# Control of a city road network: distributed exact verification of traffic safety

Alessandro Colombo, *Member, IEEE,* Gabriel Rodrigues de Campos , Fabio Della Rossa

***Abstract*—A least restrictive supervisor for vehicle collision avoidance is a control algorithm that can detect an unsafe manoeuvre by a set of human-driven or autonomous vehicles, intervening with a corrective action only when needed to avoid a collision. It can help prevent collisions, and facilitate coexistence of autonomous and human-driven vehicles. Such an algorithm is based on a formal verification problem which, unfortunately, is known to be NP-hard in many cases of interest, for instance at traffic intersections. Here we propose a strategy to dynamically decompose the formal verification problem of a large road network, exploiting vehicle dynamics and the constraints induced by road topology to separate non-conflicting vehicles. We split the global problem into smaller and treatable subproblems, while still allowing to compute an exact solution. We illustrate our results on three different scenarios.**

***Index Terms*—Multiagent systems, intelligent transportation systems, distributed control, connected vehicles, capture set, supervisory control, vehicle collision avoidance, intersection control.**

## I. Introduction

Automation is revolutionizing vehicle dynamics. By trading the direct links between driver and actuators for digital systems that modulate driver commands, it can turn even the most clumsy driver into a fairly good chauffeur, or race pilot, depending on parameters setup. The trend has been going for decades, but is gaining momentum now that cheap sensors, embedded computation, and vehicular communication technologies are reaching the market. The natural evolution is to use automatic control to improve a car's behaviour not only in terms of dynamics, but in relation to its surroundings. The first signs of this evolution are on all newest car models, which employ arrays of sensors to detect nearby objects, either to implement emergency braking and safety-distance-keeping or, in some cases, to allow for autonomous driving in sufficiently simple scenarios (such as on a highway). A leap forward in this new trend of automotive technology will take place once vehicle-to-vehicle and vehicle-to-infrastructure communications will become widespread. A connected vehicles environment enables situational awareness beyond what a single vehicle's sensors could ever provide: it can allow more time to plan a car's reaction to an unexpected event or to warn the driver about it, and it provides the means to coordinate the behaviour of multiple cars and improve their collective dynamics [1]–[4].

In the context of automatic control for collision avoidance, the connected vehicles environment is a much needed opportunity to design safety systems that exploit coordination of multiple vehicles. Rear-end collision avoidance is reasonably handled by enforcing a safety distance against the worst-case manoeuvre of each vehicle's predecessor, but the competitive paradigm falls short when it comes to handling junctions and intersections (and therefore in most complex city traffic scenarios). In this context, the coordinated action of multiple vehicles is paramount for an effective safety system. Coordination is not, however, the only hurdle. In the presence of human drivers, a major theoretical challenge is to precisely distinguish a safe, if maybe daring, driving behaviour, from an unsafe one. In a multi-vehicle scenario, and without the help of the many visual hints that allow a human driver to guess the intention of its neighbours, the design of a control system capable of preventing collisions without reacting unnecessarily to normal driving behaviour is a tough problem.

Control theory provides an elegant mathematical framework to address these challenges, in the shape of formal verification and least restrictive supervisor design [5], [6]. A least restrictive supervisor for collision avoidance is a control algorithm that takes as input a command from its user (in our case, a command from the human driver or, in the future, from the autonomous car driving logic), and returns as output the same command, if it is safe, or a corrected one. Most importantly, the correction is applied only when necessary to prevent a collision, thus avoiding by design any unnecessary intervention. The mathematical tool used to distinguish safe and unsafe commands is the formal verification of a mathematical model. Given a model of the system (for instance, of a set of cars), the set of configurations that correspond to collisions is called the *bad set* [7], [8], while the set of configurations from which entering the bad set is unavoidable is called the *capture set* (or the *backward reachable set* of the bad set [9], [10]). A command issued by drivers is *safe*, and therefore accepted by the supervisor, if it keeps the system out of the capture set; when this is not the case, the supervisor must compute a correction that prevents the system from crossing the boundary of the capture set; the problem of distinguishing between the two cases is a verification problem.

Computation of the capture set in a multi-vehicle system was proved to be NP-hard in [11], [12], therefore most verification algorithms are only applicable to fairly small systems (about 2 to 6 vehicles in [8], [12]–[17]) unless approximations are employed [8], [18] or vehicle dynamics is greatly simplified [19]–[22]. In the context of a road network the task of designing a least restrictive supervisor might seem hopeless. Fortunately the problem structure provides a natural way to decompose the task into smaller, treatable subproblems. An intuitive idea is to design controllers that register vehicles approaching an intersection and supervise them until they have left the intersection, delegating collision avoidance away from intersections to a lower-level control. Typically some

All authors are with DEIB, Politecnico di Milano, via Ponzio 34/5, 20133 Milano, Italy. This work is supported by the grant AD14VARI02 - Progetto ERC BETTER CARS - Sottomisura B.

conditions must be required on vehicle speed and spacing as they approach the intersection to ensure nonblockingness of the control algorithm. This is the spirit of the work in [4], [23]–[28].

In this paper we follow a different strategy, and discuss a way to dynamically partition vehicles into groups that need to be jointly supervised, irrespective of their location in the road network. Our main contribution is in discussing an approach to the decomposition of a road-network verification problem into subproblems, such that the exact solution of all subproblems yields an exact solution of the main problem. We show how the results obtained in [8] for a single intersection can be applied, without modifications, to a road network, and more in general prove that the verification approach of [5], [6] can be applied, without any approximation, to large road networks under mild assumptions. As we discuss in Section VII, our strategy has the added benefit of providing a problem decomposition that clusters together only geographically close vehicles, a feature useful for any practical implementation.

Our approach is based on the concept of *guaranteed hull*: this is a time-dependent set that is guaranteed to contain at least one safe trajectory, when one exists. Intuitively, a simple decomposition strategy for a supervisor for collision avoidance would be to decouple sets of vehicles with nonintersecting forward reachable sets. This is essentially the strategy behind most market-ready emergency braking systems. The (finite horizon) forward reachable set of a vehicle is relatively simple to compute, and obviously trajectories of groups of vehicles with nonintersecting reachable sets can be verified independently; however the decoupling strategy is useless when most reachable sets intersect, as is the case in virtually all complex road network scenarios. The guaranteed hull is an improvement on this intuition; it shares the simplicity of computation of the reachable sets, while allowing the designer to define a useful partitioning strategy.

The guaranteed-hull approach stands on the assumption that paths followed by all cars are known. This is a restrictive assumption, but road rules and inference methods [29]–[31] allow to predict the (short term) path of a car, and uncertainty between a small number of possible paths can be handled with moderate additional complexity.

We introduce the problem in Section II, then formalise the concepts of guaranteed hull and independent partition in Sections III. In Section IV we explain how these can be used to define a distributed supervisor. In the remaining sections we provide example implementations of the general idea in three scenarios of increasing complexity.

## II. Problem definition

We consider a road network where human-driven or autonomous vehicles move on a set of intersecting paths; occasional failure in the coordination of some vehicles (human driver distraction or poorly handled interaction between autonomous and nonautonomous vehicles) may cause the issue of unsafe control inputs, which would cause a collision. Each vehicle $i$ is modelled as a point mass with dynamics

$$\ddot{x}_i = f(\dot{x}_i, u_i), \tag{1}$$

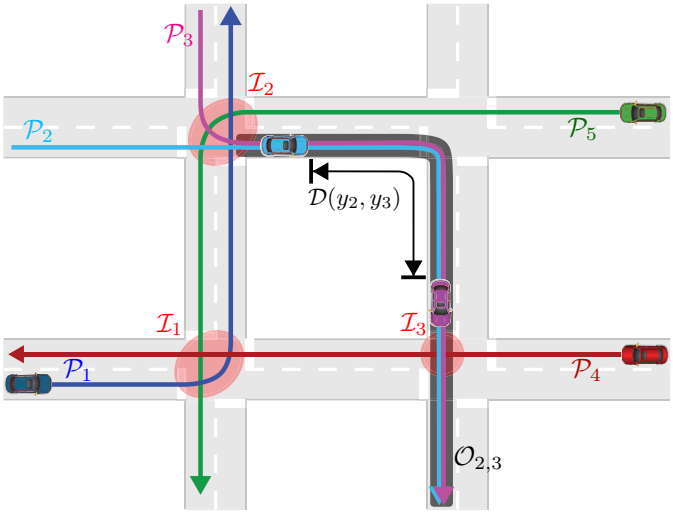


Fig. 1. Five vehicles on intersecting paths $\mathcal{P}_1, \dots, \mathcal{P}_5$. The red regions identify the conflict regions $\mathcal{I}_k$, while the black shaded region identifies an overlapping interval $\mathcal{O}_{i,j}$ between the paths of the cyan and purple vehicle.

where $x_i \in \mathbb{R}$ is the position of the vehicle along its path and $u_i \in \mathbb{R}^m$ is the control input. We call $\mathbf{x}_i := (x_i, \dot{x}_i)$ the state of vehicle $i$, and use the symbols $\mathbf{x}$ and $\mathbf{u}$ (with no subscript) to indicate the aggregate state and input of all vehicles; $\mathbf{x}(t, \mathbf{u})$ denotes the state reached at time $t$ with input $\mathbf{u}$, starting from $\mathbf{x}(0)$. The terminology introduced next is illustrated in Fig. 1. We assume that the road network is planar (an extension to a 3D structure, for instance to allow overpasses, is quite simple), and call

$$\mathcal{P}_i : \mathbb{R} \to \mathbb{R}^2$$

the path followed by vehicle $i$ on the plane. $y_i := \mathcal{P}_i(x_i)$ is the position of vehicle $i$ on the plane, when its arclength coordinate along the path is $x_i$. Vehicles are assumed to always move in the increasing direction of $x$, so that $\dot{x}_i \geq 0$. Vector $\mathbf{y}$ denotes the aggregate positions of all vehicles.

A collision between vehicles may only take place at points where their paths overlap, and may take one of two forms: a rear-end collision, when two paths overlap over an interval and vehicles approach each other below a minimum safe distance; a side collision, when two vehicles traverse simultaneously a conflict region. Call $\mathcal{O}_{i,j} \subset \mathbb{R}^2$ a closed segment of the real line with nonempty interior, embedded in $\mathbb{R}^2$, where paths $\mathcal{P}_i$ and $\mathcal{P}_j$ overlap. Vehicles on overlapping path intervals are assumed to move in the same direction. Given a set of vehicles travelling on a common path interval, we call *first* vehicle the vehicle that drives in front of the set, and *last* vehicle the one that drives in the back. Given an enumeration $1, \dots, n$ of vehicles on a common path interval, we say that they are in *topological order* if 1 is the first vehicle, 2 is the second, and so on. Given an interval $\mathcal{O}_{i,j}$ and two vehicles with $y_i, y_j \in \mathcal{O}_{i,j}$, we call $\mathcal{D}(y_i, y_j) \in \mathbb{R}$ the distance between $y_i \in \mathcal{P}_i$ and $y_j \in \mathcal{P}_j$ measured along the paths. We denote by

$$B_- := \{\mathbf{y} \in \mathbb{R}^{2n} : y_i, y_j \in \mathcal{O}_{i,j}, \mathcal{D}(y_i, y_j) < d\}$$

the subset of $\mathbb{R}^{2n}$ of the rear-end collision points. Call $\mathcal{I}_k$

an open region around the $k$-th intersection in the road network, representing a conflict region, where the simultaneous presence of two vehicles on different paths would constitute a side collision. We denote by

$$B_+ := \left\{\mathbf{y} \in \mathbb{R}^{2n} : y_i, y_j \in \mathcal{I}_k \setminus \mathcal{O}_{i,j}\right\}$$

the subset of $\mathbb{R}^{2n}$ of side-collision points. Finally, we define the bad set

$$B := B_+ \cup B_-.$$

The main objective of this paper is to design a *distributed least restrictive supervisor*, numerically implemented as a discrete-time algorithm with time step $\tau$. The supervisor is a map $S : (\mathbf{x}, \mathbf{v}, \{\mathcal{P}_i\}) \mapsto \mathbf{u}$, from the states $\mathbf{x}_i$ and paths $\mathcal{P}_i$ of all vehicles, and from the set $\mathbf{v}$ of all inputs that the vehicle drivers request (by acting on brake and accelerator), to the set of inputs $\mathbf{u}$ that the cars are allowed to use. The supervisor is called

- **Least restrictive:** if, whenever $\mathbf{u} \neq \mathbf{v}$, returning $\mathbf{u} = \mathbf{v}$ would inevitably result in a future collision.
- **Distributed:** if it can be defined as a set of algorithms running simultaneously on different processors, with minimal exchange of information between processors.

At time $t$, a partitioning algorithm (distributed among cars) takes care of forming vehicle clusters that can be supervised separately. The state reached at $t + \tau$ with input $\mathbf{u} = \mathbf{v}$ is predicted using a mathematical model. Then, a verification algorithm (distributed among clusters) checks whether the predicted state admits a safe future evolution or not; in case of a negative answer, an override input is used instead of the user-requested one for all vehicles in the cluster that failed verification. The design of such a supervisor requires to solve the following *verification problem*:

**Verification Problem – VP** *Given a set of paths $\{\mathcal{P}_i\}$ and initial conditions $\mathbf{x}$, determine if there exists an input signal $\mathbf{u}$ that guarantees that $\mathbf{y}(t, \mathbf{u}) \notin B$ for all $t \geq 0$.*

We write $\mathbf{x} \in \mathrm{VP}$ if VP has a positive answer for initial conditions $\mathbf{x}$.

Design of least-restrictive supervisors for road intersection scenarios was treated for instance in [8], [12], [14], [15], [20], [22], [32]–[35]. While most of the cited results can be used in the verification step, focus of this paper is on how to construct a partition of vehicles to ensure that exact verification on the clusters yields exact verification of the whole network, without the need to exchange information between different clusters. To formalise this, call $\kappa := \{C_1, \ldots, C_c\}$ a partition of the set of vehicles into the clusters $C_1, \ldots, C_c$. We denote with a subscript $C$ the restriction of a vector or set to the vehicles in cluster $C$; for instance $\mathbf{x}_C$ is the aggregate state of all vehicles in cluster $C$, while $\mathrm{VP}_C$ is VP restricted to vehicles in $C$.

**Definition 1.** *A partition $\kappa := \{C_1, \ldots, C_c\}$ is independent if*

$$\Big(\mathbf{x}_C \in \mathrm{VP}_C, \forall\, C \in \kappa\Big) \Leftrightarrow \Big(\mathbf{x} \in \mathrm{VP}\Big)$$

**Definition 2.** *A partition $\kappa := \{C_1, \ldots, C_c\}$ is time-$\tau$ independent if, for all $t \in [0, \tau]$ and for all $\mathbf{u} \in \mathcal{U}$,*

$$\Big(\mathbf{x}_C(t, \mathbf{u}) \in \mathrm{VP}_C, \forall\, C \in \kappa\Big) \Leftrightarrow \Big(\mathbf{x}(t, \mathbf{u}) \in \mathrm{VP}\Big)$$

In other words, a time-$\tau$ independent partition is a partition which is guaranteed to remain independent for a time $\tau$, for any possible evolution of the state $\mathbf{x}$. Even though it is sufficient to obtain a (time-0) independent partition to decompose VP, it is necessary to obtain a time-$\tau$ independent partition to ensure that a decomposition computed at time $t$ is still independent at time $t + \tau$. This is necessary to implement the discrete-time supervisor mentioned before, which we introduce in Sec. IV.

