KTH ROYAL INSTITUTE
OF TECHNOLOGY

Degree Project in Electrical Engineering

Second cycle, 30 credits

Formal Verification of Regional Rules for Connected and Automated Vehicles

KAJ MUNHOZ ARFVIDSSON

Stockholm, Sweden, 2024

Formal Verification of Regional Rules for Connected and Automated Vehicles

KAJ MUNHOZ ARFVIDSSON

Degree Programme in Electrical Engineering
Date: January 24, 2024

Supervisor: Frank J. Jiang
Examiner: Jonas Mårtensson
School of Electrical Engineering and Computer Science
Swedish title: Formell Verifiering av Lokala Regler för Uppkopplade och Automatiserade Fordon

Abstract

In this work, we develop a temporal logic-based safety framework for Autonomous Intersection Management (AIM) to ensure the safety of connected vehicles navigating through intelligent intersections. Traditional AIM systems face challenges in balancing safety and efficiency, often relying on central decision-making to enhance intersection coordination. Our approach addresses this challenge by leveraging formal methods, specifically temporal logic and reachability analysis, to provide rigorous safety guarantees. We begin by specifying the required behavior for vehicles passing through intersections as linear temporal logic formulas. These specifications are then decomposed into a series of Hamilton-Jacobi reachability analyses using temporal logic trees, allowing for the automated verification of intersection behaviors. This method enables the computation of safe time-state corridors for vehicles, facilitating explicit safety-efficiency trade-offs while considering decision uncertainties. Additionally, we determine safe driving limits to ensure vehicles remain within their designated corridors. Our framework is evaluated using simulations of both 3-way and 4-way intersections, demonstrating its ability to verify and enforce safety in real-time across various scenarios.

Keywords

Safety framework, Intelligent Intersections, Connected Vehicles, Traffic Safety, Temporal Logic, Reachability Analysis, Formal Methods

Sammanfattning

I detta arbete presenterar vi ett nytt säkerhetssystem för intelligenta regioner som möter utmaningarna med att koordinera uppkopplade och automatiserade fordon. Intelligenta regioner omfattar områden som korsningar, motorvägar och stadsgator, där trafiken hanteras av intelligenta transportsystem som anpassar sig till realtidsförhållanden, regionala trafikmönster och specifika händelser. Med hjälp av temporallogik och nåbarhetsanalys erbjuder vår metod starka säkerhetsgarantier över olika trafiksituationer. Inledningsvis definierar vi det önskade beteendet för alla fordon som navigerar genom korsningen med hjälp av temporallogiska påståenden i form av en säkerhetsspecifikation. Därefter dekonstrueras specifikationen automatiskt till en sekvens av mindre, hanterbara Hamilton-Jacobi nåbarhetsanalyser genom en nyligen utvecklad beräkningsmetod kallad temporala logikträd. Detta möjliggör automatiserad verifiering av fordonsbeteenden och en explicit avvägning mellan säkerhet och effektivitet. I detta steg tilldelas fordonen färdvägskorridorer som garanterar att både traditionella trafikregler och specifika beteendekrav uppfylls. Vi fastställer även säkra körgränser för att säkerställa att fordonen förblir inom sina tilldelade korridorer. Vårt system demonstreras genom simuleringar av trevägs- och fyrvägs-korsningar, där fordon med potentiellt kolliderande färdvägar garanteras säker passage.

Nyckelord

Intelligenta Korsningar, Uppkopplade och Automatiserade Fordon, Trafiksäkerhet, Temporallogik, Nåbarhetsanalys, Formella Metoder

Contents

Chapter 1

Introduction

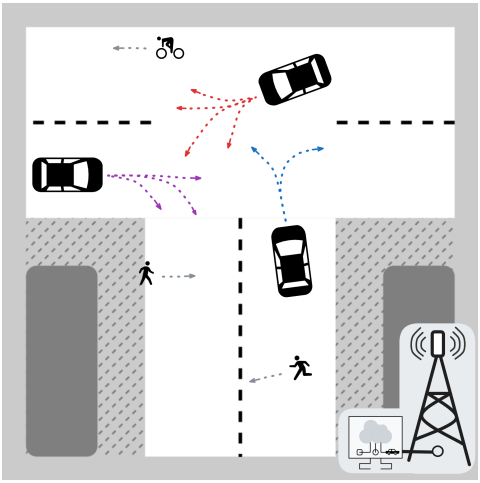


Figure 1.1: An intelligent intersection that collaboratively help road users coordinate safely.

1.1 Overview

Over the last decade, a new generation of Intelligent Transportation Systems (ITS) has begun to emerge, driven by advances in computing and networking [1]. Vehicles are expected to communicate with each other, infrastructure, and the cloud to improve safety, efficiency, sustainability, and passenger comfort [2]. Furthermore, with the rise of edge- and cloud-services, significant effort has been directed into facilitating these objectives using off-board intelligence in the local infrastructure [1, 3]. Autonomous Intersection Management

(AIM) has, in particular, received considerable attention due to the complex challenges that arise at intersections and the limited time that vehicles spend there [4]. Intelligent intersections have shown that they can, even in mixed-traffic, collaborate with partially automated vehicles to act as a safety filter, alerting or possibly overriding the human driver if deemed unsafe [5, 6]. Notably, some researchers state that safety should never be compromised to achieve efficiency [7, 8]. However, although many proposed systems claim such improvements, the trade-off between safety and efficiency still remains an open challenge and an important research direction [9].

Proving that behavior is safe poses major challenges [10]. To this end, formal methods have been used to ensure safe collision avoidance [11], perform automata-based verification of collision zones [12], and construct specification-compliant driving corridors [13]. One of the core challenges is the difficulty in computing the maximal controlled invariant sets for intersections in a general, computationally-tractable way, since finding the exact solution is an NP-complete problem [10]. To address this, several approaches propose approximate solutions to the problems where the maximal controlled invariant set is conservatively approximated and leverage assumptions made about how the vehicles will pass through the intersection [10, 14, 15]. Alternatively, some take a probabilistic approach and provide lower bounds on vehicle collision probabilities in intelligent intersections [16]. While these approaches do provide safety guarantees, they are built upon specific intersection traffic rules. Due to the diverse and evolving requirements of traffic passing through intersections, there is interest to further investigate approaches that are able to provide more flexible safety guarantees that easily adapt to updates to changing intersection traffic rules.

For more flexible safety guarantees in ITS, researchers have recently proposed several approaches based on the formalization of traffic rules. Used in the specification and verification of various types of complex systems [17], temporal logic offers a compelling approach for formalizing requirements on systems in a way that is both flexible and approachable to human designers. For example, [18] shows that they can formalize current German intersection traffic rules using metric temporal logic. Specifically for designing intersection management, [12] use linear temporal logic to specify and verify the safety of an intersection management algorithm. While they do not use temporal logic, [19] similarly develop formal specifications for intersection management by formalizing the responsibility-sensitive safety model [20] using Hoare Logic that can be used for discovering conditions that guarantee safety of the intersection. In this work, we show how to take

intersection rules that are formalized in linear temporal logic and leverage temporal logic trees [21] to directly use Hamilton-Jacobi (HJ) reachability analysis to verify the feasibility of the intersection rules.

1.2 Problem Formulation

Although many of the current approaches to AIM provide complete solutions, they are often difficult to safely extend or adapt as the safety guarantees are typically built on particular design decisions of their intersection management algorithm. Like earlier work, we wish to centralize traffic management to address the challenges that arise at intersections. We also intend to leverage formal methods to ensure specification satisfaction of encoded safe behaviors. However, in contrast to earlier work, we intend to combine these elements with dynamically relevant vehicle models. Explicitly, given a multi-vehicle specification in linear temporal logic for an intersection, we seek to automatically verify its feasibility to guarantee the full specification is satisfied, while considering all of the vehicle's dynamics and decision uncertainty.

1.3 Contribution

The main contribution of this paper is a safety framework that help develop provably safe AIM systems for intelligent intersections. In summary, this paper's contributions are three-fold:

1. we present a linear temporal logic-based sequential path planning approach for intelligent intersections that allow AIMs to take conscious design decisions about the trade-off between safety and efficiency,

2. we design a driving limits service for the intelligent intersection that further facilitate the development of safe AIMs,

3. we evaluate the practical feasibility of the safety framework in multiple scenarios of a 3-way and 4-way intersections.

To achieve strong safety guarantees, noted as a requirement by [7, 8], our approach formalizes a safety specification using logic statements. Taking inspiration from the sequential path planning approaches developed for aerial vehicles in [22, 23], the contributed approach starts with formalizing an intersection specification with, specifically, linear temporal logic formulae.

Then, by using temporal logic trees for the verification of these formulae, we develop an approach that results in both safety guarantees for the intelligent intersection and is also able to automatically handle any changes to the specifications. By computing the temporal logic trees with Hamilton-Jacobi (HJ) reachability analysis, we also enabling efficient, real-time computation of safe acceleration sets [24]. Although the computational complexity of these methods have previously been impractical, preliminary indications now show that computation time is fast enough for real systems. By providing these features, which are both important and practically necessary for intelligent intersections, we help accelerate the development of AIM and vehicle automation generally.

Chapter 2

Background

2.1 Preliminaries

This section outlines the preliminary material for our intelligent intersection framework. We start by introducing the intersection model and system dynamics. Then, we use temporal logic as a way to formulate behavior specifications. Finally, for verification, we employ reachability analysis to check specification satisfaction.

2.1.1 Intersection Model

Since the full system includes all road users in the intelligent region, we will use a multi-vehicle model to describe the full system dynamics. Generally, we do not require road users to be modelled equally. However, for simplicity, in this work we assume that all road users are vehicles with the same single-vehicle model.

For vehicle i, we define its state $z_i = [x_i, y_i, \theta_i, \delta_i, v_i]^\top$, where $x_i, y_i, \theta_i, \delta_i$ and v_i represent the vehicle's x-position, y-position, heading angle, steering angle, and velocity, respectively. The vehicle's input is denoted as $u_i = [s_i, a_i]^\top$, where s_i is the steering rate and a_i is the acceleration. We write its dynamics as a control-affine system

$$\dot{z}_i = f_i(z_i) + g_i(z_i)u_i, \tag{2.1}$$

where

$$f_i(z_i) = \begin{bmatrix} v_i \cos\theta_i \\ v_i \sin\theta_i \\ \frac{v_i \tan\delta_i}{L_i} \\ 0 \\ 0 \end{bmatrix}, \; g_i(z_i) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

In this model, L_i is the wheelbase length of the vehicle, $z_i \in \mathbb{R}^5$ and $u_i \in \mathbb{U} \subset \mathbb{R}^2$. The dynamics $\dot{z}_i$ is uniformly continuous, bounded, and Lipschitz continuous in x_i for fixed u_i. The control function $u_i(\cdot)$ is measurable and, provided $u_i(t)$ remains within $\mathbb{U}$ for all t, then $u_i(\cdot)$ belongs to the function space U which contains all admissible control functions. The trajectory of vehicle i is $\zeta_i(\cdot; z_{i,0}, t_0, u_i(\cdot))$, starting from the initial state $z_{i,0}$ at t_0 under $u_i(\cdot)$. For brevity, $\zeta_i(\cdot)$ may be used to indicate the trajectory of vehicle i.

