Multiresolution Motion Planning for Autonomous Agents via Wavelet-Based Cell Decompositions

Raghvendra V. Cowlagi, *Member, IEEE*, and Panagiotis Tsiotras, *Senior Member, IEEE*

***Abstract*—We present a path- and motion-planning scheme that is "multiresolution" both in the sense of representing the environment with high accuracy only locally and in the sense of addressing the vehicle kinematic and dynamic constraints only locally. The proposed scheme uses rectangular multiresolution cell decompositions, efficiently generated using the wavelet transform. The wavelet transform is widely used in signal and image processing, with emerging applications in autonomous sensing and perception systems. The proposed motion planner enables the simultaneous use of the wavelet transform in both the perception and in the motion-planning layers of vehicle autonomy, thus potentially reducing online computations. We rigorously prove the completeness of the proposed path-planning scheme, and we provide numerical simulation results to illustrate its efficacy.**

***Index Terms*—Discrete wavelet transforms, dynamics, mobile robots, motion planning, path planning.**

I. Introduction

MOTION planning for autonomous terrestrial and aerial vehicles has been extensively studied [1], [2]. However, important issues, such as dealing with uncertain, partially known, and/or dynamically changing environments, and the satisfaction of vehicle kinematic and dynamic constraints are yet to be thoroughly and satisfactorily addressed, particularly when considering additional constraints stemming from the limited computational resources on-board the vehicle.

In this paper, we present a fast multiresolution motion-planning scheme that guarantees the satisfaction of the vehicle's kinematic and dynamic constraints. To introduce the various interrelated aspects of the proposed scheme, we will use the following terminology: We will use the term *path* to refer to the locus of continuous motion of a point and the term *trajectory* to refer to a path parameterized by time. Depending on the context, we will use the term *path* to also refer to a sequence of successively adjacent vertices in a graph. Finally, we will synonymously use the terms *workspace* and *environment* to refer to a planar region over which the vehicle moves.

Manuscript received May 14, 2011; revised August 25, 2011 and December 21, 2011; accepted March 2, 2012. Date of publication May 7, 2012; date of current version September 12, 2012. This work was supported in part by the NSF under Award CMMI-0856565 and in part by ARO-MURI under Award W911NF-11-1-0046. This paper was recommended by Associate Editor J. Su.

R. V. Cowlagi is with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139-4307 USA (e-mail: rcowlagi@mit.edu).

P. Tsiotras is with the School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0150 USA (e-mail: tsiotras@gatech.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCB.2012.2192268

Multiresolution path planning involves the representation of the vehicle's environment with different levels of accuracy. For example, the popular quadtree method [3]–[5] generates a planar cell decomposition consisting of small cell sizes that accurately capture obstacle boundaries and larger cell sizes that efficiently represent large areas in the free space. Other path-planning schemes that use multiresolution cell decompositions have appeared, for instance, in [6]; in [7] (triangular cells); in [8] (receding horizon path-planning scheme using multiresolution estimates of object locations); in [9] (multiresolution potential field); and in [10] (hierarchy of imaginary encapsulating spheres for collision avoidance).

We consider planar cell decompositions such that the environment is represented with high accuracy (i.e., using small cell sizes) in the agent's immediate vicinity, and with lower accuracy in regions farther away, similar to the multiresolution grids considered in [6] and [11]. Multiresolution cell decompositions are compact data structures that encode large environment maps, and thus enable efficient online path- and motion planning. Furthermore, multiresolution decompositions of the environment naturally capture the graded levels of uncertainty about the environment as functions of the distance from the current location of the agent. In other words, such decompositions encode the notion that the *uncertainty* or *incomplete knowledge* about the environment is higher in regions farther away from the vehicle's current location.

The discrete wavelet transform (DWT) is a mathematical tool widely used in multiresolution signal processing [12], [13]. Applications of the DWT to vision-based navigation and vision-based simultaneous localization and mapping (SLAM) for autonomous vehicles have recently appeared in [14] (appearance-based vision-only SLAM); in [15] and [16] (local feature extraction); and in [17] (stereo image processing). With the plethora of available sensors and in light of the fact that multiple sensors are typically used for autonomous navigation [11], the wavelet transform may soon become the common standard for representing and analyzing signals [18]. In this context, wavelet-based data representation for path-planning problems has been recently addressed in [19] (occupancy grids); in [20] (standardized representation of road-roughness characteristics); in [21] (terrain depiction for pilot situational awareness); and in [22] (image registration for vision-based navigation).

In light of the ubiquitous use of the DWT in signal and image processing and its emerging applications in autonomous sensing and perception, it is natural to investigate the seamless integration of sensing and path planning using multiresolution wavelet analysis. To this end, we propose a path-planning scheme that directly uses a DWT representation

of the environment. Applications of the DWT in multiresolution path-planning schemes have previously appeared, for instance, in [23] and [24] (preliminary implementations of the proposed path-planning scheme); in [25] (path refinement based on successively finer approximations of a terrain map); and in [26] and [27] (multiresolution schemes for vision-based path planning).

H-Cost Motion Planning: Motion-planning schemes often involve a *geometric path planner* that uses an abstract discrete representation (e.g., graphs associated with cell decompositions) of the workspace and deals with the satisfaction of task specifications such as obstacle avoidance. However, the resultant geometric path may be found to be infeasible or unacceptably suboptimal if the vehicle's kinematic and dynamic constraints are ignored. To address this issue, we introduced in [28] a motion-planning approach based on assigning costs, called *H-costs*, to *multiple*-edge transitions in the cell decomposition graph. These H-costs allow the vehicle's kinematic and dynamic constraints to be incorporated in the geometric path planner via the (implicit) construction of a so-called *lifted graph*, which is closely related to the original cell decomposition graph. In this paper, we discuss a multiresolution implementation of this H-cost motion-planning approach, such that the overall scheme is "multiresolution," both in the sense of representing the environment with high accuracy only locally *and* in the sense of considering the vehicle dynamical model for path planning only locally.

In summary, the proposed motion-planning scheme consists of the following main elements: 1) a wavelet-based multiresolution cell decomposition algorithm that creates and modifies a graph that represents the environment (see Section II); 2) a local trajectory generation algorithm called TilePlan that associates H-costs in the aforesaid cell decomposition graph to (implicitly) construct a "partially" lifted graph (see [29] and [30]); and 3) a discrete path planner that finds paths in the "partially" lifted graph, which, in turn, correspond to trajectories that satisfy the vehicle's kinematic and dynamic constraints (see Section IV). The interactions between the various models and methods involved in the proposed scheme are illustrated in Fig. 1; here, hollow arrows indicate the creations and modifications of various models by the methods shown, whereas bold arrows indicate the dependency between the various models and methods shown. For example, the hollow arrow from TilePlan to the "partially" lifted graph model indicates modification of the edge transition costs of the latter.

The main contributions of this paper are as follows. First, we present a multiresolution cell decomposition technique that is completely encoded in the DWT coefficients of the environment map. This approach allows for the development of highly integrated and efficient navigation and path-planning architectures, where the DWT coefficients are used as a common data structure both for scene understanding and motion planning. Second, we demonstrate one such integrated approach by proposing a path-planning scheme based on the aforesaid cell decompositions, and we rigorously prove its completeness. To the best of our knowledge, such proofs of completeness are absent from other similar multiresolution path-planning schemes [6], [11]. Finally, we discuss a method of incorporating vehicle dynamic constraints in the multiresolution path-planning scheme using the H-cost motion-planning approach in [28]. The issue of consistency between the geometric and dynamic planning layers is well known in the robotics community [31]. To date, this problem has been addressed by planning in the (high-dimensional) state space, instead of the workspace, where the obstacles naturally lie. We show that, for mobile robotic applications, planning can be restricted to the low-dimensional workspace. The overall motion-planning scheme is thus an important step in the development of a hierarchically *consistent* autonomous navigation and motion-planning system, i.e., one that guarantees the satisfaction of vehicle kinematic and dynamic constraints while retaining the computational efficiency of discrete multiresolution path planning.

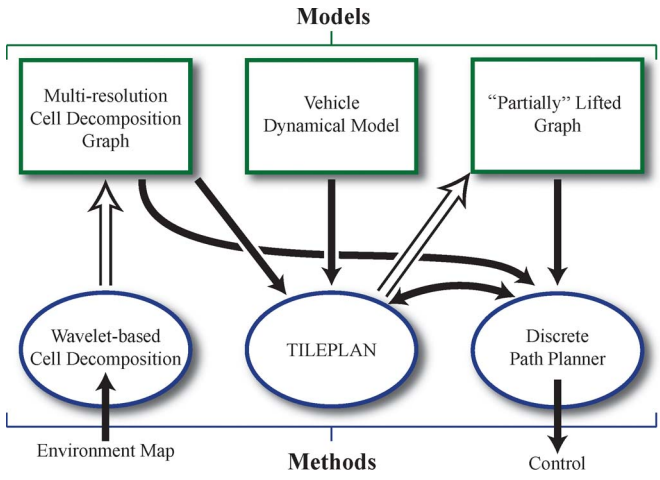


Fig. 1. Schematic of the proposed motion-planning scheme.

The rest of this paper is organized as follows. In Section II, we describe the proposed wavelet-based multiresolution cell decomposition technique. In Section III, we describe a path-planning scheme using this cell decomposition, and we prove its completeness. In Section IV, we discuss the inclusion of vehicle dynamical constraints in the multiresolution path planner via the H-cost approach. In Section V, we provide numerical simulation results illustrating the successful operation of the overall motion planner, and in Section VI, we conclude this paper with comments on possible future extensions.

II. Multiresolution Cell Decompositions Using the DWT

Cell decomposition is a common technique used in geometric path planning [1], which involves partitioning the workspace into convex regions called *cells*. A graph is associated with this partition, such that each cell corresponds to a unique vertex and each pair of geometrically adjacent cells corresponds to a unique edge. The original path-planning problem is thus transformed to the problem of finding a path in this graph, which can be solved, for instance, by the A* algorithm [2]. In what follows, we introduce a multiresolution cell decomposition technique based on the 2-D DWT. We provide a brief introduction to the DWT in Appendix B, and we refer the reader to [12] and [13] for further details.

A. Multiresolution Cell Decompositions

We define an *image* as a pair (R, F), where $R \subset \mathbb{R}^2$ is a compact square region and $F : R \to \mathbb{R}$, $F \in \mathbb{L}^2(R)$, is an intensity map. We will assume that $R = [0, 2^D] \times [0, 2^D]$, with $D \in \mathbb{Z}$, and that the image intensity map F is known at a finite resolution $m_f > -D$, i.e., the function F is piecewise constant over each of the square regions $S_{m_f,k,\ell}$ [defined in (B.2)], for $k, \ell = 0, 1, \ldots, 2^{D+m_f} - 1$. We will assume, without loss of generality, that $m_f = 0$. In the context of path planning, the intensity map F may represent, for instance, terrain elevation [25], a risk measure [23], or a probabilistic occupancy grid [19], [32].

We assume that the least cell size of interest is $2^{-m_f} = 1$, and we define a cell decomposition Ω consisting of uniformly spaced square cells, each of size 1, i.e.,

$$\Omega := \left\{ S_{m_f,k,\ell} : k, \ell \in \{0, 1, \ldots, 2^D - 1\} \right\}.$$

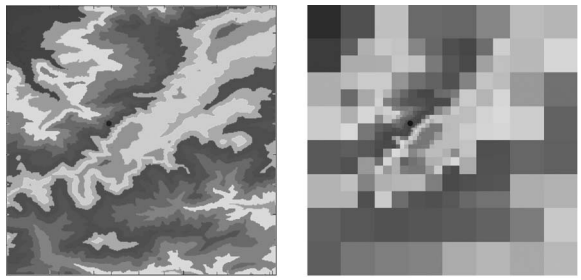
The intention of the geometric path planner is to find a path in the graph associated with Ω. However, the number of cells in Ω is 2^{2D}, which makes the graph search impractical when D is large. To enable fast online computation, multiresolution cell decompositions may be used to approximate large environment maps. Such decompositions correspond to graphs with significantly fewer vertices, thus requiring lesser computational resources for path planning at each iteration. Furthermore, such decompositions also naturally capture the vehicle's sensing limitations by progressively relaxing, with increasing distance from the vehicle's location, the accuracy at which the intensities of cells in Ω are known.

