Motion-planning with Temporal Logic Specifications for a Nonholonomic Vehicle Kinematic Model

Raghvendra V. Cowlagi*† and Zetian Zhang*

***Abstract*— We present a new technique for the control of a nonholonomic vehicle kinematic model subject to linear temporal logic (LTL) specifications. The proposed technique is based on partitioning of the vehicle's planar workspace into cells. The main result of this paper is the precise characterization of acceptable sequences of cells, which can be traversed by admissible state trajectory of the vehicle model while satisfying the given LTL specifications. The proposed approach does not require complete controllability of the vehicle model in the presence of workspace constraints, and no linearization of the model is involved. The key technical innovation is the so-called *lifted graph*. Edge transitions in the lifted graph are associated with certain forward- and backward reachable sets of the vehicle model. We discuss numerical methods to compute the aforesaid forward- and backward reachable sets of the vehicle model, and illustrate the proposed technique with numerical simulation examples.**

I. Introduction

Unmanned aerial and terrestrial vehicles are foreseen to serve in many military and civilian applications in the near future with a high degree of autonomy enabled by motion-planning and control algorithms. Traditionally, such algorithms have focused on point-to-point obstacle-free path generation and tracking [1], or, in the absence of environmental obstacles, on trajectory optimization [2], [3]. Recent research has broadened the motion-planning problem to include constraints of satisfying a higher-level specification [4], as described next.

Consider a vehicle designed to move in a known environment – its *workspace* – containing obstacles. The vehicle is assigned a specification in terms of a formula of linear temporal logic (LTL). The motion of the vehicle is modeled by a controlled nonlinear dynamical system Γ, and a trajectory cost is defined by a functional over admissible control inputs. The *motion-planning problem* is to find admissible control inputs for this dynamical such that the resultant vehicle motion satisfies the aforesaid specification and also minimizes the trajectory cost.

The inclusion of task satisfaction constraints into the motion-planning problem necessitates hierarchical separation to the meet high-level specifications and the low-level control objectives. In turn, this separation necessitates the development of new techniques to supplement traditional geometric path-planning and trajectory optimization algorithms. In particular, techniques related to the generation of *discrete abstractions* of the dynamical system Γ have been proposed [5], [6]. Loosely speaking, these techniques involve the generation of a finite state transition system $\mathcal{G}$ such that the transition sequences of $\mathcal{G}$ are equivalent – via appropriately defined equivalence relations – to admissible state trajectories of Γ. Transition sequences of $\mathcal{G}$ can be chosen to satisfy temporal logic constraints. By construction of $\mathcal{G}$, the existence of admissible state trajectories of Γ that are equivalent to such transition sequences, and the existence of the associated admissible control inputs, is guaranteed. This guarantee is tantamount to "compatibility" between hierarchically separated algorithms for satisfying high-level specifications and low-level trajectory optimization.

Related Work: The idea of generating a discrete abstraction of a dynamical system Γ has been developed [5]–[10] for synthesizing control laws that produce state trajectories satisfying a given LTL specification. Specifically, a finite state transition system $\mathcal{G}$ is created such that it "resembles" the dynamical system in the sense of bisimulation, simulation, or language equivalence [6]. The states in $\mathcal{G}$ are associated with regions in the state space of Γ. Control laws are designed to steer trajectories of Γ between these regions and thereby emulate state transitions in $\mathcal{G}$. The LTL specification is represented by a Büchi automaton $\mathcal{B}$ [11], [12]. Finally, a product automaton of $\mathcal{B}$ and $\mathcal{G}$ is searched: any run of this product automaton can be projected to a path in $\mathcal{G}$, which in turn can be associated with control laws and admissible state trajectories of Γ. These state trajectories are guaranteed to satisfy the given LTL specification.

For continuous-time and discrete-time linear systems, a well-studied method for constructing the discrete abstraction $\mathcal{G}$ relies on partitioning the state space into polytopes [13]–[15]. Control laws are developed either to steer the state trajectories between adjacent polytopes or to maintain the state within a particular polytope.

A closely related approach, especially for applications involving motion-planning for mobile robotic vehicles [8], [9] relies on partitioning the *output* space of Γ into non-overlapping convex regions called *cells*. A graph $\mathcal{G}$ is defined such that each vertex of $\mathcal{G}$ uniquely corresponds to a cell, and edges in $\mathcal{G}$ are defined according to geometric adjacency of cells. Control laws are then determined to steer the *output* trajectories between adjacent cells, and a discrete abstraction of Γ is thereby constructed (see, for instance, [8] for further details). However, this approach is not suitable for creating discrete abstractions of nonholonomic vehicle dynamical models. Consider, for example, the Dubins car, which models a forward-moving vehicle with a minimum turn radius. In this case, transitions to adjacent cells depend on the vehicle's initial state: it is *in principle* neither possible to construct

*Worcester Polytechnic Institute, Worcester, MA, USA.
† Corresponding author, rvcowlagi@wpi.edu.

control laws that steer the vehicle between adjacent cells from *every initial state*, nor can the absence of existence of such control laws be established.

None of the results in the existing literature can be applied to solve the problem of motion-planning for a nonholonomic vehicle subject to temporal logic specifications. Discrete abstractions of nonlinear dynamical systems (including those with nonholonomic constraints) based on state-space discretization has been recently proposed [16], [17], but this approach is computationally impractical[1].

In this paper, we propose a computationally efficient approach to motion-planning subject to temporal logic specifications for a nonholonomic vehicle model. The proposed approach is based on workspace cell decomposition. The key innovation is the so-called *lifted graph* (Section III), which is the discrete mathematical structure used for creating a discrete abstraction of the vehicle model. Briefly, edges in the lifted graph are associated with successions of adjacent cells in the cell decomposition. The nonholonomic constraint in the vehicle model imposes an upper bound on the curvature of admissible *workspace* trajectories of the vehicle. We analyze the traversability of successions of cells by curvature-bounded workspace trajectories, which in turn can be associated with some reachability properties of the vehicle model. We show that for every edge of the lifted graph, there exist admissible control inputs that enable the vehicle to "traverse" this edge. Finally, we construct a finite state automaton as the product of the lifted graph with the Büchi automaton associated with the given LTL specification. Every feasible run of this product automaton has a unique projection on the collection of all paths of the lifted graph.

The contributions of this paper are as follows. Firstly, we provide a new method to find state trajectories satisfying LTL specifications for a nonholonomic vehicle model. The existing approaches in this area either involve linearization (followed by discrete abstraction of the linearized model) or lead to abstractions with extremely large numbers of discrete states. The proposed approach does not involve linearization.

