Kinematic Feasibility Guarantees in Geometric Path Planning using History-based Transition Costs over Cell Decompositions

Raghvendra V. Cowlagi and Panagiotis Tsiotras

***Abstract*— The hierarchical decomposition of motion planning tasks into geometric path planning, followed by kinodynamic motion planning is useful for designing efficient algorithms, but it entails the possibility of inconsistency between the two layers of planning. In an earlier paper, we proposed a general framework, based on rectangular cell decompositions, for incorporating information about the kinodynamic behavior of the vehicle in the geometric planning layer itself. In this paper, we use this framework to design a geometric path planning scheme which *simultaneously* finds an obstacle-free channel of cells from the initial point to the goal, as well as a vehicle state trajectory lying within that channel.**

I. Introduction

The tasks of motion planning and control rank among the indispensable requirements for achieving autonomy for mobile vehicles. Surveys of path planning and motion planning algorithms for mobile vehicles are available in [1]–[3]. Fundamentally, the motion planning problem – namely, that of finding an obstacle-free trajectory from a given initial point to a given destination in the environment – is a single optimization problem, but due to the lack of numerically efficient algorithms for its real-time solution, it is usually decomposed and solved over two layers of hierarchy. The higher – and more abstract – layer is the geometric path planning layer, which is concerned with obstacle avoidance. This layer produces a geometric, obstacle-free *path* from the initial point to the destination. The lower layer accounts for the kinematic and dynamic constraints (abbreviated as *kinodynamic* constraints [4]) that real vehicles must obey, and it involves smoothening of the geometric curve found by the path planner, and then imposing a suitable time parametrization along this curve to obtain a reference *trajectory*.

A widely used class of geometric path planning methods is the class of methods based on rectangular cell decompositions. These methods partition the obstacle-free configuration space into convex, non-overlapping regions, called cells, and then search the associated topological graph for a sequence of adjacent cells from the initial point to the goal [1, Ch. 5 and 6]. Multiresolution schemes that use cell decompositions of varying fine/coarse resolution are computationally efficient, and examples of such schemes include the widely used quadtree method [5], [6], and the wavelet-based cell decomposition schemes in [7]–[9].The result of a geometric path planning algorithm based on a cell decomposition is a finite sequence of cells leading from the initial point to the destination. The task of the lower level planner is then to compute a trajectory that lies completely within this channel.

Apart from the obvious lack of optimality, such a hierarchical approach to the motion planning problem may lead to kinodynamically infeasible paths, since the geometric path planner has no prior knowledge of the kinodynamic constraints of the vehicle. Therefore, a fundamental requirement of hierarchical motion planning algorithms is a guarantee of “compatibility” between the two levels of planning, i.e., a guarantee that the geometric path planning layer never produces a path that may be infeasible for the vehicle to follow. To provide such a guarantee, it is necessary to characterize the kinodynamic feasibility of traversal of such channels of cells. To provide such a guarantee, it is necessary to incorporate some information about the characteristics of feasible trajectories of the vehicle in the geometric path planning layer itself. To this end, in [10] we proposed a framework for path planning based on cell decompositions, which uses such information as a transition cost function on an appropriately constructed graph. A key ingredient of the approach in [10] was the availability of a consistent manner for ranking “good” and “bad” sequences of cells. In [11], we have proposed such an algorithm that ranks cell sequences based on the existence of feasible paths within the given sequence. In this paper, we use the results of [11], in conjunction with the extended graph search algorithm of [10] to obtain a generic path planning scheme which searches for a channel of cells from the initial point to the goal, as well as for a feasible path lying within that channel simultaneously, thus guaranteeing (by construction) that the channel will always contain a feasible path.

Our work is related to the work of Refs. [12]–[14], with some critical differences which we highlight here. References [12]–[14] consider a triangular decomposition of the environment and use controllers (designed in [12]) for the vehicle model that either transfer the vehicle from one cell to another, or confine the vehicle within a cell. For vehicle models that are *completely controllable in the presence of obstacles*, these controllers can guarantee transition from a given cell to an adjacent cell, without intersecting any other cell, from *every* initial state of the vehicle. Consequently, the geometric path planning algorithm is free to plan *any* path (compatible with higher level logic specifications) on the topological graph associated with the cell decomposition, in light of the guarantee that all cell transitions are feasible for the vehicle.

When the complete controllability assumption is violated,

R. V. Cowlagi is a graduate student at the School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA, Email: rcowlagi@gatech.edu

P. Tsiotras is with the School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA, Email: tsiotras@gatech.edu

the central tenet of the motion planning schemes presented in [12]–[14] is no longer valid: arbitrary sequences of cell transitions cannot *in principle* be guaranteed from arbitrary initial states. The simplest example of a vehicle kinematical model that violates the complete controllability assumption is the Dubins car model. For any given sequence of cell transitions, there may exist a set of initial states of the vehicle from which it is impossible for the vehicle to execute that sequence of cell transitions.

The Dubins car kinematical model is given by

$$\dot{x}(t) = u_1 \cos\theta(t),$$
$$\dot{y}(t) = u_1 \sin\theta(t),$$
$$\dot{\theta}(t) = u_2(t),$$

where x, y, and θ are, respectively, the position coordinates and the orientation of the vehicle with respect to a pre-specified inertial axes system; $u_1 = 1$ is the forward speed of the vehicle; and u_2 is the steering control input. The set of admissible controls $\mathcal{U}_T$ is the set of piecewise continuous functions defined on the interval $[0, T]$ that take values in $[-1/r, 1/r]$, for a pre-specified $r > 0$. The set of kinematically feasible paths for the Dubins car is the set of continuously differentiable paths with curvature at most r^{-1}.