Given a control signal $\mathbf{u}$ defined over the time interval $[0, \infty)$, we call $\mathbf{u}_{[t_1,t_2]}$ the restriction of $\mathbf{u}$ to the time interval $[t_1, t_2]$. We use the symbol

$$\underline{\left|\mathbf{u}_{[t_1,t_2)}, \mathbf{u}_{[t_2,t_3)}, \ldots\right|}$$

to write the concatenation of multiple input signals defined over nonoverlapping time intervals. We use the elementwise inequality sign for vectors: $\mathbf{x} \geq \mathbf{x}'$ means that each element of $\mathbf{x}$ is greater than or equal to the corresponding element of $\mathbf{x}'$. The same notation is used for $\mathbf{u}$. We call

$$\mathcal{U}_s(\mathbf{x}) := \{\mathbf{u} \in \mathcal{U} : \mathbf{y}(t, \mathbf{u}) \notin B,\, \forall\, t \geq 0\}, \tag{2}$$

where $\mathbf{y}(t, \mathbf{u}) := \mathcal{P}(\mathbf{x}(t, \mathbf{u}))$, the set of *safe inputs*, that is, all control choices which do not cause a collision when vehicles have initial state $\mathbf{x}$. Similarly,

$$\mathcal{U}_s(\mathbf{x}_C) := \{\mathbf{u}_C \in \mathcal{U}_C : \mathbf{y}_C(t, \mathbf{u}_C) \notin B_C,\, \forall\, t \geq 0\} \tag{3}$$

is the set of safe inputs for vehicles in cluster $C$ with initial state $\mathbf{x}_C$.

## III. Partitioning algorithm and distributed network supervisor

The partitioning algorithm that we are about to define is based on the concept of guaranteed hull. Consider a set-valued function $I_i(t) : \mathbb{R}_+ \to 2^{\mathbb{R}^2}$, which attaches to a time $t$ a subset of $\mathbb{R}^2$. Take the vector $I_C(t) := \{I_i(t)\}$, $i \in C$.

**Definition 3.** *$I_C(t)$ is a guaranteed hull for cluster $C$ if, whenever $\mathcal{U}_s(\mathbf{x}_C) \neq \emptyset$, there exists at least one $\mathbf{u}_C \in \mathcal{U}_s(\mathbf{x}_C)$ such that $y_i(t, u_i) \in I_i(t)$, for all $t \geq 0$ and for all $i \in C$.*

In simple terms, a guaranteed hull is a time-dependent set which, when $\mathcal{U}_s(\mathbf{x}_C)$ is nonempty, contains at least one safe trajectory $\mathbf{y}_C(t, \mathbf{u}_C)$, $\mathbf{u}_C \in \mathcal{U}_s(\mathbf{x}_C)$. A concrete definition of guaranteed hull requires further assumptions (e.g., vehicle model, geometry of paths). For fully specified examples we refer the reader to equation (7) in Section V, and its subsequent redefinitions in Sections VI and VII.

Algorithm 1 computes a partition of the vehicles using guaranteed hulls. In the algorithm, we use the guaranteed hull $I_{\{i,j\}}$ for pairs $\{i, j\}$ of vehicles, and the bad set $B_{\{i,j\}}$ relative to a cluster $C := \{i, j\}$.

**Theorem 1.** *Algorithm 1 terminates and finds an independent partition.*

*Proof:* Part 1: the algorithm terminates.
At each iteration of the algorithm, either the number of clusters in $\kappa$ remains the same (in which case the algorithm returns the partition $\kappa$), or it is reduced. Since the number of clusters is

**Algorithm 1** Partitioning algorithm

1: **procedure** *Partition*($\mathbf{x}$)
2: Define a partition $\kappa$ where all vehicles are singleton clusters
3: Tag all clusters *not done*
4: **while** there exists a cluster *not done* **do**
5: Compute $I_C$ for all $C \in \kappa$
6: Form a new partition $\kappa'$ by merging all $C_a, C_b \in \kappa$ such that $\exists\, t, i \in C_a, j \in C_b : I_{\{i,j\}}(t) \cap B_{\{i,j\}} \neq \emptyset$
7: Tag as *done* all clusters that were not merged (i.e. those that were not modified between $\kappa$ and $\kappa'$)
8: Set $\kappa = \kappa'$
9: **return** $\kappa$

finite it must eventually decrease to one. Once a single cluster is left the algorithm terminates.

Part 2: $\kappa$ is independent

$\big(\mathbf{x} \in \mathrm{VP}\big) \Rightarrow \big(\mathbf{x}_C \in \mathrm{VP}_C, \forall C\big)$ is trivial. To prove $\big(\mathbf{x} \in \mathrm{VP}\big) \Leftarrow \big(\mathbf{x}_C \in \mathrm{VP}_C, \forall C\big)$ we proceed as follows. First, note that $\mathbf{x}_C \in \mathrm{VP}_C$ equals the condition $\mathcal{U}_s(\mathbf{x}_C) \neq \emptyset$. Given $\mathcal{U}_s(\mathbf{x}_C) \neq \emptyset$ and since all $I_C$ are guaranteed hulls, by Definition 3 for all $C \in \kappa$ there exists a trajectory $\mathbf{y}_C(t, \tilde{\mathbf{u}}_C)$, with $\mathbf{y}_C(t, \tilde{\mathbf{u}}_C) \in I_C(t)$, such that $\mathbf{y}_C(t, \tilde{\mathbf{u}}_C) \notin B_C$. This and Line 6 of Algorithm 1 imply that

$$\begin{bmatrix} \mathbf{y}_{C_a}(t, \tilde{\mathbf{u}}_{C_a}) \\ \mathbf{y}_{C_b}(t, \tilde{\mathbf{u}}_{C_b}) \end{bmatrix} \cap B_{C_a \cup C_b} = \emptyset, \forall\, t \geq 0,$$

for any pair $C_a$ and $C_b$ of clusters. This implies $\mathbf{y}(t, \tilde{\mathbf{u}}) \notin B$, for all $t \geq 0$, and therefore $\mathbf{x} \in \mathrm{VP}$. ∎

With $n$ cars, the above algorithm terminates in at most $n$ steps. If Line 6 is executed by each car, which compares its guaranteed hull with those of the cars of at most $n-1$ other clusters, the algorithm complexity is $O(n^2)$. If guaranteed hulls are bounded for all $t \geq 0$, complexity can be reduced further, as we see next. Call $\|\cdot\|$ the Euclidean norm.

**Hypothesis 1.** *There exists a finite value $L$ such that*

$$\forall\, i, \forall\, t \geq 0, \text{ and } \forall\, y_i \in I_i(t), \|y_i - y_i(0)\| \leq L.$$

A necessary condition for the above hypothesis is that vehicles can be safely stopped in finite distance, i.e., when $\mathcal{U}_s(\mathbf{x}_C) \neq \emptyset$, there exist $\bar{\mathbf{u}}_C \in \mathcal{U}_s(\mathbf{x}_C)$ such that $\lim_{t\to\infty} x_i(t, \bar{u}_i) - x_i(0) \leq L,\ \forall\, i \in C$.

**Definition 4.** *Given two clusters $C_1$ and $C_2$, call*

$$\mathrm{dist}(C_1, C_2) := \min_{\substack{i \in C_1 \\ j \in C_2}} \|y_i - y_j\|.$$

*Moreover, denote $|\mathcal{I}_k|$ the diameter of $\mathcal{I}_k$, that is,*

$$|\mathcal{I}_k| := \sup_{y_i, y_j \in \mathcal{I}_k} \|y_i - y_j\|.$$

**Remark 1.** *Assume Hypothesis 1, and that the vehicles of two clusters $C_1$ and $C_2$ are such that $\mathrm{dist}(C_1, C_2) > 2L + \max\{d, \max_k |\mathcal{I}_k|\}$. Then at Line 6 of Algorithm 1, $C_1$ and $C_2$ need not be compared.*

*Proof:* The assumptions $\mathrm{dist}(C_1, C_2) > 2L + \max\{d, \max_k |\mathcal{I}_k|\}$ and Hypothesis 1 imply that, for all $y_i(t) \in I_i(t)$ and $y_j(t) \in I_j(t)$, with $i \in C_1, j \in C_2$,

$$\|y_i(t) - y_j(t)\| \geq \max\{d, \max_k |\mathcal{I}_k|\},\ \forall\, t \geq 0.$$

This means that, for all $i \in C_1$ and $j \in C_2$, $I_{\{i,j\}}(t) \cap B_{\{i,j\}} = \emptyset,\ \forall\, t \geq 0$. ∎

The above remark states that, when guaranteed hulls are bounded (Hypothesis 1), Line 6 of Algorithm 1 can be executed by only comparing cars in nearby clusters. Using the same reasoning as in the above proof, we can give sufficient conditions for Algorithm 1 to converge to a nontrivial partition.

**Remark 2.** *Take cluster $C_1$ and its complement $C_2$, defined to contain all vehicles not in $C_1$. Assume that $C_1$ and $C_2$ satisfy Remark 1. Then Algorithm 1 will not merge $C_1$ with any other cluster.*

This formalises the intuition that, if a set of cars is sufficiently far from all other cars, Algorithm 1 leaves it as a separate cluster. Note that the assumption on $\mathrm{dist}(C_1, C_2)$ is restrictive. In the following, with additional information on vehicles dynamics and paths configuration, we will be able to formulate less restrictive conditions.

The following definition extends the properties of the guaranteed hull defined before, to hold for any state reachable by the system in a time $\tau$.

**Definition 5.** *$I_C^\tau(t)$ is a time-$\tau$ guaranteed hull for cluster $C$ if, whenever $\mathcal{U}_s(\mathbf{x}_C) \neq \emptyset$, for all $\mathbf{u}_{C,[0,\tau]} \in \mathcal{U}_s(\mathbf{x}_C)$ there exists at least one $\mathbf{u}_{C,(\tau,\infty)} \in \mathcal{U}_s(\mathbf{x}_C(\tau, \mathbf{u}_{C,[0,\tau]}))$ such that $y_i(t, u_i) \in I_i^\tau(t)$, for all $t \geq 0$ and for all $i \in C$.*

A time-$\tau$ guaranteed hull is thus a guaranteed hull for any element of the safe reachable set

$$\big\{\mathbf{y}_C(t, \mathbf{u}_C) : \mathbf{u}_C \in \mathcal{U}_s(\mathbf{x}_C), t \in [0, \tau]\big\}.$$

Note that, as expected, the above definition of $I_C^\tau$ coincides with the one in Definition 3 if $\tau = 0$.

**Theorem 2.** *The partition $\kappa$ computed using Algorithm 1 and $I_C^\tau$ in stead of $I_C$ is time-$\tau$ independent.*

*Proof:* The proof is similar to Part 2 of the proof of Theorem 1. ∎

In the rest of the paper we assume that Algorithm 1 is set up to compute a time-$\tau$ independent partition.

## IV. Supervisor synthesis

Using the machinery introduced in the previous sections, we can now define a distributed supervisor for a road network. The supervisor must obviously possess the two basic properties of *correctness* and *nonblockingness*, defined as follows.

**Definition 6.** *We say that a supervisor $S : (\mathbf{x}, \mathbf{v}, \{\mathcal{P}_i\}) \mapsto \mathbf{u}$ is correct if $\mathbf{y}(t, S(\mathbf{x}(0), \mathbf{v}(0), \{\mathcal{P}_i\})) \notin B$ for all $t \in [0, \tau]$, nonblocking if $S\big(\mathbf{x}(\tau, S(\mathbf{x}(0), \mathbf{v}(0), \{\mathcal{P}_i\}), \mathbf{v}(\tau), \{\mathcal{P}_i\})\big) \neq \emptyset$.*

We begin by defining, in Algorithm 2, a supervisor for each one of the clusters obtained with Algorithm 1. Then, in Algorithm 3 we show how the partitioning algorithm (Algorithm 1) and the cluster supervisor (Algorithm 2) can be used to jointly supervise the network.

In Algorithm 2, $\mathbf{v}_C$ is the vector of desired inputs issued by the drivers of vehicles in cluster $C$ at each discrete time

**Algorithm 2** Supervisor of a cluster

1: **procedure** *ClusterSupervisor*($\mathbf{x}_C, \mathbf{v}_C$)
2: $\quad\bar{\mathbf{u}}_C(t) \leftarrow \mathbf{v}_C \,\forall\, t \in [0,\tau]$
3: $\quad\mathbf{x}_{C,\text{next}} \leftarrow \mathbf{x}_C(\tau, \bar{\mathbf{u}}_C, \mathbf{x}_C)$
4: $\quad$**if** $\mathbf{x}_{C,\text{next}} \in \text{VP}_C$ and $\mathbf{y}_C(t, \bar{\mathbf{u}}_C) \notin B_C$ for all $t \in [k\tau, (k+1)\tau]$ **then**
5: $\quad\quad$**return** $\bar{\mathbf{u}}_C$
6: $\quad$**else**
7: $\quad\quad$**return** $\mathbf{u}_{C,\text{safe}}$

step, while $\mathbf{u}_{C,\text{safe}}$ is an input in $\mathcal{U}_s(\mathbf{x}_C(k\tau))$, used by the supervisor to override the desired input $\mathbf{v}_C$ when necessary. Computation of $\mathbf{u}_{C,\text{safe}}$ is not discussed here, but numerically efficient methods to compute a safe input for different traffic scenarios are found in the literature [8], [16], [27], [38]. The algorithm assumes synchronous communication from all vehicles; note however that asynchronous communication can be handled quite simply, e.g., as explained in [36].

**Theorem 3.** *Assume $\mathbf{x}_C(0) \in$VP$_C$. Then, the cluster supervisor in Algorithm 2 is correct.*

*Proof:* If the supervisor returns $\mathbf{u}_{C,\text{safe}}$, correctness is guaranteed by definition. If it returns $\bar{\mathbf{u}}_C$, correctness is explicitly checked at Line 4 by the condition $\mathbf{y}_C(t, \bar{\mathbf{u}}_C) \notin B_C$ for all $t \in [k\tau, (k+1)\tau]$. ∎

**Algorithm 3** Supervisor of the network

1: **procedure** *NetworkSupervisor*($\mathbf{x}, \mathbf{v}$)
2: $\quad$**while** True **do**
3: $\quad\quad\kappa = Partition(\mathbf{x})$ ▷ defined in Algorithm 1
4: $\quad\quad$**for all** $C \in \kappa$ **do**
5: $\quad\quad\quad\mathbf{u}_C = ClusterSupervisor(\mathbf{x}_C, \mathbf{v}_C)$ ▷ defined in Algorithm 2

**Theorem 4.** *The network supervisor in Algorithm 3 is correct and nonblocking*

*Proof:* Correctness: Let the network supervisor run with time stepping $\tau$. By Theorem 3, the cluster supervisor in Algorithm 2 is correct, that is, it ensures that $\mathbf{x}_C(t, \mathbf{u}_C) \in \text{VP}_C$ for all $t \in [0,\tau]$. By Theorem 2 and Definition 2 this implies that $\mathbf{x}(t, \mathbf{u}) \in$ VP for all $t \in [0,\tau]$, and therefore that collisions are avoided for $t \in [0,\tau]$. The network supervisor in Algorithm 3 is thus correct.

