A Hierarchical On-Line Path Planning Scheme using Wavelets

Panagiotis Tsiotras and Efstathios Bakolas

***Abstract*— We present an algorithm for solving the shortest (collision-free) path planning problem for an agent (e.g., wheeled vehicle, UAV) operating in a partially known environment. The agent has detailed knowledge of the environment and the obstacles only in the vicinity of its current position. Far away obstacles or the final destination are only partially known and may even change dynamically at each instant of time. We obtain an approximation of the environment at different levels of fidelity using a wavelet approximation scheme. This allows the construction of a directed weighted graph of the obstacle-free space in a computationally efficient manner. In addition, the dimension of the graph can be adapted to the on-board computational resources. By searching this graph we find the desired shortest path to the final destination using Dijkstra's algorithm, provided that such a path exists. Simulations are presented to test the efficiency of the algorithm using non trivial scenarios.**

Keywords: mobile agent, wavelet decomposition, adjacency matrix, shortest path, collision avoidance, Dijkstra's algorithm.

I. Introduction

The path planning problem has been the focus of attention for many years in several research areas, ranging from operation research, vehicle navigation, network optimization, etc. In this work we deal with the vehicle navigation problem (e.g., ground vehicle, UAV) moving in a partially known environment $\mathcal{W}$. The general framework for this class of problems is as follows: *given a topological space $\mathcal{F} \subset W$ of admissible states* x*, find a path connecting the prescribed initial state $\mathsf{x}_0 \in \mathcal{F}$ with the destination state $\mathsf{x}_f \in \mathcal{F}$ such that the path lies completely inside $\mathcal{F}$*. The construction of such a path can take place using either a continuous representation of $\mathcal{W}$ or a discrete one. In this work we propose an algorithm which solves the path planning problem using an adaptive, discrete, cell-based approximation of $\mathcal{W}$.

A common method for solving path planning problems using cell-based approximations of the environment is to employ a graph representation. By transcribing the free space $\mathcal{F}$ on a graph $\mathcal{G}$ the problem reduces to one of finding a sequence of adjacent nodes in the graph $\mathcal{G}$ from the current node to the destination node. The problem therefore is reduced to a network minimization or a graph search problem. The difficulty of this approach lies in the fact that the dimension of $\mathcal{G}$ (i.e., the number of nodes) becomes very large as the fidelity of the approximation of $\mathcal{F}$ and/or $\mathcal{W}$ increases. Sooner or later, the computational resources on-board the agent reach their limit; hence, path planning algorithms that use such global graph representations of the free environment are limited in their capacity to deal with rapidly changing situations (e.g., moving obstacles or popup threats). In most realistic situations where continuous, on-line replanning of the optimal path to the target is required, high-accuracy global methods are impractical. Local approximations reduce the computational complexity but may fail to find a solution even if such a solution exists (see Fig. 1).

A common method for solving path planning problems using cell-based approximations of the environment is to employ a graph representation. However, the dimension of the graph becomes very large as the fidelity of a global approximation of the environment increases. In this work we introduce a hybrid local/global path planning algorithm that uses district levels of fidelity (resolution) of the environment 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. Far away obstacles or threats do not (or should not) have a large effect of the vehicle's *immediate* motion. Therefore, the most accurate and reliable information of the environment is required at the vicinity of the vehicle. Second, the plethora of sensory information used by typical robotic vehicles (e.g., cameras, radars, laser scanners, satellite imagery) do have different ranges and resolutions. A computationally efficient path planning algorithm should be able to blend together the information provided by all these sensors and focus its computational resources on the part of the path (spatial and temporal) that needs it most. In a nutshell, a computationally efficient algorithm suitable for *on-line* implementation should be able to combine short-term tactics (reaction to unforeseen threats) with long-term strategy (planning towards the ultimate goal).

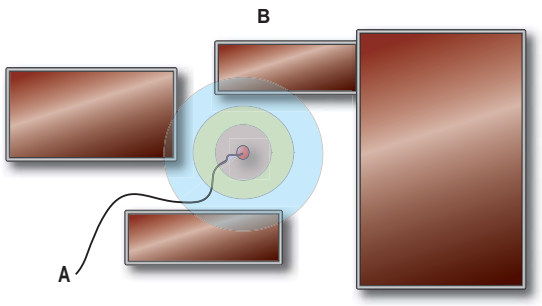


Fig. 1. A path based on a short visibility/exploration horizon may fail to find a feasible path even if such a path exists. As the exploration horizon is increased the construction of a collision-free path becomes more likely. The increase of the exploration horizon however comes at the expense of an enormous increase of the required on-board computational resources. Here A is the starting point and B is the final point.

P. Tsiotras is a Professor at the School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0150, USA, Email: tsiotras@gatech.edu

E. Bakolas is a graduate student at the School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0150, USA, Email: gth714d@mail.gatech.edu

The success of a multi-resolution path planning algorithm hinges on its ability to compute the obstacle boundaries well in advance and with sufficient accuracy, by keeping a balance between an overconservative approximation of the environment and the computation of a collision-free path.

In this work we use wavelets to obtain multiresolution approximations of the configuration space at different distances from the vehicle. The wavelet transform provides a very fast decomposition[1] of the environment at different levels of resolution. The number of resolution levels, their scale, and range can all be readily adapted at each time step to yield graph representations that are commensurate to the available on-board computational resources.