II. Geometric Path Planning in Rectangular Cell Decompositions

A. Basic Framework

We consider a uniform decomposition $\mathcal{C}_d$ of the environment $\mathcal{W}$, consisting of N cells, such that every cell in $\mathcal{C}_d$ is a square of size d. A cell $c(i) \in \mathcal{C}_d$ is identified by the location (x_i, y_i) of its center in some pre-specified set of Cartesian axes. We may then construct a graph $\mathcal{G} \stackrel{\text{def}}{=} (V, E)$, such that each element in the set of nodes V corresponds to a unique, obstacle-free cell. We label the nodes as $1, 2, \ldots, N$. Two nodes are *adjacent* if the corresponding cells are geometrically adjacent[1]. The edge set $E \subseteq V \times V$ consists of all pairs (i, j), $i, j \in V$ with nodes i and j adjacent.

Consider a non-negative edge cost function $g : E \to \mathbb{R}_+$ that assigns to each pair of adjacent nodes in $\mathcal{G}$ a non-negative number (the cost of transition between the two nodes defining the edge). For given initial and terminal nodes i_{S}, $i_{\mathrm{G}} \in V$, an *admissible path* $\pi \stackrel{\text{def}}{=} (j_0^\pi, j_1^\pi, \ldots, j_P^\pi)$ in $\mathcal{G}$ is such that $j_k^\pi \in V$, $(j_{k-1}^\pi, j_k^\pi) \in E$, $k = 1, \ldots, P$, with $j_0^\pi = i_{\mathrm{S}}$, $j_P^\pi = i_{\mathrm{G}}$, and $j_p^\pi \neq j_r^\pi$, for $p, r \in \{0, \ldots, P\}$, with $p \neq r$. Geometric path planning algorithms based on cell decomposition typically attempt to solve the following problem, for a suitably defined cost function g.

Problem 1: Let the cost of an admissible path π be

$$J(\pi) = \sum_{k=1}^{P} g((j_{k-1}^\pi, j_k^\pi)). \tag{1}$$

Find an admissible path π^* in $\mathcal{G}$ such that $J(\pi^*) \leq J(\pi)$ for every admissible path π in $\mathcal{G}$.

[1]We consider 4-connectivity for this work, that is, cells that have two vertices in common are said to be adjacent.

In [10], we demonstrated, using a counter-example, that transition costs defined on E cannot encode information about path curvature, and consequently, there may not exist any function g for which the solution to Problem 1 would result in a channel of cells *guaranteed* to contain a feasible path (for a given upper bound on the curvature of the path). As a remedy to this problem, we proposed the use of transition costs defined $k-$tuples of nodes (i.e. histories of previous transitions), for some fixed $k > 2$, such that the elements of each $k-$tuple are pairwise adjacent. The algorithm in [10] works on a lifted graph $\mathcal{G}_H = (V_H, E_H)$, with the set of nodes defined as $V_H \stackrel{\text{def}}{=} \{(i_0, i_1, \ldots, i_H) : (i_{k-1}, i_k) \in E,\ k = 1, \ldots, H,\ \ i_p \neq i_r,\ \text{ for } p, r \in \{0, \ldots, H\}, \text{ with } p \neq r\}$, where H is a positive integer. An element $I \in \mathcal{V}_H$ is defined to be adjacent to $J \in \mathcal{V}_H$ if $(J^{(H+1)}, I^{(H+1)}) \in \mathcal{E}$, $J^{(k)} = I^{(k-1)}$, for every $k = 2, \ldots, H+1$, and $J^{(1)} \neq I^{(H+1)}$, where $I^{(k)}$ denotes the k^{th} element of the $(H+1)-$tuple I. The edge set E_H is then the set of all pairs (J, I), such that I is adjacent to J.

For given initial and terminal nodes i_{S}, $i_{\mathrm{G}} \in V$, an admissible path $\Pi \stackrel{\text{def}}{=} (J_0^\Pi, J_1^\Pi, \ldots, J_Q^\Pi)$ in $\mathcal{G}_H$ is such that $J_k^\Pi \in V_H$, $(J_{k-1}^\Pi, J_k^\Pi) \in E_H$, $k = 1, \ldots, Q$, and $J_0^{\Pi,(1)} = i_{\mathrm{S}}$, $J_Q^{\Pi,(H+1)} = i_{\mathrm{G}}$. Note that every admissible path $\Pi \stackrel{\text{def}}{=} (J_0^\Pi, J_1^\Pi, \ldots, J_Q^\Pi)$ in $\mathcal{G}_H$ uniquely corresponds to an admissible path $\pi \stackrel{\text{def}}{=} (j_0^\pi, j_1^\pi, \ldots, j_P^\pi)$ in $\mathcal{G}$, with $P = Q + H$ and $J_k^{\Pi,(1)} = j_k^\pi$, for $k = 0, 1, \ldots, P - H - 1$, and $J_Q = (j_{P-H}, \ldots, j_P)$.

