Hierarchical Trajectory Optimization in Hybrid Dynamical Systems

Raghvendra V. Cowlagi [a]

[a] *Assistant Professor, Aerospace Engineering Program, Worcester Polytechnic Institute, Worcester, MA 01609, USA.*

Abstract

We discuss trajectory optimization for hybrid systems with a natural, hierarchical separation of discrete and continuous dynamics. The trajectory optimization problem considered requires that a discrete state sequence and a continuous state trajectory must be both determined to minimize a single cost function, such that the discrete state sequence also solves a symbolic planning problem. We model this symbolic planning problem as a search on a planning graph, and we introduce a family of graphs called *lifted planning graphs* parametrized by an integer H. We define a family of continuous state trajectory optimization problems and associate them with edge costs in the lifted planning graphs. Next, we present an algorithm for finding an optimal solution to the hybrid trajectory optimization problem, which includes mapping paths in the lifted planning graphs to discrete state sequences and continuous state trajectories. We show that the cost of optimal hybrid trajectories is a nonincreasing function of H, and that there exists a finite H for which this cost attains a minimum. We illustrate the proposed algorithm with numerical simulation results for two application examples: an autonomous mobile vehicle and an autonomous robotic manipulator.

1 Introduction

The typical topics addressed in the literature on the control of hybrid dynamical systems include mathematical modeling and existence of solutions (Goebel et al., 2012); Lyapunov-based robust stabilization and and switching conditions for stability (Branicky, 1998; Liberzon, 2003; Haddad et al., 2006; Sanfelice, 2013); abstraction, verification, reachability, and dead-lock (Alur et al., 2000; Tabuada, 2008); and hybrid optimal control, which is the subject of this paper.

The *mode switched system* model dominates the literature on hybrid optimal control. For problems where the mode switching sequence is fixed, first-order necessary conditions involving adjoint states, similar to the Pontryagin Minimum Principle, are developed (Sussmann, 1999; Piccoli, 1998; Garavello and Piccoli, 2005). Explicit algorithms and applications for control design based on such necessary conditions are presented (Shaikh and Caines, 2007; Pakniyat and Caines, 2015a). For controlled switching, the determination of derivatives of the value function with respect to switching times is addressed (Xu and Antsaklis, 2002; Ding et al., 2009; Kamgarpour and Tomlin, 2012). Cassandras, Pepyne, and Wardi (2001) address the case where switching time instants are described by a controlled event-driven dynamical system. Extensions of dynamic programming to hybrid systems are addressed (Hedlund and Rantzer, 2002; Barles et al., 2010; Rungger and Stursberg, 2011), including connections to the adjoint variable in the necessary conditions (Pakniyat and Caines, 2015b), but the fundamental "curse of dimensionality" remains at least as severe as for continuous systems. The determination of optimal mode selection in systems with autonomous continuous dynamics is addressed using gradient descent-like algorithms (Axelsson et al., 2008; Sager, 2009). Zhu and Antsaklis (2013) provide an excellent survey of recent advances.

Hybrid systems involving a natural hierarchy of discrete and continuous subsystems are not directly addressed by the aforesaid methods. Such hierarchies arise, for instance, in autonomous robotics, where the high-level autonomy and intelligence are modeled by discrete state automata or state transition systems, and the low-level physical system and its control are modeled by continuous-time dynamical systems. Similar hierarchies arise in the chemical process industry (high-level production scheduling and low-level process control) and the need for algorithms for their seamless integration to achieve overall optimality is emphasized (Engell et al., 2000; Engell and Hajunkoski, 2012). In the optimal control of such hybrid systems, the discrete state sequence and the continuous state trajectory must be both determined with the objective of minimizing a single cost function. Furthermore, the discrete state sequence is often associated with a *symbolic* planning problem (Bemporad and Giorgetti, 2006; Belta et al., 2007). To this end, a *hybrid automaton* model (Alur et al., 2000) is more suitable, and the preservation of the natural hierarchy of the system of engineering importance (Engell and Hajunkoski, 2012).

We discuss a hierarchical trajectory optimization algorithm for hybrid control problems involving symbolic planning at the higher level and continuous state trajectory optimization at the lower level. We model the symbolic planning problem as a graph search on a *planning graph*. Next, we introduce a family of graphs called *lifted planning graphs* parametrized by an integer H. Next, we define a family of low-level trajectory optimization problems and associate them with edge costs in the lifted planning graphs. Next, we present an algorithm for finding an optimal solution to the hybrid trajectory optimization problem, which involves mapping paths in the lifted planning graphs to discrete-state sequences and continuous-state trajectories. We show that the cost of optimal hybrid trajectories is a nonincreasing function of H, and that there exists a finite H for which this cost attains a minimum. We illustrate the proposed algorithm with two examples: an autonomous mobile vehicle, and an autonomous robotic manipulator.

Contributions: First, we propose a new hybrid trajectory optimization algorithm for the case where an optimal discrete state sequence and an optimal continuous state trajectory must be both simultaneously computed. The proposed algorithm is applicable to hybrid systems with a natural hierarchy of discrete and continuous subsystems. The proposed algorithm introduces a loose coupling between the optimization problems at the two hierarchical levels, without entirely eliminating the hierarchical problem structure. This feature is beneficial because the optimization problems at the two hierarchical levels are typically solved using fundamentally different methods. Second, we provide important application examples from robotics that involve a symbolic planning problem at the higher level, and control for a dynamical system at the lower level. We demonstrate that the proposed approach produces plans with significantly lower cost compared to the ad hoc approach of hierarchical separation. Third, we provide a fundamental result for hierarchical hybrid systems on the relationship of sequences of discrete state transitions to costs defined on continuous state trajectories.

Preliminary results of this paper were previously presented (Cowlagi, 2015). This paper presents broader theoretical results, formal proofs, and an additional application example, which were not previously discussed.

2 Problem Formulation

We consider a hybrid dynamical system $\mathcal{H}$ with discrete state in a finite set V, with $|V| = N^{\mathrm{V}} \in \mathbb{Z}_{>0}$, and continuous state in an open set $\mathcal{D} \subseteq \mathbb{R}^n$. The state of the system is denoted by (v, ξ), where $v \in V$ and $\xi \in \mathcal{D}$. We label vertices in V with superscripts, e.g. $v^1, v^2, \ldots, v^{N^{\mathrm{V}}}$ whereas we use *sub*script to denote an index within a sequence of discrete states. The evolution of the discrete state is described by a labeled transition system $\mathcal{T}$ consisting of a finite set of labels Ω, with $|\Omega| = N^{\mathrm{L}} \in \mathbb{Z}_{>0}$, and a transition map $\delta : V \times \Omega \to V$. We assume that each discrete state $v \in V$ is associated with a compact set $\Phi_v \subseteq \mathcal{D}$, such that for every pair $v^1, v^2 \in V$,

$$\left(v^k \neq v^\ell\right) \Leftrightarrow \left(\Phi_{v^k} \cap \Phi_{v^\ell} = \varnothing\right), \quad k, \ell \in \{1, \ldots, N^{\mathrm{V}}\}.$$

The initial state of the system $\mathcal{H}$ is denoted by $(v^{\mathrm{s}}, \xi^{\mathrm{s}})$, and unless otherwise stated, we assume that $\xi^{\mathrm{s}} \in \Phi_{v^{\mathrm{s}}}$. The evolution of the continuous state is described by

$$\dot{\xi}(t) = f\left(\xi(t), u(t)\right), \tag{1}$$

where u is the control input and $U \in \mathbb{R}^m$ is the set of admissible control input values. The initial time is assumed to be zero. We denote by $\mathcal{U}$ the set of all piecewise continuous functions of time taking values in U. We assume that $f : \mathbb{R}^{n+m} \to \mathbb{R}^n$ is globally Lipschitz continuous in $\mathbb{R}^{n+m}$ due to which the global existence and uniqueness of a solution to (1), denoted $\xi(t; \xi^{\mathrm{s}}, u)$, is guaranteed for all $(\xi, u) \in \mathcal{U}$ (Haddad and Chellaboina, 2008).

