t)$ is convex in (13e) and (13f). Consequently, the convex combination (14) of $z_i^*$ and $z_i^0$ is again feasible for (13b) - (13i). The same result holds for $p > 1$, what follows by induction. ■

*Lemma 1:* The Jacobi iteration (14) has a non-diverging behavior.

*Proof:* Following the reasoning in Chapter 2.6 of [12], we find due to the convexity of (11) that

$$\begin{aligned} V(z^{p+1}) &= \sum_{i=1}^{N_v} V_i(z_i^{p+1}) = \sum_{i=1}^{N_v} V_i\left(\omega_i z_i^* + (1-\omega_i) z_i^p\right) \\ &\leq \sum_{i=1}^{N_v} \left(\omega_i V_i(z_i^*) + (1-\omega_i) V_i(z_i^p)\right) \\ &\leq \sum_{i=1}^{N_v} \left(\omega_i V_i(z_i^p) + (1-\omega_i) V_i(z_i^p)\right) = \sum_{i=1}^{N_v} V_i(z_i^p) \\ &= V(z^p). \end{aligned}$$

■

### B. Feasible Initial Guess

In *Step 4* of Algorithm 1 the trajectory candidate $\hat{z}_i$, for the next time step $t+1$, is determined and in *Step 1* the received neighbor information $\hat{z}_j$ is applied. In order to preserve feasibility of the overall constraint-coupled distributed problem these trajectory candidates have to present a feasible guess for problem (13).

We set

$$\hat{z}_i(t+1\!:\!t+M|t+1) = z_i^p(t+1\!:\!t+M|t), \tag{15}$$

then it remains to determine $\hat{z}_i(t+M+1|t+1)$. In general, it is not trivial to find $\hat{z}_i(t+M+1|t+1)$, which guarantees (13c) - (13f) for all involved vehicles. Therefore, we guess the following trivial candidate, as proposed similarly in [9], [10]:

$$x_i(t+M+1|t+1) = \begin{pmatrix} d_i(t+M|t) \\ 0 \end{pmatrix}, \tag{16a}$$

$$u_i(t+M+1|t+1) = 0, \tag{16b}$$

with $x_i, u_i \in z_i$.

The following proposition shows that this is a feasible guess.

*Proposition 2:* Let the local vehicle dynamics be modeled by (3). Then, the candidate trajectory $\hat{z}_i$ consisting of (15) and (16) is a feasible solution for problem (13).

*Proof:* Since trajectory $z_i^p(k|t)$, $k \in \mathbb{I}_{t:t+M}$ was feasible at time step $t$, it follows that $z_i^p(k|t)$, $k \in \mathbb{I}_{t+1:t+M}$ is a feasible solution at time step $t+1$. Furthermore, due to the terminal conditions (13h) and (13i), (16) fulfills (13c) - (13f) for $k = t+M+1$. Thus, $\hat{z}_i$ is a feasible trajectory candidate for (13). ■

Now, we can conclude feasibility for the overall connected vehicle framework.

*Definition 1:* A MPC problem is *recursive feasible* if the following conditions are met:

1) $x_i(k|t) \in \hat{\mathbb{X}}_i \Rightarrow x_i(k+1|t+1) \in \hat{\mathbb{X}}_i, \ k \in \mathbb{I}_{t:t+M}$,
2) $u_i(k|t) \in \mathbb{U}_i \Rightarrow u_i(k+1|t+1) \in \mathbb{U}_i, \ k \in \mathbb{I}_{t:t+M-1}$,

where $\hat{\mathbb{X}}_i$ defines a set containing all state constraints of the MPC problem, with a terminal state set $\mathbb{X}_i^M \subseteq \hat{\mathbb{X}}_i$.

*Theorem 1:* Each iteration step in the network $\mathcal{G}$, following Algorithm 1, ensures recursive feasibility for vehicles $i$'s local control problem (13), $\forall i \in \mathcal{V}$.