To study the behavior of multiple vehicles at an intersection, a multi-vehicle model is developed. In an intersection where there are N vehicles, the full multi-vehicle state is $z = [z_1, \ldots, z_N]^\top$. The collective input and disturbance are written as $u = [u_1, \ldots, u_N]^\top$ and $w = [w_1, \ldots, w_N]^\top$. Following (2.1) we write the multi-vehicle dynamics

$$\dot{z} = f(z) + g(z)u. \tag{2.2}$$

Here, the self-dynamics comprises each individual vehicle, $f(z) = [f_1(z_1), \ldots, f_N(z_N)]^\top$ and, we similarly aggregate $g(u) = [g_1(u_1), \ldots, g_N(u_N)]^\top$. Then, $\zeta(\cdot; z_0, t_0, u(\cdot), w(\cdot))$ is the full multi-vehicle trajectory. For brevity, this trajectory may be simply referred to as $\zeta(\cdot)$.

In this work, we will primarily direct our attention to sets of states in the multi-vehicle system, rather than vehicles' individual states. This approach enables us to consider groups of trajectories collectively. Specifically, our analysis will involve sets of states that incorporate a temporal dimension. This is particularly useful when assessing the feasibility of intersections scenarios. Let us denote the entire state space of the multi-vehicle system as $\mathbb{S}$. We represent time-state sets for this system as $\mathcal{A} \subseteq \mathbb{S} \times \mathbb{R}$. To retrieve state sets at specific times, we introduce time-state set maps $\Omega : \mathbb{R} \to \mathbb{S}$. That is, for a given time-state set $\mathcal{A}$, there is an associated map $\Omega_{\mathcal{A}}$ where

$$\mathcal{A} = \bigcup_{t \in \mathbb{R}} \{(z, t) \mid z \in \Omega_{\mathcal{A}}(t)\}. \tag{2.3}$$

Similarly, we introduce the associated time window $T_{\mathcal{A}} = [t_a, t_b]$ where

$$t_a = \inf\{t \in \mathbb{R} \mid \Omega_{\mathcal{A}}(t) \neq \varnothing\} \text{ and}$$
$$t_b = \sup\{t \in \mathbb{R} \mid \Omega_{\mathcal{A}}(t) \neq \varnothing\}.$$

Finally, if $\{(z,t) \in \mathcal{A} \mid \exists t' \in T_{\mathcal{A}}, t' \neq t, z \notin \Omega_{\mathcal{A}}(t')\}$ is non-empty, we say that $\mathcal{A}$ is invariant over the time window $T_{\mathcal{A}}$. The time-state sets, their corresponding time-state set maps, associated time windows and the invariant property will all be important for atomic propositions and the following reachability analysis.

2.1.2 Linear Temporal Logic

To ensure safe navigation through the region, we will define temporal logic specifications of permissible behaviors for vehicles. This allows us to create high-level, human-readable requirements on how vehicles should be moving through the intersection. For example, in Fig. 1.1, we might specify the simple requirements "vehicle should turn left and avoid collisions with other vehicles". Using temporal logic, one can write a formal equation representing this requirement using operators that correspond to intuitive concepts in human language, i.e. "always stay below 50 km/h", "always stay in lane", "eventually turn left", or "eventually exit intersection with 30 km/h" (more examples listed in Fig. 2.2). Others have leveraged the intuitive and rich specification capability of temporal logic in a variety of transportation problems [18, 25, 26]. In this work, we work with Linear Temporal Logic (LTL) since it yields the benefits of temporal logic, while being simple to work with and understand. Specifically, we use the operators $\{\neg, \vee, \wedge, \mathsf{U}, \Diamond, \Box\}$, which correspond to the Boolean operators "not", "or", "and", and the temporal operators "until", "eventually", and "always", respectively. We note that similar methods to the one we present in this work can be applied to Signal Temporal Logic by adapting the work with the approach presented in [27].

An LTL formula is constructed using a finite set of atomic propositions, $\mathcal{AP}$, of both logic and temporal operators [17]. The syntax of LTL can be described by the following:

$$\varphi ::= \mathit{true} \mid p \in \mathcal{AP} \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \,\mathsf{U}\, \varphi_2. \tag{2.4}$$

With the negation and conjunction operators, we can define the disjunction operator, $\varphi_1 \vee \varphi_2 = \neg(\neg\varphi_1 \wedge \neg\varphi_2)$. Using the "until" operator, we can further define two more temporal operators: (1) "eventually", $\Diamond\varphi = \mathit{true}\,\mathsf{U}\,\varphi$ and

(2) "always", $\square\varphi = \neg\Diamond\neg\varphi$. In this work, in which we use continuous time models, we exclude the discrete time operator $\bigcirc$. The semantics of LTL in the context of multi-vehicle system trajectories are defined accordingly.

Definition 2.1.1 (LTL Semantics)**.** *For an LTL formula φ, a multi-vehicle trajectory $\zeta(\cdot)$, and a time instant check: $t \geq t_0$, the satisfaction relation $(\zeta(\cdot), t) \models \varphi$ is defined as*

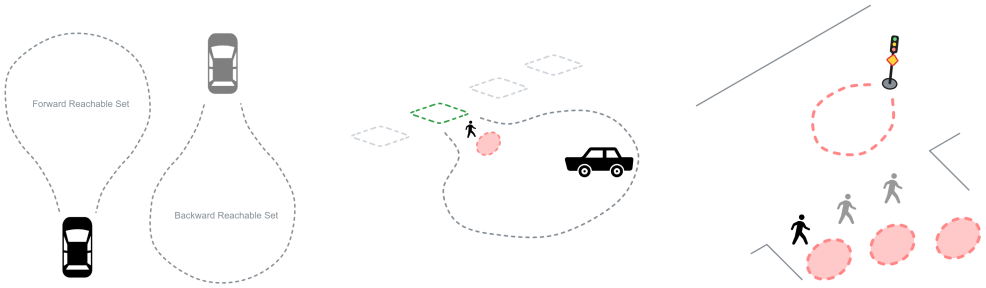
$$
\begin{aligned}
&(\zeta(\cdot), t) \models p \in \mathcal{AP} \Leftrightarrow p \in l(\zeta(t)),\\
&(\zeta(\cdot), t) \models \neg\varphi \Leftrightarrow (\zeta(\cdot), t) \not\models \varphi,\\
&(\zeta(\cdot), t) \models \varphi_1 \wedge \varphi_2 \Leftrightarrow (\zeta(\cdot), t) \models \varphi_1 \wedge (\zeta(\cdot), t) \models \varphi_2,\\
&(\zeta(\cdot), t) \models \varphi_1 \vee \varphi_2 \Leftrightarrow (\zeta(\cdot), t) \models \varphi_1 \vee (\zeta(\cdot), t) \models \varphi_2,\\
&(\zeta(\cdot), t) \models \varphi_1 \,\mathsf{U}\, \varphi_2 \Leftrightarrow \exists t_1 \in [t, \infty) \textit{ s.t.}\\
&\qquad\qquad\qquad\qquad \begin{cases} (\zeta(\cdot), t_1) \models \varphi_2, \\ \forall t_2 \in [t, t_1), (\zeta(\cdot), t_2) \models \varphi_1, \end{cases}\\
&(\zeta(\cdot), t) \models \Diamond\varphi \Leftrightarrow \exists t_1 \in [t, \infty), \textit{ s.t. } (\zeta(\cdot), t_1) \models \varphi,\\
&(\zeta(\cdot), t) \models \square\varphi \Leftrightarrow \forall t_1 \in [t, \infty), \textit{ s.t. } (\zeta(\cdot), t_1) \models \varphi,
\end{aligned}
$$

where p is an atomic proposition and $l(\cdot)$ is a labeling function defined as $l : \mathbb{S} \times \mathbb{R} \to 2^{\mathcal{AP}}$. For a proposition p, we define the following function for relating the proposition to a state set: $\mathcal{L}^{-1}(p) = \{z \in \mathbb{S} \times \mathbb{R}\}$. Using $l(\cdot)$ and $\mathcal{L}^{-1}(\cdot)$, we are able to associate sets in the multi-vehicle state space with atomic propositions.

2.1.3 Reachability Analysis

After writing a behavior specification, we wish to automatically verify its satisfaction. For this, we will employ reachability analysis, a method to analyze how the system evolves. In reachability analysis, we ask: "Which states can the system reach given a certain starting point?" With Fig. 2.1, we highlight three common concepts. For this work, we will use the following kinds of reachable sets:

Definition 2.1.2 (Robust Control Invariant Set)**.** *For the full multi-vehicle system (2.2), $\mathcal{A} \subseteq \mathbb{S} \times \mathbb{R}$ is said to be a robust control invariant set (RCIS) over a time horizon T if, for all $(z, t) \in \mathcal{A}$, there $\exists u(\cdot) \in U$ such that $\forall \tau \in [t, T] : \zeta(\tau; z, t, u(\cdot)) \in \mathcal{A}$.*



(a) Forward and Backward Reachable Sets.

(b) Maximal and Minimal Reachable Sets.

(c) Reachable Sets and Reachable Tubes

Figure 2.1: Common concepts when working with reachability analysis. (a) By following trajectories forward or backward in time we can collect states which we can reach from a particular target or states from which reach the target. (b) While following trajectories, we can choose to look at the reachable space using any or all control inputs. This corresponds to choosing $\exists u(\cdot)$ or $\forall u(\cdot)$ in Definitions 2.1.3 and 2.1.4. (c) Some targets depend on time and others don't. With reachable sets, we follow trajectories that will reach the target at a specific time. In contrast, with reachable tubes, we follow trajectories that reach the target at any given time.

Definition 2.1.3 (Backward Reachable Set)**.** *Given the full multi-vehicle system* (2.2)*, computation time horizon T, a target set $\mathcal{G} \subseteq \mathbb{S} \times \mathbb{R}$ and a constraint set $\mathcal{K} \subseteq \mathbb{S} \times \mathbb{R}$, we define the backward reachable set (BRS) as*

$$\begin{aligned}\mathcal{R}_B(\mathcal{G},\mathcal{K}) = \{(z,t) \mid \exists u(\cdot) \in U,& \\ \exists \tau \in [t,T),\ \zeta(\tau; z, t, u(\cdot)) \in \Omega_{\mathcal{G}}(\tau),& \\ \forall \tau' \in [t,\tau),\ \zeta(\tau'; z, t, u(\cdot)) \in \Omega_{\mathcal{K}}(\tau')\}.&\end{aligned}$$

Namely, $\mathcal{R}_B(\mathcal{G},\mathcal{K})$ is the set of states from which the system is able to reach the target $\mathcal{G}$ while respecting the constraints $\mathcal{K}$.