For every pair of adjacent nodes in $\mathcal{G}_H$ we define a non-negative cost function $g_H : E_H \to \mathbb{R}_+$ that assigns to each pair of adjacent nodes in $\mathcal{G}_H$ a non-negative number. Consider now the following shortest path problem on $\mathcal{G}_H$.

Problem 2: Let the cost of an admissible path $\Pi = (J_0, J_1, \ldots, J_Q)$ in $\mathcal{G}_H$ be

$$\mathcal{J}_H(\Pi) = \sum_{k=1}^{Q} g_H \left(J_{k-1}^\Pi, J_k^\Pi\right). \tag{2}$$

Find an admissible path Π^* such that $\mathcal{J}_H(\Pi^*) \leq \mathcal{J}_H(\Pi)$ for every admissible path Π in $\mathcal{G}_H$.

In [10], we held that a geometric path planner that seeks to solve Problem 2 can be designed to incorporate information about the feasibility of traversal of the channel of cells that it finds. In Section II-B, we formulate such a geometric planning algorithm.

We label the elements of E_H by natural numbers, and introduce a bijection $\phi : E_H \to \{1, 2, \ldots, |E_H|\}$ that uniquely associates each element in E_H with a natural number as follows. Let $(I, J) \in E_H$, $I, J \in V_H$ be an edge in E_H. We define the *tile* associated with (J, I) as the sequence of cells associated with $I^{(1)}, \ldots, J^{(H+1)}$, and we label this tile by $\alpha = \phi(I, J)$. Alternatively, we denote the edge associated with a particular tile labeled α by $(I^\alpha, J^\alpha) \stackrel{\text{def}}{=} \phi^{-1}(\alpha)$. With each tile[2] α, we may associate a vector $\eta^\alpha \in \mathbb{R}^2$ such that

[2]In a minor abuse of notation, we denote a tile labeled α by the same symbol α.

η^α is normal to the segment $\partial c\left(I^{\alpha,(1)}\right) \cap \partial c\left(I^{\alpha,(2)}\right)$ and points inside the cell $c\left(I^{\alpha,(2)}\right)$.

We may now define a *tile motion planning algorithm* TILEPLAN as any algorithm which determines if a given tile may be feasibly traversed by the vehicle from a *specific* initial condition. More precisely, we specify TILEPLAN as an algorithm which:

1) takes as input a tile α and a vehicle state $\xi_0^\alpha = (x_0, y_0, \theta_0) \in \left(\partial c\left(I^{\alpha,(1)}\right) \cap \partial c\left(I^{\alpha,(2)}\right)\right) \times [-\pi, \pi]$ satisfying

$$\begin{bmatrix} \cos\theta_0 & \sin\theta_0 \end{bmatrix} \eta^\alpha \quad \geq \quad 0, \tag{3}$$

2) determines if there exist
   a) numbers $T_1^\alpha, T_2^\alpha, \ldots, T_H^\alpha$ satisfying $T_1^\alpha \leq T_2^\alpha \leq \ldots \leq T_H^\alpha$, and
   b) an admissible control $u_2^\alpha \in \mathcal{U}_{T_h^\alpha}$,

   such that the resultant state trajectory $\xi\left(t; \xi_0^\alpha, u^\alpha\right) = (x(t), y(t), \theta(t))$ satisfies

$$\Big(x(t), y(t)\Big) \quad \in \quad \begin{cases} c\left(I^{\alpha,(2)}\right), & t \in [0, T_1^\alpha], \\ \vdots, & \vdots \\ c\left(I^{\alpha,(H+1)}\right), & t \in \left[T_{H-1}^\alpha, T_H^\alpha\right], \\ c\left(J^{\alpha,(H+1)}\right), & t = T_H^\alpha, \end{cases}$$

3) returns failure if $T_1^\alpha, T_2^\alpha, \ldots, T_H^\alpha$ and u^α are found to not exist, or else returns T_1^α, the state $\xi\left(T_1^\alpha; \xi_0^\alpha, u_2^\alpha\right)$ and the control u_2^α if $J_1^{\alpha,(H+1)} \neq i_{\mathrm{G}}$, or returns T_H^α and the control u_2^α if $J_1^{\alpha,(H+1)} = i_{\mathrm{G}}$.

In summary, TILEPLAN accepts a tile $\alpha \in V_H$ and an initial state $\xi_{\mathrm{i}} \in \left(\partial c\left(I^{\alpha,(1)}\right) \cap \partial c\left(I^{\alpha,(2)}\right)\right) \times [-\pi, \pi] \times \Psi$ as inputs and returns outputs $\mathrm{feas}, T, \xi_{\mathrm{f}}$, and u (as described in requirement 3 above), where feas is a boolean output indicating success or failure of TILEPLAN; T is the duration of traversal; u is the control input that enables traversal of the tile α; and ξ_{f} is the state at time T. We indicate the execution of TILEPLAN with inputs α and ξ by TILEPLAN(α, ξ).

B. The Proposed Algorithm

Suppose that a tile motion planning algorithm which satisfies the above requirements is available. We are now ready to describe the proposed geometric path planning algorithm, as a modified label correcting algorithm (cf. [15], [16]), which searches for a path in $\mathcal{G}_H$. We introduce three functions in addition to the label and backpointer functions of the standard label correcting algorithm: a function $\Xi : V_H \to \mathbb{R}^2 \times \mathbb{S}^1$ that associates a vehicle state with each node in V_H, a function $\Theta : V_H \to \mathbb{R}_+$ that associates a time of traversal with each node in V_H, and a function $\Upsilon : V_H \to \mathcal{U}$ that associates a control input with each node in V_H. The proposed algorithm is then as follows.