Nonblockingness: Consider $\mathbf{x}(\tau, \mathbf{u})$ reached from $\mathbf{x}(0)$ in a time step. Algorithm 3 is blocking if, at time $\tau$, line 5 cannot be evaluated, which means that Algorithm 2 cannot compute its output. We have $\mathbf{x}(\tau, \mathbf{u}) \in$ VP. By Theorem 2 and Definition 2 this insures that, whatever partition $\kappa$ is computed at time $\tau$, for all $C \in \kappa$, $\mathbf{x}_C(\tau, \mathbf{u}) \in$VP$_C$. Because of this, the output of Algorithm 2 at time $\tau$ can always be computed ($\mathbf{u}_C$ exists), so Algorithm 3 is nonblocking. ∎

The above results define a least restrictive supervisor for the whole network which only requires computation on a partition of the network. This is an important theoretical result: it shows that the exact verification of a large road network can be recast in the simpler problem of independently verifying a number of vehicle clusters. The complexity of the smaller problems depends on the number and configuration of vehicles in each cluster. The coming sections show that, in many cases, the resulting problems are simple enough for an exact and real-time solution with existing computational methods. The sections are organised so as to discuss scenarios of increasing complexity. Exact and approximate algorithms to test $\mathbf{x}_{C,\text{next}} \in \text{VP}_C$ at Line 3 of Algorithm 2 and to synthesize $\mathbf{u}_{C,\text{safe}}$ at Line 7 are discussed, for different road scenarios and number of vehicles, in [8], [12], [37], [38] among others. The simulations at the end of each section use the results of [8] for this task.

## V. Scenario 1: vehicles on a lane

We now apply the ideas described above to a simple scenario where a set of vehicles move on a single path. The results we obtain for this scenario are propaedeutic to the more complex road network scenarios of the two following sections.

We assume all vehicles have the same dynamics: to simplify notation we drop the index $i$ when not needed.

**Scenario 1.** *A set of $N$ vehicles, with identical dynamics, drive on the same path: $\mathcal{P}_i = \mathcal{P}_j = \mathcal{O}_{i,j} = \mathcal{P}$, $\forall\, i, j$.*

We make the following assumptions.

(A.1) there exists two elements $\mathbf{u}_{\min}$ and $\mathbf{u}_{\max}$ of $\mathcal{U}$ such that $\mathbf{u}_{\min} \le \mathbf{u}(t)$ and $\mathbf{u}_{\max} \ge \mathbf{u}(t)$ for all $t$ and for all $\mathbf{u} \in \mathcal{U}$, and $f(\dot{x}_i, u_i)$ is non-decreasing in $u_i$.

(A.2) system (1) has unique solutions, depending continuously on initial conditions and parameters.

(A.3) $\dot{x}_i$ is bounded to the interval $[0, \dot{x}_{\max}]$, with $\dot{x}_{\max} > 0$.

(A.4) $|f(\dot{x}_i, u_i)|$ is bounded.

(A.5)
$$\begin{aligned}\lim_{t\to\infty} \dot{x}_i(t, u_{\max}) &= \dot{x}_{\max},\\ \lim_{t\to\infty} \dot{x}_i(t, u_{\min}) &= 0\end{aligned} \tag{4}$$
(min and max velocities are attained at least asymptotically by applying $u_{\min}$ and $u_{\max}$).

As shown in [39], (A.1) implies
$$\begin{gathered}\mathbf{x}_i(0) \ge \mathbf{x}'_i(0), u_i(t) \ge u'_i(t)\,\forall\, t \ge 0\\ \Downarrow\\ \mathbf{x}_i(t) \ge \mathbf{x}'_i(t)\,\forall\, t \ge 0.\end{gathered} \tag{5}$$
and
$$\begin{gathered}\mathbf{x}_i(0) > \mathbf{x}'_i(0) \text{ and } u_i(t) \ge u'_i(t)\,\forall\, t \ge 0\\ \Downarrow\\ \mathbf{x}_i(t) > \mathbf{x}'_i(t)\,\forall\, t \ge 0\end{gathered} \tag{6}$$
(recall that $\mathbf{x}_i := (x_i, \dot{x}_i)$).

We can now construct the set $I_C^\tau$ needed in Algorithm 1. We begin by computing the minimum worst-case stopping distance of a vehicle:

**Definition 7.** $D_{\text{stop}} := \lim_{t\to\infty} x_i(t, u_{\min})$ *with* $x_i(0) = 0$, $\dot{x}_i(0) = \dot{x}_{\max}$.

Next, we define the worst-case difference between the stopping distance when a braking manoeuvre is started at time 0, and when it is started at time $\tau$:

**Definition 8.**
$$\Delta_{i,\text{stop}}(\tau) := \max_{\dot{x}_i(0)\in[0,\dot{x}_{\max}]} \lim_{t\to\infty} x_i(t, \underbrace{\lfloor u_{\max,[0,\tau]}, u_{\min,(\tau,\infty)} \rfloor}) - x_i(t, u_{\min}).$$

*and*

$$\Delta_{\text{stop}}(\tau) := \max_i \Delta_{i,\text{stop}}(\tau).$$

For sake of simplicity, in the following we drop the argument $\tau$ of $\Delta_{\text{stop}}$. The quantity $\Delta_{\text{stop}}$ is an upper bound to the difference in the stopping position of a vehicle applying an arbitrary input for $t \in [0, \tau]$ and then applying $u_{min}$ for $t > \tau$, or applying $u_{min}$ for all $t \geq 0$. The sum of $\Delta_{\text{stop}}$ and $D_{\text{stop}}$ provides an upper bound to the stopping distance of a vehicle that is allowed to use an arbitrary input for a time interval $[0, \tau]$, as shown in the following remark.

**Remark 3.** *Let the signal $u_i$ assume any value in the time interval $[0, \tau]$ and be equal to $u_{min}$ for $t \geq \tau$:*

$$u_i := \underline{\left| u_{i,[0,\tau]}, u_{\min,(\tau,\infty)} \right|}.$$

*Then*

$$\lim_{t\to\infty} x_i(t, u_i) \leq x_i(0) + D_{\text{stop}} + \Delta_{\text{stop}}.$$

*Proof:* Using Definition 7, the fact that the vector field (1) is invariant to translation in the position $x$, and (5), we have

$$\lim_{t\to\infty} x_i(t, u_{\min}) \leq D_{\text{stop}} + x_i(0)$$

while by Definition 8

$$\lim_{t\to\infty} x_i(t, \underline{\left| u_{i,[0,\tau]}, u_{\min,(\tau,\infty)} \right|}) - x_i(t, u_{\min}) \leq \Delta_{\text{stop}}.$$

Summing the two we obtain

$$\lim_{t\to\infty} x_i(t, u_i) \leq x_i(0) + D_{\text{stop}} + \Delta_{\text{stop}}.$$

■

We now have all ingredients to define a guaranteed hull for Scenario 1. Take a cluster $C$ with vehicles numbered in topological order, so that $x_1 > \ldots > x_n$. Let $\mathcal{P}\big([\cdot,\cdot]\big)$ be the set-valued image of $\mathcal{P}(x_i)$ applied to all $x_i$ in the interval $[\cdot,\cdot]$. Define

$$I_i^\tau(t) := \mathcal{P}\big([x_i(t, u_{\min}), x_i(t, \bar{u}_i)]\big), \tag{7}$$

where $\bar{u}_i$ is a bang-bang input

$$\bar{u}_i(t) := \underline{\left| u_{\max,[0,t_i^*]}, u_{\min,(t_i^*,\infty)} \right|} \tag{8}$$

switching at a time $t_i^* \geq 0$ so as to satisfy

$$t_n^* := \tau$$

and

$$t_i^* := \max\left\{\tau, t_i^* \text{ s.t. } \lim_{t\to\infty} x_i(t, \bar{u}_i) = \lim_{t\to\infty} x_{i+1}(t, \bar{u}_{i+1}) + d\right\}.$$

The guaranteed hull $I_C^\tau$ can be constructed as the Cartesian product of $I_i^\tau$ defined above, for all $i \in C$.

**Theorem 5.** *$I_C^\tau$ defined above satisfies the property in Definition 5 for Scenario 1.*

The above theorem, proven in the Appendix, states that $I_C^\tau$ can indeed be used to partition vehicles using Algorithm 1 in this scenario.

Building on Remark 3, we can provide an upper bound to the distance $\lim_{t\to\infty} x_i(t, \bar{u}_i) - x_i(0)$ travelled by an arbitrary member of a cluster within the guaranteed hull.

**Lemma 6.** *Consider a cluster $C$ with vehicles on a single path $\mathcal{P}$. Name the vehicles $1, \ldots, n$, in topological order. If $I_C^\tau(t)$ is defined as in (7)–(8) and $C$ is an element of a partition formed using Algorithm 1, then*

$$\lim_{t\to\infty} x_i(t, \bar{u}_i) \leq x_i(0) + D_{\text{stop}} + \Delta_{\text{stop}}, \; \forall\, i.$$

*Proof:* The proof is by induction. $\bar{\mathbf{u}}$ is defined in (8), and for vehicle $n$, $t_n^* = \tau$ and

$$\bar{u}_n := \underline{\left| u_{\max,[0,\tau]}, u_{\min,(\tau,\infty)} \right|}.$$

By Remark 3,

$$\lim_{t\to\infty} x_n(t, \bar{u}_n) \leq x_n(0) + D_{\text{stop}} + \Delta_{\text{stop}}. \tag{9}$$

For vehicle $n-1$, using (8) we have

$$\lim_{t\to\infty} x_{n-1}(t, \bar{u}_{n-1}) \leq \max\Big\{ \lim_{t\to\infty} x_{n-1}(t, \underline{\left| u_{\max,[0,\tau]}, u_{\min,(\tau,\infty)} \right|}), \lim_{t\to\infty} x_n(t, \bar{u}_n) + d\Big\}.$$

Again by Remark 3 we have

$$\lim_{t\to\infty} x_{n-1}(t, \underline{\left| u_{\max,[0,\tau]}, u_{\min,(\tau,\infty)} \right|}) \leq x_{n-1}(0) + D_{\text{stop}} + \Delta_{\text{stop}},$$

while using (9) and the fact that, having $\mathcal{U}_s(\mathbf{x}(0)) \neq \emptyset$, $x_n(0) + d \leq x_{n-1}(0)$, we obtain

$$\lim_{t\to\infty} x_n(t, \bar{u}_n) + d \leq x_{n-1}(0) + D_{\text{stop}} + \Delta_{\text{stop}}.$$

The two above inequalities give

$$\lim_{t\to\infty} x_{n-1}(t, \bar{u}_{n-1}) \leq x_{n-1}(0) + D_{\text{stop}} + \Delta_{\text{stop}}.$$

Iterating the above reasoning we complete the proof. ■

A consequence of this lemma is that, if a safe input exists, then there exists one that stops all vehicles in at most $D_{\text{stop}} + \Delta_{\text{stop}}$. Using Lemma 6 we can improve the condition of Lemma 2 to ensure a nontrivial partition. The improved sufficient condition is schematized in Fig. 2 and described in the following corollary.

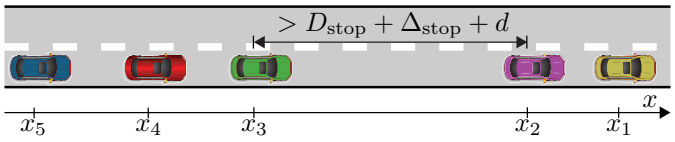


Fig. 2. A sufficient condition for the two sets of vehicles $C_1 = \{1, 2\}$ and $C_2 = \{3, 4, 5\}$ to be clustered separately is that $x_2 - x_3 \geq D_{\text{stop}} + \Delta_{\text{stop}} + d$

**Corollary 7.** *Consider two clusters $C_1$ and $C_2$ on a single path and assume that $x_i(0) > x_j(0)$, $\forall\, i \in C_1, j \in C_2$. Name $n+m, \ldots, n+1$ the vehicles of $C_2$, and $n, \ldots, 1$ those of $C_1$, in topological order. If $I_C^\tau(t)$ is defined as in (7)–(8), a sufficient condition for Algorithm 1 to return a nontrivial partition is that $x_n(0) - x_{n+1}(0) > D_{\text{stop}} + \Delta_{\text{stop}} + d$.*

*Proof:* By Lemma 6, if $x_n(0) - x_{n+1}(0) > D_{\text{stop}} +$

$\Delta_{\text{stop}} + d$, then $x_{n+1}(t, \bar{u}_{n+1}) + d \leq x_n(0)$, for all $t \geq 0$. Since $n$ is the last vehicle of $C_1$ and $n+1$ is the first vehicle of $C_2$,

$$I^{\tau}_{i,j}(t) \cap B_{i,j} = \emptyset, \forall\, t \geq 0, \forall\, i \in C_1, j \in C_2,$$

and Algorithm 1 will not merge the two clusters. ■

According to the above corollary, at Line 6 of Algorithm 1 vehicles of clusters father apart than $D_{\text{stop}} + \Delta_{\text{stop}} + d$ need not communicate.

### A. Simulation of Scenario 1

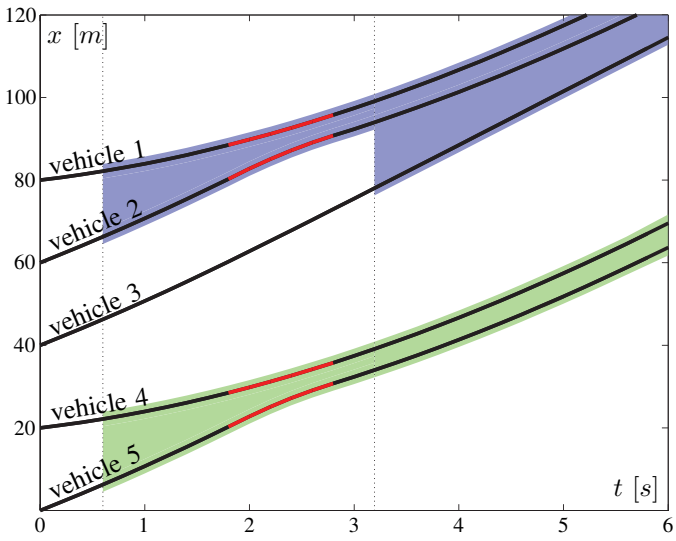


Fig. 3. Simulation of Scenario 1 with $\mathbf{x}(0) = ((80,3),(60,10),(40,10),(20,3),(0,10))$. Trajectories of vehicles in the same cluster are envelopped in background of the same color. Trajectories are red when the supervisor is overriding the driver input, black otherwise. Dotted vertical lines mark the times when the the partition structure changes.

We show in Fig. 3 a simulation of the current scenario, with vehicle longitudinal dynamics modelled as

$$\ddot{x}_i = u - 0.0005\dot{x}_i^2,$$

where the quadratic term accounts for air drag, $\tau = 0.2\ s$, $u_i \in [-5, 2]$, $\dot{x}_{max} = 13.9\ m/s$, $d = 5\ m$. The chosen parameters give $D_{\text{stop}} = 17.66\ m$, $\Delta_{\text{stop}} = 2.78\ m$. We assumed in the simulation that all drivers attempt to maximise their speed, by requesting an input $u_i$ proportional to $\dot{x}_{i,\max} - \dot{x}_i$. At time 0 vehicles are far apart, and Algorithm 1 places them in separate clusters. Around $t = 0.6$ we have

$$\lim_{t\to\infty} x_2(t, \underline{\left|u_{\max,[0,\tau]}, u_{\min,(\tau,\infty)}\right|}) > \lim_{t\to\infty} x_1(t, u_{\min}),$$

that is, vehicle 2 approaches vehicle 1 and the two are clustered together. The same happens for vehicles 4 and 5. Around $t = 3.2$ vehicle 3 joins the cluster of 1 and 2. Note that the supervisor overrides the inputs of vehicles in each of the clusters only for a short time between $t = 1.8$ and 2.8. Override intervals of the two clusters coincide due to symmetry in the initial conditions and inputs of the respective vehicles.