Definition 2.1.4 (Forward Reachable Set)**.** *Given the full multi-vehicle system* (2.2)*, computation time horizon T, a target set $\mathcal{G} \subseteq \mathbb{S} \times \mathbb{R}$ and a constraint set $\mathcal{K} \subseteq \mathbb{S} \times \mathbb{R}$, we define the forward reachable set (FRS) as*

$$\begin{aligned}\mathcal{R}_F(\mathcal{G},\mathcal{K}) = \{(z,t) \mid \exists u(\cdot) \in U,\ \exists (z_0,t_0) \in \mathcal{G}& \\ \exists t \in [t_0,T),\ \zeta(t; z_0, t_0, u(\cdot)) = z,& \\ \forall \tau \in [t_0,t],\ \zeta(\tau; z_0, t_0, u(\cdot)) \in \Omega_{\mathcal{K}}(\tau)\}.&\end{aligned}$$

Namely, $\mathcal{R}_F(\mathcal{G},\mathcal{K})$ is the set of states that the system can reach when it starts at the target $\mathcal{G}$ and respects the constraints $\mathcal{K}$.

2.1.4 Temporal Logic Trees

Once we have specified an LTL specification for our multi-vehicle model, we need to check the feasibility of the specification. In this work, we will perform these computations using a computational model called Temporal Logic Trees [21] (example illustrated in Fig. 2.2). Intuitively speaking, the leaf nodes of the temporal logic tree are the goals of the multi-vehicle system. From the goals, we perform a series of reachability analyses to find the joint set of feasible trajectories that satisfy the intersection specification. When the computation finishes, if the TLT has been successfully constructed, we know the intersection specification is feasible and possible to satisfy. This verification result is detailed in [21, Theorem V.1]. By using $\mathcal{R}(\cdot)$ and $\mathcal{RCI}(\cdot)$, we are able to fully construct temporal logic trees. For more details, we refer readers to [21].

Definition 2.1.5 (Temporal Logic Tree)**.** *A temporal logic tree (TLT) is a tree for which*

- *each node is either a state set node, a subset of* $\mathbb{R}^{n_z}$*, or an operator node, from* $\{\wedge, \vee, \mathsf{U}, \Box\}$*;*
- *the root node and the leaf nodes are state set nodes;*
- *if a set node is not a leaf node, its unique child is an operator node;*
- *the children of any operator node are set nodes.*

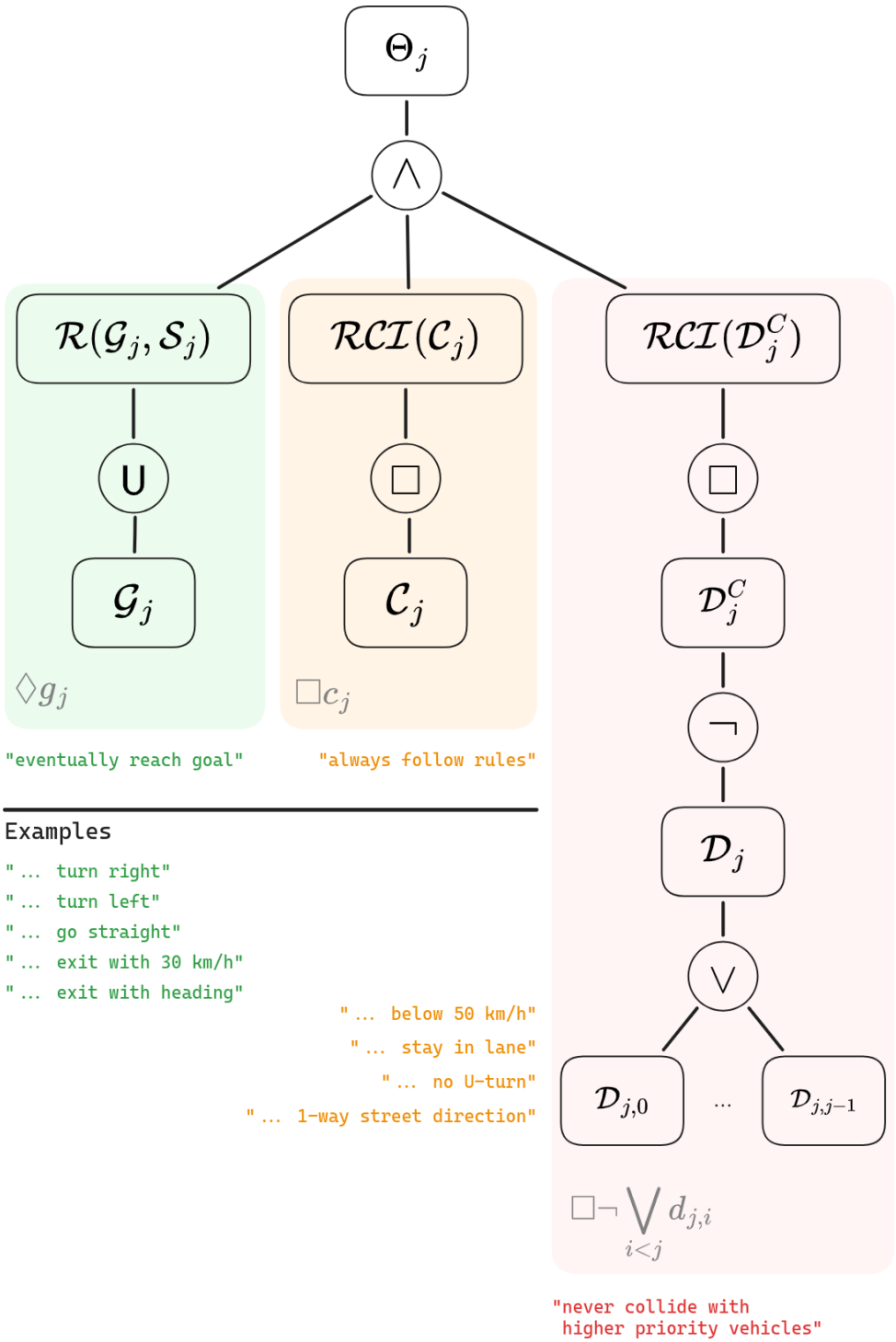


Figure 2.2: Illustration of the constructed TLT for the jth vehicle's individual LTL specification (3.1) that ensures safety at intersections.

2.2 Motivation

In this work, we are interested in developing an approach for guaranteeing that vehicles are safely coordinated through intelligent intersections. For example, in Fig. 1.1, we illustrate a 3-way intersection example. The three vehicles have reached the intersection at a similar time, posing a potential risk of collision. For each of these vehicles, we can specify their overall behavior using their corresponding LTL formulae:

$$\begin{aligned}\varphi_{\text{red}} &= \varphi_{\text{left}} \wedge \varphi_{\text{safe}},\\ \varphi_{\text{blue}} &= \varphi_{\text{left}} \wedge \varphi_{\text{safe}},\\ \varphi_{\text{purple}} &= \varphi_{\text{straight}} \wedge \varphi_{\text{safe}}.\end{aligned}$$

These LTL formulae reflect each vehicle's individual specification of passing through the intersection while staying safe. Then, the specification for the intelligent intersection itself can be described by the following multi-vehicle LTL formula:

$$\varphi = \varphi_{\text{red}} \wedge \varphi_{\text{blue}} \wedge \varphi_{\text{purple}}. \tag{2.5}$$

This simple example is representative of the primary safety challenge of intersections: how can vehicles pass through the intersection while avoiding collisions? For the rest of the work, we will refer to this challenge as the "intersection safety challenge." As was mentioned earlier, this challenge is addressed and solved by previous works. Thus, much like [12, 18], we seek to leverage the richness of temporal logic to develop a safety framework that can solve for solutions to the intersection safety challenge in a way that can be easily adapted and built upon. Moreover, this allows us to create high-level, human-readable requirements that correspond to intuitive concepts such as "always stay below 50 km/h", "always stay in lane", "eventually turn left", or "eventually exit intersection with 30 km/h" (more examples listed in Fig. 2.2).

At the same time, we need to ensure that the admissible behaviors of individual vehicles are not too conservative for the full intersection. For example, consider a vehicle's entire collection of safe trajectories, under some safety specification such as φ_{red}, at the intersection show in Fig. 1.1. Although 2.5 would ensure safety of all managed vehicles, it could still be overly conservative with respect to φ_{blue} and φ_{purple} by prioritizing the red vehicle's free movement. Imagine, for instance, the red vehicle driving very slowly through the intersection. This behavior is sometimes necessary as it might be the only safe way through the area. However, when unnecessary, this behavior should not be allowed as it obstructs following vehicles and

reduces intersection throughput. In essence, this is a trade-off relating to how permissive the intelligent intersection is designed to be. Like with the intersection safety challenge, throughout the rest of this work, we will refer to this as "the problem of permissible planning." When addressing the main problem formulation of this work, see Section 1.2, we are further motivated by the problem of permissible planning and how intelligent intersections should ensure safety, yet make conscious design decisions about the trade-off between safety and efficiency.

Chapter 3

The Local Traffic Management System

In this chapter, we develop the Local Traffic Management System (LTMS). We first outline the high-level system design and our approach to finding solutions to the intersection safety challenge described in Section 2.2. Then, we present the computational methods developed for the LTMS to produce guaranteed-safe time-state corridors and driving limits. Additionally, we address some challenges that arise when employing those methods for traffic management at intersections.

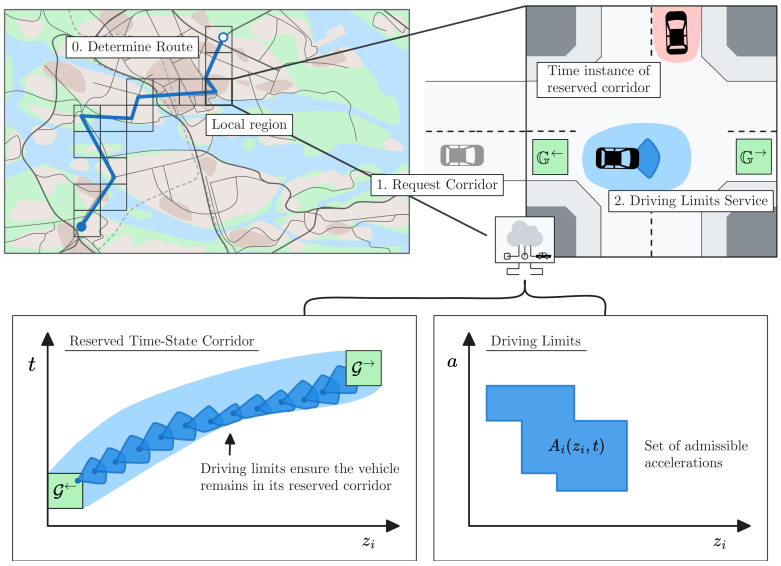


Figure 3.1: Conceptual design of the Local Traffic Management System.