Let $a_{m_0,k,\ell}$ and $d^p_{m,k,\ell}$ be the DWT coefficients of F, where $m_0 \in \mathbb{Z}$ is prespecified; let $\mathcal{A} \subset \{(m, k, \ell) \in \mathbb{Z}^3 : m_0 \leqslant m < 0, 0 \leq k, \ell \leqslant 2^{D+m}\}$ be a set of triplets of integers; and let $\hat{d}^p_{m,k,\ell}$ be defined by

$$\hat{d}^p_{m,k,\ell} := \begin{cases} d^p_{m,k,\ell} & \text{if } p = 1, 2, 3 \text{ and } (m, k, \ell) \in \mathcal{A}, \\ 0 & \text{otherwise.} \end{cases}$$

Then, the image $(R, \hat{F})$, where $\hat{F}$ is obtained by the reconstruction of $a_{m_0,k,\ell}$ and $\hat{d}^p_{m,k,\ell}$, is called the *approximation* of (R, F) associated with the set $\mathcal{A}$. Informally, an approximation is obtained by neglecting certain detail coefficients in the DWT of F; the set $\mathcal{A}$ contains the indices of detail coefficients that are considered "significant." A specific approximation that is of interest in this paper is one that retains detail coefficients only in the immediate vicinity of the vehicle's current location $(x_0, y_0) \in R$ and gradually discards them in regions farther away. To precisely define this approximation, let $\varrho : \mathbb{Z} \to \mathbb{N}$ be a "window" function that specifies, for each level of resolution, the distance from the vehicle's location up to which the detail coefficients at that level are significant. The set $\mathcal{A}_{\text{win}}(x_0, y_0)$ of indices of significant detail coefficients is then defined by

$$\begin{aligned} \mathcal{A}_{\text{win}}(x_0, y_0) := \{(m, k, \ell) : m_0 \leqslant m < 0, \\ \lfloor 2^m x_0 \rfloor - \varrho(m) \leqslant k \leqslant \lfloor 2^m x_0 \rfloor + \varrho(m), \\ \lfloor 2^m y_0 \rfloor - \varrho(m) \leqslant \ell \leqslant \lfloor 2^m y_0 \rfloor + \varrho(m)\}, \end{aligned} \tag{1}$$



(a) (b)

Fig. 2. Example of an image and its multiresolution approximation. (a) Original image. (b) Approximation.

where $m_0 \in \mathbb{Z}$. An example of an image and its approximation using (1) are shown in Fig. 2. In this example, $m_0 = -10$, $(x_0, y_0) = (390, 449)$, and $\varrho(m) = 4$ for each $m_0 \leqslant m < 0$.

B. Computing Cell Locations and Intensities

The cell decomposition Ω^{mr} associated with $(R, \hat{F})$ is a partition of R into square cells of different sizes, such that $\hat{F}$ is constant over each of the cells. In this section, we describe a procedure to determine the locations, sizes, and values of $\hat{F}$ over each of the cells in Ω^{mr}.

In this paper, we use the Haar wavelet family, and the Haar scaling function satisfies the following dilation equation [13]:

$$\phi(t) = \phi(2t) + \phi(2t - 1). \tag{2}$$

Equation (2) implies, for the 2-D case, that the square support of the scaling functions $\Phi_{m,k,\ell}$ is exactly the union of the supports of $\Phi_{m+1,k,\ell}$, $\Phi_{m+1,k-1,\ell}$, $\Phi_{m+1,k,\ell-1}$, and $\Phi_{m+1,k-1,\ell-1}$. Consequently, a map F is constant over the support of $\Phi_{m,k,\ell}$ if and only if the detail coefficients of F at level m and at higher resolution levels $m + 1, m + 2, \ldots$ are all zero. Furthermore, for the Haar scaling function and wavelet, one may associate with each detail coefficient a region in $\mathbb{R}^2$, such that this coefficient affects the values of the map only in this region. Specifically, we make the following association:

$$d^p_{m,k,\ell} \leftrightarrow S_{m,k,\ell} = 2^{-m} \left([k, (k + 1)] \times [\ell, (\ell + 1)] \right), \tag{3}$$

for each $m_0 \leqslant m < m_f = 0$, where $m_0 \in \mathbb{Z}$ is prespecified.

Based on the preceding observations, we formulate the following Rules to determine the locations and sizes of cells in the cell decomposition Ω^{mr} associated with a set $\mathcal{A}$ of indices of the significant detail coefficients.

1) $\{S_{m_0,\hat{k},\hat{\ell}} : 0 \leqslant \hat{k}, \hat{\ell} < 2^{D+m_0}\} \in \Omega^{\text{mr}}$. If $\mathcal{A}$ is empty, then these cells form a uniform decomposition.
2) $\{S_{m+1,\hat{k},\hat{\ell}} : \hat{k} \in \{2k, 2k + 1\}, \hat{\ell} \in \{2\ell, 2\ell + 1\}\} \in \Omega^{\text{mr}}$ whenever $(m, k, \ell) \in \mathcal{A}$. This Rule arises from the fact that the support of the Haar scaling function at each resolution level is equal to the union of the supports of the scaling functions at the next higher resolution level.
3) $\{S_{\hat{m}+1,\hat{k},\hat{\ell}} : \hat{k} \in \{\lfloor 2^{\hat{m}-m} k \rfloor - 1, \lfloor 2^{\hat{m}-m} k \rfloor\}, \hat{\ell} \in \{\lfloor 2^{\hat{m}-m} \ell \rfloor - 1, \lfloor 2^{\hat{m}-m} \ell \rfloor\}, m_0 \leqslant \hat{m} < m\} \in \Omega^{\text{mr}}$ whenever $(m, k, \ell) \in \mathcal{A}$. This Rule decomposes into square nonconvex regions that arise when the indices of a detail coefficient at level m are in $\mathcal{A}$, but the indices

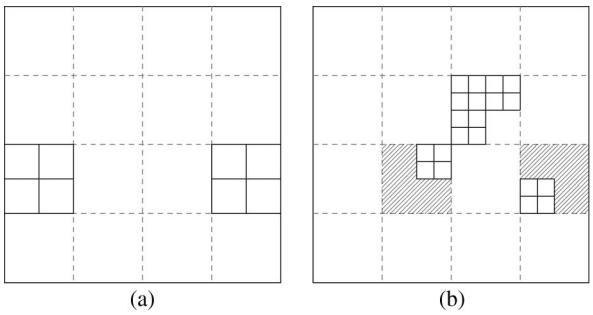


Fig. 3. Computations of cell locations and dimensions from $\mathcal{A}$. (a) Rule 2 for $m = -2$. (b) Rule 2 for $m = -1$.

of coefficients associated with the same region [given by (3)] at all levels lower than m are not in $\mathcal{A}$.

4) $\{S_{\hat{m},\hat{k},\hat{\ell}} : \hat{k} = \lfloor 2^{\hat{m}-m}k \rfloor, \hat{\ell} = \lfloor 2^{\hat{m}-m}\ell \rfloor, m_0 \leqslant \hat{m} \leqslant m\} \notin \Omega^{\mathrm{mr}}$ whenever $(m,k,\ell) \in \mathcal{A}$. This rule indicates that a cell, once decomposed, cannot belong to Ω^{mr}.

Exclusions from Ω^{mr} prescribed by Rule 4 take precedence over inclusions prescribed by Rule 3. Note that the preceding rules are valid only for the Haar system.

Figs. 3 and 4 illustrate the application of the preceding rules for the approximation associated with $\mathcal{A} = \{(-2,0,2),(-2,3,2),(-1,3,4),(-1,4,2),(-1,4,3),\ (-1,5,2),(-1,6,5)\}$. In Fig. 3(a), the cells with dotted borders are due to Rule 1, and the cells with solid borders are due to Rule 2 for the indices with $m = -2$. The shaded cells in Fig. 3(b) illustrate the nonconvex regions that may arise due to nonzero coefficients at higher resolution levels, which need to be decomposed using Rule 3. The shaded cells in Fig. 4(a) are those that arise twice: due to Rule 2 for level $m = -2$ coefficients and due to Rule 3 for level $m = -1$ coefficients. Fig. 4(b) shows the final cell decomposition. After determining the elements of the multiresolution cell decomposition, i.e., the locations and sizes of all the cells, the adjacency relations between cells can be determined by geometric arguments (cf., [24]). To calculate the cell intensities, we recursively use the following relation:

$$\begin{bmatrix} \hat{F}(S_{m+1,2k,2\ell}) \\ \hat{F}(S_{m+1,2k+1,2\ell}) \\ \hat{F}(S_{m+1,2k,2\ell+1}) \\ \hat{F}(S_{m+1,2k+1,2\ell+1}) \end{bmatrix} = 2^{m_0} E \begin{bmatrix} 2^{-m}\hat{F}(S_{m,k,\ell}) \\ d^1_{m,k,\ell} \\ d^2_{m,k,\ell} \\ d^3_{m,k,\ell} \end{bmatrix}, \quad (4)$$

for $0 \leqslant k,\ell < 2^{D+m}$, with $\hat{F}(S_{m_0 k\ell}) = 2^{m_0} a_{m_0,k,\ell}$ for $0 \leqslant k,\ell < 2^{D+m_0}$, where E is a constant matrix. The intensities of the cells due to Rule 2 for a triplet $(m,k,\ell) \in \mathcal{A}$ are given in (4). The intensities of the cells due to Rule 3 for a triplet $(m,k,\ell) \in \mathcal{A}$ are each equal to $F(S_{m_1,k_1,\ell_1})$, where $(m_1,k_1,\ell_1) \in \mathcal{A}$ is the triplet with the greatest $m_1 < m$ satisfying $S_{m,k,\ell} \subset S_{m_1,k_1,\ell_1}$. If no such triplet exists, then the intensities of these cells are each equal to $F(S_{m_0,k_1,\ell_1})$, where k_1,ℓ_1 are the unique indices satisfying $S_{m,k,\ell} \subset S_{m_0,k_1,\ell_1}$.

We re-emphasize that *all* information needed to completely define the cell decomposition Ω^{mr} is encoded in set $\mathcal{A}$, and it is straightforward to extract this information. Furthermore, expression (1) lends itself to a fast update of set $\mathcal{A}$ in accordance with the changes in the vehicle's position in the environment, as we will demonstrate in Section III-B.

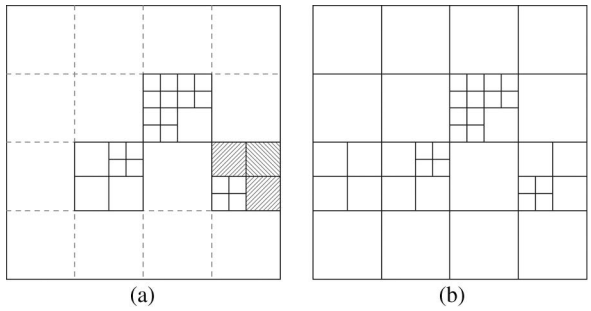


Fig. 4. Computations of cell locations and dimensions from $\mathcal{A}$. (a) Rule 3 for $m = -1$. (b) Overall decomposition.

III. Multiresolution Path Planning

We denote by $\bar{\mathcal{G}} = (\bar{V}, \bar{E})$ the graph associated with the cell decomposition Ω, such that each cell in Ω corresponds to a unique vertex in $\bar{V}$. We will denote by $\mathsf{cell}(j;\Omega^{\mathrm{mr}})$ the cell in Ω^{mr} associated with a vertex $j \in \bar{V}$ and by $\mathsf{vert}(c;\bar{\mathcal{G}})$ the vertex of $\bar{\mathcal{G}}$ associated with a cell $c \in \Omega^{\mathrm{mr}}$. Two vertices are adjacent if the corresponding cells are geometrically adjacent, and $\bar{E}$ is the collection of all ordered pairs $(\bar{i},\bar{j})$ of vertices in $\bar{V}$, such that $\bar{i}$ and $\bar{j}$ are adjacent. In what follows, we will distinguish by overlines symbols denoting vertices, paths, or functions associated with Ω or $\bar{\mathcal{G}}$. We introduce an edge cost function $\bar{g} : \bar{E} \to \mathbb{R}_+$ that assigns to each edge of $\bar{\mathcal{G}}$ a nonnegative cost of transitioning this edge.

For given initial and terminal vertices $\bar{i}_S,\ \bar{i}_G \in \bar{V}$, an *admissible path* $\bar{\pi}(\bar{i}_S,\bar{i}_G)$ in $\bar{\mathcal{G}}$ is a finite sequence $(\bar{i}_0,\ldots,\bar{i}_{\bar{P}})$ of vertices (with no repetition) such that $\{\bar{i}_{k-1},\bar{i}_k\} \in \bar{E}$ for each $k = 1,\ldots,P$, with $\bar{i}_0 = \bar{i}_S$ and $\bar{i}_P = \bar{i}_G$. For brevity, and when there is no ambiguity, we will henceforth suppress the arguments in $\bar{\pi}$. The cost $\bar{\mathcal{J}}(\bar{\pi})$ of an admissible path $\bar{\pi}$ in $\bar{\mathcal{G}}$ is the sum of costs of edges in $\bar{\pi}$, and the *path-planning problem* is to find an admissible path $\bar{\pi}^*(\bar{i}_S,\bar{i}_G)$ with minimum cost.

Next, we associate with the multiresolution cell decomposition Ω^{mr} a graph $\mathcal{G} = (V,E)$ such that each cell in Ω^{mr} corresponds to a unique vertex in V. Note that each vertex $j \in V$ corresponds to a set $W(j,V) \subset \bar{V}$, and the collection $\{W(j,V)\}_{j\in V}$ is a partition of $\bar{V}$. Specifically,

$$W(j,V) := \left\{\bar{j} \in \bar{V} : \mathsf{cell}(\bar{j};\Omega) \subseteq \mathsf{cell}(j;\Omega^{\mathrm{mr}})\right\}.$$

The multiresolution cell decomposition graph $\mathcal{G}$ approximates graph $\bar{\mathcal{G}}$ by representing each set of vertices $W(j,V) \subset \bar{V}$ with a single vertex $j \in V$. For the Haar wavelet, it can be shown that, for each $j \in V$

$$\hat{F}\left(\mathsf{cell}(j;\Omega^{\mathrm{mr}})\right) = \frac{1}{|W(j,V)|} \sum_{\bar{j}\in W(j,V)} F\left(\mathsf{cell}(\bar{j};\Omega)\right). \quad (5)$$

Finally, two vertices $i,j \in V$ are said to be adjacent in $\mathcal{G}$, i.e., $(i,j) \in E$, if and only if there exist $\bar{i} \in W(i,V)$ and $\bar{j} \in W(j,V)$ such that $\{\bar{i},\bar{j}\} \in \bar{E}$. We will denote by $\mathsf{cell}(j;\Omega^{\mathrm{mr}})$ the cell in Ω^{mr} associated with the vertex $j \in V$ and by $\mathsf{vert}(S;\mathcal{G})$ the vertex in $\mathcal{G}$ associated with the cell $S \in \Omega^{\mathrm{mr}}$.

