Motion-planning with Global Temporal Logic Specifications for Multiple Nonholonomic Robotic Vehicles

Zetian Zhang* and Raghvendra V. Cowlagi*†

***Abstract*— We investigate motion-planning for a team of robotic vehicles assigned to a collaborative intelligent task in the form of global linear temporal logic (LTL) specifications. Specifically, we extend recent results from the literature to include nonholonomic kinematic constraints on the robotic vehicles. The problem formulation relies on workspace cell decompositions, where certain regions of interest in the robots' shared workspace are defined. The proposed algorithm involves two graphs: first, the topological graph arising from the workspace cell decomposition, and second, a graph arising from vertex aggregation on the previous graph. The main technical innovation is the application of the so-called *method of lifted graphs* to determine the feasibility of edge transitions in these graphs. We illustrate the proposed approach with numerical simulation examples.**

I. Introduction

The demand for higher degrees of autonomy in robotic vehicles has led to recent research in intelligent control, and specifically, in the synthesis of control laws subject to temporal logic (TL) specifications. TL specifications can concisely encode intelligent tasks for robotic vehicles. The problem of control law synthesis is then performed in two broad steps: (1) *discrete abstraction,* which involves the generation of a finite-state model of the vehicle's motion, and (2) the application of formal methods to determine sequences of transitions in this model that satisfy the given TL specifications. The first step of discrete abstraction is arguably more difficult, and is a subject of ongoing research in the context of nonlinear vehicle dynamical models.

We investigate the archetypal problem of motion-planning for a team of robotic vehicles assigned to a collaborative intelligent task in the form of global TL specifications to be satisfied by the team as a whole. Specifically, we extend recent results from the literature on this problem to include nonholonomic kinematic constraints on the robotic vehicles.

The application of formal methods to generate control laws satisfying TL specifications has been studied extensively for linear dynamical systems [1]–[4], including applications to mobile robotic vehicles [5]–[8]. Research in this area has largely focused on linear temporal logic (LTL) specifications [9]. State space partitioning is used for discrete abstraction of the dynamical system. More recently, discrete abstraction for nonlinear dynamical systems has been investigated [10], [11], including the application of randomized sampling-based methods [12]–[14].

*Aerospace Engineering Program, Worcester Polytechnic Institute, Worcester, MA, USA. {zzhang, rvcowlagi}@wpi.edu
† Corresponding author.

The literature on multi-vehicle robotic teams subject to LTL specifications falls into two broad categories. On the one hand, global specifications for the entire team are assumed, and centralized motion-planning algorithms for the team are developed [2], [15]. On the other hand, separate specifications for individual vehicles in the team are assumed, and distributed motion-planning algorithms for the team are developed [16], [17]. Distributed algorithms for satisfying global specifications are yet an open subject of research.

The proposed work falls into the former category. Specifically, we seek to extend the important work [15] to include nonholonomic kinematic constraints on the robotic vehicles. A crucial assumption in [15] is that a discrete abstraction of the vehicle dynamical model is readily available. This assumption is valid for holonomic vehicles that can be modeled as single or double integrators. However, for nonholonomic vehicles, the generation of a finite state model of the vehicle's motion is non-trivial, and is addressed in this paper.

We make two simplifying assumptions for the proposed algorithm development. The first assumption relates to temporal synchronization of vehicle motion: here, exact traversal times of vehicles are ignored. Instead, within the proposed finite state model of vehicle motion, one state transition is assumed to be carried out over one time-step. The justification for this assumption is the need for temporal abstraction in motion-planning to reduce complexity. The second assumption is related to the workspace partitioning used for motion-planning (see Section II): mutiple vehicles are allowed to be within the same region at the same time-step. The justifications for this assumption are that (a) owing to the preceding temporal abstraction, the duration of one time-step may suffice to create a posteriori a sufficient temporal gap in the presence of different vehicles in the same cell, and (b) the size of cells is assumed to be significantly larger than the dimensions of the vehicle, thereby allowing enough room for more than one robot within a cell.

In this paper, we propose a centralized motion-planning algorithm for a team of robotic vehicles subject to nonholonomic kinematic constraints and global LTL specifications. The problem formulation relies on workspace cell decompositions, where certain regions of interest in the robots' shared workspace are defined. The proposed algorithm involves two graphs: first, the topological graph $\mathcal{G}$ arising from the workspace cell decomposition, and second, a graph $\mathcal{G}^{\mathrm{R}}$ arising from vertex aggregation on $\mathcal{G}$, such that each region of interest is a vertex $\mathcal{G}^{\mathrm{R}}$. The main technical innovation in the proposed algorithm is the application of the *method of lifted graphs* to determine feasibility of edge transitions in $\mathcal{G}$ and

$\mathcal{G}^{\mathrm{R}}$. Briefly, this method allows the reachability properties of the vehicle kinematic model to be associated with these edge transitions. As in [15], a team transition system is first established. Next, a product transition system is constructed from this team transition system and the Büchi automaton associated with the global LTL specifications. Runs of this product transition system can be uniquely projected to paths (for each vehicle) that are compatible with the nonholonomic constraints and also ensure that the global LTL specifications are satisfied.

The main contributions of this paper are as follows. Firstly, we present an novel motion-planning method to enable a team of nonholonomic robots to satisfy global LTL specifications. This method relies on vertex aggregation in the team's shared workspace, and on the new method of lifted graphs. Secondly, the proposed motion-planning method relies on workspace partitioning, instead of state-space partitioning. Finally, we propose an incremental algorithm computing the desired motion plan. The proposed approach promises better scalability to higher-dimensional vehicle dynamical models.

The rest of this paper is organized as follows. We present the mathematical problem formulation in Section II. The idea of lifted graphs is discussed in Section III, whereas the team and product transition systems, and the overall motion-planningapproach are discussed in Section IV. We present illustrative numerical simulation results in Section V. Finally, we conclude the paper in Section VI with comments about the future work.

II. Problem Formulation

In this section, we introduce preliminary ideas involved in the problem formulation. Each robotic vehicle in a team of N^{V} vehicles is modeled as follows.

Vehicle model: Let $\xi = (x, y, \psi) \in \mathcal{D} := \mathbb{R}^2 \times \mathbb{S}^1$ denote the vehicle state, namely, the position of the vehicle center of mass and the direction of its velocity vector in a prespecified Cartesian coordinate system. The vehicle state evolves according to the differential equations

$$\dot{x}(t) = \cos\psi(t), \quad \dot{y}(t) = \sin\psi(t), \quad \dot{\psi}(t) = u(t), \tag{1}$$

where u is the control input. The set of admissible control input values is the interval $U := \left[-\frac{1}{\rho}, \frac{1}{\rho}\right]$, with $\rho > 0$. For every $t_{\mathrm{f}} \in \mathbb{R}_+$, let $\mathcal{U}_{t_{\mathrm{f}}}$ be the set of all piecewise continuous functions on $[0, t_{\mathrm{f}}]$ taking values in U. For any $u \in \mathcal{U}_{t_{\mathrm{f}}}$, and $\xi_0 \in \mathcal{D}$, the state trajectory $\xi(t; \xi_0, u)$, $t \in [0, t_{\mathrm{f}}]$, obtained by integrating (1) is called an *admissible state trajectory.*

Workspace cell decomposition: Let $\mathcal{W} \in \mathbb{R}^2$ denote a planar region where the robotic vehicles collaboratively operate. Consider a cell decomposition, i.e. a partition of $\mathcal{W}$ into convex subregions called *cells.* We denote by $N^{\mathrm{C}} \in \mathbb{Z}_+$ the number of cells, and by $R_i \subset \mathcal{W}$ the subregion associated with the i^{th} cell, for each $i = 1, \ldots, N^{\mathrm{C}}$. We associate with this partition a graph $\mathcal{G} := (V, E)$ such that each vertex of $\mathcal{G}$ is uniquely associated with a cell, and each edge of $\mathcal{G}$ is uniquely associated with a pair of geometrically adjacent cells. We denote by $\mathsf{cell}(v)$ the element of $\{R_i\}_{i=1}^{N^{\mathrm{C}}}$ associated with the vertex $v \in V$. A *path* $\mathbf{v}$ in $\mathcal{G}$ is a finite or infinite sequence $(v_0, v_1, \ldots)$ of vertices, such that $v_0, v_k \in V$, and $(v_{k-1}, v_k) \in E$, for each $k \in \mathbb{N}$. The number of vertices in a path is called its *length.* Note that, according to the preceding definition, a path in $\mathcal{G}$ can contain cycles. We denote by $\mathcal{L}_{\mathcal{G}}$ the collection of all paths in $\mathcal{G}$.