3.1 System Design

While acting between high-level city-scale route planning and on-board control, the LTMS performs central planning and coordination of vehicles at local regions. As illustrated in Fig. 3.1, it accounts for the safe operational limits of all involved vehicles while ensuring they reach their destination. Specifically, for an approaching vehicle j, we suggest the following:

1. The vehicle notifies the LTMS that it will enter the region within time window $T_j^{\leftarrow}$ at state set $\mathbb{G}_j^{\leftarrow}$. Together with a requested exit $\mathbb{G}_j^{\rightarrow}$, this is necessarily communicated to pass through the region.

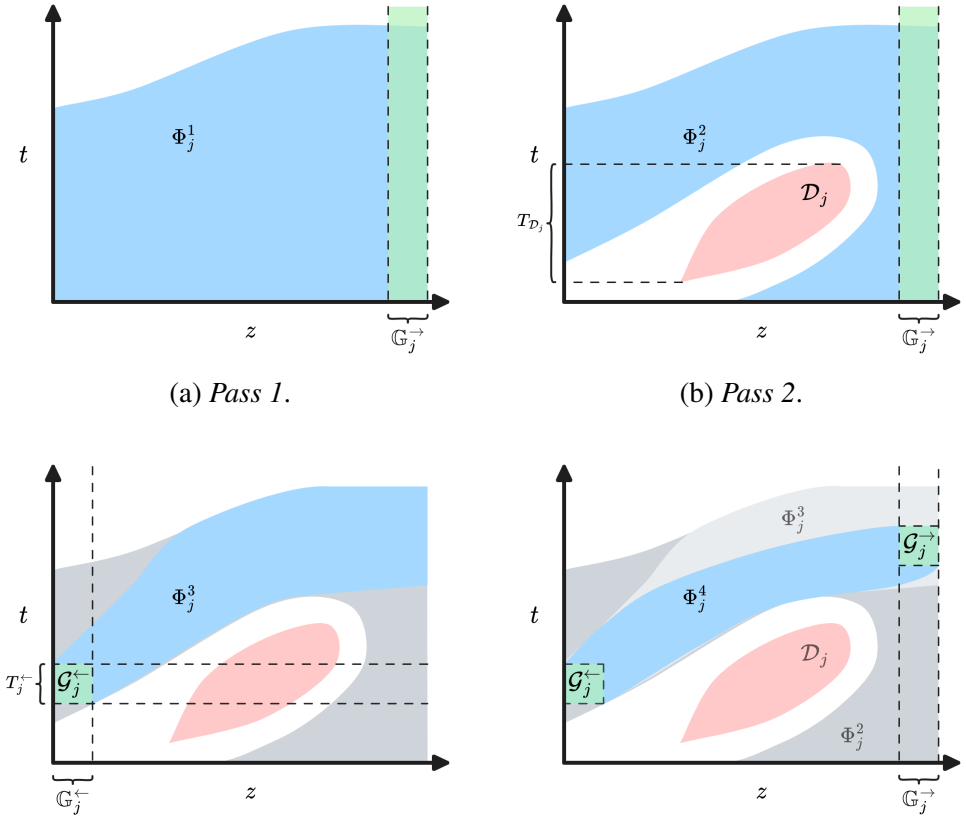
2. Taking already scheduled vehicles into account, a safety analysis inspired by the method developed in [22] produces a safe time-state corridor for the approaching vehicle j.

3. Now, to address the problem of permissible planning, the trajectories are further pruned such that only those satisfying the entry and exit conditions imposed by $T_j^{\leftarrow}$, $\mathbb{G}_j^{\leftarrow}$ and $\mathbb{G}_j^{\rightarrow}$ remain. Additionally, by following these trajectories, that start sometime in $T_j^{\leftarrow}$, we implicitly get an exit time window $T_j^{\rightarrow}$.

4. These results are then stored for as long as vehicle j is in the region. Using them, the LTMS computes safe control bounds, such as acceleration limits. If followed, these ensure that vehicle j stays within the reserved corridor.

Steps 2. and 3. are particularly important and will form the main body of this thesis. We will divide our safety analysis and planning into four "passes". To explain these, Fig. 3.2 conceptually illustrates the ideas we will present in this chapter. At a high level, the goal is to compute the reserved time-state corridor as shown in Fig. 3.1.

Notably, with the proposed level of abstraction, the procedure becomes general enough to use with any type of environment or road topology. Moreover, while we will not address inter-region management in this thesis, we suggest that our approach provides the necessary interfaces for composition. That is, when going between regions A and B that overlap in $\mathbb{G}^{AB}$, then the handover could be done in $T^{AB} \times \mathbb{G}^{AB}$, where T^{AB} is the time window for exiting region A and entering region B. Since entry time drives exit time, $T^{AB} \subseteq T^{A\rightarrow}$. However, to discuss inter-regional aspects, we must first formalize our approach for an individual region which, in this work, will be single intersections.



(a) *Pass 1*.

(b) *Pass 2*.

(c) *Pass 3*.

(d) *Pass 4*.

Figure 3.2: Conceptual illustration of the four passes in our analysis. (a) shows Φ_j^1, the result of *Pass 1*, which ensures conformity to invariant constraints, such as road geometries or static obstacles. This is illustrated by its shape on the top. (b) shows Φ_j^2 when non-invariant constraints are added in the form of $\mathcal{D}_j$. (c) shows Φ_j^3 in the first step of refining the trajectories. After this pass, all remaining trajectories that are embedded in Φ_j^3 satisfy the entry conditions $\mathcal{G}_j^{\leftarrow}$. (d) shows Φ_j^4 after it has similarly pruned trajectories even further, now also considering exit conditions $\mathcal{G}_j^{\rightarrow}$.

3.2 Ensuring Safety at Intersections

In this section, we develop our approach to finding solutions to the intersection safety challenge described in Section 2.2. We start by posing the intersection safety challenge as a sequential path planning problem, inspired by the method developed in [22]. Then, we formalize the sequential path planning problem into LTL formulae. Finally, after we have obtained an LTL formulae, we detail how to use TLT to compute satisfaction sets for the sequential path planning specification using reachability analysis.

3.2.1 Sequential Path Planning for Intersections

As the basis of our approach to solving the intersection safety challenge, we propose the formalization of the Sequential Path Planning (SPP) method, which is outlined in [22, 23]. SPP is a structured approach for path planning in multi-vehicle scenarios. As suggested by the name, the key idea in SPP is to plan the paths of vehicles sequentially, prioritizing them based on a predefined order. In this method, when a higher priority vehicle i plans its path, it does so without considering subsequent vehicles. Since the path planning is sequential, when planning for a lower priority vehicle j, where $i < j$, all admissible trajectories of vehicle i are already known. Consequently, vehicle i can be reserved in space and time, making it a known and deterministic obstacle for vehicle j. The path planning problem for vehicle j is then solved by computing the backward reachable set from a single-vehicle target set. After computing the backward reachable set, similar to Definition 2.1.3, we have a set that includes states from which vehicle j can reach its target within a specified time frame while avoiding all obstacles. To make SPP more extensible and easily adaptable, we specify an LTL formulae that we will be able to leverage to create new intersection rules.

3.2.2 LTL Specification of Sequential Path Planning

For specifying LTL formulae for SPP, we start by outlining the required specifications. First, we would like the vehicles to eventually reach their targets. Second, while they reach their targets, they should adhere to the traffic rules (speed limits, lane restrictions, etc.) in the intersection. Finally, while they reach their targets, they should also avoid colliding with other vehicles. As is done in SPP, instead of asking that the vehicles should avoid all other vehicles, we specify that it is enough that the vehicles only avoid the vehicles

that have higher priority. We can write an LTL formula that collectively covers the listed specifications for an individual vehicle j:

$$\varphi_j = \Diamond g_j \wedge \Box c_j \wedge \Box \neg \bigvee_{i<j} d_{j,i}. \tag{3.1}$$

Here, $\Diamond g_j$ corresponds to the requirement that vehicle j should eventually reach its goal, which is denoted by goal time-state set $\mathcal{G}_j \subseteq \mathbb{S} \times \mathbb{R}$. By including specific parts of vehicle j's state space in the goal set, a variety of goals can be represented, such as turning right, turning left, going straight, exiting the intersection with a specific speed, or exiting the intersection with a specific heading (as is listed in green in Fig. 2.2). Then, $\Box c_j$ corresponds to the requirement that vehicle j should follow traffic rules, which means its trajectories stays within a time-state constraint set $\mathcal{C}_j \subseteq \mathbb{S} \times \mathbb{R}$. Similarly to the goal set, a variety of traffic rules can be expressed through the constraint set, such as staying below the speed limit, staying in a specific lane while passing through the intersection, not allowing U-turns, and enforcing one-way street directions (as is listed in orange in Fig. 2.2). Finally, the last term, $\Box \neg \bigvee_{i<j} d_{j,i}$, corresponds to the requirement that vehicle j avoids collisions with higher-priority vehicles. The proposition $d_{j,i}$ corresponds to the danger time-state set $\mathcal{D}_{j,i} \subseteq \mathbb{S} \times \mathbb{R}$, which are states where vehicle j is able to collide with a higher priority vehicle i. Notably, the highest priority vehicle ($i = 1$) is not required to avoid any other vehicle. To handle this, we include a virtual vehicle, $i = 0$, which cannot be collided with. Consequently, for any vehicle j, $\mathcal{D}_{j,0} = \varnothing$ and $d_{j,0} = \mathit{false}$.

We note that the construction or computation of $\mathcal{D}_{j,i}$ is critically important to the efficiency of the intersection. When $\mathcal{D}_{j,i}$ is large, (3.1) will result in vehicle j driving more conservatively and, in turn, less efficiently. To maximize the efficiency of the intersection, $\mathcal{D}_{j,i}$ should closely follow the true trajectory of vehicle i. However, the less conservative $\mathcal{D}_{j,i}$ is, the higher the risk that vehicle i will accidentally leave $\mathcal{D}_{j,i}$. In other words, the computation of $\mathcal{D}_{j,i}$ introduces an important trade-off between safety and efficiency. The integration and development of computational approaches to computing $\mathcal{D}_{j,i}$ will be addressed in Section 3.3.

3.2.3 Connecting LTL to Reachability Analysis

We will now aim to bridge the gap between the LTL specification in (3.1) and the subsequent reachability analyses by constructing the TLT shown in Fig. 2.2. We keep in mind that the collective objective is to satisfy $\varphi = \bigwedge_j \varphi_j$,

yet the actual analyses will be made sequentially for each vehicle j's objective φ_j.