Secondly, we provide a new method to develop discrete abstractions of a nonlinear system based on *output space decompositions* instead of state space decompositions, as is often done in the literature. This approach is valuable in general, for nonlinear systems beyond the model considered in this paper, because the output space is typically of a smaller dimension than the state space. For differentially flat systems [18], [19], there exist *algebraic* mappings between the output space trajectories and state space trajectories and control inputs. Previous attempts at using output space decompositions [8] have either ignored control input constraints or have imposed strong controllability assumptions [8], [20].

Finally, and in contrast to the existing literature [7], [8], [14], [20], the proposed discrete abstraction does not rely on feedback control. Stated differently, the proposed approach does not use up control authority for the sake of creating a discrete abstraction of the vehicle model. Therefore, optimal control laws can be designed for executing the vehicle's motion using the complete admissible control value set.

[1]In the example illustrated [17], for example, a three state model results in a discrete abstraction with 91,035 states and over 34 *million* transitions.

The rest of this paper is organized as follows. In Section II, we precisely formulate the problem. In Section III, we discuss the lifted graph and a discrete abstraction of the vehicle model. Edge transitions in the lifted graph are discussed in context with the reachability properties of the vehicle model. In Section IV, we discuss a product transition system that enables the search for admissible trajectories satisfying the given LTL specification. In Section V, we present illustrative examples of implementation of the proposed approach, and conclude the paper in Section VI.

II. Problem Formulation

In this section, we introduce the following elements of the problem: the vehicle model, the workspace cell decomposition, and the LTL specification.

Vehicle model: Let $\xi = (x, y, \psi) \in \mathcal{D} := \mathbb{R}^2 \times \mathbb{S}^1$ denote the state of the vehicle, namely, the position of the vehicle center of mass and the direction of its velocity vector in a prespecified Cartesian coordinate system. For convenience, we denote by $\mathsf{x}(\xi)$ the position coordinates of any state $\xi = (x, y, \psi) \in \mathcal{D}$, i.e., $\mathsf{x}(\xi) = (x, y)$. We consider a vehicle kinematic model described by the differential equations

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

where u is the control input. We assume that the set of admissible control input values is the closed interval $U := \left[-\frac{1}{\rho}, \frac{1}{\rho}\right]$, where $\rho > 0$. For every $t_{\mathrm{f}} \in \mathbb{R}_+$, let

$$\mathcal{U}_{t_{\mathrm{f}}} := \{u : [0, t_{\mathrm{f}}] \to U \mid u \text{ piecewise continuous}\},$$

be the set of all admissible control inputs. For any $u \in \mathcal{U}_{t_{\mathrm{f}}}$, and initial state $\xi_0 \in \mathcal{D}$, the state trajectory $\xi(t; \xi_0, u)$, $t \in [0, t_{\mathrm{f}}]$, obtained by integrating Eqn. (1) is called an *admissible state trajectory*.

Workspace cell decomposition: Let $\mathcal{W} \in \mathbb{R}^2$ denote a planar region with obstacles. 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 $i = 1, \ldots, N^{\mathrm{C}}$. Therefore, $\cup_{i=1}^{N^{\mathrm{C}}} R^i = \mathcal{W}$. We associate with this partition a graph $\mathcal{G} := (V, E)$ such that each vertex of $\mathcal{G}$ is uniquely associated with an obstacle-free cell, and each edge of $\mathcal{G}$ is uniquely associated with a pair of geometrically adjacent cells. We denote by $\mathsf{cell}(j)$ the element of $\{R^i\}_{i=1}^{N^{\mathrm{C}}}$ associated with the vertex $j \in V$. A *path* π in $\mathcal{G}$ is a finite or infinite sequence $(j_0, j_1, \ldots)$ of vertices, such that $j_0, j_k \in V$, and $(j_{k-1}, j_k) \in E$, 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 $\mathcal{G}$.

For every $t_{\mathrm{f}} \in \mathbb{R}_+$ and $u \in \mathcal{U}_{t_{\mathrm{f}}}$, we define the *$\mathcal{G}$-trace* of the trajectory $\xi(\cdot; \xi_0, u)$ as the path $tr(\xi, \mathcal{G}) = (j_0, j_1, \ldots) \in$

$\mathcal{L}_{\mathcal{G}}$ of minimal length such that $\mathsf{x}(\xi\,(0;\xi_0,u)) \in \mathsf{cell}(j_0)$, and

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

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 [14], 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 simplicity of exposition of the proposed work. A brief overview of LTL$_{-X}$ is provided in Appendix I; the reader is referred to [14], [21] for further details.

We assume a finite number of regions of interest in the workspace $\mathcal{W}$, and we label these regions by $\lambda_1,\ldots,\lambda_N^{\mathrm{R}}$. Each symbol λ_k is an atomic proposition defined by a set membership relation in $\mathcal{D}$ of the form

$$\lambda_k \equiv \quad \mathsf{x}(\xi) \in \cup_{i\in\varsigma_k} R^i, \quad \text{for each } k = 1,\ldots,N^{\mathrm{R}}. \quad (3)$$

where $\varsigma_k \subseteq \{1,\ldots,N^{\mathrm{C}}\}$ is prespecified for each $k = 1,\ldots,N^{\mathrm{R}}$. Each region of interest is assumed to be a (possibly disconnected) union of cells. Each path $\pi = (j_0,j_1,\ldots) \in \mathcal{L}_{\mathcal{G}}$ defines a word $\omega(\pi) = (w_0,w_1,\ldots,)$, where

$$w_\ell := \left\{\lambda_k \mid \mathsf{cell}(j_\ell) \subseteq \cup_{i\in\varsigma_k} R^i\right\}. \quad (4)$$

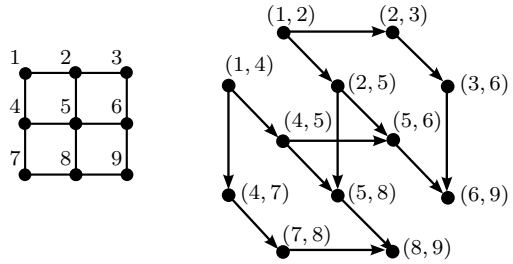
The path π is said to satisfy a LTL$_{-X}$ formula ϕ if the word $\omega(\pi)$ satisfies ϕ (as defined in Appendix I).

The main problem of interest in this paper is the following.