A *plan* $\varpi = (\omega_0, \omega_1, \ldots, \omega_{P-1})$ is uniquely associated with a finite sequence of discrete states $\mathbf{v} = (v_0, v_1, \ldots, v_P)$ such that $\delta(v_k, \omega_k) = v_{k+1}$ for each $k = 0, \ldots, P-1$. We make this association explicit by denoting the plan as $\varpi(\mathbf{v})$, which is said to *transfer* the discrete state from v_0 to v_P. For a plan ϖ, an *executive control* is a pair (t^{f}, u) such that the control input u drives the continuous state through each of the regions Φ_{v_k} in sequential order in a finite time t^{f}. More precisely, (t^{f}, u) are such that there exists a strictly increasing sequence $\{t_1, t_2, \ldots, t_P\} \in \left[0, t^{\mathrm{f}}\right]$ satisfying $t_P = t^{\mathrm{f}}$, and $\xi(t_k; \xi^{\mathrm{s}}, u) \in \Phi_{v_k}$, for each $k = 1, 2, \ldots, P$. We denote $(t^{\mathrm{f}}, u) \vdash \varpi$ to indicate that (t^{f}, u) is an executive control for ϖ. The *cost* Λ of an executive control (t^{f}, u) is:

$$\Lambda(t^{\mathrm{f}}, u) := \int_0^{t^{\mathrm{f}}} L(\xi(t; \xi^{\mathrm{s}}, u),\ u(t))\ \mathrm{d}t, \tag{2}$$

where $L : \mathbb{R}^{n+m} \to \mathbb{R}_+$ is a bounded function.

The main problem of interest in this paper as follows.

Problem 1 *Let $v^{\mathrm{s}}, v^{\mathrm{g}} \in V$ and $\xi^{\mathrm{s}} \in \mathcal{D}$ be prespecified. Find a plan ϖ^* that transfers the discrete state from v^{s} to v^{g}. Also find an executive control $(t^{\mathrm{f}*}, u^*) \vdash \varpi^*$ such that, for every plan ϖ that achieves the same discrete state transfer, and every executive control $(t^{\mathrm{f}}, u) \vdash \varpi$,*

$$\Lambda(t^{\mathrm{f}*}, u^*) \leqslant \Lambda(t^{\mathrm{f}}, u).$$

Note that $t^{\mathrm{f}*}$ may be prespecified in certain applications. We assume that the discrete state transition system $\mathcal{T}$

models a high-level symbolic planning problem, a specific example of which is the *classical planning problem* (CPP) in the artificial intelligence literature (Russell and Norvig, 2003). A CPP consists of:

(1) A finite set of *objects* $\{o^1, \ldots, o^{N^{\mathrm{O}}}\}$.
(2) A finite set of *predicates* $\{p^1, \ldots p^{N^{\mathrm{P}}}\}$. Each predicate accepts one or more arguments from the set of objects. A predicate evaluated for specific object(s) is a literal, which takes values in $\{\mathsf{true}, \mathsf{false}\}$.
(3) A finite set of *CPP states* $V = \{v^1, \ldots v^{N^{\mathrm{V}}}\}$, where each state is a conjunction of literals, which can take values in $\{\mathsf{true}, \mathsf{false}\}$. The *current CPP state* is the unique state that is true. An initial CPP state v^{s} and a goal CPP state v^{g} are prespecified.
(4) A finite set of *actions* $\Omega = \{\omega^1, \ldots \omega^{N^{\mathrm{L}}}\}$. Each action $\omega \in \Omega$ is associated with a *precondition* $\mathsf{pre}(\omega)$ and an *effect* $\mathsf{eff}(\omega)$. The precondition $\mathsf{pre}(\omega)$ describes the conditions that must be true before the action ω can be executed, whereas the effect $\mathsf{eff}(\omega)$ describes the changes to the current CPP state, i.e., there is a unique $v \in V$ that is true when $\mathsf{eff}(\omega)$ is true, denoted by $v \equiv \mathsf{eff}(\omega)$.

Clearly, a CPP is modeled by the transition system $\mathcal{T}$ with transition map δ defined by

$$\delta(v^k, \omega^\ell) := \begin{cases} v^m \equiv \mathsf{eff}(\omega^\ell), & \text{if } \mathsf{pre}(\omega^\ell) = \mathsf{true}, \\ v^k, & \text{if } \mathsf{pre}(\omega^\ell) = \mathsf{false}. \end{cases}$$

Problem 1 has a two-level hierarchical structure. First, the discrete state transfer problem can be solved by a graph search or other symbolic planning methods (Russell and Norvig, 2003) to determine a plan $\overline{\omega}$. Subsequently, an executive control can be computed by formulating and solving the associated optimal control problem (see Section 2.1). On the one hand, it is advantageous to maintain this hierarchical separation to solve Problem 1, because existing methods of symbolic planning and optimal synthesis can be employed without significant modification. On the other hand, this hierarchical separation can lead to sub-optimal solutions. Symbolic planning methods often consider the number of state transitions as a cost function (Russell and Norvig, 2003), but for Problem 1, the costs of two plans with an equal number of transitions can be significantly different.

In the light of these observations, an exhaustive, but naïve, approach to the solution of Problem 1 is the enumeration of all plans that solve the symbolic planning problem – there are a finite number of such plans – followed by the evaluation of the associated cost (2) for each plan to determine the least cost plan. Whereas this naïve approach is not computationally practical, it is guaranteed to solve Problem 1. In what follows, we present a computationally efficient procedure that *asymptotically* emulates the preceding exhaustive approach.

2.1 Optimal Executive Control (OEC) Problem

Problem 2 (OEC($\mathbf{v}, \xi_0$)) *Given a discrete state transition sequence $\mathbf{v} = (v_0, \ldots, v_P)$, and a continuous state $\xi_0 \in \Phi_{v_0}$, find $(t^{\mathrm{f}*}, u^*) \vdash \overline{\omega}(\mathbf{v})$ to minimize*

$$\int_0^{t^{\mathrm{f}*}} L(\xi(t; \xi_0, u), u(t))\, \mathrm{d}t. \tag{3}$$

An implicit constraint (by definition of executive control) is that there must exist a positive and strictly increasing sequence $\{t_1^*, t_2^*, \ldots, t_P^*\}$, with $t_P^* = t^{\mathrm{f}*}$ such that

$$\xi\left(t_\ell^*; \xi_0, u^*\right) \in \Phi_{v_\ell}, \quad \textit{for each } \ell = 1, \ldots, P. \tag{4}$$

As previously mentioned, $t^{\mathrm{f}*}$ may be prespecified in certain applications. The OEC problem is a standard trajectory optimization problem (with the exception of the additional constraints in (4)). For cases where analytical solutions to OEC are not readily available, numerical solutions can be obtained by "off-the-shelf" algorithms such as pseudospectral methods (Garg et al., 2010). Here, we focus on the solution of Problem 1, assuming that a procedure **oec** (see Algorithm 1) is available to solve the OEC problem.

procedure **oec**($\mathbf{v}, \xi_0$)
1: Find $(t^{\mathrm{f}*}, u^*) \vdash \overline{\omega}(\mathbf{v})$ and the sequence of time instants in (4), as discussed in Problem 2
2: Return $t_1^*, t^{\mathrm{f}*}$, and u^*

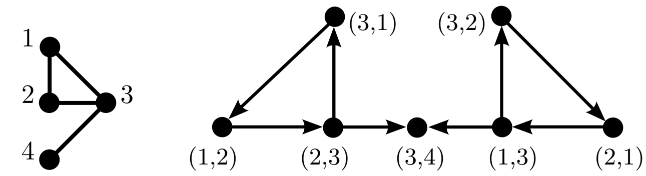
Algorithm 1. A procedure to solve the OEC problem is assumed to be available.

