Multiresolution Rough Terrain Motion Planning*

Dinesh K. Pai and L.-M. Reissell
Department of Computer Science
University of British Columbia
Vancouver, B.C. V6T 1Z4
Canada
e-mail: {pai|reissell}@cs.ubc.ca

October, 1994

Abstract

We describe a new approach to the problem of motion planning for mobile robots on natural, nonhomogenous terrain. Our approach computes a multiresolution representation of the terrain using wavelets, and hierarchically plans the path through sections which are well approximated on coarser levels and relatively smooth. Unlike most methods, the hierarchical approximation errors are used explicitly in a cost function to distinguish preferred terrain sections. The error is computed using the corresponding wavelet coefficients. The path planning algorithm uses a new non-scalar path cost measure based on the sorted terrain costs along the path. This measure can be incorporated into standard global path search algorithms and yields intuitively good paths. Additional constraints for specific robots can be integrated into this approach for efficient hierarchical motion planning on rough terrain. We present the algorithms and experimental results for real terrain data.

*Supported in part by NSERC, BC Advanced Systems Institute, and IRIS NCE. The authors would like to thank R. J. Woodham, J. J. Little, and the UBC Remote Sensing Council for the St. Mary Lake terrain data, and A. K. Mackworth, D. G. Lowe, RJW and JJL for discussions.

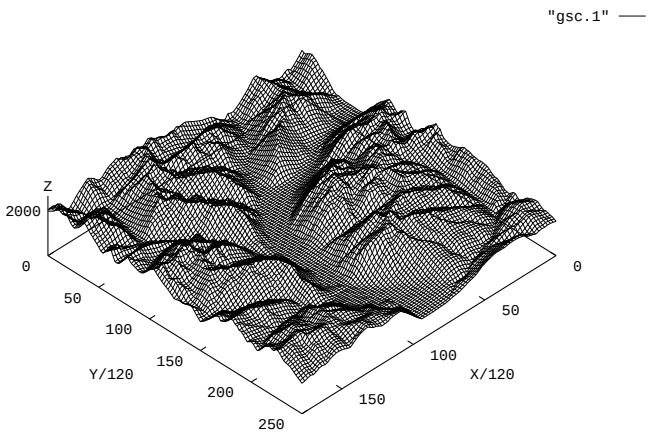


Figure 1: Digital Elevation Map of St. Mary Lake Region

1 Introduction

We consider motion planning for mobile robots on rough terrain, i.e. we wish to navigate a robot such as the Platonic Beast legged robot developed in our laboratory [24] to a prescribed goal, without violating terrain dependent constraints. We assume that the terrain is given by a digital elevation map, $z = f(u, v)$, which has been sampled on a uniform grid of points. Such data is obtained from geographic data bases, satellite imagery, or range finding using lasers [2]. Figure 1 depicts an example of such terrain.

We propose a general multiresolution method which involves four parts: terrain preprocessing, the optional inclusion of terrain-dependent forbidden regions ("obstacles"), path finding, and path refinement. The terrain is first decomposed into its wavelet coefficients, and these are used to obtain a multilevel terrain cost function described below. A suitable level of the multiresolution is chosen to initiate the algorithm; possible additional terrain-dependent obstacles are computed, and an optimal coarse path is found at that level. This computation is repeated hierarchically, moving towards higher resolution levels, and using information from the lower resolution path to restrict the search. Thus only a small subset of the terrain has to be searched at higher resolutions, resulting in a fast algorithm suitable for real-time, and "anytime" planning.

A key feature of our algorithm which makes it work well in practice is that the approximation error in the low resolution representation is explicitly taken into account in the cost of paths. A general shortcoming of some hierarchical methods is that they cannot be guaranteed to find a globally optimal path since some information is ignored at low resolution

levels. In our method, low resolution paths are encouraged to traverse well approximated regions, thereby increasing the likelihood that the traversed terrain "looks the same" at higher resolution. The well approximated regions consist of areas where the terrain is smooth (has small variation on all scales). The approximation error (called terrain roughness here) is efficiently estimated using the wavelet coefficients of the multiresolution.

Another important feature is a new non-scalar path cost measure based on sorted terrain costs along the path. We show that this measure is well suited for rough terrain motion planning, makes it possible to use standard global path search algorithms, and yields intuitively good paths.

We have implemented the algorithm and tested it on real terrain data. We plan to use the algorithms for rough terrain navigation of the "Platonic Beast" spherically symmetric legged robot [24] designed for rough terrain, and other mobile robots in our laboratory [3].

The paper is organized as follows: § 2 describes how to efficiently construct roughness measures for terrain using the wavelet decomposition – small values of these measures imply that the terrain is well approximated at lower resolution. In § 3 the roughness measures are used to design a new, non-scalar measure on paths. The path measure is used to search for good paths at each level, i.e. paths that go through regions with small approximation error. Details of the algorithms are given in § 4. We have implemented our algorithms and in § 5 we present the experimental results of using our method with real terrain data from the St. Mary Lake region of British Columbia.