Special cells called *base locations* and *regions of interest (ROI)* are prespecified. The number of base locations is equal to the number N^{V} of robotic vehicles in the team, and each robot is assumed to start from one of these locations. The vertices in V associated with these cells are denoted by $v_{\mathrm{B},1}, \ldots, v_{\mathrm{B},N^{\mathrm{V}}}$. The number of ROIs is N^{R}. Each ROI is a connected union of cells, i.e. the k^{th} ROI is $\cup_{i \in \varsigma_k} R_i$, where $\varsigma_k \subseteq \{1, \ldots, N^{\mathrm{C}}\}$, $k = 1, \ldots, N^{\mathrm{R}}$ are prespecified. The associated vertices are denoted $v_{\mathrm{R},\ell k}$, $\ell \in \varsigma_k$.

For every $t_{\mathrm{f}} \in \mathbb{R}_+$ and $u \in \mathcal{U}_{t_{\mathrm{f}}}$, we define the *$\mathcal{G}$-trace* of a trajectory $\xi(\cdot; \xi_0, u)$ as the path $tr(\xi, \mathcal{G}) = (j_0, j_1, \ldots) \in \mathcal{L}_{\mathcal{G}}$ with minimum length such that $\mathsf{x}(\xi(0; \xi_0, u)) \in \mathsf{cell}(v_0)$, and

$$\mathsf{x}(\xi(t; \xi_0, u)) \in \cup_{k=0}^{P} \mathsf{cell}(v_k), \quad t \in [0, t_{\mathrm{f}}], \quad k \in \mathbb{N}. \tag{2}$$

We denote by $\mathcal{L}_{\Gamma}(\xi_0) \subseteq \mathcal{L}_{\mathcal{G}}$ the collection of $\mathcal{G}$-traces of all admissible trajectories for every $t_{\mathrm{f}} \in \mathbb{R}_+$. Informally, the path $tr(\xi, \mathcal{G})$ is associated with the sequence of cells that defines a "channel" in $\mathcal{W}$, such that the curve $\mathsf{x}(\xi(t))$, $t \in [0, t_{\mathrm{f}}]$, lies within this channel. The curve $\mathsf{x}(\xi(t))$ and the trajectory $\xi(t)$ are said to *traverse* this channel of cells.

LTL_{-X} specification: Linear temporal logic is a convenient formal language to express specifications on the behavior of a system over time. Similar to [3], we use a restricted version of LTL, namely, LTL$_{-X}$, which does not involve the *next* operator. The choice of LTL$_{-X}$ instead of LTL is for algorithmic simplicity in the proposed work. A brief overview of LTL$_{-X}$ is provided in Appendix I, and the reader is referred to [3], [9] for further details.

We consider atomic propositions associated with each ROI, namely $\lambda_k \equiv \mathsf{x}(\xi) \in \cup_{i \in \varsigma_k} R_i$, for each $k = 1, \ldots, N^{\mathrm{R}}$. The set of all atomic propositions is denoted $\Lambda = \{\lambda_k\}_{i=0}^{N^{\mathrm{R}}}$.

Each path $\mathbf{v} = (v_0, v_1, \ldots) \in \mathcal{L}_{\mathcal{G}}$ defines a word $\mathbf{w}(\mathbf{v}) = (w_0, w_1, \ldots,)$, where

$$w_\ell := \{\lambda_k \mid \mathsf{cell}(v_\ell) \subseteq \cup_{i \in \varsigma_k} R_i\}. \tag{3}$$

The path $\mathbf{v}$ is said to satisfy a formula ϕ if the word $\mathbf{w}(\mathbf{v})$ satisfies ϕ (as defined in Appendix I).

Satisfaction of global LTL_{-X} specifications: Consider paths $\mathbf{v}_n \in \mathcal{L}_{\mathcal{G}}$, $n = 1, \ldots, N^{\mathrm{V}}$, associated with the motion of each vehicle in the team. Each of these paths defines a word $\mathbf{w}_n(\mathbf{v}_n) = (w_{0,n}, w_{1,n}, \ldots)$, which we "concatenate" to define a word for the entire team as follows

$$\mathbf{w}(\pi) := (w_{0,1}, w_{0,2}, \ldots, w_{1,1}, w_{1,2}, \ldots), \tag{4}$$

where $\pi := (\mathbf{v}_1, \ldots, \mathbf{v}_{N^{\mathrm{V}}})$. Here, the rule of "concatenation" is that w_{ℓ_1,n_1} appears before w_{ℓ_2,n_2} in $\mathbf{w}(\pi)$ if $\ell_1 < \ell_2$ or else if $\ell_1 = \ell_2$ and $n_1 < n_2$, for $\ell_1, \ell_2 \in \mathbb{Z}_+$ and $n_1, n_2 \in \{1, \ldots, N^{\mathrm{V}}\}$. The n-tuple of paths π is said to collectively satisfy a formula ϕ if $\mathbf{w}(\pi)$ satisfies ϕ. The main problem of interest is then formulated as follows.

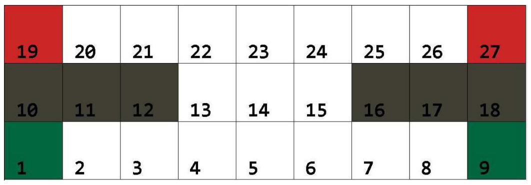


Fig. 1. An instance of Problem 1. Here, ROIs λ_1 and λ_2 are indicated in red (cells 19 and 27, respectively), and ROIs λ_3 and λ_4 are indicated in gray (cells 10–12 and 16–18, respectively).

Problem 1. *Given a LTL$_{-X}$ formula ϕ over Λ, and vehicle initial conditions $\xi_{0,n} \in \mathcal{D}$, for $n = 1, \ldots, N^{\mathrm{V}}$, determine*

$$\mathcal{L}_{\Gamma\phi} \subseteq \mathcal{L}_{\Gamma}(\xi_{0,1}) \times \ldots \times \mathcal{L}_{\Gamma}(\xi_{0,N^{\mathrm{V}}})$$

such that every N^{V}-tuple of paths in $\mathcal{L}_{\Gamma\phi}$ collectively satisfies the formula ϕ. $\square$

An instance of Problem 1 with two vehicles is illustrated in Fig. 1, where four ROIs (gray and yellow) and two base locations (green) are indicated. The given LTL specification is $(\Diamond\lambda_1) \wedge (\Diamond\lambda_2) \wedge (\Box\neg\lambda_3) \wedge (\Box\neg\lambda_4)$. This formula specifies that the yellow-colored ROIs must be eventually visited (no preference for the order of visit), and that the gray-colored ROIs must be always avoided. Intuitively, the expected motion plan will involve a vehicle from base $B1$ to visit ROI λ_1 and the second vehicle from base $B2$ to visit ROI λ_2. However, due to nonholonomic constraints, the sharp turns required to execute this plan may be infeasible, and an alternative plan is required.

III. Finite State Models of Vehicle Motion

For nonholonomic vehicle models, edge transitions in $\mathcal{G}$ (i.e., transitions between successive cells) are vehicle-state-dependent, especially when the cell dimensions are comparable to vehicle maneuvering characteristics [18]. Therefore, $\mathcal{G}$ does not suffice to serve as a finite state model of vehicle motion, as required by the aforementioned discrete abstraction step in motion-planning with LTL specifications. This observation is a critical point of distinction of the proposed work compared to, say, [15]. Therein, it is assumed that there exist vehicle maneuvers to enable edge transitions in a finite state system arising from workspace landmarks (i.e. similar to the workspace cell decomposition graph $\mathcal{G}$.) To incorporate some information in $\mathcal{G}$ about the vehicles' physical motion, we discuss the notion of a lifted graph, which was introduced in [19] in the context of motion-planning for nonholonomic vehicles.