Problem 1. *Given a LTL$_{-X}$ formula ϕ over Λ, and $\xi_0 \in \mathcal{D}$, determine a collection of paths $\mathcal{L}_{\Gamma\phi} \subseteq \mathcal{L}_{\Gamma}(\xi_0)$ such that every path in $\mathcal{L}_{\Gamma\phi}$ satisfies the formula ϕ.*

The "channel" of cells associated with every path in the collection $\mathcal{L}_{\Gamma\phi}$ can be traversed by an admissible state trajectory. Furthermore, every path in $\mathcal{L}_{\Gamma\phi}$ also satisfies the specification ϕ. The collection $\mathcal{L}_{\Gamma\phi}$ therefore represents, loosely speaking, an equivalence class of admissible state trajectories that satisfy the specification ϕ. In the context of the "compatibility" in hierarchical motion-planning discussed in Section I, the computation of $\mathcal{L}_{\Gamma\phi}$ is desirable because every high-level plan that belongs to $\mathcal{L}_{\Gamma\phi}$ is guaranteed to be "compatible" with vehicle dynamical constraints.

The computation of $\mathcal{L}_{\Gamma\phi}$ is challenging because $\mathcal{L}_{\Gamma}(\xi_0)$ is difficult to compute, and involves discrete abstraction of the continuous system Γ. In [8], for instance, feedback control laws are designed to construct a language equivalent discrete abstraction, such that $\mathcal{L}_{\Gamma} = \mathcal{L}_{\mathcal{G}}$. However, the underlying assumption therein is that the vehicle model is completely controllable in the presence of obstacles (workspace constraints). For the vehicle model considered in this paper, this controllability assumption is not true [22].



(a) Graph $\mathcal{G}$. (b) Lifted graph $\mathcal{G}_1$.

Fig. 1. Example of a lifted graph (all vertices of $\mathcal{G}_1$ are not shown).

To solve Problem 1 in the light of the preceding observations, we propose a new approach based on the so-called lifted graph, which we discuss next.

III. Lifted Graph

The proposed approach to the solution of Problem 1 relies on traversability analysis and assignment of transition costs to successions of edges in the graph $\mathcal{G}$. This approach appears in [22] for motion-planning subject to vehicle kinematic and/or dynamic constraints. To fix ideas, for every integer $H \geqslant 0$, let

$$V_H := \{(j_0,\ldots,j_H) : (j_{k-1},j_k) \in E,\ k = 1,\ldots,H,\\ j_k \neq j_m,\ \text{ for } k,m \in \{0,\ldots,H\},\ \text{ with } k \neq m\}.$$

Every element I of the set V_H is an ordered $(H+1)$-tuple of the elements of V, and this tuple corresponds to a sequence of cells (see Fig. 2). We denote by $[I]_k$ the k^{th} element of I, and by $[I]_k^m$ the tuple $\left([I]_k,[I]_{k+1},\ldots,[I]_m\right)$, for $k < m \leqslant H+1$. Let E_H be a set of all pairs (I,J), with $I,J \in V_H$, such that $[I]_k = [J]_{k-1}$, for every $k = 2,\ldots,H+1$, and $[I]_1 \neq [J]_{H+1}$. An element of the set E_H is called an *H-history*. Each H-history is an ordered $(H+2)$-tuple of elements of V. According to this notation, $V_0 = V$ and $E_0 = E$. Note that the largest integer $\bar{H}$ for which these definitions remain meaningful is the diameter of the graph $\mathcal{G}$, i.e. $\bar{H} = \mathrm{diam}(\mathcal{G})$.

The *lifted graph* $\mathcal{G}_H$ is defined as the directed graph whose vertex and edge sets are, respectively, V_H and E_H. Figure 1 illustrates an example of a lifted graph for $H = 1$. Every path $\pi = (j_0,j_1,\ldots)$ in the graph $\mathcal{G}$ can be uniquely mapped to a path $\pi_H = (J_0,J_1,\ldots)$ in the lifted graph $\mathcal{G}_H$, where $J_k = (j_k,j_{k+1},\ldots,j_{k+H}) \in V_H$ for each $k \in \mathbb{N}$. We denote this map by $b_0^H : \mathcal{L}_{\mathcal{G}} \to \mathcal{L}_{\mathcal{G}_H}$, where $\mathcal{L}_{\mathcal{G}_H}$ is the collection of all paths in $\mathcal{G}_H$. Therefore, $\pi_H = b_0^H(\pi)$ and $\pi = b_H^0(\pi_H)$. For every integer $H \geqslant 0$, the map b_0^H is bijective and b_H^H is the identity map. For integers $H,L \geqslant 0$, we define $b_H^L := b_0^L \circ b_H^0$. In general, $b_L^M \circ b_H^L \equiv b_H^M$ for integers $H,L,M \geqslant 0$.

A. Edge Transition Costs

We first characterize the collection $\mathcal{L}_{\Gamma}$. To this end, we first define a transition cost function $g_H : E_H \to [0,\chi]$, by which we can compute costs of paths in the lifted graph $\mathcal{G}_H$. Here, $\chi \in \mathbb{Z}$ is sufficiently large with $\chi \gg \mathrm{diam}(\mathcal{G})$. Then we assert that a path $\pi \in \mathcal{L}_{\mathcal{G}}$ belongs to the collection $\mathcal{L}_{\Gamma}(\xi_0)$

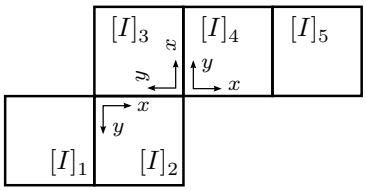


Fig. 2. Illustration of a sequence of cells associated with $I \in V_4$. In this case, the underlying cell decomposition consists of square cells.

if and only if any subpath of $b_0^H(\pi)$ of length less than χ has cost less than χ. To this end, we define the function g_H and then prove this assertion next.

Consider an element $(I, J) \in E_H$, with $I, J \in V_H$. Let $\mathcal{S}(I) \subset \mathcal{D}$ be a set of states associated with the element I in V_H such that $\mathsf{x}(\mathcal{S}(I)) \subseteq \mathsf{cell}([I]_1) \cap \mathsf{cell}([I]_2)$. Thus, the position components of the elements in $\mathcal{S}(I)$ lie on the boundary between the first and second cells corresponding to the vertices of V that constitute the ordered H-tuple I. Next let $\mathcal{Q}(J) \subset \mathcal{D}$ be a set of states such that $\mathsf{x}(\mathcal{Q}(J)) \subseteq \mathsf{cell}([J]_1) \cap \mathsf{cell}([J]_2)$ and for every state $\xi_{\mathrm{q}} \in \mathcal{Q}(J)$ there exists a traversal time t_{q} and an admissible control input $u_{\mathrm{q}} \in \mathcal{U}_{t_{\mathrm{q}}}$ such that $\mathsf{x}(\xi(t; \xi_{\mathrm{q}}, u_{\mathrm{q}})) \in \bigcup_{k=1}^{H+1} \mathsf{cell}([J]_k)$, for all $t \in [0, t_{\mathrm{q}}]$. Informally, $\mathcal{Q}(J)$ is the set of all states whose position components lie on the boundary between the first and second cells of J, and such that the traversal of the geometric region defined by the cells associated with the tuple J is possible from any initial state within $\mathcal{Q}(J)$. Sets such as $\mathcal{Q}(\cdot)$ are called *backward reachability sets* [23].