*Proof:* Summarize the zero velocity condition as the set $\mathbb{V}_i := \{x_i(t) | v_i(t) = 0\}$ and the distance constraint set as $\mathbb{D}_i := \{x_i(t) | d_{ji}(t) \geq d_s \wedge d_{ij}(t) \geq d_s\}$, then the terminal set of (13) is $\mathbb{X}_i^M := \mathbb{V}_i \cap \mathbb{D}_i$. For each vehicle entering the safe-crossing zone we know by Assumption 1 that $x_i(k|t) \in \mathbb{D}_i,\ k \in \mathbb{I}_{t:t+M}$, what consequently enables a feasible initial trajectory with $x_i(k|t) \in \mathbb{X}_i,\ k \in \mathbb{I}_{t:t+M}$ and $x_i(t+M|t) \in \mathbb{X}_i^M \subseteq \hat{\mathbb{X}}_i$. Proposition 1 guarantees that $x_i^p(k|t) \in \hat{\mathbb{X}}_i,\ \mathbb{I}_{t:t+M}$, and $x_i^p(t+M|t) \in \mathbb{X}_i^M$ remain feasible for all inter-sampling iterations $p$. Furthermore, in Proposition 2 we prove feasibility for a time-step transition and thus achieve $u_i(t+M|t+1) \in \mathbb{U}_i$ and $x_i(t+M+1|t+1) \in \mathbb{X}_i^M$, what ensures conditions 1) and 2) from Definition 1. This concludes recursive feasibility of each iteration step. ■

A fundamental difference between the centralized problem formulation (10) and the distributed decomposition (13) are the terminal constraints (13h) and (13i). These are required to guarantee Proposition 2.

Yet, the difference between the central and the distributed formulation likely leads also to a different behavior of the modeled vehicles. The goal is to avoid the deviation, coming from this source, in normal driving mode, i.e. driving without braking to full-stop. For that reason, we propose the method in the following Section III-C.

## C. Emulation of a Terminal-state-less Solution

In order to emulate the behavior of a trajectory which is not required to come to a full stop, we propose to apply time-dependent objective weights. Therefore, we introduce two planning phases within the horizon length $M$ of a local MPC problem (13). The first one is referred to as *normal-planning*, for $1 \leq k < k_{brake}$, and the second as *planning-to-full-stop*, for $k_{brake} \leq k \leq M$. Accordingly, we define the weights

$$Q_i(k)\,[R_i(k)] = \begin{cases} Q_i^n\,[R_i^n] & \text{for } 1 \leq k < k_{brake} \\ 0 & \text{for } k_{brake} \leq k \leq M \end{cases}, \tag{17}$$

where $Q_i^n$ is a constant weight for the normal-planning phase and similar for $R_i^n$. The point $k_{brake}$ is the prediction step from which onward the trajectory shall be planned to a full-stop of vehicle $i$, which is achieved by applying (17) in the local MPC objectives (13a). The trajectories are re-computed in each sampling time step in a receding horizon fashion, and only the first sample (as proposed in *Step 4* of Algorithm 1), or the first several samples, will be tracked by the respective vehicle. Thus, the goal is to move the planning-to-full-stop phase as far back in the prediction horizon as possible, and therefore achieve an actual driving behavior which is not affected of the planning-to-full-stop phase.

In what follows, we find the latest possible $k_{brake}$, i.e.

$$k_{brake} = \max\ K, \tag{18}$$

with

$$\begin{aligned} K = &\\ \{\tilde{k} \mid &\exists u_i(k|t) = u_i^*(k|t), k \in \mathbb{I}_{0:\tilde{k}-1}, \wedge \\ &u_i(k|t) \in \mathbb{U}_i, k \in \mathbb{I}_{\tilde{k}:M-1}, \text{ such that (13b) – (13i)}, \\ &\text{with } \tilde{k} \in \mathbb{I}_{1:M-1}\}. \end{aligned}$$

Here, $u_i^*(k|t)$ results from the solution of problem (13a) - (13g).

Let the backward reach-set

$$\begin{aligned} &\mathbb{X}_i^A(r) = \\ &\{x_i(r|t) | \exists z_i(k|t), k \in \mathbb{I}_{r:M}, \text{ such that (13b) – (13i)}\} \end{aligned} \tag{19}$$

be the set of admissible states at time step $r$ for which a trajectory exists that fulfill the problem constraints. Now, we apply the following method to find $k_{brake}$:

**Algorithm 2** Brake point $k_{brake}$

*Step 1:* Compute the desired trajectory $z_i^{des}$, with respective state values $x_i^{des}$, by solving (13) without (13h) and (13i). Set $r \leftarrow M-1$.

*Step 2:* Compute the backward reach-set $\mathbb{X}_i^A(r)$. If $x_i^{des}(r|t) \in \mathbb{X}_i^A(r)$, then $k_{brake} = r$ and continue with *Step 3*, else $r \leftarrow r-1$ and repeat *Step 2*.

*Step 3:* Compute (13) using (17) and including (13h) as well as (13i). Apply the result in the Jacobi negotiation scheme of Algorithm 1.