A. Path-Planning Algorithm

We present an algorithm that iteratively finds an admissible path $\bar{\pi}$ in $\bar{\mathcal{G}}$ by first constructing multiresolution cell decompositions, then by finding paths in these multiresolution cell decompositions, and then by moving along these paths. The proposed algorithm is a modification of the multiresolution path-planning algorithm presented in [23], and these modifications ensure that the proposed algorithm is complete.

We assume that $F(\mathsf{cell}(\bar{j};\Omega)) \in [0,1]$ for each $\bar{j} \in \bar{V}$, such that more favorable cells in the environment have a lower intensity and such that $\mathsf{cell}(\bar{j};\Omega)$ represents an obstacle if $F(\mathsf{cell}(\bar{j};\Omega)) > 1-\varepsilon$, for some prespecified $\varepsilon \in (0,1)$. We define the transition cost of an edge $(\bar{i},\bar{j}) \in \bar{E}$ by

$$\bar{g}\left((\bar{i},\bar{j})\right) := \begin{cases} \lambda_1 F(\mathsf{cell}(\bar{j};\Omega)) + \lambda_2, & F(\mathsf{cell}(\bar{j};\Omega)) \leqslant 1-\varepsilon \\ M, & \text{otherwise,} \end{cases} \tag{6}$$

where $\lambda_1, \lambda_2 \in (0,1]$ and $M \gg 2|\bar{V}|$ are constants.

We denote by $\hat{F}_n$ the approximation of the environment constructed at iteration n of the proposed algorithm, by $\mathcal{A}(n)$ the associated set of detail coefficients, by $\Omega^{\mathrm{mr}}(n)$ the associated multiresolution cell decomposition, and by $\mathcal{G}(n) = (V(n), E(n))$ the associated topological graph. We define the goal vertex $i_{G,n} \in V(n)$ as the unique vertex that satisfies $\bar{i}_G \in W(i_{G,n}V(n))$.

For each vertex $\bar{j} \in \bar{V}$, the proposed algorithm maintains an estimate $\mathcal{K}_G(\bar{j})$ of the least cost of any path in $\bar{\mathcal{G}}$ from the vertex $\bar{j}$ to the goal vertex $\bar{i}_G$ and a record $\mathcal{K}_S(\bar{j})$ of the least cost of any path in $\bar{\mathcal{G}}$ from the initial vertex $\bar{i}_S$ to the vertex $\bar{j}$. The algorithm also associates with each vertex $\bar{j} \in \bar{V}$ another vertex $b(\bar{j}) \in \bar{V}$ called the *backpointer* of $\bar{j}$. At each iteration, the algorithm performs a computation (see Line 18 or Line 20 in Fig. 5) whose result is a unique vertex in $\bar{V}$. We refer to this computation as a *visit* to this vertex, and we denote by $\bar{j}_n$ the vertex visited at iteration n, with $\bar{j}_0 := \bar{i}_S$. Finally, let $j_n := \mathsf{vert}(\mathsf{cell}(\bar{j}_n;\Omega^{\mathrm{mr}}(n));\mathcal{G}(n))$.

An admissible path $\pi_n(j_n, i_{G,n})$ in $\mathcal{G}(n)$ is a finite sequence $(i_0,\ldots,i_{P(n)})$ of vertices (with no repetition) in $V(n)$ excluding $b(\bar{j}_n)$ and excluding vertices in $V(n)$ corresponding to cells in the path from $\bar{i}_S$ to $\bar{j}_n$. Note that this definition precludes cycles in the concatenation of path $(j_0,\ldots,j_{n-1})$ with path π_n. We introduce an edge cost function $g_n : E(n) \to \mathbb{R}_+$, which assigns to each edge of $\mathcal{G}(n)$ a nonnegative cost of transitioning this edge, defined by

$$g_n\left((i,j)\right) \coloneqq \begin{cases} (\lambda_1 \hat{F}_j + \lambda_2)|W\left(j, V(n)\right)|, & \hat{F}_j \leqslant 1-\varepsilon \\ M, & \text{otherwise,} \end{cases} \tag{7}$$

where $\hat{F}_j := \hat{F}(\mathsf{cell}(j;\Omega^{\mathrm{mr}}(n)))$. The cost $\mathcal{J}(\pi_n)$ of path π_n is the sum of the costs of edges in the path. Note that, by (5) and (7), the cost of an obstacle-free path in $\mathcal{G}(n)$ is less than or equal to $2|\bar{V}|$, and hence, an admissible path π_n in $\mathcal{G}(n)$ is obstacle-free if and only if $\mathcal{J}(\pi_n) < M$.

The proposed algorithm associates with each vertex $\bar{j} \in \bar{V}$ a binary value $\textsc{visited}(\bar{j})$ that records whether vertex $\bar{j}$ has been previously visited by the algorithm; at any iteration of the algorithm's execution, for each $\bar{j} \in \bar{V}$, $\textsc{visited}(\bar{j})=0$ indicates that the algorithm has never visited $\bar{j}$ in any previous iteration, whereas $\textsc{visited}(\bar{j})=1$ indicates that the algorithm has visited $\bar{j}$ during a previous iteration. The algorithm also maintains a cumulative cost $\bar{\mathcal{J}}(\bar{\pi})$ of the path $\bar{\pi}(\bar{i}_S, \bar{j}_n)$ in $\bar{\mathcal{G}}$. The proposed multiresolution path-planning algorithm is described by the pseudocode in Fig. 5. Here, $x(\bar{j})$ and $y(\bar{j})$ respectively denote the x- and y-coordinates of the center of $\mathsf{cell}(\bar{j};\bar{\mathcal{G}})$, and MR-GRAPH denotes the procedure described in Section II-B to obtain the cell decomposition graph associated with a set of indices of significant detail coefficients.

Multi-resolution Path Planning Scheme

```
procedure MR-APPROX(j̄)
 1: A ← A_win(x(j̄), y(j̄))
procedure MAIN
 1: π̄ ← ī_S, j̄_0 ← ī_S, n ← 0, AtGoal ← 0, J̄(π̄) ← 0
 2: For each j̄ ∈ V̄, VISITED(j̄) ← 0
 3: while ¬AtGoal and J̄(π̄) < M and K_G(j̄_n) < M do
 4:    b(j̄_n) ← j̄_{n−1}
 5:    A(n) ← MR-APPROX(j̄_n)
 6:    G(n) ← MR-GRAPH(A(n))
 7:    if VISITED(j̄_n) = 1 then
 8:       π*_n ← arg min {J(π) : π obstacle-free in G(n)},
          subject to J(π*_n) > K_G(j̄_n)
 9:       J̄(π̄) ← K_S(j̄_n)
10:    else
11:       π*_n ← arg min {J(π) : π obstacle-free in G(n)}
12:       K_S(j̄_n) ← J̄(π̄)
13:    VISITED(j̄_n) ← 1
14:    if π*_n does not exist then
15:       if j̄_n = ī_S then
16:          Report failure
17:       else
18:          j̄_{n+1} ← b(j̄_n)
19:    else
20:       j̄_{n+1} ← vert(cell(i_1; G(n)); Ḡ)
21:    K_G(j̄_n) ← J(π*_n)
22:    AtGoal ← (K_G(j̄_n) = 0),
23:    π̄ ← (π̄, j̄_n)
24:    J̄(π̄) ← J̄(π̄) + ḡ(j̄_n, j̄_{n+1})
25:    n ← n + 1
26: if J̄(π̄) ⩾ M or K_G(j̄_n) ⩾ M then
27:    Report failure
```

Fig. 5. Pseudocode for the proposed path-planning algorithm.

Remark 1: The constrained optimization problem in Line 8 can be solved by an algorithm that finds the K shortest paths in a graph. Such algorithms have been reported, for instance, in [33]. We assume that the K shortest paths will have strictly increasing costs. This assumption is not required for the algorithm's successful execution, but it enables a concise statement of the algorithm. □

Remark 2: Due to Line 9, the cost of "backtracking" is not added to the cumulative cost $\bar{\mathcal{J}}(\bar{\pi})$. In addition, it follows from (7) and Line 21 that $\mathcal{K}_G(\bar{j}) = 0$ if and only if $\bar{j} = \bar{i}_G$. □

We may now state the main result of this section as follows.

Proposition 1: The proposed algorithm is complete: If there exists an obstacle-free path in $\bar{\mathcal{G}}$ from $\bar{i}_S$ to $\bar{i}_G$, then the algorithm finds an obstacle-free path in a finite number of iterations. Otherwise, the algorithm reports failure after a finite number of iterations.

Proof: See Appendix A. ■

B. Efficient Updates of $\mathcal{A}(n)$ and $\mathcal{G}(n)$

The set $\mathcal{A}(n)$ of the significant detail coefficient indices and the associated multiresolution cell decomposition graph both depend on the vehicle's current position. Consequently, both $\mathcal{A}(n)$ and $\mathcal{G}(n)$ are updated (see Lines 5 and 6 in Fig. 5) at each iteration of the algorithm. To enable faster computations, we describe, in this section, a method to obtain $\mathcal{A}(n)$ incrementally from $\mathcal{A}(n-1)$. Specifically, we first compute the elements of sets $\mathcal{B}_1 := \mathcal{A}(n) \cap \mathcal{A}^c(n-1)$ and $\mathcal{B}_{-1} := \mathcal{A}(n-1) \cap \mathcal{A}^c(n)$, and we then evaluate $\mathcal{A}(n) = \mathcal{A}(n-1) \cup \mathcal{B}_1 \backslash \mathcal{B}_{-1}$. To this end, we observe that for each $\bar{j} \in \bar{V}$, $x(\bar{j}) = \lfloor x(\bar{j}) \rfloor + 1/2$. It follows that, for every $m \leqslant 0$

$$\lfloor 2^m x(\bar{j}_n) \rfloor = \lfloor 2^m \left(\lfloor x(\bar{j}) \rfloor + 1/2 \right) \rfloor . \tag{8}$$

Next, we note that $\lfloor x(\bar{j}_n) \rfloor = \lfloor x(\bar{j}_{n-1}) \rfloor + \Delta_x$, where, at iteration n, $\Delta_x = 1$ if the vehicle moves one cell to the right, $\Delta_x = -1$ if the vehicle moves one cell to the left, and $\Delta_x = 0$ otherwise. From (8), it may be shown [30] that

$$\lfloor 2^m x(\bar{j}_{n+1}) \rfloor = \lfloor \lfloor 2^m x(\bar{j}_n) \rfloor + 2^m \Delta_x + r_x^m \rfloor , \tag{9}$$

where $r_x^m := 2^m (\lfloor 2^m x(\bar{j}_n) \rfloor + 1/2) - \lfloor 2^m x(\bar{j}_n) \rfloor$. Similarly

$$\lfloor 2^m y(\bar{j}_{n+1}) \rfloor = \lfloor \lfloor 2^m y(\bar{j}_n) \rfloor + 2^m \Delta_y + r_y^m \rfloor , \tag{10}$$

where $r_y^m := 2^m (\lfloor 2^m y(\bar{j}_n) \rfloor + 1/2) - \lfloor 2^m y(\bar{j}_n) \rfloor$. The elements of sets $\mathcal{B}_1$ and $\mathcal{B}_{-1}$ are then determined from (9) and (10) as follows. We first define scalars δ_x and δ_y by

$$\delta_\alpha := \begin{cases} -1, & 0 > 2^m \Delta_\alpha + r_\alpha^m, \\ 0, & 0 \leqslant 2^m \Delta_\alpha + r_\alpha^m < 1, \\ 1, & 1 \leqslant 2^m \Delta_\alpha + r_\alpha^m, \end{cases} \quad \text{for } \alpha \in \{x, y\}, \tag{11}$$

and for $p \in \{-1, 1\}$, we define sets $\mathcal{B}_p^{m,x}$ and $\mathcal{B}_p^{m,x}$ by

$$\begin{aligned} \mathcal{B}_p^{m,x} := \{ & (m,k,\ell) : k = \lfloor 2^m x(\bar{j}_n) \rfloor + p\delta_x, \\ & \lfloor 2^m y(\bar{j}_n) \rfloor - \varrho(m) \leqslant \ell \leqslant \lfloor 2^m y(\bar{j}_n) \rfloor + \varrho(m) \} \\ \mathcal{B}_p^{m,y} := \{ & (m,k,\ell) : \ell = \lfloor 2^m y(\bar{j}_n) \rfloor + p\delta_y, \\ & \lfloor 2^m x(\bar{j}_n) \rfloor - \varrho(m) \leqslant k \leqslant \lfloor 2^m x(\bar{j}_n) \rfloor + \varrho(m) \} . \end{aligned}$$

Then, sets $\mathcal{B}_{-1}$ and $\mathcal{B}_1$ are given by the following equation:

$$\mathcal{B}_p = \bigcup_{\alpha \in \{x,y\}} \bigcup_{m_0 \leqslant m < 0} \mathcal{B}_p^{m\alpha}, \quad p \in \{-1, 1\}. \tag{12}$$

The advantage of computing $\mathcal{A}(n)$ using the modified procedure MOD-MR-APPROX described in Fig. 6 instead of the procedure MR-APPROX is that sets $\mathcal{B}_{-1}$ and $\mathcal{B}_1$ have significantly fewer elements than $\mathcal{A}(n)$. More precisely, the number of elements in set $\mathcal{A}(n)$ is $O(\bar{\varrho}^2)$, whereas the numbers of elements in sets $\mathcal{B}_{-1}$ and $\mathcal{B}_1$ are both $O(\bar{\varrho})$, where $\bar{\varrho} := \max_{m_0 \leqslant m \leqslant 0} \{\varrho(j)\}$.