We employ the hierarchical path planning principle to find the optimal path on a topological graph $\mathcal{G}$ induced by the cell decomposition generated by the wavelet transform. Namely, the path may contain mixed nodes at all resolution levels except the finer resolution level, where it is assumed that nodes can be confidently resolved as either free or occupied. Mixed nodes, on the other hand are not known with certainty whether they belong to the free or the obstacle space. Hierarchical path planning is known to be more flexible than methods that search only through free nodes [2]. In hierarchical path planning the mixed nodes are subsequently resolved to free or occupied nodes, as the agent gets closer to the obstacles and more information about their shape and location becomes available.

Several multi-resolution or hierarchical algorithms have been proposed in the literature for path planning [3], [4], [5], [6], [7], [8], [9]. The majority of those use some form of quadtree decomposition of the environment. One drawback of quadtree-based decompositions is the use of a fine resolution close to the boundaries of all obstacles, regardless of their distance from the agent. This tends to waste computational resources. One of the central references in the context of such quadtree-based cell decompositions is perhaps [8], where the authors present a hierarchical path planning scheme based on a multistage quadtree decomposition. Both free and mixed nodes are included in the search, which is conducted using the A^* algorithm. A 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 in many cases and easy to implement, it fails to take full advantage of the local information around the agent. Wavelets for multi-resolution decomposition of the environment have also been used in [7]. The approach in [7] combines a more efficient model for the local behavior of the approximation, with improved computational characteristics, compared to the one proposed in [8]. A motion planning strategy which combines all the past lower resolution information with the one that is available at each stage is developed. The main emphasis in [8] however is to construct a smooth path. This is easily achieved using the information provided by the detail coefficients in the wavelet expansion. The smoothness requirement is then embedded in the transition cost of the agent. The reference most closely related to our approach is [9]. Therein, the author also uses the idea of coarse/fine grid at close/far 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 [9] requires a rather careful handling of the cell connectivity at the boundaries between two different resolution levels. This is handled automatically in our approach.

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

II. Technical Background

In this section we present some basic notions that we use extensively in this work. They can be found in the robot motion planning literature [10], [11]. For a more detailed presentation of the shortest path problem the reader is encouraged to consult one of the excellent resources on this topic [12], [13].

A. Topology of the problem

In this work we employ a point mass representation of the agent. We assume that the agent is allowed to navigate in the world environment $\mathcal{W} \subset \mathbb{R}^2$. Given $\mathcal{W}$ and the obstacle-free configuration space $\mathcal{F} \subset W$ that contains all the feasible states of the agent, our objective is to find a continuous function $\pi : [0,1] \mapsto \mathcal{F}$, which we call the "path," such that $\pi(0) = \mathsf{x}_0$ and $\pi(1) = \mathsf{x}_f$, where $\mathsf{x}_0, \mathsf{x}_f \in \mathcal{F}$ are given initial and final states respectively.

The space $\mathcal{F}$ may be path connected or not path connected. In the latter case, the problem is infeasible. Path planning algorithms like Dijkstra's algorithm and the A^* algorithm will always find a path if $\mathcal{F}$ is path connected and will determine if the problem is infeasible for the given x_0 and x_f if $\mathcal{F}$ is not path connected.

We denote by $\mathcal{O}_i \subset \mathcal{W}$ obstacles in $\mathcal{W}$. Assuming there are $M \geq 1$ obstacles in $\mathcal{W}$, the obstacle space $\mathcal{O} \triangleq \mathcal{W} \backslash \mathcal{F}$ is then given by

$$\mathcal{O} = \bigcup_{i=1}^{M} \mathcal{O}_i. \tag{1}$$

For simplicity, we will assume no overlapping between the obstacles, so that $\mathcal{O}_i \cap \mathcal{O}_j = \varnothing$ for all $1 \leq i, j \leq M$ with $i \neq j$.

B. Cell decomposition of $\mathcal{W}$ and graph representation of $\mathcal{F}$

In Section III we present a cell decomposition that will allow us to efficiently partition $\mathcal{W}$ into classes of cells of different dimension. First, recall that an m-cell decomposition $\mathcal{C}_d$ of $\mathcal{W}$ is a finite collection of m cells

$$\mathcal{C}_d = \{c_i \in \mathcal{W} : \ i = 1, \ldots, m\} \tag{2}$$

with the following properties:

1) $\mathcal{W} = \bigcup_{i=1}^{m} c_i$
2) $\operatorname{int} c_i \bigcap \operatorname{int} c_j = \varnothing$

Given two cell decompositions $\mathcal{C}_d$ and $\mathcal{C}'_d$ of $\mathcal{W}$ we say that $\mathcal{C}'_d$ is a *finer, or higher resolution* decomposition of $\mathcal{W}$ than $\mathcal{C}_d$ if and only if for every cell $c_i \in \mathcal{C}_d$ there exists an integer $p_i > 1$ such that $c_i = \bigcup_{l=1}^{p_i} c'_l$ where $c'_l \in \mathcal{C}'_d$.

We may define the following three categories of cells:

1) empty cells, when $c_i \cap \mathcal{O} = \varnothing$
2) mixed cells, when $c_i \cap \mathcal{O} \neq \varnothing$
3) full cells, when $c_i \subseteq \mathcal{O}$.