3 Lifted Planning Graph

The transition system $\mathcal{T}$ is naturally associated with a graph $\mathcal{G}$, with each vertex uniquely associated with states in V, and each edge associated with a label. We refer to $\mathcal{G}$ as the *planning graph*, and in a minor abuse of notation, we denote the vertex set of this graph by V, and label individual vertices as $v^1, v^2, \ldots$ and so on. As before, we use the symbol v with a *sub*script to denote an index within a sequence. The edge set $E \subseteq V \times V$ is the set of all pairs (v^i, v^j) such that there exists a label $\omega \in \Omega$ with $\delta(v^i, \omega) = v^j$. A *path* in the graph $\mathcal{G}$ is a finite sequence of successively adjacent vertices.

The central idea of the proposed approach, namely, *lifted planning graphs*, is defined next. A similar approach is discussed in the literature for robot motion-planning (Rippel et al., 2005; Cowlagi and Tsiotras, 2012). For every integer $H \geqslant 0$, let

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



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

Fig. 1. Example of a lifted graph.

Every element $\mathbf{i} \in V_H$ is an ordered $(H+1)$-tuple of elements of V. We denote by $[\mathbf{i}]_k$ the k^{th} element of $\mathbf{i}$ for $k \leqslant H+1$. 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 $[\mathbf{i}]_1 \neq [\mathbf{j}]_{H+1}$. Each edge in E_H is an ordered $(H+2)$-tuple of elements of V. The *lifted planning 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 $H = 1$. Note that $\mathcal{G}_0 = \mathcal{G}$.

Proposition 3 *For every $H \in \mathbb{Z}_{\geqslant 0}$, and every integer $\hat{P} > H$, there exists a bijective mapping b_H from the set of state transition sequences of length $\hat{P}$ in the system $\mathcal{T}$ to the set of paths of length $\hat{P} - H$ in the graph $\mathcal{G}_H$.*

Proof. Let $P := \hat{P} - 1$, and consider a sequence $\mathbf{v} = (v_0, \ldots, v_P)$ in $\mathcal{T}$. Define $\bar{\mathbf{i}} := (\mathbf{i}_0, \ldots, \mathbf{i}_{P-H})$, where $\mathbf{i}_k := (v_k \ldots, v_{k+H}) \in V_H$, for each $k = 0, \ldots, P-H$.. By definition, $[\mathbf{i}_k]_m = [\mathbf{i}_{k+1}]_{m-1}$ for each $m = 2, \ldots, H+1$. Therefore, $(\mathbf{i}_k, \mathbf{i}_{k+1}) \in E_H$ and $\bar{\mathbf{i}}$ is a path in $\mathcal{G}_H$. Because $\bar{\mathbf{i}}$ is uniquely defined, the mapping defined by $b_H(\mathbf{v}) := \bar{\mathbf{i}}$ is one-to-one. Now let $\bar{\mathbf{i}} := (\mathbf{i}_0, \mathbf{i}_1, \ldots, \mathbf{i}_{P-H})$ be any path in $\mathcal{G}_H$, and let $\mathbf{v} := (v_0, v_1, \ldots, v_P)$, where

$$v_m := \begin{cases} [\mathbf{i}_m]_1, & m = 0, \ldots, P-H, \\ [\mathbf{i}_{P-H}]_{m-P+H+1}, & m = \hat{P} - H, \ldots, P. \end{cases} \tag{5}$$

By definition of $\mathcal{G}_H$, $\mathbf{v}$ is a state transition sequence in $\mathcal{T}$. Also $b_H(\mathbf{v}) = \bar{\mathbf{i}}$, which implies that b_H is onto, and therefore bijective, because $\bar{\mathbf{i}}$ was arbitrarily chosen. □

Corollary 4 *Let $\bar{\mathbf{i}} = (\mathbf{i}_0, \ldots, \mathbf{i}_{P-H})$ be a path in the graph $\mathcal{G}_H$. The plan $\varpi(b_H^{-1}(\bar{\mathbf{i}}))$ transfers the discrete state from $[\mathbf{i}_0]_1$ to $[\mathbf{i}_{P-H}]_{H+1}$.*

Proof. Immediate from (5) for $m = 0$ and $m = P$. □

Let $\mathbf{v} = (v_0, \ldots, v_P)$ be a state transition sequence in $\mathcal{T}$, and define $\mathbf{v}_k := (v_k, \ldots, v_{k+H+1})$ for $k = 0, \ldots, P_H$, where $P_H := P - H - 1$. Let $t^*_{1,k}, t^{\text{f}*}_k$, and u^*_k denote the outputs of the procedure **oec**$(\mathbf{v}_k, \xi_k)$, where $\xi_0 = \xi^{\text{s}}$ and $\xi_k := \xi\left(t^*_{1,k-1}; \xi_{k-1}, u^*_{k-1}\right)$ for $k \geqslant 1$. The *H-cost* of the sequence $\mathbf{v}$ is defined by $\mathcal{J}_H(\mathbf{v}) = \sum_{k=0}^{P_H} \Lambda_k$, where

$$\Lambda_k := \begin{cases} \int_0^{t^*_{1,k}} L(\xi(t; \xi^*_k, u^*_k), u^*_k(t))\mathrm{d}t, & k < P_H, \\ \int_0^{t^*_{\mathrm{f},k}} L(\xi(t; \xi^*_k, u^*_k), u^*_k(t))\mathrm{d}t, & k = P_H. \end{cases} \tag{6}$$

If an executive control $(t^{\text{f}*}_k, u^*_k)$ does not exist for some $k \in \{0, \ldots, P_H\}$, then we assign $\Lambda_k = \chi$ for each $\ell = k, \ldots, P_H$, where $\chi \gg 1$ is sufficiently large. A crucial observation is that the H-cost of $\mathbf{v}$ is the sum of edge transition costs in the path $\bar{\mathbf{i}} := b_H(\mathbf{v}) = (\mathbf{i}_0, \ldots, \mathbf{i}_{P-H})$. Specifically, we define an edge transition cost function $g_H(\mathbf{i}_k, \mathbf{i}_{k+1}) := \Lambda_k$, for each $k = 0, \ldots, P_H$, and (with a minor abuse of notation) the cost of the path $\bar{\mathbf{i}}$ as $\mathcal{J}_H(\bar{\mathbf{i}}) := \sum_{k=0}^{P_H} g_H(\mathbf{i}_k, \mathbf{i}_{k+1})$.