A. Lifted Graph

Consider the workspace cell decomposition graph $\mathcal{G} = (V, E)$, and for every integer $H \geqslant 0$, define

$$\begin{aligned} V_H := \{&(v_0, \ldots, v_H) : (v_{k-1}, v_k) \in E,\ k = 1, \ldots, H, \\ &v_k \neq v_m, \quad \text{for } k, m \in \{0, \ldots, H\}, \text{ with } k \neq m\}. \end{aligned}$$

Every element $\mathbf{i} \in V_H$ is an ordered $(H+1)$-tuple of the elements of V, and this tuple corresponds to a sequence of successively adjacent cells. We denote by $[\mathbf{i}]_k$ the k^{th} element of $\mathbf{i}$, for $k < m \leqslant H + 1$. Let E_H be a set of all pairs $(\mathbf{i}, \mathbf{j})$, with $\mathbf{i}, \mathbf{j} \in V_H$, such that $[\mathbf{i}]_k = [\mathbf{j}]_{k-1}$, for every $k = 2, \ldots, H+1$, and $[\mathbf{i}]_1 \neq [\mathbf{j}]_{H+1}$. The *lifted graph* $\mathcal{G}_H$ is defined as the directed graph (V_H, E_H). Every path $\mathbf{v} = (v_0, v_1, \ldots)$ in the graph $\mathcal{G}$ can be uniquely mapped to a path $\mathbf{v}^h = (\mathbf{i}_0, \mathbf{i}_1, \ldots)$ in $\mathcal{G}_H$, where $\mathbf{i}_k = (v_k, v_{k+1}, \ldots, v_{k+H}) \in V_H$ for each $k \in \mathbb{N}$. We refer to the construction of $\mathcal{G}_H$ from $\mathcal{G}$ as *lifting* of $\mathcal{G}$.

B. Edge Transition Costs in $\mathcal{G}_H$

The primary benefit of lifting $\mathcal{G}$ is that edge transitions costs in $\mathcal{G}_H$ can encode characteristics of vehicles' workspace traversal in accordance with its kinematic and dynamic constraints. One example of such edge transition costs is the following [20], which associates certain forward- and backward reachability of the vehicle dynamical model with edge transitions in $\mathcal{G}_H$

Consider an element $(\mathbf{i}, \mathbf{j}) \in E_H$, with $\mathbf{i}, \mathbf{j} \in V_H$. Let $\mathcal{S}(\mathbf{i}) \subset \mathcal{D}$ be a set of states associated with $\mathbf{i} \in V_H$ such that $\mathsf{x}(\mathcal{S}(\mathbf{i})) \subseteq \mathsf{cell}([\mathbf{i}]_1) \cap \mathsf{cell}([\mathbf{i}]_2)$. Thus, the position components of the elements in $\mathcal{S}(\mathbf{i})$ lie on the boundary between the first and second cells corresponding to the vertices of V that constitute the ordered H-tuple $\mathbf{i}$. Next let $\mathcal{Q}(\mathbf{j}) \subset \mathcal{D}$ be a set of states such that $\mathsf{x}(\mathcal{Q}(\mathbf{j})) \subseteq \mathsf{cell}([\mathbf{j}]_1) \cap \mathsf{cell}([\mathbf{j}]_2)$ and for every state $\xi_{\mathrm{q}} \in \mathcal{Q}(\mathbf{j})$ there exists a traversal time t_{q} and an admissible control input $u_{\mathrm{q}} \in \mathcal{U}_{t_{\mathrm{q}}}$ such that $\mathsf{x}(\xi(t; \xi_{\mathrm{q}}, u_{\mathrm{q}})) \in \bigcup_{k=1}^{H+1} \mathsf{cell}([\mathbf{j}]_k)$, for all $t \in [0, t_{\mathrm{q}}]$. Informally, $\mathcal{Q}(\mathbf{j})$ is the set of all states whose position components lie on the boundary between the first and second cells of $\mathbf{j}$, and such that the traversal of the geometric region defined by the cells associated with the tuple $\mathbf{j}$ is possible from any initial state within $\mathcal{Q}(J)$.

Next, let $\mathcal{R}_{\mathbf{i}} : \mathcal{S}(\mathbf{i}) \to 2^{\mathcal{D}}$ be a reachability map associated with the sets $\mathcal{S}(\mathbf{i})$, defined by

$$\begin{aligned} \mathcal{R}_{\mathbf{i}}(\xi_{\mathrm{s}}) := \Big\{ &\xi_{\mathrm{t}} \in \mathcal{D} \mid \xi_{\mathrm{t}} \in \cup_{t \in \mathbb{R}_+} \cup_{u \in \mathcal{U}_t} \xi(t; \xi_{\mathrm{s}}, u), \text{ and} \\ &(\cup_{\tau \in [0,t]} \mathsf{x}(\xi(\tau; \xi_{\mathrm{s}}, u))) \cap (\mathcal{W} \backslash \mathsf{cell}([\mathbf{i}]_2)) = \varnothing \Big\}, \end{aligned} \tag{5}$$

where $\xi_{\mathrm{s}} \in \mathcal{S}(\mathbf{i})$ and $2^{\mathcal{D}}$ is the collection of all subsets of $\mathcal{D}$. Informally, $\mathcal{R}_{\mathbf{i}}(\xi_{\mathrm{s}})$ is the set of all states that can be reached from ξ_{s} by trajectories whose position components always remain within the second cell of $\mathbf{i}$. Finally, for $(\mathbf{i}, \mathbf{j}) \in E_H$, define $\hat{\mathcal{S}}(\mathbf{i}, \mathbf{j}) := \{\xi_{\mathrm{s}} \in \mathcal{S}(\mathbf{i}) : \mathcal{R}_{\mathbf{i}}(\xi_{\mathrm{s}}) \cap \mathcal{Q}(\mathbf{j}) \neq \varnothing\}$.

Now consider a path $\mathbf{v} = \{v_0, v_1, \ldots\} \in \mathcal{L}_{\mathcal{G}}$. For $k \in \mathbb{N}$, let $\mathbf{i}_k := (v_k, \ldots, v_{k+H})$. Clearly, $(\mathbf{i}_k, \mathbf{i}_{k+1}) \in E_H$. We define $g_H : E_H \to \mathbb{R}_+$ and $\mathcal{S}(\cdot)$ as follows:

$$\mathcal{S}(\mathbf{i}_{k+1}) := \bigcup_{\xi_{\mathrm{s}} \in \hat{\mathcal{S}}(\mathbf{i}_k, \mathbf{i}_{k+1})} \left(\mathcal{R}_{\mathbf{i}_k}(\xi_{\mathrm{s}}) \cap \mathcal{Q}(\mathbf{i}_{k+1}) \right), \tag{6}$$

$$g_H(\mathbf{i}_k, \mathbf{i}_{k+1}) := \begin{cases} \chi, & \text{if } \mathcal{S}(\mathbf{i}_{k+1}) = \varnothing, \\ 1, & \text{otherwise}, \end{cases} \tag{7}$$

where $\chi \gg 1$. The transition cost of $(\mathbf{i}_k, \mathbf{i}_{k+1}) \in E_H$ is $g_H(\mathbf{i}_k, \mathbf{i}_{k+1})$. Finally, the *$H$-cost* of a path $\mathbf{v} \in \mathcal{L}_{\mathcal{G}}$ is:

$$\mathcal{J}_H(\mathbf{v}) := H + \sum_{k=0}^{P-H} g_H(\mathbf{i}_k, \mathbf{i}_{k+1}). \tag{8}$$

$$\begin{tikzcd}
\mathcal{G} \arrow[r, "\text{Aggregation}"] \arrow[d, "\text{Lifting}"'] & \mathcal{G}^{\mathrm{R}} \arrow[r, "\text{Lifting}"] & \mathcal{G}^{\mathrm{R}}_{M} \\
\mathcal{G}_H \arrow[rr, dashed, "\text{Calculation of}", "\text{Transition Costs}"'] & & {} \arrow[u, dashed]
\end{tikzcd}$$

Fig. 2. Conceptual relationship between the various graphs involved.