A. Computing Temporal Logic Trees

As indicated by Fig. 2.2, the leaf nodes in the TLT represent the goal proposition g_j, the state constraint proposition c_j and the collision proposition $d_{j,i}$, respectively. Through the use of the target, state constraint, and collision propositions, we are able to freely encode and adapt different desired behaviors for the intersection. Once the specification is designed, the construction of the TLT proceeds by computing the reachable tubes $\mathcal{R}_B(\cdot)$ and $\mathcal{RCI}(\cdot)$ underlying the "until" (U) and the "always" ($\Box$) temporal operators, respectively. The "eventually" operator is a special case of the "until" operator and, thus, is also captured by computing $\mathcal{R}_B(\cdot)$. Then, the presence of Boolean operators "not" ($\neg$), "or" ($\vee$), and "and" ($\wedge$) correspond to applying set complements, union, and intersection, respectively.

The application of the set operations underlying the Boolean operators is well-known and often exact. However, when an intersection is naively applied to two reachable tubes, this can result in an approximation error. This is due to the fact that the two reachable tubes may have targets or objectives that are conflicting and are not possible to simultaneously satisfy. This problem is sometimes known as the "leaking corner problem" [24, 28, 29]. We will refer to it as the "conflicting objectives problem" in this work. We address this conflicting objectives problem by recomputing the reachable tube of lower priority vehicles starting from the point in time where their tube intersects with the danger time-state set of higher priority vehicles. In the rest of this section, we detail the computation of the reachable tubes necessary for these SPP LTL formulae and the different computational techniques we use to reduce computational costs and avoid the conflicting objectives problem.

B. Reachability Analysis for Intersections

For intelligent intersection specifications, we construct temporal logic trees using HJ reachability analysis. The computational approach we use for HJ reachability analysis is a powerful approach that is based on finding the viscosity solution to a Hamilton-Jacobi-Isaacs Variational Inequality (HJI VI). For more details about this approach, we refer readers to [30]). HJ reachability analysis is especially beneficial for addressing the intersection safety challenge due to it's ability to easily handle the nonlinear dynamics of vehicles and non-convex road geometries. Moreover, the resultant value functions from

HJ reachability analysis can be used to efficiently compute the acceleration and steering rate limits for each vehicle [24].

C. Simplifying the Analysis

In search of reducing the computational cost further, we find that for our particular problem, we can combine two temporal statements into one. Specifically, a common case for intersections is that when vehicles reach their goals in the intersection, it is the same as leaving the intersection. This allows us to represent the satisfaction of $\Diamond g_j \wedge \Box c_j$ with the single reachable tube $\mathcal{R}(\mathcal{G}_j; \mathcal{C}_j)$. Normally, this reachable tube only represents the satisfaction of $c_j \,\mathsf{U}\, g_j$. However, since in the case where vehicles leave the intersection when they reach their goal, to satisfy $\Diamond g_j$, we only need to keep track of trajectories that stop at $\mathcal{G}_j$ and do not need to worry about trajectories that will leave $\mathcal{G}_j$ afterward. We find the same idea applies for $\Box\neg \bigvee_{i<j} d_{j,i}$ in the full specification (3.1). That is, we simplify (3.1) to

$$(c_j \wedge \neg \bigwedge_{i<j} d_{j,i}) \,\mathsf{U}\, g_j \tag{3.2}$$

to combine temporal operators for computational reasons, which means the reachability analysis becomes

$$\mathcal{R}_B(\mathcal{G}_j; \mathcal{C}_j \cap \mathcal{D}_j^C). \tag{3.3}$$

In the next section, we will detail how to make this approach more practical for traffic management.

3.3 Traffic Management at Intersections

In this section, we describe our approach to computing (3.3). Finally, we demonstrate using a driving limits service how these corridors can provide safety filters for intelligent intersections.

3.3.1 *Pass 1* and *Pass 2*

While we formally wish to compute (3.3), higher priority vehicles will not necessarily interact with vehicle j at all times. When higher priority vehicles are not present, then computing $\mathcal{R}(\mathcal{G}_j; \mathcal{C}_j)$ is sufficient to verify (3.2). If $\mathcal{G}_j$ and $\mathcal{C}_j$ are invariant, we can improve performance by pre-computing the

corresponding TLT subtrees for $\Diamond g_j \wedge \Box c_j$ in Fig. 2.2 offline. That is, $\mathcal{D}_j$ will often only impose constraints during a certain time window $T_{\mathcal{D}_j} = [t_a, t_b]$ during which vehicle j interact with higher priority vehicles. Consequently, for any $t > t_b$ it is sufficient to verify $\Diamond g_j \wedge \Box c_j$ since $\mathcal{D}_j$ will be empty and, consequently, $\Box \neg d_{j,i}$ will be true. This in turn simplifies (3.3) to $\mathcal{R}_B(\mathcal{G}_j; \mathcal{C}_j)$, and since lane geometries, traffic rules and other state constraints that constitute $\mathcal{C}_j$ are often static, it is possible to precompute $\mathcal{R}_B(\mathcal{G}_j; \mathcal{C}_j)$ offline. On the other hand, for any $t \leq t_b$ we need to recompute the analysis with the added collision constraint that ensures a safe interaction with higher priority vehicles.

By doing these steps, we reduce the total amount of reachability analysis as performed online and avoid the conflicting objectives problem. That is, we precompute $\mathcal{R}_B(\mathcal{G}_j; \mathcal{C}_j)$ offline, fetch it from a lookup table at runtime and then update with (3.3) only for the relevant time $t \leq t_b$. The purpose of the update is to remove all trajectories that would enter $\mathcal{D}_j$ and collide with higher priority vehicles. As such, we introduce the "Offline Pass", *Pass 1*, that precomputates $\Phi_j^1 = \mathcal{R}_B(\mathcal{G}_j; \mathcal{C}_j)$ and the subsequent online passes start with *Pass 2* which produces $\Phi_j^2 \subseteq \Phi_j^1$ by updating the solution using

$$\mathcal{R}(\mathcal{G}_j \cup \{(z, t_b) \mid z \in \Omega_{\mathcal{R}(\mathcal{G}_j, \mathcal{C}_j)}(t_b)\};\ \Phi_j^1 \cap \mathcal{D}_j^C). \tag{3.4}$$

The passes are illustrated in Fig. 3.2a and 3.2b, respectively.

3.3.2 *Pass 3* and *Pass 4*

As mentioned in the previous section, although the steps so far produce time-state sets that ensure safe behavior, (3.1) also allows for behavior that negatively impacts throughput. This is a drawback of sequentially planning in general; we are not considerate of lower priority vehicles and the system as a whole. We will now introduce two important additions that refine the admissible trajectories of higher priority vehicles, *Pass 3* and *Pass 4*.

Given entry location $\mathbb{G}_j^{\leftarrow}$ and time window $T_j^{\leftarrow}$ for vehicle j, we want to find all trajectories that start from the entry time-state set $\mathcal{G}_j^{\leftarrow} = T_j^{\leftarrow} \times \mathbb{G}_j^{\leftarrow}$ and still satisfy the constraints imposed by (3.2). The latter is true as long as the vehicle stays within Φ_j^2. To achieve the former, we need to evaluate trajectories using forward reachability analysis. Specifically, *Pass 3* computes

$$\Phi_j^3 = \mathcal{R}_F(\mathcal{G}_j^{\leftarrow}; \Phi_j^2).$$

The resulting time-state set, illustrated in Fig. 3.2c, starts at the entry and

grows into Φ_j^2. This highlights a common aspect of AIMs. Even though it is the vehicle that suggests entry conditions, there is a requirement that $\mathcal{G}_j^{\leftarrow} \subseteq \Phi_j^2$. If this is not the case, there exists no safe entry that both vehicle j and the intelligent intersection can agree on. This could, for example, trigger a renegotiation depending on the control architectures of the intelligent intersection.

After *Pass 3*, vehicle j must enter the region at $\mathcal{G}_j^{\leftarrow}$ and exit at location $\mathbb{G}_j^{\rightarrow}$. However, Φ_j^3 still allows the vehicle to exit at a number of different times. The last step in pruning the trajectories is done by selecting a time window $T_j^{\rightarrow}$, and consequently $\mathcal{G}_j^{\rightarrow} = T_j^{\rightarrow} \times \mathbb{G}_j^{\rightarrow}$, for exiting the region. However, deciding $T_j^{\rightarrow}$ can be done in different ways, either manually or automatically. In this work, we suggest finding the earliest possible time window of length Δ. Using the time-state sets, this is easily done by finding the earliest time we can safely reach the exit:

$$\mathcal{E} = \Phi_j^3 \cap (T_{\Phi_j^2} \times \mathbb{G}_j^{\rightarrow}),$$
$$t = \inf\{\tau \in T_{\mathcal{E}} \mid [\tau, \tau + \Delta] \subseteq T_{\mathcal{E}}\}.$$

From this, we get the exit time window $T_j^{\rightarrow} = [t, t + \Delta]$.

With $\mathcal{G}_j^{\rightarrow} = T_j^{\rightarrow} \times \mathbb{G}_j^{\rightarrow}$ determined, we reach the final step of the refinement. Simply, in *Pass 4* we once again perform another backward reachability analysis:

$$\Phi_j^4 = \mathcal{R}_B(\mathcal{G}_j^{\rightarrow}, \Phi_j^3).$$

As with the entry, it is required that $\mathcal{G}_j^{\rightarrow} \subseteq \Phi_j^3$, meaning that exit conditions are satisfiable considering previous passes. In comparison with Φ_j^2, which subsumes most of the intersection, the resulting time-state set Φ_j^4 is now a narrow corridor as shown in Fig. 3.2d. Notably, a large Δ means a large exit time window, which in turn leads to a larger Φ_j^4. Yet, if Δ is too small, there might not be any feasible trajectory that satisfy all conditions. Therefore, Δ directly affects the efficiency of the intersection, and constitutes a part of the trade-off with safety.

3.3.3 Driving Limits Service

The result of *Pass 4* is, for vehicle j, a corridor that is safe to traverse. With this in mind, we suggest the design of a service that issues driving limits such that the vehicle is ensured to stay inside its safe corridor.

For a set of states, our service computes least-restrictive control sets $A_j(z_j, t)$ that guarantee satisfaction of (3.1). To compute the driving limits,

we follow the steps described in [24, Section V.B]. Let $V_{\Phi_j^4}(z_j, t)$ be the underlying value function in the HJ reachability analysis representing the safe corridor. Using $V_{\Phi_j^4}$, we compute $A_j(z_j, t)$ as the half-space $a + b^\top u_j \leq 0$ with

$$\begin{aligned} a &= V_{\Phi_j^4}(z_j, t) + D_t V_{\Phi_j^4}(z_j, t)\delta t \\ &\quad + D_z V_{\Phi_j^4}(z_j, t)^\top f_j(z_j, u_j)\delta t \\ b^\top &= D_z V_{\Phi_j^4}(z_j, t)^\top g_j(z_j)\delta t. \end{aligned}$$

Taken over time, this is a set of admissible accelerations such as depicted in Fig. 3.1. For further details, we refer to [24]. In the next chapter, we show that this method practically feasible with current compute capabilities.