We will say that two cells c_i and c_j are adjacent[2] if

$$\partial c_i \cap \partial c_j \neq \varnothing, \quad i \neq j, \tag{3}$$

where ∂c_i denotes the boundary of the cell c_i.

To a cell decomposition $\mathcal{C}_d$ we will associate a directed graph $\mathcal{G} = (V, E)$ with nodes V and edges E, known as the connectivity graph, such that:

1) The nodes of $\mathcal{G}$ correspond to the free and mixed cells of $\mathcal{C}_d$
2) The edges of $\mathcal{G}$ correspond to cells that are adjacent to each other

It is easy to see that $\mathcal{G}$ is a topological graph [10].

III. A Multiresolution Decomposition of $\mathcal{W}$

A. The 2D wavelet transform

The idea behind the theory of the wavelet transform is to represent a function $f \in \mathcal{L}^2(\mathbb{R})$ as a summation of elementary basis functions $\phi_{J,k}$ and $\psi_{j,k}$ as follows

$$f(x) = \sum_{k\in\mathbb{Z}} a_{J,k}\phi_{J,k}(x) + \sum_{j\geq J}\sum_{k\in\mathbb{Z}} d_{j,k}\psi_{j,k}(x), \tag{4}$$

where $\phi_{j,k}(x) = 2^{j/2}\phi(2^j x - k)$ and $\psi_{j,k} = 2^{j/2}\psi(2^j x - k)$. In the ideal case both $\phi(x)$ (scaling function) and $\psi(x)$ (mother wavelet) have compact support or they decay very fast outside a small interval so they can capture localized features of f. The first summation in (4) gives a low resolution or coarse approximation of f. The second term in (4) gives the difference (details) between the original function and its low resolution approximation. For example, when analyzing a signal at the coarsest level (low resolution) only the general, most salient features of the signal will be revealed. The index j denotes the resolution level. For each increasing index j, a higher, or finer resolution term is added, which adds more and more details. The expansion (4) thus reveals the properties f at different levels of resolution [14], [15], [16].

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

$$\Phi_{j,k,\ell}(x,y) = \phi_{j,k}(x)\phi_{j,\ell}(y) \tag{5}$$
$$\Psi^1_{j,k,\ell}(x,y) = \phi_{j,k}(x)\psi_{j,\ell}(y) \tag{6}$$
$$\Psi^2_{j,k,\ell}(x,y) = \psi_{j,k}(x)\phi_{j,\ell}(y) \tag{7}$$
$$\Psi^3_{j,k,\ell}(x,y) = \psi_{j,k}(x)\psi_{j,\ell}(y) \tag{8}$$

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

Given a function $f \in \mathcal{L}^2(\mathbb{R}^2)$ we can then write

$$\begin{aligned} f(x,y) = &\sum_{k,\ell\in\mathbb{Z}} a_{J,k,\ell}\Phi_{J,k,\ell}(x,y) \\ &+ \sum_{i=1}^{3}\sum_{j\geq J}\sum_{k,\ell\in\mathbb{Z}} d^i_{j,k,\ell}\Psi^i_{j,k,\ell}(x,y), \end{aligned} \tag{9}$$

where, for the case of orthonormal wavelets the approximation coefficients are given by[3]

$$a_{j,k,\ell} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\,\Phi_{j,k,\ell}(x,y)\,\mathrm{d}x\,\mathrm{d}y \tag{10}$$

and the detail coefficients by

$$d^i_{j,k,\ell} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\,\Psi^i_{j,k,\ell}(x,y)\,\mathrm{d}x\,\mathrm{d}y. \tag{11}$$

The key property of wavelets used in this paper is the fact that the expansion (9) induces the following decomposition of $\mathcal{L}^2(\mathbb{R}^2)$

$$\mathcal{L}^2(\mathbb{R}^2) = \mathcal{V}_J \oplus \mathcal{W}^{detail}_J \oplus \mathcal{W}^{detail}_{J+1} \oplus \cdots \tag{12}$$

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

In this paper we use Haar wavelets for reasons that will become apparent below. 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 [14], [17]. Subsequently, we may associate the functions $\Phi_{j,k,\ell}$ and $\Psi^i_{j,k,\ell}$ $(i = 1, 2, 3)$ with the rectangular cell $c^j_{k,\ell} \triangleq I_{j,k} \times I_{j,\ell}$.

B. Wavelet decomposition of the risk measure

Without loss of generality, in the sequel we take $\mathcal{W} = [0,1] \times [0,1]$, which is described 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. It follows from the previous discussion that the Haar wavelet decomposition of a function f defined over $\mathcal{W}$ at resolution level $J \geq J_{\min}$

$$\begin{aligned} f(x,y) = &\sum_{k,\ell=0}^{2^{J_{\min}}-1} a_{J_{\min},k,\ell}\,\Phi_{J_{\min},k,\ell}(x,y) \\ &+ \sum_{i=1}^{3}\sum_{j=J_{\min}}^{J-1}\sum_{k,\ell=0}^{2^j-1} d^i_{j,k,\ell}\Psi^i_{j,k,\ell}(x,y) \end{aligned} \tag{13}$$

induces a cell decomposition of $\mathcal{W}$ of square cells of size $1/2^J \times 1/2^J$.