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

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

where $\xi_{\mathrm{s}} \in \mathcal{S}(I)$ and $2^{\mathcal{D}}$ is the collection of all subsets of $\mathcal{D}$. Informally, $\mathcal{R}_I(\xi_{\mathrm{s}})$ 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 I. Finally, for $(I, J) \in E_H$, define $\hat{\mathcal{S}}(I, J) := \{\xi_{\mathrm{s}} \in \mathcal{S}(I) : \mathcal{R}_I(\xi_{\mathrm{s}}) \cap \mathcal{Q}(J) \neq \varnothing\}$.

Now consider a path $\pi = \{j_0, j_1, \ldots\} \in \mathcal{L}_{\mathcal{G}}$. For each $k \in \mathbb{N}$, let $I_k := (j_k, \ldots, j_{k+H})$. Clearly, $(I_k, I_{k+1}) \in E_H$. We iteratively define g_H and the set association $\mathcal{S}(\cdot)$ as follows:

$$\mathcal{S}(I_{k+1}) := \bigcup_{\xi_{\mathrm{s}} \in \hat{\mathcal{S}}(I_k, I_{k+1})} \left(\mathcal{R}_{I_k}(\xi_{\mathrm{s}}) \cap \mathcal{Q}(I_{k+1})\right), \quad (6)$$

$$g_H(I_k, I_{k+1}) := \begin{cases} \chi, & \text{if } \mathcal{S}(I_{k+1}) = \varnothing, \\ 1, & \text{otherwise.} \end{cases} \quad (7)$$

The transition cost of any edge $(I_k, I_{k+1}) \in E_H$ is $g_H(I_k, I_{k+1})$. We define the *H-cost* of the path $\pi \in \mathcal{L}_{\mathcal{G}}$ as

$$\mathcal{J}_H(\pi) := H + \sum_{k=0}^{P-H} g_H(I_k, I_{k+1}). \quad (8)$$

Proposition 1. *Let $\pi = (j_0, \ldots, j_P)$ be a path with length less than χ in $\mathcal{L}_{\mathcal{G}}$, and let $\xi_0 \in \mathcal{D}$ be prespecified such that $\mathsf{x}(\xi_0) \in \mathsf{cell}(j_0) \cap \mathsf{cell}(j_1)$. Then $\pi \in \mathcal{L}_\Gamma(\xi_0)$ if and only if $\mathcal{J}_H(\pi) < \chi$.*

Proof: Define $I_k := (j_k, \ldots, j_{k+H})$, for each $k \in \{0, \ldots, P-H\}$. First, suppose that $\mathcal{J}_H(\pi) < \chi$. By Eqns. (7) and (8), it follows that, for each $k \in \{0, \ldots, P-H\}$, $g_H(I_k, I_{k+1}) = 1$, and that $\mathcal{S}(I_{k+1})$ is nonempty. By Eqn. (6), for every state $\xi_{\mathrm{s}} \in \mathcal{S}(I_{k+1})$, there exists $pre(\xi_{\mathrm{s}}) \in \hat{\mathcal{S}}(I_k, I_{k+1}) \subseteq \mathcal{S}(I_k)$ such that $\xi_{\mathrm{s}} \in \mathcal{R}_{I_k}(pre(\xi_{\mathrm{s}}))$.

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

$$\mathsf{x}(\xi(t; \xi_{P-H}, u_{P-H})) \in \cup_{\ell=1}^{H+1} \mathsf{cell}([I_{P-H}]_\ell) \quad (9)$$

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

$$\mathsf{x}(\xi(t; \xi_k, u_k)) \in \mathsf{cell}([I_k]_2), \qquad t \in [0, t_k]. \quad (10)$$

Note that $\xi_k \in \mathcal{S}(I_k)$, which implies that $\xi_0 \in \mathcal{S}(I_0) = \xi_0$. Now define the concatenated trajectory

$$\xi^*(t) := \xi(t; \xi_k, u_k), \qquad t \in [\tau_k, \tau_{k+1}], \quad (11)$$

where $\tau_0 = 0$ and $\tau_{k+1} := \tau_k + t_k$, for each $k = 0, \ldots, P-H$. By Eqns. (9) and (10), it follows that $\pi = tr(\xi^*, \mathcal{G})$, and, by consequence, that $\pi \in \mathcal{L}_\Gamma(\xi_0)$.

To prove the converse, suppose that $\pi \in \mathcal{L}_\Gamma(\xi_0)$. By definition, $\mathsf{x}(\xi(0; \xi_0, u)) \in \mathsf{cell}(j_0)$ and $\mathsf{x}(\xi(t_{\mathrm{f}}; \xi_0, u)) \in \mathsf{cell}(j_P)$. Define for each $k \in \{0, \ldots, P-H\}$,

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

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 $k \in \{1, \ldots, P-H\}$, $\xi_{k+1}^{\mathrm{s}} \in \mathcal{R}_{I_k}(\xi_k^{\mathrm{s}})$. By definition of the sets $\mathcal{Q}(\cdot)$, it is also clear that $\xi_{k+1}^{\mathrm{s}} \in \mathcal{Q}(I_{k+1})$. Therefore, $\xi_{k+1}^{\mathrm{s}} \in \mathcal{S}(k+1)$, and by Eqns. (6) and (7), $g_H(I_k) = 1$ for each $k \in \{1, \ldots, P-H\}$. It follows that $\mathcal{J}_H(\pi) < \chi$.

Proposition 1 asserts that the channel of cells associated with any path in $\mathcal{L}_{\mathcal{G}}$ with length less than χ is traversable by an admissible state trajectory of Γ if and only if the H-cost of this path is less than χ. Equivalently, consider the graph $\tilde{\mathcal{G}}_H$ obtained by deleting from the lifted graph $\mathcal{G}_H$ all edges with transition costs greater than or equal to χ, per Eqns. (6) and (7). Proposition 1 asserts that the graph $\tilde{\mathcal{G}}_H$ is a discrete abstraction of the system Γ.

