J Intell Robot Syst
DOI 10.1007/s10846-011-9631-z

Multiresolution Hierarchical Path-Planning for Small UAVs Using Wavelet Decompositions

Panagiotis Tsiotras · Dongwon Jung · Efstathios Bakolas

Received: 5 February 2011 / Accepted: 15 August 2011

Abstract We present an algorithm for solving the shortest (collision-free) path planning problem for an agent (e.g., a small UAV) with limited on-board computational resources. The agent has detailed knowledge of the environment and the obstacles only in the vicinity of its current position. Far away obstacles are only partially known and may even change dynamically. The algorithm makes use of the wavelet transform to construct an approximation of the environment at different levels of resolution. We associate with this multiresolution representation of the environment a graph, whose dimension can be made commensurate to the on-board computational resources of the agent. The adjacency list of the graph can be efficiently constructed directly from the approximation and detail wavelet coefficients, thus further speeding up the whole process. Simulations are presented to test the efficiency of the algorithm using non-trivial scenarios.

Keywords Path-planning · Wavelets · Multiresolution · UAV

P. Tsiotras (✉) · D. Jung · E. Bakolas
School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0150, USA
e-mail: tsiotras@gatech.edu

E. Bakolas
e-mail: ebakolas@gatech.edu

Present Address:
D. Jung
Korea Aerospace Research Institute, Yuseong-gu, Daejeon, Korea
e-mail: djung@kari.re.kr

1 Introduction

Autonomous operation of unmanned aerial vehicles requires both the trajectory design (planning) and trajectory tracking (control) tasks to be completely automated. On-board, real-time path-planning is particulary challenging for small-size aerial vehicles. The short response time scales of these vehicles pose increased difficulties for existing real-time route optimizers. Owing to their size, power restrictions, and lifting capabilities, small UAVs may not have the on-board computational resources (CPU and memory) to implement some of the sophisticated path-planning algorithms proposed in the literature. In most applications thus far this problem is being bypassed by providing navigation way-points, which have been computed off-line and/or are provided by a (human) supervisor. Furthermore, in a typical mission a UAV may be presented with a large amount of information collected using a plethora of diverse sensors (e.g., cameras, radars, laser scanners, satellite imagery), each with different range and resolution. A computationally efficient path-planning algo-

Published online: 16 September 2011

rithm should be able to filter/merge the information provided by these sensors, focusing the available computational resources on the part of the path (spatially and temporally) that is most relevant. Consequently, a computationally efficient algorithm suitable for on-line implementation for small UAVs should be able to combine short-term tactics (reaction to unforeseen threats) with long-term strategy (planning towards the ultimate goal). Recent examples that implement such a hierarchical path-planning strategy in an effort to combine the global/local (alternatively, strategic/tactical) layers under uncertain, dynamically changing scenarios are [3, 24, 31].

In this paper we propose a computationally efficient, hierarchical path-planning algorithm for autonomous agents (e.g., UAVs) navigating in a partially known environment $\mathscr{W}$ using an adaptive, discrete, cell-based approximation of $\mathscr{W}$. The innovation of our approach hinges on the use of wavelets to encode the district levels of fidelity (resolution) of $\mathscr{W}$ at different distances from the agent's current position. The motivation for this approach is simple: first, the agent's immediate reaction to an obstacle or a threat is needed only at the vicinity of its current position. Faraway obstacles or threats do not (or should not) have a large effect on the vehicle's *immediate* motion. Therefore, it is not prudent, from a computational point of view, to find a solution with great accuracy over large distances or over a long time horizon. The most accurate and reliable information of the environment is required (or it is even available) only in the vicinity of the vehicle. In that sense, the proposed algorithm can be classified under the category of "agent-centered" search algorithms [19]. Pictorially, such a multiresolution decomposition of the environment (say, an elevation map in this case) can be visualized as shown in Fig. 1.

The majority of prior multi-resolution and/or hierarchical algorithms proposed in the literature [2, 4, 13, 17, 18, 27–29, 36] use some form of quadtree decomposition of the environment. One drawback of quadtree-based decompositions is that a finer resolution is used close to the boundaries of all obstacles, regardless of their distance from the agent [1, 13]. This tends to waste computational resources. Ramifications to deal with this issue are possible, but, still, computing the adjacency relationships between cells of varying size is not easy. In a departure from these quadtree-based methods, in this paper we make use of the wavelet transform to perform the required multiresolution decompositions of the environment. This has several advantages. First, the wavelet transform provides a very fast decomposition[1] of a function at different levels of resolution. Since the range and resolution levels can be chosen by the user, the proposed algorithm results in search graphs of known prior complexity. The algorithm is therefore scalable, and can be tailored to the available computational resources of the agent. Second, the use of wavelet transform has the added benefit of allowing the construction of the associated cell connectivity relationships directly from the wavelet coefficients, thus eliminating the need for quadtrees. Finally, the use of wavelet transform provides flexibility in terms of the choice of the wavelet basis functions, which can help in reducing the complexity of the resulting abstraction (i.e., topological search graph) of the environment.

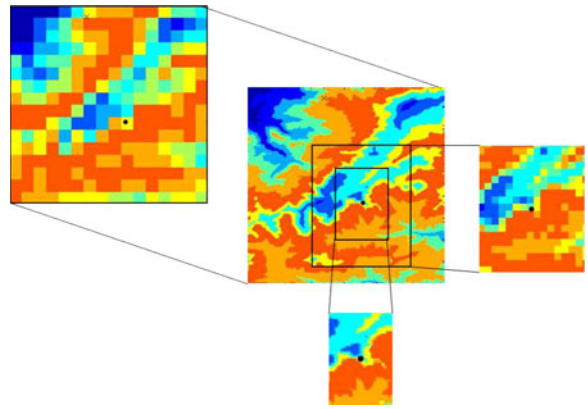


Fig. 1 Multiresolution representation of the environment according to the distance from the current location of the agent

[1]The computational complexity of the wavelet transform is of order $O(n)$ where n is the input data size [11]. This is even better than the Fast Fourier Transform which has complexity of order $O(n \log_2 n)$.

One of the earlier references where multiresolution path planning (for terrestrial mobile vehicles) was considered is Ref. [17]. In this reference the authors present a hierarchical path-planning scheme based on a multistage quadtree decomposition. The path to the target is first computed using a coarse grid and subsequently refined using information from higher resolution levels *uniformly along the path*. Even though this technique is efficient and easy to implement, it fails to take full advantage of the local information around the agent. Among the few references that have previously used wavelets in the context of multiresolution path-planning (again, for terrestrial mobile vehicles) is Ref. [28]. The approach in [28] combines an efficient model for the local behavior of the approximation, with improved computational characteristics, compared to the one proposed in [17]. In [28] however the wavelet coefficients are used to merely obtain an estimate of the terrain roughness (equivalently, terrain "transversibility") which is then included in the cost function. Wavelets are not used to construct the search graph as it is done in the current paper. Specifically for MAV/UAV applications, the authors of Ref. [29] used an adaptive multiresolution-based learning algorithm to estimate the obstacle locations. The algorithm generates an approximation of the environment based on estimates of the locations of points on the surface of an obstacle, typically obtained using structure from motion (SfM) ideas. A receding horizon controller is then used to control the UAV. The focus on that article is efficient, real-time estimation of obstacles as they appear in the field of view of the vehicle. Path-planning for small UAVs/MAVs was also investigated in [35] where a vision-based local multi-resolution mapping and path planning technique for MAVs using a forward-looking onboard camera was proposed. The vision data are used to construct a multiresolution map in polar coordinates, similarly to what was done in [25]. Dijkstra's algorithm is subsequently employed to find a collision-free path in the body frame. The main emphasis of [35] is the degradation of the measurements with the distance from the vehicle and not on providing a scalable, general purpose multi-scale algorithm that can be tailored to the observed data as done in the current paper.

The two references most closely related to our work are probably [31] and the article by Behnke [2]. In [31] the authors describe a multiresolution approach for vision-based path-planning for autonomous UAVs. The overall problem is divided into a global offline part based on a coarse model of the environment and a probabilistic local online computation, based both on the original model and on the information provided by the vision system. The main objective of that paper is however to update the original map with the data obtained on-line via a vision system and not on obtaining a numerically efficient path-planning algorithm that works directly with the wavelet coefficients. In [2] the author uses the idea of coarse/fine grid at close/faraway distances from the current location of the agent in order to avoid the demerits of uniform grids or standard quadtrees. Nonetheless, no connection with wavelets is attempted. In addition, the multiresolution scheme in [2] requires a rather careful handling of the cell connectivity at the boundaries between two different resolution levels. This is handled automatically by our approach.

The rest of the paper is organized as follows. After presenting the basics of wavelet decompositions of function spaces, we provide the main ingredient of our approach, which uses the environment wavelet coefficients to construct the graph representing the abstract topology of the problem. We then briefly investigate the complexity of resulting graph in terms of the problem data. These results provide guidelines how to adjust the transform parameters to construct graphs of prior known complexity. A standard graph search algorithm is then applied in an iterative fashion on these generated graphs to come up with a sequence of cells to the final destination. If the data about the environment are collected by onboard sensors, it is more efficient to use sector-like cell decompositions of the environment, instead of rectangular cell decompositions as with the standard wavelet transform. This can be easily achieved by applying the wavelet transform in polar coordinates. This scheme is briefly elaborated upon in the second part of the paper. The paper

concludes with numerical simulation results and with some suggestions for possible extensions and improvements of the proposed algorithm.

