

## MIT Open Access Articles

*Multi-Vehicle Motion Planning for  
Social Optimal Mobility-on-Demand*

The MIT Faculty has made this article openly available. **Please share**  
how this access benefits you. Your story matters.

**Citation:** Karlsson, Jesper, et al. "Multi-Vehicle Motion Planning for Social Optimal Mobility-on-Demand." 2018 IEEE International Conference on Robotics and Automation (ICRA), 21-25 May 2018, Brisbane, Australia, IEEE, 2018, pp. 7298–305.

**As Published:** <http://dx.doi.org/10.1109/ICRA.2018.8462968>

**Publisher:** Institute of Electrical and Electronics Engineers (IEEE)

**Persistent URL:** <http://hdl.handle.net/1721.1/118967>

**Version:** Author's final manuscript: final author's manuscript post peer review, without publisher's formatting or copy editing

**Terms of use:** Creative Commons Attribution-Noncommercial-Share Alike



# Multi-vehicle motion planning for social optimal mobility-on-demand

Jesper Karlsson<sup>1</sup>, Cristian-Ioan Vasile<sup>2</sup>, Jana Tumova<sup>1</sup>, Sertac Karaman<sup>2</sup> and Daniela Rus<sup>2</sup>

**Abstract**—In this paper we consider a fleet of self-driving cars operating in a road network governed by rules of the road, such as the Vienna Convention on Road Traffic, providing rides to customers to serve their demands with desired deadlines. We focus on the associated motion planning problem that trades-off the demands’ delays and level of violation of the rules of the road to achieve social optimum among the vehicles. Due to operating in the same environment, the interaction between the cars must be taken into account, and can induce further delays. We propose an integrated route and motion planning approach that achieves scalability with respect to the number of cars by resolving potential collision situations locally within so-called *bubble spaces* enclosing the conflict. The algorithms leverage the road geometries, and perform joint planning only for *lead* vehicles in the conflict and use queue scheduling for the remaining cars. Furthermore, a framework for storing previously resolved conflict situations is proposed, which can be used for quick querying of joint motion plans. We show the mobility-on-demand setup and effectiveness of the proposed approach in simulated case studies involving up to 10 self-driving vehicles.

## I. INTRODUCTION

In this paper we consider the problem of deploying a fleet of self-driving vehicles providing rides to customers to meet their demands within certain deadlines, and at the same time obeying the rules of the road, such as speed limits, construction areas, and traffic lights. The two goals may not be always compatible, since the rules of the road may slow down the autonomous vehicles and preclude the satisfaction of their respective customer demands by their deadlines. Moreover, by operating in the same environment, the vehicles interact with each other which might induce further delays. Our goal is to compute motion plans for all vehicles that are guaranteed to meet the customers’ demands within their deadlines while obeying the rules of the road or, if this is not possible, with optimal social cost that encodes a fair distribution of individual delays in servicing the demands among the vehicles.

In our prior work, we explored similar motion planning problems for a single vehicle with conflicting rules of the road [1], [2], [3]. The notion of level of violation of rules was introduced in [1], [2] for the design of minimum-violation planners, where the vehicle was tasked to arrive at a goal location. Richer demands were considered in [4] for vehicle

routing over topological models of road networks with the goal of minimizing delays of demands that could not be serviced within their desired deadlines. An integrated route and motion planning algorithm was proposed in [3] for a single vehicle with limited sensing tasked with servicing rich demands and obeying the rules of the road in a minimum-violating way. The approach overcomes spatial and temporal scalability issues, allowing the methods to handle large road networks with time-varying estimated travel times available to the vehicle. In this work, we focus on fundamentally different issues associated with the consideration of multiple vehicles. Our scalable integrated route and motion planning approach resolves interactions between vehicles based on their assigned demands and the level the road rules violation to achieve a social optimum.

The vehicles operate in a road network which is captured as a hierarchical model describing the vehicles’ dynamics, the road segments the vehicles traverse, and a high-level finite abstraction of roads and intersections. For simplicity, the customer demands are given as goal locations that need to be reached by desired deadlines. While servicing the demands, the vehicles must avoid colliding with each other and satisfy the rules of the road which are encoded as syntactically co-safe Linear Temporal Logic (scLTL) formulae. The choice of scLTL is motivated by its resemblance to natural language, rigorousness, and expressiveness allowing to formalize a variety of reachability and sequencing tasks, such as “Pick me up at work, then go to the school to pick up the kids and then bring us home. Somewhere on our way, stop by at a shopping mall or a bakery.” The interaction between the vehicles determines a social cost over the entire fleet in terms of delays and level of violation. Based on our previous work, we formulate a motion planning problem for multi-vehicle systems with the objective of minimizing the social cost. We propose a receding horizon solution, that leverages our previous planning method [3] when no collision between vehicles are encountered. However, when possible collisions are detected, we resolve the situations locally within so-called *bubble spaces* enclosing the conflict, and performing joint planning to overcome them. We deal with scalability with respect to the number of cars using a two-step method that leverages the geometry of roads, where joint planning is performed only for the *lead* vehicles in the conflict, and using queue scheduling to resolve the conflict for the remaining cars. Further optimization is achieved by storing conflict scenarios and returning the previously computed motion plans. The contributions of the paper can be summarized as follows:

\*Toyota Research Institute (“TRI”) provided funds to assist the authors with their research but this article solely reflects the opinions and conclusions of its authors and not TRI or any other Toyota entity.

<sup>1</sup>Jesper Karlsson, and Jana Tumova are with KTH Royal Institute of Technology, Stockholm, Sweden {jeskarl, tumova}@kth.se

<sup>2</sup>Cristian-Ioan Vasile, Sertac Karaman, and Daniela Rus are with the Massachusetts Institute of Technology, Cambridge, MA, USA {cvasile, sertac}@mit.edu, rus@csail.mit.edu

- We formalize the motion planning problem in Pb. 1 for multiple self-driving vehicles that are assigned demands, must avoid collisions, and satisfy the rules of the road.
- We design a receding horizon solution that allows the self-driving vehicles to follow trajectories that provably minimizes the social cost given in terms of specification violation.
- The proposed solution deals with scalability with respect to the number of vehicles by resolving conflicts locally using joint planning for lead vehicles and queuing.
- We demonstrate the applicability of the proposed approach in simulation case studies.