Algorithms for computing the sets $\mathcal{R}(\cdot), \mathcal{S}(\cdot)$, and $\mathcal{Q}(\cdot)$ are described, based on geometric arguments, in [21].

The following observation is crucial for the proposed motion-planning algorithm.

Proposition 1. *Let $\mathbf{v} = (v_0, \ldots, v_P)$ be a path with length less than χ in $\mathcal{L}_{\mathcal{G}}$, and let $\xi_0 \in \mathcal{D}$ be prespecified such that $\mathsf{x}(\xi_0) \in \mathsf{cell}(v_0) \cap \mathsf{cell}(v_1)$. Then $\mathbf{v} \in \mathcal{L}_{\Gamma}(\xi_0)$ if and only if $\mathcal{J}_H(\mathbf{v}) < \chi$.*

Proof. See Appendix II. □

The lifted graph $\mathcal{G}_H$ with the preceding edge transition costs is a finite state model for an individual vehicle's motion. Proposition 1 characterizes the paths in $\mathcal{G}$ that can be feasibly traversed while satisfying the vehicle's kinematic and dynamic constraints.

The motion of the entire team of vehicles can be modeled by N^{V} copies of $\mathcal{G}_H$. Specifically, the idea is to construct a *team transition system* [15] through a Cartesian product-like operation with N^{V} copies of $\mathcal{G}_H$. Whereas this approach is viable, it is wasteful when the number of ROI N^{R} is small compared to the number of cells N^{C}. Instead, we consider an undirected graph – henceforth referred to as the *ROI graph* – $\mathcal{G}^{\mathrm{R}} = (V^{\mathrm{R}}, E^{\mathrm{R}})$, which is constructed by aggregating vertices in $\mathcal{G}$. To this end, we introduce a map $ROI : 2^V \to V^{\mathrm{R}}$ that uniquely associates each vertex in V^{R} with either a ROI or a robot base, i.e. the total number of vertices in V^{R} is $N^{\mathrm{R}} + N^{\mathrm{V}}$. Precisely:

$$V^{\mathrm{R}} := \bigcup_{k=1,\ldots,N^{\mathrm{R}}} ROI(\{v_{\mathrm{R},\ell k}\}_{\ell\in\varsigma_k}) \cup \bigcup_{k=1,\ldots,N^{\mathrm{V}}} ROI(v_{\mathrm{B},k}).$$

The edge set E^{R} is defined to be complete, i.e. each vertex in V^{R} is adjacent to every other vertex in V^{R}. Informally, paths in $\mathcal{G}^{\mathrm{R}}$ "abstractly" denote vehicle routes to complete the specified task (LTL formula). In a minor abuse of notation we denote by ROI^{-1} the set association of every element $q \in V^{\mathrm{R}}$ with vertices in V, e.g., $ROI^{-1}(q) = \{v_{\mathrm{R},\ell k}\}_{\ell\in\varsigma_k}$.

The finite state model of the vehicle team in the proposed approach is intricately linked with the motion-planning algorithm, which we discuss in the next Section. Informally, this finite state model is obtained by lifting $\mathcal{G}^{\mathrm{R}}$ to construct $\mathcal{G}^{\mathrm{R}}_M$ for $M \geqslant 0$ followed by a product-like operation with N^{V} copies of $\mathcal{G}^{\mathrm{R}}_M$. Transition costs in $\mathcal{G}^{\mathrm{R}}_M$ are assigned by computing optimal paths in $\mathcal{G}_H$. Figure 2 illustrates the relationship between the various graphs discussed here.

IV. Multi-Vehicle Motion-planning

For $H, M \geqslant 1$, consider $\mathcal{G}^{\mathrm{R}}_M$. According to the definition of lifting in Section III-A, each edge transition in $\mathcal{G}^{\mathrm{R}}_M$ corresponds to a sequence of $M+2$ locations (either ROIs or base locations) in $\mathcal{G}$. Accordingly, we can assign a transition cost to this edge in $\mathcal{G}^{\mathrm{R}}_M$ based on a vehicle's cost of traversal between these $M + 2$ locations.

To this end, the *team state* $\mu = (\mathbf{q}_1, \ldots, \mathbf{q}_{N^{\mathrm{V}}}) \in (V^{\mathrm{R}}_M)^{N^{\mathrm{V}}}$ is defined as the N^{V}-tuple of vertices in V^{R}_M indicating the location of each vehicle. Specifically, the team state μ indicates that the k^{th} vehicle is at the location associated with $[\mathbf{q}_k]_1 \in V^{\mathrm{R}}$. We denote by Q_M the set of all team states. A *team state transition* is the relation $\delta_{\mathrm{R}} \subseteq (V^{\mathrm{R}}_M)^{2N^{\mathrm{V}}}$ defined by

$$(\mu, \gamma) \in \delta_{\mathrm{R}} \text{ iff } (\mathbf{q}_k, \mathbf{r}_k) \in E^{\mathrm{R}}_M, \text{ for each } k = 1, \ldots, N^{\mathrm{V}},$$

where $\mu = (\mathbf{q}_1, \ldots, \mathbf{q}_{N^{\mathrm{V}}})$ and $\gamma = (\mathbf{r}_1, \ldots, \mathbf{r}_{N^{\mathrm{V}}})$. Note that each $(\mu, \gamma) \in \delta_{\mathrm{R}}$ is associated with N^{V} sequences $\{([\mathbf{q}_k]_1, \ldots, [\mathbf{q}_k]_{M+1}, [\mathbf{r}_k]_{M+2})\}$ of vertices of V^{R}, where $k = 1, \ldots, N^{\mathrm{V}}$. Let $\mathbf{v}^*_k(\mu, \gamma) \in \mathcal{L}_{\mathcal{G}}$ denote a path with minimal H-cost that passes through all of the vertices $ROI^{-1}([\mathbf{q}_k]_1), \ldots, \quad ROI^{-1}([\mathbf{q}_k]_{M+1}), ROI^{-1}([\mathbf{r}_k]_{M+2})$ in that order. Then we define the cost of the transition (μ, γ) by $\sum_{k=1}^{N^{\mathrm{V}}} \mathcal{J}_H(\mathbf{v}^*_k(\mu, \gamma))$.

The computation of these transition costs is time-consuming, but these computations can be preprocessed offline, and therefore do not constitute a computational burden on the proposed motion-planning algorithm.

Assuming that the initial location of the k^{th} vehicle is the k^{th} base location, we can define a set $Q_{0,M} \subset (V^{\mathrm{R}}_M)^{N^{\mathrm{V}}}$ of initial team states as:

$$Q_{0,M} := \{\mu \in Q_M \mid [\mathbf{q}_k]_1 = ROI(v_{\mathrm{B,k}}), \quad k = 1, \ldots, N^{\mathrm{V}}\}.$$

The triplet $\mathcal{Q}_M = (Q_M, Q_{0,M}, \delta_{\mathrm{R}})$ defines the *team state transition system*, which is a finite state model of the motion of the entire vehicle team.

Next, we turn to the satisfaction of the given LTL specifications. It is known [22], [23] that every LTL formula ϕ over the alphabet Λ is associated with a Büchi automaton $\mathcal{B}_\phi$ with input alphabet 2^Λ, such that the collection of accepting runs of $\mathcal{B}_\phi$ is exactly the collection of infinite strings over Λ that satisfy ϕ. A Büchi automaton is a finite state machine that accepts infinite input words. The acceptance condition of a Büchi automaton involves a set of accepting states that must be visited infinitely often during any run. Algorithms for translating a LTL formula to the associated Büchi automaton are available [23]–[25]. For the Büchi automaton $\mathcal{B}_\phi$, we denote by S the set of states, by $\delta_{\mathcal{B}_\phi} \subseteq S \times 2^\Lambda \times S$ the transition relation, and by $S_0, S_{\mathrm{f}} \subseteq S$, respectively, the sets of initial and accepting states.