2 A Multiresolution Decomposition of $\mathscr{W}$

2.1 Topology of the Problem

We assume that the UAV is represented by a point mass, and is allowed to navigate in the horizontal plane, which we will denote by $\mathscr{W} \subset \mathbb{R}^2$. It is assumed that the UAV cannot change altitude to avoid obstacles, but rather it has to navigate around them. Given the environment $\mathscr{W}$ and the obstacle-free configuration space $\mathscr{F} \subseteq \mathscr{W}$ that contains all the feasible (i.e., obstacle-free) configurations of the vehicle, our objective is to find a path $\mathscr{P}$ from an initial configuration $\mathsf{x}_0 \in \mathscr{F}$ to a final configuration $\mathsf{x}_f \in \mathscr{F}$. We assume that the obstacle space $\mathscr{O} \triangleq \mathscr{W} \backslash \mathscr{F}$ is the union of a finite number of obstacles. We only assume that the obstacles are represented by compact subsets of $\mathscr{W}$. No other assumptions will be imposed on $\mathscr{O}$. In particular, the obstacles can be non-convex, non-polygonal subsets of $\mathscr{W}$.

We will use a cell decomposition that will allow us to efficiently partition $\mathscr{W}$ (and hence $\mathscr{F}$) into classes of cells of different size. An m-cell decomposition $\mathscr{C}_d = \{c_i \in \mathscr{W} : i = 1, \ldots, m = |\mathscr{C}_d|\}$ of $\mathscr{W}$ is a finite collection of subsets (cells) of $\mathscr{W}$ with non-overlapping interiors, whose union is exactly $\mathscr{W}$. As usual with hierarchical decomposition schemes, cells in $\mathscr{C}_d$ will be classified as either empty cells (when $c_i \cap \mathscr{O} = \varnothing$), mixed cells (when $c_i \cap \mathscr{O} \neq \varnothing$ and $c_i \cap \mathscr{F} \neq \varnothing$), and full cells (when $c_i \subseteq \mathscr{O}$). Two cells $c_i, c_j \in \mathscr{C}_d$ are adjacent[2] if the intersection of their boundaries is neither empty nor a singleton.

Given two cell decompositions $\mathscr{C}_d$ and $\mathscr{C}'_d$ of $\mathscr{W}$ of respective cardinalities $|\mathscr{C}_d|$ and $|\mathscr{C}'_d|$, we say that $\mathscr{C}'_d$ is a *finer, or higher resolution* decomposition of $\mathscr{W}$ than $\mathscr{C}_d$ if $|\mathscr{C}'_d| \geq |\mathscr{C}_d|$ and for every cell $c_i \in \mathscr{C}_d$ there exists an integer $p_i \geq 1$ such that $c_i = \bigcup_{j=1}^{p_i} c'_j$ where $c'_j \in \mathscr{C}'_d$, where at least one $p_i > 1$.

Later on, we will associate to a cell decomposition $\mathscr{C}_d$ a graph $\mathscr{G} = (\mathscr{V}, \mathscr{E})$ with node set $\mathscr{V} = \mathscr{V}(\mathscr{G})$ and edge set $\mathscr{E} = \mathscr{E}(\mathscr{G})$, such that: (i) the node set $\mathscr{V}$ corresponds to the free and mixed cells of $\mathscr{C}_d$, and (ii) the edge set $\mathscr{E}$ describes the relationship between the cells in $\mathscr{C}_d$ that are adjacent to each other. It is easy to see that $\mathscr{G}$ is a topological graph [21]. Clearly, there is an one-to-one correspondence between the elements of $\mathscr{V}(\mathscr{G})$ and the free and mixed cells of $\mathscr{C}_d$. We write $v \sim c_i$ to denote this correspondence. In particular, we associate each node $v \in \mathscr{V}(\mathscr{G})$ to the center of the cell c_i. We use $\mathsf{cell}_{\mathscr{G}}(v)$ to denote the center of the corresponding cell in this case. Finally, if $\mathsf{x} \in c_i$, we will write $v = \mathsf{node}_{\mathscr{G}}(\mathsf{x})$, where $v \sim c_i$ to denote the node of $\mathscr{G}$ that represents the cell of $\mathscr{C}_d$ that contains x.

2.2 The 2D Wavelet Transform

We use wavelets as the main tool to construct hierarchical, multiresolution cell decompositions of $\mathscr{W}$. The idea behind the theory of the wavelet transform is to represent a function $f \in \mathscr{L}^2(\mathbb{R})$ as a summation of elementary basis functions $\phi_{J,k}$ and $\psi_{j,k}$ as follows

$$f(x) = \sum_{k \in \mathbb{Z}} a_{J,k} \phi_{J,k}(x) + \sum_{j \geq J} \sum_{k \in \mathbb{Z}} d_{j,k} \psi_{j,k}(x), \qquad (1)$$

where $\phi_{j,k}(x) = 2^{j/2} \phi(2^j x - k)$ and $\psi_{j,k} = 2^{j/2} \psi(2^j x - k)$ and $\phi(x)$ and $\psi(x)$ are the scaling function and mother wavelet functions, respectively. In the ideal case, both $\phi(x)$ and $\psi(x)$ have compact support or decay very fast outside a small interval, so they can capture localized features of f.

The first summation in Eq. 1 gives a low resolution or coarse approximation of the function f. The second term in Eq. 1 gives the difference (the remaining details) between the original function and its low resolution approximation. The index j denotes the resolution level. For each increasing index j, a higher (or finer) resolution term is added, which adds more details. The expansion Eq. 1 thus reveals the properties of f at different levels of resolution [5].

[2]This definition induces an 4-cell neighborhood for each cell, but 8-neighborhoods can be handled easily as well.

This idea can be readily extended to the two-dimensional case by introducing the following families of functions

$$\Phi_{j,k,\ell}(x, y) = \phi_{j,k}(x)\phi_{j,\ell}(y),$$
$$\Psi^1_{j,k,\ell}(x, y) = \phi_{j,k}(x)\psi_{j,\ell}(y),$$
$$\Psi^2_{j,k,\ell}(x, y) = \psi_{j,k}(x)\phi_{j,\ell}(y),$$
$$\Psi^3_{j,k,\ell}(x, y) = \psi_{j,k}(x)\psi_{j,\ell}(y), \tag{2}$$

which are derived from taking tensor product of the scaling and the mother wavelet function in Eq. 1. Given the function $f \in \mathscr{L}^2(\mathbb{R}^2)$ we can then write

$$f(x, y) = \sum_{k,\ell\in\mathbb{Z}} a_{J,k,\ell}\Phi_{J,k,\ell}(x, y) + \sum_{i=1}^{3}\sum_{j\geq J}\sum_{k,\ell\in\mathbb{Z}} d^i_{j,k,\ell}\Psi^i_{j,k,\ell}(x, y), \tag{3}$$

where, for the case of orthonormal[3] wavelets, the approximation coefficients are given by the projections of f on the scaling functions, as follows

$$a_{j,k,\ell} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x, y)\,\Phi_{j,k,\ell}(x, y)\,\mathrm{d}x\,\mathrm{d}y, \tag{4}$$

and the detail coefficients are given by projections of f on the wavelet functions as follows

$$d^i_{j,k,\ell} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x, y)\,\Psi^i_{j,k,\ell}(x, y)\,\mathrm{d}x\,\mathrm{d}y, \quad i = 1, 2, 3. \tag{5}$$

The key property of wavelets used in this paper is the fact that the expansion (Eq. 3) induces the following decomposition of $\mathscr{L}^2(\mathbb{R}^2)$

$$\mathscr{L}^2(\mathbb{R}^2) = \mathscr{V}_J \oplus \mathscr{W}_J^{\text{detail}} \oplus \mathscr{W}_{J+1}^{\text{detail}} \oplus \cdots \tag{6}$$

where $\mathscr{V}_J = \overline{\text{span}}\{\Phi_{J,k,\ell}\}_{k,\ell\in\mathbb{Z}}$ and similarly $\mathscr{W}_j^{\text{detail}} = \overline{\text{span}}\{\Psi^1_{j,k,\ell}, \Psi^2_{j,k,\ell}, \Psi^3_{j,k,\ell}\}_{k,\ell\in\mathbb{Z}}$ for $j \geq J$.

In this paper we will use the Haar family of wavelets for reasons that will become apparent shortly. Each scaling function $\phi_{j,k}(x)$ and wavelet function $\psi_{j,k}(x)$ in the Haar system is supported on the dyadic interval $I_{j,k} \triangleq [k/2^j, (k+1)/2^j]$ of length $1/2^j$ and does not vanish in this interval [5]. Subsequently, one may associate the functions $\Phi_{j,k,\ell}$ and $\Psi^i_{j,k,\ell}$ $(i = 1, 2, 3)$ with the rectangular cell $c^j_{k,\ell} \triangleq I_{j,k} \times I_{j,\ell}$. This correspondence between scaling and wavelet functions and their corresponding interval of influence is the key idea behind our approach. The details of establishing this correspondence in a numerically efficient manner is given in a later section.

[3] In the more general case of biorthogonal wavelets projections on the space spanned by the dual wavelets and dual scaling functions should be used instead.

2.3 Wavelet Decomposition of the Risk Measure

Without loss of generality, in the sequel we assume that the environment is given by the unit square, that is, $\mathscr{W} = [0, 1] \times [0, 1]$, and is represented numerically using a discrete (fine) grid of $2^N \times 2^N$ dyadic points. The finest level of resolution $J_{\max}$ is therefore bounded by N. The decomposition of a given function f defined over $\mathscr{W}$ at resolution level $J \geq J_{\min}$, given by,

$$f(x, y) = \sum_{k,\ell=0}^{2^{J_{\min}}-1} a_{J_{\min},k,\ell}\,\Phi_{J_{\min},k,\ell}(x, y) + \sum_{i=1}^{3}\sum_{j=J_{\min}}^{J-1}\sum_{k,\ell=0}^{2^j-1} d^i_{j,k,\ell}\Psi^i_{j,k,\ell}(x, y), \tag{7}$$

induces a cell decomposition of $\mathscr{W}$ of square cells of dimension no smaller than $1/2^J \times 1/2^J$.