1.1 Related work

Robot motion planning methods have been investigated extensively (see [17] for a survey). Most of this work considers the robot environment as consisting of obstacles, and treats these at a single resolution. Some authors have approached the collision free motion planning problem using hierarchical techniques, for instance, [5, 35]. Multiresolution collision free motion planning for mobile robots using quadtrees has been considered by [15], [23]. Most of these techniques rely on the labeling of terrain cells on any level as "free" or "mixed"; there are no measures to differentiate between two mixed cells on a level. In our method, there are no obstacles as such and since we wish to take the approximation error into account, all cells are "mixed". Paths prefer coarse level cells which approximate the original data better. Somewhat closer to our strategy, [25] uses a filtering method to obtain coarse level information from the finer levels in the context of obstacle avoidance using dynamic programming.

There has been recent interest in planning for robots capable of navigating rough terrain (e.g. [2, 12, 18, 13]). The work of Simeon and Dacre-Wright ([30, 9]) performs all-terrain path planning using placement constraints for a robot with n wheels and spring suspension. The feasible and stable placements of the robot define the "free space" for the robot in a three-dimensional slice of the configuration space. The free space is subdivided using an octree decomposition and searched using $A*$. In [29], a point robot with tip-over constraints

is used to find a time optimal path through rough terrain. The JPL robot [12] uses a gradient method for path planning. For the RAMI robot [13] a multi-layer controller plans paths using a heuristic based on ground filtering. A multi-layer strategy is also used in ARCANE [28].

Our method relies on the type of multiresolution filtering techniques common in computer vision (e.g.,[16, 19, 22, 27, 33, 32, 34]). Wavelets are an outgrowth of both pyramid methods (e.g. [6]) and the time-frequency localization methods of signal processing – discrete multiresolution wavelet approximation was introduced by Mallat and Meyer for computer vision applications ([20]). Wavelets concentrate on the properties of the approximation error, and a large number of theoretical results are available. They can now be considered an important mathematical tool with applications areas which include edge detection, signal compression and approximation theory, and many others ([20],[21], [11]).

2 Measures for terrain roughness

A key criterion for a reasonable path in a multiresolution application is that the path traverses sections of terrain that are well approximated. We call such areas *smooth*[1] and poorly approximated regions as *rough*. To obtain a hierarchical measure for terrain roughness, will use a wavelet decomposition of the 2-dimensional terrain data. By providing successive smoother approximations of the original data, and a handle on the approximation error, the decomposition yields natural roughness measures.

Is there a reason to use wavelets rather than other methods of hierarchical data smoothing, such as pyramid filtering? The methods are obviously related and both methods provide successively smoothed approximations of the original data. However, unlike the usual pyramid algorithms, wavelet filtering emphasizes the approximation error, and a precise measure of the order of approximation is given by the number of vanishing moments of the wavelet. Wavelets fit into a well studied mathematical framework, and have been shown to have good, polynomial-like approximation properties, which can be extended to apply to different norms ([11]). Wavelet methods therefore have an advantage: the degree of "smoothing" performed by wavelet filtering is well quantified. In addition, the wavelet decomposition provides a compact representation from which to calculate the roughness measures.

In this section we first briefly review the notation used and then discuss wavelet based measures of terrain roughness.

2.1 Wavelet notation

We consider the usual discrete "multiresolution" construction of wavelets obtained by dilations by powers of 2 and integer translations from a basic wavelet ψ:

[1] This corresponds to the C^r notion of smoothness with certain assumptions on the wavelet.

$$\psi_{i,j} = 2^{-i/2}\psi(2^{-i}x - j), \quad i,j \in \mathbf{Z}.$$

We will use the interpolating *pseudocoiflets* of [26] - these belong to the class of biorthogonal wavelets ([8]), which generalize the standard orthonormal construction: the wavelets ψ and its *dual* wavelet, $\tilde{\psi}$), form dual bases for L^2. The wavelets are constructed together with a *scaling function* ϕ and the dual $\tilde{\phi}$. The translated and dilated scaling functions give the bases for a multiresolution hierarchy.

The wavelet terminology we use is standard: the *wavelet decomposition* of a function f is the representation of f in the wavelet basis. The *wavelet coefficients* are the coefficients $w_{ij}(f) = < f, \tilde{\psi_{i,j}} >$ of this decomposition. We also use the *scaling coefficients* $s_{ij}(f)$ of f at level i: $s_{ij}(f) =< f, \tilde{\phi_{i,j}} >$.