IV. Product Transition System

A Büchi automaton is a finite state machine that accepts infinite input words. Loosely speaking, the acceptance condition of a Büchi automaton involves a set of accepting states that must be visited infinitely often during any run. It is known [24], [25] 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. Algorithms for translating a LTL formula to the associated Büchi automaton are available [11], [12], [25]. The reader interested is referred to [21], [26] for further details on finite state automata in general and Büchi automata in particular.

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. For an integer $H \geqslant 1$, we now define a *product transition system* $\mathcal{T}_{\phi,H} := (T, \delta_{\mathcal{T}_{\phi,H}})$ as follows:

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

$$(\theta_k|_S, w_k, \theta_\ell|_S) \in \delta_{\mathcal{B}_\phi}, (\theta_k|_{V_H}, \theta_\ell|_{V_H}) \in E_H, \quad (12)$$

$$w_k = \big\{\lambda_i \mid \mathsf{cell}([\theta_k|_{V_H}]_l) \subseteq \cup_{j \in \varsigma_i} R_j, \text{ for any } \ell = 1, \ldots, H+1\big\} \quad (13)$$

A *run* of $\mathcal{T}_{\phi,H}$ is a sequence $\Theta = (\theta_0, \theta_1, \ldots,)$ such that $\theta_k \in T$ for each $k \in \mathbb{N}$, and $(\theta_k, w_k, \theta_{k+1}) \in \delta_{\mathcal{T}_{\phi,H}}$, with w_k as defined in (13). We denote by $\Theta|_S = (\theta_0|_S, \theta_1|_S, \ldots,)$ and $\Theta|_{V_H} = (\theta_0|_{V_H}, \theta_1|_{V_H}, \ldots,)$, respectively, the projections of Θ on S and V_H. Note that $\Theta|_{V_H} \in \mathcal{L}_{\mathcal{G}_H}$, and therefore $b^0_H(\Theta|_{V_H}) \in \mathcal{L}_\mathcal{G}$.

Similar to the approach taken in [14], we restrict attention to runs of $\mathcal{T}_{\phi,H}$ 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 $\theta_\mathrm{f}, \theta_\mathrm{f} \in S_\mathrm{f} \times V_H$. The "prefix" run $\Theta_\mathrm{p} = (\theta_0, \ldots, \theta_M)$ is a finite sequence such that $\theta_0 \in S_0 \times V_H$ and $(\theta_M, w_M, \theta_\mathrm{f}) \in \delta_{\mathcal{T}_{\phi,H}}$.

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

Theorem 1. *Let $\Theta = (\theta_0, \theta_1, \ldots,)$ be a run of $\mathcal{T}_{\phi,H}$. If $\mathcal{J}_H(b^0_H(\Theta_\mathrm{p}|_{V_H})) < \chi$ and $\mathcal{J}_H(b^0_H(\Theta_\mathrm{s}|_{V_H})) < \chi$, then*

$$b^0_H(\Theta|_{V_H}) \in \mathcal{L}_{\Gamma\Phi}.$$

Conversely, for every path $\pi \in \mathcal{L}_{\Gamma\Phi}$, there exists a run Θ of $\mathcal{T}_{\phi,H}$, such that $b^0_H(\Theta|_{V_H}) = \pi$.

Proof: By Prop. 1,

$$b^0_H(\Theta_\mathrm{p}|_{V_H}) \in \mathcal{L}_\Gamma(\xi_0), \qquad b^0_H(\Theta_\mathrm{s}|_{V_H}) \in \mathcal{L}_\Gamma(\xi_0),$$

which implies that $b^0_H((\Theta_\mathrm{p}, \Theta_\mathrm{s}, \Theta_\mathrm{s}, \ldots)|_{V_H}) \in \mathcal{L}_\Gamma(\xi_0)$. By (4), (12), and (13), the path $\pi := b^0_H(\Theta|_{V_H}) \in \mathcal{L}_\mathcal{G}$ defines the word $\omega(\pi) = (w_0, w_1, \ldots)$. By definition of $\mathcal{T}_{\phi,H}$, $(\theta_k, w_k, \theta_{k+1}) \in \delta_{\mathcal{T}_{\phi,H}}$, and $(\theta_k|_S, w_k, \theta_{k+1}|_S) \in \delta_{\mathcal{B}_\phi}$ for each $k \in \mathbb{N}$. Therefore, the word $\omega(\pi)$ is accepted by the Büchi automaton $\mathcal{B}_\phi$, which in turn means that the path π satisfies the formula ϕ and, by consequence, $\pi = b^0_H(\Theta_\mathrm{p}|_{V_H}) \in \mathcal{L}_{\Gamma\Phi}$.

To prove the converse, consider a path $\pi = (j_0, j_1, \ldots) \in \mathcal{L}_{\Gamma\phi} \subseteq \mathcal{L}_\Gamma \subseteq \mathcal{L}_\mathcal{G}$. Because the path satisfies the formula ϕ, the word $\omega(\pi) = (w_0, w_1, \ldots)$, where w_k is defined in Eqn. (4), is accepted by the Büchi automaton $\mathcal{B}_\phi$. Let $s_0, s_1, \ldots$ be the states of $\mathcal{B}_\phi$ visited in the run associated with the input word $\omega(\pi)$. Also, let $I_k := (j_k, \ldots, j_{k+1}) \in V_H$, and define $\theta_k := (s_k, I_k)$ for each $k \in \mathbb{N}$. Clearly, $(\theta_k, w_k, \theta_{k+1}) \in \delta_{\mathcal{T}_{\phi,H}}$, and it follows that $\Theta := (\theta_0, \theta_1, \ldots)$ is a run of $\mathcal{T}_{\phi,H}$, and that $\Theta|_{V_H} = b^H_0(\pi)$, which implies that $b^0_H(\Theta|_{V_H}) = \pi$.

A. Motion-planning Algorithm

Theorem 1 solves Problem 1 in that it precisely characterizes the collection $\mathcal{L}_{\Gamma\phi}$. Recall that $\mathcal{L}_{\Gamma\phi}$ is an equivalence class of admissible state trajectories that satisfy the specification ϕ. Following Theorem 1, it is easy to determine a specific plan for a given initial state $\xi_0 \in \mathcal{D}$ and a given LTL$_{-X}$ specification ϕ. To do so, we execute Dijkstra's algorithm to search for a prefix and suffix run of the product transition system that satisfy the conditions given in Theorem 1. A straightforward algorithm for finding a run of a product automaton with minimum total number of vertices in the concatenation of prefix and suffix runs appears in [14], which we can appropriately modify for finding a run of $\mathcal{T}_{\phi,H}$.