Let us now assume that the function of interest is a certain *risk function* (or *risk measure*) $\mathsf{rm} : \mathscr{W} \mapsto [0, 1]$, defined by

$$\mathsf{rm}(\mathsf{x}) = \begin{cases} (d_{\max} - \min_{\mathsf{y}\in\mathscr{O}} \|\mathsf{x}-\mathsf{y}\|_\infty)/d_{\max}, & \text{if } \mathsf{x} \in \mathscr{F}, \\ 1, & \text{if } \mathsf{x} \in \mathscr{O}, \end{cases} \tag{8}$$

where $d_{\max} \triangleq \max_{\mathsf{x}\in\mathscr{F}} \min_{\mathsf{y}\in\mathscr{O}} \|\mathsf{x}-\mathsf{y}\|_\infty$, and $\mathsf{x} = (x, y)$. The function rm defined in Eq. 8 provides a measure of the distance of the closest obstacle from the current location. Alternatively, one may choose rm either as the probability that $(x, y) \in \mathscr{O}$ or as the 2D elevation map (i.e., a digital elevation model or DEM) of terrain for a UAV flying at constant altitude. The actual choice of the function rm is not central to the developments in this

paper, but the choice in Eq. 8 is quite general and captures most cases of interest.

Let the neighborhood, of radius r, around the current location of the agent, say $\mathsf{x}_0 = (x_0, y_0)$, be defined by

$$\mathcal{N}(\mathsf{x}_0, r) \triangleq \{\mathsf{x} \in \mathcal{W} : \|\mathsf{x}_0 - \mathsf{x}\|_\infty \leq r\}. \tag{9}$$

Given the risk measure function over $\mathcal{W}$, we construct approximations of $\mathcal{W}$ at distinct levels of resolution $J_{\min} \leq j \leq J_{\max}$, where $J_{\min}, J_{\max} \in \{1, \ldots, N\}$, with corresponding ranges for each resolution level $r_j > 0$ from the agent's current location x_0 where $r_j < r_{j-1}$. By this we mean that resolution level $J_{\max}$ is used for all points inside the set $\mathcal{N}(\mathsf{x}_0, r_{J_{\max}})$, resolution level j is used for all points inside the set $\mathcal{N}(\mathsf{x}_0, r_{j-1}) \backslash \mathcal{N}(\mathsf{x}_0, r_j)$ for $j = J_{\max}, \ldots, J_{\min} + 2$, and resolution level $J_{\min}$ is used inside the set $\mathcal{W} \backslash \mathcal{N}(\mathsf{x}_0, r_{J_{\min}+1})$. Let now the index set $\mathcal{I}(j) \triangleq \{0, 1, \ldots, 2^j - 1\}$ and let

$$\mathcal{K}(j) \triangleq \{k \in \mathcal{I}(j) : I_{j,k} \cap [x_0 - r_j, x_0 + r_j] \neq \varnothing\}, \tag{10a}$$

$$\mathcal{L}(j) \triangleq \{\ell \in \mathcal{I}(j) : I_{j,\ell} \cap [y_0 - r_j, y_0 + r_j] \neq \varnothing\}. \tag{10b}$$

Then the wavelet decomposition of rm,

$$\begin{aligned}\mathsf{rm}(x, y) = &\sum_{k,\ell \in \mathcal{I}(J_{\min})} a_{J_{\min},k,\ell}\, \Phi_{J_{\min},k,\ell}(x, y) \\ &+ \sum_{i=1}^{3} \sum_{j=J_{\min}}^{J_{\max}-1} \sum_{\substack{k \in \mathcal{K}(j) \\ \ell \in \mathcal{L}(j)}} d^i_{j,k,\ell} \Psi^i_{j,k,\ell}(x, y),\end{aligned} \tag{11}$$

induces the following cell decomposition on $\mathcal{W}$

$$\mathcal{C}_d = \Delta C_d^{J_{\min}} \cup \cdots \cup \Delta C_d^{J_{\max}}, \tag{12}$$

where ΔC_d^j is a union of cells $c^j_{k,\ell}$ of dimension $1/2^j \times 1/2^j$.

The values of $J_{\max}$, $J_{\min}$ and r_j can be chosen according to the desired local quality of the data. Typically, the choice of $J_{\max}$ will be dictated by the requirement that at this level of resolution all cells should be resolvable into either free or occupied cells. The choice of $J_{\min}$, as well as the values of r_j, are typically dictated by the on-board computational resources. See discussion in Section 3.2 below on how the choices of the values of $J_{\min}$, $J_{\max}$ and r_j affect the computational complexity of the algorithm.

Remark 1 In practice, the fast lifting wavelet transform implementation (FLWT) [33] may be used to compute Eq. 11. The FLWT has a number of algorithmic advantages, such as faster computation speed (twice as fast as the usual discrete wavelet transform), *in-place* calculation of the coefficients (thus reducing the computer memory requirements), immediate inverse transform, and generality for extension to problems with irregular boundaries. In conjunction with the possibility to perform all operations of the FLWT in integer arithmetic [6], these properties make the FLWT especially suitable for processing using small micro-controllers. For the sake of brevity–and since this topic is not central to the developments in the rest of the paper–we will not give the details of the lifting implementation of the Haar wavelet transform. Instead, we refer the interested reader to [14, 15] for additional details on the FLWT implementation of the proposed algorithm on actual autopilot hardware.

Remark 2 The choice of an appropriate fine resolution level for the path planner may also be assisted by the consideration of the kinematic constraints of vehicle. Given a map in an appropriate scale, one should choose, if possible, the cell size at the finest level of the cell decomposition so that it is larger than the minimum turning radius of the vehicle. This will ensure that a feasible trajectory will always exist that is consistent with the discrete geometric path. However, this requirement may conflict with the minimum cell size required to unambiguously resolve obstacles in the vicinity of the vehicle.

3 From the Wavelet Decomposition to the Connectivity Graph

3.1 Adjacency List from the FLWT

In this section, we describe an algorithm to assign a topological graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ to the cell decomposition $\mathcal{C}_d$. The set of nodes $\mathcal{V}$ of $\mathcal{G}$ represents the cells $c^j_{k,\ell}$ in $\mathcal{C}_d$ and the set of edges $\mathcal{E}$

represents the connectivity relationship between those cells. Specifically, we show that the connectivity of the graph $\mathscr{G}$ (encoded via the adjacency list for each cell) can be constructed from the wavelet decomposition coefficients. For an alternative approach to construct directly the adjacency matrix of $\mathscr{G}$ from the wavelet coefficients, the interested reader may refer to [8].

The key observation of the approach is the fact that since the scaling function $\Phi_{j,k,\ell}$ and the wavelet functions $\Psi^i_{j,k,\ell}$ $(i = 1, 2, 3)$ are uniquely associated with the square cell $c^j_{k,\ell} = I_{j,k} \times I_{j,\ell}$, the corresponding nonzero approximation and detail coefficients encode the necessary information regarding the cell geometry (size and location) [16, 34]. To this end, consider a cell $c^{j_0}_{k,\ell}$ at level j_0, whose dimension is $1/2^{j_0} \times 1/2^{j_0}$ with its center located at (k, ℓ). A cell will be called *independent* if it is associated with one non-zero approximation coefficient $a_{j_0,k,\ell}$, while the corresponding detail coefficients $d^i_{j,k,\ell}$ $(i = 1, 2, 3)$ at levels $j_0 \le j \le J_{\max}$ are all zero. Otherwise, the cell is marked as a *parent* cell and is subdivided into four *leaf* cells at level $j_0 + 1$. In Fig. 2 the top-most parent cell $c^{j_0}_{k,\ell}$ is subdivided into three independent cells at level $j_0 + 1$, whereas at quadrant IV the cell is further subdivided into four independent leaf cells at level $j_0 + 2$.

Our goal is to locate all independent cells and extract their connectivity relationships. In order to simplify the discussion, below we consider only two distinct resolution levels. Let us assume that the agent is located at $\mathsf{x} = (x, y)$ and the high resolution horizon is r. Starting from a coarse cell $c^{j_0}_{k,\ell}$ we can determine if the cell either partially intersects or totally belongs to the set $\mathscr{N}(\mathsf{x}, r)$ (and hence decide whether the cell needs to be subdivided into higher resolution cells) by checking whether $(k, \ell) \in (\mathscr{K}(j_0), \mathscr{L}(j_0))$. If the cell needs to be subdivided into higher resolution cells, the Inverse Fast Lifting Wavelet Transform (IFLWT) is first performed on the current cell (local reconstruction) in order to recover the four approximation coefficients at level $j_0 + 1$ and the corresponding detail coefficients.

We adopt the raster scan method (namely, a zigzag search: I→II→III→IV) to examine each cell inside the parent cell overlapping with $\mathscr{N}(\mathsf{x}, r)$. This procedure is recursively repeated until the maximum resolution level is reached. Figure 3 illustrates this recursive raster scan search. Once a cell is recognized as independent, we assign a node in the graph $\mathscr{G}$ with the node cost being the approximation coefficient representing the average risk measure over the cell. In addition, the detail coefficients associated with the current cell are all set to zero.

The next step is to determine the connectivity/adjacency relationships between all identified independent cells. To this end, after a cell has been identified as an independent cell, one searches its neighboring cells. Following the recursive raster search used for cell identification, the adjacency search is initiated at the current cell, confining the search to the following four directions: Left, top-left, top, and top-right from the current cell (for 8-connectivity). By doing this, one establishes half of the links from the current cell, with the remaining links to the current cell to be determined later as the recursive raster scan progresses to the next cells.