Next, we define a *product transition system* $\mathcal{T}_{\phi,M} := (T, \delta_{\mathcal{T}_{\phi,M}})$ as the product of $\mathcal{Q}$ and $\mathcal{B}_\phi$ as follows:

1) The set of states of $\mathcal{T}_{\phi,M}$ is $T := S \times Q_M$. For every state $\theta \in T$, we denote by $\theta|_S$ and $\theta|_{Q_M}$, respectively, the projection of θ on S and Q_M.
2) The transition relation of $\mathcal{T}_{\phi,M}$ is $\delta_{\mathcal{T}_{\phi,M}} \subseteq T \times 2^\Lambda \times T$ defined as the set of all triplets $(\theta_k, w_k, \theta_\ell)$ such that

$$(\theta_k|_S, w_k, \theta_\ell|_S) \in \delta_{\mathcal{B}_\phi}, \qquad (\theta_k|_{Q_M}, \theta_\ell|_{Q_M}) \in \delta_{\mathrm{R}}, \quad (9)$$

$$w_k = \left\{\lambda_i \mid \theta_k|_{Q_M} \subseteq (\mathbf{q}_{\mathrm{R},1}, \ldots, \mathbf{q}_{\mathrm{R},N^{\mathrm{V}}})\right\}. \quad (10)$$

Here $\mathbf{q}_{\mathrm{R},n} \in V_M^{\mathrm{R}}$ is an tuple that contains $ROI(\{v_{\mathrm{R},\ell i}\}_{\ell \in \varsigma_i})$.

A *run* of $\mathcal{T}_{\phi,M}$ is a sequence $\Theta = (\theta_0, \theta_1, \ldots,)$ such that $\theta_k \in T$ for each $k \in \mathbb{N}$, and $(\theta_k, w_k, \theta_{k+1}) \in \delta_{\mathcal{T}_{\phi,M}}$, with w_k as defined in (10). We denote by $\Theta|_S = (\theta_0|_S, \theta_1|_S, \ldots,)$ and $\Theta|_{Q_M} = (\theta_0|_{Q_M}, \theta_1|_{Q_M}, \ldots,)$, respectively, the projections of Θ on S and Q_M. Note that $\Theta|_{Q_M}$ corresponds to N^{V} paths in $\mathcal{L}_{\mathcal{G}}$, i.e. one path for each vehicle in the team. The k^{th} path $\pi_k(\Theta) \in \mathcal{L}_{\mathcal{G}}$ is defined by the concatenation

$$\pi_k(\Theta) := (\mathbf{v}_k^*(\theta_0|_{Q_M}, \theta_1|_{Q_M}), \mathbf{v}_k^*(\theta_1|_{Q_M}, \theta_2|_{Q_M}), \ldots)$$

Similar to [3], [15], we restrict attention to runs of $\mathcal{T}_{\phi,M}$ of a "prefix-suffix" form $\Theta = (\Theta_{\mathrm{p}}, \Theta_{\mathrm{s}}, \Theta_{\mathrm{s}}, \ldots,)$. Here, the "suffix" run $\Theta_{\mathrm{s}} = (\theta_{\mathrm{f}}, \ldots, \theta_{\mathrm{f}})$, which is repeated infinitely often in Θ, is a finite sequence such that $\theta_{\mathrm{f}} \in S_{\mathrm{f}} \times Q_M$. The "prefix" run $\Theta_{\mathrm{p}} = (\theta_0, \ldots, \theta_P)$ is a finite sequence such that $\theta_0 \in S_0 \times Q_M$ and $(\theta_P, w_P, \theta_{\mathrm{f}}) \in \delta_{\mathcal{T}_{\phi,M}}$.

Now we state the main result of this paper as follows.

Theorem 1. *Let $\Theta = (\Theta_{\mathrm{p}}, \Theta_{\mathrm{s}}, \Theta_{\mathrm{s}}, \ldots,)$ be a run of $\mathcal{T}_{\phi,M}$. If $\mathcal{J}_H(\pi_k(\Theta_{\mathrm{p}})) < \chi$ and $\mathcal{J}_H(\pi_k(\Theta_{\mathrm{s}})) < \chi$, for each $k = 1, \ldots, N^{\mathrm{V}}$, then*

$$(\pi_1(\Theta), \ldots, \pi_{N^{\mathrm{V}}}(\Theta)) \in \mathcal{L}_{\Gamma\Phi}.$$

Proof. See Appendix II. □

Theorem 1 solves Problem 1 by characterizing the multi-vehicle paths that are not only feasible for traversal by the vehicles, but also satisfy the global LTL specification. The search for such multi-vehicle paths can be performed, in principle, by executing Dijkstra's algorithm to find an optimal run of the product transition system $\mathcal{T}_{\phi,M}$. However, this approach can be computationally expensive. On the one hand, if a large value of M is chosen, there may be a significant computational delay before *any* result is found. On the other hand, it is beneficial to choose large values of M because the total cost of paths is a nonincreasing function of M (similar to [26, Prop. 2]). To remedy this situation, we propose an incremental approach for finding optimal plans in $\mathcal{T}_{\phi,M}$, which iteratively increases the value of M.

The proposed algorithm is shown in pseudocode-form Fig. 3. A fixed value of $H \geqslant 1$, with higher values of H preferred. As a preprocessing step, we compute the optimal paths $\mathbf{v}_k^*(\mu, \gamma)$ for all transitions (μ, γ) in δ_{R}, for various values of M, and the H-costs of these paths are stored in a lookup table.

The main algorithm is initialized with $M = 0$, and an optimal run in the product transition system $\mathcal{T}_{\phi,M}$ is identified using, say, Dijkstra's algorithm. This run is projected to paths for individual vehicles, and the H-costs of these paths are calculated. By Prop. 1, these paths are feasible for traversal by admissible vehicle trajectories iff their H-costs are less than χ. If all the paths are found to be feasible, then their total cost is recorded and a feasible solution to the overall problem is now available. The algorithm then increments M, either to determine a first feasible solution, or to reduce the cost of the last known feasible solution. This algorithm can be terminated either when the first feasible solution is found, or whenever a predetermined maximum execution time is reached. It is easy to show that this algorithm is *complete*, i.e., if M and H are large enough, then this algorithm will find a feasible solution to the overall motion-planning problem if it exists (see [26, Proposition 2] for the proof of a similar statement).

Incremental Multi-Vehicle Motion-planning

```
1:  $M := 0$, choose $H \geqslant 1$
2:  total_cost := $\chi N^{\mathrm{V}}$
3:  while true do
4:      Find an optimal run $\Theta^*$ of $\mathcal{T}_{\phi,M}$
5:      all_paths_feasible := true
6:      for $k = 1, \ldots, N^{\mathrm{V}}$ do
7:          if $\mathcal{J}_H(\pi_k(\Theta^*)) \geqslant \chi$ then
8:              all_paths_feasible := false
9:              break;
10:     if all_paths_feasible then
11:         total_cost := $\min\left(\text{total\_cost}, \sum_{k=1}^{N^{\mathrm{V}}} \mathcal{J}_H(\pi_k(\Theta^*))\right)$
12:     $M := M + 1$
```

Fig. 3. Pseudocode description of the proposed incremental algorithm for solving the multi robot task/path planning problem.

V. Illustrative Example and Discussion

Figure 4 illustrates the application of the proposed algorithm to the sample problem introduced in Fig. 1. The proposed motion-planning algorithm starts with $M = 0$, and preprocessed information with $H = 0$. The product transition system $\mathcal{T}_{\phi,0}$ is searched. The resulting paths for each vehicle is shown in Fig. 4(a), and matches the intuitively expected solution discussed in Section I. Since $H = 0$, no information regarding the vehicle dynamical model is included.