The proposed approach results in a path in the workspace cell decomposition, which may be considered a (discrete) "route" for the vehicle. This path defines a channel through the workspace. The existence of an admissible control input for the vehicle model Γ is guaranteed; the actual computation of such an input can be performed by a trajectory optimization algorithm (e.g. [3]) which we do not consider in this paper.

To search the product automaton, it is necessary to compute the sets $\mathcal{R}(\cdot)$ and $\mathcal{S}(\cdot)$ discussed in Section III. The forward reachability sets $\mathcal{R}(\cdot)$ are relatively straightforward to compute using, for example, the analysis described in [27]. The main difficulty lies in the computation of the sets and $\mathcal{S}(\cdot)$ In the next section, we discuss the computation of these sets for the specific case of rectangular workspace cell decompositions.

V. Numerical Implementation and Discussion

For the vehicle model discussed in Section II, the workspace trajectories traced by admissible state trajectories are continuously differentiable planar curves with curvature at most $\frac{1}{\rho}$. In this case, the sets $\mathcal{S}(\cdot)$ can be numerically computed using planar geometric arguments, which are discussed in detail in [28]. We present the relevant ideas and terminology from [28] here for the sake of completeness.

A *path between points* P and Q in the plane is a differentiable curve $\gamma := \{s \mapsto (x(s), y(s)) \in \mathbb{R}^2 : 0 \leqslant s \leqslant 1\}$ such that $P = (x(0), y(0))$ and $Q = (x(1), y(1))$. We denote by $\gamma'(W)$ the angle of the tangent to γ at $W = (x(s), y(s))$, for all $s \in [0, 1]$, and by (W, α) the state in $\mathcal{D}$ specified by $W \in \mathbb{R}^2$ and $\alpha \in \mathbb{S}^1$.

Note from Eqn. (6) that the computation of the sets $\mathcal{S}(\cdot)$ requires the computation of the backward reachability sets $\mathcal{Q}(\cdot)$. When rectangular cell decompositions are used, every vertex $J \in V_H$ is associated with a "channel" of

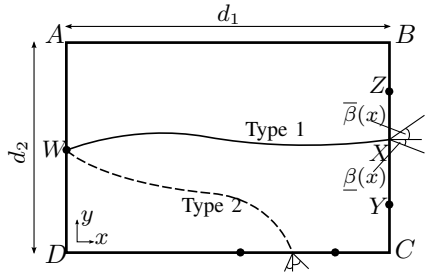


Fig. 3. Illustration of Type 1 and Type 2 paths.

rectangular cells, as shown in Fig. 2. To compute $\mathcal{Q}(J)$, we must analyze the traversal of this channel with curvature-bounded continuously differentiable curves, which we refer to as *curvature-bounded traversability analysis* (CBTA).

Definition 1. Associated with the vertex $J \in V_H$ is a *rectangular channel* R_J^{H+1}, which is a sequence of rectangular cells $\{R_n\}_{n=1}^{H+1}$, with disjoint interiors, such that

1) Exactly one edge of R_n has a non-empty intersection with exactly one edge of R_{n+1} for each $n \in \{1, \ldots, H\}$,
2) For all $m, n \in \{1, \ldots, H+1\}$, the edges of R_n and R_m do not intersect whenever $m \notin \{n-1, n, n+1\}$.

An example of such a rectangular channel is shown in Fig. 2. In this context, consider the following CBTA problem.

Problem 2 (CBTA-R). *Let W be a point on the segment $R_1 \cap R_2$. Let $\alpha \in \mathbb{S}^1$ and $\rho \in \mathbb{R}_{>0}$ be prespecified. Determine if there exists a path $\gamma_{W,\alpha}$ such that:*

1) $\gamma_{W,\alpha}(0) = W$ and $\gamma'_{W,\alpha}(W) = \alpha$,
2) The point $X := \gamma_{W,\alpha}(1)$ lies on the segment $R_H \cap R_{H+1}$,
3) The path $\gamma_{W,\alpha}$ does not leave R_J^{H+1}, i.e. $(x(s), y(s)) \in \cup_{n=1}^{H+1} R_n$ for every $s \in [0,1]$,
4) The curvature of γ at any point is at most ρ^{-1}.

Clearly, the set $\mathcal{Q}(J)$ for any $J \in V_H$ is the collection of all states (W, α) such that there exists a path $\gamma_{W,\alpha}$ as previously described. Therefore, the solution of CBTA-R is an important step in the computation of the set $\mathcal{Q}(J)$. A numerical algorithm to solve CBTA-R problem can be formulated using recursive CBTA of the individual rectangles within the rectangular channel. To this end, consider a rectangle $ABCD$ with a Cartesian coordinate axes system as shown in Fig. 3. Let the dimensions of the rectangle be d_1 and d_2, and let $\rho \in \mathbb{R}_{>0}$.

Definition 2. Let $\underline{\beta}(x), \overline{\beta}(x)$, $x \in [0, d_2]$ be functions such that $-\frac{\pi}{2} \leqslant \underline{\beta}(x) \leqslant \overline{\beta}(x) \leqslant \frac{\pi}{2}$. Let $Y = (d_1, y), Z = (d_1, z)$ be points on the segment BC with $y \leqslant z$. A path γ is called a *Type 1 path* if

1) The endpoints of γ lie on the DA and YZ,
2) The curvature at any point on γ is at most r^{-1},
3) All points of γ lie in the closed interior of $ABCD$, and
4) $\gamma'(X) \in \left[\underline{\beta}(x), \overline{\beta}(x)\right]$.

A Type 2 path is defined analogously for traversal across adjacent edges (see Fig. 3 for illustrations of such paths).

CBTA-R can be solved by recursively solving the following two problems defined on a single rectangle.

Problem 3 (CBTA-S1, resp. CBTA-S2). *Find $\underline{\alpha}, \overline{\alpha}$ such that for every $\alpha \in [\underline{\alpha}, \overline{\alpha}]$, there exists a Type 1 path (respectively, Type 2 path), γ with $\gamma(0) = W$ and $\gamma'(W) = \alpha$.*

We attach a coordinate axes system to each rectangle of R^C consistent with the axes system used to define Type 1 and Type 2 paths (see Figs. 2 and 3). For $n = 1, \ldots, H+1$, we denote by $d_{n,1}$ and $d_{n,2}$ the dimensions of each rectangle along the x and y axes. We can identify rigid geometric transformations (i.e. a sequence of rotations and reflections) that align the entry and exit segments of rectangle R_n to the segments AD and BC, respectively, for traversal across parallel edges, or to the segments AD and DC, respectively, for traversal across adjacent edges. Let ϱ_n denote the minimum number of reflections involved in the transformation associated with the rectangle R_n. For $n = 2, 3, \ldots, H$, we denote by U_n and V_n (respectively, Y_n and Z_n), the endpoints of the segments $R_{n-1} \cap R_n$ (respectively, $R_n \cap R_{n+1}$). We denote the coordinates of the points U_n, V_n, Y_n, Z_n, by the corresponding lower case letters, i.e., $V_n = (0, v_n)$, etc.