Fig. 7(a) and (b) shows data that confirm the preceding observations; these figures show the ratio of the execution time required by the combination of the procedures MR-APPROX and MR-GRAPH to the execution time required by the combination of the procedures MOD-MR-APPROX and MOD-MR-GRAPH for computing the graph $\mathcal{G}(n)$. The data shown in Fig. 7(a) and (b) are averages computed over 30 simulations. As it is evident from these results, the multiresolution path-planning algorithm with the modified procedures of construction of $\mathcal{A}(n)$ and $\mathcal{G}(n)$ executes up to 10 times faster than that with the original procedures.

Recomputation of Multi-resolution Cell Decomposition

procedure MOD-MR-APPROX($\mathcal{A}$)

1: Compute $\mathcal{B}_{-1}$ and $\mathcal{B}_1$ with (12)
2: $\mathcal{A}(n) \leftarrow \mathcal{A}(n-1) \cup \mathcal{B}_1 \backslash \mathcal{B}_{-1}$

procedure MOD-MR-GRAPH($\Omega^{\mathrm{mr}}(n-1), \mathcal{B}_{-1}, \mathcal{B}_1$)

1: $\Omega_{-1}^{\mathrm{mr}} \leftarrow \varnothing$, $\Omega_1^{\mathrm{mr}} \leftarrow \varnothing$
2: **for all** $(m,k,\ell) \in \mathcal{B}_1$ **do**
3: $\quad \Omega_1^{\mathrm{mr}} \leftarrow \Omega_1^{\mathrm{mr}} \cup \{S_{m+1,\hat{k},\hat{\ell}} : \hat{k} \in \mathcal{K},\ \hat{\ell} \in \mathcal{L}\}$
4: $\quad \Omega_{-1}^{\mathrm{mr}} \leftarrow \Omega_{-1}^{\mathrm{mr}} \cup \{S_{\hat{m},\hat{k},\hat{\ell}} : \hat{k} = \lfloor 2^{\hat{m}-m} k \rfloor,\ \hat{\ell} = \lfloor 2^{\hat{m}-m} \ell \rfloor,\ m_0 \leqslant \hat{m} \leqslant m\}$
5: **for all** $(m,k,\ell) \in \mathcal{B}_{-1}$ **do**
6: $\quad \Omega_{-1}^{\mathrm{mr}} \leftarrow \Omega_{-1}^{\mathrm{mr}} \cup \{S_{m+1,\hat{k},\hat{\ell}} : \hat{k} \in \mathcal{K},\ \hat{\ell} \in \mathcal{L}\}$
7: $\quad \Omega_1^{\mathrm{mr}} \leftarrow \Omega_1^{\mathrm{mr}} \cup \{S_{m,k,\ell}\}$
8: $\Omega^{\mathrm{mr}}(n) \leftarrow \Omega^{\mathrm{mr}}(n-1) \cup \Omega_1^{\mathrm{mr}} \backslash \Omega_{-1}^{\mathrm{mr}}$

Fig. 6. Pseudocode for the procedure mod-mr-graph.

IV. Multiresolution H-Cost Motion Planning

It has been noted in several previous works [34]–[36], including ours [28], that single-edge transition costs in cell decomposition graphs cannot adequately capture the vehicle's kinematic and dynamic constraints. In light of this observation, we introduced in [28] a motion-planning approach based on assigning costs to *multiple*-edge transitions (called *histories*) in cell decomposition graphs.

Consider the multiresolution cell decomposition graph[1] $\mathcal{G} = (V, E)$ at any iteration of the path-planning algorithm previously discussed. To formalize the concept of cell histories, we define, for every integer $H \geqslant 0$, set

$$\begin{aligned} V_H := \{ & (j_0, \ldots, j_H) : \{j_{k-1}, j_k\} \in E, k = 1, \ldots, H, \\ & j_k \neq j_\ell, \quad \text{for} \quad k, \ell \in \{0, \ldots, H\}, \quad \text{with} \quad k \neq \ell \} . \end{aligned}$$

An element of V_{H+1} is called an H-*history*. Let $I \in V_H$, and denote by $[I]_k$ the k^{th} element of this $(H+1)$-tuple and by $[I]_k^\ell$ the tuple $([I]_k, [I]_{k+1}, \ldots, [I]_\ell)$ for $k < \ell \leqslant H+1$. We associate with each H a nonnegative cost function $g_H : V_{H+1} \to \mathbb{R}_+$ and we state a shortest path problem with transition costs defined on histories as follows.

Problem 1 (H-Cost Shortest Path Problem): Let $H \geqslant 0$, and let $i_S, i_G \in V$ be initial and goal vertices such that any admissible path in $\mathcal{G}$ contains at least $H+1$ vertices.

[1]For the sake of clarity, we drop from the notation of the cell decomposition graph the explicit reference to the nth iteration.

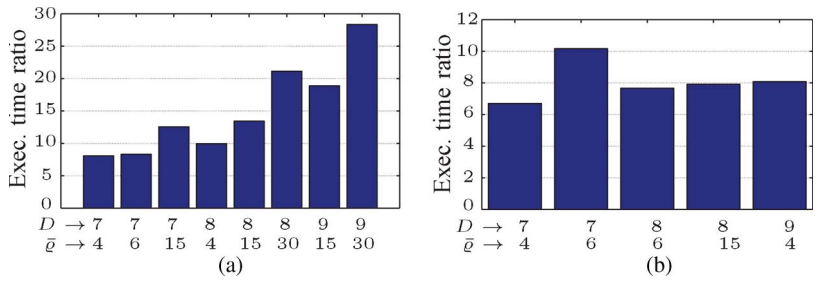


Fig. 7. Sample data illustrating benefits of incremental updates to $\mathcal{A}$ and $\mathcal{G}$. (a) Comparisons of execution times for the computations of $\mathcal{A}$ and $\mathcal{G}$. (b) Comparisons of execution times for overall path planning.

The H-cost of an admissible path $\pi = (j_0, \ldots, j_P)$ in $\mathcal{G}$ is defined by

$$\mathcal{J}_H(\pi) := \sum_{k=H+1}^{P} g_H\left((j_{k-H-1}, j_{k-H}, \ldots, j_k)\right). \tag{13}$$

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

Problem 1 may be transformed into an equivalent standard shortest path problem on a *lifted graph* $\mathcal{G}_H$. The vertices of $\mathcal{G}_H$ are the elements of V_H, and the edge set E_H of the lifted graph $\mathcal{G}_H$ is the set of all ordered pairs (I, J), such that $I, J \in V_H$, with $[I]_k = [J]_{k-1}$, for every $k = 2, \ldots, H+1$, and $[I]_1 \neq [J]_{H+1}$. For given initial and terminal vertices i_S, $i_G \in V$, an admissible path Π in $\mathcal{G}_H$ is a finite sequence $(J_0, \ldots, J_Q)$ of vertices (with no repetition) such that $(J_{k-1}, J_k) \in E_H$, for each $k = 1, \ldots, Q$, with $[J_0]_1 = i_S$, and $[J_Q]_{H+1} = i_G$. Note that every admissible path $\Pi = (J_0, \ldots, J_Q)$ in $\mathcal{G}_H$ uniquely corresponds to an admissible path $\pi = (j_0, \ldots, j_P)$ in $\mathcal{G}$, with $P = Q + H$ and $[J_k]_\ell = j_{kH+\ell-1}$, for each $k = 0, 1, \ldots, Q-1$, and $J_Q = (j_{P-H}, \ldots, j_P)$.

We introduce a nonnegative cost function $\tilde{g}_H : E_H \to \mathbb{R}_+$ defined by $\tilde{g}_H((I, J)) := g_H(([I]_1^{H+1}, [J]_{H+1}))$, for every pair $(I, J) \in E_H$. It follows that Problem 1 is equivalent to the standard shortest path problem on $\mathcal{G}_H$, where the cost of an edge $(I, J) \in E_H$ given by $\tilde{g}_H((I, J))$. However, solving the H-cost shortest path problem by first transforming it to the standard problem is computationally intensive because $|\mathcal{G}_H|$ is large and exponentially grows with H.

In [30], we discuss an efficient and flexible algorithm for solving the H-cost shortest path problem, as well as a motion-planning framework that incorporates vehicle kinematic and dynamic constraints by obtaining H-costs from a local trajectory generation algorithm called the *tile motion planner* (TilePlan). A precise statement of the tile motion-planning problem and its solution based on model predictive control are available in [30]. Briefly, we specify TilePlan as an algorithm that takes as the input a sequence of cells and an initial state and returns as the output a control input (if it exists) that enables vehicle's traversal through the given sequence of cells from the given initial state.

The overall motion planner searches for H-cost shortest paths in the multiresolution cell decomposition graphs described in Section III. However, it is unnecessary and computationally expensive to consider history-based transition costs on the entire multiresolution cell decomposition graph due to the following reasons: a) Large cells in Ω^{mr} correspond to coarse information about the environment in the regions associated with those cells, and hence, trajectories passing through large cells will be refined and/or replanned in future iterations; and b) curvature-constrained paths are guaranteed to exist [37] in rectangular channels wider than a certain threshold width (compared to the upper bound on curvature).

In light of the preceding observations and in keeping with the multiresolution idea of using high-accuracy information only locally, the proposed motion planner searches for H-cost shortest paths on a "partially" lifted graph, such that the vehicle dynamical constraints are considered (via history-based transition costs) only locally. To precisely state this notion of a "partially" lifted graph, we define, for each $J = (j_0, \ldots, j_H) \in V_H$ and each $L \in \{1, \ldots, H-1\}$, the *projection* $\mathcal{P}_L(J)$ of J onto V_L by $\mathcal{P}_L(J) := (j_0, \ldots, j_L) \in V_L$. For each $L \in \{1, \ldots, H\}$, we define set $U_L \subseteq V_L$ by

$$U_L := \Big\{(j_0, \ldots, j_L) \in V_L : \mathsf{size}\left(\mathsf{cell}(j_k)\right) < \hat{d}, \\ \text{for } k = 0, \ldots, L-1, \text{and } \mathsf{size}\left(\mathsf{cell}(j_L)\right) \leqslant (d)\Big\}, \tag{14}$$

where $\bar{d}$ is prespecified and $\mathsf{size}(\mathsf{cell}(j_k))$ denotes the size of the cell that corresponds to the vertex j_k in the multiresolution cell decomposition graph. By (14), set U_L consists of $(L+1)$-tuples of vertices in the cell decomposition graph such that the sizes of the first L cells in each $(L+1)$-tuple are strictly lower than $\bar{d}$, whereas the size of the $(L+1)$ cell is at most $\bar{d}$. This definition alludes to the previously stated notion of including in the "partially" lifted graph only the cells small enough for the curvature constraints to be significant. The "partially" lifted graph $\tilde{\mathcal{G}}_H = (\tilde{V}_H, \tilde{E}_H)$ is then defined by

$$\tilde{V}_H := \bigcup_{L=1}^{H} U_L \backslash \mathcal{P}_L(U_H)$$

$$\tilde{E}_H := \bigcup_{L=1}^{H} \left\{(I, J) : I \in U_L, J \in U_{L-1}, [I]_1^L = J\right\}.$$

The overall motion planner then operates as follows. At each iteration, a multiresolution cell decomposition is first constructed. The cells in this decomposition may be categorized into two classes, namely, cells with sizes at most $\bar{d}$ and cells with sizes greater than $\bar{d}$. We define *boundary cells* as the cells of sizes at most $\bar{d}$ that have at least one neighboring cell in

Multi-resolution Motion Planning

1: $i \leftarrow \mathsf{vert}(\mathsf{cell}(\bar{i}_{\mathrm{S}}; \Omega); \mathcal{G})$, *AtGoal* $\leftarrow 0$
2: **while** $\neg$*AtGoal* **do**
3: Define $V_{\leqslant \bar{d}} := \{j \in V : \mathsf{size}(\mathsf{cell}(j)) \leqslant \bar{d}\}$, $V_{>\bar{d}} := \{j \in V : \mathsf{size}(\mathsf{cell}(j)) > \bar{d}\}$
4: Find the set of boundary vertices V_{bd} defined by

$$V_{\mathrm{bd}} := \{j \in V : \exists\, i_1 \in V_{\leqslant \bar{d}},\ i_2 \in V_{>\bar{d}} \text{ such that } (i_1, j) \in E,\ (i_2, j) \in E\}$$

5: For each $j \in V_{\mathrm{bd}}$, define $\mathcal{K}(j) :=$ minimum cost of a path in $\mathcal{G}$ from j to the goal vertex i_{G}
6: Define $V_H^{\mathrm{S}} := \{I \in \tilde{V}_H : [I]_1 = i\}$, $V_H^{\mathrm{G}} := \{I \in \tilde{V}_H : [I]_{\mathrm{last}} \in V_{\mathrm{bd}}\}$
7: Find the shortest path $\Pi^* = (I_0, \ldots, I_P)$ in $\tilde{\mathcal{G}}_H$ from any vertex $I_{\mathrm{S}} \in V_H^{\mathrm{S}}$ to any vertex in $I_{\mathrm{G}} \in V_H^{\mathrm{G}}$, with terminal penalty $\mathcal{K}([I_{\mathrm{G}}]_{\mathrm{last}})$
8: $i \leftarrow [I_0]_2$
9: **if** $\mathsf{cell}(i; \Omega^{\mathrm{mr}}) = \mathsf{cell}(\bar{i}_{\mathrm{G}}; \Omega)$ **then**
10: *AtGoal* $\leftarrow 1$

Fig. 8. Pseudocode describing the overall motion planner.

each of the two previously defined classes [see Fig. 17(a)]. A multiple-source single-goal implementation of the A* algorithm may be used to determine the costs of optimal paths in the multiresolution cell decomposition graph from the vertices associated with each of the boundary cells to the goal vertex. These costs are then used as terminal penalty costs in the execution of the H-cost path planner on the "partially" lifted graph previously discussed. This H-cost path planner returns a sequence of cells from the current location to one of the boundary cells, along with an admissible vehicle control input that enables the traversal of this sequence of cells. The vehicle state is advanced by traversing one cell using this control input, and the process is repeated until the vehicle reaches the goal.