## VI. Scenario 2: single isolated intersection

Let us now consider a more complex scenario where a set of paths intersect at a single point and multiple vehicles may travel on each of the paths.

**Scenario 2.** *A set of $N$ vehicles drive on $M$ different paths, which intersect at a single isolated point, that is, for all vehicles $i$ and $j$,*

$$\mathcal{P}_i = \mathcal{P}_j = \mathcal{O}_{i,j}$$

*or*

$$\mathcal{P}_i \cap \mathcal{P}_j \neq \emptyset \text{ and } \mathcal{P}_i \cap \mathcal{P}_j \in \mathcal{I}.$$

*We assume that vehicles on the same path have identical dynamics.*

For each path $\mathcal{P}_h$, let us call $(a_h, b_h)$ the interval $\{x : P_h(x) \in \mathcal{I}\}$, and assume that $b_h - a_h > d$, for all $h$ (i.e., the width of the intersection along each path is greater than the minimum safety distance between two vehicles on the same path). We keep the formal definition of $I^{\tau}_C$ used in the previous section, as the Cartesian product of

$$I^{\tau}_i(t) := \mathcal{P}_h\big([x_i(t, u_{\min}), x_i(t, \bar{u}_i)]\big)$$

but we define $\bar{u}_i$ differently depending on whether the vehicles of a cluster are near an intersection or not, in a sense that is explained next.

Assume that cluster $C$ has $n$ vehicles on path $\mathcal{P}_h$, number them in topological order so that 1 is the first and $n$ is the last of these vehicles. We must consider two cases:

(i) $[x_n(0), x_1(0) + D_{\text{stop}} + \Delta_{\text{stop}}] \cap (a_h, b_h) = \emptyset$, i.e., vehicles in cluster $C$ can safely stop before entering the intersection, regardless of their current velocity and of their input in the time interval $[0, \tau]$ (see Lemma 6), or they are all past the intersection;

(ii) $[x_n(0), x_1(0) + D_{\text{stop}} + \Delta_{\text{stop}}] \cap (a_h, b_h) \neq \emptyset$, i.e., some vehicles within the cluster may not stop before the intersection.

We proceed as follows:

- in case (i), define $\bar{u}_i$ as in (8),
- in case (ii) then:
  - for all vehicles $i$ such that $x_i(0) + D_{\text{stop}} + \Delta_{\text{stop}} \leq a_h$ (i.e., vehicles that can surely reach a full stop before reaching the intersection, by Lemma 6), define $\bar{u}_i$ as in (8); assume there are $n - m$ such vehicles;
  - for the remaining $m$ vehicles, define $\bar{u}_i$ as a bang-bang input

$$\bar{u}_i(t) := \underline{\left|u_{\max,[0,t_i^*]}, u_{\min,(t_i^*,\infty)}\right|} \qquad (10)$$

switching at a time $t_i^* \geq 0$ so as to satisfy

$$t_i^* := \max\Big\{\tau,$$
$$t_i^* \text{ s.t. } \lim_{t\to\infty} x_i(t, \bar{u}_i) = b_h + D_{\text{stop}} + (m-i)d\Big\}.$$

**Theorem 8.** *$I^{\tau}_C(t)$ defined above satisfies the property in Definition 5 for Scenario 2.*

The above theorem is proven in the Appendix.

To better understand how clusters are formed using the above-defined guaranteed hull, consider the following properties.

**Definition 9.**

Pa : *all vehicles of $C$ are on the same path $\mathcal{P}_h$*

Pb : *for all paths $\mathcal{P}_h$, calling $1,\dots,n$ the vehicles of $C$ on $\mathcal{P}_h$, $[x_n(0), x_1(0) + D_{\text{stop}} + \Delta_{\text{stop}}] \cap [a_{h,k}, b_{h,k}] = \emptyset$, $\forall\, k$*

Pc : *there exists a unique intersection $\mathcal{I}_k$ such that, for all paths $\mathcal{P}_h$, calling $1,\dots,n$ the vehicles of $C$ on $\mathcal{P}_h$, $[x_n(0), x_1(0) + D_{\text{stop}} + \Delta_{\text{stop}}] \cap [a_{h,k}, b_{h,k}] \neq \emptyset$*

Pd : *if $C$ has vehicles $1$ and $n$ on a path $\mathcal{P}_h$ and $x_1 > x_n$, then all vehicles $i$ with $x_i \in [x_n(0), x_1(0)]$ belong to $C$.*

All clusters formed by Algorithm 1 in this scenario either have properties $\text{Pa} \wedge \text{Pb} \wedge \text{Pd}$ (where $\wedge$ is the logical *and*), or have properties $\text{Pc} \wedge \text{Pd}$ (a proof of this is given in Lemma 10 in the Appendix). In the first case, they can be treated as an isolated set of vehicles on a single path, thus falling under the umbrella of Scenario 1, discussed in the previous section. In the second case they require the tools introduced in this section.

The above properties help us formalise one more interesting observation about the guaranteed hull for this scenario.

**Lemma 9.** *Consider a cluster $C$ with vehicles on a path $\mathcal{P}_h$ and assume $C$ has property* Pc*. Name $1,\dots,n$, in topological order, the vehicles of $C$ in $\mathcal{P}_h$ with $x_i(0) + D_{\text{stop}} + \Delta_{\text{stop}} \geq a_h$. If $C$ was formed using Algorithm 1, then*

$$\lim_{t\to\infty} x_1(t, u_{\min}) \leq b_h + D_{\text{stop}} + (n-1)(d + \Delta_{\text{stop}})$$

*and*

$$\lim_{t\to\infty} x_1(t, \bar{u}_1) \leq b_h + D_{\text{stop}} + \Delta_{\text{stop}} + (n-1)(d + \Delta_{\text{stop}}).$$

The proof of the above lemma is reported in the Appendix. In simple terms, the lemma states that a cluster that handles vehicles at an intersection (has property Pc) will never include vehicles farther than $D_{\text{stop}} + \Delta_{\text{stop}} + (n-1)(d + \Delta_{\text{stop}})$ from the intersection along a given path, where $n$ is the number of vehicles the cluster has on the path. This provides a useful tool to decouple nearby intersections, as we do in Scenario 3. It also allows to improve scalability of Algorithm 1, stating that vehicles in a cluster with property Pc need only to communicate with vehicles within a distance $D_{\text{stop}} + \Delta_{\text{stop}} + (n-1)(d + \Delta_{\text{stop}})$ of the intersection.

### A. Simulation of Scenario 2

We simulate this scenario using the same model and parameters as in Scenario 1. Figure 4 portrays the trajectories of 7 vehicles on a two-path example. Initially Algorithm 1 separates all vehicle in distinct clusters. At $t = 2.4$ both vehicles 1 and 5 have $[x_i, x_i + D_{\text{stop}} + \Delta_{\text{stop}}] \cap (a_h, b_h) \neq \emptyset$ (case (ii) in the definition of $I_i^\tau$). Algorithm 1 detects a possible side collision, and merges them in the same cluster. Vehicle 5 is soon forced to brake by the supervisor, to avoid occupying the intersection at the same time as vehicle 1. At $t = 3.6$ $[x_2, x_2 + D_{\text{stop}} + \Delta_{\text{stop}}] \cap (a_h, b_h) \neq \emptyset$: vehicle 2 joins the same cluster and is forced to brake in order to avoid

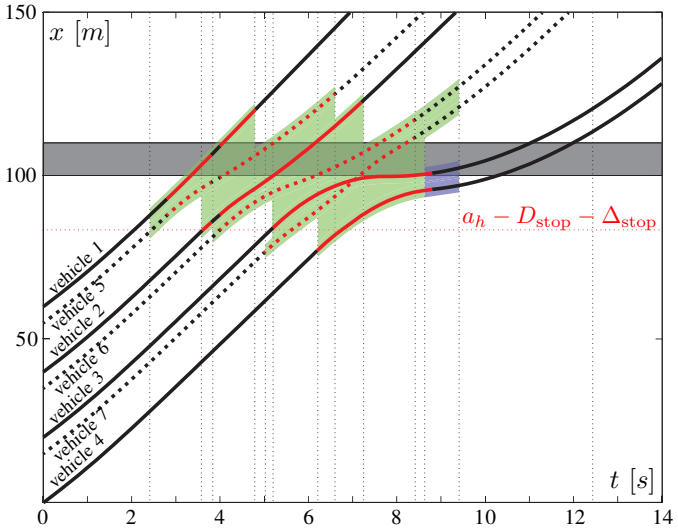


Fig. 4. Simulation of Scenario 2 with two paths. Solid and dotted trajectories are of vehicles on path 1 (vehicles $1\dots4$) and 2 (vehicles $5, 6, 7$), respectively. Initial conditions are $\mathbf{x}(0) = ((60,10),(40,10),(20,10),(0,10),(55,10),(35,10),(15,10))$. The gray band represents the intersection along both paths, clusters and supervisor override are represented as in Fig. 3.

a side collision with vehicle 5. At $t = 3.8$ vehicle 6 joins the cluster as a consequence of the braking manoeuvre initiated by vehicle 5 (when 6 joins the cluster it is still "far" from the intersection: $[x_6, x_6 + D_{\text{stop}} + \Delta_{\text{stop}}] \cap (a_h, b_h) = \emptyset$), and it is soon forced to brake by the supervisor. At $t = 4.8$ vehicle 1 leaves the cluster. At $t = 5$ vehicle 7 joins the cluster due to the braking manoeuvre of vehicle 6 and soon it is forced to brake. At $t = 5.2$ $[x_3, x_3 + D_{\text{stop}} + \Delta_{\text{stop}}] \cap (a_h, b_h) \neq \emptyset$: vehicle 3 joins the cluster and is immediately forced to brake to avoid a side collision. At $t = 6.2$ vehicle 4 joins the cluster, due to the braking manoeuvre of vehicle 3. At $t = 6.6$ and $t = 7.2$ vehicles 5 and 2, respectively, leave the cluster. Note that these two vehicles travel more slowly and have more successors in the same cluster than vehicle 1 had when it left the intersection. Consequently, they leave the cluster farther from the intersection than vehicle 1 did. At $t = 8.4$ vehicle 4 leaves the intersection and has no successors. As a consequence the cluster is split in two, one containing vehicles 6 and 7, the other 3 and 4.

## VII. Scenario 3: Road Network

**Scenario 3.** *A set of $N$ vehicles drive on $M$ different paths transversely intersecting at $K$ intersections. Vehicles on the same path have identical dynamics.*

In this scenario, we formally keep the definition of $I_C^\tau$ that we used in Section VI, simply adapting the notation. We take

$$I_i^\tau(t) := \mathcal{P}_h([x_i(t, u_{i,\min}), x_i(t, \bar{u}_i)]),$$

where $\bar{u}_i$ is constructed as follows. Call $n$ the last vehicle that a cluster $C$ has on a path $\mathcal{P}_h$, let $[a_{h,k}, b_{h,k}]$ be the interval $\{x : \mathcal{P}_h(x) \in \mathcal{I}_k\}$ (i.e., the extent of the intersection $\mathcal{I}_k$ along

$\mathcal{P}_h$), and let $k_1$ be the first intersection after vehicle $n$, i.e., let $k_1$ be such that $b_{h,k_1} := \min_k(b_{h,k} > x_n)$.

Let $m$ be the number of vehicles of $C$ in $\mathcal{P}_h$ such that $x(0) + D_{\text{stop}} + \Delta_{\text{stop}} > a_{h,k_1}$. Then:

(i) if $m = 0$, define $\bar{u}_i$ as in (8);

(ii) if $m > 0$ then:

- for all vehicles $i$ such that $x_i(0) + D_{\text{stop}} + \Delta_{\text{stop}} \leq a_{h,k_1}$ define $\bar{u}_i$ as in (8),
- for all other vehicles call $a_h := a_{h,k_1}$; $b_h := b_{h,k_1}$ and define $\bar{u}_i$ as in (10).

We assume the following condition, which sets an upper bound to the number of vehicles that can occupy a stretch of road linking two intersections.

**Definition 10** (Sparse traffic condition)**.** *Take any two intersections $\mathcal{I}_1$, $\mathcal{I}_2$ along a path $\mathcal{P}_h$, with $(a_{h,1}, b_{h,1}) < (a_{h,2}, b_{h,2})$. Assume $L := a_{h,2} - b_{h,1} > 2D_{\text{stop}} + \Delta_{\text{stop}}$. Traffic is sparse with respect to a time-step $\tau$ if there are at most $(L - 2D_{\text{stop}} - \Delta_{\text{stop}})/(d + \Delta_{\text{stop}}) + 1$ vehicles lying in the left-open interval $(a_{h,1} - D_{\text{stop}} - \Delta_{\text{stop}}, a_{h,2}]$.*

A simulation of sparse traffic in a scenario similar to Fig. 5, run using the algorithm discussed in this paper, is available at https://vimeo.com/119876036. Under the above condition we can prove that the partitioning algorithm separates in different clusters traffic in the neighbourhood of different intersections. This allows to perform the verification of each cluster by the same means as in the previous section, as each intersection is an isolated problem. To formalise and prove this statement we use properties Pa-Pd introduced in Definition 9. In a nutshell, a cluster having properties $\mathrm{Pa} \wedge \mathrm{Pb} \wedge \mathrm{Pd}$ can be handled as in Scenario 1, while a cluster having $\mathrm{Pc} \wedge \mathrm{Pd}$ can be handled as in Scenario 2. The main result of this section is to prove that all elements of a partition returned by Algorithm 1 have either $\mathrm{Pa} \wedge \mathrm{Pb} \wedge \mathrm{Pd}$ or $\mathrm{Pc} \wedge \mathrm{Pd}$. This is done in Lemma 10. Using this result, Theorem 11 states the correctness of $I_C^\tau$ for this scenario, by simply building on the results we already proved for Scenarios 1 and 2.

**Lemma 10.** *If traffic is sparse, for all clusters computed by Algorithm 1 the statement $\mathrm{Pd} \wedge \big((\mathrm{Pa} \wedge \mathrm{Pb}) \vee \mathrm{Pc}\big)$ is true.*

In the above lemma, $\vee$ is the logical *or*. The full proof of the above lemma is rather involved, and is reported in the appendix.

**Theorem 11.** *If traffic is sparse and clusters are computed using Algorithm 1, then $I_C$ defined above satisfies the property in Definition 5 for Scenario 3.*

*Proof:* The proof follows trivially from Lemma 10 and Theorem 8. ■

### A. Simulation of Scenario 3

Here we test our result on the road scenario of Fig. 5, using the same longitudinal dynamics model and parameters as in the preceding scenarios. Vehicles travel along three paths intersecting at two points. Their trajectories are portrayed in Fig. 6. Initial conditions were chosen so as to ensure that the sparsity condition is satisfied. As expected, vehicles in a neighbourhood of each intersection tend to be clustered together, but a single cluster never spans both intersections (as a consequence of properties Pb and Pc).