Chapter 4

Numerical Results

To demonstrate this work, we present numerical results that highlight the benefits of our method. First, we present a 3-way intersection that highlights the safety verification. Then, we show that the illustrations in Fig. 3.2 reflect the computations of *Pass 2* and *Pass 4*. Next, we use the computed time-state sets to calculate least-restrictive control sets that runs the driving limits service. Finally, we end this section with examples of three-vehicle scenarios and report their computational time. The code for all results is publicly available on GitHub*.

4.1 3-way Intersection Scenario

The 3-way intersection scenario is presented in simulation, shown in Fig. 4.1, and includes three vehicles that cross the intersection in a way that risk collision if there is no coordination between them. Each vehicle is modelled by (2.1) with working space and control constraints: $\mathbb{S}_i = \{z_i \in \mathbb{R}^5 \mid -1.2 \leq x_i \leq 1.2, -1.2 \leq y_i \leq 1.2, -\pi \leq \theta_i \leq \pi, -\pi/5 \leq \delta_i \leq \pi/5, 0 \leq v_i \leq 1\}$, $\mathbb{U}_i = \{u_i \in \mathbb{R}^2 \mid -\pi \leq s_i \leq \pi, -0.5 \leq a_i \leq 0.5\}$. The roads enforce constraints on the vehicles' heading, except in the middle of the intersection. Furthermore, a lower-bound speed limit of 0.4 m/s is set to prevent the vehicles from stopping and blocking the way for following vehicles. Finally, for each road we define entry and exit targets. For example, vehicle 2 will enter at $\mathbb{G}_2^{\leftarrow} = \{z_2 \in \mathbb{S}_2 \mid 0 \leq x_2 \leq 0.5, -1.2 \leq y_2 - 0.7\}$. The reachability analysis is performed using the Python package `hj_reachability`† which solves

*https://github.com/kaarmu/safe_intersections

†https://github.com/StanfordASL/hj_reachability

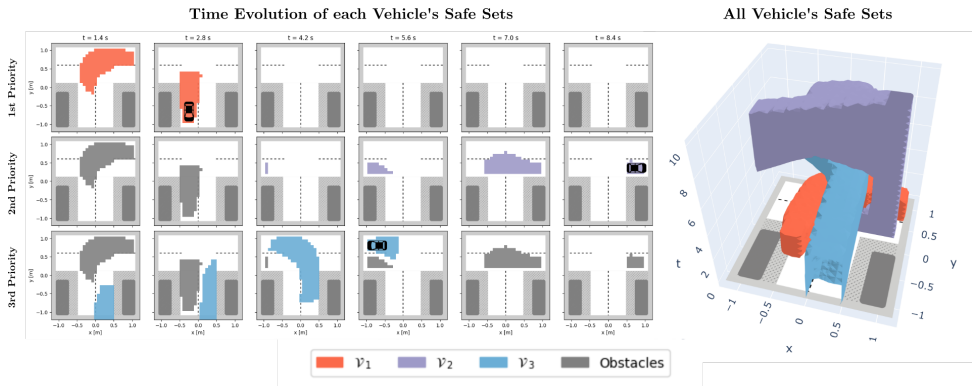
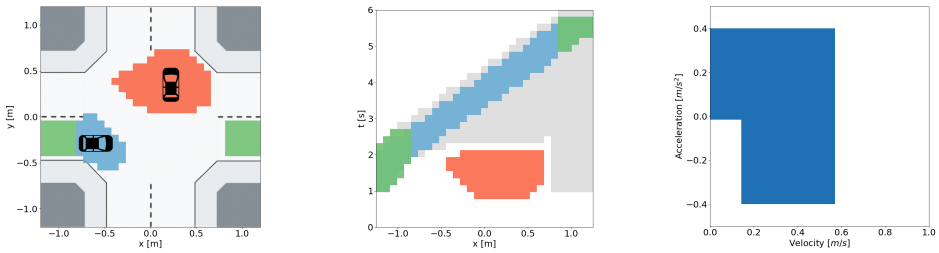


Figure 4.1: Shown are the safe sets for all vehicles over time (left part) and the sliced time snapshots of the safe sets for each vehicle, vehicle 1 (red), vehicle 2 (blue) and vehicle 3 (purple), with priority in the given order (right part). Since the safe sets are computed starting from the goal state, we mark the goal state of each vehicle with an icon. The top row of the right part shows the analysis done for vehicle 1 w.r.t its individual objective φ_1. Similarly, the second and third rows show the analyses done for vehicle 2 and vehicle 3, respectively, where the gray regions are the reachable sets of higher priority vehicles seen as obstacles.

the HJI VI on a $31 \times 31 \times 31 \times 7 \times 11$ grid of the discretized single-vehicle state space $\mathbb{S}_i$.

In Figure 4.1, we show the time evolution of the computed safe sets of each vehicle. The analysis' time horizon starts at $t = 0$, which is the current time, and ends at $t = 10$. The top row shows the evolution of the highest priority vehicle. The plotted safe set corresponds to all of the locations the highest priority vehicle can be in to satisfy the intersection specification. Then, after the safe set of the highest priority vehicle is computed, the safe set is passed to the computation of the next vehicle to be used as a danger set (grey set). We repeat this for each lower priority vehicle. In other words, the red and purple sets are the danger sets ($\mathcal{D}_{3,1}$ and $\mathcal{D}_{3,2}$) for the 2nd and 3rd priority vehicles, respectively. Interestingly, we see the reachable set Θ_3 as computed in the online pass for vehicle 3 in blue. Here, we clearly see how the reachability analysis ensures that vehicle 3 avoid the higher priority vehicles. The online pass is computed over the time frame $T_{\mathcal{D}_3} = [0, 6]$ during which it removes states that would otherwise lead to a collision with either vehicle 1 or vehicle 2.



(a) Seen from above, we see the safe corridors of both vehicles at a time instance. Additionally, entry and exit locations for the second vehicle are shown in green.

(b) Slicing the intersection at $y = -0.3$, we view the progression of the sets in x through time. We compare the result of *Pass 4* (blue) with that of *Pass 2* (gray).

(c) A set of admissible accelerations computed by the driving limits service for the second vehicle at $x_2 = -0.75$, $y_2 = -0.25$ and $\theta_2 = 0$.

Figure 4.2: Two vehicles passing through the 4-way intersection perpendicular to each other. The first vehicle (red) going in the vertical direction, and the second vehicle (blue) going in the horizontal direction. Both vehicles stay within their indicated reserved time-state sets. In more detail, we show important aspects of the second vehicle's safety analyses.

4.2 Perpendicular Two-Vehicle Scenario

In this scenario, shown in Fig. 4.2a, two vehicles are at a 4-way intersection. We are interested in following the second vehicle (blue) going straight through the intersection, left to right. It is preceded by another vehicle (red), that enters the area 0.9 s earlier, also going straight, but from bottom to top. With this scenario, we wish to show results that reinforce our conceptual understanding from Fig. 3.1 and Fig. 3.2.

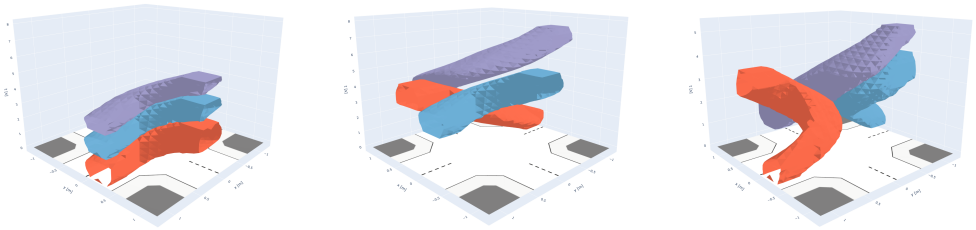
4.2.1 Reservation of Time-State Corridors

Using entry and exit locations defined for left, right, bottom, and top, we perform the online safety verification with *Pass 2*, *Pass 3* and *Pass 4*. In Fig. 4.2b, we show a slice of the intersection at y = -0.3 m to compare with the illustrations in Fig. 3.2. Specifically, the red region is a slice of the reserved time-state set for the higher-priority vehicle, the gray region shows the second vehicle's time-state set from the safety analysis in *Pass 2*, the green regions show the entry and exit conditions, and the blue region is *Pass 4*'s final time-state set that is then reserved for the second vehicle.

4.2.2 Output of Driving Limits Service

In this work, we have suggested that vehicles at the intersection subscribe to a driving limits service. For a given set of states, the service returns saturation points for the vehicle's control inputs that ensure it stays within its reserved time-state set. For example, at $x = -0.50, y = -0.25$ in Fig. 4.2a, the second vehicle (blue) must adhere to control constraints given by Fig. 4.2c. That is, when $0.2 \leq v < 0.6$, the vehicle can safely utilize the full acceleration space. However, if standing still, the vehicle is not allowed to decelerate. Naturally, we can also see that there is no allowed control for $v \geq 0.6$ since this is above the region's speed limit and thus already outside the safe time-state set.

4.3 Multiple Three-Vehicle Scenarios



(a) Three consecutive right turns.

(b) Straight, right, and a left turn.

(c) Two U-turns and a left turn.

Figure 4.3: Shown are scenarios that illustrate the flexibility of our approach. By providing only $\mathcal{G}_j^{\leftarrow}$ and $\mathbb{G}_j^{\rightarrow}$, it is possible to compute these safe corridors.

Lastly, we show three vehicles approaching the 4-way intersection in three different scenarios. For each scenario, the vehicles take different routes starting at slightly different times. Before reaching the intersection, each vehicle $j \in \{1, 2, 3\}$ requests a safe corridor from $\mathcal{G}_j^{\leftarrow}$ to $\mathbb{G}_j^{\rightarrow}$, corresponding to their selected route, with $\Delta = 2$ s. In Fig. 4.3 we show the safe corridors in dimensions x, y and t, using colors corresponding to the priority of the vehicles. In this case, priority is given according to FCFS principles, with red, blue, and purple corridors being in highest-to-lowest order.