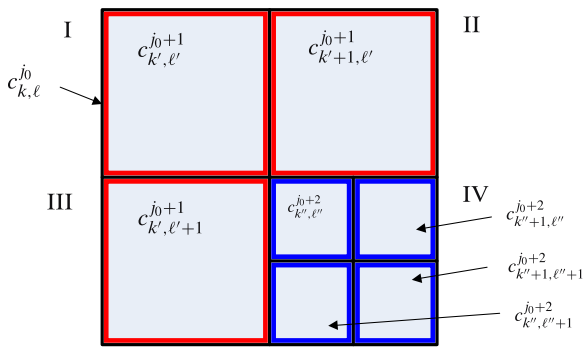


Fig. 2 Multi-resolution cell subdivision across different levels

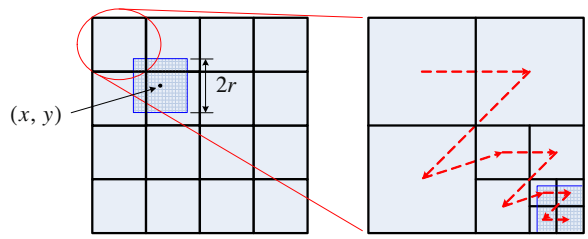


Fig. 3 Recursive raster scan for identifying independent cells

Along each search direction, it is necessary to search for independent cells adjacent to the current cell. Because the cells may be of different size, the adjacency search requires the determination of the search region over which the detail coefficients are examined to find independent cells. Two search regions are used for each leaf cell, depending on its location: External search, which looks for connections from a leaf cell beyond the parent cell and internal search which looks for connections between leaf cells inside the parent cell. The external and internal search regions are represented by solid and dashed arrows, respectively in Fig. 4. For the internal search region, the search is confined over the adjacent leaf cell while the external search region is determined based on parent-leaf cell inheritance properties. That is, in order to determine the external search region of a leaf cell, the parent cell of that leaf cell is back-traced until the search region of the parent cell becomes an internal search region as one of the solid arrows in Fig. 4. As an illustrative example, Fig. 5 shows this inheritance property. In Fig. 5 the current cell is chosen to be $c_{\mathrm{I}}^{j_0+2}$ (blue cell). This cell is a leaf cell of the parent cell $c_{\mathrm{IV}}^{j_0+1}$, which, in turn, is a leaf cell of the top-most parent cell $c_{k,\ell}^{j_0}$. The cell $c_{\mathrm{IV}}^{j_0+1}$ is located in the fourth quadrant inside the top-most parent cell $c_{k,\ell}^{j_0}$, hence the search region for $c_{\mathrm{IV}}^{j_0+1}$ ends up with the internal searches at level j_0+1 shown

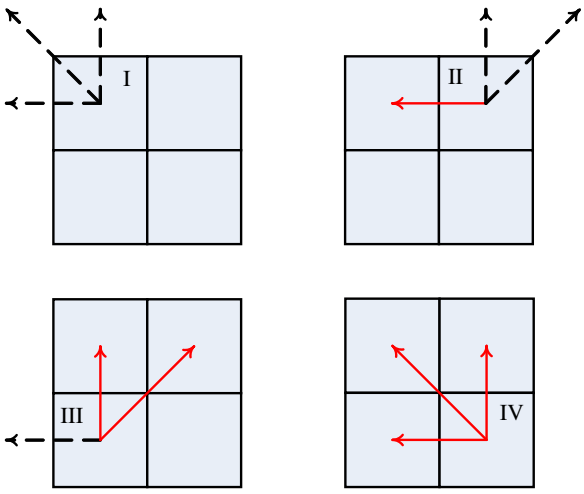


Fig. 4 Basic connectivity properties with respect to the location of the leaf cell

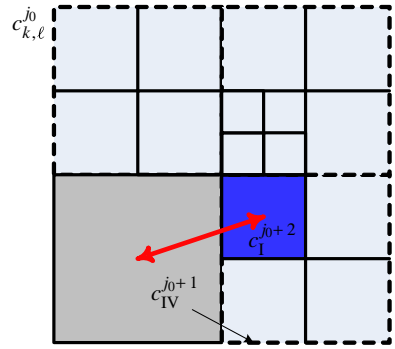
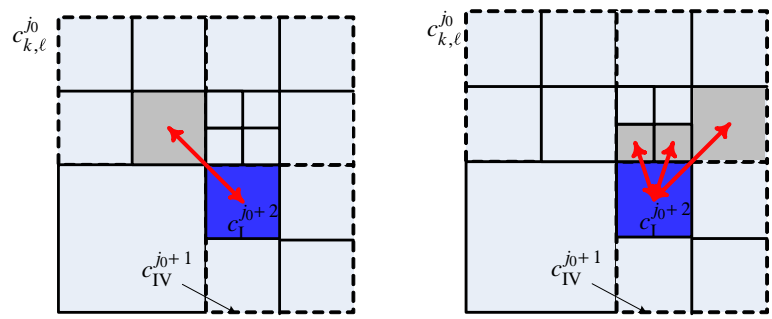


Fig. 5 Searching an adjacent cell along the left search direction

in the bottom right diagram in Fig. 4. This search region is then inherited by the cell $c_{\mathrm{I}}^{j_0+2}$ and thus the adjacency search of the blue cell looks for any independent cells over $c_{\mathrm{III}}^{j_0+1}$, which, in this case, is independent.

The previous adjacency search algorithm refines the search region at higher levels if one of the adjacent cells is found not to be an independent cell, that is, if it is comprised of finer resolution cells. Subsequently, the detail coefficients at the higher levels inside the search region are examined in order to find the next finer cell that is adjacent to the current cell. In Fig. 6a during the top-left search direction of $c_{\mathrm{I}}^{j_0+2}$, the process initially examines the cell $c_{\mathrm{I}}^{j_0+1}$ located at the top-left corner of the current cell through the corresponding detail coefficients. Provided that the detail coefficients associated with the cell $c_{\mathrm{I}}^{j_0+1}$ take a non-zero value, the search region is subsequently subdivided. The process repeats at level j_0+2 when the neighboring cell to the current cell becomes an independent cell. In Fig. 6a, since there exist no other independent cells along the top-left direction except the shaded one, a bidirectional link is established between the current and the opposite cells. Similarly, in Fig. 6b for the top search direction, after determining the search region and refining it repeatedly, the two cells at level j_0+3 and one at level j_0+2 are found to be independent and adjacent to the current cell.

The final result of this process is the graph $\mathscr{G}$ that describes the connectivity between all the cells in $\mathscr{C}_d$. In the next section we explain how the



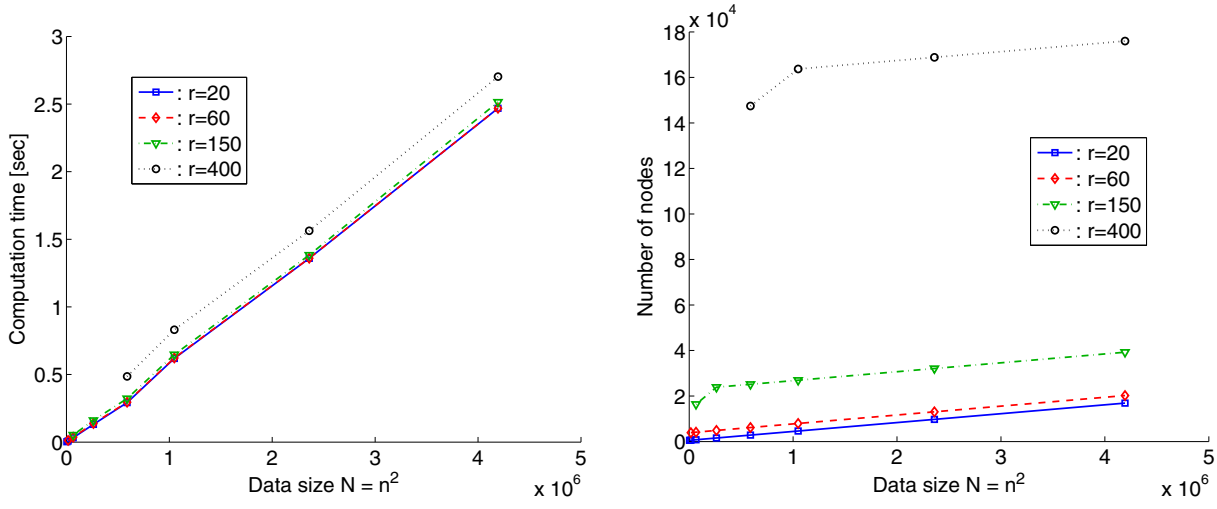
(a) Search top-left (b) Search top

Fig. 6 Refined adjacency search algorithm

planning is executed on this graph. Before doing that, next, we briefly investigate the complexity of the proposed algorithm and the dimensionality of the resulting graph in terms of the wavelet transform parameters (i.e., resolution levels and ranges).

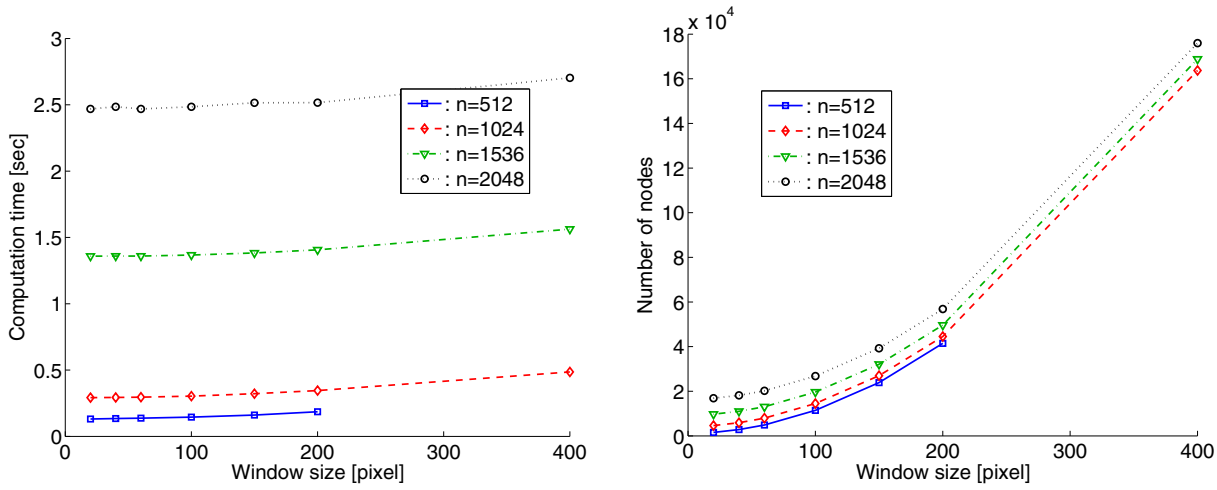
3.2 Complexity Analysis

In order to investigate the scalability of the proposed adjacency search algorithm and, furthermore, get an idea of the ability for real-time execution on a given processor, it is desirable to analyze the algorithm both in terms of computational time and memory requirements. Since the proposed algorithm uses recursive calls in both the steps of identification of cells and for establishing the links between the cells, obtaining an analytic expression for the computational complexity is a non-trivial task. In this section, we instead opt for estimating the computational complexity of the algorithm numerically. To this end, suppose that the input data to the algorithm are given by a vector of dimension $N = n^2$, containing the wavelet coefficients from the FLWT of the map of the environment (assumed to be of dimension $n \times n$). Again, in order to simplify the analysis, we incorporate only two distinct levels of resolution: a coarse level $J_{\min}$ away from the agent and a fine level or resolution $J_{\max}$ close to the agent. The size of the high resolution window is assumed to be r.