Assume now that we are given a function $\mathsf{rm} : \mathcal{W} \mapsto [0,1]$ that represents the "risk measure" at the location $\mathsf{x} = (x, y)$. For instance, one may choose

$$\mathsf{rm}(\mathsf{x}) = \begin{cases} (d_{\max} - \min_{\mathsf{y}\in\mathcal{O}} \|\mathsf{x} - \mathsf{y}\|_\infty)/d_{\max}, & \text{if } \mathsf{x} \in \mathcal{F}, \\ 1, & \text{if } \mathsf{x} \in \mathcal{O}, \end{cases} \tag{14}$$

[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 in (10) and (11).

where $d_{\max} \triangleq \max_{\mathsf{x}\in\mathcal{F}} \min_{\mathsf{y}\in\mathcal{O}} \|\mathsf{x}-\mathsf{y}\|_\infty$. Alternatively, one may think of rm as the probability that $(x, y) \in \mathcal{O}$.

We will use the $\|\cdot\|_\infty$ norm to measure distances in $\mathcal{W}$. Consequently, all points within range r from the current location of the agent are given by

$$\mathcal{N}(\mathsf{x}, r) \triangleq \{\mathsf{y} \in \mathcal{W} : \|\mathsf{x}-\mathsf{y}\|_\infty \le r\}. \tag{15}$$

Suppose now that we are given the desired levels of resolution of $\mathcal{W}$ as $J_{\min} \le j \le J_{\max}$ where $J_{\min}, J_{\max} \in \{1, \dots, N\}$, with corresponding ranges r_j from the agent's current location. By this we mean that we wish all points $\mathsf{y} \in \mathcal{N}(\mathsf{x}, r_{J_{\max}})$ to be described by resolution $J_{\max}$, all points $\mathsf{y} \in \mathcal{N}(\mathsf{x}, r_{j-1})\backslash\mathcal{N}(\mathsf{x}, r_j)$ to be described by resolution j, where $J_{\min} < j \le J_{\max}$, and all points $\mathsf{y} \notin \mathcal{N}(\mathsf{x}, r_{J_{\min}+1})$ to be described by resolution $J_{\min}$. Since we require finer resolution closer to the agent we assume, of course, that $r_{j-1} > r_j$. The situation is depicted in Fig. 2.

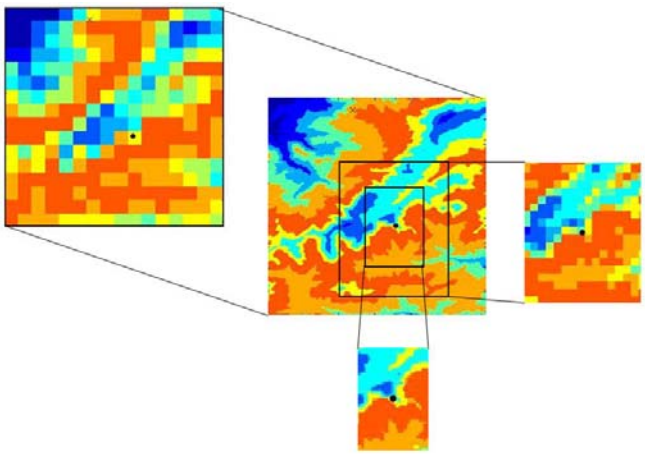


Fig. 2. Multiresolution representation of the environment according to the distance from the current location of the agent.

The choice of $J_{\max}$ is dictated by the requirement that at this level all cells can be resolved 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.

We obtain the distinct resolution levels at the given required distances from the current location of the agent by applying the Haar wavelet transform to rm. The use of Haar wavelets is mainly dictated by the choice of the norm in (15). To this end, let $\mathcal{I}(j) \triangleq \{0, 1, \dots, 2^j - 1\}$ and let

$$\mathcal{K}(j) \triangleq \{k \in \mathcal{I}(j) : I_{j,k} \cap [x_0 - r_j, x_0 + r_j] \neq \varnothing\},$$
$$\mathcal{L}(j) \triangleq \{\ell \in \mathcal{I}(j) : I_{j,\ell} \cap [y_0 - r_j, y_0 + r_j] \neq \varnothing\},$$

where (x_0, y_0) is the current location of the agent. The wavelet decomposition of rm, given by

$$\begin{aligned}\mathsf{rm}(x, y) = &\sum_{k,\ell\in\mathcal{I}(J_{\min})} a_{J_{\min},k,\ell}\, \Phi_{J_{\min},k,\ell}(x, y) \\ &+ \sum_{i=1}^{3} \sum_{j=J_{\min}}^{J_{\max}-1} \sum_{\substack{k\in\mathcal{K}(j)\\ \ell\in\mathcal{L}(j)}} d^i_{j,k,\ell} \Psi^i_{j,k,\ell}(x, y)\end{aligned} \tag{16}$$

induces, via a slight abuse of notation, the following cell decomposition on $\mathcal{W}$

$$\mathcal{C}_d = \Delta C_d^{J_{\min}} \oplus \cdots \oplus \Delta C_d^{J_{\max}}. \tag{17}$$