---

procedure Initialize()

$\mathcal{P} \leftarrow \{I_{\mathrm{S}}\}$, $d(I_{\mathrm{S}}) \leftarrow 0$;
for all $J \in V_H \backslash \{I_{\mathrm{S}}\}$ **do**
  $d(J) = \infty$;
$I \leftarrow I_{\mathrm{S}}$
$\Xi(I) = \xi_0$, $\Theta(I) = 0$

procedure Main()

Initialize();
while $\mathcal{P} \neq \varnothing$ **do**
  $\mathcal{P} \leftarrow \mathcal{P} \backslash \{I\}$;
  for all $J \in V_H$ such that $(I, J) \in E_H$ **do**
    $\alpha \equiv \{I^{(1)}, \ldots, I^{(H+1)}, J^{(H+1)}\}$
    $(\mathrm{feas}, T, \xi, u) \leftarrow$ TILEPLAN$(\alpha, \Xi(I))$
    if $d(I) + g_H(I, J) < d(J)$ and $\mathrm{feas} = 1$ **then**
      $d(j) \leftarrow d(i) + g(i, j)$;
      $b(j) \leftarrow i$; $\Xi(J) \leftarrow \xi$;
      $\Theta(J) \leftarrow T$; $\Upsilon(J) \leftarrow u_2$;
      $\mathcal{P} \leftarrow \mathcal{P} \cup \{J\}$;
  Choose $I \in \mathcal{P}$;

The proposed algorithm produces a path $\Pi^* = (J_0^{\Pi^*}, J_1^{\Pi^*}, \ldots, J_P^{\Pi^*})$ where $J_0^{\Pi^*,(1)} = i_{\mathrm{S}}$ and $J_P^{\Pi^*,(H+1)} = i_{\mathrm{G}}$. As noted previously, Π^* corresponds to a path π^* in $\mathcal{G}$. The control input for the vehicle which traverses the channel of cells associated with π^* is given by

$$u_2(t) \stackrel{\mathrm{def}}{=} \Upsilon(J_k^{\Pi^*}),\ t \in \left[\sum_{\ell=1}^{k-1} \Theta(J_\ell^{\Pi^*}), \sum_{\ell=1}^{k-1} \Theta(J_\ell^{\Pi^*}) + \Theta(J_k^{\Pi^*})\right),$$

for each $k = 1, \ldots, P$. Thus, by construction, the proposed algorithm satisfies the requirement of hierarchical consistency.

III. TILE MOTION PLANNING

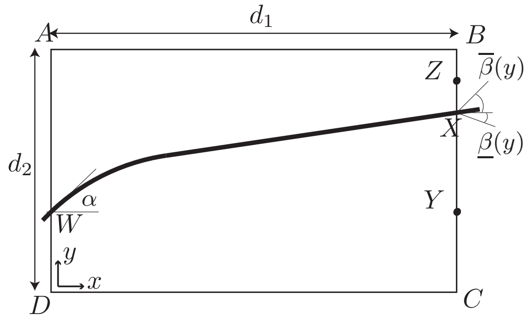
The description of the proposed motion planning algorithm in Section II-B relied heavily upon the tile motion planning TILEPLAN. We specified what TILEPLAN was *required* to do, but did not specify how exactly TILEPLAN would accomplish those requirements. In this section, we shed some light on this issue. Consider the following problem.

Problem 3: Let $\bar{\mathcal{R}}^C$ be a rectangular channel[3], and let W be a point on any of the three edges of R_1 which does not intersect R_2. Let $\alpha \in [-\pi, \pi]$ be a specified angle. For any set of positive real numbers $r_n > 0$, $n = 1, \ldots, C$, determine if there exists a path Π such that:

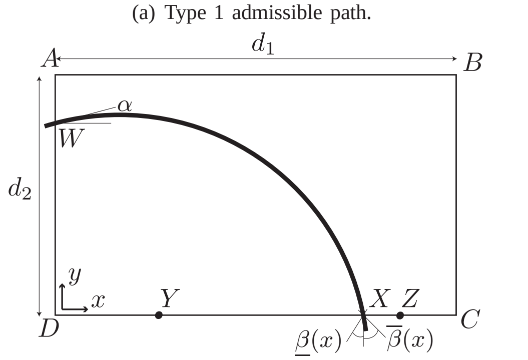
1) The initial configuration of Π is (W, α),
2) The final configuration of the path Π lies in a set defined by the Cartesian product of a specified edge of the rectangle R_C (different from the edge coinciding with rectangle R_{C-1}) with a specified set of allowable terminal tangent angles,
3) The path Π does not leave $\bar{\mathcal{R}}^C$, i.e. $(x(s), y(s)) \in \cup_{n=1}^C R_n$ for every $s \in [0, 1]$,
4) The curvature of Π at any point in rectangle R_n is at most r_n^{-1}, for every $n = 1, \ldots, C$.

To the solve Problem 3, we consider two basic problems defined on a single rectangle. These problems, which we denote Problem T1 (respectively, Problem T2) are concerned

[3] By a *rectangular channel*, we mean a sequence of disjoint rectangles of arbitrary dimensions such that every pair of successive rectangles has a common edge.