(a) Computational time (b) Number of nodes

Fig. 7 Computational cost for the adjacency search algorithm in terms of data size



(a) Computational time (b) Number of nodes

Fig. 8 Computational cost for the adjacency search algorithm in terms of window size

We compare the complexity of the algorithm both in terms of the computational time and the required memory (e.g., number of nodes of the resulting graph) with respect to the input data set and the resolution window size r. For a given input data of size N, Fig. 7a and b show the corresponding computational time and the number of nodes, respectively, and for different values of r. As it is evident from Fig. 7, the computational complexity grows linearly with N, that is, the complexity is of order $\mathcal{O}(N)$.

Figure 8 shows the computational complexity of the algorithm with respect to the window size r, which turns out to be a quadratic relationship, that is, of order $\mathcal{O}(r^2)$. This is not surprising, since the number of cells inside a square cell of size r is proportional to r^2.

In practice, one can use these complexity trends in order, for instance, to choose the size of the window r to limit the size of the resulting graph so that it is commensurate to the available memory and processing hardware.

4 Multiresolution Path Planning

4.1 Cost Assignment

To each directed edge $(u, v) \in \mathcal{E}$ of the graph $\mathcal{G}$ describing the connectivity of the multiresolution cell decomposition in Eq. 12, we then assign an edge cost, as follows

$$\mathcal{J}(u, v) = \mathsf{rm}(\mathsf{cell}_{\mathcal{G}}(v)) + \alpha \|\mathsf{cell}_{\mathcal{G}}(u) - \mathsf{cell}_{\mathcal{G}}(v)\|_2, \tag{13}$$

where $\alpha \geq 0$ is a weight constant. The first term in Eq. 13 is proportional to the probability that the target node is close to an obstacle, while the second term penalizes the (Euclidean) distance between $\mathsf{cell}_{\mathcal{G}}(u)$ and $\mathsf{cell}_{\mathcal{G}}(v)$. The larger the α, the more emphasis we place on a shorter path.

Given the graph $\mathcal{G}$ along with the associated edge and node costs, we then invoke the A* (or Dijkstra's) algorithm [7, 22] to find a path minimizing the accumulated cost from the initial to the destination node in the graph, or determine that such a path does not exist.

4.2 Replanning

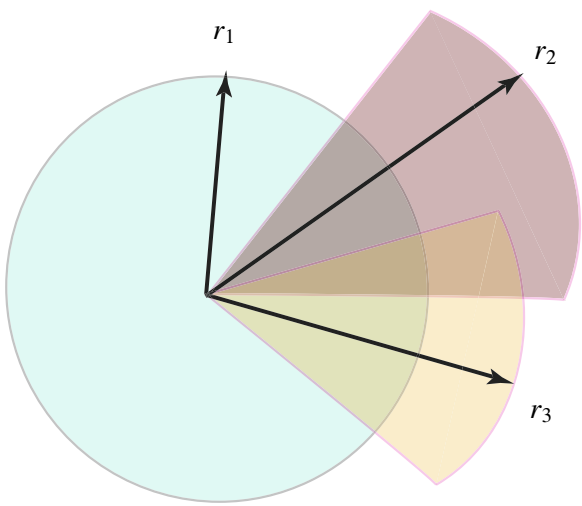
The proposed multiresolution path-planning algorithm proceeds as follows. Assume the agent is at location $\mathsf{x}(t_i)$ at time t_i. We construct a multiresolution decomposition $\mathcal{C}_d(t_i)$ of $\mathcal{W}$ around $\mathsf{x}(t_i)$ with corresponding graph $\mathcal{G}(t_i)$. The A* algorithm yields a path $\mathcal{P}(t_i) = (v_1^i, v_2^i, \cdots, v_{\ell_i-1}^i, v_{\ell_i}^i = v_f^i)$ in $\mathcal{G}(t_i)$ of mixed and free nodes of length ℓ_i, where $v_1^i = \mathsf{node}_{\mathcal{G}(t_i)}(\mathsf{x}(t_i))$ and $v_f^i = \mathsf{node}_{\mathcal{G}(t_i)}(\mathsf{x}_f)$ if such

a path exists. It is assumed that v_2^i is free owing to the high resolution decomposition of $\mathscr{W}$ close to $\mathsf{x}(t_i)$. The agent subsequently moves from v_1^i to v_2^i and the process is continued until some time t_f when $\|\mathsf{x}(t_f) - \mathsf{x}_f\| < 1/2^{J_{\max}}$, at which time the algorithm terminates. At the last step the agent moves from $\mathsf{x}(t_f)$ to x_f.

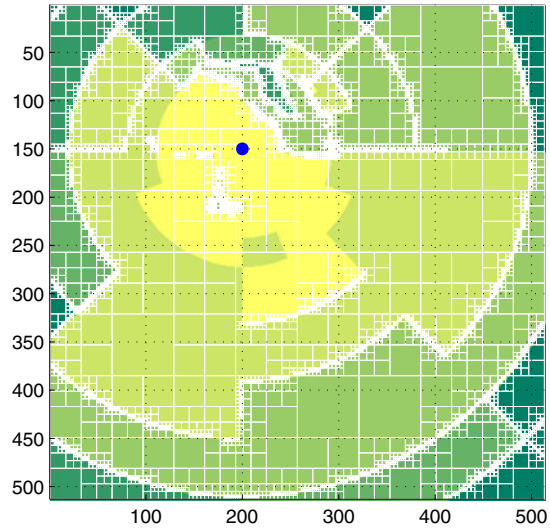
Note that the actual path $\mathsf{x}(t_0), \mathsf{x}(t_1), \ldots, \mathsf{x}(t_f)$ followed by the agent is given by the sequence of nodes $\mathsf{node}_{\mathscr{G}(t_0)}(\mathsf{x}(t_0))$, $\mathsf{node}_{\mathscr{G}(t_1)}(\mathsf{x}(t_1))$, $\ldots$, $\mathsf{node}_{\mathscr{G}(t_f)}(\mathsf{x}(t_f))$. Since the connectivity graph $\mathscr{G}(t)$ changes at each time step, it is therefore possible that the same state may be visited twice since it may correspond to nodes of two distinct graphs. That is, it is possible that $\mathsf{cell}_{\mathscr{G}(t_i)}(v_k^i) = \mathsf{cell}_{\mathscr{G}(t_j)}(v_m^j)$ for $i \neq j$. This will cause the agent to repeat the previous (optimal) decision ending up in an endless loop. In order to avoid such pathological situations, one needs to maintain a list $\mathsf{L}_{\mathsf{Visited}}(i) = \{\mathsf{x}(t_0), \mathsf{x}(t_1), \ldots, \mathsf{x}(t_i)\}$ of all visited states up to the current time step t_i and allow backtracking, similar to what it is done in other "short-sighted" path-planning algorithms in uncertain environments, such as D^*. The specific implementation of this step is beyond the scope of this paper and can be found in [10], where the completeness of the algorithm is also established.

5 From Rectangular to Sector Cell Decompositions

The cell decomposition of $\mathscr{W}$ is given in terms of square cells, hence also the choice of the particular norm in Eq. 9. The use of this norm was motivated by the choice of the Haar family of wavelets in the decomposition (Eq. 7). In practice however, information about the environment is collected via the use of sensors (cameras, radars, antennas, etc) whose area of influence or field of view does not conform to the box-like topology of Eq. 9. In fact, most typical sensor devices provide sector-like representations of the environment (see Fig. 9a). This type of information is not in the most efficient form for path planning algorithms that employ square or polygonal cell approximations of the environment. Such decompositions are not compatible to the sector-like



(a) Typical sensor topology.



(b) Quadtree decomposition in sector topology.

Fig. 9 **a** Typical sensors have different ranges, fields of view and resolution. Ideally, the algorithm that processes the sensor data should conform to this topology. **b** A cell decomposition based on the available sector approximation of the environment obtained by the on-board sensor devices of the agent (denoted with the *blue dot*). In order to resolve the geometry of the arc-boundary of each sector the standard quadtree algorithm generates a large number of cells close to the boundaries of these arcs

representations obtained by the onboard sensor devices [1, 35]. For instance, standard quadtree decompositions that use rectangular cells will tend to generate many small-size cells to resolve the sector boundaries, as seen in Fig. 9b.

Although it is possible to use spherical wavelets [12, 23] to capture the sector topology of the onboard vehicle sensors, a simpler way to overcome this difficulty is to employ a conformal mapping to relate sector cells to rectangular cells in a new (polar) coordinate system. Recall that the polar transformation maps a sector $\mathscr{R}$ specified by the radii $\rho_{\min}$ and $\rho_{\max}$ and the angles $\theta_{\min}$ and $\theta_{\max}$ from the Cartesian (x, y) plane to the (ρ, θ) plane using the polar transformation

$$x = x_0 + \rho\cos\theta, \qquad y = y_0 + \rho\sin\theta, \tag{14}$$

with origin the current location of the agent $\mathsf{x}_0 = (x_0, y_0)$. We can thus obtain a *rectangular* domain $\mathscr{R}'$ in the (ρ, θ) plane, defined by the same radii and angles. See Fig. 10.