An essential point is the computation of $\mathbb{X}_i^A(r)$ in *Step 2* of Algorithm 2. Regarding this, we apply the set-projection-algorithm presented in [13]. The resulting admissible set $\mathbb{X}_i^A(r)$ is a polyhedral formulation of the form $S = \{x_i(r|t) | P x_i(r|t) + \gamma \leq 0\}, P \in \mathbb{R}^{l \times 2}$ and $\gamma \in \mathbb{R}^l$, as we assume an LTI model with linear constraints. Most parts of the set-projection-algorithm are computationally simple. However, the resulting matrix $P$, describing the set $S$ with $l$ inequality constraints is a redundant representation, i.e. there exists a $S' = \{x_i(r|t) | P' x_i(r|t) + \gamma' \leq 0\}$,$P' \in \mathbb{R}^{l' \times 2}$ and $\gamma' \in \mathbb{R}^{l'}$, with $l' < l$, such that $S = S'$. To keep the computational burden low during the recursive determination of $\mathbb{X}_i^A(r)$, it is required to find a $S'$ representing the minimal, or close-to-minimal representation. Efficient methods for that exist, as it is a well studied problem in the field of linear programming [14]. A possible method, which we apply, is presented in [15].

For the computation of $k_{brake}$, we assume that a vehicle's trajectory can be planned to full-stop during its horizon length, what results in a solution guarantee for Algorithm 2.

*Assumption 3:* The horizon $M$ of vehicle $i$'s MPC problem (13) is large enough to guarantee $\mathbb{X}_i \subseteq \mathbb{X}_i^A(1)$.

*Lemma 2:* Algorithm 2 terminates for problem (13) with a solution for $k_{brake}$.

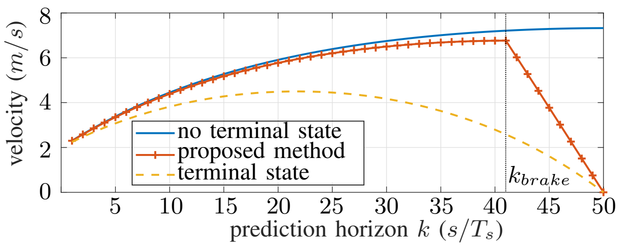

Fig. 2: Velocity state of example trajectory comparing solutions with no terminal state, proposed method of terminal-state-less emulation, and terminal state without time-varying objective weights.

TABLE I: Simulation parameters.

| Name | Parameter | Value |
|---|---|---|
| sampling time | $T_s$ | $0.1s$ |
| MPC horizon length | $M$ | 50 |
| state weights | $Q_i^n$ | $\text{diag}(0, 10)$ |
| input weight | $R_i^n$ | 5 |
| velocity constraints | $[v_{i,min}, v_{i,max}]$ | $[0, 10m/s]$ |
| acceleration constraints | $[a_{i,min}, a_{i,max}]$ | $[-7m/s^2, 4m/s^2]$ |
| safety distance | $d_s$ | $2m$ |
| update variable | $\omega_i$ | $1/N_v$ |

*Proof:* Termination is guaranteed if *Step 3* is reached. *Step 2* is always reached after *Step 1* and it continues to *Step 3* if state $x_i^{des}(r|t) \in \mathbb{X}_i^A(r)$, which is at the latest ensured for $r = 1$ due to Assumption 3. ∎

Figure 2 exemplary illustrates the difference between planning methods by plotting the predicted velocity state of a vehicle $i$. The solid line represents a solution without the terminal condition of coming to full-stop at the end of the prediction horizon. This would be the case in the centralized solution (10) extracted for a single vehicle. The method proposed in Algorithm 2 and plotted with the crossed line, shows a similar behavior to the terminal-state-less solution until $k_{brake}$. When using a terminal state without the emulation procedure, however, the resulting trajectory shows a significant divergence from the respective solution without terminal constraint (dashed line).

*Remark 3:* Note that the result of Algorithm 2 does not influence the feasibility guarantee of Theorem 1. It solely varies the MPC's objective function by (17) and effects thus only performance-related changes.

## IV. NUMERICAL EXAMPLES

In this section we stress the benefits and functionality of the proposed negotiation method by simulations. Table I lists the applied parameters of the respective MPC problems.

As a first scenario we simulate a negotiation process of $N_v = 6$ vehicles stimulated by a changing reference value. Figure 3 illustrates the influence of the inter-sampling iterations in the same simulation setup as above. Given a reference change form $v_{i,ref} = 4m/s \rightarrow 9m/s$, the trajectory approaches the centralized solution (dashed line) with increasing inter-sampling iterations $p_{max}$.