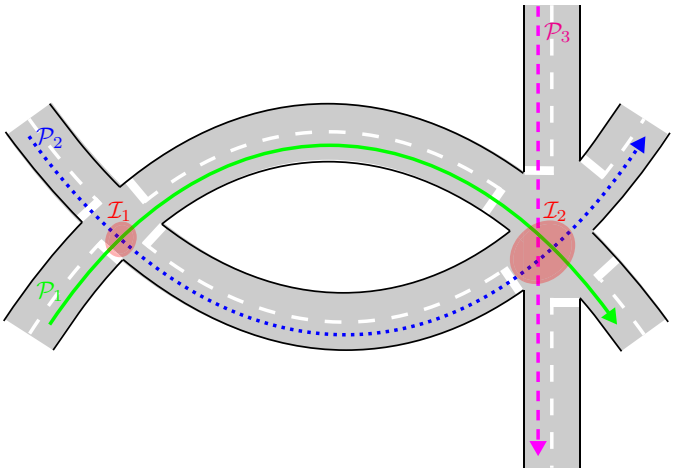


Fig. 5. Two intersections of three paths, the conflict regions around each intersection are shaded red.

Note that, despite having 16 vehicles in the simulation, never more than 6 vehicles belong to the same cluster. This is a consequence of our choice of guaranteed hull favouring spatially localized clusters. Clusters of this size can be handled in real time by existing exact verification algorithms (in our example the cluster supervisor ran in less than $0.15s$ on a 2GHz Intel Xeon E5-2650 with 2 GB of RAM). For larger clusters, approximate polynomial-time algorithms would be necessary (see e.g. [8]). Note that the size of the simulated

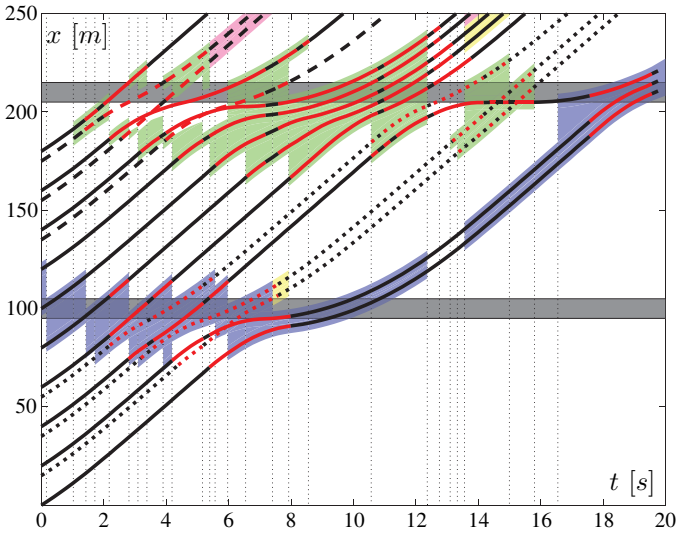


Fig. 6. Vehicles in the road network of Fig. 5. Solid, dashed, and dotted trajectories are of vehicles on path 1 (vehicles $1 \ldots 10$), 2 (vehicles $11, 12$, and 13), and 3 (vehicles $14, 15$, and 16), respectively. Initial conditions are $\mathbf{x}(0) = ((180, 10), (160, 10), (140, 10), (120, 10), (100, 10), (80, 10), (60, 10), (40, 10), (20, 10), (0, 10), (55, 10), (35, 10), (15, 10), (175, 10), (155, 10), (135, 10))$. Supervisor override, clusters, and intersection intervals are coloured as in Figs. 3 and 4.

road network was limited by the need to graphically represent the result, but a large network could be treated by the same means.

## VIII. Conclusions

We have proved that, under fairly mild assumptions, a large vehicle network can be partitioned through a distributed algorithm in a set of vehicle clusters, so that safety of the manoeuvres of cars in one cluster can be verified independently of the other clusters. We have used the above result to define a distributed and least restrictive supervisor for the vehicle network. Note, however, that independence of the partition (as defined in Section II) implies that, if any of the approximate verification algorithm proposed in the literature (e.g. [8], [33], [38]) is used to verify a cluster, its error bound guarantees are preserved at the network level. This is a straightforward consequence of partition independence. This means that existing approximate verification algorithms can be used, with no modification, to scale the presented approach to cases where clusters are too large for exact verification.

The key idea in this construction is the independent partition, which guarantees that no approximation is introduced when the supervisor of the road network is decomposed in a set of independent cluster supervisors. The partitioning algorithm is designed so that, once a partition is computed, supervisory control requires no more communication between vehicles of different clusters. Exchanging information between clusters provides no further improvement in terms of safety or restrictiveness. This may of course not be the case if different performance metrics were considered.

In Sections V-VII we have provided an example application in three different scenarios, using the exact verification algorithms discussed in [8] to implement the cluster supervisors. The choice of the example scenarios was dictated by the path configurations for which exact and reasonably efficient verification algorithms are currently known. However, work is in progress to extend verification methods to handle much more complex scenarios (see e.g. [38]), including multiple nearby intersections, merging and splitting paths. To handle such scenarios the definitions of guaranteed hull in Sections V-VII will have to be modified, though we believe that the general structure outlined in this paper should be flexible enough to apply to most reasonable road network scenarios.

Two critical aspects of our result are that all network-wide computation must be completed within a given time $\tau$, and that, in Scenario 3, sparsity is required to hold throughout the evolution of the system. Regarding the time constraint, further work will need to assess the size of the largest network that can be practically handled within a reasonable $\tau$, taking into account all technological constraints. Also, if the algorithm is run on a closed system, a way is needed to register new vehicles as they join the system without causing the supervisor to block. Similar issues were addressed in [23], [25], [27]. Regarding sparsity, our approach could be improved by having the supervisor enforce it, at the cost of prohibiting some safe manoeuvres. Note that sparsity is only needed to insure that Scenario 3 decomposes in clusters fitting Scenarios 1 and 2. In this case, verification and computation of an override (lines 4 and 7 of Algorithm 2) can be done with methods presented in [8]. In the lack of sparsity, the main results of this paper, in Sections III and IV, still hold, but other methods must be developed for verification and override computation.

## Appendix

The three following lemmas are used in the proofs of Theorems 5 and 8.

**Lemma 12** (Continuity)**.** *Let*

$$u_i := \underline{\left| u_{i,[0,t_1]}, u_{i,(t_1,t_2]}, \dots \right|}$$

*($u_i$ is a piecewise smooth signal, discontinuous at times $t_1, t_2, \dots$). The trajectory $x_i(t, u_i)$ depends continuously on the switching instants $t_1, t_2, \dots$*

The above property, proved in [8], is used to construct continuous families of trajectories parametrized in the times of switching of the corresponding inputs.

**Lemma 13.** *Consider vehicles $i$ and $j$ with identical dynamics, and an input signal $u_i$. Let $\lim_{t\to\infty} x_i(t, u_i) \leq \lim_{t\to\infty} x_j(t, u_{\min}) < \infty$. Then $x_j(t, u_{\min}) \geq x_i(t, u_i)$ for all $t \geq 0$.*

*Proof:* The proof is by contradiction. Assume that the condition $x_j(t, u_{min}) \geq x_i(t, u_i)$ is violated for some $t \geq 0$. This would imply that $\dot{x}_i > \dot{x}_j$ at the time when $x_j - d$ and $x_i$ intersect. By the monotonicity property (6), we would have $\lim_{t\to\infty} x_j(t, u_{\min}) < \lim_{t\to\infty} x_i(t, u_i)$, contradicting our hypothesis. ∎

Note that the above lemma also applies in the case $i = j$.

**Lemma 14.** *Consider vehicle 2 and its predecessor 1 on the same path. Assume $\mathcal{U}_s(\mathbf{x}) \neq \emptyset$. If there exists $(u_1, u_2) \in \mathcal{U}_s(\mathbf{x})$ such that $\lim_{t\to\infty} x_2(u_2) = \alpha$ for some $\alpha \in \mathbb{R}$, then there exists $(u_1', u_2) \in \mathcal{U}_s(\mathbf{x})$ (with the component $u_2$ unchanged) such that $\lim_{t\to\infty} x_1(t, u_1') \leq \max\{\alpha + d, \lim_{t\to\infty} x_1(t, u_{\min})\}$.*

*Proof:* Consider the family of inputs in the parameter $\theta$

$$u_1^\theta := \underline{\left| u_{\min,[0,\theta]}, u_{\max,[\theta,\infty)} \right|}.$$

Assume at first that $(u_1^\infty, u_2) \in \mathcal{U}_s(\mathbf{x})$. This implies that $\lim_{t\to\infty} x_1(t, u_1^\infty) \geq \alpha + d$, since otherwise $\mathbf{x}(t)$ would enter $B_-$ for sufficiently large $t$. We thus have $\lim_{t\to\infty} x_1(t, u_1^\infty) = \lim_{t\to\infty} x_1(t, u_{\min}) \geq \alpha + d$, and the lemma statement is true.

Consider now the case where $(u_1^\infty, u_2) \notin \mathcal{U}_s(\mathbf{x})$. This means

$$\exists\, t : x_1(t, u_1^\infty) = x_1(t, u_{\min}) < x_2(t, u_2) + d.$$

Thus, by the continuity property in Lemma 12, there exists a $\theta_u < \infty$ such that

$$\exists\, t : x_1(t, u_1^{\theta_u}) < x_2(t, u_2) + d,$$

while by (A.5) $\lim_{t\to\infty} x_1(u_1^{\theta_u}) = \infty$ and therefore

$$\exists\, t_u \geq 0 : x_1(t, u_1^{\theta_u}) \geq x_2(t, u_2) + d,\ \forall\, t \geq t_u$$

(recall that $\lim_{t\to\infty} x_2(t, u_2) = \alpha < \infty$). This means that $x_1(t, u_1^{\theta_u})$ lies above $x_2(t, u_2) + d$ at all $t$ except for a

closed interval contained in $(0, t_u)$. Then, by (A.1) and (5), $x_1(t, u_1^0) \geq x_1(t, u_1)$ for all $t \geq 0$ and for all $u_1$. Given that $\mathcal{U}_s(\mathbf{x}) \neq \emptyset$, this implies

$$x_1(t, u_1^0) \geq x_2(t, u_2) + d, \, \forall \, t \geq 0.$$

Now consider the function $\theta \mapsto \mathbb{R}$

$$\min_{t \in (0, t_u)} (x_1(t, u_1^\theta) - d - x_2(t, u_2)).$$

We have shown above that it is non-negative for $\theta = 0$, and negative for $\theta = \theta_u$, so by Lemma 12 and the intermediate value theorem it must have a zero for some $\theta_{\text{tng}} \in [0, \theta_u)$. Such a $\theta_{\text{tng}}$ corresponds to a trajectory $x_1(t, u_1^{\theta_{\text{tng}}})$ which is tangent to $x_2(t, u_2) + d$ at some time $t_{\text{tng}} \in (0, t_u)$. Let

$$u_1' := \underline{\left| u_{1,[0,t_{\text{tng}}]}^{\theta_{\text{tng}}}, u_{2,(t_{\text{tng}}, \infty)} \right|}.$$

Given that

- vehicles 1 and 2 have identical dynamics,
- $x_1(t_{\text{tng}}, u_1') = x_2(t_{\text{tng}}, u_2) + d$,
- $\dot{x}_1(t_{\text{tng}}, u_1') = \dot{x}_2(t_{\text{tng}}, u_2)$,
- and the vehicles use the same inputs for $t > t_{\text{tng}}$,

$x_1(t, u_1') = x_2(t, u_2) + d$ for all $t \geq t_{\text{tng}}$, so that $\lim_{t \to \infty} x_1(u_1') = \alpha + d$. This completes the proof. ∎

*Proof of Theorem 5:* We must show that if $\mathcal{U}_s(\mathbf{x}_C) \neq \emptyset$ then for any $\mathbf{u}_{[0,\tau]} \in \mathcal{U}_s(\mathbf{x}_C)$ it is possible to construct a $\mathbf{u}_{(\tau,\infty)} \in \mathcal{U}_s(\mathbf{x}_C(\tau, \mathbf{u}_{[0,\tau]}))$ such that

$$\mathbf{y}_C(t, \mathbf{u}) \in I_C^\tau(t), \, \forall \, t \geq 0.$$

We first construct $\mathbf{u}_{(\tau,\infty)}$ so that, for any $\mathbf{u}_{[0,\tau]} \in \mathcal{U}_s(\mathbf{x}_C)$, $\mathbf{x}_C(t, \mathbf{u}_C)$ is lower-bounded by $\mathbf{x}_C(t, u_{\min})$ and upper-bounded by $\mathbf{x}_C(t, \bar{u}_C)$. Then we will prove that $\mathbf{u}_{(\tau,\infty)} \in \mathcal{U}_s(\mathbf{x}_C(\tau, \mathbf{u}_{[0,\tau]}))$.

The theorem only refers to vehicles of a given cluster $C$; therefore, to keep notation simple, we will assume throughout the proof that all vehicles of the system belong to $C$, and will avoid the use of subscript $C$ to restrict vectors to the cluster.

Take $n$ as the last vehicle of $C$, i.e., the only vehicle that is not a predecessor of another vehicle in $C$. We must show that, for any $u_{n,[0,\tau]} \in \mathcal{U}_s(\mathbf{x})$, there exists a $u_{n,(\tau,\infty)}$ such that

$$x_n(t, u_{\min}) \leq x_n(t, u_n) \leq x_n(t, \bar{u}_n), \, \forall \, t \geq 0.$$

The first inequality follows from (5) no matter how we construct $u_{n,(\tau,\infty)}$. To prove the second inequality, define $u_{n,(\tau,\infty)} := u_{\min}$. In (8) we have $\tau = t_n^*$ and $\bar{u}_{n,[0,t_n^*]} = u_{\max}$, therefore the second inequality is true in the time interval $[0, \tau]$. Moreover, (5) and (8) ensure that $x_n(\tau, u_n) \leq x_n(\tau, \bar{u}_n)$, and $\dot{x}_n(\tau, u_n) \leq \dot{x}_n(\tau, \bar{u}_n)$, and we have constructed $u_{n,(\tau,\infty)}$ so that $u_n(t) = \bar{u}_n(t), \, \forall \, t > \tau$. Therefore by (5) the inequality is true for all $t \geq 0$.