The result of executing the proposed algorithm with $M = 0$ and $H = 3$ is shown in Fig. 4(b). Here, the minimum H-cost of any path between the base $B1$ and ROI λ_1 (and similarly, between base $B2$ and ROI λ_2) is found to be greater than χ. The solution illustrated in Fig. 4(b) indicates that $\pi_2 = \varnothing$, and

$$\begin{aligned}\pi_1 =& (1, 2, 3, 4, 13, 14, 15, 24, 25, 26, 27, 26, 25, 24, \\ & 23, 22, 21, 20, 19).\end{aligned}$$

These paths for the two vehicles satisfy the given specification of visiting ROIs 19 and 27, and avoiding the gray-colored ROIs. However, the overall path π_1 is still infeasible for traversal due to the sharp change in direction at ROI 27. The reason for this solution is that $M = 0$, and only single edge transitions in the ROI graph are encoded with H-costs of paths. In this case, with $H = 3$, the segments $(1, 2, 3, 4, 13, 14, 15, 24, 25, 26, 27)$ and $(27, 26, 25, 24, 23, 22, 21, 20, 19)$ of π_1 *each* have H-cost lower than χ, but their concatenation π_1 does not.

The algorithm finds this infeasibility of traversal in Line 9 of the proposed algorithm, and proceeds to search for another solution the the product transition system with $M = 1$. The result is illustrated in Fig. 4(c), which indicates the paths for the two vehicles as:

$$\pi_1 = (1, 2, 3, 4, 13, 14, 15, 24, 25, 26, 27),$$
$$\pi_2 = (9, 8, 7, 6, 15, 14, 13, 22, 21, 20, 19).$$

Both of these paths are feasible for traversal (i.e. the H-cost of each path is less than χ), and together the global specification is satisfied.

Figure 5 illustrates the result of application of the proposed algorithm to a 4-vehicle problem. The LTL specifications are similar to the previous problem.

Next, we address an issue that arises due to the simplifying assumption related to temporal synchronization, as discussed in Section I. Consider the workspace shown in Fig. 6(a), where two base locations (green) and two ROIs to be visited (red) are indicated. If the order of visit is irrelevant (e.g. the LTL specification $\phi = (\Diamond\lambda_1) \wedge (\Diamond\lambda_2)$, then temporal synchronization is not important. In this case, the projection of a run of the product system to Q_M, namely, $\Theta|_{Q_M}$ is

$$\Theta|_{Q_M} = \Big((v_{\mathrm{B},1}, v_{\mathrm{B},2}), (v_{\mathrm{R},1}, v_{\mathrm{B},2}), (v_{\mathrm{R},1}, v_{\mathrm{R},2})\Big).$$

This result means that the paths (in the ROI graph) to be followed by the two vehicles are, respectively, $(v_{\mathrm{B},1}, v_{\mathrm{R},1})$ and $(v_{\mathrm{B},2}, v_{\mathrm{R},2})$ for the vehicles starting from base locations 1 and 2. These paths are illustrated in Fig. 6(b).

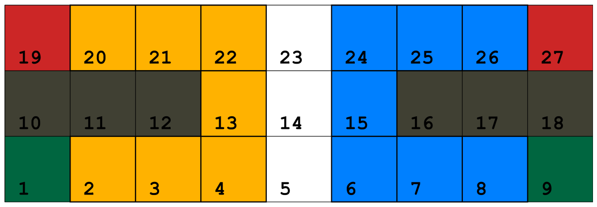
Now, consider the formula $\phi = \Diamond(\lambda_2 \wedge (\Diamond\lambda_1))$, which specifies that λ_2 be visited before λ_1. In this case, the projection of a run of the product system to Q_M results in

$$\Theta|_{Q_M} = \Big((v_{\mathrm{B},1}, v_{\mathrm{B},2}), (v_{\mathrm{B},1}, v_{\mathrm{R},2}), (v_{\mathrm{R},1}, v_{\mathrm{R},2})\Big).$$

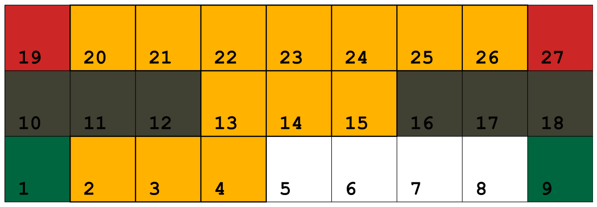
Because we have excluded temporal synchronization, the paths for the individual vehicles in these two cases are the same. However, it is clear from the two expressions for $\Theta|_{Q_M}$ that temporal information is encoded within the ROI graph paths, and needs to additionally encoded with time-stamps for temporal synchronization.

VI. Conclusions and Future Work

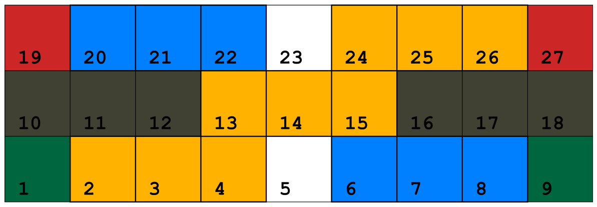
We presented a multi-vehicle motion-planning algorithm for satisfying global LTL specifications, such that the resultant paths are traversable with admissible trajectories of the vehicle dynamical model. The overall centralized planning approach is an extension to recent results in the literature to include vehicle kinematic and dynamic constraints in multi-vehicle motion-planning with LTL specifications. The main idea proposed was that of lifting graphs to associate with edge transitions in lifted graphs information about traversal with admissible vehicle trajectories. An incremental motion-planning algorithm was proposed, and illustrative numerical simulation results were presented. Future work includes removing the previously discussed simplifying assumptions, including temporal synchronization, and the extension to LTL specifications involving the *next* operator.



(a) Result of searching the product transition system with $M = 0, H = 0$.



(b) Result with $M = 0, H = 3$.



(c) Result with $M = 1, H = 3$.

Fig. 4. Solutions to the motivating example in Fig. 1. The yellow- and blue-colored cells indicate, respectively, the paths planned for the first and second vehicle (additional details are in Section V).

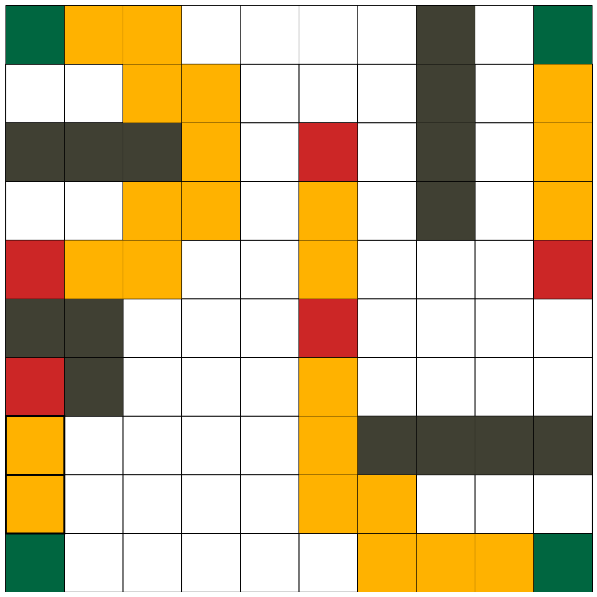
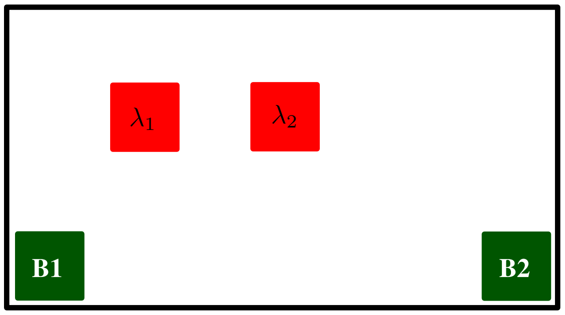
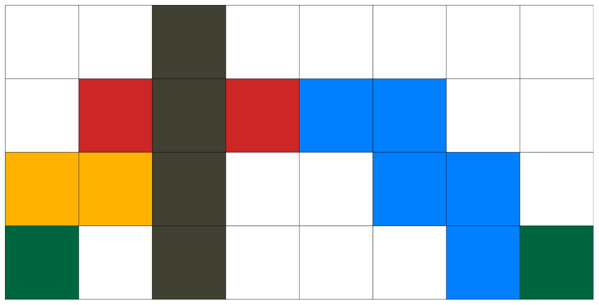


Fig. 5. Illustration of motion-planning for 4 vehicles. The LTL specification is similar to that of the example in Fig. 1. Here, ROIs to be always avoided are indicated in gray, and ROIs to be eventually visited are indicated in red (order of visit is not relevant). The base locations are indicated in green, and the paths found for each vehicle are indicated in yellow.