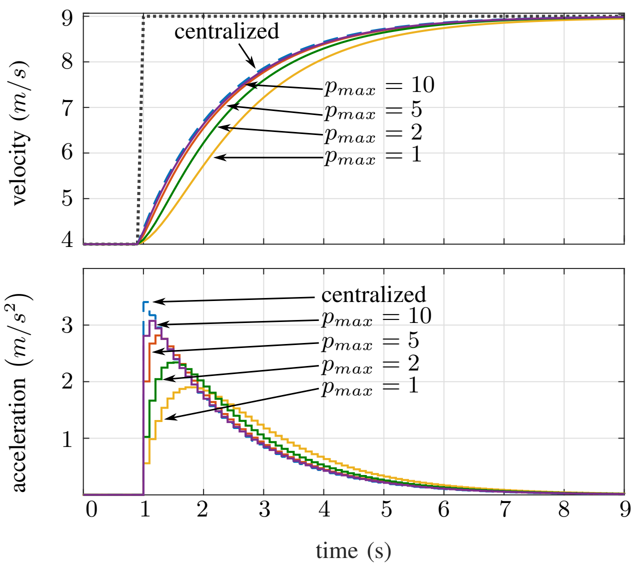

Fig. 3: First vehicle of six simulated vehicles. Varying number of inter-sampling iterations of the proposed emulation method compared to centralized solution (dashed line).

*Remark 4:* In general a convergence to the centralized solution is not guaranteed for a dynamical-decoupled decomposition, as applied in this case. Due to the homogeneous vehicle models, however, the iterations approach to the centralized solution. We use this effect to illustrate the best possible solution of the distributed Jacobi negotiation algorithm.

Now, we simulate the intersection crossing scenario illustrated in Figure 1 for vehicles with a given crossing order order $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6$. Vehicles $\{1, 2, 3\}$ are traveling E $\rightarrow$ W, vehicles $\{4, 5\}$ W $\rightarrow$ N, and vehicle $\{6\}$ S $\rightarrow$ N. We set a constant reference velocity $v_{i,ref} = 7m/s, \forall i \in \mathbb{I}_{1:6}$. Assume a pedestrian becomes visible for vehicle 4 at time $t_1 = 5s$ and crosses the road until $t_2 = 11s$. Vehicle 4 will then incorporate this information by constraining its distance state accordingly and thus plan to full-stop before the cross walk. Consecutive vehicles, which not necessary see the pedestrian, will adjust their trajectories accordingly. Figure 4 illustrates the resulting velocity profile for vehicles $3, 4,$ and 5 (dots) and inter-sampling trajectories at time steps $t = 5s$ and $t = 5.3s$ for $p_{max} = 5$ (solid lines). Each proposed inter-sampling trajectory is a feasible solution of the overall networked control problem. The guaranteed safety distance $d_s$ is kept, as show in Figure 5 which illustrates the relative inter-vehicle distances between vehicles $3-4$, $4-5$, and $5-6$ for the above described scenario.

## V. CONCLUSION

We proposed an iterative MPC-based trajectory negotiation algorithm based on convex Jacobi updates for the coordination of connected and autonomous vehicles. The method

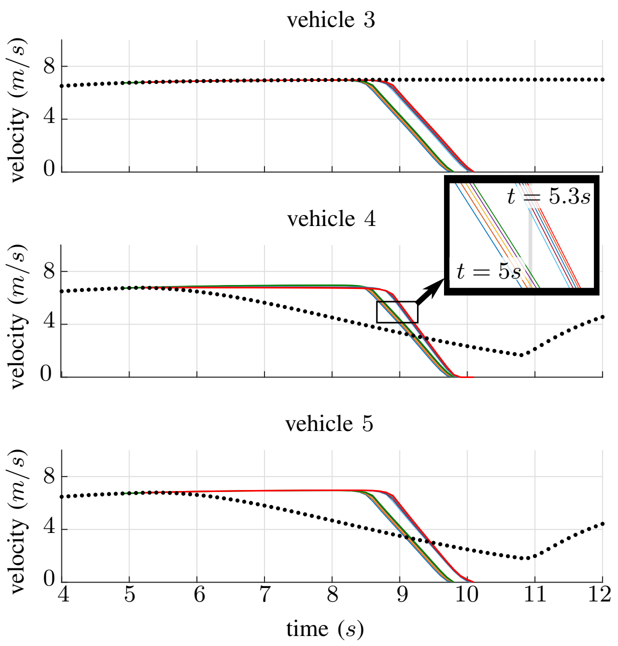

Fig. 4: Intersection crossing scenario with resulting velocity profiles for vehicles $3, 4,$ and $5$ (dots) and predicted trajectories at time steps $t = 5s$ and $t = 5.3s$ (solid lines). Simulated pedestrian crossing before vehicle $4$ at time $t = 5s$.