Planning under infeasible temporal logic specifications has been addressed in [5], [6], [7], where metrics of level of satisfaction of formulae are defined and used to solve optimal planning problems with respect to the chosen metric. Mobility-on-demand for fleets of autonomous vehicles has been considered e.g., in [8], where a real-time rebalancing policy was developed to maximize the throughput of the system. Queue scheduling methods were used in [9] to compute motion plans for safe navigation of intersections without lights. A reactive sampling-based framework for robots with limited sensing was proposed in [10], [11], but that work does not consider minimum violation of conflicting temporal constraints. On the other hand, an RRT\*-based approach tailored for finding minimum-violation motion plans was proposed in [1], but it did not consider sensing, all information was available off-line. In previous work, we proposed an integrated route and motion planning framework [3] that minimized specification violation, but only for a single vehicle.

## II. PRELIMINARIES AND NOTATION

Let  $\mathbb{R}$  be the set of real and  $\mathbb{N}$  the set of natural numbers. We use  $\bar{\mathbb{R}} = \mathbb{R} \cup \{\pm\infty\}$ ,  $\bar{\mathbb{N}} = \mathbb{N} \cup \{\infty\}$ , and  $\mathbb{R}_{\geq a} = [a, \infty)$ . Given a set  $S$ , we denote by  $2^S$ , and  $|S|$  the set of all subsets of  $S$ , and the cardinality of  $S$ , respectively. A sequence of elements from  $S$  is called a *word*, and we use  $w^j$  to denote the *suffix*  $s_j s_{j+1} s_{j+2} \dots$  starting at the  $j$ -th position of a finite or infinite word  $w = s_1 s_2 s_3 \dots$ .

Let  $X \subset \mathbb{R}^m$ ,  $U \subset \mathbb{R}^n$  be compact sets. Consider a dynamical system given by  $\dot{x}(t) = f(x(t), u(t))$ ,  $x(0) = x_0$ , where  $x_0$  is the initial state, and a labeling function  $\mathcal{L} : X \rightarrow 2^\Pi$  that maps each state  $x$  to a subset of atomic propositions  $\mathcal{L}(x) \subseteq \Pi$  that this state satisfies. A state trajectory  $x : \mathbb{R}_{\geq 0} \rightarrow X$  of the system is associated with the *duration output word*  $\mathbf{o} = (\sigma_1, d_1)(\sigma_2, d_2) \dots$  defined such that  $\mathcal{L}_i(x([t_k, t_{k+1}))) = \sigma_k$  and  $\sigma_k \neq \sigma_{k+1}$  for all  $k \geq 1$ , where  $t_{k+1} = t_k + d_k$  and  $t_1 = 0$ . We denote by  $\sigma = \sigma_1 \sigma_2 \dots$  the output word produced by  $x$ .

A syntactically co-safe Linear Temporal Logic (scLTL) formula over alphabet  $\Sigma$  is defined as

$$\varphi ::= \pi \mid \neg\pi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid X\varphi \mid F\varphi \mid \varphi U \varphi,$$

where  $\pi \in \Pi$ ,  $\neg$  (negation),  $\wedge$  (conjunction), and  $\vee$  (disjunction) are Boolean operators, and  $U$  (until),  $X$  (next), and  $F$  (eventually) are temporal operators [12].

An scLTL formula is interpreted over infinite words over  $2^\Sigma$ , such as the output words produced by a state trajectories of a dynamical system. The *satisfaction* of an scLTL formula  $\varphi$  by a word  $w = w_1 w_2 w_3 \dots$  over  $2^\Sigma$  is defined through the satisfaction relation  $\models$  as follows:

$$\begin{aligned} w \models \pi &\iff \pi \in w_1 w \models \neg\pi \iff \pi \notin w_1, \\ w \models \varphi \vee \psi &\iff w \models \varphi \vee w \models \psi, \\ w \models \varphi \wedge \psi &\iff w \models \varphi \wedge w \models \psi, \\ w \models X\varphi &\iff w^2 \models \varphi, \\ w \models F\varphi &\iff \exists i \geq 1. w^i \models \varphi, \\ w \models \varphi U \psi &\iff \exists i \geq 1. w^i \models \psi \wedge \forall 1 \leq j < i. w^j \models \varphi \end{aligned}$$

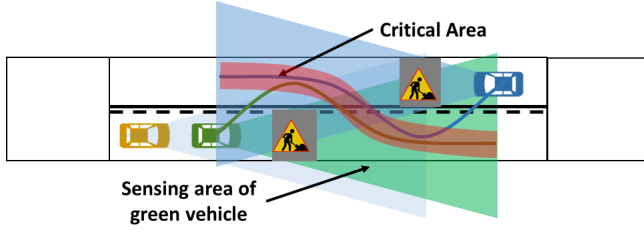
Although scLTL formulas are generally defined over infinite words, their satisfaction is decided in finite time [12], which enables to express them as finite-state automata.

## III. PROBLEM FORMULATION

We consider a fleet of controlled (autonomous) vehicles  $\mathcal{V}$  operating in a road network. We assume that each vehicle  $\nu_i \in \mathcal{V}$  has been assigned a task to travel from one location to another, and a route that executes this demand optimally, i.e. in the shortest possible time. When following their optimal routes, the vehicles should obey given road rules and avoid collisions. In this work, we focus on handling situations when this is not possible, i.e. when several autonomous vehicles prevent each other from following the route optimally. In other words, we restrict our attention on multi-vehicle motion planning problem with the aim to reach a compromise leading to a social optimum.

**Example 1** An illustration of a situation that we aim to address is given in Fig. 1. A part of the given optimal route for each of the three vehicles is to traverse the depicted road segment in minimum time. However, the time-optimal motion plans lead through construction zones and violate the road rules. The motion plans that minimize traversal time and satisfy the road rules are depicted in Fig. 1 for the green and blue vehicles. These motion plans are computed assuming no cooperation, and if each vehicle follows it, a collision will be unavoidable. The three autonomous vehicles thus have to decide on which one of them traverses the narrow passage first, which second, and which third. We aim to maximize the overall performance of the fleet, hence the decisions will take into account the cost incurred by each vehicle in terms of road rules violation and delays in arriving to the goal location.

We model each vehicle as a dynamical system with limited sensing at the level of a road segment. Each road rule is captured as an scLTL formula associated with priority. We introduce a *social cost* based on accumulated delays w.r.t. targeted deadlines of all demands, and penalties for violating the road rules by the individual vehicles. Finally,



**Fig. 1:** A road segment with two construction areas and three autonomous vehicles traversing it. The sensing areas and *ego-trajectories* for the green and blue vehicles are shown.

we introduce our multi-vehicle motion planning problem optimizing the social cost. We call non-cooperative motion plans *ego-trajectories* and we focus on traffic situations where the vehicles can not safely implement their *ego-trajectories*.