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

IV. Cost Assignment

Each cell $c^j_{k,\ell}$ in the cell decomposition (17) has a value $\mathsf{val}(c^j_{k,\ell})$, which for the case of Haar wavelets is the weighted average of the risk measure function over the cell. To the cell decomposition (17) we now assign a graph $\mathcal{G}$ having as nodes $V(\mathcal{G})$ all the cells with $\mathsf{val}(c^j_{k,\ell})$ less than or equal to a certain constant. The edges $E(\mathcal{G})$ of $\mathcal{G}$ correspond to the adjacency relationships of $V(\mathcal{G})$, as usual. Clearly, there is an one-to-one correspondence between the elements of $V(\mathcal{G})$ and the free and mixed cells of $\mathcal{C}_d$. We write $v \sim c^j_{k,\ell}$ to denote this correspondence. Moreover, since $\mathcal{G}$ is a topological graph we may associate each node $v \in V(\mathcal{G})$ with any point $\mathsf{x} \in c^j_{k,\ell}$. Without loss of generality we choose the center of the cell. Let $\mathsf{cell}_{\mathcal{G}}(v)$ denote the center of the corresponding cell in this case. Finally, if $\mathsf{x} \in c^j_{k,\ell}$ we will write $v = \mathsf{node}_{\mathcal{G}}(\mathsf{x})$ where $v \sim c^j_{k,\ell}$.

To each edge $(u, v) \in E(\mathcal{G})$ we assign a cost $\mathcal{J}(u, v)$, which is the cost of transitioning from node u to node v. We may use the transition cost as follows

$$\mathcal{J}(u, v) = \mathsf{rm}(\mathsf{cell}_{\mathcal{G}}(v)). \tag{18}$$

That is, the cost of transitioning from node u to the adjacent node v depends only on the risk measure of the final node, v and is independent of the starting node u. This situation is depicted in Fig. 3. Note that, in general, $\mathcal{J}(u, v) \neq \mathcal{J}(v, u)$.

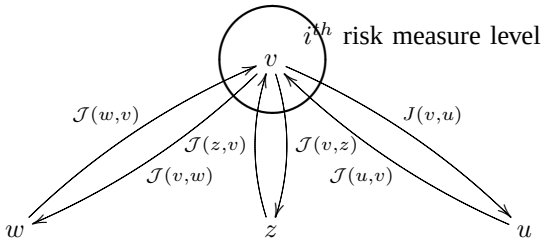


Fig. 3. The costs $\mathcal{J}(u, v)$, $\mathcal{J}(w, v)$, $\mathcal{J}(z, v)$, are all equal to $\mathsf{rm}(\mathsf{cell}_{\mathcal{G}}(v))$. Since the connectivity graph is directed, the costs $\mathcal{J}(v, u)$, $\mathcal{J}(v, w)$, $\mathcal{J}(v, z)$ may be different, depending on the risk measure level of the nodes u, w, z.

Another alternative would be to choose the transition cost so as to also penalize the (Euclidean) distance between $\mathsf{cell}_{\mathcal{G}}(u)$ and $\mathsf{cell}_{\mathcal{G}}(v)$. In this case the cost becomes

$$\mathcal{J}(u, v) = \mathsf{rm}(\mathsf{cell}_{\mathcal{G}}(v)) + \alpha\|\mathsf{cell}_{\mathcal{G}}(u) - \mathsf{cell}_{\mathcal{G}}(v)\|_2. \tag{19}$$

where $\alpha \ge 0$ is a weight constant. The larger the α the more emphasis we place on a shorter path.

Suppose now that we are given a path of q consecutive, adjacent nodes in $\mathcal{G}$ as follows $\mathcal{P} = (v_0, v_1, \dots, v_q)$. We can then assign a cost to each node in the path $\mathcal{P}$, induced by the two-node transitioning cost, iteratively, via

$$\mathcal{H}(v_i) = \mathcal{H}(v_{i-1}) + \mathcal{J}(v_{i-1}, v_i), \quad i = 1, \dots, q. \tag{20}$$

The value of $\mathcal{H}(v_k)$ represents the (accumulated) cost of the path from v_0 to v_k $(k \leq q)$. The shortest path problem is then to find a path that minimizes the accumulated cost from the initial to the destination node, or determine that such a path does not exist.

A widely used algorithm which solves the shortest path problem is Dijkstra's algorithm. Dijkstra's algorithm is based on a greedy strategy and always finds the optimal solution, provided that this solution exists. Dijkstra's algorithm is computationally more appealing than other standard shortest path algorithms, like Bellman-Ford's algorithm [13], especially when the adjacency matrix of $\mathcal{G}$ is sparse. An alternative approach is to use A^* if we have a good heuristic that underestimates the distance from the current node in the path to the destination node. For simplicity, in this paper we use Dijkstra's algorithm using the transition cost in (18) or (19).

V. Multiresolution Path Planning

The proposed multiresolution path planning algorithm proceeds as follows. Starting from $\mathsf{x}(t_0) = \mathsf{x}_0$ at time $t = t_0$, we construct using the approach of Section III, a cell decomposition $\mathcal{C}_d(t_0)$ of $\mathcal{W}$. Let the corresponding graph be $\mathcal{G}(t_0)$ and let $v_1^0 \in \mathcal{G}(t_0)$ and $v_f^0 \in \mathcal{G}(t_0)$ be the initial and final nodes, respectively such that $v_1^0 = \mathsf{node}_{\mathcal{G}(t_0)}(\mathsf{x}_0)$ and $v_f^0 = \mathsf{node}_{\mathcal{G}(t_0)}(\mathsf{x}_f)$. Using Dijkstra's algorithm (or any other similar algorithm) we find a path $\mathcal{P}(t_0)$ in $\mathcal{G}(t_0)$ of free and mixed nodes from v_1^0 to v_f^0 assuming that such a path exists. Let $\mathcal{P}(t_0)$ be given by the ordered sequence of l_0 nodes as follows