(a) Type 1 admissible path.



(b) Type 2 admissible path.

Fig. 1. Setup for the basic problems.

with the curvature-bounded traversal across a pair of parallel edges (respectively, adjacent edges) of the rectangle with a given initial condition, while satisfying an additional constraint: the terminal orientation must lie in a pre-specified set $\left[\underline{\beta}, \overline{\beta}\right]$. In [11], [17], we propose numerical solutions to Problems T1 and T2 and propose an algorithm that recursively solves Problems T1 and T2 for individual rectangles of the channel to solve Problem 3. Here, we outline this algorithm using an illustrative example.

Let $\bar{\mathcal{R}}^4 = \{R_n\}_{n=1}^4$ be a rectangular channel with four rectangles, as shown in Fig. 2, and let $r_n > 0$, $n = 1, \ldots, 4$, be given. Let U_n, V_n, $n = 1, \ldots, 4$ and Y_4, Z_4 be points as shown in Fig. 2. Given a prescribed initial entry point W on the segment U_1V_1 and given a prescribed initial tangent angle $\alpha \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$, we wish to determine if there exists a path satisfying the conditions described above. We attach a coordinate axes system to each rectangle R_n with the origin at the point U_n, with the positive x-axis along U_nY_n, and with the positive y-axis along U_nV_n. The dimensions of each rectangle along the x and y axes are denoted, respectively, as $d_{n,1}$ and $d_{n,2}$. For each rectangle R_n, the solution of Problem T1 (or Problem T2, as applicable) provides angles $\underline{\alpha}(q)$ and $\overline{\alpha}(q)$ for any $q \in [0, d_{n,2}]$, such that the curvature-bounded traversal of R_n is possible if the initial orientation at point $Q = (0, q)$ lies in the interval $[\underline{\alpha}(q), \overline{\alpha}(q)]$.

We define functions $\overline{\alpha}_5(q) \overset{\text{def}}{=} \frac{\pi}{2}, \underline{\alpha}_5(q) \overset{\text{def}}{=} -\frac{\pi}{2}, q \in [0, d_{4,2}]$, and we note that the last rectangle R_4 involves traversal across parallel edges. Next, we note that the total number of reflections occurring in the transformations required for R_4 and (the fictitious rectangle) R_5 is zero, and

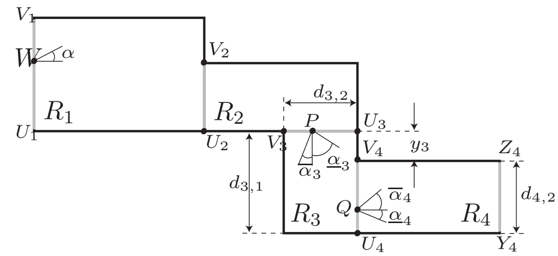


Fig. 2. Illustrative example for the proposed algorithm.

we set $\overline{\beta}_4 = \overline{\alpha}_5 = \frac{\pi}{2}, \underline{\beta}_4 = \underline{\alpha}_5 = -\frac{\pi}{2}$. We solve Problem T1 for each point $Q = (0, q)$, $q \in [0, d_{4,2}]$ on the segment U_4V_4, and we obtain the values taken by the functions $\underline{\alpha}_4(q)$ and $\overline{\alpha}_4(q)$.

Rectangle R_3 involves traversal across adjacent edges, and the entry and exit segments of R_3 may be aligned with segments AD and DC of Fig. 1 after a reflection about an axis parallel to the segment U_4V_4, followed by a rotation through $\frac{\pi}{2}$ rad. Since the total number of reflections occurring in the transformations required for R_3 and R_4 is one (odd), we set

$$\begin{aligned} \overline{\beta}_3(q) &= -\underline{\alpha}_4(y_3 - (q - v_4)), \\ \underline{\beta}_3(q) &= -\overline{\alpha}_4(z_3 - (q - u_4)), \quad q \in [y_3, z_3], \end{aligned}$$

where $z_3 = d_{3,1}$, $y_3 = \ell(U_3V_4)$, $v_4 = d_{4,2}$, and $u_4 = 0$ (see Fig. 2). We solve Problem T2 for each point $P = (0, p)$, $p \in [0, d_{3,2}]$ on the segment U_3V_3 to obtain values taken by the functions $\underline{\alpha}_3(p)$ and $\overline{\alpha}_3(p)$. Proceeding further similarly, we may obtain the values taken by the functions $\underline{\alpha}_2(q), \overline{\alpha}_2(q)$, $q \in [0, d_{2,2}]$ and by the functions $\underline{\alpha}_1(q), \overline{\alpha}_1(q)$, $q \in [0, d_{1,2}]$. Let the prescribed entry point W have coordinates $(0, w)$ in the coordinate axes attached to R_1. Then there exists a path satisfying the requirements stated in Problem 3 if the prescribed initial tangent angle α satisfies $\alpha \in [\underline{\alpha}_1(w), \overline{\alpha}_1(w)]$.

The recursive analysis outlined above allows us to synthesize paths cell by cell. The synthesis method relies on the fact that the cone analysis described above provides a set of allowable terminal conditions for each cell, i.e., a cone of allowable orientations associated with each point on the exit segment of that cell. A feasible path traversing the remainder of the channel exists from all such terminal conditions. One may then pick any of these terminal conditions and compute a unique path, since the initial condition is already specified.