#### A. Model and Specification

1) *Vehicle model:* A vehicle  $\nu_i$  is defined as a tuple  $\nu_i = (f_i, X, U, h, Sense_i)$ , where  $X \subset \mathbb{R}^m$ ,  $U \subset \mathbb{R}^n$ , and  $\mathfrak{R} \subset \mathbb{R}^2$  are the common state, control, and work spaces of the vehicles,  $Sense_i : \mathbb{R}^2 \rightarrow 2^{\mathbb{R}^2}$ . The vehicles' dynamics are given by

$$\dot{x}_i = f_i(x_i, u_i), \quad x_i(0) = x_{0,i} \quad (1)$$

$$y_i = h(x_i) \quad (2)$$

where  $x_{0,i}$  is the initial state at time  $t = 0$ ,  $f_i : X \times U \rightarrow X$  and  $h : X \rightarrow \mathbb{R}^2$  are the Lipschitz continuous dynamics and observation (location) functions of vehicle  $\nu_i$ , respectively. The workspace  $\mathfrak{R}$  here corresponds to a road network, which is a compact planar region corresponding to road lanes and intersections. The limited sensing area of  $\nu_i$  at location  $y_i$  is given by  $Sense_i(y_i) \subset \mathbb{R}^2$ . The state trajectory under a control policy  $u_i(\cdot)$  is said to be feasible if the location of the vehicle in the planar environment stays inside  $\mathfrak{R}$  for all times.

2) *Vehicle task:* Each controlled vehicle  $\nu_i$  is assigned a task to reach a goal region  $g_i \subseteq \mathfrak{R}$  that is associated with a deadline  $\Delta_i \in \mathbb{N}$  and a priority  $\pi_i \in \mathbb{N}$ . Assume that each controlled vehicle has an access to a service that provides information about the traffic conditions in the road network, e.g., INRIX, Google Traffic, Waze. This means that the vehicle is given a nominal path (route) in the road network and an estimate time to complete the task  $t_{E,i}$  from its current position. Let  $t_{P,i}$  denote the time passed since the task arrival.

$$D_i = \Delta_i - (t_{P,i} + t_{E,i})$$

is then the estimated delay of servicing the task.

3) *Road markings and rules:* We denote by  $\Pi$  the common set of signs and markings that annotate the road network. Each vehicle  $\nu_i$  in the road network has its own road labeling map  $\mathcal{L}_i(t)$  that assigns labels from  $\Pi$  to the vehicle's sensing area  $Sense_i(y_i(t))$ . It is used to enforce the rules of the road (e.g., speed limit, stop, and construction signs, lane and intersection delimiters). Note that the regions induced

**TABLE I:** Symbols table.

$\nu_i$	vehicle model
$X, U, \mathfrak{R}$	state, control, and work spaces
$f_i, h, Sense_i$	dynamics, observation (location) and sensing maps of vehicle $\nu_i$
$x_{0,i}$	initial state of vehicle $\nu_i$
$\Pi$	set of all signs and markings
$\mathcal{L}_i$	road labeling map of vehicle $\nu_i$
$\mathbf{o}_i, \sigma_i$	duration output word and output word of a trajectory $x_i$ induced by $\mathcal{L}_i$
$g_i$	demand, i.e. desired goal region of vehicle $\nu_i$
$\pi_i$	priority of demand $g_i$
$\Delta_i, D_i$	deadline and delay associated with optimal route of $\nu_i$
$\Theta_i$	road rules for vehicle $\nu_i$ enforced until reaching $g_i$
$\theta_j^a, \theta_j^g$	assume and guarantee parts of road rule $\theta_{i,j} \in \Theta_i$
$p_j$	priority associated with road rule $\theta_{i,j}$

by  $\mathcal{L}_i$  are interpreted as the regions of the road network that a sign or marking applies to. For example,  $\mathcal{L}_i$  designates as construction the entire closed-off region of the road segment, and not just the location of the start and end construction signs. With a slight abuse of notation, we use  $g_i \in \Pi$  to label the goal region  $g_i \subseteq \mathfrak{R}$ .

The rules of the road are build from the sets of Boolean assumption formulas  $\Theta^a$  and scLTL guarantee formulas  $\Theta^g$  over  $\Pi$ . For vehicle  $\nu_i$ , the set  $\Theta_i$  consists of road rules taking the reactive form

$$\theta_{i,j} = (\theta_j^a \stackrel{\text{Re}}{\Rightarrow} \theta_j^g) \cup g_i, \quad (3)$$

where  $\theta_j^a \in \Theta^a$ ,  $\theta_j^g \in \Theta^g$ ,  $\stackrel{\text{Re}}{\Rightarrow}$  is a reactive implication, and  $g_i$  indicates that  $\nu_i$  reached  $g_i$ . We say that a traffic rule  $\theta_{i,j}$  is active if  $\theta_j^a$  is satisfied. The rules in  $\Theta_i$  may become active at any time during the traversal of a road segment.

**Example 2** Recall Example 1. Examples of road rules for each vehicle are: 1) stay in the right lane; 2) if the road is under construction, then avoid construction areas; 3) if no overtaking is allowed, then do not cross center line into left lane; and 4) if speed is limited to 50mph, then drive at under 50mph. The road rules written as an scLTL formula given in Eq.(3), where

$j$	$\theta_j^a$	$\theta_j^g$
1	$\top$	RightLane
2	UnderConstruction	$\neg$ ConstructionArea
3	NoOvertake	$\neg$ LeftLane
4	50mphLimit	Under50mph

#### B. Solution cost

We focus on situations when the goal regions cannot be reached by their associated deadlines without violating the road rules and without cooperation and negotiation between the controlled vehicles. That is, we focus on motion planning problems with social optimum.

1) *Level of road rules violation:* Similarly as in [1], we define the level of violation with respect to the road rules. Each road rule  $\theta_{i,j} \in \Theta_i$  is associated with a priority  $p_j \in \mathbb{N}$ . Intuitively, the level of violation is measured as

the cumulative time spent satisfying the individual road rule assumptions, but not the guarantees, weighted by the corresponding priority.

Formally, given a duration output word  $\mathbf{o}_i = (\sigma_1, d_1)(\sigma_2, d_2) \dots$  corresponding to a trajectory  $x_i$  of vehicle  $\nu_i$ , the level of violation is

$$P_i(x_i) = \sum_{\theta_{i,j} \in \Theta_i} \left( p_j \cdot \sum_{k \in \{k | \sigma_k \models \theta_j^a \wedge \neg \theta_j^g\}} d_k \right).$$

2) *Collision avoidance*: The collision avoidance of the controlled vehicles needs to be strictly enforced throughout the entire mission. For simplicity, we say that a vehicle  $\nu_i$  is collision-free if  $\|y_i(t) - y_j(t)\| > \varepsilon$  for all  $\nu_j \in \mathcal{V} \setminus \{\nu_i\}$  and  $t \geq 0$ , where  $\|\cdot\|$  is the Euclidean norm and  $\varepsilon$  is the collision threshold. The collision threshold captures the geometric sizes of the vehicles.