The pseudocode for the overall motion planner is provided in Fig. 8. Note that Line 7 in Fig. 8 involves local trajectory generation (TilePlan) for the particular vehicle dynamical model considered. We refer the reader to [28] for further details on finding the shortest path in the lifted graph in conjunction with TilePlan.

V. Simulation Results and Discussion

In this section, we present numerical simulation results of implementations of the proposed multiresolution path- and motion-planning schemes. All of these simulations were implemented in the MATLAB environment, on a computer with an Intel Core i5-2410M 2.3-GHz CPU and 4-GB RAM.

A. Completeness of the Path-Planning Algorithm

First, we focus on the path-planning algorithm, which does not consider vehicle dynamics. Figs. 9 and 10 illustrate a simulation example demonstrating the capability of the multiresolution path-planning scheme to recover from a cul-de-sac. This simulation "illustrates" the path planner's completeness.

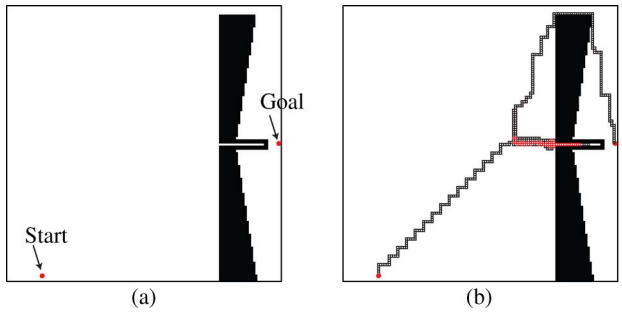


Fig. 9. Illustration of the multiresolution path-planning algorithm's ability to recover from a cul-de-sac: the red-colored cells were multiply visited. (a) Map of the environment. (b) Resultant path.

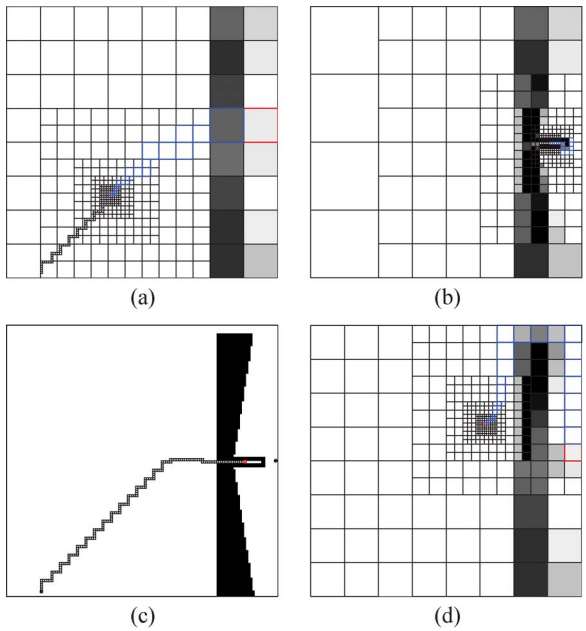


Fig. 10. Intermediate iterations in the multiresolution path-planning algorithm's implementation for the environment shown in Fig. 9(a). (a) An iteration before the cul-de-sac is explored. (b) Iteration at which the cul-de-sac is encountered. (c) Location of the vehicle at the iteration illustrated in Fig. 10(b). (d) Iteration at which the algorithm finds a channel that contains a path to the goal.

As shown in Fig. 9(a), we designed the shape of the obstacle and the location of the goal to lead the multiresolution path-planning algorithm into the cul-de-sac in the "central" region of the obstacle, whereas the goal can be only reached from the "top" region of the obstacle. Fig. 10 illustrates some intermediate iterations in the execution of the multiresolution path-planning algorithm on this environment. Specifically, the algorithm leads the vehicle into the cul-de-sac, but in later iterations, it successfully recovers and finds a path to the goal.

B. Optimality of the Path-Planning Scheme

Whereas we can guarantee the algorithm's capability of finding a *feasible* path whenever such a path exists, we do not yet have theoretical results on the *optimality* of the resultant path. Here, we present numerical simulation results concerning the optimality of paths resulting from the multiresolution path-planning algorithm.

TABLE I
WINDOW FUNCTION VALUES

m	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1
ϱ_1	1	1	2	2	2	2	2	2	3	3	4	4
ϱ_2	1	2	2	3	4	5	6	7	7	7	8	8
ϱ_3	3	3	5	5	6	7	8	8	9	9	10	10

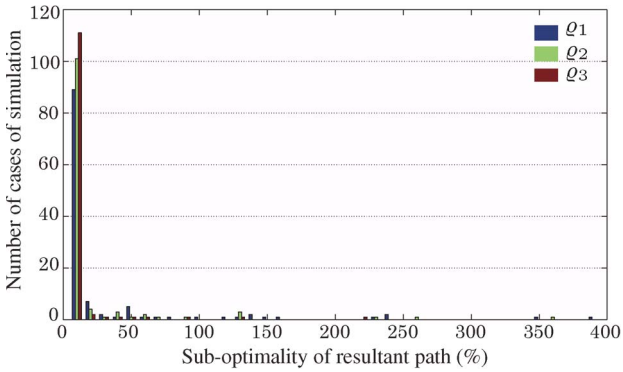


Fig. 11. Histogram showing the distribution according to percentage suboptimality of simulated cases, for different window functions.

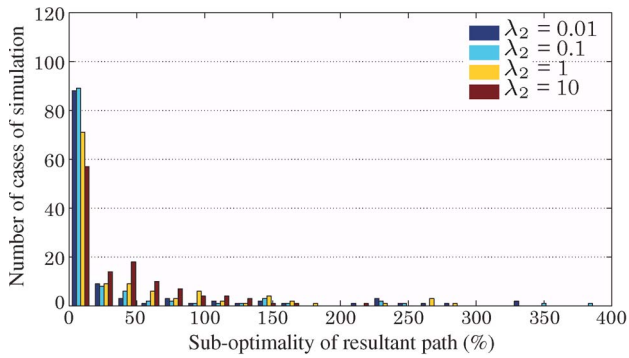


Fig. 12. Histogram showing the distribution according to percentage suboptimality of simulated cases, for different values of λ_2 with $\lambda_1 = 1$.

We compared the cost of the resultant paths with the cost of an optimal path found by executing the A* algorithm on the finest level decomposition graph $\bar{\mathcal{G}}$. For these comparative simulations, we chose an environment represented by the image shown in Fig. 2(a), with three different "window" functions, as described in Table I. Window ϱ_1 retains very few significant detail coefficients and results in a multiresolution cell decomposition with high-fidelity representation of the environment in a very small neighborhood of the vehicle's location, whereas windows ϱ_2 and ϱ_3 result in decompositions with progressively larger neighborhoods of high-fidelity representations. We scaled the environment with $D = 6, 7, 8, 9$, and for each value of D, we performed 30 simulations with the initial and goal cells randomly chosen for each simulation. We executed the multiresolution path-planning algorithm proposed in Section III with each window function for each simulation (a total of 120 simulations for each window function), with $m_0 = -D$.

Fig. 11 shows the distribution of the number of simulated cases according to percentage suboptimality, where the cost of a path in $\bar{\mathcal{G}}$ by (6) was defined with $\lambda_1 = 1$ and $\lambda_2 = 0.1$. For all three window functions, the suboptimality in most cases is under 20%, with window ϱ_3 resulting in the most cases of low suboptimality, as intuitively expected. Overall, Fig. 11 shows that very few cases of extremely high suboptimality occurred; these cases typically occurred when the algorithm encountered cul-de-sacs.

Fig. 12 shows the distribution of the number of simulated cases according to suboptimality for different values of cost function parameters λ_1 and λ_2, all with window function ϱ_1. From (6), note that λ_1 simply scales the image intensity, whereas λ_2 is a constant penalty on each edge in the path. As shown in Fig. 12, the proposed multiresolution path planner results in paths of low suboptimality more often[2] for small values of λ_2. This behavior occurs due to the fact that, for each edge $(i, j) \in E(n)$, expression (12) involves a worst case estimate[3] of the number of vertices of $\bar{\mathcal{G}}$ in the path in $\bar{\mathcal{G}}$ corresponding to the path searched in $\mathcal{G}(n)$. Furthermore, by (7), the multiresolution path planner's estimate of the cost of the actual path becomes progressively more conservative with increasing values of λ_2.

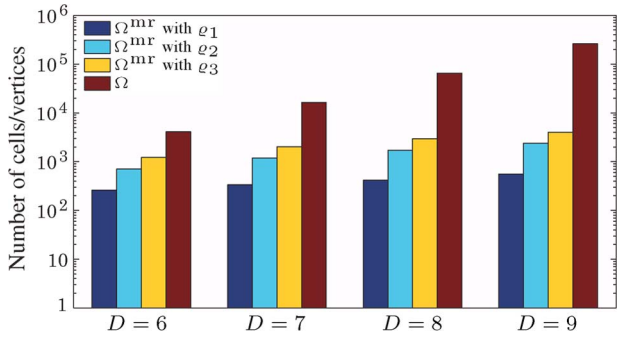


Fig. 13. Comparison of the number of vertices in the multiresolution cell decomposition graphs with different window functions and with the finest level cell decomposition Ω.

C. Performance of the Path-Planning Scheme

Fig. 13 shows the comparison of the (average) number of vertices in the graphs associated with the multiresolution cell decompositions corresponding to different window functions and with the finest level cell decomposition Ω. As expected, window function ϱ_3, which has the largest neighborhood of high-fidelity approximation of the environment (i.e., a large number of significant detail coefficients), results in cell decompositions with the largest number of cells among the three multiresolution decompositions. Note, however, that the numbers of vertices in each of the three multiresolution decompositions

[2]Note that Fig. 12 shows a large number of cases of low suboptimality for *all* values of λ_2.

[3]This worst case estimate is necessary for completeness of the path planner.

TABLE II
COMPARISONS BETWEEN VARIOUS MULTIRESOLUTION MOTION PLANNERS

	Decomposition depends on	Completeness	Dynamical constraints
Ref. [4]	Environment map	Yes	No
Ref. [5]	Environment map	Yes[a]	No
Ref. [10]	Environment map	Yes[a]	No
Ref. [7]	Environment map	Yes[a]	No
Ref. [6]	Vehicle location	No[a]	No
Ref. [9]	Vehicle location	No[a]	No
Ref. [8]	Environment map and vehicle location	No[a]	Yes[b]
Ref. [11]	Vehicle location	No[a]	Yes
Proposed	Vehicle location	Yes	Yes

[a] Not addressed.
[b] Dynamical constraints addressed separately from high-level path planning.

are of the same order of magnitude, whereas the numbers of vertices in $\bar{\mathcal{G}}$ are one to three orders of magnitude greater than those in the multiresolution cell decomposition graphs. For instance, with $D = 9$, the number of vertices in $\bar{\mathcal{G}}$ was 262 144, whereas the average number of vertices was only 561 for the multiresolution cell decomposition with window ϱ_1. In this context, one may recall that the time complexity of the execution on a sparse graph $\mathcal{G} = (V, E)$ of Dijkstra's algorithm and the A* algorithm is $O(|V| \log |V|)$, whereas the memory complexity is $O(|V|)$ [2].

D. Comparisons With Other Multiresolution Path Planners

In this section, we present a comparison between the proposed multiresolution motion-planning scheme and some of the standard multiresolution planners reported in the literature (e.g., [6], [10], and [11]). The comparison will be based on the numbers of vertices and edges of the resulting graph, as the main objective of all multiresolution planners is to provide graph representations of the environment with low complexity. These graphs are then searched using standard algorithms such as the A* algorithm. This allows a fair comparison of the available multiresolution motion cell decompositions, as the particular *search* algorithms of the resulting graph are the same across all such schemes.