For every point $X_n = (x_n, 0)$ (or $X_n = (d_{n,1}, x_n)$, as applicable) on the segment $Y_n Z_n$, we denote by $\underline{\beta}_n(x_n)$ and $\overline{\beta}_n(x_n)$, respectively, the lower and upper bounds of the allowable terminal tangent angles. Similarly, for every point $W_n = (0, w_n)$ on the segment $U_n V_n$, we denote by $\underline{\alpha}_n(w_n)$ and $\overline{\alpha}_n(w_n)$, respectively, the results of the solution of CBTA-S1 (or CBTA-S2, as applicable). This means that for every $\alpha \in [\underline{\alpha}_n(w_n), \overline{\alpha}_n(w_n)]$, there exists a curve γ such that $\gamma(0) = W_n$, $X_n := \gamma(1) \in Y_n Z_n$, $\gamma'(W_n) = \alpha$, and $\gamma'(X_n) \in \left[\underline{\beta}_n(x_n), \overline{\beta}_n(x_n)\right]$. The angles $\underline{\alpha}_n(\cdot), \overline{\alpha}_n(\cdot), \underline{\beta}_n(\cdot)$, and $\overline{\beta}_n(\cdot)$ are measured in the coordinate axes attached to R_n. In this notation, the set $\mathcal{Q}(J)$ is

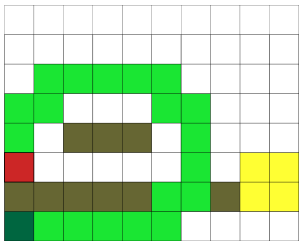
$$\mathcal{Q}(J) := \{(w_2, 0, \alpha_2) : w_2 \in [u_2, v_2], \alpha_2 \in [\underline{\alpha}_2(w_2), \overline{\alpha}_2(w_2)]\}.$$

The solution of CBTA-S1 and CBTA-S2 involves geometric constructions described in detail in [28], [29].

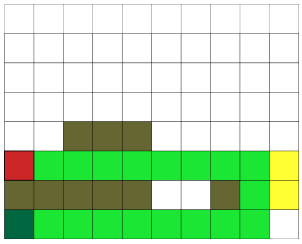
A. Illustrative Examples and Discussion

Figures 4 and 5 illustrate the application of the proposed approach. In each example, a workspace cell decomposition with uniformly-sized square cells was considered. The indices assigned to these cells are not shown in Figures 4 and 5. Instead, for the reader's convenience, different colors are used to indicate associations with atomic propositions, as in (3). In what follows, we use the notation of Section II.

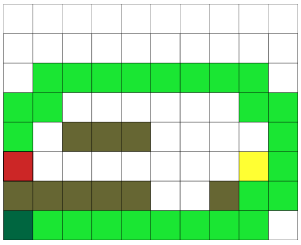
In the context of Fig. 4, the number of atomic propositions was set to $K = 4$. The cells associated with atomic propositions $\lambda_1, \lambda_2, \lambda_3$, and λ_4, respectively, are indicated with Fig. 4 in white (free space), gray (obstacles), red, and yellow colors. The dark green cell in the lower left corner is the cell containing $\mathsf{x}(\xi_0)$. The sequence of light green colored cells in Figure 4(a) indicate the prefix portion of the path obtained as a result of searching the product transition system described in Section IV, for the LTL$_{-X}$



(a) Specification ϕ_1 and $\rho = 2.5$ units.

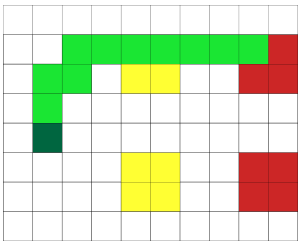


(b) Specification ϕ_2 and $\rho = 1$ unit.

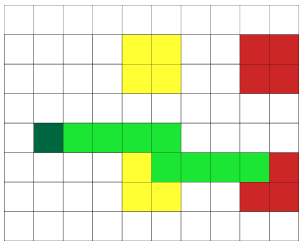


(c) Specification ϕ_2 and $\rho = 2.5$ unit.

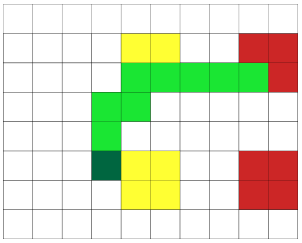
Fig. 4. Application of the proposed approach: illustration of the effects on the resultant path of changes in the control input constraint. The numerical values ρ are in dimensionless distance units, where 1 unit is equal to the side of a cell in the uniform decomposition.



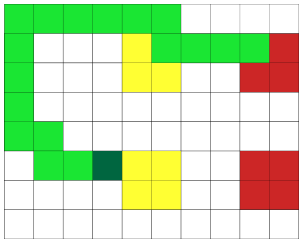
(a) $\xi_0 = (1.5, 4, \pi/2)$ units, $\rho = 2$ units.



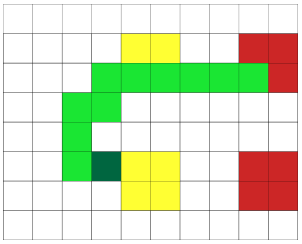
(b) $\xi_0 = (2, 3.5, 0)$ units, $\rho = 2$ units.



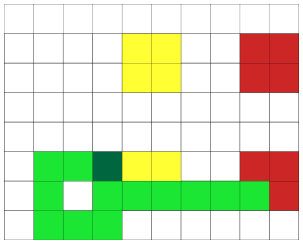
(c) $\xi_0 = (3.5, 3, \pi/2)$ units, $\rho = 2$ units.



(d) $\xi_0 = (3, 2.5, \pi)$ units, $\rho = 3.5$ units.



(e) $\xi_0 = (3, 2.5, \pi)$ units, $\rho = 1.5$ units.



(f) $\xi_0 = (3, 2.5, \pi)$ units, $\rho = 1$ unit.