The first scenario seen in Fig. 4.3a places all vehicles on the same route, taking a right-turn at the intersection. As vehicle 1 (red) is reserving a corridor, vehicle 2 (blue) also requests to pass through the region. The intelligent intersection reserves a second corridor tightly after the first one, yet never so they overlap. Finally, vehicle 3 (purple) makes the same request. Due to

		Pass 2 [s]	Pass 3 [s]	Pass 4 [s]	Total [s]
Scenario 1	Vehicle 1	0	1.39	1.14	**2.57**
	Vehicle 2	0.86	0.17	0.89	**1.98**
	Vehicle 3	0	0.18	0.93	**1.16**
Scenario 2	Vehicle 1	0	1.36	1.23	**2.63**
	Vehicle 2	0.88	0.18	0.90	**2.02**
	Vehicle 3	0.90	0.18	0.95	**2.10**
Scenario 3	Vehicle 1	0	1.33	1.11	**2.48**
	Vehicle 2	0.84	0.17	0.12	**1.20**
	Vehicle 3	0.12	0.16	0.95	**1.29**

Table 4.1: Computational time for online safety verification. Computed with AMD Ryzen Threadripper 2970X and NVIDIA GeForce RTX 2080 Ti.

being placed shortly next to each other, the two latter corridors are noticeably smaller. The second scenario seen in Fig. 4.3b presents a more dynamic situation. The same vehicles are now passing through the intersection using different routes. We also see that, if determined safe, vehicles can be scheduled at the same time. In this case, vehicle 2 (blue) and vehicle 3 (purple) drive in opposite directions, making it possible to safely pass through simultaneously. Finally, the third scenario seen in Fig. 4.3c shows that our method is flexible enough to accommodate non-conventional routes. At the same time, vehicles 1 (blue) and 2 (red) request to do mirrored U-turns. Shortly after, vehicle 3 (purple) wishes to drive left in the intersection, following the first vehicle.

These examples demonstrate that our method is highly flexible and can be used for various scenarios. For each scenario and vehicle, we show the computational time of the online verification in Table 4.1. The scenarios were simulated on a system equipped with an AMD Ryzen Threadripper 2970X processor and an NVIDIA GeForce RTX 2080 Ti graphics card. Moreover, to measure the performance of the driving limits service, we compute least-restrictive control sets for 100 randomly selected time-states for each scenario and vehicle. From this, the average computational time of a driving limits request was below 2 ms in all cases. These results indicate that, with modern software tools and hardware, our method is computationally feasible considering the real-time requirements of intelligent intersections.

Chapter 5

Conclusion

5.1 Discussion

In this paper, we develop a method to improve AIM safety by computing safe time-state corridors using reachability analysis. By addressing the problem of permissible planning, we can explicitly manage the trade-off between safety and efficiency. Additionally, we propose the driving limits service as a safety filter for AIM. Specifically, a vehicle can request a safe corridor before entering the intersection. Once inside, the vehicle can use the service to compute input saturation points, ensuring it remains within the safe corridor. To demonstrate our method, we show multiple scenarios of a simulated 4-way intersection. The results show promising performance, indicating feasibility for intelligent intersections and opportunities for integration with AIM.

Compared to other methods, our approach provides explicit guarantees of safety while maintaining flexibility in handling diverse vehicle behaviors and dynamic intersection scenarios. Many existing AIM systems rely on heuristic-based or centralized optimization methods to produce single trajectories. Following single trajectories, vehicles will be sensitive to deviations from the plan. In contrast, the use of temporal logic and Hamilton-Jacobi reachability allows our method to formalize and verify intersection specifications for all existing admissible trajectories.

However, there are two major issues left unresolved in our method. Firstly, how to inter-connect regions separately managed by different LTMS. Secondly, what are the rules around managing reservations. On the former, we suggested in Section 3.1 that our method provides the necessary interfaces for composing regions. The general idea is that the reservations should be made such that two corridors overlap in their $\mathcal{G}^{\leftarrow}$ and $\mathcal{G}^{\rightarrow}$ respectively. This

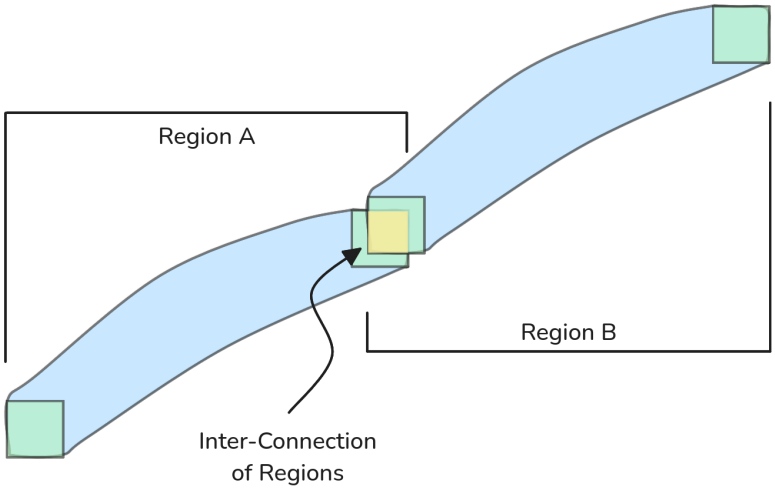


Figure 5.1: Illustration outlining inter-region coordination with our method, showing the overlap of safe time-state corridors during handover between two regions managed by different LTMS.

is illustrated by Fig. 5.1. During hand over, the vehicle must then comply with both regions. This is easily achieved by subscribing to respective driving limits service and intersecting the received least-restrictive control sets. The operation prunes away all controls, and consequently all trajectories, that do not satisfy both regions' safety specifications.

On to the latter issue, there are a number of relating practical problems. For example, when is the earliest and latest time a vehicle can call for a reservation with the LTMS? It is unfeasible that a corridor should be reserved for a vehicle that is very far away, perhaps hours in advanced. That causes the system to become very inflexible, requiring robust mechanisms for increased cancellations and re-reservations to handle vehicles deviating from their original plans. However, if they reserve too late, there may be little to no available time-state space to reserve a safe corridor. The LTMS could perhaps re-reserve other nearby vehicles, but this would likely violate their priority order which in turn invalidates assumptions in SPP, ultimately causing a full re-computation of all affected reservations.

Although these issues still need to be addressed, our method shows promising opportunities. It is very flexible and could be applied to other settings, such as highway merging, parking lots, and other transport sites that require a high level of interactions, e.g. goods depots, bus terminals and electric charging stations. With an LTMS at such sites we can create safe time-state corridors that could coordinate vehicles in these scenarios as well.

This highlights the potential of our method to enhance safety and efficiency across various transportation scenarios.

This technology motivates new actors to take an active part in traffic management, leading to a changing market and, possibly, the emergence of completely new industries. Though, it is not well understood how to scale this technology or for which transportation actor it makes practical sense to provide these services. Moreover, it is unclear which industry should develop the technology, protocols and collaboration with road users. Such concerns affect, for example, whether the LTMS should be hosted in the cloud or deployed at the edge. Cloud-based systems offer centralized computational resources and easier maintenance but may introduce higher latency, especially in areas with unreliable connectivity. On the other hand, edge-based deployments reduce latency and enhance responsiveness but require significant local computational infrastructure. Generally, there have not been enough tests and demonstrators that quantitatively evaluate advanced C-ITS deployed in different architectures.

There are also several theoretical questions that remain unresolved. At the considered transport sites (goods depots, bus terminals, electric charging stations, etc.), interactions are cooperative and possibly ephemeral, i.e., short-lived and non-recurring interactions between vehicles and infrastructure. What control techniques are suitable for such interactions and how does independent control systems, which only temporarily interact, affect each other? Current theories of distributed optimization, transient stability, and robust control provide partial solutions but lack the necessary integration to address the unique requirements of advanced C-ITS. Additionally, questions about data trust, secure communication, and real-time adaptability to unforeseen conditions remain critical areas for further research.

5.2 Future Work

In future work we wish to further improve performance using system decomposition such as [29]. In addition to techniques like [29], there are still many optimizations that could be developed for HJ reachability analysis, for example, using sparse representation of the level sets.

We also plan to implement an intelligent intersection for small-scale connected and automated vehicles. Through this, we aim to evaluate the performance of our framework with real hardware and communication similar to [31]. Such an implementation would allow us to demonstrate its integration with other C-ITS technologies such as set-based pedestrian detection [32] or

movement prediction of occluded road users [33].

Finally, an important future work will be to study how to compose safety analyses for inter-regional coordination. This includes investigating the interaction between local and regional control layers. It is important to ensure that traffic coordination is sound even as vehicles transition across regions. In the long term, such work could possibly develop the theoretical frameworks to bridge higher-level control layers, such as traffic flow analysis, to local traffic planning and direct vehicle control with formal guarantees.

References

[1] P. Arthurs, L. Gillam, P. Krause, N. Wang, K. Halder, and A. Mouzakitis, "A Taxonomy and Survey of Edge Cloud Computing for Intelligent Transportation Systems and Connected Vehicles," en, *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 6206–6221, Jul. 2022, ISSN: 1524-9050, 1558-0016. DOI: 10.1109/TITS.2021.3084396. [Online]. Available: https://ieeexplore.ieee.org/document/9447825/ (visited on 09/20/2023).

[2] I. Soto, M. Calderon, O. Amador, and M. Urueña, "A survey on road safety and traffic efficiency vehicular applications based on C-V2X technologies," en, *Vehicular Communications*, vol. 33, p. 100 428, Jan. 2022, ISSN: 22142096. DOI: 10.1016/j.vehcom.2021.100428. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2214209621000978 (visited on 04/17/2024).

[3] T. Gong, L. Zhu, F. R. Yu, and T. Tang, "Edge Intelligence in Intelligent Transportation Systems: A Survey," en, *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 9, pp. 8919–8944, Sep. 2023, ISSN: 1524-9050, 1558-0016. DOI: 10.1109/TITS.2023.3275741. [Online]. Available: https://ieeexplore.ieee.org/document/10133894/ (visited on 04/14/2024).

[4] L. Chen and C. Englund, "Cooperative Intersection Management: A Survey," en, *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 2, pp. 570–586, Feb. 2016, ISSN: 1524-9050, 1558-0016. DOI: 10.1109/TITS.2015.2471812. [Online]. Available: http://ieeexplore.ieee.org/document/7244203/ (visited on 04/21/2024).

[5] H. Ahn and D. Del Vecchio, "Semi-autonomous Intersection Collision Avoidance through Job-shop Scheduling," en, in *Proceedings of the 19th International Conference on Hybrid Systems: Computation and*

Control, Vienna Austria: ACM, Apr. 2016, pp. 185–194, ISBN: 978-1-4503-3955-1. DOI: 10.1145/2883817.2883830. [Online]. Available: https://dl.acm.org/doi/10.1145/2883817.2883830 (visited on 04/22/2024).

[6] F. Altche, X. Qian, and A. De La Fortelle, "Least restrictive and minimally deviating supervisor for Safe semi-autonomous driving at an intersection: An MIQP approach," en, in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, Rio de Janeiro, Brazil: IEEE, Nov. 2016, pp. 2520–2526, ISBN: 978-1-5090-1889-5. DOI: 10.1109/ITSC.2016.7795961. [Online]. Available: http://ieeexplore.ieee.org/document/7795961/ (visited on 04/22/2024).