We can use this idea to perform path-planning exclusively in the (ρ, θ) domain. In this case, the Haar wavelet decomposition of the function $f'(\rho, \theta) = f \circ \phi_{\mathsf{x}_0}(\rho, \theta)$, where $\phi_{\mathsf{x}_0} : \mathbb{R} \times S^1 \mapsto \mathbb{R}^2$ is the function defined in Eq. 14 for each given x_0 at resolution level $J \geq J_{\min}$ is given by

$$f'(\rho,\theta) = \sum_{k,\ell=0}^{2^{J_{\min}}-1} a_{J_{\min},k,\ell}\,\Phi_{J_{\min},k,\ell}(\rho,\theta) + \sum_{i=1}^{3}\sum_{j=J_{\min}}^{J-1}\sum_{k,\ell=0}^{2^j-1} d^i_{j,k,\ell}\Psi^i_{j,k,\ell}(\rho,\theta), \tag{15}$$

and induces a square cell decomposition of $\mathscr{W}$ of square cells of dimension $1/2^J \times 1/2^J$ in the (ρ, θ) coordinate system, similarly to Eq. 7. The approach leads to cell decompositions that are reminiscent of log-polar [25, 35] and hyperbolic-polar [30] representation, but with an additional, naturally embedded, sector multiresolution structure. An illustration of this idea is demonstrated in the next section.

6 Numerical Simulation Results

6.1 Comparison with Full Information Path-Planning

In this section, we compare the performance of the proposed multiresolution algorithm with the one utilizing the full (fine resolution) information about the environment. The latter is assumed to be square of dimension 512×512 units. To search the path on the associated graph, we employed Dijkstra's algorithm which always ensures finding an optimal path, if one exists. A highly cluttered environment shown in Fig. 11 was chosen as the test scenario. The multiresolution algorithm was executed based on a coarse level $J_{\min} = 4$ and a fine level $J_{\max} = 9$. The size of the fine resolution window around the agent is $r = 20$. Both algorithms were written in C/C++ using a dynamic heap structure for the sake of computational efficiency, and were executed on a desktop PC equipped with a dual-core CPU at 1.6 GHz with 2 GB of RAM memory.

Table 1 summarizes the performance of the two algorithms. In the table, the percentage values represent the performance of the proposed multiresolution algorithm over the fine resolution

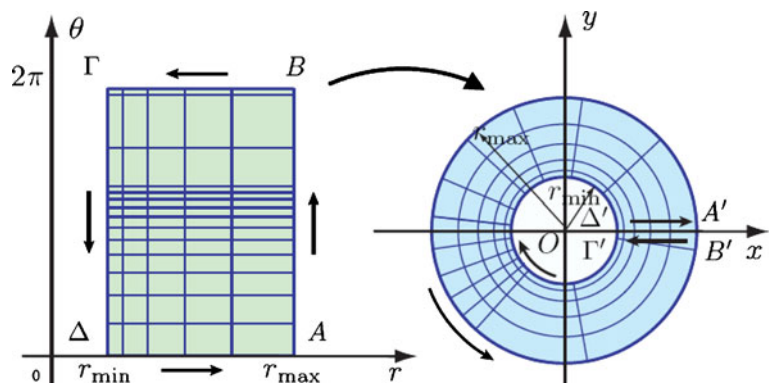
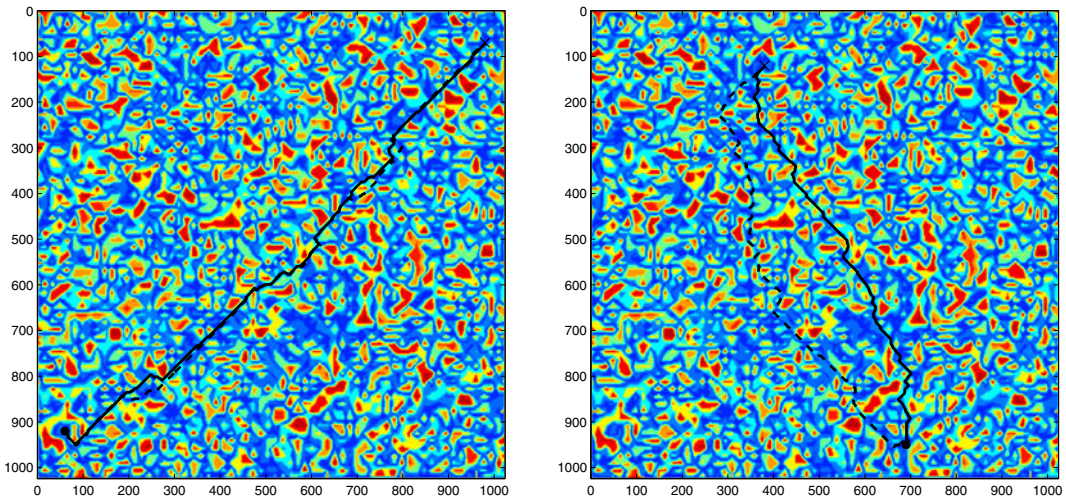


Fig. 10 A multiresolution approximation of the rectangular domain in (ρ, θ) system defined by the radii $\rho_{\min}$ and $\rho_{\max}$ under the inverse conformal mapping gives a multiresolution sector approximation of an annulus cut defined by the same radii



(a) Long travel scenario (b) Challenging scenario

Fig. 11 Qualitative comparison of path optimality. The *solid line* shows the optimal path while the *dashed line* gives the sequence of the visited nodes, i.e., the actual path followed by the agent. In the *right figure*, the early deviation of the path owing to the obstacles around the start node results in a different path than the optimal one obtained when full information about the environment is used. The difference in length between the two paths is not significant however

path-planning algorithm. It is seen that for the scenarios tested, the multiresolution path-planning algorithm gives a slightly longer path (about 5%) than the optimal one, while using less than 1% of the cells of the full information case. Furthermore, the task execution time (TET) for the multiresolution case is approximately 50–60% of the computational load of the fine resolution case.

The multiresolution path planning algorithm has been tested on a different environment, where large obstacles are supposed to stand in a possible route of the agent. Figure 12 compares the performance of the multiresolution algorithm with respect to the size of the fine resolution window. The dashed line represents the optimal path obtained from the standard path planning algorithm on a uniform, fine resolution grid. The solid line represents the path from the multiresolution algorithm followed by the agent. From the starting location (lower-left corner of the figure) to the goal (lower-right corner), the optimal path is found to pass through a narrow passage between the rectangle and the circular obstacles. However, in the first case in Fig. 12a, where a small size of fine resolution window is assumed, the multiresolution algorithm finds a different path that circumvents the

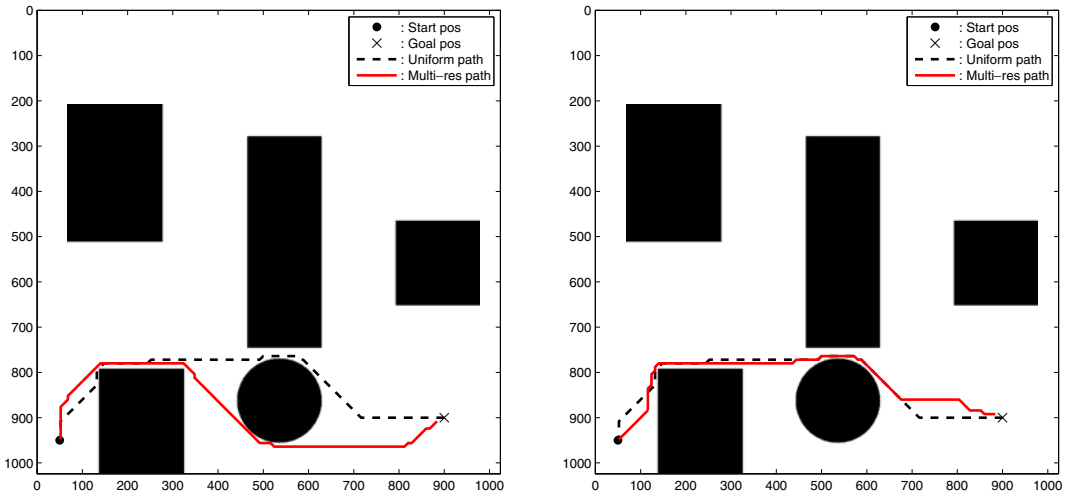
Table 1 Performance of full and multiresolution path-planning for the selected scenarios

Scenarios	No. of nodes[a]	TET[b] (s)	Path length[c]
Easy	1667 (0.64%)	53.19/4.49 (8.44%)	119/132 (110%)
Intermediate	1648 (0.63%)	94.92/5.60 (5.9%)	170/171 (101%)
Difficult	1636 (0.62%)	237.90/13.88 (5.83%)	418/442 (106%)
Long travel	1610 (0.61%)	236.40/16.82 (7.11%)	518/534 (103%)

[a] The total number of nodes determines the memory requirements. For the multiresolution case, the number represents the average number of nodes over $i = 0, \cdots, f$. For the full information case the number of nodes is fixed at 262144

[b] The task execution time (TET) varies depending on the location of the goal node relative to the start node

[c] Given a static scenario the full resolution path-planning yields a single optimal path for each case. For the multiresolution case the path length is the same as the size of the visited set at $t = t_f$. For simplicity, the length of the path is the number of discrete steps



(a) Short horizon (b) Long horizon

Fig. 12 Qualitative comparison of path optimality in an office environment. The *dashed line* shows the optimal path while the *solid line* represents the actual path followed by the agent. In the *right figure*, the increased fine resolution window helps to detect the narrow passage between obstacles. Hence, it results in a path similar to the optimal path

circular obstacle. This is explained by the fact that in the early stage of planning the multiresolution path planning algorithm is unable to recognize the narrow passage as probable obstacle region owing to the coarse level of representation at far away distance. Subsequently, the multiresolution algorithm suggests an alternative path routed below the circular obstacle. The multiresolution algorithm tends to stick to the earlier path, while trying to find a best path at each time step. In contrast, when the size of fine resolution is increased in Fig. 12b the multiresolution algorithm can see the narrow passage and result in a path close to the optimal one.