3) *Social cost*: Finally, we define a cost function that reflects the inter-vehicle importance of servicing the tasks in time and satisfying the road rules

$$J(\mathcal{V}) = c(D_1, \dots, D_{|\mathcal{V}|}, \pi_1, \dots, \pi_{|\mathcal{V}|}, P_1(x_1), \dots, P_{|\mathcal{V}|}(x_{|\mathcal{V}|})).$$

**Example 3** Recall Examples 1 and 2. Suppose that the green (G), blue (B) and yellow (Y) vehicles all have a priori expected delay 0, equal priorities 1, and levels of violation 0, 0, and 1, respectively. Moreover, their traversal times corresponding through the critical corridor are 2, 2, 3, respectively. Let  $J_1(\mathcal{V}) = \sum_{i \in \mathcal{V}} (\pi_i D_i + \beta P_i(x_i))$  with  $\beta = 2$ , and  $J_2(\mathcal{V}) = \max_{i \in \mathcal{V}} \{\pi_i D_i\}$ , be the average and bottleneck social cost functions. Their values corresponding to the six decision choices based are given in the table below

order	$D_i$	$P_i(x_i)$	$J_1(\mathcal{V})$	$J_2(\mathcal{V})$
(G, Y, B)	0, 0, 3	1, 3, 1	13	3
(G, B, Y)	0, 4, 2	1, 3, 1	16	4
(Y, G, B)	3, 0, 5	1, 3, 1	18	5
(Y, B, G)	5, 0, 3	1, 3, 1	18	5
(B, G, Y)	2, 2, 0	1, 3, 1	14	2
(B, Y, G)	5, 2, 0	1, 3, 1	17	5

If the fleet minimizes  $J_1$ , then the green vehicle should go first, followed by the yellow and then blue vehicles. This ordering corresponds to the first line in the table above. On the other hand, if  $J_2$  is minimized, then the optimal traversal order is blue, green, and yellow vehicles, and corresponds to fifth line.

### C. Problem formulation

**Problem 1** Given a set of controlled vehicles,  $\mathcal{V}$  operating in the road network  $\mathcal{R}$ , goal regions  $g_i$ , deadlines  $\Delta_i$ , priorities  $\pi_i$ , a set of road rules  $\Theta_i$  with priorities  $p_j, \theta_{i,j} \in \Theta_i$ , and a level of violation function  $P_i(x_i)$  for each vehicle  $\nu_i \in \mathcal{V}$ , find a control policy  $u_i(t) : \mathbb{R}_{\geq 0} \rightarrow U$  such that the state trajectory  $x_i(t)$  is feasible for all  $\nu_i \in \mathcal{V}$ , and the social cost  $J(\mathcal{V})$  is minimized while maintaining collision avoidance.

**Remark 1** The autonomous vehicles must make local decisions to solve conflicting traffic situations such that the social

optimum is achieved. The solution cost associated with the fleet of vehicles involved in the conflict case depends on the traveling times and levels of violation induced by the decision options. Thus, although we do not change the routing decisions, changing the vehicles' motion plans to resolve the conflict means changing their remaining estimated times  $t_{E,i}$  to task completion.

## IV. PROBLEM SOLUTION

Let us first restrict our attention to a simplified version of Problem 1, where we only consider one vehicle  $\mathcal{V} = \{\nu\}$ . This means that there is no risk of collision, and our aim is to find a feasible ego-trajectory  $x(t)$  minimizing the cost function  $J(\mathcal{V})$ . We have addressed this problem earlier in [3] with a *minimum-violation scLTL RRT\** planner and we will utilize that solution in this work.

In the case of multiple vehicles, we could extend the aforementioned approach and look for a joint feasible trajectory in  $X^{|\mathcal{V}|}$ . This straightforward solution is, however, exponential with respect to the number of vehicles, and not tractable. In this paper, we propose an alternative, scalable solution to Problem 1 summarized in Alg. 1, where we construct a nominal feasible ego-trajectory  $\bar{x}_i(t)$  for each vehicle  $\nu_i \in \mathcal{V}$  while assuming that the vehicle is alone in the work space (lines 2-4), and iteratively execute the following steps. We build an undirected *communication graph* (line 5) that connects a vehicle  $\nu_i$  to all vehicles  $\nu_j \in \text{Sense}_i(y_i)$  in its limited sensing area. This graph guides the choice of a replanning strategy for each vehicle in  $\mathcal{V}$  and determines the order in which the vehicles are considered. In particular, there are three replanning strategies: If all vehicles that are part of a connected component in the *communication graph* have pairwise collision free nominal feasible state trajectories (line 7), then all these vehicles implement their nominal trajectories  $\bar{x}_i(t)$  (line 8). If two or more vehicles are at risk of collision, then the so-called *lead vehicles* are determined (line 11) that intuitively correspond to the closest vehicles to the potential collision. For the lead vehicles, we compute a so-called *bubble space* (line 12), i.e. a part of the work space, where replanning needs to take place. We compute a *joint plan* (line 17) for  $\nu_i$  and  $\nu_j$  in  $X^2$  leveraging the algorithm in [3] (see Sec. IV-B). This joint plan projects onto feasible state trajectories for  $\nu_i$  and  $\nu_j$ , ensures collision freedom, and at the same time minimizes  $J(\{\nu_i, \nu_j\})$ . We consider storing computed motion plans for later use (line 18) for instance in a cloud repository, by summarizing the traffic situation (see Sec. IV-B.2), and querying the repository (line 13) before attempting joint planning. For the remaining vehicles, we propose to adapt their trajectory to avoid collisions with the previously considered vehicles (line 23), where the order is by a queue scheduling method optimizing  $J(\cdot)$  (line 20). This step is called *enforced planning* (see Sec. IV-C). Finally, all vehicles execute one step of the motion plans (trajectories) for time  $\Delta t$ , and the process is repeated until all goal regions are reached.

For simplicity, we will assume throughout the paper that only two vehicles may be involved in traffic scenarios that

require coordination. However, the presented algorithms can handle cases that may involve multiple vehicles, e.g., at 3-way or 4-way intersections. It is important to note that even in this cases, joint planning is performed for a small number of vehicles that depends on the local road geometry (e.g., at most 4 vehicles for 4-way intersection) and not on the size of the entire fleet. For the remaining vehicles, *enforced planning* is employed.