Consider now vehicle $n-1$, the predecessor of $n$. We have by assumption that $\mathbf{u}_{[0,\tau]} \in \mathcal{U}_s(\mathbf{x})$, and therefore that $\mathcal{U}_s(\mathbf{x}(\tau, \mathbf{u})) \neq \emptyset$. We can use Lemma 14 to construct $u_{n-1,(\tau,\infty)}$ such that

$$\lim_{t \to \infty} x_{n-1}(t, u_{n-1}) \leq \max \Big\{ \lim_{t \to \infty} x_n(t, u_n) + d, \lim_{t \to \infty} x_{n-1}\big(t, \underline{\left| u_{n-1,[0,\tau]}, u_{\min,(\tau,\infty)} \right|}\big) \Big\}. \quad (11)$$

We now show that for any $u_{n-1,[0,\tau]}$ this $u_{n-1,(\tau,\infty)}$ satisfies

$$x_{n-1}(t, u_{\min}) \leq x_{n-1}(t, u_{n-1}) \leq x_{n-1}(t, \bar{u}_{n-1}), \, \forall \, t \geq 0.$$

As before, the first inequality follows from (5). The second inequality is true in the time interval $[0, t^*]$ by (5), given definition (8). Now, notice that the definition of $t^*$ in (8) implies that

$$\lim_{t \to \infty} x_{n-1}(t, \bar{u}_{n-1}) = \max \Big\{ \lim_{t \to \infty} x_n(t, \bar{u}_n) + d, \lim_{t \to \infty} x_{n-1}\big(t, \underline{\left| u_{\max,[0,\tau]}, u_{\min,(\tau,\infty)} \right|}\big) \Big\},$$

and that we have

$$\lim_{t \to \infty} x_n(t, \bar{u}_n) + d \geq \lim_{t \to \infty} x_n(t, u_n) + d$$

(we proved it before), and

$$\lim_{t \to \infty} x_{n-1}\big(t, \underline{\left| u_{\max,[0,\tau]}, u_{\min,(\tau,\infty)} \right|}\big) \geq \lim_{t \to \infty} x_{n-1}\big(t, \underline{\left| u_{n-1,[0,\tau]}, u_{\min,(\tau,\infty)} \right|}\big)$$

(by (5)). Using (11) and the two above inequalities, we obtain

$$\lim_{t \to \infty} x_{n-1}(t, \bar{u}_{n-1}) \geq \lim_{t \to \infty} x_{n-1}(t, u_{n-1}).$$

By Lemma 13, and since $\bar{u}_{n-1,(t^*,\infty)} = u_{\min}$, this implies

$$x_{n-1}(t, \bar{u}_{n-1}) \geq x_{n-1}(t, u_{n-1}), \, \forall \, t \geq t^*,$$

and therefore $x_{n-1}(t, \bar{u}_{n-1}) \geq x_{n-1}(t, u_{n-1}), \, \forall \, t \geq 0$. By sequentially performing the same reasoning for vehicles $n-2, \ldots, 1$, we can construct $\mathbf{u}$ such that $\mathbf{y}(t, \mathbf{u}) \in I^\tau(t), \, \forall \, t \geq 0$. Moreover, the use of Lemma 14 ensures the constructed $\mathbf{u}_{(\tau,\infty)} \in \mathcal{U}_s(\mathbf{x}(\tau, \mathbf{u}_{[0,\tau]}))$. This concludes the proof. ∎

*Proof of Theorem 8:* The theorem only refers to vehicles of a given cluster $C$; therefore, to keep notation simple, we will assume throughout the proof that all vehicles of the system belong to $C$, and will avoid the use of subscript $C$ to restrict vectors to the cluster.

First, we will show that, given any $\mathbf{u}_{[0,\tau]} \in \mathcal{U}_s(\mathbf{x})$, the input $\mathbf{u}$ can be completed over the interval $(\tau, \infty)$ so that, for any vehicle $i$,

$$x_i(t, u_{\min}) \leq x_i(t, u_i) \leq x_i(t, \bar{u}_i), \, \forall \, t \geq 0. \quad (12)$$

Let us start by considering all the vehicles for which $\bar{u}_i$ is defined as in (8): on all paths that fall into case (i), construct the input as in the proof of Theorem 5. Similarly, on paths which fall into case (ii), for all vehicles such that $x_i(0) + D_{\text{stop}} + \Delta_{\text{stop}} \leq a_h$, i.e., for all vehicles which can surely stop before reaching the beginning of the intersection, construct the input as in the proof of Theorem 5. We already proved that such an input satisfies (12), this takes care of rear-end collisions. Moreover, by Lemma 6, for all considered vehicles

$$\lim_{t \to \infty} x_i(t, \bar{u}_i) \leq a_h;$$

this excludes side-collisions.

Let us now consider the only remaining vehicles, i.e., those for which

$$x_i(0) + D_{\text{stop}} + \Delta_{\text{stop}} > a_h. \quad (13)$$

The structure of the proof is similar to the one used in Theorem 5. Assume that vehicles are numbered in topological order. We construct an input for the last vehicle on each path which satisfies (13)—call it vehicle $m$—so that the vehicle's position remains within $I_m^\tau$; then we construct an input for its predecessor $m-1$ so that its position remains within $I_{m-1}^\tau$ and so that it does not collide with $m$; then we repeat the reasoning to construct safe inputs for all remaining vehicles. The difference from Theorem 5 is that now these inputs must be constructed so as to avoid also lateral collisions with vehicles from other paths.

First, in order to avoid lateral collisions, let us construct inputs so as to drive each vehicle safely to the end of the intersection. Since $\mathcal{U}_s(\mathbf{x}) \neq \emptyset$, there exists $\tilde{\mathbf{u}} \in \mathcal{U}_s(\mathbf{x})$ that brings the vehicles beyond $b_h$ avoiding rear and lateral collisions. For each vehicle, let $\tilde{t}_i$ be the earliest non-negative time when $x_i(\tilde{t}_i, \tilde{u}_i) \geq b_h$, that is

$$\tilde{t}_i := \arg\min_{t\geq 0}\{t : x_i(\tilde{t}_i, \tilde{u}_i) \geq b_h\}, \tag{14}$$

and call

$$\bar{t}_i := \max\{\tau, \tilde{t}_i\}. \tag{15}$$

Let each vehicle $i$ use input $\tilde{u}_i$ up to time $\bar{t}_i$. With this input vehicles reach the end of the intersection without collisions; we are only left to show that we can construct $u_{i,(\bar{t}_i,\infty)}$ so as to satisfy (12).

Define

$$u_{m,(\bar{t}_m,\infty)} := u_{\min}. \tag{16}$$

If $m$ has a successor $m+1$, we have constructed $u_{m+1}$ so that $\lim_{t\to\infty} x_{m+1}(t, u_{m+1}) \leq a_h$; we have constructed $u_{m,(0,\bar{t}_m)}$ to avoid collisions before $m$ reaches $b_h$; and we have assumed $b_h - a_h \geq d$. Therefore, $m$ and $m+1$ cannot collide.

We now show that input (16) satisfies (12). The first inequality in (12) follows directly from (5). The second inequality is true in the time interval $[0, t^*]$ by (5), given (10). We prove the inequality over the interval $(t^*, \infty)$ considering separately the cases $\tau > \tilde{t}_m$ and $\tau \leq \tilde{t}_m$.

($\tau > \tilde{t}_m$):
In this case $\bar{t}_m = \tau$. We therefore have

$$u_m := \underline{\left| \tilde{u}_{m,[0,\tau]}, u_{\min,(\tau,\infty)} \right|},$$

while

$$\bar{u}_m := \underline{\left| u_{\max,[0,t^*]}, u_{\min,(t^*,\infty)} \right|},$$

and, by (10), $t^* \geq \tau$. Using (5) we conclude that

$$x_m(t, \bar{u}_m) \geq x_m(t, u_m),\ \forall\, t \geq t^*.$$

($\tau \leq \tilde{t}_m$):
We have

$$\lim_{t\to\infty} x_m(t, u_m) = x_m(\tilde{t}, u_m) + \left(\lim_{t\to\infty} x_m(t, u_m) - x_m(\tilde{t}, u_m)\right).$$

Using (16), Definition 7, and the fact that the vector field (1) is invariant to translation in the position $x$, (5) ensures that

$$\lim_{t\to\infty} x_m(t, u_m) - x_m(\tilde{t}_m, u_m) \leq D_{\text{stop}}.$$

By the definition of $\tilde{t}_m$, $x_m(\tilde{t}_m, u_m) = b_h$, thus obtaining

$$\lim_{t\to\infty} x_m(t, u_m) \leq b_h + D_{\text{stop}}. \tag{17}$$

This, together with (10), gives that

$$\lim_{t\to\infty} x_m(t, u_m) \leq \lim_{t\to\infty} x_m(t, \bar{u}_m).$$

The above inequality and $\bar{u}_{m,(t^*,\infty)} = u_{\min}$ allow to use Lemma 13 to conclude that

$$x_m(t, \bar{u}_m) \geq x_m(t, u_m),\ \forall\, t \geq t^*.$$

Therefore, for both $\tau > \tilde{t}_m$ and $\tau \leq \tilde{t}_m$,

$$x_m(t, \bar{u}_m) \geq x_m(t, u_m),\ \forall\, t \geq 0.$$

Consider now vehicle $m-1$, the predecessor of $m$. We have by assumption that $\mathcal{U}_s([x_{m-1}(\bar{t}_{m-1}), x_m(\bar{t}_{m-1})]) \neq \emptyset$ (since $x_m(0) < x_{m-1}(0)$ implies $\bar{t}_m \geq \bar{t}_{m-1}$), therefore we can use Lemma 14 to construct $u_{m-1,(\bar{t}_{m-1},\infty)}$ such that

$$\lim_{t\to\infty} x_{m-1}(t, u_{m-1}) \leq \max\Big\{ \lim_{t\to\infty} x_m(t, u_m) + d, \lim_{t\to\infty} x_{m-1}\big(t, \underline{\left| u_{m-1,[0,\bar{t}_{m-1}]}, u_{\min,(\bar{t}_{m-1},\infty)} \right|}\big)\Big\}. \tag{18}$$

We now show that for any $u_{m-1,[0,\bar{t}_{m-1}]}$ this $u_{m-1,(\bar{t}_{m-1},\infty)}$ satisfies

$$x_{m-1}(t, u_{\min}) \leq x_{m-1}(t, u_{m-1}) \leq x_{m-1}(t, \bar{u}_{m-1}).$$

As before the first inequality follows from (5). The second inequality is true in the time interval $[0, t^*]$ by (5), given (10). To prove the inequality over the interval $(t^*, \infty)$ we consider separately the two cases $\tilde{t}_{m-1} \leq \tau$ and $\tilde{t}_{m-1} > \tau$.

($\tau > \tilde{t}_{m-1}$):
The argument is identical to the one used for vehicle $m$ in the case ($\tau > \tilde{t}_m$).

($\tau \leq \tilde{t}_{m-1}$):
Using (16), Definition 7, and the fact that the vector field (1) is invariant to translation in the position $x$, (5) ensures that

$$\lim_{t\to\infty} x_{m-1}\big(t, u_{m-1}\big) := \lim_{t\to\infty} x_{m-1}\big(t, \underline{\left| u_{m-1,[0,\tilde{t}_{m-1}]}, u_{\min,(\tilde{t}_{m-1},\infty)} \right|}\big) \leq b_h + D_{\text{stop}}.$$

Moreover, since we have $\tilde{t}_{m-1} \leq \tilde{t}_m$, we are in the case where $\tau \leq \tilde{t}_m$, and we have shown in (17) that

$$\lim_{t\to\infty} x_m(t, u_m) \leq b_h + D_{\text{stop}}.$$

Using the two above inequalities in (18), we obtain

$$\lim_{t\to\infty} x_{m-1}(t, u_{m-1}) \leq b_h + D_{\text{stop}} + d,$$

while (10) ensures that

$$\lim_{t\to\infty} x_{m-1}(t, \bar{u}_{m-1}) \geq b_h + D_{\text{stop}} + d$$

and therefore that

$$\lim_{t\to\infty} x_{m-1}(t, u_{m-1}) \leq \lim_{t\to\infty} x_{m-1}(t, \bar{u}_{m-1}).$$

Having $\bar{u}_{m-1,(t^*,\infty)} = u_{\min}$, the above inequality allows to

use Lemma 13 to conclude that

$$x_{m-1}(t, \bar{u}_{m-1}) \geq x_{m-1}(t, u_{m-1}), \forall\, t \geq t^*.$$

Therefore, for both $\tau > \tilde{t}_{m-1}$ and $\tau \leq \tilde{t}_{m-1}$,

$$x_{m-1}(t, \bar{u}_{m-1}) \geq x_{m-1}(t, u_{m-1}), \forall\, t \geq 0.$$

By performing the same reasoning sequentially on all vehicles we complete any $\mathbf{u}_{[0,\tau]} \in \mathcal{U}_s(\mathbf{x})$ with an input $\mathbf{u}_{(\tau,\infty)}$ so that $\mathbf{u}$ satisfies (12). Moreover, $\mathbf{u} \in \mathcal{U}_s(\mathbf{x}(0))$ by construction. As a consequence, $I_C^\tau$ satisfies the property in Definition 5 for Scenario 2. ∎

*Proof of Lemma 9:* Let $j$ be the first vehicle such that $x(0) < b_h$. By the definition of $I_C^\tau$ and (10), we have

$$\lim_{t\to\infty} x_j(t, \bar{u}_j) \leq \max\Big\{b_h + D_{\text{stop}} + (n-j)d, \lim_{t\to\infty} x_j(t, \underline{\left|u_{\max,[0,\tau]}, u_{\min,(\tau,\infty)}\right|})\Big\}; \quad (19)$$

by Definition 7 we have

$$\lim_{t\to\infty} x_j(t, u_{\min}) \leq x_j(0) + D_{\text{stop}}; \quad (20)$$

and by Remark 3, we have

$$\lim_{t\to\infty} x_j(t, \underline{\left|u_{\max,[0,\tau]}, u_{\min,(\tau,\infty)}\right|}) \leq x_j(0) + D_{\text{stop}} + \Delta_{\text{stop}}. \quad (21)$$

Moreover, we have

$$x_j(0) \leq b_h. \quad (22)$$

Using (20) and (22), we obtain $\lim_{t\to\infty} x_j(t, u_{\min}) \leq b_h + D_{\text{stop}}$, while using (19), (21), and (22) we find

$$\lim_{t\to\infty} x_j(t, \bar{u}_j) \leq b_h + D_{\text{stop}} + \Delta_{\text{stop}} + (n-j)(d + \Delta_{\text{stop}}). \quad (23)$$

Now consider vehicle $j-1$, the predecessor of $j$. If Algorithm 1 merged $j$ and $j-1$ in the same cluster,

$$\lim_{t\to\infty} x_{j-1}(t, u_{\min}) \leq \lim_{t\to\infty} x_j(t, \bar{u}_j) + d, \quad (24)$$

that is,

$$\lim_{t\to\infty} x_{j-1}(t, u_{\min}) \leq b_h + D_{\text{stop}} + (n-j+1)(d + \Delta_{\text{stop}}).$$

Furthermore, by (10),

$$\lim_{t\to\infty} x_{j-1}(t, \bar{u}_{j-1}) \leq \max\Big\{b_h + D_{\text{stop}} + (n-j+1)d, \lim_{t\to\infty} x_{j-1}(t, \underline{\left|u_{\max,[0,\tau]}, u_{\min,(\tau,\infty)}\right|})\Big\}$$

and, by Definition 8,

$$\lim_{t\to\infty} x_{j-1}\left(t, \underline{\left|u_{\max,[0,\tau]}, u_{\min,(\tau,\infty)}\right|}\right) \leq \lim_{t\to\infty} x_{j-1}(t, u_{\min}) + \Delta_{\text{stop}}$$

thus obtaining

$$\lim_{t\to\infty} x_{j-1}(t, \bar{u}_{j-1}) \leq \max\Big\{b_h + D_{\text{stop}} + (n-j+1)d, \lim_{t\to\infty} x_{j-1}(t, u_{\min}) + \Delta_{\text{stop}}\Big\}.$$

Using (23) and (24) in the above we obtain

$$\lim_{t\to\infty} x_{j-1}(t, \bar{u}_{j-1}) \leq b_h + D_{\text{stop}} + \Delta_{\text{stop}} + (n-j+1)(d + \Delta_{\text{stop}}).$$

Iterating the above reasoning for vehicles $j-2, \ldots, 1$, we obtain

$$\lim_{t\to\infty} x_1(t, u_{\min}) \leq b_h + D_{\text{stop}} + (n-1)(d + \Delta_{\text{stop}}),$$

and

$$\lim_{t\to\infty} x_1(t, \bar{u}_1) \leq b_h + D_{\text{stop}} + \Delta_{\text{stop}} + (n-1)(d + \Delta_{\text{stop}}),$$

as we sought to prove. ∎

The following lemma is used in the proof of Lemma 10.