Fig. 5. Application of the proposed approach: illustration of the effects on the resultant path of changes in the initial state. Here, the LTL$_{-X}$ specification is ϕ_3 (described in text). The numerical values of position coordinates and of ρ are in dimensionless distance units, where 1 unit is equal to the side of a cell in the uniform decomposition.

formula $\phi_1 := \Box\lambda_1 \wedge \Box\neg\lambda_2 \wedge \Diamond\lambda_3$. For the result indicated in Figure 4(a), the control input was assumed to be constrained to the interval $\left[-\frac{1}{\rho}, \frac{1}{\rho}\right]$, with $\rho = 2.5$ units. Here, the unit of measuring ρ is the size of a cell in the (uniformly-sized) cell decomposition. Informally, this specification requires the vehicle to avoid the gray-colored regions, and visit the red-colored region.

The sequences of light green colored cells in Fig. 4(b) and Fig. 4(c) indicate the prefix portions of resultant paths for the LTL$_{-X}$ formula $\phi_2 := \Box\lambda_1 \wedge \Box\neg\lambda_2 \wedge \Diamond\lambda_3 \wedge \Diamond\lambda_4$. Informally, this specification requires the vehicle to avoid the gray-colored regions, and visit the red- and yellow-colored regions both. The different results in Fig. 4(b) and Fig. 4(c) are a consequence of imposing different control input constraints. In particular, for the result indicated in Fig. 4(b), we set $\rho = 1$ unit, whereas for the result indicated in Fig. 4(c), we set $\rho = 2.5$ units. Notice that the *same* LTL$_{-X}$ specification is satisfied by two markedly different paths due to the different control input constraints.

In the context of Fig. 5, the number of atomic propositions was set to $K = 3$. The cells associated with atomic propositions λ_1, λ_2, and λ_3, respectively, are indicated with Fig. 5 in white, red, and yellow colors. The dark green cell near the center is the cell containing $\mathsf{x}(\xi_0)$. The sequence of light green colored cells in each of Figs. 5(a)–5(f) indicate the prefix portion of the path obtained as a result of searching the product transition system, for the LTL$_{-X}$ formula $\phi_3 := \Box\lambda_1 \wedge \Diamond\lambda_2 \wedge \Diamond\lambda_3$. Notice that the regions associated with the propositions λ_2 and λ_3 (the red- and yellow-colored regions, respectively) are unions of disconnected subregions. This ability to associate disconnected regions with a single atomic proposition is convenient because it affords the use of fewer atomic propositions. Furthermore, this ability is not available in other approaches reported in the literature, e.g. [14], which depend on partitioning the vehicle's *state space* instead of its workspace (output space). The different resultant paths in each of Figs. 5(a)–5(f) are due to different initial states, as indicated in the captions.

VI. Conclusions and Future Work

In this paper, we discussed a new technique for the control with LTL$_{-X}$ specifications of a nonholonomic vehicle model. The proposed technique relies on workspace cell decompositions, which involve partitioning in a space of smaller dimension than the state space, as is often done in the literature. The proposed technique involved discrete abstraction of the vehicle kinematic model using the so-called lifted graph, and by apropriately assigning edge transition costs in the lifted graph. The relationship of these edge transition costs with certain forward- and backward reachable sets of the vehicle model was discussed.

The proposed approach is applicable more generally to differentially flat nonlinear systems, especially when control

input constraints can be mapped to constraints on the flat output-space trajectories. The control constraints for the vehicle model considered in this paper were mapped to curvature constraints on the admissible workspace (output space) trajectories. Therefore, analysis of forward- and backward reachable sets with curvature-bounded curves was directly applied for finding lifted graph edge transition costs for this vehicle model. Future work includes the application of the proposed approach for control with temporal logic specifications of other differentially flat nonlinear systems.

Acknowledgments: This research was funded in part by US Air Force SBIR contract # FA8501-14-P-0034 awarded to Aurora Flight Sciences (AFS) Corp., Cambridge, MA, in collaboration with WPI.

REFERENCES

[1] H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. The MIT Press, 2005.

[2] A. E. Bryson and Y.-C. Ho, *Applied Optimal Control*. New York, NY, USA: Taylor & Francis, 1975.

[3] J. T. Betts, "Survey of numerical methods for trajectory optimization," *Journal of Guidance, Control, and Dynamics*, vol. 21, no. 2, pp. 193–204, 1998.

[4] C. Belta, A. Bicchi, M. Egerstedt, E. Frazzoli, E. Klavins, and G. J. Pappas, "Symbolic planning and control of robot motion," *IEEE Robotics and Automation Magazine*, pp. 61 – 70, March 2007.

[5] R. Alur, T. A. Henzinger, G. Lafferriere, and G. J. Pappas, "Discrete abstractions of hybrid systems," *Proceedings of the IEEE*, vol. 88, pp. 971–984, July 2000.

[6] P. Tabuada, *Verification and Control of Hybrid Systems: A Symbolic Approach*. Springer, 2008.

[7] 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.

[8] 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.

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

[10] P. Tabuada, "Controller synthesis for bisimulation equivalence," *Systems and Control Lett.*, vol. 57, pp. 443–452, June 2008.

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

[12] 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.

[13] 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.

[14] 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.

[15] Y. Yordanov, J. Tùmova, 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.

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

[17] M. Zamani, G. Pola, M. Mazo, and P. Tabuada, "Symbolic models for nonlinear control systems without stability assumptions," *IEEE Transactions on Automatic Control*, vol. 57, no. 7, pp. 1804–1809, 2012.

[18] M. Fliess, J. Levine, P. Martin, and P. Rouchon, "Flatness and defect of nonlinear systems: Introductory theory and examples," *International Journal of Control*, vol. 71, no. 5, pp. 745–765, 1995.

[19] M. J. van Nieuwstadt and R. M. Murray, "Real-time trajectory generation for differentially flat systems," *International Journal of Robust and Nonlinear Control*, vol. 8, pp. 995–1020, 1998.

[20] 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 45^{th} IEEE Conference on Decision and Control*, (San Diego, CA, USA), pp. 2406–2411, December 2006.

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

[22] 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.

[23] D. P. Bertsekas and I. B. Rhodes, "On the minimax reachability of target sets and target tubes," *Automatica*, vol. 7, pp. 233–247, 1971.

[24] 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.

[25] 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.

[26] J. E. Hopcroft, R. Motwani, and J. D. Ullman, *Introduction to automata theory, languages, and computation*. Boston, MA, USA: Addison-Wesley, 2nd ed., 2001.

[27] E. J. Cockayne and G. W. C. Hall, "Plane motion of a particle subject to curvature constraints," *SIAM Journal on Control*, vol. 13, no. 1, pp. 197–220, 1975.

[28] 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.

[29] R. V. Cowlagi, *Hierarchical Motion Planning for Autonomous Aerial and Terrestrial Vehicles*. PhD thesis, Georgia Institute of Technology, 2011.

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 $j < i$, $\omega_j^P \models \phi_1$.