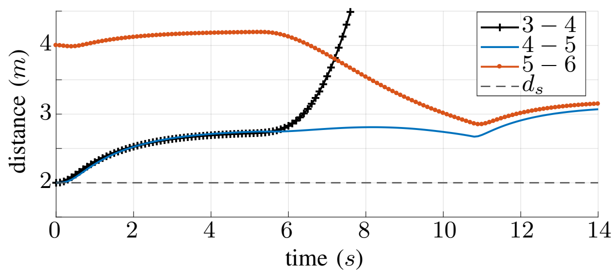

Fig. 5: Relative inter-vehicle distances for the intersection crossing scenario between vehicles $3-4$, $4-5$, and $5-6$, as well as the distance lower bound $d_s$.

is applied to an intersection crossing scenario, where the crossing order has been determined beforehand and vehicles keep a safe distance when they enter a safe-crossing zone. Vehicles are modeled with decoupled dynamical models with coupling inter-vehicle distance constraints. This enables a distributed setup without sharing model data with neither other vehicles nor the infrastructure unit. Computation of the local MPC problems is conducted in parallel while we assume a time-synchronized information exchange between neighboring vehicles. Multiple inter-sampling updates lead to a improved overall coordination result compared to the optimal central solution. However, the iteration process can be stopped after each information exchange with an guaranteed feasible and thus safe solution.

Future work includes the extension of the approach with a scheduling unit enabling re-scheduling decisions in case of uncertain events. Furthermore, the algorithm shall be applied to further traffic scenarios, such as on-ramp merging. Therefore, an integration into a high-fidelity multi-vehicle simulation environment will be conducted to enable close to real-world testing.

## References

[1] Y. Zheng, S. E. Li, K. Li, F. Borrelli, and J. K. Hedrick, "Distributed model predictive control for heterogeneous vehicle platoons under unidirectional topologies," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 3, pp. 899–910, 2017.

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

[3] X. Qian, J. Gregoire, A. De La Fortelle, and F. Moutarde, "Decentralized model predictive control for smooth coordination of automated vehicles at intersection," in *Control Conference (ECC), 2015 European*, pp. 3452–3458, IEEE, 2015.

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

[5] M. Zanon, S. Gros, P. Falcone, and H. Wymeersch, "An asynchronous algorithm for optimal vehicle coordination at traffic intersections," in *20th IFAC World Congress*, 2017.

[6] M. Zanon, R. Hult, S. Gros, and P. Falcone, "A feasibility-enforcing primal-decomposition sqp algorithm for optimal vehicle coordination," *arXiv preprint arXiv:1704.01081*, 2017.

[7] R. Hult, M. Zanon, S. Gros, and P. Falcone, "Optimal coordination of automated vehicles at intersections: Theory and experiments," *IEEE Transactions on Control Systems Technology*, 2018.

[8] A. Katriniok, P. Kleibaum, and M. Joševski, "Distributed model predictive control for intersection automation using a parallelized optimization approach," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 5940–5946, 2017.

[9] B. T. Stewart, A. N. Venkat, J. B. Rawlings, S. J. Wright, and G. Pannocchia, "Cooperative distributed model predictive control," *Systems & Control Letters*, vol. 59, no. 8, pp. 460–469, 2010.

[10] M. D. Doan, M. Diehl, T. Keviczky, and B. De Schutter, "A jacobi decomposition algorithm for distributed convex optimization in distributed model predictive control," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 4905–4911, 2017.

[11] P. Polack, F. Altché, B. d'Andréa Novel, and A. de La Fortelle, "The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 812–818, IEEE, 2017.

[12] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and distributed computation: numerical methods*, vol. 23. Prentice hall Englewood Cliffs, NJ, 1989.

[13] S. Keerthi and E. Gilbert, "Computation of minimum-time feedback control laws for discrete-time systems with state-control constraints," *IEEE Transactions on Automatic Control*, vol. 32, no. 5, pp. 432–435, 1987.

[14] S. Paulraj and P. Sumathi, "A comparative study of redundant constraints identification methods in linear programming problems," *Mathematical Problems in Engineering*, 2010.

[15] H. Berbee, C. Boender, A. R. Ran, C. Scheffer, R. L. Smith, and J. Telgen, "Hit-and-run algorithms for the identification of nonredundant linear inequalities," *Mathematical Programming*, vol. 37, no. 2, pp. 184–207, 1987.