$$\mathcal{P}(t_0) = (v_1^0, v_2^0, \cdots, v_{l_0-1}^0, v_{l_0}^0 = v_f^0).$$

It is assumed that v_2^0 is free owing to the high resolution decomposition of $\mathcal{W}$ close to x_0. The agent subsequently moves from v_1^0 to v_2^0. Let now t_1 be the time the agent is at the location $\mathsf{x}(t_1) = \mathsf{cell}_{\mathcal{G}(t_0)}(v_2^0)$ and let $\mathcal{C}_d(t_1)$ be the multiresolution cell decomposition of $\mathcal{W}$ around $\mathsf{x}(t_1)$ with corresponding graph $\mathcal{G}(t_1)$. Applying again Dijkstra's algorithm we find a (perhaps new) path in $\mathcal{G}(t_1)$ from $v_1^1 = \mathsf{node}_{\mathcal{G}(t_1)}(\mathsf{x}(t_1))$ to $v_f^1 = \mathsf{node}_{\mathcal{G}(t_1)}(\mathsf{x}_f)$ if such a path exists. Let $\mathcal{P}(t_1)$ be given by the ordered sequence of l_1 nodes as follows

$$\mathcal{P}(t_1) = (v_1^1, v_2^1, \cdots, v_{l_1-1}^1, v_{l_1}^1 = v_f^1).$$

The agent subsequently moves to $\mathsf{x}(t_2) = \mathsf{cell}_{\mathcal{G}(t_1)}(v_2^1)$ at time t_2.

In general, 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 $\mathcal{G}(t_i)$. Dijkstra's algorithm yields a path $\mathcal{P}(t_i)$ in $\mathcal{G}(t_i)$ of mixed and free noes of length l_i,

$$\mathcal{P}(t_i) = (v_1^i, v_2^i, \cdots, v_{l_i-1}^i, v_{l_i}^i = v_f^i),$$

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. 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 $\mathsf{node}_{\mathcal{G}(t_0)}(\mathsf{x}(t_0)), \mathsf{node}_{\mathcal{G}(t_1)}(\mathsf{x}(t_1)), \ldots, \mathsf{node}_{\mathcal{G}(t_f)}(\mathsf{x}(t_f))$. Since the connectivity graph $\mathcal{G}(t)$ changes at each time step, it is therefore possible that the same state x may be visited twice since it may correspond to nodes of two distinct graphs. That is, it is possible that

$$\mathsf{cell}_{\mathcal{G}(t_i)}(v_k^i) = \mathsf{cell}_{\mathcal{G}(t_j)}(v_m^j), \quad i \neq j. \tag{21}$$

This will cause the agent to repeat the previous (optimal) decision ending up in a continuous loop. In order to avoid such pathological situations, we maintain a list $\mathsf{L}_{\mathsf{Visited}} = \{\mathsf{x}(t_0), \mathsf{x}(t_1), \ldots, \mathsf{x}(t_i)\}$ of all visited states up to the current time step t_i. At the next time step t_{i+1} we remove from $V(\mathcal{G}(t_{i+1}))$ all nodes v such that

$$\mathsf{cell}_{\mathcal{G}(t_{i+1})}(v) \in \mathsf{L}_{\mathsf{Visited}}. \tag{22}$$

A pseudo-code implementation of the above algorithm is given in Fig. 4.

```
BEGIN PATH PLANNING ALGORITHM
{
    i = 0;
    x_i ⟵ x_0;
    {L_Visited} = {∅};
    (while ‖x_f − x_i‖ > ε)
    {
      compute rm(x, i) for all x ∈ W;
      construct C_d(i);
      construct G(i) = (E(i), V(i));
      (if L_Visited is nonempty)
        for v ∈ V(i)
          x_v(i) = cell_G(i)(v);
          if x_v(i) ∈ L_Visited
            extract v from V(i);
            for all u adjacent to v
              remove (v, u) from E(i);
          end if;
      end if;
      v_1^i ⟵ node_G(i)(x_0);
      v_f^i ⟵ node_G(i)(x_f);
      P(i) ⟵ Dijkstra(v_1^i, v_f^i, V(i), E(i));
      if P(i) = {∅}
        report FAILURE;
        break;
      x_i(1) = cell_G(i)(v_1^i);
      x_i(2) = cell_G(i)(v_2^i);
      {L_Visited} ⟵ {L_Visited} ⨁ {x_i(1), x_i(2)};
      x_{i+1} ⟵ x_i(2);
      i ⟵ (i + 1);
      x_0 ⟵ x_i(2);
    }
}
END PATH PLANNING ALGORITHM
```

Fig. 4. Pseudo-code implementation of proposed multiresolution path planning scheme.

Several improvements and refinements of the previous baseline path-planning algorithm are possible. First, for dynamically changing environments the use of D^* in lieu of Dijkstra's algorithm should speed up the calculation of the shortest path. Second, we can postpone the calculation of the time consuming path search over $\mathcal{G}(t_i)$ until the agent visits node v_{1+r}^i where $r > 1$. This is especially true if the node v_{1+r}^i corresponds to the finest resolution scale in the cell decomposition at time t_i. By construction, all nodes

$v_1^i, v_2^i, \ldots, v_{r-1}^i$ in the path $\mathcal{P}(t_i)$ also represent cells in the finest resolution and hence the path from v_1^i to v_r^i is composed only of free cells.