3.1 Hybrid Trajectory Optimization

We propose to solve Problem 1 by finding an H-cost optimal discrete state sequence, which is equivalent to finding an optimal path in the graph $\mathcal{G}_H$. We show that, for a sufficiently large H, the H-cost optimal sequence solves Problem 1. To this end, we propose Algorithm 2, which is a label-correcting algorithm similar to Dijkstra's algorithm (Bertsekas, 2000).

The inputs to this algorithm are $v^{\text{s}}, v^{\text{g}}$ and ξ^{s}. The algorithm is initialized by finding a set $I^{\text{s}} \subset V_H$, where

$$I^{\text{s}} := \{\mathbf{i}_0 \in V_H \mid v^{\text{s}} = [\mathbf{i}_0]_1\}.$$

Note that I^{s} is the set of possible initial vertices in V_H in the desired optimal path in $\mathcal{G}_H$. To address multiple elements in I^{s}, we add to V_H a new vertex $\mathbf{i}^{\text{s}}$, and we add edges $(\mathbf{i}^{\text{s}}, \mathbf{i}_0)$ to E_H, for each $\mathbf{i}_0 \in I^{\text{s}}$. The transition costs of each of these new edges are set to zero. Similar to Dijkstra's algorithm, the proposed algorithm maintains for each vertex $\mathbf{i} \in V_H$ a *label* $d(\mathbf{i})$, and a *backpointer* $b(\mathbf{i})$, along with a *fringe* $\mathcal{P}$, which is a collection of vertices whose labels can potentially be reduced. Additionally, the algorithm maintains for each vertex in V_H a continuous state $\Xi(\mathbf{i}) \in \mathcal{D}$, a time instant $\Theta(\mathbf{i}) \in \mathbb{R}_+$, and an admissible control input $\Upsilon(\mathbf{i}) \in \mathcal{U}$.

At each iteration of the **main** procedure in the proposed algorithm, a vertex from the fringe is removed in Line 3. The procedure **remove** returns a vertex with the least value of the label. The procedure **insert** in Line 15 depends on the specific data structure (e.g. a Fibonacci tree) used to sort the fringe by the values of labels. In Line 10, edge transition costs are evaluated by solving an OEC problem. The continuous state resulting from the OEC solution is stored (Line 21), and supplied as an initial condition for successive OEC problems (Line 4).

The output of the proposed algorithm is a path $\bar{\mathbf{i}}^*_H = (\mathbf{i}_0, \ldots, \mathbf{i}_{P-H})$ in $\mathcal{G}_H$, where $[\mathbf{i}_0]_1 = v^{\text{s}}$ and $[\mathbf{i}_{P-H}]_{H+1} = v^{\text{g}}$. By Corollary 4, the plan $\varpi^* := \varpi(b_H^{-1}(\bar{\mathbf{i}}^*_H))$ transfers

Algorithm parameter: $H \geqslant 1$

Input: $v^{\mathrm{s}}, v^{\mathrm{g}} \in V$ and $\xi^{\mathrm{s}} \in \Phi_{v^{\mathrm{s}}}$, Output: $\bar{\mathbf{i}}_H^*$

procedure **initialize**

1: Compute I^{s}, and augment V_H with $\mathbf{i}^{\mathrm{s}}$
2: $\mathcal{P} := \{\mathbf{i}^{\mathrm{s}}\}$, $d(\mathbf{i}^{\mathrm{s}}) := 0$
3: $\Xi(\mathbf{i}^{\mathrm{s}}) = \xi^{\mathrm{s}}$, $\Theta(\mathbf{i}^{\mathrm{s}}) = 0$
4: goal_found := false, $\bar{\mathbf{i}}_H^* := \varnothing$
5: **for** each $\mathbf{i} \in V_H \backslash \{\mathbf{i}^{\mathrm{s}}\}$ **do**
6: $\quad d(\mathbf{i}) = \infty$;

procedure **main**

1: **initialize**
2: **while** $(\mathcal{P} \neq \varnothing)$ & $\neg$goal_found **do**
3: $\quad \mathbf{i} :=$ **remove**$(\mathcal{P})$
4: $\quad \xi_0 := \Xi(\mathbf{i})$
5: $\quad$**for** each $\mathbf{j} \in V_H$ such that $(\mathbf{i}, \mathbf{j}) \in E_H$ **do**
6: $\quad\quad$**if** $\mathbf{i} = \mathbf{i}^{\mathrm{s}}$ **then**
7: $\quad\quad\quad d_{\mathbf{ij}} := 0, \quad u^* := 0, \quad \xi^* := \xi^{\mathrm{s}}, \quad t^* := 0$
8: $\quad\quad$**else**
9: $\quad\quad\quad \mathbf{v} := (\mathbf{i}, [\mathbf{j}]_{H+1})$
10: $\quad\quad\quad t_1^*, t^{\mathrm{f}*}, u^* :=$ **oec**$(b_H(\mathbf{v}), \xi_0)$
11: $\quad\quad\quad$**if** $\exists \ell \in \{2, \ldots, H+1\}$ such that $[\mathbf{v}]_\ell = v^{\mathrm{g}}$ **then**
12: $\quad\quad\quad\quad t^* := t_\ell^*$, goal_found:= true
13: $\quad\quad\quad\quad \mathbf{i}^{\mathrm{g}} := \mathbf{j}$
14: $\quad\quad\quad$**else**
15: $\quad\quad\quad\quad t^* := t_1^*$
16: $\quad\quad\quad d_{\mathbf{ij}} := \int_0^{t^*} L(\xi(t; \xi_0, u^*), u^*(t))$,
17: $\quad\quad\quad \xi^* := \xi\left(t^*; \xi_0, u^*\right)$
18: $\quad\quad$**if** $d(\mathbf{i}) + d_{\mathbf{ij}} < d(\mathbf{j})$ **then**
19: $\quad\quad\quad d(\mathbf{j}) := d(\mathbf{i}) + d_{\mathbf{ij}}$
20: $\quad\quad\quad b(\mathbf{j}) := \mathbf{i}, \quad \Theta(\mathbf{j}) := t^*$
21: $\quad\quad\quad \Xi(\mathbf{j}) := \xi^*, \quad \Upsilon(\mathbf{j}) := u^*$
22: $\quad\quad\quad$**insert**$(\mathcal{P}, \mathbf{j})$
23: **while** $b(\mathbf{i}^{\mathrm{g}}) \neq \mathbf{i}^{\mathrm{s}}$ **do**
24: $\quad \bar{\mathbf{i}}_H^* := (\mathbf{i}^{\mathrm{g}}, \bar{\mathbf{i}}_H^*), \quad \mathbf{i}^{\mathrm{g}} = b(\mathbf{i}^{\mathrm{g}})$

Algorithm 2. The proposed solution to Problem 1.

the discrete state from v^{s} to v^{g}. An executive control input $(t^{\mathrm{f}*}, u^*) \vdash \overline{\omega}^*$ is defined by

$$t^{\mathrm{f}*} := t_{P-H}^{\mathrm{f}}, \quad u^*(t) := \Upsilon(\mathbf{i}_k), \text{ for } t \in \left[t_{k-1}^{\mathrm{f}}, t_k^{\mathrm{f}}\right),$$

where $t_k^{\mathrm{f}} := \sum_{m=0}^{k} \Theta(\mathbf{i}_m)$, for each $k = 1, \ldots, P-H$. Algorithm 2 implements Dijkstra's algorithm on the graph $\mathcal{G}_H$, and therefore the following result is self-evident.