(a) Conceptual illustration of workspace.



(b) Illustration of numerical simulation result.

Fig. 6. Illustration of motion-planning when temporal synchronization is important. The paths of the two vehicle are indicated with yellow- and blue-colored cells.

REFERENCES

[1] P. Tabuada and G. J. Pappas, "Model checking LTL over controllable linear systems is decidable," in *Hybrid Systems: Computation & Control* (O. Maler and A. Pnueli, eds.), LNCS 2623, pp. 498–513, Berlin: Springer-Verlag, 2003.

[2] M. Kloetzer and C. Belta, "Temporal logic planning and control of robotic swarms by hierarchical abstractions," *IEEE Transaction on Robotics*, vol. 23, no. 2, pp. 320–330, 2007.

[3] M. Kloetzer and C. Belta, "A fully automated framework for control of linear systems from temporal logic specifications," *IEEE Transactions on Automatic Control*, vol. 53, pp. 287–297, February 2008.

[4] Y. Yordanov, J. Tůmova, I. Černá, J. Barnat, and C. Belta, "Temporal logic control of discrete-time piecewise affine systems," *IEEE Transactions on Automatic Control*, vol. 57, no. 6, pp. 1491–1505, 2012.

[5] C. Belta, V. Isler, and G. J. Pappas, "Discrete abstractions for robot motion planning and control in polygonal environments," *IEEE Transactions on Robotics*, vol. 21, pp. 864 – 874, October 2005.

[6] G. E. Fainekos, H. Kress-Gazit, and G. J. Pappas, "Hybrid controllers for path planning: A temporal logic approach," in *Proceedings of the 44th IEEE Conference on Decision and Control*, (Seville, Spain), pp. 4885 – 4890, 12 – 15 Dec. 2005.

[7] S. R. Lindemann, I. I. Hussein, and S. M. LaValle, "Real time feedback control for non-holonomic mobile robots with obstacles," in *Proceedings of the 45th IEEE Conference on Decision and Control*, (San Diego, CA, USA), pp. 2406–2411, December 2006.

[8] T. Wongpiromsarn, *Formal Methods for Design and Verification of Embedded Control Systems: Application to an Autonomous Vehicle*. PhD thesis, California Institute of Technology, 2010.

[9] V. S. Alagar and K. Periasamy, *Specification of Software Systems*. London, UK: Springer-Verlag, 2nd ed., 2011.

[10] M. Mazo and P. Tabuada, "Symbolic approximate time-optimal control," *Systems & Control Letters*, vol. 60, pp. 256–263, 2011.

[11] G. Reissig, "Computing abstractions of nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 56, no. 11, pp. 2583–2598, 2011.

[12] S. Karaman, R. Sanfelice, and E. Frazzoli, "Optimal control of mixed logical dynamical systems with linear temporal logic specifications," in *Proceedings of the 47th IEEE Conference on Decision and Control, 2008.*, (Cancun, Mexico), pp. 2117–2122, Dec 2008.

[13] A. Bhatia, M. R. Maly, L. E. Kavraki, and M. Y. Vardi, "Motion planning with complex goals," *IEEE Robotics Automation Magazine*, vol. 18, pp. 55–64, 2011.

[14] P. Chaudhari, T. Wongpiromsarn, and E. Frazzoli, "Incremental minimum-violation control synthesis for robots interacting with external agents," in *Proceedings of the 2014 American Control Conference (ACC)*, (Portland, OR, USA), June 4–6 2014.

[15] A. Ulusoy, S. L. Smith, X. C. Ding, C. Belta, and D. Rus, "Optimality and robustness in multi-robot path planning with temporal logic constraints," *International Journal of Robotics Research*, vol. 32, no. 8, pp. 889–911, 2013.

[16] M. Guo and D. V. Dimarogonas, "Reconfiguration in motion-planning of single- and multi-agent systems under infeasible local LTL specifications.," in *Proceedings of the 52nd IEEE Conference on Decision & Control*, (Florence, Italy), December 2013.

[17] 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, 2015.

[18] R. V. Cowlagi and P. Tsiotras, "Shortest distance problems in graphs using history-dependent transition costs with application to kinodynamic path planning," in *Proceedings of the 2009 American Control Conference*, (St. Louis, MO, USA), pp. 414 – 419, 9 – 12 Jun 2009.

[19] R. V. Cowlagi and P. Tsiotras, "Hierarchical motion planning with dynamical feasibility guarantees for mobile robotic vehicles," *IEEE Transactions on Robotics*, vol. 28, no. 2, pp. 379 – 395, 2012.

[20] R. V. Cowlagi and D. N. Kordonowy, "Geometric abstractions of vehicle dynamical models for intelligent autonomous motion," in *Proceedings of the 2014 American Control Conference*, (Portland, OR.), pp. 4840–4845, June 4 – 6 2014.

[21] R. V. Cowlagi and P. Tsiotras, "Curvature-bounded traversability analysis for motion planning of mobile robots," *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 1011–1019, 2014.

[22] P. Wolper, M. Vardi, and A. Sistla, "Resoning about infinite computations," in *Proceedings of the 24th Symposium on Foundations of Computer Science*, (Tucson, AZ), pp. 185–194, 1983.

[23] P. Wolper, "Constructing automata from temporal logic formulas: A tutorial," in *Formal Methods Performance Analysis: First EEF/Euro Summer School on Trends in Computer Science*, New York, NY: Springer-Verlag, 2001.

[24] G. Holzmann, "The model checker SPIN," *IEEE Transactions on Software Engineering*, vol. 23, no. 5, pp. 279–295, 1997.

[25] P. Gastin and D. Oddoux, "Fast LTL to Büchi automata translation," in *Proceedings of the 13th Conference on Computer Aided Verification (CAV' 01)* (H. C. G. Berry and A. Finkel, eds.), vol. 2102 of *Lecture Notes in Computer Science*, (New York), pp. 53–65, Springer-Verlag, 2001.

[26] Z. Zhang and R. V. Cowlagi, "Incremental path repair in hierarchical motion-planning with dynamic feasibility guarantees for mobile robotic vehicles," in *European Control Conference ECC'15*, (Linz, Austria), July 15 – 17 2015. to appear.

APPENDIX I
LTL$_{-X}$ SEMANTICS

The LTL$_{-X}$ syntax involves operators $\neg$ (negation), $\vee$ (disjunction) and $\triangleright$ (*until*). Let $\Lambda = \{\lambda_k\}_{i=0}^{N^{\mathrm{R}}}$, with $N^{\mathrm{R}} \in \mathbb{Z}_{\geqslant 0}$ be a prespecified set of atomic propositions. A *LTL$_{-X}$ formula over* Λ is recursively defined as follows:

1) Every atomic proposition $\lambda_k \in \Lambda$ is a LTL$_{-X}$ formula.
2) If ϕ_1 and ϕ_2 are LTL$_{-X}$ formulae, then $\neg\phi_1$, $(\phi_1 \vee \phi_2)$, and $(\phi_1 \triangleright \phi_2)$ are also LTL$_{-X}$ formulae.