VI. Simulation Results

In this section we present simulation results of the proposed algorithm for two non-trivial scenarios. In both cases, the environment is assumed to be square of dimension 512×512 units. Hence $N = 9$ is the finest resolution possible. For simplicity, for both scenarios only two levels of resolution have been chosen to represent the environment. Inside an area of 100×100 unit cells we employ a high resolution approximation and outside this area we employ a low resolution approximation of $\mathcal{W}$.

In the first scenario, the environment $\mathcal{W}$ is an actual topographic (elevation) map of a certain US state with fractal-like characteristics, shown in Fig. 5. The initial and final positions of the agent are also shown in this figure. The objective is for the agent (e.g., a UAV) to follow a path from A to B while flying as low as possible, and below a certain elevation threshold. Areas with bright colors in Fig. 5 correspond to areas of low risk (elevation in this case) and darker colors correspond to areas of high risk (elevation in this case) that should be avoided. Solving the path-planning problem on-line at this resolution is computationally prohibitive.

The results from the multiresolution path-planning algorithm using a fine resolution level $J_{\max} = 5$, and a low resolution at level $J_{\min} = 3$ are shown in Fig. 6. Specifically, Fig. 6 shows the evolution of the path at different time steps as the agent moves to the final destination. Figure 6(a) shows the agent's position at time step $t = t_{15}$ along with the best proposed path to the final destination at that time. Similarly, Fig. 6(b) shows the agent's position at time step $t = t_{50}$ along with the best proposed path to the final destination at that time. As seen in Fig. 6(c), the actual path followed by the agent differs significantly from the one predicted in either Figs. 6(a) or 6(b). This is due to the fact that at time t_{15} and t_{50} the agent does not have complete information outside the high resolution zone, and the predicted path actually penetrates the obstacle space $\mathcal{O}$. At time t_{50}, for example, the agent – being far from any obstacle – fails to anticipate the upcoming collision. As the agent gets closer to the obstacle however, and new information is gathered, the existence of the obstacle forces the agent to redirect its path. The agent reaches the final destination $\times_f$ in a collision free manner, as seen in Fig. 6(c). The actual path followed lies inside areas with a low elevation level, which verifies the optimal nature of the path.

In the previous scenario the cost to be minimized along the path is derived solely from the risk measure shown. When the environment is very fragmented, this cost may result in excessively long, meandering paths. To avoid this problem for a cluttered environment, as the one shown in Fig. 7, we may add an additional term that also penalizes the total length of the path. This forces the agent to follow shorter paths in the Euclidean metric. Obstacles in Fig. 7 are denoted in red. The final path obtained using the proposed multiresolution path-planning algorithm is also shown in the same figure.

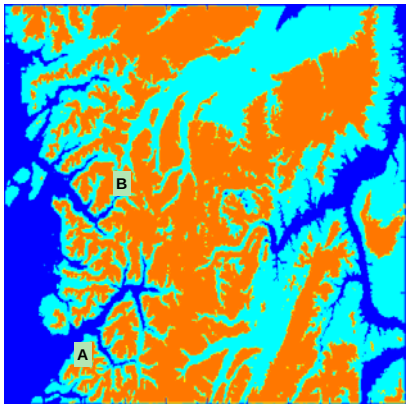


Fig. 5. Plot of risk measure (elevation) for the whole configuration space using a 512×512 unit cell resolution. The blue color corresponds to areas of obstacles. The initial configuration of the agent is denoted by A and the desired final configuration is denoted by B.

Conclusions

In this paper we have proposed a new hierarchical path planning scheme for navigating an autonomous agent inside an environment full of obstacles. The algorithm computes at each step a multiresolution representation of the environment using the wavelet transform. The idea is to use a higher resolution close to the agent where is needed most, and a coarser resolution at large distances from the current location of the agent. This is motivated by the natural observation that for on-line implementations it is not prudent from a computational point of view to compute a solution with great accuracy over large ranges of over a very long time horizon. The algorithm is scalable and can be tailored to the available computational resources of the agent. Several extensions of the baseline methodology presented here are possible, which will be addressed in the future.

Acknowledgement: This work has been supported in part by NSF (award no. CMS-0510259) and ARO (award no. W911NF-05-1-0331). The second author also acknowledges support from the A. Onassis Public Benefit Foundation.

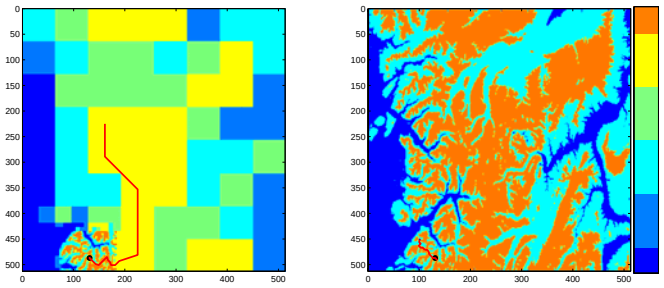
References

[1] I. Daubechies and W. Sweldens, "Factoring wavelets transforms into lifting steps," *Journal of Fourier Analysis and Applications*, vol. 4, no. 3, 1998.