Proposition 5 *For every integer parameter $H \geqslant 1$, the output $\bar{\mathbf{i}}_H^*$ of Algorithm 2 satisfies $\mathcal{J}_H(\bar{\mathbf{i}}_H^*) \leqslant \mathcal{J}_H(\bar{\mathbf{i}}_H)$ for every other path $\bar{\mathbf{i}}_H$ in $\mathcal{G}_H$ that achieves the same discrete state transfer.*

The main result of this paper is the following.

Proposition 6 *If a solution exists, then Algorithm 2 solves Problem 1 for a sufficiently large parameter H.*

Proof. For sufficiently large H, there exists an edge in E_H such that elements of the $H+2$-tuple defined by this edge include v^{s} and v^{g}, i.e., there are vertices $\mathbf{i}_0, \mathbf{i}_1 \in verts_H$ with $(\mathbf{i}_0, \mathbf{i}_1) \in E_H$ such that $[\mathbf{i}_0]_1 = v^{\mathrm{s}}$ and $[\mathbf{i}_1]_\ell = v^{\mathrm{g}}$, for some $\ell \in \{2, \ldots, H+1\}$. Therefore, the discrete state sequence $([\mathbf{i}_0]_1, [\mathbf{i}_1]_{H+1})$ achieves the desired state transfer, and an optimal executive control for the corresponding plan is found in Line 10 of Algorithm 2. Furthermore, Algorithm 2 iterates over *all* such edges in E_H, and the discrete state sequence associated with at least one of these edges solves Problem 1. □

Note that Prop. 6 states that Algorithm 2 emulates the exhaustive procedure for solving Problem 1 discussed in Section 2. Whereas Prop. 6 asserts the soundness of Algorithm 2, the following result asserts the computational efficiency of the proposed approach. To this end, we make a strong controllability assumption.

Assumption 7 *Given arbitrary $\xi_0, \xi_1 \in \mathcal{D}$, there exists an admissible control input that drives the continuous state of the hybrid system $\mathcal{H}$ from ξ_0 to ξ_1 in finite time.*

Proposition 8 *Let $\bar{\mathbf{i}}_H^*$ and $\bar{\mathbf{i}}_{H+1}^*$ be, respectively, the outputs of Algorithm 2 with parameters H and $H+1$. Then*

$$\mathcal{J}_{H+1}(\bar{\mathbf{i}}_{H+1}^*) \leqslant \mathcal{J}_H(\bar{\mathbf{i}}_H^*).$$

Proof. Define the path $\bar{\mathbf{j}}$ in $\mathcal{G}_{H+1}$ as:

$$\bar{\mathbf{j}} := b_{H+1}(b_H^{-1}(\bar{\mathbf{i}}_H^*)),$$

i.e. if $\bar{\mathbf{i}}_H^* = (\mathbf{i}_0, \ldots, \mathbf{i}_{P-H})$, then $\bar{\mathbf{j}} := (\mathbf{j}_0, \ldots, \mathbf{j}_{P-H-1})$ with $\mathbf{j}_k = (\mathbf{i}_k, [\mathbf{i}_{k+1}]_1)$ for each $k \in \{0, \ldots, P_H\}$. Note that $(\mathbf{i}_k, \mathbf{i}_{k+1})$ is an edge in E_H, and therefore a path of unit length in $\mathcal{G}_H$, and by definition, $(\mathbf{j}_k, \mathbf{j}_{k+1})$ is an edge in E_{H+1}, and therefore a path of unit length in $\mathcal{G}_{H+1}$. For each $k \in \{0, \ldots, P_H - 1\}$, let $(t^{\mathrm{f}*}, u^*)$ be an optimal executive control for the plan $\overline{\omega}(b_{H+1}^{-1}(\mathbf{j}_k, \mathbf{j}_{k+1}))$ associated with the path $(\mathbf{j}_k, \mathbf{j}_{k+1})$, and let $\{t_{1,k}^*, \ldots, t_{H+1,k}^*\}$ be the sequence of time instants as described in (4). The existence of this optimal executive control is guaranteed by Assumption 7. Let $u^*|_{t_{H,k}^*}$ denote the restriction of u^* to the interval $\left[0, t_{H,k}^*\right]$, and note that

$$(t_{H,k}^*, u^*|_{t_{H,k}^*}) \vdash \overline{\omega}(b_H^{-1}(\mathbf{i}_k, \mathbf{i}_{k+1})),$$

for each $k \in 0, \ldots, P_H - 1$. Therefore, by (6) and the subsequent definition of g_H, it follows that $g_{H+1}(\mathbf{j}_k, \mathbf{j}_{k+1}) = g_H(\mathbf{i}_k, \mathbf{i}_{k+1})$ for $k < P_H - 1$ and $g_{H+1}(\mathbf{j}_k, \mathbf{j}_{k+1}) \leqslant g_H(\mathbf{i}_k, \mathbf{i}_{k+1}) + g_H(\mathbf{i}_{k+1}, \mathbf{i}_{k+2})$, for $k = P_H - 1$. Therefore, $\mathcal{J}_{H+1}(\bar{\mathbf{j}}) \leqslant \mathcal{J}_H(\bar{\mathbf{i}}_H^*)$. By Prop. 5, it follows that

$$\mathcal{J}_{H+1}(\bar{\mathbf{i}}_{H+1}^*) \leqslant \mathcal{J}_{H+1}(\bar{\mathbf{j}}) \leqslant \mathcal{J}_H(\bar{\mathbf{i}}_H^*). \qquad \square$$

Proposition 8 states that the cost of the plan returned by Algorithm 2 is a nonincreasing function of the parameter H. This statement suggests an incrementally improving approximation to the solution of Problem 1, where Algorithm 2 is executed with successively larger values of H. Such an approach must be accompanied by an appropriate search heuristic to allow Algorithm 2 to converge faster using results previously obtained with smaller values of the parameter H.

4 Application Example 1: Autonomous Vehicle

Consider a planar route planning problem for a mobile robotic vehicle modeled by a Dubins vehicle, namely, a particle that moves forward at a constant speed and has a bounded rate of turn. A finite number of points of interest (POIs) are prespecified. A finite number of "cargo" pieces are stored at these POIs, with no more one than piece per POI. The vehicle's task is to interchange the locations of the cargo pieces, assuming that it can carry no more than one piece at a time. This task can be formulated as a CPP, the solution to which results in a prescribed vehicle route specifying the order in which POIs are to be visited (possibly with repetitions).

The Dubins vehicle model is described by the state $\xi = (x, y, \psi)$, where (x, y) are planar position coordinates of the vehicle center of mass, and $\psi \in \mathbb{S}^1$ is the direction of the velocity vector. The vehicle's motion is described by

$$\dot{x}(t) = \cos\psi(t), \quad \dot{y}(t) = \sin\psi(t), \quad \dot{\psi}(t) = u(t), \tag{7}$$

where u is the control input. We assume that the set of admissible control input values is the closed interval $U := \left[-\rho^{-1}, \rho^{-1}\right]$, where $\rho > 0$ is the minimum turn radius of this vehicle. The cost of an admissible trajectory is defined to be its length, i.e., $L \equiv 1$.