The multiresolution approximations of the environment reported in the literature belong to either of the following two broad classes: those primarily governed by the environment map and those primarily governed by the vehicle's location in the environment. The former ones (such as those based on quadtree decompositions [4]) ensure that the resulting path will be entirely obstacle free, but they tend to create larger graphs. The latter methods (e.g., [6]) result in smaller graphs, but obstacle-free cells are ensured only in the immediate vicinity of the vehicle's location, and replanning of paths is necessary as the vehicle moves through the environment. The main disadvantage of multiresolution planners that use such vehicle location-dependent decompositions is that they are prone to a lack of completeness. The proposed approach in this paper belongs to the second class of planners, but we provide a guarantee of its completeness.

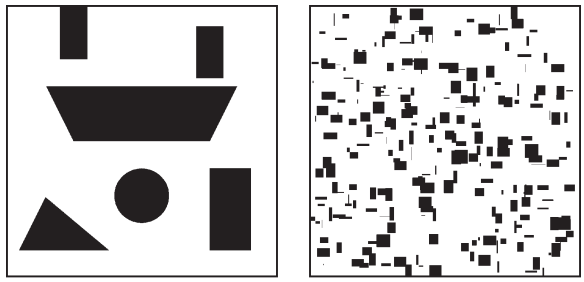


Fig. 14. Environment maps used for comparative analysis. (a) Sparse large obstacles. (b) Cluttered small obstacles.

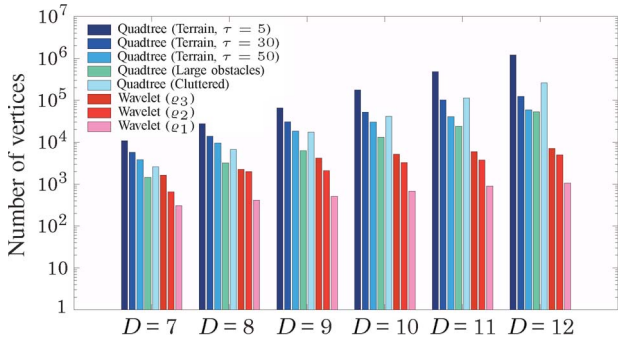


Fig. 15. Comparison in the numbers of vertices of the resultant cell decompositions of the proposed approach against the quadtree decomposition.

Table II provides a qualitative comparison between the proposed work and the various multiresolution motion-planning schemes reported in the literature.

To quantitatively illustrate our claim (echoed also in [6]) that environment-dependent cell decompositions usually consist of significantly more cells than vehicle location-dependent cell decompositions, we chose three environment maps, i.e., a terrainlike environment similar to Fig. 2(a), an environment consisting of a small number of large obstacles, and an environment consisting of a large number of small obstacles (see Fig. 14). We used the basic quadtree decomposition described in [4] as the basic example of an environment-dependent multiresolution cell decomposition. Fig. 15 shows a comparison of the number of vertices in the cell decomposition graphs arising from this quadtree decomposition, using different thresholds[4] and levels of decomposition, against those arising from the proposed wavelet-based decomposition, the latter using different window functions.

The number of cells from the environment-dependent quadtree decomposition for the environment with a sparse obstacle distribution is an order of magnitude larger than that obtained by the proposed vehicle-dependent decomposition. Improvements to environment-dependent decompositions, such as allowing for large "gray" cells, can reduce the number of vertices by an order of magnitude [4], and hence, for this environment with sparse obstacles, we may consider the two schemes of decomposition to be equally efficient. However, for

[4]The threshold τ, applicable for the terrain-like environment map, governs the quadtree decomposition as follows: A cell is further decomposed if and only if the difference in the maximum and minimum intensities of pixels within that cell exceeds τ.

the terrain-like environment, where cell intensities take values in the interval [0, 1] (as opposed to binary values in the previous case), the difference in the number of cells is up to three orders of magnitude. A similar observation holds true for the highly cluttered environment [see Fig. 14(b)]. Such a large difference in the number of vertices in the cell decomposition graph may render infeasible the implementation of environment-dependent decompositions (particularly for large environments) for on-board computing systems with limited processing and memory resources.

Cell decompositions governed by the vehicle's location in the environment are numerically more efficient. However, as the corresponding graph changes with the vehicle's motion, the completeness of the overall path-planning scheme is *not* guaranteed *a priori*, although the graph search algorithm used at *each iteration* may be complete. Specifically, the path planner can get trapped in loops where it visits a certain sequence of cells *ad infinitum*.

In addition to showing completeness, the most significant difference of our work compared with other similar works on multiresolution path planning in the literature is the systematic incorporation of vehicle kinematic/dynamic constraints in path planning. In particular, we note in the third column of Table II that most of the other works do not address vehicle kinematic/dynamic constraints. Reference [8] discusses a receding horizon scheme for incorporating vehicle dynamical constraints, but this scheme is disconnected from the high-level discrete path planner. Consequently, there is no consistency between the two levels of planning (i.e., a guarantee that the path found by the high-level planner can be feasibly traversed by the vehicle). This issue is addressed in this paper via the H-cost motion-planning approach discussed in Section IV. In [28], we have also provided extensive comparative analysis establishing the superiority of the H-cost approach using uniform cell decompositions over state-of-the-art randomized sampling-based algorithms.

E. Multiresolution Motion-Planning Example

To illustrate a typical application of the overall multiresolution motion-planning scheme that incorporates the vehicle dynamic constraints, we consider the problem of navigating an aircraft among a topographic relief of varying elevation. The equations of motion and the implementation of a local trajectory generation algorithm for this vehicle are described in detail in [30].

Fig. 16 shows the result of the numerical simulation of the proposed motion planner for the aircraft navigational model. Aircraft speed was assumed to be constant, and the control input is the heading angle, which is controlled by the bank angle. To show the flexibility of the algorithm in incorporating dynamic constraints, an asymmetric bound on the bank angle control input was assumed (for instance, owing to an aileron failure [38]) as follows: $\phi_{\min} = -45^\circ$ and $\phi_{\max} = 20^\circ$. The objective was to minimize a cost defined on the environment (indicated by regions of different intensities in Fig. 16, where the darker regions correspond to higher costs).

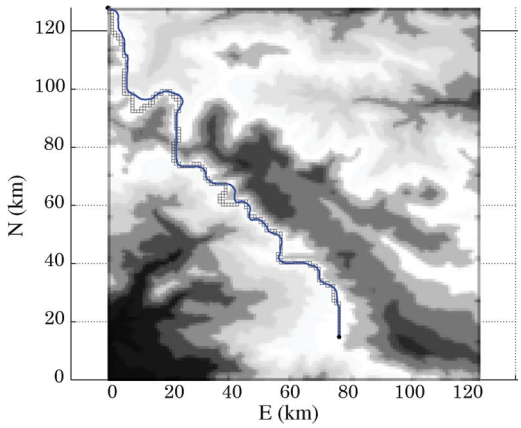


Fig. 16. Result of motion-planning simulation using the aircraft navigational model. The blue curve corresponds to the resultant state trajectory, whereas the channel of cells in black is the result of executing the A* algorithm (without vehicle dynamical constraints). The initial position is at the top left corner.

Fig. 17 illustrates an intermediate iteration of this simulation example. Fig. 17(a) shows the cells of size at most $\bar{d}$, with the boundary cells indicated in red. The sequence of cells outlined in blue and the blue-colored curve within this cell sequence are the results of the H-cost motion planner. The yellow-colored cells indicate the vertices explored during the H-cost search. Fig. 17(b) shows the overall multiresolution cell decomposition at the same iteration. The blue-colored cells indicate the optimal path to the goal from the boundary cell chosen by the H-cost motion planner. The blue-colored curve in Fig. 17(b) indicates the geometric path traversed by the vehicle in previous iterations.

VI. Conclusion

In this paper, we have introduced a multiresolution path- and motion-planning scheme that considers accurate models of the environment and the vehicle dynamics only for local planning and considers coarse models for global planning. The proposed path planner uses cell decompositions obtained from multiresolution wavelet processing of the environment map. Specifically, we introduced a scheme that encodes in the DWT coefficients of the environment map all the necessary information about these cell decompositions. We provided a method for fast incremental updates to these cell decompositions in accordance with changes in the vehicle's location.

We rigorously proved the completeness of the proposed path-planning scheme, where such proofs of completeness have been so far absent in the literature related to dynamic multiresolution path planning. Furthermore, we provided numerical simulation data demonstrating that the proposed path planner results in near-optimal paths in a large majority of the simulated cases. Finally, we proposed a method, based on the H-cost approach, for incorporating vehicle kinematic and dynamic constraints for planning the vehicle's motion in its immediate vicinity.

Future work includes generalizations of the proposed multiresolution path-planning scheme to allow a dynamic "window" and the incorporation of a D*-like update of the multiresolution environment map based on newly acquired information. The generalization of the wavelet-based cell

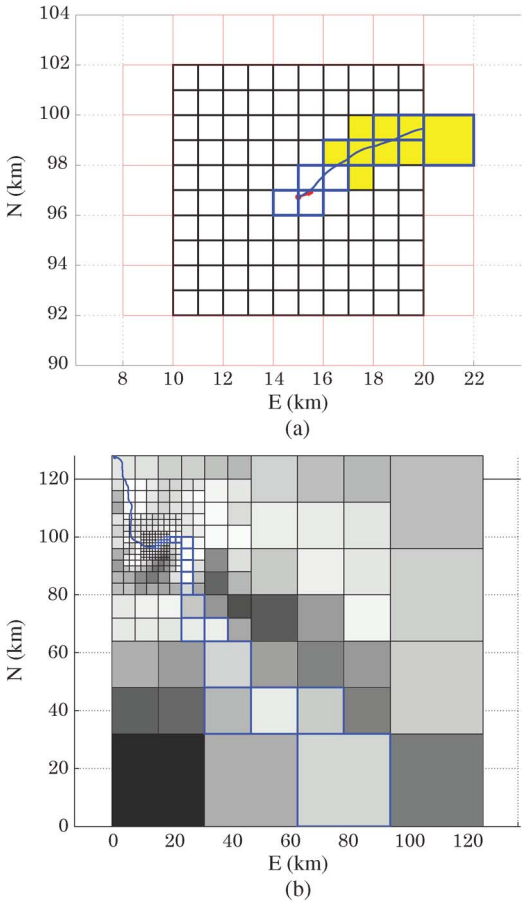


Fig. 17. Illustration of an intermediate iteration of the overall motion planner. (a) Local perspective: the vehicle's configuration is indicated in red. (b) Global perspective.

decomposition algorithm for wavelet families other than the Haar family will be also beneficial, particularly when the environment map is processed and encoded using other wavelets.

APPENDIX A

We provide a series of technical results, concerning the algorithm in Fig. 5, that are needed for the proof of Proposition 1. To this end, we associate with each path $\pi_n(j_n, i_{G,n}) = \{i_0, \ldots, i_{P(n)}\}$ in $\mathcal{G}(n)$ set $\mathcal{W}(\pi_n)$ defined by

$$\mathcal{W}(\pi_n) := \bigcup_{p=0}^{P(n)} W\left(i_p, V(n)\right). \tag{A.1}$$

The algorithm is said to *meet a setback* at iteration n if there exists no obstacle-free path $\pi_n(j_n, i_{G,n})$ in $\mathcal{G}(n)$ satisfying $\mathcal{W}(\pi_n) \subseteq \mathcal{W}(\pi^*_{n-1})$.

Proposition A.1: Let $\bar{j} \in \bar{V}$, and $\mathcal{A} = \text{MR–APPROX}(\bar{j})$. Let Ω^{mr} and $\mathcal{G} = (V, E)$ be, respectively, the multiresolution cell decomposition and the topological graph associated with $\mathcal{A}$. If there exists an obstacle-free path in $\bar{\mathcal{G}}$ from $\bar{j}$ to $\bar{i}_G$, then there exists an obstacle-free path in $\mathcal{G}$ from $j := \mathsf{vert}(\mathsf{cell}(\bar{j}; \Omega^{\text{mr}}); \mathcal{G})$ to i_G, where $i_G \in V$ is the unique vertex that satisfies $\bar{i}_G \in W(i_G, V)$.

Proof: Let $\bar{\pi}(\bar{j}, \bar{i}_G) = (\bar{j}_0, \ldots, \bar{j}_{\bar{P}})$ be an obstacle-free path in $\bar{\mathcal{G}}$ from $\bar{j}_0 = \bar{j}$ to $\bar{j}_{\bar{P}} = \bar{i}_G$. For each $m = 0, \ldots, \bar{P}$, there exists a unique set $W_m \in \{W(j, V)\}_{j \in V}$ such that $\bar{j}_m \in W_m$. Let $i_m \in V$ be such that $W_m = W(i_m, V)$. Since $\bar{\pi}$ is a path in $\bar{\mathcal{G}}$, $(\bar{j}_{m-1}, \bar{j}_m) \in \bar{E}$ for each $m = 1, \ldots, \bar{P}$, and it follows that either $W_{m-1} = W_m$ or $(i_{m-1}, i_m) \in E$. Thus, $\pi(j, i_G) := \{j_0, \ldots, j_P\}$, where $P \leqslant \bar{P}$, is a path in $\mathcal{G}$.