[7] K. Dresner and P. Stone, "A Multiagent Approach to Autonomous Intersection Management," en, *Journal of Artificial Intelligence Research*, vol. 31, pp. 591–656, Mar. 2008, ISSN: 1076-9757. DOI: 10.1613/jair.2502. [Online]. Available: https://jair.org/index.php/jair/article/view/10542 (visited on 04/17/2024).

[8] S. Chamideh, W. Tärneberg, and M. Kihl, "A Safe and Robust Autonomous Intersection Management System Using a Hierarchical Control Strategy and V2I Communication," en, *IEEE Systems Journal*, vol. 17, no. 1, pp. 50–61, Mar. 2023, ISSN: 1932-8184, 1937-9234, 2373-7816. DOI: 10.1109/JSYST.2022.3221620. [Online]. Available: https://ieeexplore.ieee.org/document/9962817/ (visited on 04/14/2024).

[9] E. Namazi, J. Li, and C. Lu, "Intelligent Intersection Management Systems Considering Autonomous Vehicles: A Systematic Literature Review," en, *IEEE Access*, vol. 7, pp. 91 946–91 965, 2019, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2019.2927412. [Online]. Available: https://ieeexplore.ieee.org/document/8756239/ (visited on 02/13/2024).

[10] A. Colombo and D. Del Vecchio, "Efficient algorithms for collision avoidance at intersections," en, in *Proceedings of the 15th ACM international conference on Hybrid Systems: Computation and Control*, Beijing China: ACM, Apr. 2012, pp. 145–154, ISBN: 978-1-4503-1220-2. DOI: 10.1145/2185632.2185656. [Online]. Available: https://dl.acm.org/doi/10.1145/2185632.2185656 (visited on 02/13/2024).

[11] M. R. Hafner, D. Cunningham, L. Caminiti, and D. Del Vecchio, "Cooperative Collision Avoidance at Intersections: Algorithms and Experiments," en, *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1162–1175, Sep. 2013, ISSN: 1524-9050, 1558-0016. DOI: 10.1109/TITS.2013.2252901. [Online]. Available: https://ieeexplore.ieee.org/document/6495719 (visited on 04/22/2024).

[12] M. Saraoglu, J. Pintscher, and K. Janschek, "Designing a Safe Intersection Management Algorithm using Formal Methods," en, *IFAC-PapersOnLine*, vol. 55, no. 14, pp. 22–27, 2022, ISSN: 24058963. DOI: 10.1016/j.ifacol.2022.07.577. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2405896322009880 (visited on 02/13/2024).

[13] E. Irani Liu and M. Althoff, "Specification-Compliant Driving Corridors for Motion Planning of Automated Vehicles," en, *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 9, pp. 4180–4197, Sep. 2023, ISSN: 2379-8904, 2379-8858. DOI: 10.1109/TIV.2023.3289580. [Online]. Available: https://ieeexplore.ieee.org/document/10163843/ (visited on 04/24/2024).

[14] X. Chen and J. Martensson, "Heterogeneous Traffic Intersection Coordination Based on Distributed Model Predictive Control with Invariant Safety Guarantee," en, in *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, Macau, China: IEEE, Oct. 2022, pp. 3617–3624, ISBN: 978-1-66546-880-0. DOI: 10.1109/ITSC55140.2022.9922184. [Online]. Available: https://ieeexplore.ieee.org/document/9922184/ (visited on 02/13/2024).

[15] H. Kowshik, D. Caveney, and P. R. Kumar, "Provable Systemwide Safety in Intelligent Intersections," en, *IEEE Transactions on Vehicular Technology*, vol. 60, no. 3, pp. 804–818, Mar. 2011, ISSN: 0018-9545, 1939-9359. DOI: 10.1109/TVT.2011.2107584. [Online]. Available: http://ieeexplore.ieee.org/document/5699411/ (visited on 02/13/2024).

[16] J. Thunberg, G. Sidorenko, K. Sjoberg, and A. Vinel, "Efficiently Bounding the Probabilities of Vehicle Collision at Intelligent Intersections," en, *IEEE Open Journal of Intelligent Transportation Systems*, vol. 2, pp. 47–59, 2021, ISSN: 2687-7813. DOI: 10.1109/OJITS.20

21.3058449. [Online]. Available: https://ieeexplore.ieee.org/document/9352029/ (visited on 02/13/2024).

[17] C. Baier and J.-P. Katoen, *Principles of model checking*, en. Cambridge, Mass: The MIT Press, 2008, OCLC: ocn171152628, ISBN: 978-0-262-02649-9.

[18] S. Maierhofer, P. Moosbrugger, and M. Althoff, "Formalization of Intersection Traffic Rules in Temporal Logic," en, in *2022 IEEE Intelligent Vehicles Symposium (IV)*, Aachen, Germany: IEEE, Jun. 2022, pp. 1135–1144, ISBN: 978-1-66548-821-1. DOI: 10.1109/IV51971.2022.9827153. [Online]. Available: https://ieeexplore.ieee.org/document/9827153/ (visited on 08/12/2024).

[19] J. Haydon, M. Bondu, C. Eberhart, J. Dubut, and I. Hasuo, "Formal Verification of Intersection Safety for Automated Driving," en, in *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, Bilbao, Spain: IEEE, Sep. 2023, pp. 107–114, ISBN: 9798350399462. DOI: 10.1109/ITSC57777.2023.10421868. [Online]. Available: https://ieeexplore.ieee.org/document/10421868/ (visited on 04/22/2024).

[20] S. Shalev-Shwartz, S. Shammah, and A. Shashua, "On a Formal Model of Safe and Scalable Self-driving Cars," Aug. 2017, arXiv: 1708.06374. [Online]. Available: http://arxiv.org/abs/1708.06374.

[21] Y. Gao, A. Abate, F. J. Jiang, M. Giacobbe, L. Xie, and K. H. Johansson, "Temporal Logic Trees for Model Checking and Control Synthesis of Uncertain Discrete-Time Systems," *IEEE Transactions on Automatic Control*, vol. 67, no. 10, pp. 5071–5086, Oct. 2022, arXiv: 2007.02271 Publisher: Institute of Electrical and Electronics Engineers Inc., ISSN: 15582523. DOI: 10.1109/TAC.2021.3118335.

[22] M. Chen, S. Bansal, K. Tanabe, and C. J. Tomlin, *Provably Safe and Robust Drone Routing via Sequential Path Planning: A Case Study in San Francisco and the Bay Area*, en, arXiv:1705.04585 [cs], May 2017. [Online]. Available: http://arxiv.org/abs/1705.04585 (visited on 07/06/2023).

[23] S. Bansal, M. Chen, K. Tanabe, and C. J. Tomlin, "Provably Safe and Scalable Multivehicle Trajectory Planning," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 6, pp. 2473–2489, Nov. 2021, Conference Name: IEEE Transactions on Control Systems Technology, ISSN: 1558-0865. DOI: 10.1109/TCST.2020.3042815.

[24] F. J. Jiang, K. M. Arfvidsson, C. He, M. Chen, and K. H. Johansson, *Guaranteed Completion of Complex Tasks via Temporal Logic Trees and Hamilton-Jacobi Reachability*, en, arXiv:2404.08334 [cs, eess], Apr. 2024. [Online]. Available: http://arxiv.org/abs/2404.08334 (visited on 04/22/2024).

[25] T. Wongpiromsarn, *Formal Methods for Design and Verification of Embedded Control Systems: Application to an Autonomous Vehicle*. 2010, ISBN: 978-1-267-41233-1.

[26] J. Tumova, S. Karaman, C. Belta, and D. Rus, "Least-Violating Planning in Road Networks from Temporal Logic Specifications," en, in *2016 ACM/IEEE 7th International Conference on Cyber-Physical Systems (ICCPS)*, Vienna, Austria: IEEE, Apr. 2016, pp. 1–9, ISBN: 978-1-5090-1772-0. DOI: 10.1109/ICCPS.2016.7479106. [Online]. Available: http://ieeexplore.ieee.org/document/7479106/ (visited on 08/12/2024).

[27] P. Yu, Y. Gao, F. J. Jiang, K. H. Johansson, and D. V. Dimarogonas, *Online Control Synthesis for Uncertain Systems under Signal Temporal Logic Specifications*, en, arXiv:2103.09091 [cs, eess], Mar. 2023. [Online]. Available: http://arxiv.org/abs/2103.09091 (visited on 12/20/2023).

[28] M. Chen, S. L. Herbert, M. S. Vashishtha, S. Bansal, and C. J. Tomlin, "Decomposition of Reachable Sets and Tubes for a Class of Nonlinear Systems," *IEEE Transactions on Automatic Control*, vol. 63, no. 11, pp. 3675–3688, Nov. 2018, Conference Name: IEEE Transactions on Automatic Control, ISSN: 1558-2523. DOI: 10.1109/TAC.2018.2797194.

[29] C. He, Z. Gong, M. Chen, and S. Herbert, "Efficient and Guaranteed Hamilton-Jacobi Reachability via Self-Contained Subsystem Decomposition and Admissible Control Sets," en, *IEEE Control Systems Letters*, pp. 1–1, 2023, ISSN: 2475-1456. DOI: 10.1109/LCSYS.2023.3344659. [Online]. Available: https://ieeexplore.ieee.org/document/10365682/ (visited on 12/28/2023).

[30] M. Chen and C. J. Tomlin, "Hamilton–Jacobi Reachability: Some Recent Theoretical Advances and Applications in Unmanned Airspace Management," en, *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, pp. 333–358, May 2018, ISSN: 2573-5144, 2573-5144. DOI: 10.1146/annurev-control-060117

-104941. [Online]. Available: https://www.annualreviews.org/doi/10.1146/annurev-control-060117-104941 (visited on 07/03/2023).

[31] K. M. Arfvidsson *et al.*, "Small-Scale Testbed for Evaluating C-V2X Applications on 5G Cellular Networks," en, in *2024 IEEE Intelligent Vehicles Symposium (IV)*, Jeju Island, Korea, Republic of: IEEE, Jun. 2024, pp. 149–155, ISBN: 9798350348811. DOI: 10.1109/IV55156.2024.10588559. [Online]. Available: https://ieeexplore.ieee.org/document/10588559/ (visited on 08/15/2024).

[32] K. Fragkedaki, F. J. Jiang, K. H. Johansson, and J. Mårtensson, *Pedestrian Motion Prediction Using Transformer-based Behavior Clustering and Data-Driven Reachability Analysis*, en, arXiv:2408.15250 [cs, eess], Aug. 2024. [Online]. Available: http://arxiv.org/abs/2408.15250 (visited on 09/12/2024).

[33] J. M. G. Sánchez, T. Nyberg, C. Pek, J. Tumova, and M. Törngren, "Foresee the unseen: Sequential reasoning about hidden obstacles for safe driving," in *2022 IEEE Intelligent Vehicles Symposium (IV)*, 2022, pp. 255–264. DOI: 10.1109/IV51971.2022.9827171.

TRITA-EECS-EX-2024:953