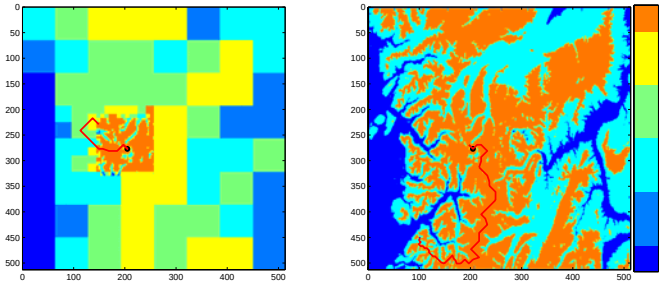
[2] T. Lozano-Perez and M. A. Wesley, "Automatic planning for planning collision-free paths among polyhedral obstacles," in *IEEE Trans. Syst., Man, Cybern.*, vol. 11, pp. 681–698, 1981.

[3] R. A. Brooks and T. Lozano-Perez, "A subdivision algorithm on configuration space for findpath with rotation," in *IEEE Trans. Syst., Man, Cybern.*, vol. 15, pp. 224–233, 1985.

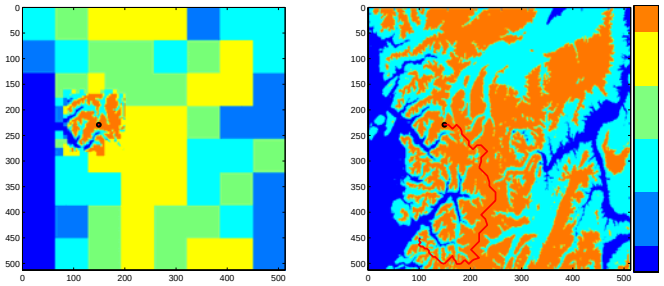
[4] D. Zhu and J. Latombe, "New heuristic algorithms for efficient hierarchical path planning," in *IEEE Transactions on Robotics and Automation*, vol. 7, pp. 9–20, 1991.



(a) $t = t_{15}$



(b) $t = t_{50}$



(c) $t = t_f$

Fig. 6. Path evolution and replanning. Figures on the left show the currently optimal projected path from Dijkstra's algorithm based on the available multiresolution approximation of the environment at different time steps. Figures on the right show the actual path followed by the agent.

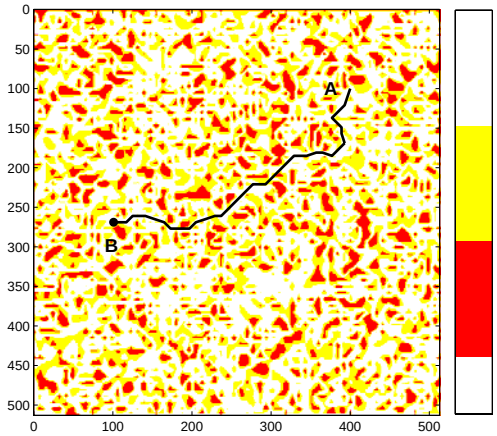


Fig. 7. Final path for the second scenario. For such highly fragmented environments it is advisable to also include a penalty on the Euclidean distance between successive nodes of the path.

[5] H. Noborio, T. Naniwa, and S. Arimoto, "A quadtree-based path-planning algorithm for a mobile robot," *Journal of Robotic Systems*, vol. 7, no. 44, pp. 555–574, 1990.

[6] D. Godbole, T. Samad, and V. Gopal, "Active multi-model control for dynamic maneuver optimization of unmanned air vehicles," in *Proceedings of of the IEEE International Conference of Robotics and Automation*, (San Francisco, CA), pp. 1257–1262, 2000.

[7] D. K. Pai and L. M. Reissell, "Multiresolution rough terrain motion planning," in *IEEE Transactions on Robotics and Automation*, vol. 14, pp. 19–33, 1998.

[8] S. Kambhampati and L. S. Davis, "Multiresolution path planning for mobile robots planning for mobile robots," *IEEE Journal of Robotics and Automation*, pp. 135–145, 1986.

[9] S. Behnke, "Local multiresolution path planning," *Lecture Notes in Computer Science*, vol. 3020, pp. 332–343, 2004.

[10] J. C. Latombe, *Robot Motion Planning*. Boston, MA: Kluwer Academic Publishers, 1991.

[11] S. M. Lavalle, *Planning Algorithms*. New York, NY: Cambridge University Press, 2006.

[12] D. Bertsekas, *Dynamic Programming and Optimal Control*, vol. 1. Cambridge, MA: Athena Scientific, 1995.

[13] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. McGraw Hill and MIT Press, second edition ed., 2001.

[14] C. S. Burrus, R. A. Gopinath, and H. Guo, *Introduction to Wavelets and Wavelet Transforms*. New Jersey, NJ: Prentice Hall, 1998.

[15] S. G. Mallat, "A theory for multiresolution signal decomposition, the wavelet representation," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 2, 1989.

[16] A. Cohen, *Numerical Analysis of Wavelet Methods*, vol. 32. Amsterdam: Elsever Science, 2003.

[17] D. F. Walnut, *An Introduction to Wavelet Analysis*, vol. 32. Boston: Birkhauser, 2003.