In the rest of this section, we provide the details of the solution.

---

#### Algorithm 1: Planner

---

**Input:** Input to Problem 1, including  $\mathcal{V}$ ,  $\mathfrak{R}$ ,  $g_i$ , and  $\Theta_i$ , for all  $\nu_i \in \mathcal{V}$   
**Output:** A set of feasible collision-free state trajectories  
**Storage:** *Repository* of pairs of Bubble spaces and RRT\* trees

```

1: while  $x_i \notin g_i$  for some  $\nu_i \in \mathcal{V}$  do
2:   for all  $\nu_i \in \mathcal{V}$  do
3:      $\bar{x}_i(t) \leftarrow \text{MinViolationRRT}^*(\nu_i, \mathfrak{R}, g_i, \Theta_i)$  [3]
4:   end for
5:   Build the communication graph  $G(t)$ 
6:   for all connected components  $C$  of  $G(t)$  do
7:     if  $\|\bar{x}_i(t) - \bar{x}_j(t)\| > \varepsilon, \forall \nu_i, \nu_j \in C, \nu_i \neq \nu_j$  then
8:        $x_i(t) \leftarrow \bar{x}_i(t)$ 
9:       continue to next connected component
10:    end if
11:    Find the lead vehicles  $\nu_{\ell_1}, \nu_{\ell_2}$ 
12:    Compute the bubble space  $\mathcal{B}(\nu_{\ell_1}, \nu_{\ell_2})$ 
13:    if  $\mathcal{B}(\nu_{\ell_1}, \nu_{\ell_2})$  is found in Repository then
14:       $T_{\ell_1, \ell_2} \leftarrow \text{RRT}^*$  tree from Repository
15:       $(x_{\ell_1}(t), x_{\ell_2}(t)) \leftarrow$  best motion plan in  $T_{\ell_1, \ell_2}$ 
16:    else
17:       $(x_{\ell_1}(t), x_{\ell_2}(t)) \leftarrow$ 
        BubbleMinViolationRRT* $((\nu_{\ell_1}, \nu_{\ell_2}), \mathfrak{R}, g_i, \Theta_i)$  in
         $\mathcal{B}(\nu_{\ell_1}, \nu_{\ell_2})$ 
18:      Save the attribute graph of  $\mathcal{B}(\nu_{\ell_1}, \nu_{\ell_2})$  and the
        corresponding RRT* tree in Repository
19:    end if
20:    Compute the queue  $Q(t)$  containing  $C \setminus \{\nu_{\ell_1}, \nu_{\ell_2}\}$ 
21:    while  $Q(t) \neq \emptyset$  do
22:       $\nu_i \leftarrow Q(t).pop()$ 
23:       $x_i(t) \leftarrow$ 
        EnforcedMinViolationRRT* $(\nu_i, \mathcal{V}, \mathfrak{R}, g_i, \Theta_i)$  with
        rejecting samples leading to collisions
24:    end while
25:  end for
26:  for all  $\nu_i \in \mathcal{V}$  do
27:    Execute  $x_i(t)$  for  $\Delta t$ 
28:  end for
29: end while

```

---

#### A. Communication graph

The communication graph at time  $t$  is an undirected graph  $G(t) = (\mathcal{V}, E(t), W(t))$ , where the set of vertices  $\mathcal{V}$  is the set of vehicles,  $(\nu_i, \nu_j) \in E(t)$  if  $\nu_j \in \text{Sense}_i(y_i(t))$  or  $\nu_i \in \text{Sense}_j(y_j(t))$ , and  $W(t)(\nu_i, \nu_j) = \|y_i(t) - y_j(t)\|$ , for all  $(\nu_i, \nu_j) \in E(t)$ . The communication graph reflects dependencies between the vehicles in  $\mathcal{V}$ . Any two vehicles that are connected by an edge are at a potential risk of collision, while any two vehicles that are in two different connected components of  $G(t)$  can be safely treated independently at time  $t$ .

Suppose that the communication graph is connected. Otherwise, we apply the following to each connected component separately. We use the communication graph to determine the order in which the vehicles should be considered; first of all it should be the pair of vehicles that are the closest to a potential collision, called the *lead vehicles*  $L(t) = \{\nu_{\ell_1}(t), \nu_{\ell_2}(t)\}$ , which we will denote simply as  $\nu_{\ell_1}, \nu_{\ell_2}$  if  $t$  is clear from the context. Formally, this is the pair of vehicles in the opposite direction to each other satisfying  $(\nu_{\ell_1}, \nu_{\ell_2}) \in E(t)$ , and  $W(t)(\nu_{\ell_1}, \nu_{\ell_2}) \leq W(t)(\nu_i, \nu_j)$ , for all  $(\nu_i, \nu_j) \in E(t)$ , where  $\nu_i$  and  $\nu_j$  are in the opposite directions at time  $t$ . In the running example shown in Fig. 1, the lead vehicles are the green and the blue ones. The lead vehicles are subject to joint planning discussed in Sec. IV-B.

The rest of the vehicles are iteratively added to the *processing queue*  $Q(t)$ , which is initially empty. The next state to be added to  $Q(t)$  is

$$\nu = \arg \min_{\nu' \in Q(t) \cup L(t)} \min_{\nu' \in Q(t) \cup L(t), (\nu, \nu') \in E(t)} W(t)(\nu, \nu').$$

The vehicles in the processing queue  $Q(t)$ , such as the yellow one in Fig. 1 will be treated by enforced planning presented in Sec. IV-C, in the order they were placed in the queue.