The vehicle task is formulated as a CPP as follows. Consider a set of objects $\{o^1, \dots, o^{(N^{\mathrm{C}}+N^{\mathrm{P}}+1)}\}$, where $N^{\mathrm{P}} > N^{\mathrm{C}} \geqslant 1$ are prespecified integers. The objects in the set $\mathcal{O}^{\mathrm{C}} := \{o^1, \dots, o^{N^{\mathrm{C}}}\}$ are each associated with a unique piece of cargo. Each object in $\mathcal{O}^{\mathrm{P}} := \{o^{N^{\mathrm{C}}+1}, \dots, o^{N^{\mathrm{C}}+N^{\mathrm{P}}}\}$ is uniquely associated with a point of interest (POI) in the plane. We denote by (x_{o^k}, y_{o^k}) the planar position coordinates of the POI associated with object $o^k \in \mathcal{O}^{\mathrm{P}}$. The object $o^{\mathrm{v}} := o^{(N^{\mathrm{C}}+N^{\mathrm{P}}+1)}$ is associated with the vehicle.

The predicate $p^1 \equiv \mathit{At}$ defines the literal $\mathit{At}(o_1, o_2)$, where $o_1 \in \mathcal{O}^{\mathrm{C}} \cup o^{\mathrm{v}}$, and $o_2 \in \mathcal{O}^{\mathrm{P}}$. The literal $\mathit{At}(o_1, o_2)$, is `true` whenever o_1 is present at the POI associated with o_2. Actions are defined by functions $A^1 \equiv \mathit{MoveVehicle}$ and $A^2 \equiv \mathit{MoveCargo}$. The function A^1 accepts arguments $o_1, o_2 \in \mathcal{O}^{\mathrm{P}}$ and defines an action $\omega = \mathit{MoveVehicle}(o_1, o_2)$ such that

$$\begin{aligned}\mathsf{pre}(\omega) &:= \mathit{At}(o^{\mathrm{v}}, o_1),\\ \mathsf{eff}(\omega) &:= \mathit{At}(o^{\mathrm{v}}, o_2) \wedge \neg \mathit{At}(o^{\mathrm{v}}, o_1).\end{aligned}$$

The function A^2 accepts arguments $o_1, o_2 \in \mathcal{O}^{\mathrm{P}}$, and $o_3 \in \mathcal{O}^{\mathrm{C}}$, and defines an action $\omega = \mathit{MoveCargo}(o_1, o_2, o_3)$ such that

$$\begin{aligned}\mathsf{pre}(\omega) &:= \mathit{At}(o^{\mathrm{v}}, o_1) \wedge \mathit{At}(o_3, o_1) \wedge_{o_4 \in \mathcal{O}^{\mathrm{C}} \setminus o_3} \neg \mathit{At}(o_4, o_2),\\ \mathsf{eff}(\omega) &:= \mathit{At}(o_3, o_2) \wedge \mathit{At}(o^{\mathrm{v}}, o_2) \wedge \neg \mathit{At}(o_3, o_1) \wedge \neg \mathit{At}(o^{\mathrm{v}}, o_1).\end{aligned}$$

Each CPP state v is defined by the conjunctive formula

$$v \equiv \mathit{At}\left(o^{\mathrm{v}}, o_{v,\mathrm{v}}\right) \wedge_{k=1}^{N^{\mathrm{C}}} \mathit{At}\left(o^k, o_{v,k}\right), \tag{8}$$

where $o_{v,\mathrm{v}}, o_{v,k} \in \mathcal{O}^{\mathrm{P}}$, and the set $\{o_{v,k}\}_{k=1}^{N^{\mathrm{C}}}$ has distinct elements. The set of actions is obtained by exhaustive enumeration of all possible combinations of arguments to the functions A^1 and A^2. The mapping Φ_v is defined by $\Phi_v := (x_{o_{v,\mathrm{v}}}, y_{o_{v,\mathrm{v}}}) \times \mathbb{S}^1$.

For numerical simulation, we choose $N^{\mathrm{C}} = 3$, and $N^{\mathrm{P}} = 5$. The coordinates of the planar locations of POIs are arbitrarily chosen, and indicated these POIs are shown in Fig. 2. The POIs $1, \dots, 5$ are designated, respectively, as objects $o^4, \dots, o^8$ in the CPP formulation. The initial and goal CPP states are

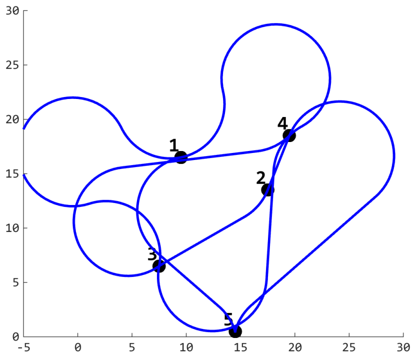
$$\begin{aligned}v_{\mathrm{in}} &\equiv \mathit{At}(o^1, o^6) \wedge \mathit{At}(o^1, o^4) \wedge \mathit{At}(o^1, o^5) \wedge \mathit{At}(o^{\mathrm{v}}, o^8),\\ v_{\mathrm{gl}} &\equiv \mathit{At}(o^1, o^4) \wedge \mathit{At}(o^1, o^6) \wedge \mathit{At}(o^1, o^7) \wedge \mathit{At}(o^{\mathrm{v}}, o^8).\end{aligned}$$

The vehicle's minimum turn radius is $\rho = 5$ units. The OEC problem here is equivalent to that of finding the shortest Dubins curve through a sequence of points, with the tangent direction unspecified at these points. An exact solution to this problem is time-consuming. Therefore, we implement the fast but approximate solution reported by Lee et al. (2000).

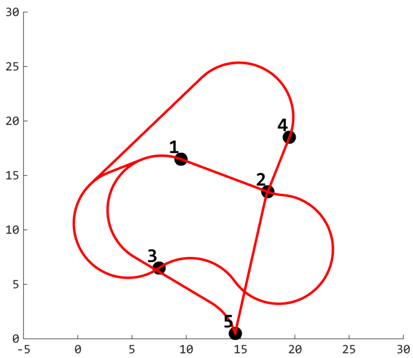
Table 1 shows a comparison of the plans obtained for this instance of Problem 1 by the "standard", hierarchically separated approach (i.e. by first solving the CPP to find a plan with minimum number of transitions and then finding an optimal executive control) with the proposed approach with $H = 1$. Each row in Table 1 indicates the POI location of the vehicle and each piece of cargo. The number of actions in each of these plans is the same, namely, nine. However, the total trajectory length in the "standard" plan is 242.9 units, whereas that in the plan found by the proposed approach is 187.7 units, which represents a 23% reduction in the trajectory length. These trajectories are shown in Figs. 2(a) and 2(b) respectively. A MATLAB-based software simulation of this application example is available at `http://users.wpi.edu/~rvcowlagi/software.html`.

5 Application Example 2: Robotic Manipulator

Consider a variant of the problem described in Section 4, where the movement of cargo pieces is to be executed by the end-effector of a robotic manipulator instead of



(a) Execution of “standard” plan.



(b) Execution of plan found by Alg. 2.

Fig. 2. Comparison of the paths associated with executive control for different plans for Application Example 1. The black-colored dots in each figure indicate POIs.

Table 1
Plans obtained for Application Example 1.