The result of the synthesis is a path (within *each cell*) which is a concatenation of circular arcs of radius r and straight line segments. For the Dubins vehicle kinematic model, the steering input required for following such a concatenated path is uniquely determined: $u_2 = 1$ along a counter-clockwise circular arc, $u_2 = -1$ along a clockwise circular arc, and $u_2 = 0$ along a straight line segment. Since the synthesis is performed cell-wise, the duration for traversal of each cell (required to be computed TILEPLAN) is easily determined. In summary, the existence analysis and synthesis of curvature-bounded paths outlined here serves as a tile motion planning algorithm for the Dubins vehicle.

IV. Simulation Results and Further Remarks

In this section, we define the cost function g_H as a function that associates with each tile the length of the path returned by TILEPLAN. In particular, for a tile labeled by $\alpha \in \{1, 2, \ldots |E_H|\}$ and an initial condition $\xi_0^\alpha = (x_0, y_0, \theta_0)$ let T_1^α be the duration of traversal returned by TILEPLAN. Then we define

$$g_H(I^\alpha, J^\alpha) = \begin{cases} u_1 \int_0^{T_1^\alpha} \mathrm{d}t = u_1 T_1^\alpha, & J^{\alpha,(H+1)} \neq i_{\mathrm{G}}, \\ u_1 \int_0^{T_H^\alpha} \mathrm{d}t = u_1 T_H^\alpha, & J^{\alpha,(H+1)} = i_{\mathrm{G}}. \end{cases} \quad (4)$$

Figures 3 and 4 show results of simulating the proposed algorithm over simple environments with the cost function (4). Figure 3 shows the solutions to the benchmark example we discussed in [10] to emphasize the fact that cost functions based on edge transitions in $\mathcal{G}$ alone are not sufficient to capture curvature information. In particular, the four results in Fig. 3 show the different channels obtained for different bounds on the curvature, whereas any cost function defined on $\mathcal{E}$ would result in the channel shown in Fig. 3(a). Similarly, Fig. 4 shows the difference between resultant channels when different bounds on the curvature are imposed.

A. Dependence on H

In this section, we highlight an important monotonicity property satisfied by the cost function defined in (4). In what follows, we denote the path in $\mathcal{G}_H$ corresponding to a path π in $\mathcal{G}$ by Π^H, and we denote the path computed in $\mathcal{G}_H$ by the proposed algorithm by $\Pi_H^{*,H}$. The subscript denotes that the proposed algorithm used the lifted graph $\mathcal{G}_H$ in its computations, while the superscript denotes that the path is *represented* in $\mathcal{G}_H$. Thus, $\Pi_H^{*,L}$, $L \in \mathbb{N}$, denotes the path in $\mathcal{G}_L$ corresponding to $\Pi_H^{*,H}$, and both $\Pi_H^{*,L}$ and $\Pi_H^{*,H}$ correspond to the same path in $\mathcal{G}$. On the other hand, $\Pi_L^{*,H}$, $L \in \mathbb{N}$ is the path computed by the proposed algorithm when it uses the lifted graph $\mathcal{G}_L$ for its computations. Note that, in general,the paths in $\mathcal{G}$ corresponding to $\Pi_L^{*,H}$ and $\Pi_H^{*,H}$ are different.

Lemma 1: Let $\pi = \{j_0^\pi, j_1^\pi, \ldots j_P^\pi\}$ be an admissible path in $\mathcal{G}$, and let $\Pi^H = \{J_{0,H}, J_{1,H}, \ldots, J_{P-H,H}\}$ be the corresponding path in $\mathcal{G}_H$, where $J_{m,H} = \left(j_m^\pi, j_{m+1}^\pi, \ldots, j_{m+H}^\pi\right)$, $m = 0, 1, \ldots, P-H$. Let g_H be the cost function defined in (4). Then for every $H > 0$,

$$\mathcal{J}_{H+1}(\Pi^{H+1}) \leq \mathcal{J}_H(\Pi^H). \quad (5)$$

Proof: For a given $H > 0$, the cost $\mathcal{J}_H\left(\Pi^H\right)$ of the path Π^H is computed by executing TILEPLAN for each tile in Π^H, i.e., for each pair $(J_{m,H}, J_{m+1,H})$, $m = 0, 1, \ldots, P-H-1$. By definition of the lifted graph $\mathcal{G}_H$, the edge $(J_{m,H}, J_{m+1,H})$ in $\mathcal{G}_H$ is a node in V_{H+1}, and the tile $(J_{m,H+1}, J_{m+1,H+1})$ in $\mathcal{G}_{H+1}$ is the triplet $(J_{m,H}, J_{m+1,H}, J_{m+2,H})$. Thus, the first node of the tile $(J_{m,H+1}, J_{m+1,H+1})$ in $\mathcal{G}_{H+1}$ is the same as the first node of the tile $(J_{m,H}, J_{m+1,H})$ in $\mathcal{G}_H$.