#### B. Joint planning

Consider the lead vehicles  $\nu_{\ell_1}$  and  $\nu_{\ell_2}$  and assume that their nominal feasible state trajectories  $\bar{x}_{\ell_1}(t)$  and  $\bar{x}_{\ell_2}(t)$  yield a collision. We define the bubble space as the subset of the state space corresponding to the subset of the road segment where replanning should take place. Intuitively, this is the road segment part between the two vehicles. Here, for the simplicity of presentation, and without loss of generality, we assume that each road segment is modeled as a rectangle with bottom left corner in  $[0, 0]$  and top right corner in  $[x_{max}, y_{max}]$  and that it is associated with two directions: left to right (going from 0 towards  $x_{max}$  along the  $x$ -axis), and the opposite direction right to left (going from  $x_{max}$  towards 0 along the  $x$ -axis).

$$\mathcal{B}(\nu_{\ell_1}, \nu_{\ell_2}) = \{x \in X \mid \min(h(x_{\ell_1})_1, h(x_{\ell_2})_1) \leq h(x) \leq \max(h(x_{\ell_1})_1, h(x_{\ell_2})_1)\}, \quad (4)$$

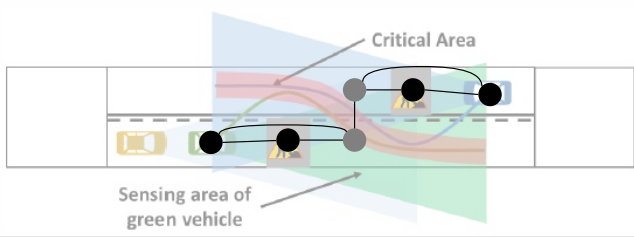
where  $h(x) = (h(x)_1, h(x)_2)$ .

1) *Planning in the bubble space:* To find a joint plan for  $\nu_{\ell_1}$  and  $\nu_{\ell_2}$  in the bubble space, we modify the minimum-violation scLTL RRT\* planner from [3] to plan in  $X^2$  by allowing to sample from  $\mathcal{B}(\nu_{\ell_1}, \nu_{\ell_2})$  only. Moreover, every time a new sample is taken, it is accepted only if it corresponds to a collision-free state in  $X^2$ , i.e., for  $x = (x_{\ell_1}, x_{\ell_2}) \in X^2$  we have  $\|h(x_{\ell_1}) - h(x_{\ell_2})\| > \varepsilon$ .

2) *Model matching:* For efficiency of computations, we furthermore propose a procedure to immediately obtain solutions without computations when we recognize a situation that has been addressed before.

Every time we perform joint planning in the bubble space, we store in a repository (possibly in the cloud) an attributed graph of the bubble space, which we treat as an object in the object recognition problem. The nodes of the attributed

graph correspond to the interesting features (e.g., obstacles, construction zones, vehicles) and lanes, and their attributes. Some examples of these attributes are: the type of feature (e.g., a construction zone), or its location in the scene. The edges represent adjacency of the interesting features. Furthermore, each of them is connected to the corresponding lane, and lanes are connected if they are adjacent, too. Together with the attributed graph, we store the RRT\* tree computed in the  $\mathcal{B}(\nu_{\ell_1}, \nu_{\ell_2})$ .



**Fig. 2:** A figure showing an attributed graph used for model matching in the traffic scenario from Example 1. Each black vertex corresponds to a feature of interest in the bubble space, in this case: construction zones and vehicles, and each gray vertex corresponds to a lane. Each vertex has attributes, for example, the attributes of the green vehicle are: *type*:  $\nu_2$ , *coordinates*:  $\{x, y\}$ , and the lanes have attributes: *type*: *right.lane* or *left.lane*, *corners*:  $\{x_{start}, y_{start}, x_{end}, y_{end}\}$ .

Hence, the first step after the construction of  $\mathcal{B}(\nu_{\ell_1}, \nu_{\ell_2})$  is not the joint planning itself, but an attempt to match the attributed graph model of the current bubble space to one stored in the repository. The model matching procedure is inspired by 3-D object recognition algorithms Kim et al. [13] and adapted to our 2-D case. First, several filters are implemented checking the number of edges, vertices, and their degrees to reduce the set of candidate matching bubble space models from the repository. Second, bipartite matching and validation steps find among the remaining models the matching bubble space of  $\mathcal{B}(\nu_{\ell_1}, \nu_{\ell_2})$ , if one exists. In such a case, we can directly use the stored RRT\* tree instead of recomputing it from scratch as described above.

### C. Enforced planning

After the lead vehicles' trajectories are set, we plan for the remaining vehicles in  $Q(t)$  one by one. For each vehicle  $\nu$ , we again build on the minimum-violation scLTL RRT\* algorithm from [3] with one major modification: we reject samples that lead to a collision with one of the trajectories of the already processed vehicles, i.e. either the lead vehicles, or the vehicles that are in  $Q(t)$  before  $\nu$ .

### D. Complexity

**Theorem 1** *The size of the attributed graph is  $O(n + m)$ , where  $n$  is the number of scene's features in the bubble space and  $m$  is the number of adjacencies between them. The filters are all in  $O(n + m)$ . The complexity of bipartite matching, and hence, the overall complexity of the model matching procedure is  $O(m\sqrt{n})$ .*

**Theorem 2** *Assume that all calls of the minimum scLTL RRT\* procedure are successful (lines 3, 17, 23 in Alg. 1), i.e., terminate and return motion plans in finite time. The complexity of each on-line planning step of Alg. 1 is  $O(|\mathcal{V}| \cdot |\Theta| \cdot N^2 \log N + |\mathcal{V}|^2 + \frac{|\mathcal{V}|}{2} \cdot C_{MM} \cdot |\Theta| \cdot N_J^2 \log N_J + |\mathcal{V}| \log |\mathcal{V}| + |\mathcal{V}| \cdot |\Theta| \cdot N_E^2 \log N_E)$ , where  $C_{MM}$  is the complexity of the model matching methods, and  $N$ ,  $N_J$ , and  $N_E$  are the maximum number of nodes in the RRT\* for the nominal, joint and enforced planning among all vehicles, respectively. Moreover, all primitive functions employed by the RRT\* algorithm are independent of the number of vehicles  $|\mathcal{V}|$ .*

*Proof:* The first term in the complexity bound  $|\mathcal{V}| \cdot |\Theta| \cdot N^2 \log N$  corresponds to generating the feasible nominal trajectories. From [3], we know that each step of the procedure takes  $O(|\Theta| \cdot N \log N)$ , and using Stirling's approximation formula we obtain the desired upper bound. Computing the communication graph takes at most  $|\mathcal{V}|^2$  when it is fully connected. Similarly to the nominal trajectory generation step, the joint planning step takes at most  $O(|\Theta| \cdot N_J^2 \log N_J)$ . Joint planning is performed for at least two vehicles at a time. Thus, there are at most  $\frac{|\mathcal{V}|}{2}$  connected components, and in the worst case each computing a bubble space. Creating all queues for all connected components takes at most  $O(|\mathcal{V}| \log |\mathcal{V}|)$  using for instance heap queues. Enforced planning takes  $O(|\Theta| \cdot N_E^2 \log N_E)$  similar to the previous steps. Again, in the worst case enforced planning is called for all vehicles, except the lead ones. Putting all terms together, we obtained the claimed bound.