“Standard plan”				Plan by Alg. 2			
Vehicle	o^1	o^2	o^3	Vehicle	o^1	o^2	o^3
5	3	1	2	5	3	1	2
1	3	1	2	2	3	1	2
4	3	4	2	4	3	1	4
3	3	4	2	3	3	1	4
1	1	4	2	2	2	1	4
4	1	4	2	1	2	1	4
3	1	3	2	3	2	3	4
2	1	3	2	2	2	3	4
4	1	3	4	1	1	3	4
5	1	3	4	5	1	3	4

a mobile vehicle. Specifically, we consider a planar manipulator with two revolute joints (see Fig. 3), where the two joint angles are denoted q_1, q_2 and the fixed link lengths are denoted a_1, a_2. The discrete state and CPP are identical to those in Section 4; the continuous state in this case is $\xi = (\mathbf{q}, \mathbf{r})$, where $\mathbf{q} = (q_1, q_2)$ is the manipulator *configuration* and $\mathbf{r} = \dot{\mathbf{q}}$. Owing to servomotors, it is often appropriate to model manipulator motion by a linear system (Spong et al., 2006)

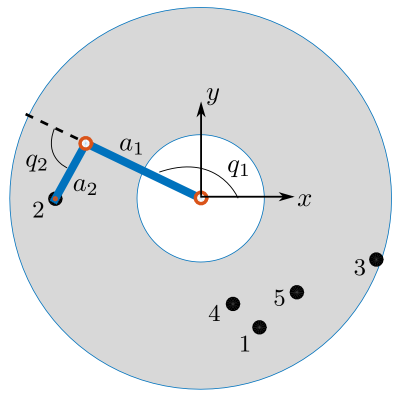


Fig. 3. The two-link planar manipulator considered in Section 5. The shaded region indicates the reachable workspace of the end-effector. The black-colored dots indicate the POIs.

$$\dot{\mathbf{q}}(t) = \mathbf{r}(t), \qquad \dot{\mathbf{r}}(t) = u(t),$$

where u is the control input. The cost of a trajectory is weighted control effort, i.e. $L(\xi, u) := u^{\mathrm{T}} R u$, where $R \in \mathbb{R}^{2\times 2}$ is a prespecified positive-definite weight matrix. The manipulator's reachable workspace is the annulus defined by concentric planar circles of radii $a_1 + a_2$ and $\max(a_1 - a_2, 0)$, as shown in Fig. 3. The kinematic relation between the configuration $\mathbf{q}$ and the end-effector position (x, y) is (Spong et al., 2006)

$$\begin{bmatrix} x \\ y \end{bmatrix} = T\mathbf{q} := \begin{bmatrix} a_1 \cos q_1 + a_2 \cos(q_1 + q_2) \\ a_1 \sin q_1 + a_2 \sin(q_1 + q_2) \end{bmatrix}, \tag{9}$$

assuming that the origin of the Cartesian coordinate axes system coincides with the base of the manipulator, as in Fig. 3. This kinematic relation does not have a unique inverse (Spong et al., 2006). In particular, for any given position (x, y) of the end-effector in the plane, there are two configurations $\mathbf{q}_1, \mathbf{q}_2$ that satisfy $T\mathbf{q}_1 = T\mathbf{q}_2 = [x \;\; y]^{\mathrm{T}}$. This situation is not unique to the two-link manipulator considered here; indeed in *kinematically redundant manipulators*, where the number of degrees of freedom exceeds the dimension of the desired workspace, every end-effector position can be achieved by an infinite number of configurations. In the light of this observation, we note that the mapping Φ_v between CPP states and continuous states is defined by $\Phi_v := T^{-1}(x_{o_{v,\mathrm{v}}}, y_{o_{v,\mathrm{v}}})$, where $T^{-1}(x, y)$ denotes, for the two-link manipulator, the set $\{\mathbf{q}_1, \mathbf{q}_2\}$ of configurations that satisfy the kinematic relation (9). Because the sets Φ_v are finite, we can solve the OEC problem by enumerating all possible combinations of intermediate configurations in conjunction with an LQR control law for finite-duration transfer between successive intermediate configurations. We set this transfer duration as proportional to the distance

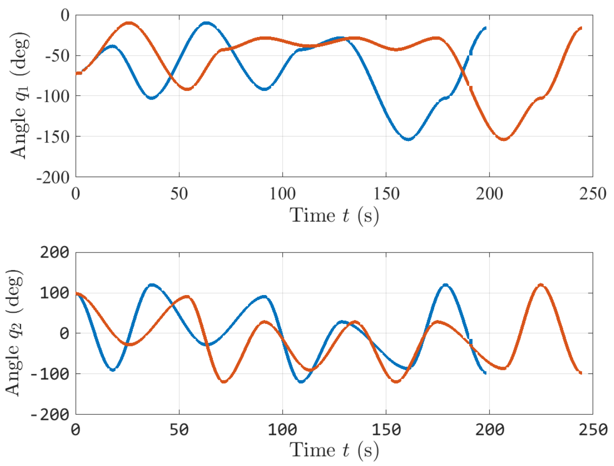


Fig. 4. Evolution of $\mathbf{q}$ over time resulting from the optimal executive control for each of the two plans shown in Table 2. "Standard" plan in blue, and proposed plan in orange.

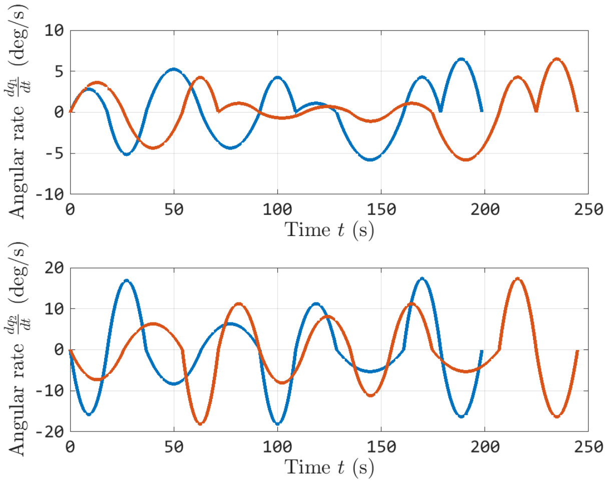


Fig. 5. Evolution of $\mathbf{r}$ over time resulting from the optimal executive control for each of the two plans shown in Table 2. "Standard" plan in blue, and proposed plan in orange.

between configurations. For numerical simulation, we choose $a_1 = 1$ unit, $a_2 = 0.5$ units, $R = \text{diag}(10, 1)$, and we set the POI locations as indicated in Fig. 3.

Table 2 shows a comparison of the plans obtained for this instance of Problem 1 by the "standard", hierarchically separated approach with the proposed approach with $H = 1$. Each row in Table 2 indicates the POI location of the end-effector and each piece of cargo. As expected, the "standard" plan is the same as in Table 1. For this application, however, the plan resulting from the proposed approach has two additional transitions. The cost of the "standard" plan is 0.2909 units, whereas that of the plan found by the proposed approach is 0.2366 units, which represents a 19% reduction in cost. The configuration space trajectories for these two cases are shown in Figs. 4 and 5 respectively.

We remark that it would be difficult for a hierarchically separated hybrid trajectory optimization scheme to find this counterintuitive result – namely, the fact that the plan with lower cost has two additional transitions.

Table 2
Plans obtained for Application Example 1: note that the plan with lower cost in fact has two additional transitions.