Let TILEPLAN(H) and TILEPLAN$(H+1)$ denote, respectively, the execution of TILEPLAN on tiles in $\mathcal{G}_H$ and on tiles in $\mathcal{G}_{H+1}$. Let ξ be the initial state provided as an input to both TILEPLAN(H) and TILEPLAN$(H+1)$. Suppose TILEPLAN(H) computes $u_{m,H}$ as the control required to traverse the cell $c\left(J_{m,H}^{(2)}\right)$ of the tile $(J_{m,H}, J_{m+1,H})$, and TILEPLAN$(H+1)$ computes $u_{m,H+1}$ as the control required to traverse the cell $c\left(J_{m,H+1}^{(2)}\right)$ of the tile $(J_{m,H+1}, J_{m+1,H+1})$. Correspondingly, let $\xi_{m,H}$ and $\xi_{m,H+1}$ be the terminal states, respectively, resulting after the controls $u_{m,H}$ and $u_{m,H+1}$ are applied at ξ.

Since our path synthesis algorithm, based on which TILEPLAN computes the controls required for traversal of the first cell, performs the synthesis cell-wise, and since the first cell of both the tiles under consideration is the same, $u_{m,H+1}$ will differ from $u_{m,H}$ only if there exists no control from the state $\xi_{m,H}$ that would enable traversal through the remainder of the tile $(J_{m,H+1}, J_{m+1,H+1})$, i.e., through the sequence of cells $J_{m,H+1}^{(3)}, \ldots, J_{m,H+1}^{(H+3)}$. As a consequence, TILEPLAN(H) would return failure when it processes the tile $(J_{m+1,H}, J_{m+2,H})$, since the first $H+1$ cells of this tile are precisely $J_{m,H+1}^{(3)}, \ldots, J_{m,H+1}^{(H+3)}$. In other words, for every tile $(J_{m,H}, J_{m+1,H})$, $m = 0, 1, \ldots, P-H$, if the cost returned by TILEPLAN(H) is different from the cost returned by TILEPLAN$(H+1)$ for the tile $(J_{m,H+1}, J_{m+1,H+1})$, then TILEPLAN(H) must return an infinite cost (failure) for the tile $(J_{m+1,H}, J_{m+2,H})$. The required result (5) then follows. ■

We can now show an important monotonicity property about the optimality of paths found by the proposed algorithm. Let $\bar{P}$ be the maximum number of nodes that may occur in any path from i_{S} to i_{G} with no cycles.

Proposition 2: Let H be a positive integer, and let $\Pi_H^{*,H}$ be in the path in $\mathcal{G}_H$ resulting from the proposed geometric path planning algorithm. Then $\mathcal{J}_H(\Pi_H^{*,H})$ decreases monotonically with increasing H.

Proof: Let π be an admissible path in $\mathcal{G}$, and let Π^H be the corresponding path in $\mathcal{G}_H$. It follows from Lemma 1 that the sequence of cost functions $\mathcal{J}_H(\Pi^H)$, $H = 0, 1, \ldots, \bar{P}-1$ is monotonically decreasing, i.e.,

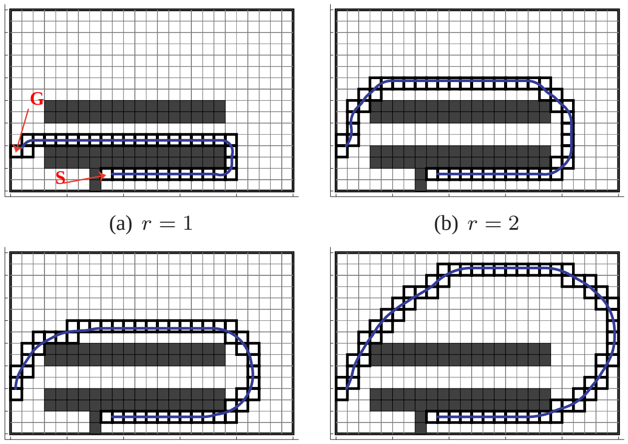
$$\mathcal{J}_{\bar{P}-1}(\Pi^{\bar{P}}) \leq \ldots \leq \mathcal{J}_1(\Pi^1) \leq \mathcal{J}_0(\Pi^0). \quad (6)$$

The proposed algorithm solves Problem 2, and hence $\Pi_H^{*,H}$ minimizes $\mathcal{J}_H$, i.e., $\mathcal{J}_H(\Pi_H^{*,H}) \leq \mathcal{J}_H(\Pi^H)$. Since Π^H was arbitrary, this inequality holds for $\Pi^H = \Pi_{H-1}^{*,H}$, i.e., $\mathcal{J}_H(\Pi_H^{*,H}) \leq \mathcal{J}_H(\Pi_{H-1}^{*,H})$. Then it follows from (6) that $\mathcal{J}_H(\Pi_H^{*,H}) \leq \mathcal{J}_{H-1}(\Pi_{H-1}^{*,H-1})$, and in general, $\mathcal{J}_{\bar{P}-1}(\Pi_{\bar{P}-1}^{*,\bar{P}-1}) \leq \mathcal{J}_{\bar{P}-2}(\Pi_{\bar{P}-2}^{*,\bar{P}-2}) \leq \cdots \leq \mathcal{J}_0(\Pi_0^{*0})$. ■

An informal interpretation of the above results which is evident in the proof of Lemma 1 is that, as H is increased, the proposed algorithm will erroneously reject fewer paths in $\mathcal{G}$ from i_{S} to i_{G} as infeasible.