To show that the path π is also obstacle free in $\mathcal{G}$, we note that, since $\bar{\pi}$ is obstacle free in $\bar{\mathcal{G}}$, $F(\bar{j}_m) \leqslant 1 - \varepsilon$, for each $m = 0, 1, \ldots, \bar{P}$. It follows by (5) that $\hat{F}(\mathsf{cell}(j_m; \Omega^{\text{mr}})) < (1 - \varepsilon)$ for each $m = 0, 1, \ldots, P$ and by (7) that $\mathcal{J}(\pi) < M$, i.e., π is an obstacle-free path. ■

Corollary A.1: If there exists an obstacle-free path in $\bar{\mathcal{G}}$ from the initial vertex $\bar{i}_S$ to the goal vertex $\bar{i}_G$, then the cost of the initial path π^*_0 computed by the algorithm is finite.

Proof: By Proposition A.1, if there exists an obstacle-free path in $\bar{\mathcal{G}}$ from $\bar{j}$ to $\bar{i}_G$, then there exists an obstacle-free path $\pi^*_0(i_S, i_{G,0})$ in $\mathcal{G}(0)$ from vertex $i_S := \mathsf{vert}(\mathsf{cell}(\bar{i}_S); \Omega); \mathcal{G}(0)$ to vertex $i_{G,0}$, where $i_{G,0} \in V(0)$ is the unique vertex that satisfies $\bar{i}_G \in W(i_{G,0}, V(0))$. Because π^*_0 is obstacle free, $\mathcal{J}(\pi^*_0) < M$, i.e., $\mathcal{J}(\pi^*_0)$ is finite. ■

Proposition A.2: Suppose that the algorithm does not meet a setback at iteration $n \in \mathbb{N}$ of its execution, and suppose also that $\text{VISITED}(\bar{j}_n) = 0$. If there exists a path in the graph $\mathcal{G}(n)$ from vertex $j_n = \mathsf{vert}(\mathsf{cell}(\bar{j}_n; \Omega^{\text{mr}}(n)); \mathcal{G}(n))$ to vertex $i_{G,n}$, then $\mathcal{K}_G(\bar{j}_{n-1}) - \mathcal{K}_G(\bar{j}_n) \geqslant \lambda_2$, where $i_{G,n} \in V(n)$ is the unique vertex that satisfies $\bar{i}_G \in W(i_{G,n}, V(n))$.

Proof: Let $\pi^*_n(j_n, i_{G,n}) = (j_0, \ldots, j_{P(n)})$ denote the optimal path in the graph $\mathcal{G}(n)$ computed by the algorithm at Line 11. First, suppose that the cell decomposition $\Omega^{\text{mr}}(n)$ is identical to the cell decomposition $\Omega^{\text{mr}}(n-1)$ (in particular, $i_{G,n-1} = i_{G,n}$). If there exists a path in $\mathcal{G}(n)$ from j_n to $i_{G,n}$, then there exists an optimal path in $\mathcal{G}(n)$ from j_n to $i_{G,n}$ because $\mathcal{G}(n)$ is finite. Then, by Bellman's principle of optimality, the path $\pi^*_{n-1}(j_{n-1}, i_{G,n-1}) = (i_0, \ldots, i_{P(n-1)})$, computed at iteration $n-1$ of the algorithm, contains the path π^*_n, with $P(n) = P(n-1) - 1$, and $j_{m-1} = i_m$ for each $m = 1, 2, \ldots, P(n)$, and hence, $\mathcal{J}(\pi^*_n) \leqslant \mathcal{J}(\pi^*_{n-1})$.

Next, suppose that the cell decomposition $\Omega^{\text{mr}}(n)$ is not identical to the cell decomposition $\Omega^{\text{mr}}(n-1)$. Let $\pi_n(j_n, i_{G,n})$ and $\pi_{n-1}(j_{n-1}, i_{G,n-1})$ be paths in graphs $\mathcal{G}(n)$ and $\mathcal{G}(n-1)$, respectively. If $\mathcal{W}(\pi_n) \subseteq \mathcal{W}(\pi_{n-1})$, then due to the second and third terms on the right-hand side of (7), $\mathcal{J}(\pi_n) \leqslant \mathcal{J}(\pi_{n-1})$. In particular, if $\mathcal{W}(\pi^*_n) \subseteq \mathcal{W}(\pi^*_{n-1})$, then $\mathcal{J}(\pi^*_n) \leqslant \mathcal{J}(\pi^*_{n-1})$.

Now, suppose $\mathcal{W}(\pi^*_n) \not\subseteq \mathcal{W}(\pi^*_{n-1})$. Let $\pi_n(j_n, i_{G,n})$ be any path in $\mathcal{G}(n)$ from u_n to $i_{G,n}$ satisfying $\mathcal{W}(\pi_n) \subseteq \mathcal{W}(\pi^*_{n-1})$. There exists at least one such path π_n in $\mathcal{G}(n)$ because the algorithm does not meet a setback at iteration n. By the arguments in the preceding paragraph, $\mathcal{J}(\pi_n) \leqslant \mathcal{J}(\pi^*_{n-1})$. Furthermore, because π^*_n is an optimal path in $\mathcal{G}(n)$ from j_n to $i_{G,n}$, $\mathcal{J}(\pi^*_n) \leqslant \mathcal{J}(\pi_n)$, and it follows that $\mathcal{J}(\pi^*_n) \leqslant \mathcal{J}(\pi^*_{n-1})$.

Finally, note that the cell corresponding to the first vertex $j_0 \in V(n)$ in the path π^*_n is the same as the cell corresponding to the second vertex $i_1 \in V(n-1)$ in π^*_{n-1}, and furthermore, this cell corresponds to vertex $\bar{j}_n \in \bar{V}$. Then, $\mathcal{K}_G(\bar{j}_{n-1}) - \mathcal{K}_G(\bar{j}_n) = \mathcal{J}(\pi^*_{n-1}) - \mathcal{J}(\pi^*_n) \geqslant \bar{g}(\bar{j}_{n-1}, \bar{j}_n) \geqslant \lambda_2$ by (6). ■

Proposition A.3: Let $\bar{j}$ be an arbitrary vertex in $\bar{V}$. Then, either the algorithm never visits $\bar{j}$ or *the algorithm visits $\bar{j}$ finitely many times*.

Proof: Suppose, for the sake of contradiction, that the algorithm visits $\bar{j} \in \bar{V}$ infinitely many times at iterations $n_1, \ldots, n_k \ldots$, i.e., $\bar{j}_{n_1} = \bar{j}_{n_2} \ldots = \bar{j}$. By Line 8, $\mathcal{K}_G(\bar{j}_{n_k}) - \mathcal{K}_G(\bar{j}_{n_{k-1}}) > 0$, and hence, there exists $N \in \mathbb{N}$, such that $\mathcal{K}_G(\bar{j}_{n_N}) \geqslant M$. It follows that the algorithm terminates in at most n_N iterations, which is a contradiction. ■

Propositon A.4: Let $\pi_n^*(j_n, i_{G,n}) = (j_0, \ldots, j_{P(n)})$ be the path found by the algorithm at Line 8 or *Line 11 in iteration $n \in \mathbb{N}$, and suppose there exists an obstacle-free path in $\bar{\mathcal{G}}$ from $\bar{j}_n$ to $\bar{i}_G$ that is contained within set $\mathcal{W}(\pi_n^*)$. Then, the algorithm does not visit vertex $\bar{j}_n$ at any future iteration*.

Proof: We note that the cell corresponding to the second vertex in the path π_n^* is a cell at the finest resolution, and hence, $W(j_1, V(n)) = \bar{j}_{n+1}$. Then, it follows due to (A.1) and due to the hypothesis that there exists an obstacle-free path $\bar{\pi}(\bar{j}_n, \bar{i}_G) = (\bar{i}_0, \ldots, \bar{i}_{\bar{P}})$ in $\bar{\mathcal{G}}$ from $\bar{j}_n$ to $\bar{i}_G$ such that $\bar{i}_1 = \bar{j}_{n+1}$. Thus, there exists an obstacle-free path in $\bar{\mathcal{G}}$ from $\bar{j}_{n+1}$ to $\bar{i}_G$; in particular, $(\bar{i}_1, \ldots, \bar{i}_{\bar{P}})$ is such a path. Then, it follows by Proposition A.1 that the algorithm does not execute Line 18 at iteration $n+1$.

By the preceding arguments, the following statement is true: If there exists an obstacle-free path in $\bar{\mathcal{G}}$ from $\bar{j}_{n+k}$ to $\bar{i}_G$ contained within π_{n+k}^*, then the algorithm does not execute Line 18 at iteration $n+k+1$.

Now suppose, for the sake of contradiction, that there exists $\ell > 1$ such that the algorithm visits vertex $\bar{j}_n$ again at iteration $n+\ell$, i.e., $\bar{j}_n = \bar{j}_{n+\ell}$ and $\bar{j}_{n+1} = \bar{j}_{n+\ell-1}$. Then, there exists $m < \ell$ such that for each $k = m, m+1, \ldots, \ell$, the algorithm executes Line 18 at iteration $n+k$, i.e., $\bar{j}_{n+k+1} = b(\bar{j}_{n+k})$. Due to the statement in the preceding paragraph, it follows that either there exists no obstacle-free path in $\bar{\mathcal{G}}$ from $\bar{j}_{n+k}$ to $\bar{i}_G$ or the second vertex of every obstacle-free path in $\bar{\mathcal{G}}$ from $\bar{j}_{n+k}$ to $\bar{i}_G$ is $b(\bar{j}_{n+k})$. However, neither of these holds true for $k = \ell - 1$ because we showed earlier that $(\bar{i}_1, \ldots, \bar{i}_{\bar{P}})$ is an obstacle-free path in $\bar{\mathcal{G}}$ from $\bar{j}_{n+1} = \bar{j}_{n+\ell-1}$ to $\bar{i}_G$, and this path does not contain $\bar{j}_n$. Thus, we arrive at a contradiction, and it follows that there exists no $\ell > 1$ such that $\bar{j}_n = \bar{j}_{n+\ell}$, i.e., the algorithm does not visit $\bar{j}_n$ at any future iteration. ■

Proof of Proposition 1: Note that because the set of vertices in $\bar{V}$ is finite, it follows by Proposition A.3 that the algorithm terminates after a finite number of iterations.

To show completeness, suppose first that there exists an obstacle-free path in $\bar{\mathcal{G}}$ from $\bar{i}_S$ to $\bar{i}_G$.

Suppose first that the algorithm never visits any vertex in $\bar{V}$ more than once and that the algorithm does not meet any setbacks. By Proposition A.2, $\mathcal{K}_G(\bar{j}_{n-1}) - \mathcal{K}_G(\bar{j}_n) \geqslant \lambda_2$, and sequence $\mathcal{K}_G(\bar{j}_n)$ decreases strictly monotonically. Since $\mathcal{K}_G(\bar{j}_n) \geq 0$ for each $n \in \mathbb{N}$ and $\mathcal{K}_G(\bar{j}_1)$ is finite (by Corollary A.1), there exists $Q \in \mathbb{N}$, such that $\mathcal{K}_G(\bar{j}_n) = 0$ for each $n \geqslant Q$. It follows from Line 22 in Fig. 5 that the algorithm terminates after Q iterations, and since $\mathcal{K}_G(\bar{j}_Q) = 0$, the algorithm visits the goal $\bar{i}_G$ at iteration Q.

Next, suppose that the algorithm visits some vertices in $\bar{V}$ multiple times and that the algorithm never meets any setbacks. Note that the number of multiply visited vertices is finite because the algorithm terminates after a finite number of iterations. Then, either of the following statements holds: a) the algorithm terminates at iteration $Q \in \mathbb{N}$, such that $\bar{j}_Q$ is a multiply visited vertex or b) there exists $Q \in \mathbb{N}$ such that for each $n = Q+1, Q+2, \ldots$, vertex $\bar{j}_n$ is visited exactly once by the algorithm. If statement (a) holds, then $\bar{j}_Q \neq \bar{i}_G$ due to Lines 3 and 22 in Fig. 5, which, in turn, implies that the algorithm reports failure in Line 16 in Fig. 5. It follows by Line 15 that $\bar{j}_Q = \bar{i}_S$. Then, by Proposition A.1 and A.4, there exists no admissible path in $\bar{\mathcal{G}}$ from $\bar{i}_S$ to $\bar{i}_G$, which is a contradiction. On the other hand, if statement (b) holds, then by the monotonicity arguments in the preceding paragraph, the algorithm visits the goal after a finite number of iterations after iteration Q.

Next, suppose that the algorithm never visits any vertex in $\bar{V}$ more than once, and suppose that the algorithm meets some setbacks. The number of setbacks met by the algorithm is finite because the algorithm terminates in a finite number of iterations. Then, either of the following statements holds: c) the algorithm terminates at iteration $Q \in \mathbb{N}$ such that the algorithm meets a setback at iteration Q or d) there exists $Q \in \mathbb{N}$ such that for each $n = Q+1, Q+2, \ldots$, such that the algorithm does not meet any setbacks after iteration Q. Statement (c) leads to the same contradiction that follows statement (a), whereas statement (d) leads to the same conclusion that follows statement (b).