Standard plan				Plan by Alg. 2			
End-eff.	o^1	o^2	o^3	End-eff.	o^1	o^2	o^3
5	3	1	2	5	3	1	2
1	3	1	2	3	3	1	2
4	3	4	2	1	3	1	2
3	3	4	2	4	3	4	2
1	1	4	2	3	3	4	2
4	1	4	2	1	1	4	2
3	1	3	2	3	1	4	2
2	1	3	2	4	1	4	2
4	1	3	4	3	1	3	2
5	1	3	4	2	1	3	2
				4	1	3	4
				5	1	3	4

6 Conclusions

A trajectory optimization algorithm is presented for hybrid systems with a natural hierarchy of discrete and continuous subsystems. The proposed algorithm introduces a loose coupling between optimization methods at each of the hierarchical levels, without entirely eliminating the hierarchical problem structure. As the main parameter H becomes large, the proposed algorithm approaches an exhaustive enumeration of solutions to the problem, thereby finding an optimal solution. Furthermore, this algorithm is shown to possess the crucial property of finding solutions with progressively lower costs, with increasing H. This property makes the proposed approach amenable to future "incremental" versions, where the optimal solution is approached by successively closer approximations with increasing H.

References

R. Alur, T. A. Henzinger, G. Lafferriere, and G. J. Pappas. Discrete abstractions of hybrid systems. *Proceedings of the IEEE*, 88(7):971–984, July 2000.

H. Axelsson, Y. Wardi, M. Egerstedt, and E. I. Verriest. Gradient descent approach to optimal mode scheduling in hybrid dynamical systems. *Journal of Optimization Theory and Applications*, 136(2):167–186, 2008.

G. Barles, S. Dharmatti, and M. Ramaswamy. Unbounded viscosity solutions of hybrid control systems. *ESAIM: Control, Optimisation, and Calculus of Variations*, 16: 176–193, 2010.

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*, pages 61 – 70, March 2007.

A. Bemporad and N. Giorgetti. Logic-based solution methods for optimal control of hybrid systems. *IEEE Transactions on Automatic Control*, 51(6):963–976, 2006.

D. P. Bertsekas. *Dynamic Programming and Optimal Control.* Athena Scientific, Belmont, MA, 2000.

M. S. Branicky. Multiple lyapunov functions and other analysis tools for switched and hybrid systems. *IEEE Transactions on Automatic Control*, 43(4):475–482, 1998.

C. G. Cassandras, D. L. Pepyne, and Y. Wardi. Optimal control of a class of hybrid systems. *IEEE Transactions on Automatic Control*, 46(3):398–415, March 2001.

R. V. Cowlagi. Hierarchical hybrid control with classical planning and trajectory optimization. In *Proceedings of the 5th IFAC Conference on Analysis and Design of Hybrid Systems*, pages 175–180, Atlanta, GA, October 14 – 16 2015.

R. V. Cowlagi and P. Tsiotras. Hierarchical motion planning with dynamical feasibility guarantees for mobile robotic vehicles. *IEEE Transactions on Robotics*, 28(2):379 – 395, 2012.

X. Ding, Y. Wardi, and M. Egerstedt. Online optimization of switched-mode dynamical systems. *IEEE Transactions on Automatic Control*, 54(9):2266–2271, 2009.

S. Engell and I. Hajunkoski. Optimal operation scheduling, advanced control and their integration. *Computers and Chemical Engineering*, 47:121–133, 2012.

S. Engell, S. Kowalewski, C. Schulz, and O. Stursberg. Continuous-discrete interactions in chemical processing plants. *Proceedings of the IEEE*, 88(7):1050–1068, July 2000.

M. Garavello and B. Piccoli. Hybrid necessary principle. *SIAM Journal on Control and Optimization*, 43(5):1867–1887, 2005.

D. Garg, M. Patterson, W. W. Hager, A. V. Rao, D. A. Benson, and G. T. Huntington. A unified framework for the numerical solution of optimal control problems using pseudospectral methods. *Automatica*, 46:1843–1851, 2010.

R. Goebel, R. G. Sanfelice, and A. R. Teel. *Hybrid Dynamical Systems: Modeling, Stability, and Robustness.* Princeton University Press, Princeton, NJ, USA, 2012.

W. M. Haddad and V. Chellaboina. *Nonlinear Dynamical Systems and Control: A Lyapunov-based Approach.* Princeton University Press, Princeton, NJ, USA, 2008.

W. M. Haddad, V. Chellaboina, and S. G. Nersesov. *Impulsive and Hybrid Dynamical Systems: Stability, Dissipativity, and Control.* Princeton University Press, 2006.

S. Hedlund and A. Rantzer. Convex dynamic programming for hybrid systems. *IEEE Transactions on Automatic Control*, 47:1536–1540, 2002.

M. Kamgarpour and C. Tomlin. On optimal control of non-autonomous switched systems with a fixed mode sequence. *Automatica*, 48:1177–1181, 2012.

J.-H. Lee, O. Cheong, W.-C. Kwon, S. Y. Shin, and K.-Y. Chwa. Approximation of curvature-constrained shortest paths through a sequence of points. In M. A. Paterson, editor, *Algorithms – ESA 2000, 8th Annual European Symposium*, pages 314–325. Springer-Verlag, Saarburcken, Germany, Sep. 5 – 8 2000.

D. Liberzon. *Switching in Systems and Control.* Birkhäuser, Boston, MA, 2003.

A. Pakniyat and P. E. Caines. Time optimal hybrid minimum principle and the gear changing problem for electric vehicles. In *Proceedings of the 5th IFAC Conference on Analysis and Design of Hybrid Systems*, pages 187–192, Atlanta, GA, USA, October 14–16 2015a.

A. Pakniyat and P. E. Caines. On the relation between the hybrid minimum principle and hybrid dynamic programming: A linear quadratic example. In *Proceedings of the 5th IFAC Conference on Analysis and Design of Hybrid Systems*, pages 169–174, Atlanta, GA, USA, October 14–16 2015b.

B. Piccoli. Hybrid systems and optimal control. In *Proceedings of the 37$^{\text{th}}$ IEEE Conference on Decision and Control*, Tampa, FL, USA, Dec. 1998.

E. Rippel, A. Bar-Gill, and N. Shimkin. Fast graph-search algorithms for general aviation flight trajectory generation. *Journal of Guidance, Control, and Dynamics*, 28(4): 801–811, July-August 2005.

M. Rungger and O. Stursberg. A numerical method for hybrid optimal control based on dynamic programming. *Nonlinear Analysis: Hybrid Systems*, 5:254–274, 2011.

S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach.* Pearson Education Inc., NJ, USA, 2003.

S. Sager. Reformulations and algorithms for the optimization of switching decisions in nonlinear optimal control. *Journal of Process Control*, 19:1238–1247, 2009.

R. G. Sanfelice. On the existence of control lyapunov functions and state-feedback laws for hybrid systems. *IEEE Transactions on Automatic Control*, 58:3242–3248, 2013.

M. S. Shaikh and P. E. Caines. On the hybrid optimal control problem: Theory and algorithms. *IEEE Transactions on Automatic Control*, 52(9):1587–1603, 2007.

M. W. Spong, S. Hutchinson, and M. Vidyasagar. *Robot Modeling and Control.* John Wiley & Sons, New York, NY, USA, 1st edition, 2006.

H. Sussmann. A maximum principle for hybrid optimal control problems. In *Proc. 38th IEEE Conf. Decision & Control*, Phoenix, AZ, USA, 1999.

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

X. Xu and P. Antsaklis. Optimal control of switched systems via nonlinear optimization based on direct differentiations of value functions. *International Journal of Control*, 75 (8):1406–1426, 2002.

F. Zhu and P. Antsaklis. Optimal control of switched hybrid systems: A brief survey. Technical Report ISIS-2013-007, Interdisciplinary Studies in Intelligent Systems, University of Notre Dame, Notre Dame, IN, USA, July 2013.