V. Conclusions

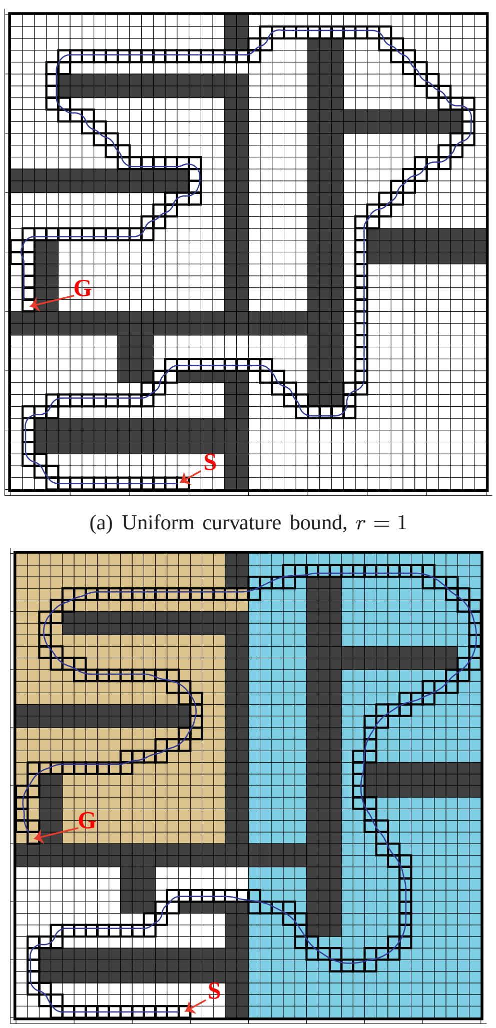
We presented a novel geometric path planning algorithm based on rectangular cell decompositions which simultaneously finds an obstacle-free channel of cells from the initial



(a) $r = 1$ (b) $r = 2$

(c) $r = 3.5$ (d) $r = 4.5$

Fig. 3. Making U-turns with different curvature bounds. The curvature bound in each case is r^{-1}. Here, $H = 3$.



(a) Uniform curvature bound, $r = 1$

(b) Patch-wise varying curvature bound, $r_{\text{white}} = 1$, $r_{\text{yellow}} = 2.25$, $r_{\text{blue}} = 3.5$.

Fig. 4. Simulation result: simple maze-like environment. The curvature bound in each case is r^{-1}. Here, $H = 3$.

point to the goal, as well as a continuously differentiable path with a pre-specified upper bound on its curvature. We point out that although we have focused on a specific example of a vehicle model (the Dubins car), the execution of the geometric planner itself is not restricted by the characteristics of this choice. In fact, we have proposed a scheme where the details of motion planning for the specific vehicle at hand are left to the so-called tile motion planning algorithm TILEPLAN, and the geometric planner uses information provided by TILEPLAN in a manner independent of the particular tile planning algorithm.

Acknowledgement: The results in this work have been supported by ARO (award no. W911NF-05-1-0331) and NASA (award no. NNX08AB94A).

REFERENCES

[1] J.-C. Latombe, *Robot Motion Planning*. Kluwer Academic Publishers, 1991.

[2] S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.

[3] 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.

[4] B. Donald, P. Xavier, J. Canny, and J. Reif, "Kinodynamic motion planning," *Journal of the Association for Computing Machinery*, vol. 40, no. 5, pp. 1048–1066, November 1993.

[5] S. Kambhampati and L. S. Davis, "Multiresolution path planning for mobile robots," *IEEE Journal of Robotics and Automation*, vol. RA-2, no. 3, pp. 135–45, September 1986.

[6] H. Noborio, T. Naniwa, and S. Arimoto, "A quadtree-based path planning algorithm for a mobile robot," *Journal of Robotic Systems*, vol. 7, no. 4, pp. 555–74, 1990.

[7] P. Tsiotras and E. Bakolas, "A hierarchical on-line path planning scheme using wavelets," in *Proceedings of the European Control Conference*, Kos, Greece, July 2–5 2007, pp. 2806–2812.

[8] R. V. Cowlagi and P. Tsiotras, "Beyond quadtrees: Cell decompositions for path planning using the wavelet transform," in *Proceedings of the 46th IEEE Conference on Decision and Control*, New Orleans, LA, 12–14 Dec. 2007, pp. 1392–1397.

[9] D. Jung, "Hierarchical path planning and control of a small fixed-wing UAV: Theory and experimental validation," Ph.D. dissertation, Georgia Institute of Technology, 2007.

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

[11] ——, "On the existence and synthesis of curvature-bounded paths inside nonuniform rectangular channels," in *Proceedings of the 2010 American Control Conference*, Baltimore, MD, USA, 30 June – 2 July 2010, to appear.

[12] 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, no. 5, pp. 864 – 874, October 2005.

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

[14] G. E. Fainekos, A. Girard, and G. J. Pappas, "Hierarchical synthesis of hybrid controllers from temproal logic specifications," in *Hybrid Systems: Computation and Control*, ser. LNCS 4416, A. Bemporad, A. Bicchi, and G. Buttazzo, Eds., 2007, pp. 203 – 216.

[15] D. P. Bertsekas, *Dynamic Programming and Optimal Control*. Athena Scientific, Belmont, MA, 2000, vol. 1.

[16] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. MIT Press, 2001.

[17] R. V. Cowlagi and P. Tsiotras, "Existence and synthesis of curvature-bounded paths in rectangular channels," Georgia Institute of Technology, Atlanta, GA, Tech. Rep., 2010.