**Lemma 15.** *Consider a cluster $C$ with properties* Pb *or* Pc. *Name $1, \ldots, n$, in topological order, the vehicles of $C$ in a path $\mathcal{P}_h$, and assume that $x_1(0) + D_{\text{stop}} + \Delta_{\text{stop}} < a_{h,k}$, for some intersection $\mathcal{I}_k$. If $C$ was formed using Algorithm 1 and the sparsity condition holds, then*

$$\lim_{t\to\infty} x_1(t, \bar{u}) \leq a_{h,k}.$$

*Proof:* Let us first consider the case where $C$ has Pb. By Lemma 6, $\lim_{t\to\infty} x_1(t, \bar{u}_1) \leq x_1(0) + D_{\text{stop}} + \Delta_{\text{stop}}$. We have assumed $x_1(0) + D_{\text{stop}} + \Delta_{\text{stop}} < a_{h,k}$, therefore

$$\lim_{t\to\infty} x_1(t, \bar{u}_1) \leq a_{h,k}.$$

Consider now the case where $C$ has Pc, and let $\mathcal{I}_1$ be the only intersection such that $[x_n(0), x_1(0)_+ D_{\text{stop}} + \Delta_{\text{stop}}] \cap [a_{h,1}, b_{h,1}] \neq \emptyset$. By Lemma 9 we have

$$\lim_{t\to\infty} x_1(t, \bar{u}_1) \leq b_{h,1} + D_{\text{stop}} + \Delta_{\text{stop}} + (m-1)(d + \Delta_{\text{stop}}),$$

where $m$ is the number of vehicles on $\mathcal{P}_h$ such that $x(0) + D_{\text{stop}} + \Delta_{\text{stop}} \geq a_{h,1}$. The sparsity condition ensures $a_{h,k} - D_{\text{stop}} \geq b_{h,1} + D_{\text{stop}} + \Delta_{\text{stop}} + (m-1)(d + \Delta_{\text{stop}})$. Using the two above inequalities we obtain $\lim_{t\to\infty} x_1(t, \bar{u}_1) \leq a_{h,k}$. ∎

*Proof of Lemma 10:* The proof is by induction. First, notice that at the beginning of Algorithm1 all clusters have the property $\text{Pd} \wedge \text{Pa} \wedge \text{Pb}$ or the property $\text{Pd} \wedge \text{Pc}$. This is because, at the first iteration, all clusters consist of a single vehicle. This is the induction basis.

The rest of the proof consists in showing that, no matter how vehicles are grouped to form larger clusters, the resulting clusters have the property Pd, and either $\text{Pa} \wedge \text{Pb}$ or Pc. We consider cases based on the properties of the merged clusters, and on the reason why they are merged. Note that Algorithm 1 may merge more than two clusters at a time. For simplicity we consider only merging of pairs of clusters, the proof obviously extending to merging of larger sets.

Two clusters $C_1$ and $C_2$ can be merged through Line 6 of Algorithm 1 if

$$\exists t, i \in C_1, j \in C_2 : (I_i^\tau(t), I_j^\tau(t)) \cap B_{+,(i,j)} \neq \emptyset \quad (25)$$

i.e., because they interfere through $B_+$, or if

$$\exists t, i \in C_1, j \in C_2 : (I_i^\tau(t), I_j^\tau(t)) \cap B_{-,(i,j)} \neq \emptyset \quad (26)$$

i.e., because they interfere through $B_-$. If $C_1$ and $C_2$ are

merged because of (25), then necessarily both $C_1$ and $C_2$ have Pc. We thus define the following cases

(c1) $C_1$ and $C_2$ have $\mathrm{Pc} \wedge \mathrm{Pd}$ and are merged because of (25)
(c2) $C_1$ and $C_2$ have $\mathrm{Pc} \wedge \mathrm{Pd}$ and are merged because of (26)
(c3) $C_1$ has $\mathrm{Pc} \wedge \mathrm{Pd}$ and $C_2$ has $\mathrm{Pa} \wedge \mathrm{Pb} \wedge \mathrm{Pd}$, and they are merged because of (26)
(c4) $C_1$ and $C_2$ have $\mathrm{Pa} \wedge \mathrm{Pb} \wedge \mathrm{Pd}$ and are merged because of (26)

Note that (25) does not exclude (26), therefore cases (c1) and (c2) are not mutually exclusive; this does not affect the proof. In the rest of the proof, we use the following symbols:

- Given a set $n, \dots, 1$ of vehicles of a cluster $C_i$ on any given path, we call $\Im_i$, the interval
$$\Im_i := [x_n(0), x_1(0) + D_{\text{stop}} + \Delta_{\text{stop}}].$$
- Given a cluster $C_1 \cup C_2$ obtained by merging $C_1$ and $C_2$, we call $\Im_{1\cup 2}$ the corresponding set $\Im$.

The quantities defined above have the properties specified in the following remarks.

**Remark 4.** *If a cluster $C_i$ has $\mathrm{Pa} \wedge \mathrm{Pb}$, with all vehicles on $\mathcal{P}$, then* $\bigcup_{j \in C_i, t \geq 0} I_j^{\mathcal{T}}(t) \subseteq \mathcal{P}(\Im_i)$.

**Remark 5.** *Take two clusters $C_1$ and $C_2$ with vehicles on a common path $\mathcal{P}$. Let all vehicles of $C_2$ on $\mathcal{P}$ precede all vehicles of $C_1$ on the same path. If $C_1$ has $\mathrm{Pa} \wedge \mathrm{Pb}$ then Algorithm 1 can merge the two clusters only if $\Im_1 \cap \Im_2 \neq \emptyset$.*

The latter remark is a consequence of Lemma 6. We can now proceed with the proof for each one of cases (c1)-(c4).

**(c1):** Since both $C_1$ and $C_2$ have Pc, there exists an intersection $\mathcal{I}_1$ such that, on each path $\mathcal{P}_h$ where $C_1$ or $C_2$ have vehicles,

$$\Im_1 \cap [a_{h,1}, b_{h,1}] \neq \emptyset, \quad \Im_1 \cap [a_{h,k}, b_{h,k}] = \emptyset, \forall\, k \neq 1, \quad (27)$$

and

$$\Im_2 \cap [a_{h,1}, b_{h,1}] \neq \emptyset, \quad \Im_2 \cap [a_{h,k}, b_{h,k}] = \emptyset, \forall\, k \neq 1. \quad (28)$$

Note that the intersection $\mathcal{I}_1$ is necessarily the same for clusters $C_1$ and $C_2$, because of (25) (they interfere through $B_+$). The interval $\Im_{1\cup 2}$ has for left bound the minimum of the left bounds of $\Im_1$ and $\Im_2$, and for right bound the maximum of the right bounds of $\Im_1$ and $\Im_2$. Therefore, (27) and (28) imply that $\Im_{1\cup 2} \subseteq \Im_1 \cup \Im_2 \cup [a_{h,1}, b_{h,1}]$. This means that

$$\Im_{1\cup 2} \cap [a_{h,1}, b_{h,1}] \neq \emptyset, \quad \Im_{1\cup 2} \cap [a_{h,k}, b_{h,k}] = \emptyset, \forall\, k \neq 1.$$

This is true on all paths, therefore $C_1 \cup C_2$ has Pc.

We now prove by contradiction that $C_1 \cup C_2$ has Pd. Let $1, \dots, n$ and $n+1, \dots, n+m$ be the vehicles of $C_1$ and $C_2$ respectively, numbered in topological order. Since $C_1$ and $C_2$ have Pd we can assume, without loss of generality, that $x_1(0), \dots, x_n(0) > x_{n+1}(0), \dots, x_{n+m}(0)$ (i.e., we assume that on $\mathcal{P}_h$ vehicles of $C_1$ precede vehicles of $C_2$). If $C_1 \cup C_2$ doesn't have Pd, there must exist a vehicle $i$ on a $\mathcal{P}_h$ such that $x_n(0) > x_i(0) > x_{n+1}(0)$. Equation (27) implies that $x_n(0) \leq b_{h,1}$, while (28) implies that $x_{n+1}(0) + D_{\text{stop}} + \Delta_{\text{stop}} \geq a_{h,1}$. Thus we can write $x_i(0) \in [a_{h,1} - D_{\text{stop}} - \Delta_{\text{stop}}, b_{h,1}]$, and this in turn implies that

$$\lim_{t\to\infty} x_i(t, u_{\min}) \leq b_{h,1} + D_{\text{stop}} + \Delta_{\text{stop}},$$

while by (10)

$$\lim_{t\to\infty} x_n(t, \bar{u}_n) \geq b_{h,1} + D_{\text{stop}} + \Delta_{\text{stop}}.$$

Thus, Line 6 of Algorithm 1 must have merged vehicle $i$ and cluster $C_1$.

**(c2):** On any path $\mathcal{P}_h$, property Pc for both $C_1$ and $C_2$ can be written as

$$\Im_1 \cap [a_{h,1}, b_{h,1}] \neq \emptyset, \quad \Im_1 \cap [a_{h,k}, b_{h,k}] = \emptyset, \forall\, k \neq 1, \quad (29)$$

and

$$\Im_2 \cap [a_{h,k_1}, b_{h,k_1}] \neq \emptyset, \quad \Im_2 \cap [a_{h,k}, b_{h,k}] = \emptyset, \forall\, k \neq k_1. \quad (30)$$

Let $1, \dots, n$ and $n+1, \dots, n+m$ be the vehicles of $C_1$ and $C_2$ respectively, numbered in topological order. Having Pd we can assume, without loss of generality, that $x_1(0), \dots, x_n(0) > x_{n+1}(0), \dots, x_{n+m}(0)$.

As a first step, we will show that $\mathcal{I}_{k_1}$ in (30) must necessarily be equal to $\mathcal{I}_1$ in (29) (i.e., it must be the same intersection as for cluster $C_2$). We prove this by contradiction using the sparsity condition. Assume that $k_1 \neq 1$, for instance $k_1 = 0$. Let $\tilde{n}_2$ be the number of vehicles of $C_2$ such that $x(0) > a_{h,0} - D_{\text{stop}} - \Delta_{\text{stop}}$. By Lemma 9 we have

$$\lim_{t\to\infty} x_{n+1}(t, \bar{u}_1) \leq b_{h,0} + D_{\text{stop}} + \Delta_{\text{stop}} + (\tilde{n}_2 - 1)(d + \Delta_{\text{stop}}). \quad (31)$$

If $C_1$ and $C_2$ interfere through $B_-$ then $\exists\, t \geq 0 : x_n(t, u_{\min}) - d \leq x_{n+1}(t, \bar{u}_{n+1})$. By Lemma 13 this implies that $\lim_{t\to\infty} x_n(t, u_{\min}) - d \leq \lim_{t\to\infty} x_{n+1}(t, \bar{u}_{n+1})$. Using (31) this gives

$$\lim_{t\to\infty} x_n(t, u_{\min}) \leq b_{h,0} + D_{\text{stop}} + \tilde{n}_2(d + \Delta_{\text{stop}}). \quad (32)$$

Let $\tilde{n}_1$ be the number of vehicles of $C_1$ such that $x(0) \leq a_{h,1} - D_{\text{stop}} - \Delta_{\text{stop}}$, so that $x_{n-\tilde{n}_1}$ is the last vehicle of $C_1$ such that

$$x(0) > a_{h,1} - D_{\text{stop}} - \Delta_{\text{stop}}. \quad (33)$$

There must be at least on vehicle satisfying (33), since $C_1$ has Pc. For vehicles $n, \dots n - \tilde{n}_1 + 1$, $\bar{u}$ is defined as in (8), therefore proceeding as in the proof of Lemma 6 we can show that

$$\lim_{t\to\infty} x_{n-\tilde{n}_1+1}(t, \bar{u}_{n-\tilde{n}_1+1}) \leq x_{n-\tilde{n}_1+1}(0) + D_{\text{stop}} + \Delta_{\text{stop}} \leq a_{h,1}.$$

Line 6 of Algorithm 1 requires $x_{n-\tilde{n}_1}(0) \leq \lim_{t\to\infty} x_{n-\tilde{n}_1+1}(t, \bar{u}_{n-\tilde{n}_1+1})$ to merge the two vehicles in the same cluster, therefore $x_{n-\tilde{n}_1}(0) \leq a_{h,1}$. As a consequence, there are at least

$$N := \tilde{n}_1 + \tilde{n}_2 + 1 \quad (34)$$

vehicles in the interval $(a_{h,0} - D_{\text{stop}} - \Delta_{\text{stop}}, a_{h,1}]$. This will be used, with the sparsity condition, to prove the contradiction.