Lastly, note that the RRT\* procedures are called for problems of dimensions that are independent of the number of vehicles. In the case of nominal and enforced planning, the statement is trivial, because the procedures are applied for one vehicle at a time. For the case of joint planning, note that the number of lead vehicles depends only on the local road geometry. For instance, a 4-way intersection would have at most 4 lead vehicles. A conservative upper bound for the number of lead vehicles is the number of lanes (all traffic directions) within a  $L_1$ -ball around a vehicle of radius equal to the size of the vehicles' sensing area. Thus, all primitives, including sampling, nearest neighbor, distance computation, and collision checking, are independent of the number of vehicles in the environment. ■

## V. EXPERIMENTAL RESULTS

We have implemented the solution in Python2.7 using the LVRmodRRT package [3]. All examples were ran on Intel Core i7 computer with 2.6Ghz processor and 8GB RAM under Ubuntu 14.04.

In this section, we demonstrate the suitability and the scalability of the proposed approach in several illustrative cases. The first one is depicted in Fig. 3, which is the running example from Fig. 1. There is a construction zone in the left-right lane as well as the right-left lane. Fig. 5 shows that the construction zones are in the limited sensing area of each of the vehicles. Furthermore, each vehicle detects all vehicles

ahead of it. This means that the fully centralized version of the minimum-violation scLTL would run in  $X^3$ .

The goal of the leftmost vehicle is to reach the goal region  $g_1$  on the right, and analogously for the two rightmost vehicles it is to reach the goal region  $g_2 = g_3$  on the left. The road rules that we consider are listed in Example 2. We set priority  $p_2 = 100$ , i.e. we give a high priority to the avoidance of the construction zones, while the remaining priorities are  $p_j = 1$ , for  $j = 1, 3, 4$  for the rules of staying in the right lane, not crossing the center line if overtaking is forbidden, and obeying the speed limit if enforced by a road sign.

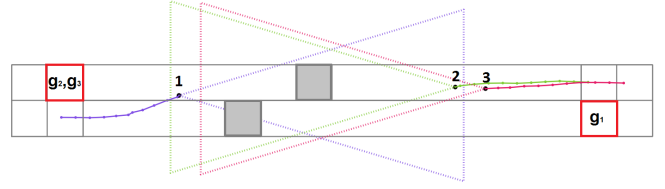
The progress of our solution in the work space is shown in Fig. 3 – 5. Fig. 3 shows the situation when vehicles  $\nu_2$  and  $\nu_3$  appear in the limited sensing area of vehicle  $\nu_1$ , and vice versa and the individual planning using the minimum-violation scLTL RRT\* planner [3] for each vehicle yields a risk of collision. The communication graph is the complete graph  $K_3$ , where the shortest edge is between vehicles  $\nu_1$  and  $\nu_2$ . These two become the lead vehicles. We compute the bubble space for them and since we could not find a corresponding bubble space in the repository, we perform joint planning using the minimum-violation scLTL RRT\* planner [3] in the bubble space. Vehicle  $\nu_3$  is in the processing queue. The resulting trajectories are illustrated in Fig. 6. Their computation took approximately 30 sec. The top two figures in Fig. 6 show the projection of the joint planning RRT\* tree onto the work spaces of the lead vehicles, whereas the bottom one shows the RRT\* tree of the remaining vehicle. Fig. 4 shows the trajectories after  $10\Delta t$ . The vehicles follow a trajectory that avoids the construction zones. Since the previously computed trajectories do not yield new risks of collisions, no recomputation is needed. This is recognized by our implementation in the order of milliseconds. Finally, Fig. 5 shows the situation after  $30\Delta t$ .

The extension of this example to 10 vehicles is illustrated in Fig. 7. Similarly as in the 3-vehicle case, the goal is to reach the marked out goal regions, where  $g_1 = g_6 = g_{10}$  and  $g_2 = g_3 = g_4 = g_5 = g_7 = g_8 = g_9$ . In this case, the obstacles are spread out further than in the running example. This means that the vehicles are able to traverse the bubble space simultaneously.

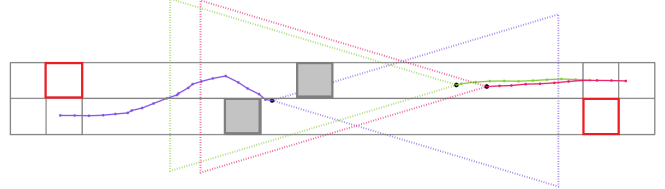
In this extended case, the communication graph is fully connected, where the closest edge is between  $\nu_3$  and  $\nu_4$ . The remaining  $\nu = \{1, 2, 5, 6, 7, 8, 9, 10\}$  are placed in the processing queue. The resulting RRT\* trees for this case is illustrated in Fig. 10.

## VI. CONCLUSION

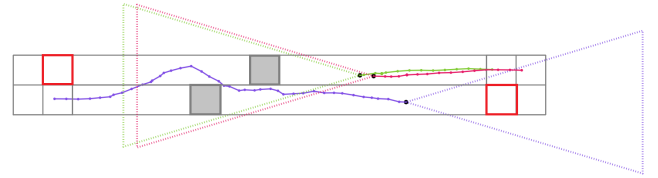
We have studied the problem of motion planning for fleets of autonomous vehicles that are given a set of customer demands and road rules specified in temporal logic. An approach has been proposed, that builds on a previously developed receding-horizon motion planner based on RRT\*. The algorithm is demonstrated using different illustrative cases.



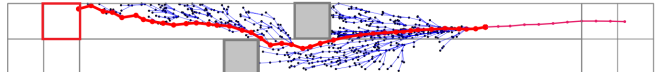
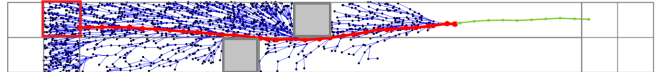
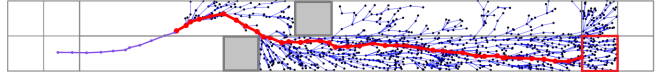
**Fig. 3:** Figure illustrating the path the three vehicles have traversed so far, the goal regions (in red), the obstacles (grey) and the limited sensing area. The scenario is after joint planning has been performed between  $\nu_1$  and  $\nu_2$ .



**Fig. 4:** Figure illustrating the path the three vehicles have traversed so far, the goal regions (in red), the obstacles (grey) and the limited sensing area. The scenario is approximately  $15 \Delta t$  after joint planning has been performed between  $\nu_1$  and  $\nu_2$ .