6.2 UAV Path-Planning

In this section we present simulation results of the proposed algorithm for a non-trivial scenario. In the first scenario, the environment $\mathscr{W}$ is an actual topographic (elevation) map of a certain state in the US. The objective is for the UAV to follow a path from start to destination (the latter denoted by an X in Fig. 13), while flying as low as possible, and below a certain elevation threshold. Blue color areas in Fig. 13 correspond to regions of low risk (elevation in this case) and red color areas correspond to regions of high risk that should be avoided. The environment is assumed to be square of dimension 128×128 units. Hence $N = 6$ is the finest resolution possible. We choose the fine level as $J_{\max} = 6$ and the coarse level as $J_{\min} = 3$. This choice makes the total number of nodes in the graph not to exceed the maximum count of 256 that corresponds to the maximum allowable variable size of the micro-controller used for this particular instance to implement the algorithm [15]. The ranges from the current location at distinct levels of resolution are selected as $(r_6, r_5, r_4) = (8, 15, 30)$.

Figure 13 shows the evolution of the path at different time steps, as the agent moves to the final destination. In each figure, the solid line represents the actual path followed by the agent, while the dashed-dot line represents the best proposed path to the destination at that particular instant.

We next present the results for the case of path-planning with sector-like cells, as described in the previous section. The environment in this case

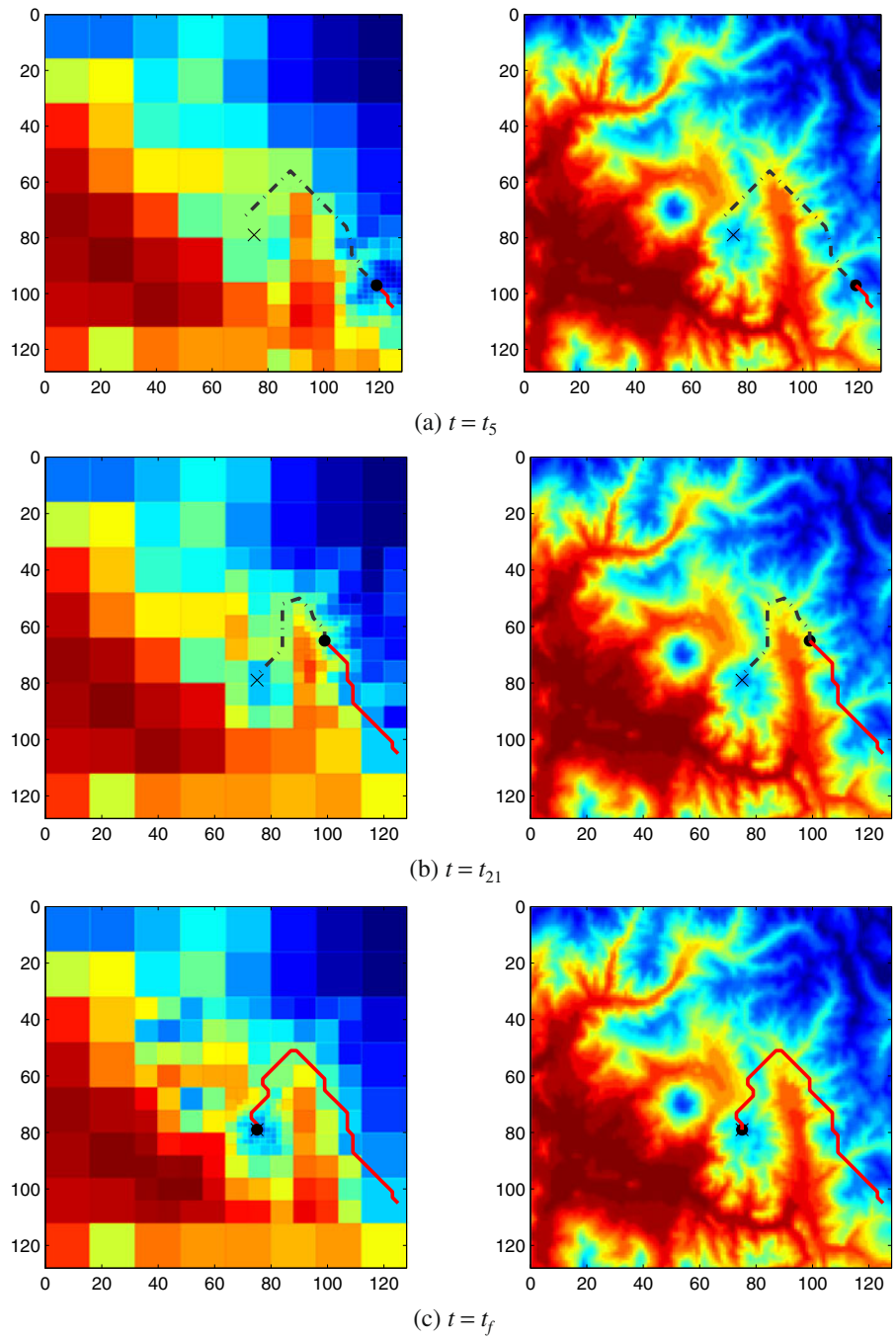
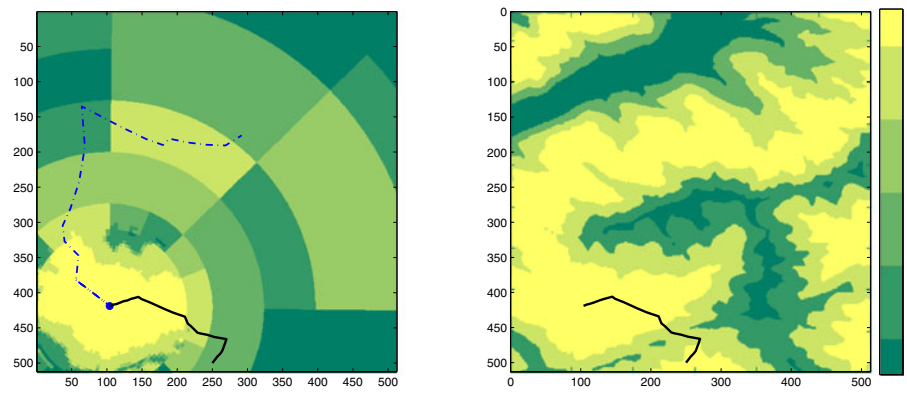


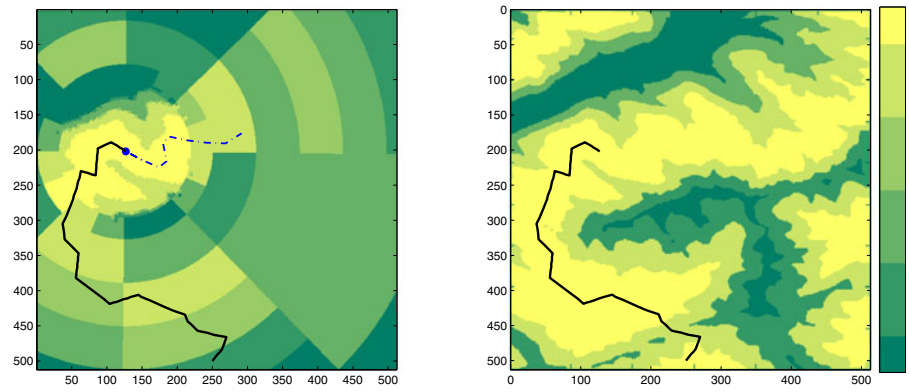
Fig. 13 Path evolution and replanning. Figures on the *left* show the multiresolution approximation of the environment with respect to the current location of the agent

is assumed to be square of dimension 512×512 units (pixels). Hence $N = 9$ is the finest resolution level. For simplicity, only two levels of resolution are chosen for this case. Inside a circle of radius of 100 unit cells, we employ a high resolution approximation; outside this area we employ a low resolution approximation of $\mathcal{W}'$.

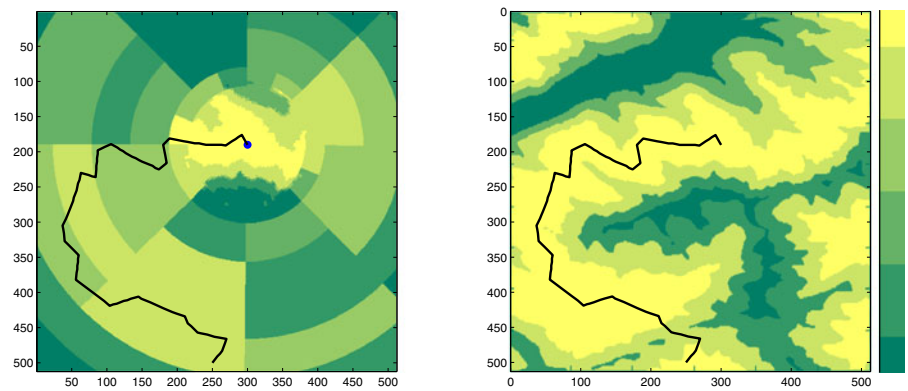
The results from the proposed multiresolution path-planning algorithm using a fine resolution level $J_{\max} = 5$, and a low resolution at level $J_{\min} =$



(a) $t = 10$



(b) $t = 22$



(c) $t = t_f$

Fig. 14 Path evolution with time. The figures show the actual path (*solid line*) along with the most recent tentative path (*dotted line*) of the agent at each time step. The agent reaches the final destination at $t = t_f$

3 are shown in Fig. 14. In the left column the local multiresolution view of the environment as seen from the onboard sensors is shown at different instances of time. The right column shows the environment and the path followed up to that instance in the original map of the environment.