We now reason inductively, starting from (32), to compute an upper bound to $x_{n-\tilde{n}_1}(t, u_{\min})$. Vehicle $n$ is the last one in $C_1$, therefore $t_n^*$ in (8) is equal to $\tau$, and

$$\lim_{t\to\infty} x_n(t, \bar{u}_n) \leq \lim_{t\to\infty} x_n(t, u_{\min}) + \Delta_{\text{stop}} \leq b_{h,0} + D_{\text{stop}} + \Delta_{\text{stop}} + \tilde{n}_2(d + \Delta_{\text{stop}}). \quad (35)$$

Furthermore, if $i$ and $i-1$ belong to the same cluster, it must be that

$$\lim_{t\to\infty} x_{i-1}(t, u_{\min}) \leq \lim_{t\to\infty} x_i(t, \bar{u}_i) + d, \quad (36)$$

in particular, if $i = n-1$

$$\lim_{t\to\infty} x_{n-1}(t, u_{\min}) \leq \lim_{t\to\infty} x_n(t, \bar{u}_n) + d.$$

The above inequality with (35) gives

$$\lim_{t\to\infty} x_{n-1}(t, u_{\min}) \leq b_{h,0} + D_{\text{stop}} + (\tilde{n}_2+1)(d+\Delta_{\text{stop}}). \quad (37)$$

Now for any $i \in \{n-1, \dots, n-\tilde{n}_1+1\}$ we have $x_i(0) < a_{h,1} + D_{\text{stop}} + \Delta_{\text{stop}}$, therefore $\bar{u}_i$ is defined as in (8). We have

$$\lim_{t\to\infty} x_i(t, \bar{u}_i) \leq \max\Big\{ \lim_{t\to\infty} x_i(t, u_{\min}) + \Delta_{\text{stop}}, \lim_{t\to\infty} x_{i+1}(t, \bar{u}_{i+1}) + d \Big\}, \quad (38)$$

where the first element in the $\max$ corresponds to the case $t_i^* = \tau$ in (8) , while the second element corresponds to the case $t_i^* > \tau$. For $i = n-1$, using (35) and (37), (38) becomes

$$\begin{aligned}\lim_{t\to\infty} x_{n-1}(t, \bar{u}_{n-1}) \leq \\ \max\Big\{ \lim_{t\to\infty} x_{n-1}(t, u_{\min}) + \Delta_{\text{stop}}, \lim_{t\to\infty} x_n(t, \bar{u}_n) + d \Big\} \leq \\ \max\Big\{ b_{h,0} + D_{\text{stop}} + \Delta_{\text{stop}} + (\tilde{n}_2+1)(d+\Delta_{\text{stop}}), \\ b_{h,0} + D_{\text{stop}} + (\tilde{n}_2+1)(d+\Delta_{\text{stop}}) \Big\} = \\ b_{h,0} + D_{\text{stop}} + \Delta_{\text{stop}} + (\tilde{n}_2+1)(d+\Delta_{\text{stop}}). \quad (39)\end{aligned}$$

Repeating the above reasoning for vehicles $n-2, \dots, n-\tilde{n}_1$ we obtain

$$\lim_{t\to\infty} x_{n-\tilde{n}_1}(t, u_{\min}) \leq b_{h,0} + D_{\text{stop}} + (\tilde{n}_1+\tilde{n}_2)(d+\Delta_{\text{stop}}).$$

However, we have by assumption (recall (33)) that

$$\lim_{t\to\infty} x_{n-\tilde{n}_1}(t, u_{\min}) \geq x_{n-\tilde{n}_1}(0) > a_{h,1} - D_{\text{stop}} - \Delta_{\text{stop}},$$

and therefore that

$$a_{h,1} - D_{\text{stop}} - \Delta_{\text{stop}} < b_{h,0} + D_{\text{stop}} + (\tilde{n}_1+\tilde{n}_2)(d+\Delta_{\text{stop}}).$$

Using (34) we can rewrite the inequality above as

$$a_{h,1} - b_{h,0} \leq 2D_{\text{stop}} + \Delta_{\text{stop}} + (N-1)(d+\Delta_{\text{stop}}).$$

This contradicts the sparsity condition in Definition 10, which ensures that

$$a_{h,1} - b_{h,0} > 2D_{\text{stop}} + \Delta_{\text{stop}} + (N-1)(d+\Delta_{\text{stop}}).$$

The above reasoning shows that, assuming the sparsity condition, intersection $k_1$ in (30) must be the same as intersection 1 in (29). We can thus prove that $C_1 \cup C_2$ has Pc and Pd following the same reasoning as in case (c1).

**(c3):** $C_1$ has Pc and $C_2$ has $\text{Pa} \wedge \text{Pb}$, therefore on each path $\mathcal{P}_h$

$$\Im_1 \cap [a_{h,1}, b_{h,1}] \neq \emptyset, \quad \Im_1 \cap [a_{h,k}, b_{h,k}] = \emptyset, \forall\, k \neq 1,$$

while all vehicles of $C_2$ are on a single path $\mathcal{P}_1$ (by Pa) and

$$\Im_2 \cap [a_{1,k}, b_{1,k}] = \emptyset, \forall\, k$$

(by Pb). Since Pd holds for both clusters, either all vehicles in $C_1$ precede all vehicles in $C_2$, or *vice-versa*.

Consider first the case $x_i(0) > x_j(0)$, for all $i \in C_1$ and $j \in C_2$. By Remark 5, if $C_1$ and $C_2$ are merged by Algorithm 1 $\Im_1 \cap \Im_2 \neq \emptyset$; as a consequence,

$$\Im_{1\cup 2} \subseteq \Im_1 \cup \Im_2.$$

Thus, $C_1 \cup C_2$ has Pc.

Consider now the case $x_i(0) < x_j(0)$, for all $i \in C_1$ and $j \in C_2$. A necessary condition to merge the two clusters (recall Remark 4 and the fact that $C_2$ has $\text{Pa}\wedge\text{Pb}$) through Line 6 of Algorithm 1 is that $\{\cup_{i\in C_1, t\geq 0}\ I_i^\tau(t)\} \cap \mathcal{P}_1(\Im_2) \neq \emptyset$, therefore

$$\mathcal{P}_1(\Im_{1\cup 2}) \subseteq I_1^\tau \cup \mathcal{P}_1(\Im_2).$$

Furthermore, we have by Lemma 15 that

$$\{\cup_{i\in C_1, t\geq 0}\ I_i^\tau(t)\} \cap \mathcal{I}_k = \emptyset, \forall\, k \neq 1.$$

Therefore, in all cases $C_1 \cup C_2$ has Pc.

We now prove by contradiction that it also has Pd. Assume $x_i(0) > x_j(0)$, for all $i \in C_1$ and $j \in C_2$ (the proof is identical in the other case). If $C_1 \cup C_2$ did not have Pd, given that $C_1$ and $C_2$ have Pd, there would exist a vehicle $i$ on a $\mathcal{P}_h$ such that $x_n(0) > x_i(0) > x_{n+1}(0)$, where $n$ and $n+1$ are the last vehicle of $C_1$ and the first of $C_2$, respectively. The two clusters are merged by interference through $B_-$, therefore $x_n(t, \bar{u}_n)$ must intersect $x_{n+1}(t, u_{\min})$. As a consequence $x_i(t, u_i)$ must intersect $x_n(t, \bar{u}_n)$ and $x_{n+1}(t, u_{\min})$, for any $\mathbf{u}$. This implies that $i$ should have been merged to $C_1$ or $C_2$ by Algorithm 1.

**(c4):** Both $C_1$ and $C_2$ have Pa and Pb. Since both clusters have property $(\text{Pa} \wedge \text{Pb})$, the vehicles of $C_2$ must necessarily be on the same path $\mathcal{P}$ for $C_1$ and $C_2$ to be merged by Algorithm1, therefore the resulting cluster has Pa.

Without loss of generality, and since property Pd holds, we can assume that $x_i(0) > x_j(0)$, for all $i \in C_1$ and $j \in C_2$.

By Remark 5 the condition $\Im_1 \cap \Im_2 \neq \emptyset$ is necessary to merge the two clusters: if it were not satisfied, any trajectory in $I_{C_2}^\tau$ would reach a stop before even reaching the initial position of the last vehicle in $C_1$. This implies that $\Im_{1\cup 2} \subseteq \Im_1 \cup \Im_2$, therefore $C_1 \cup C_2$ has also property Pb. We can finally prove Pd as in case (c3). ■

## REFERENCES


[1] K. Dresner and P. Stone, "Multiagent traffic management: An improved intersection control mechanism," in *Conference on Autonomous Agents and Multiagent systems*. ACM, 2005, pp. 471–477.

[2] ——, "A multiagent approach to autonomous intersection management," *J. Artif. Intell. Res.*, vol. 31, pp. 591–656, 2008.

[3] J. Lee and B. Park, "Development and evaluation of a cooperative vehicle intersection control algorithm under the connected vehicles environment," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, pp. 81–90, 2012.

[4] R. Tachet, P. Santi, S. Sobolevsky, L. I. Reyes-Castro, E. Frazzoli, D. Helbing, and C. Ratti, "Revisiting street intersections using slot-based systems," *PLOS ONE*, 2016.

[5] J. Lygeros, C. Tomlin, and S. Sastry, "Multi-objective hybrid controller synthesis: Least restrictive controls," in *IEEE Conference on Decision and Control*, 1997.

[6] ——, "Controllers for reachability specifications for hybrid systems," *Automatica*, vol. 35, pp. 349–370, 1999.

[7] M. R. Hafner and D. D. Vecchio, "Computational tools for the safety control of a class of piecewise continuous systems with imperfect information on a partial order," *Syst. Control Lett.*, vol. 49, pp. 2463–2493, 2011.

[8] A. Colombo and D. Del Vecchio, "Least restrictive supervisors for intersection collision avoidance: A scheduling approach," *IEEE Trans. Autom. Control*, vol. 60, pp. 1515–1527, 2015.

[9] C. Tomlin, I. Mitchell, and R. Ghosh, "Safety verification of conflict resolution maneuvers," *IEEE Trans. Intell. Transp. Syst.*, vol. 2, pp. 110–120, 2001.

[10] C. J. Tomlin, I. Mitchell, A. M. Bayen, and M. Oishi, "Computational techniques for the verification of hybrid systems," *Proc. of the IEEE*, vol. 91, pp. 986–1001, 2003.

[11] S. A. Reveliotis and E. Roszkowska, "On the complexity of maximally permissive deadlock avoidance in multi-vehicle traffic systems," *IEEE Trans. Autom. Control*, vol. 55, pp. 1646–1651, 2010.

[12] A. Colombo and D. Del Vecchio, "Efficient algorithms for collision avoidance at intersections," in *Hybrid Systems: Computation and Control*, 2012.

[13] S. M. Loos and A. Platzer, "Safe intersections: At the crossing of hybrid systems and verification," in *IEEE International Conference on Intelligent Transportation Systems*, 2011.

[14] R. Verma and D. Del Vecchio, "Semiautonomous multivehicle safety: A hybrid control approach," *IEEE Robot. Autom. Mag.*, vol. 18, pp. 44–54, 2011.

[15] M. Hafner, D. Cunningham, L. Caminiti, and D. Del Vecchio, "Cooperative collision avoidance at intersections: Algorithms and experiments," *IEEE Trans. Intell. Transp. Syst.*, 2013.

[16] G. Rodrigues de Campos, F. Della Rossa, and A. Colombo, "Optimal and least restrictive supervisory control: safety verification methods for human-driven vehicles at traffic intersections," in *IEEE Conference on Decision and Control*, 2015.

[17] H. Ahn, A. Rizzi, A. Colombo, and D. Del Vecchio, "Experimental testing of semi-autonomous multi-vehicle control for collision avoidance at intersections," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015.

[18] L. Pallottino, V. G. Scordio, A. Bicchi, and E. Frazzoli, "Decentralized cooperative policy for conflict resolution in multivehicle systems," *IEEE Trans. Robot.*, vol. 23, pp. 1170–1183, 2007.

[19] A. Colombo and D. Del Vecchio, "Supervisory control of differentially flat systems based on abstraction," in *IEEE Conference on Decision and Control*, 2011.

[20] E. Dallal, A. Colombo, D. Del Vecchio, and S. Lafortune, "Supervisory control for collision avoidance in vehicular networks using discrete event abstractions," in *American Control Conference*, 2013.

[21] H. Ahn, "Semi-autonomous control of multiple heterogeneous vehicles for intersection collision avoidance," Master's thesis, Massachusetts Institute of Technology, 2014.

[22] E. Dallal, A. Colombo, D. Del Vecchio, and S. Lafortune, "Supervisory control for collision avoidance in vehicular networks using discrete event abstractions," *Discrete Event Dynamic Systems*, vol. accepted, 2016.

[23] T.-C. Au, C.-L. Fok, S. Vishwanath, C. Julien, and P. Stone, "Evasion planning for autonomous vehicles at intersections," in *IEEE/RSJ International conference on Intelligent Robots and Systems*, 2012.

[24] J. Gregoire, S. Bonnabel, and A. De La Fortelle, "Priority-based intersection management with kinodynamic constraints," in *European Control Conference*, 2014, pp. 2901–2907.

[25] Kyoung-Dae Kim and P. R. Kumar, "An MPC-based approach to provable system-wide safety and liveness of autonomous ground traffic," *IEEE Trans. Autom. Control*, vol. 59, pp. 3341–3356, 2014.

[26] A. Colombo, "A mathematical framework for cooperative collision avoidance of human-driven vehicles at intersections," in *International Symposium on Wireless Communication Systems*, 2014, pp. 449–453.

[27] A. Colombo and H. Wymeersch, "Cooperative intersection collision avoidance in a constrained communication environment," in *IEEE International Conference on Intelligent Transportation Systems*, 2015.

[28] R. Hult, G. R. Campos, P. Falcone, and H. Wymeersch, "An approximate solution to the optimal coordination problem for autonomous vehicles at intersections," in *American Control Conference*, 2015.

[29] M. Liebner, M. Baumann, F. Klanner, and C. Stiller, "Driver intent inference at urban intersections using the intelligent driver model," in *Intelligent Vehicles Symposium*, 2012, pp. 1162–1167.

[30] D. Petrich, T. Dang, D. Kasper, G. Breuel, and C. Stiller, "Map-based long term motion prediction for vehicles in traffic environments," in *IEEE International Conference on Intelligent Transportation Systems*, 2013, pp. 2166–2172.

[31] A. Bender, G. Agamennoni, J. R. Ward, S. Worrall, and E. M. Nebot, "An unsupervised approach for inferring driver behavior from naturalistic driving data," *IEEE Trans. Intell. Transp. Syst.*, 2015.

[32] F. Fadaie and M. E. Broucke, "On the least restrictive control for collision avoidance of two unicycles," *Int. J. Robust. Nonlin.*, vol. 16, pp. 553–574, 2006.

[33] L. Bruni, A. Colombo, and D. Del Vecchio, "Robust multi-agent collision avoidance through scheduling," in *IEEE Conference on Decision and Control*, 2013.

[34] H. Ahn, A. Colombo, and D. Del Vecchio, "Supervisory control for intersection collision avoidance in the presence of uncontrolled vehicles," in *American Control Conference*, 2014.

[35] E. Dallal, A. Colombo, D. Del Vecchio, and S. Lafortune, "Supervisory control for collision avoidance in vehicular networks with imperfect measurements," in *IEEE Conference on Decision and Control*, 2013.

[36] D. Bresch-Pietri and D. Del Vecchio, "Estimation for decentralized safety control under communication delay and measurement uncertainty," *Automatica*, vol. 62, pp. 292–303, 2015.

[37] M. R. Hafner, D. Cunningham, L. Caminiti, and D. D. Vecchio, "Automated vehicle-to-vehicle collision avoidance at intersections," in *ITS World Congress*, 2011.

[38] H. Ahn and D. Del Vecchio, "Semi-autonomous intersection collision avoidance through job-shop scheduling," in *Hybrid Systems: Computation and control*, 2016.

[39] D. Angeli and E. D. Sontag, "Monotone control systems," *IEEE Trans. Autom. Control*, vol. 48, pp. 1684–1698, 2003.


**Alessandro Colombo** received the Diplôme D'Ingénieur from ENSTA in Paris in 2005, and the Ph.D. from Politecnico di Milano in 2009. He was Postdoctoral Associate at the Massachusetts Institute of Technology in 2010-2012, and he is currently Assistant Professor in the Department of Electronics, Information and Bioengineering at Politecnico di Milano. His research interests are in the analysis and control of discontinuous and hybrid systems with applications, among others, to transportation networks.

**Gabriel Rodrigues de Campos** received the Ph.D. degree in Automatic Control in 2012 from Grenoble University/Grenoble INP, France. He is currently postdoc with the Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB) at Politecnico di Milano, Italy. Prior to joining the Politecnico di Milano, he was a postdoc with the Department of Signals and Systems, Chalmers University of Technology, Sweden. His research interests include cooperative and distributed control, multi-agent systems, and intelligent transportation systems.

**Fabio Della Rossa** obtained the Master Degree in Mathematical Engineering in 2008 and the Ph.D. degree in Computer Science and Control Engineering in 2011 from Politecnico di Milano, working on Bifurcation methods of complex systems. He spent one year at the Department of Mathematics, Utrecht University, working on numerical continuation methods. He is author of more than 20 papers and the book "Modeling Love Dynamics", World Scientific, 2015. Currently he is assistant professor at Politecnico di Milano.