Next, suppose that the algorithm visits some vertices multiple times and that the algorithm meets some setbacks. We may combine the arguments in the preceding two paragraphs to conclude that either the algorithm visits the goal after a finite number of iterations or (by contradiction) that there exists no obstacle-free path in $\bar{\mathcal{G}}$ from $\bar{i}_S$ to $\bar{i}_G$.

Finally, suppose that there exists no obstacle-free path in the graph $\bar{\mathcal{G}}$ from the initial vertex $\bar{i}_S$ to the goal vertex $\bar{i}_G$. The set of vertices $\bar{V}$ is finite; hence, it follows by Proposition A.3 that the algorithm terminates after a finite number of iterations. Suppose, on the contrary, that the algorithm erroneously finds a path $\bar{\pi}$ from the initial vertex $\bar{i}_S$ to the goal vertex $\bar{i}_G$. Then, $\bar{\mathcal{J}}(\bar{\pi}) > M$ since $\bar{\pi}$ is not obstacle free. It follows by Line 24 in Fig. 5 that $\bar{\mathcal{J}}(\bar{\pi}) > M$ at some intermediate iteration of the algorithm. However, by Line 3 in Fig. 5, the algorithm terminates whenever $\bar{\mathcal{J}}(\bar{\pi}) > M$, thus leading to a contradiction. Thus, the algorithm does not erroneously find a path from $\bar{i}_S$ to $\bar{i}_G$ if no obstacle-free path exists, and by Line 26 in Fig. 5, it reports failure in this case. ■

APPENDIX B

Multiresolution analysis (MRA) of a scalar function of one variable is the construction of a hierarchy of functional approximations by projecting the function onto a sequence of nested linear spaces. The DWT provides a framework for such MRA of a function. In this framework, the sequence of nested linear spaces is generated by translated and scaled versions of two scalar functions ϕ and ψ of unit energy, called the *scaling function* and *mother wavelet*, respectively, which satisfy the so-called *orthogonality and dilation equations* (cf., [13]). For each $m, k \in \mathbb{Z}$, we define scalar functions $\phi_{m,k}$ and $\psi_{m,k}$ by $\phi_{m,k}(t) := \sqrt{2^m}\phi(2^m t - k)$ and $\psi_{m,k}(t) := \sqrt{2^m}\psi(2^m t - k)$, respectively. The *DWT* of a scalar function

$f \in \mathbb{L}^2(\mathbb{R})$ is defined by $a_{m_0,k} := \langle \phi_{m_0,k}(t), f(t) \rangle$, and $d_{m,k} := \langle \psi_{m,k}(t), f(t) \rangle$, where $m_0 \in \mathbb{Z}$. The 1-D *reconstruction equation* is

$$f(t) = \sum_{k=-\infty}^{\infty} a_{m_0,k} \phi_{m_0,k}(t) + \sum_{m=m_0}^{\infty} \sum_{k=-\infty}^{\infty} d_{m,k} \psi_{m,k}(t).$$

Scalars $a_{m_0,k}$ and $d_{m,k}$ are known as *approximation* and *detail* coefficients, respectively.

For the 2-D extension of the 1-D DWT, a scaling function is defined by $\Phi_{m,k,\ell}(x,y) := \phi_{m,k}(x)\phi_{m,\ell}(y)$, and three wavelets $\Psi^1_{m,k,\ell}, \ldots, \Psi^3_{m,k,\ell}$ are similarly defined by products of the 1-D scaling function and wavelet. The 2-D DWT coefficients of a scalar function $F \in \mathbb{L}^2(\mathbb{R}^2)$ are $a_{m_0,k,\ell} := \langle \Phi_{m_0,k,\ell}(x,y), F(x,y) \rangle$ and $d^i_{m,k,\ell} := \langle \Psi^i_{m,k,\ell}(x,y), F(x,y) \rangle$, for $i = 1,2,3, k, \ell \in \mathbb{Z}$, and $m \geq m_0 \in \mathbb{Z}$. The corresponding 2-D *reconstruction equation* is defined analogous to the 1-D case.

An example of a pair of scaling function and wavelet is the Haar family [12]. For the 1-D Haar family, functions $\phi_{m,k}$ and $\psi_{m,k}$ are compactly supported over the interval $I_{m,k} := [2^{-m}k, 2^{-m}(k+1)]$, and by consequence, functions $\Phi_{m,k,\ell}$ and $\Psi_{m,k,\ell}$ are compactly supported over

$$S_{m,k,\ell} := I_{m,k} \times I_{m,\ell}, \tag{B.2}$$

which is a square of size 2^{-m} for $k, \ell \in \mathbb{Z}$.

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*. Cambridge, MA: MIT Press, 2005.

[2] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2006.

[3] H. Samet, "The quadtree and related hierarchical data structures," *Comput. Surv.*, vol. 16, no. 2, pp. 187–260, Jun. 1984.

[4] S. Kambhampati and L. S. Davis, "Multiresolution path planning for mobile robots," *IEEE J. Robot. Autom.*, vol. RA-2, no. 3, pp. 135–145, Sep. 1986.

[5] H. Noborio, T. Naniwa, and S. Arimoto, "A quadtree-based path planning algorithm for a mobile robot," *J. Robot. Syst.*, vol. 7, no. 4, pp. 555–574, Aug. 1990.

[6] S. Behnke, "Local multiresolution path planning," in *Proc. 7th RoboCup Int. Symp.*, vol. 3020, *Lecture Notes in Artificial Intelligence*, 2004, pp. 332–343.

[7] J. Y. Hwang, J. S. Kim, S. S. Lim, and K. H. Park, "A fast path planning by path graph optimization," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 33, no. 1, pp. 121–127, Jan. 2003.

[8] R. J. Prazenica, A. J. Kurdila, R. C. Sharpley, and J. Evers, "Multiresolution and adaptive path planning for maneuver of micro-air-vehicles in urban environments," in *Proc. AIAA Guidance, Navig., Control Conf. Exhib.*, San Francisco, CA, 2005, pp. 1–12.

[9] C.-T. Kim and J.-J. Lee, "Mobile robot navigation using multi-resolution electrostatic potential field," in *Proc. 31st Annu. Conf. IECON*, 2005, pp. 1–5.

[10] B. J. H. Verwer, "A multiresolution workspace, multiresolution configuration space approach to solve the path planning problem," in *Proc. IEEE Int. Conf. Robot. Autom.*, 1990, pp. 2107–2112.

[11] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. N. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. M. Howard, S. Kolski, A. Kelly, M. Likhachev, M. McNaughton, N. Miller, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, B. Salesky, Y.-W. Seo, S. Singh, J. Snider, A. Stentz, W. Whittaker, Z. Wolkowicki, J. Ziglar, H. Bae, T. Brown, D. Demitrish, B. Litkouhi, J. Nickolaou, V. Sadekar, W. Zhang, J. Struble, M. Taylor, M. Darms, and D. Ferguson, "Autonomous driving in urban environments: Boss and the Urban Challenge," *J. Field Robot.*, vol. 25, no. 8, pp. 425–466, Aug. 2008. [Online]. Available: http://dx.doi.org/10.1002/rob.20255

[12] I. Daubechies, *Ten Lectures on Wavelets*. Philadelphia, PA: SIAM, 1994.

[13] R. M. Rao and A. S. Bopardikar, *Wavelet Transforms—Introduction to Theory and Applications*. Reading, MA: Addison-Wesley, 1998.

[14] W. L. D. Lui and R. Jarvis, "A pure vision-based approach to topological Simultaneous Localization And Mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Taipei, Taiwan, Oct. 18–22, 2010, pp. 3784–3791.

[15] J. Ilkyun, J. Seewong, and K. Youngouk, "Mobile robot navigation using difference of wavelet SIFT," in *Proc. 2nd Int. Conf. Mach. Vis.*, 2009, pp. 286–292.

[16] J. T. Cho and B. H. Nam, "A study on the fuzzy control navigation and the obstacle avoidance of mobile robot using camera," in *Proc. IEEE Syst., Man, Cybern. Conf.*, 2000, vol. 4, pp. 2993–2997.

[17] M. Shim, J. Kurtz, and A. Laine, "Multi-resolution stereo algorithm via wavelet representations for autonomous navigation," in *Proc. SPIE*, Orlando, FL, Apr. 1999, vol. 3723, pp. 319–328.

[18] B. Cipra, "Parlez-vous wavelets?" in *What's Happening in the Mathematical Sciences*. Providence, RI: AMS, 1994.

[19] M. Yguel, O. Aycard, and C. Laugier, "Wavelet occupancy grids: A method for compact map building," in *Field and Service Robotics*, P. Corke and S. Sukkarieh, Eds. Berlin, Germany: Springer-Verlag, 2006, pp. 219–230.

[20] L. Wei and T. F. Fwa, "Characterizing road roughness by wavelet transform," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1869, pp. 152–158, 2004.

[21] T. Wiesemann, J. Schiefele, and J. Bader, "Multi-resolution terrain depiction and airport navigation function on an embedded SVS," in *Proc. SPIE Enhanced Synthet. Vis.*, J. G. Verly, Ed., 2002, vol. 4713, pp. 106–117.

[22] C. Paulson, S. Ezekiel, and D. Wu, "Wavelet-based image registration," in *Proc. SPIE Evol. Bio-Inspired Comput.—Theory Applications IV*, T. H. O'Donnel, M. Blowers, and K. Priddy, Eds., 2010, vol. 7704, pp. M-1–M-12.

[23] P. Tsiotras and E. Bakolas, "A hierarchical on-line path planning scheme using wavelets," in *Proc. Eur. Control Conf.*, Kos, Greece, Jul. 2–5, 2007, pp. 2806–2812.

[24] D. Jung, "Hierarchical path planning and control of a small fixed-wing UAV: Theory and experimental validation," Ph.D. dissertation, Georgia Inst. Technol., Atlanta, GA, 2007.

[25] D. K. Pai and L.-M. Reissell, "Multiresolution rough terrain motion planning," *IEEE Trans. Robot. Autom.*, vol. 14, no. 1, pp. 19–33, Feb. 1998.

[26] L. Carrioli, "Unsupervised path planning of many asynchronously self-moving vehicles," in *Proc. IEEE/RSJ Int. Workshop IROS*, 1991, pp. 555–559.

[27] B. Sinopoli, M. Micheli, G. Donato, and T. J. Koo, "Vision based navigation for an unmanned aerial vehicle," in *Proc. IEEE Conf. Robot. Autom.*, 2001, pp. 1757–1764.

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

[29] R. V. Cowlagi and P. Tsiotras, "Kinematic feasibility guarantees in geometric path planning using history-based transition costs over cell decompositions," in *Proc. Amer. Control Conf.*, MD, Jun. 30/Jul. 2, 2010, pp. 5388–5393.

[30] R. V. Cowlagi, "Hierarchical motion planning for autonomous aerial and terrestrial vehicles," Ph.D. dissertation, Georgia Inst. Technol., Atlanta, GA, 2011.

[31] C. Belta, A. Bicchi, M. Egerstedt, E. Frazzoli, E. Klavins, and G. J. Pappas, "Symbolic planning and control of robot motion," *IEEE Robot. Autom. Mag.*, vol. 14, no. 1, pp. 61–70, Mar. 2007.

[32] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, Jun. 1989.

[33] U. Huckenbeck, *Extremal Paths in Graphs*. Berlin, Germany: Akademie Verlag, 1997.

[34] S. Winter, "Modeling costs of turns in route planning," *GeoInformatica*, vol. 6, no. 4, pp. 345–361, Dec. 2002.

[35] E. Rippel, A. Bar-Gill, and N. Shimkin, "Fast graph-search algorithms for general aviation flight trajectory generation," *J. Guid. Control Dyn.*, vol. 28, no. 4, pp. 801–811, Jul./Aug. 2005.

[36] Y. Kuwata and J. P. How, "Stable trajectory design for highly constrained environments using receding horizon control," in *Proc. Amer. Control Conf.*, Boston, MA, Jun. 30/Jul. 2, 2004, pp. 902–907.

[37] S. Bereg and D. Kirkpatrick, "Curvature-bounded traversals of narrow corridors," in *Proc. 21st Annu. Symp. Comput. Geom.*, Pisa, Italy, 2005, pp. 278–287.

[38] E. Bakolas and P. Tsiotras, "Optimal synthesis of the asymmetric sinistral/dextral Markov–Dubins problem," *J. Optim. Theory Appl.*, vol. 150, no. 2, pp. 233–250, Aug. 2011.

Raghvendra V. Cowlagi (M'05) received the Ph.D. degree in aerospace engineering from Georgia Institute of Technology, Atlanta, in 2011.

He is currently a Postdoctoral Researcher with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge. His research interests include autonomous vehicles, motion planning, and hybrid systems.

Dr. Cowlagi is a member of AIAA. He was the recipient of the Student Best Paper Award at the 2009 American Control Conference and the 2005 Aeronautical Society of India Award.

Panagiotis Tsiotras (S'90–M'92–SM'02) received the Ph.D. degree in aeronautics and astronautics from Purdue University, West Lafayette, IN, in 1993.

Currently, he is a Professor in the School of Aerospace Engineering, Georgia Institute of Technology, Atlanta. From 1994 to 1998, he was an Assistant Professor in the Department of Mechanical and Aerospace Engineering, University of Virginia, Charlottesville. His current research interests are in theoretical optimal and nonlinear control and vehicle autonomy, with applications to aerospace and mechanical systems.

Dr. Tsiotras is a Fellow of AIAA. He was a recipient of the NSF CAREER Award.