7 Conclusions

Autonomous path-planning for small UAVs (and other small size autonomous vehicles) often introduces severe restrictions during control algorithm implementation, stemming from the limitations

imposed by the on-board hardware, as well as the requirement for real-time execution. In this work we have proposed a method to overcome this problem by using a new hierarchical, multiresolution path-planning scheme. The algorithm computes at each step a multiresolution representation of the environment using the wavelet transform. The idea is to employ high resolution close to the agent and coarse resolution at large distances from the current location of the agent, thus allotting the computational resources to the part of the problem (in the tempo-spatial domain) that is most relevant or urgent. As an added benefit of the proposed algorithm, the adjacency list of the resulting cell decomposition can be computed directly from the nonzero detail coefficients of the wavelet transform, thus speeding up the whole process. The algorithm is scalable and can be tailored to the available computational resources of the mobile agent.

Several improvements and refinements of the previous baseline path-planning algorithm are possible. First, for dynamically changing environments, an incremental implementation in the spirit of D* or LPA* [20, 32] in lieu of the Dijkstra's algorithm should result in a much more efficient algorithm. In particular, it can also handle the cases where the agent has no or partial knowledge of the environment a priori. Such an extension would use the prior graph structure at each replanning step to minimize redundancy. In the current implementation, during each replanning step, all the previous information is discarded. For an initial attempt towards this direction, see [9, 10]. Second, wavelet processing invites the use of other families of wavelets beyond the Haar system. The benefits from the use of non-Haar wavelets is still to be determined however. Third, multiresolution processing opens the avenue for more general connectivity relationships between the cells, beyond the standard 4-neighborhood of 8-neighborhood connectivity. See [26] for an implementation on path-planning problems using beamlet-like connectivity between the nodes in the underlying search graph. Finally, the extension to three-dimensional navigation problems is also possible, albeit somewhat cumbersome. Conceptually not much will change, besides the fact that one now has to deal with cubes instead of square cells. Three-dimensional wavelet transforms are available in the relevant literature, and can be applied to get the representations of the environment, vis-a-vis the 2D case. Despite the increased complexity introduced when dealing with the third dimension, the benefits of the multiresolution representation are expected to be even larger for the 3D case, as the number of cells in a decomposition increases faster for the 3D than the 2D case.

Acknowledgements This work has been supported in part by NSF (awards CMS-0510259 and CMMI-0856565) and ARO (awards W911NF-05-1-0331 and W911NF-11-1-0046). The third author also acknowledges support from the A. Onassis Public Benefit Foundation.

References

1. Bakolas, E., Tsiotras, P.: Multiresolution path planning via sector decompositions compatible to on-board sensor data. In: AIAA Guidance, Navigation, and Control Conference. Honolulu, HI, 18–21 August 2008. AIAA Paper 2008-7238
2. Behnke, S.: Local Multiresolution Path Planning. Lecture Notes in Computer Science, pp. 332–343 (2004)
3. Bi, Z., Yimin, Y., Wei, Y.: Hierarchical path planning approach for mobile robot navigation under the dynamic environment. In: IEEE International Conference on Industrial Informatics, Daejeon, Korea, pp. 372–376 (2008). doi:10.1109/INDIN.2008.4618127
4. Broz, P., Kolingerova, I., Apu, R.A., Gavrilova, M., Zitka, P.: Path planning in dynamic environment using an adaptive mesh. In: Spring Conference on Computer Graphics SCCG 2007, pp. 172–178 (2007)
5. Burrus, C.S., Gopinath, R.A., Guo, H.: Introduction to Wavelets and Wavelet Transforms. Prentice Hall, New Jersey (1998)
6. Calderbank, A.R., Daubechies, I., Sweldens, W., Yeo, B.L.: Wavelet transforms that map integers to integers. Appl. Comput. Harmon. Anal. **5**(3), 332–369 (1998)
7. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms, 2nd edn. MIT Press, Cambridge (2001)
8. Cowlagi, R., Tsiotras, P.: Beyond quadtrees: cell decomposition for path planning using the wavelet transform. In: 46th IEEE Conference on Decision and Control, New Orleans, pp. 1392–1397 (2007)
9. Cowlagi, R., Tsiotras, P.: Multiresolution path planning with wavelets: a local replanning approach. In: American Control Conference, Seattle, pp. 1220–1225 (2008)
10. Cowlagi, R., Tsiotras, P.: Multi-resolution path planning: theoretical analysis, efficient implementation, and extensions to dynamic environments. In: 49th

IEEE Conference on Decision and Control, Atlanta, pp. 1384–1390 (2010)
11. Daubechies, I., Sweldens, W.: Factoring wavelets transforms into lifting steps. J. Fourier Anal. Appl. **4**(3), 247–269 (1998)
12. Freeden, W., Windheuser, U.: Spherical wavelet transform and its discretization. Adv. Comput. Math. **5**(1), 51–94 (1996). doi:10.1007/BF02124735
13. Hwang, J.Y., Kim, J.S., Lim, S.S., Park, K.H.: A fast path planning by path graph optimization. IEEE Trans. Syst. Man Cybern. **33**(1), 121–127 (2003)
14. Jung, D.: Hierarchical path planning and control of a small fixed-wing UAV: theory and experimental validation. Ph.D. Thesis, Georgia Institute of Technology, Atlanta (2007)
15. Jung, D., Ratti, J., Tsiotras, P.: Real-time implementation and validation of a new hierarchical path planning scheme for UAVs via hardware-in-the-loop simulation. J. Intell. Robot. Syst. **54**(1), 163–181 (2009). doi:10.1007/s10846-008-9255-0
16. Jung, D., Tsiotras, P.: Multiresolution on-line path planning for small unmanned aerial vehicles. In: American Control Conference, Seattle, pp. 2744–2749 (2008)
17. Kambhampati, S., Davis, L.S.: Multiresolution path planning for mobile robots planning. IEEE J. Robot. Autom. **2**(3), 135–145 (1986)
18. Kim, C.T., Lee, J.J.: Mobile robot navigation using multi-resolution electrostatic potential field. In: 32nd Annual Conference of IEEE Industrial Electronics Society, 2005, IECON 2005 (2005)
19. Koenig, A.: Agent-centered search. Artif. Intell. Mag. **22**(4), 109–131 (2001)
20. Koenig, S., Likhachev, M., Furcy, D.: Lifelong planning A*. Artif. Intell. J. **155**(1–2), 93–146 (2004)
21. Latombe, J.C.: Robot Motion Planning. Kluwer Academic Publishers, Boston (1991)
22. LaValle, S.M.: Planning Algorithms. Cambridge University Press, Cambridge (2006)
23. Li, T.H.: Multiscale representation and analysis of spherical data by spherical wavelets. SIAM J. Sci. Comput. **21**(3), 924–953 (1999). do10.1137/S1064827598341463
24. Li, Y., Li, C., Zhang, Z.: Q-learning based method of adaptive path planning for mobile robot. In: 2006 IEEE International Conference on Information Acquisition, Shandong, China, pp. 983–987 (2006). doi:10.1109/ICIA.2006.305871
25. Longega, L., Panzieri, S., Pascucci, F., Ulivi, G.: Indoor robot navigation using log-polar local maps. In: Robot Control 2003 (SYROCO'03): a Proceedings Volume from the 7th IFAC Symposium, Wrocław, Poland, 1–3 September 2003, Pergamon, p. 213 (2004)
26. Lu, Y., Huo, X., Tsiotras, P.: Beamlet-like data processing for accelerated path-planning using multiscale information of the environment. In: 49th IEEE Conference on Decision and Control, Atlanta, pp. 3808–3813 (2010)
27. Noborio, H., Naniwa, T., Arimoto, S.: A quadtree-based path planning algorithm for a mobile robot. J. Robot. Syst. **7**(44), 555–574 (1990)
28. Pai, D.K., Reissell, L.M.: Multiresolution rough terrain motion planning. IEEE Trans. Robot. Autom. **14**, 19–33 (1998)
29. Prazenica, R.J., Kurdila, A.J., Sharpley, R.C., Evers, J.: Multiresolution and adaptive path planning for maneuver of micro-air-vehicles in urban environments. In: AIAA Guidance, Navigation, and Control Conference and Exhibit, San Francisco (2005)
30. Sermanet, P., Hadsell, R., Scoffier, M., Muller, U., LeCun, Y.: Mapping and planning under uncertainty in mobile robots with long-range perception. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2525–2530 (2008). doi:10.1109/IROS.2008.4651203
31. Sinopoli, B., Micheli, M., Donato, G., Koo, T.J.: Vision based navigation for an unmanned aerial vehicle. In: Proceedings of 2001 IEEE Conference on Robotics and Automation, pp. 1757–64 (2001)
32. Stentz, A.: Optimal and efficient path planning for partially-known environments. In: Proceedings of the IEEE International Conference on Robotics and Automation, vol. 4, pp. 3310–3317 (1994)
33. Sweldens, W.: The lifting scheme: a construction of second generation wavelets. SIAM J. Math. Anal. **29**(2), 511–546 (1997)
34. Tsiotras, P., Bakolas, E.: A hierarchical on-line path-planning scheme using wavelets. In: European Control Conference, Kos, Greece, pp. 2806–2812 (2007)
35. Yu, H., Beard, R., Byrne, J.: Vision-based local multiresolution mapping and path planning for miniature air vehicles. In: American Control Conference, St. Louis, pp. 5247–5252. (2009). doi:10.1109/ACC.2009.5160065
36. Zhu, D., Latombe, J.: New heuristic algorithms for efficient hierarchical path planning. IEEE Trans. Robot. Autom. **7**(1), 9–20 (1991)