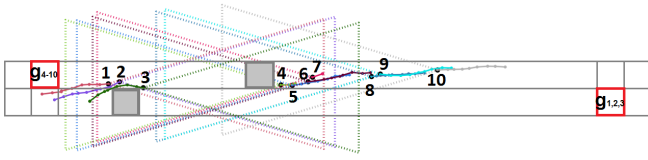


**Fig. 5:** Figure illustrating the path the three vehicles have traversed so far, the goal regions (in red), the obstacles (grey) and the limited sensing area. The scenario is approximately  $35 \Delta t$  after joint planning has been performed between  $\nu_1$  and  $\nu_2$ .

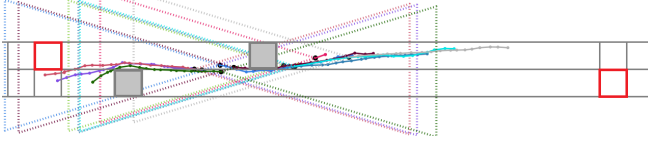


**Fig. 6:** Figure illustrating the path the three vehicles have traversed so far, the goal regions (in red), the obstacles (grey), the resulting RRT\* tree and the planned trajectory (highlighted in red). The scenario is after joint planning has been performed between  $\nu_1$  and  $\nu_2$ .

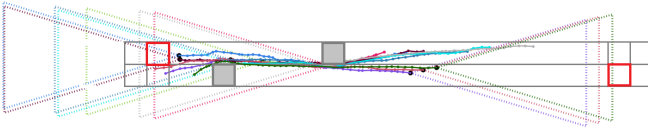
Future work includes accounting for the dynamics and size of the vehicles in question.



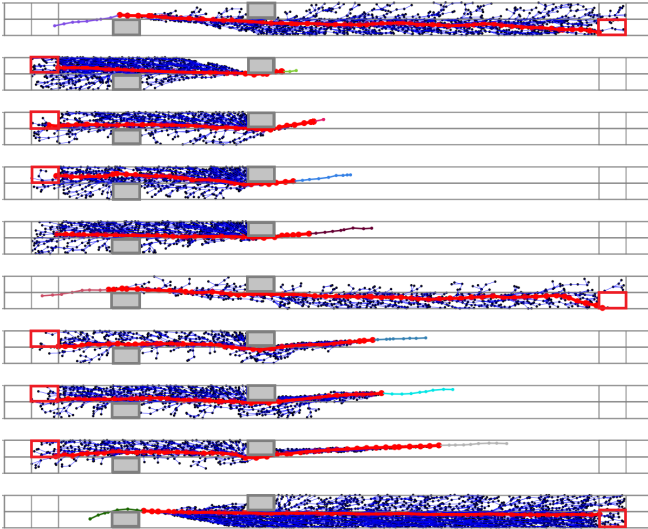
**Fig. 7:** Figure illustrating the path the ten vehicles have traversed so far, the goal regions (in red), the obstacles (grey) and the limited sensing area. The scenario is after joint planning has been performed between  $\nu_3$  and  $\nu_4$ .



**Fig. 8:** Figure illustrating the path the ten vehicles have traversed so far, the goal regions (in red), the obstacles (grey) and the limited sensing area. The scenario is approximately  $15 \Delta t$  after joint planning has been performed between  $\nu_3$  and  $\nu_4$ .



**Fig. 9:** Figure illustrating the path the ten vehicles have traversed so far, the goal regions (in red), the obstacles (grey) and the limited sensing area. The scenario is approximately  $35 \Delta t$  after joint planning has been performed between  $\nu_3$  and  $\nu_4$ .



**Fig. 10:** Figure illustrating the path the ten vehicles have traversed so far, the goal regions (in red), the obstacles (grey), the resulting  $RRT^*$  tree and the planned trajectory (highlighted in red). The scenario is after joint planning has been performed between  $\nu_3$  and  $\nu_4$ .

violating control strategy synthesis with safety rules,” in *International Conference on Hybrid Systems: Computation and Control*, Philadelphia, PA, USA, 2013, pp. 1–10.

- [3] C.-I. Vasile, J. Tumova, S. Karaman, C. Belta, and D. Rus, “Minimum-violation scLTL motion planning for mobility-on-demand,” in *IEEE International Conference on Robotics and Automation*, Singapore, Singapore, May 2017, pp. 1481–1488.
- [4] J. Tumova, S. Karaman, C. Belta, and D. Rus, “Least-violating planning in road networks from temporal logic specifications,” in *ACM/IEEE International Conference on Cyber-Physical Systems*, 2016, pp. 1–9.
- [5] K. Kim, G. Fainekos, and S. Sankaranarayanan, “On the minimal revision problem of specification automata,” *The International Journal of Robotics Research*, 2015.
- [6] M. Guo and D. V. Dimarogonas, “Multi-agent plan reconfiguration under local LTL specifications,” *International Journal of Robotics Research*, vol. 34, no. 2, pp. 218–235, Feb. 2015.
- [7] M. Maly, M. Lahijanian, L. E. Kavraki, H. Kress-Gazit, and M. Y. Vardi, “Iterative Temporal Motion Planning for Hybrid Systems in Partially Unknown Environments,” in *Int. Conference on Hybrid Systems: Computation and Control*, 2013.
- [8] E. F. M. Pavone, S. L. Smith and D. Rus, “Robotic load balancing for mobility-on-demand systems,” *International Journal of Robotics Research*, vol. 31, no. 7, pp. 839–854, 2012.
- [9] D. Miculescu and S. Karaman, “Polling-systems-based control of high-performance provably-safe autonomous intersections,” in *53rd IEEE Conference on Decision and Control*, Dec 2014, pp. 1417–1423.
- [10] C. Vasile and C. Belta, “Sampling-Based Temporal Logic Path Planning,” in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2013.
- [11] —, “Reactive Sampling-Based Temporal Logic Path Planning,” in *IEEE Int. Conference on Robotics and Automation*, Hong Kong, 2014.
- [12] O. Kupferman and M. Y. Vardi, “Model checking of safety properties,” *Formal Methods in System Design*, vol. 19, no. 3, pp. 291–314, 2001.
- [13] W.-Y. Kim and A. C. Kak, “3-d object recognition using bipartite matching embedded in discrete relaxation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 3, pp. 224–251, 1991.

## REFERENCES

- [1] L. Reyes Castro, P. Chaudhari, J. Tumova, S. Karaman, E. Frazzoli, and D. Rus, “Incremental sampling-based algorithm for minimum-violation motion planning,” in *IEEE Conference on Decision and Control*, 2013, pp. 3217–3224.
- [2] J. Tumova, G. C. Hall, S. Karaman, E. Frazzoli, and D. Rus, “Least-