Let ϕ_1 and ϕ_2 be LTL$_{-X}$ formulae. The formula $(\phi_1 \triangleright \phi_2)$ means that ϕ_2 eventually becomes true and ϕ_1 remains true until ϕ_2 becomes true. The operators $\wedge$ (conjunction), $\Rightarrow$ (implication), $\Leftrightarrow$ (equivalence) are defined as is standard in propositional logic. The temporal operators $\Diamond$ (*eventually*), and $\Box$ (*always*) are defined as follows:

$$\Diamond\phi_1 := (\phi_1 \vee \neg\phi_1) \triangleright \phi_1, \qquad \Box\phi_1 := \neg(\Diamond\neg\phi_1).$$

A *word* $\omega = (w_0, w_1, \ldots, w_P)$ is a sequence such that $w_i \in 2^\Lambda$ for each $i = 0, \ldots, P$, where 2^Λ denotes the power set of Λ. For $i, j \in \mathbb{Z}_{\geqslant 0}$, $j \geqslant i$, we denote by ω_i^j the word $(w_i, w_{i+1}, \ldots, w_j)$. The *satisfaction* of a LTL$_{-X}$ formula ϕ by the word ω is denoted by $\omega \models \phi$, and it is recursively defined as follows:

1) $\omega \models \lambda_k$ if $\lambda_k \in w_0$, and $\omega \not\models \lambda_k$ if $\lambda_k \notin w_0$.
2) $\omega \models \neg\phi$ if $\omega \not\models \phi$.
3) $\omega \models (\phi_1 \vee \phi_2)$ if $\omega \models \phi_1$ or $\omega \models \phi_2$.
4) $\omega \models (\phi_1 \triangleright \phi_2)$ if there exists $i \geqslant 0$ such that $\omega_i^P \models \phi_2$ and for every $j < i$, $\omega_j^P \models \phi_1$.

Appendix II
Technical Proofs

Proof of Prop. 1. Define $\mathbf{i}_k := (v_k, \ldots, v_{k+H})$, for each $k \in \{0, \ldots, P-H\}$. First, suppose that $\mathcal{J}_H(\mathbf{v}) < \chi$. By (7) and (8) it follows that, for each $k \in \{0, \ldots, P-H\}$, $g_H(\mathbf{i}_k, \mathbf{i}_{k+1}) = 1$, and that $\mathcal{S}(\mathbf{i}_{k+1})$ is nonempty. By (6), for every state $\xi_\mathrm{s} \in \mathcal{S}(\mathbf{i}_{k+1})$, there exists $pre(\xi_\mathrm{s}) \in \hat{\mathcal{S}}(\mathbf{i}_k, \mathbf{i}_{k+1}) \subseteq \mathcal{S}(\mathbf{i}_k)$ such that $\xi_\mathrm{s} \in \mathcal{R}_{\mathbf{i}_k}(pre(\xi_\mathrm{s}))$.

In particular, $\mathcal{S}(\mathbf{i}_{P-H})$ is nonempty, and, by Eqn. (6), $\mathcal{Q}(\mathbf{i}_{P-H})$ is nonempty. By definition, for any state $\xi_{P-H} \in \mathcal{Q}(\mathbf{i}_{P-H})$, there exist $t_{P-H} \in \mathbb{R}_+$ and a control input $u_{P-H} \in \mathcal{U}_{t_{P-H}}$ such that

$$\mathsf{x}(\xi(t; \xi_{P-H}, u_{P-H})) \in \cup_{\ell=1}^{H+1} \mathsf{cell}([\mathbf{i}_{P-H}]_\ell) \qquad (11)$$

Then we iteratively define $\xi_k := pre(\xi_{k+1})$ for each $k = P-H, \ldots, 0$. Note that $\xi_{k+1} \in \mathcal{R}_{\mathbf{i}_k}(\xi_k)$. By definition in Eqn. (5), there exist $t_k \in \mathbb{R}_+$ and $u_k \in \mathcal{U}_{t_k}$ such that $\xi\left(t; \xi_k, u_k\right) = \xi_{k+1}$, and

$$\mathsf{x}(\xi\left(t; \xi_k, u_k\right)) \in \mathsf{cell}([\mathbf{i}_k]_2), \qquad t \in [0, t_k]. \qquad (12)$$

Note that $\xi_k \in \mathcal{S}(\mathbf{i}_k)$, which implies that $\xi_0 \in \mathcal{S}(\mathbf{i}_0) = \xi_0$. Now define the concatenated trajectory

$$\xi^*(t) := \xi(t; \xi_k, u_k), \qquad t \in [\tau_k, \tau_{k+1}], \qquad (13)$$

where $\tau_0 = 0$ and $\tau_{k+1} := \tau_k + t_k$, $k = 0, \ldots, P-H$. By (11) and (12), it follows that $\mathbf{v} = tr(\xi^*, \mathcal{G})$, and, by consequence, that $\mathbf{v} \in \mathcal{L}_\Gamma(\xi_0)$.

To prove the converse, suppose that $\mathbf{v} \in \mathcal{L}_\Gamma(\xi_0)$. By definition, $\mathsf{x}(\xi\left(0; \xi_0, u\right)) \in \mathsf{cell}(v_0)$ and $\mathsf{x}(\xi\left(t_\mathrm{f}; \xi_0, u\right)) \in \mathsf{cell}(v_P)$. Define for each $k \in \{0, \ldots, P-H\}$,

$$\tau_k := \max_{\tau_k \in [0, t_\mathrm{f}]} \left\{\tau_k : \mathsf{x}(\xi\left(\tau_k; \xi_0, u\right)) \in \mathsf{cell}([\mathbf{i}_k]_1) \cap \mathsf{cell}([\mathbf{i}_k]_2)\right\},$$
$$\xi_k^\mathrm{s} := \xi\left(\tau_k; \xi_0, u\right), \qquad u_k := u|_{[\tau_k, \tau_{k+1}]}.$$

The existence of τ_k is guaranteed by continuity of ξ and by (2). It is easy to see that $\tau_0 = 0$, $\xi_k^\mathrm{s} = \xi_0$, and that for each $k \in \{1, \ldots, P-H\}$, $\xi_{k+1}^\mathrm{s} \in \mathcal{R}_{\mathbf{i}_k}(\xi_k^\mathrm{s})$. By definition of the sets $\mathcal{Q}(\cdot)$, it is also clear that $\xi_{k+1}^\mathrm{s} \in \mathcal{Q}(\mathbf{i}_{k+1})$. Therefore, $\xi_{k+1}^\mathrm{s} \in \mathcal{S}(k+1)$, and by Eqns. (6) and (7), $g_H(\mathbf{i}_k) = 1$ for each $k \in \{1, \ldots, P-H\}$. It follows that $\mathcal{J}_H(\mathbf{v}) < \chi$. $\square$

Proof of Theorem 1. By Prop. 1,

$$\pi_k(\Theta_\mathrm{p}) \in \mathcal{L}_\Gamma(\xi_{0,k}), \qquad \pi_k(\Theta_\mathrm{s}) \in \mathcal{L}_\Gamma(\xi_{0,k}),$$

for each $k = 1, \ldots, N^\mathrm{V}$, which implies that $\pi_k(\Theta_\mathrm{p}, \Theta_\mathrm{s}, \Theta_\mathrm{s}, \ldots) \in \mathcal{L}_\Gamma(\xi_0)$. By (3), (9), and (10), the path $\pi_k(\Theta) \in \mathcal{L}_\mathcal{G}$ defines the word $\mathbf{w}(\pi_k) = (w_{0,k}, w_{1,k}, \ldots)$, and by the concatenation rule in (4), the overall word defined is $\mathbf{w}(\pi) := (w_{0,1}, w_{0,2}, \ldots, w_{1,1}, w_{1,2}, \ldots)$, By definition of $\mathcal{T}_{\phi,M}$, this word is accepted by the Büchi automaton $\mathcal{B}_\phi$, which in turn implies $(\pi_1(\Theta), \ldots, \pi_{N^\mathrm{V}}(\Theta)) \in \mathcal{L}_{\Gamma\Phi}$. $\square$