In practice, the wavelet and scaling coefficients are obtained via two pairs of filters, the *scaling filters* H and $\tilde{H}$, and the *wavelet filters* G and $\tilde{G}$, using Mallat's tree algorithm:

$$\begin{array}{ccccccc} & \tilde{H} & & \tilde{H} & & \tilde{H} & \\ \mathbf{s} & \longrightarrow & \mathbf{s}_1 & \longrightarrow & \mathbf{s}_2 & \longrightarrow & \ldots \\ & \tilde{G} & & \tilde{G} & & \tilde{G} & \\ & \searrow & & \searrow & & \searrow & \\ & & \mathbf{w}_1 & & \mathbf{w}_2 & & \ldots \end{array} \tag{1}$$

Reconstruction is performed in the opposite direction with primal filters. We note that the number of wavelet and scaling coefficients drops by a factor of 2 at each level in the decomposition, and by a factor of 4 in 2 dimensions.

2.2 Roughness measures using wavelet coefficients

We begin with a hierarchical representation of the terrain data. On a given level of approximation, we wish to choose paths which remain on relatively well approximated (smooth) sections of the original terrain whenever possible. We propose the following heuristic to apply to an admissible path on a given level; in the rest of the section we make it more precise.

> **Heuristic (version 1): Do not cross areas with large errors between the approximation and the original data.**

To apply this heuristic, along with the a lower resolution approximation, we need to estimate the approximation error. This can be done efficiently using the wavelet decomposition.

First, the scaling coefficients of a function f give a smoothed approximation f_i of f at each resolution level i:

$$f_i = 2^{-i/2} \sum_j s_{ij} \phi_{ij},$$

where $\phi_{ij}(x) = 2^{-i/2}\phi(2^{-i}x - j)$.

The wavelet coefficients provide the error of this approximation – the L^2-error $E_i = \| f - f_i \|$ of the level i approximation to f is equivalent [2] to the norm of the "unused" higher level wavelet coefficients:

$$E_i \approx \left(\sum_{m \leq i} \left(\sum_{j=-\infty}^{\infty} w_{mj}^2 \right) \right)^{1/2} . \tag{2}$$

Precise results connecting the local and global smoothness of a function and the size of its wavelet coefficients are summarized for instance in [10], [21]. These results rely on the fact that the approximation error can be expressed in terms of the wavelet coefficients (Eq. 2). Here we will state the basic L^2-approximation property of wavelets, which connects global approximation properties with the number of vanishing moments of the wavelet (see [31]):

Approximation property: For functions f with at least C^N-continuity, the wavelet error of f decays as 2^{Ni}, as $i \to -\infty$, provided that the wavelet has N vanishing moments:

$$E_i \leq K 2^{Ni}. \tag{3}$$

Informally, some of the consequences of wavelet approximation can be restated as follows, combining equation 2 with 3:

- For smooth areas of the terrain, the approximation error of Equation 2 decays rapidly towards finer scales.
- Wavelets filter noise: for noisy data, high resolution coefficients are uniformly large, but lower resolution coefficients take on the characteristics of the underlying object.
- The number of vanishing moments of the wavelet controls the rate at which the data is smoothed – areas of high variation and discontinuity will stick out more prominently in the wavelet coefficients if the number of vanishing moments is larger.

Therefore, at a given level, the heuristic for planning paths which remain on the better approximated, "smoother" sections of the actual terrain data can now be restated in terms of the wavelet coefficients:

[2] Equivalence means that the norm E_i is bounded from above and below by the wavelet coefficient norm, multiplied by fixed constants. When the wavelets are orthogonal, we have equality between the two norms.

Heuristic (version 2): Do not cross areas with large higher resolution wavelet coefficients.

2.3 Approximate roughness measures

With the use of wavelets with compact support, the error E_i (equation 2), and corresponding local errors, can be computed in a finite number of steps. For faster computations, the local version of the error E_i has to be approximated. We choose the approximation to be the truncated L^2- error, $E^*(A,l)$ where $E^*(A,l)^2$ is given by the sum of the squared wavelet coefficients corresponding to the area A [3]. This quantity represents the local error between the low resolution approximation given by the scaling functions at a level l and the original data f.

We construct the approximation $E^*(A,l)^2$ as follows. For a cell A on level l, corresponding to an interval $[K,L]$ of the original data,

$$(E'^*(A,l))^2 = \sum_{m \le l} \left(\sum_{j \in I^*(A,m)} w_{mj}^2 \right), \tag{4}$$

where

$$I^*(A,m) = \{k \in \mathbf{Z} : k \in [I/2^m, J/2^m)\}.$$

We then average the error contributions of neighboring cells:

$$E^*(A,l)^2 = \sum_{A' \in nbhd(A)} E'^*(A',l)^2 / \#nbhd(A), \tag{5}$$

where $nbhd(A)$ consists of A and the cells in the immediate neighborhood of A on level l. The averaging evens out the contribution of wavelet coefficients to neighboring intervals and creates an extra "buffer zone" around an area of high variation or discontinuity. For 2-d data, the definitions are analogous.

Intuitively, the cost function captures the discrepancy resulting from using the approximation of the terrain at a lower resolution instead of with the original data. Areas with small E^* are more likely to look the same at finer resolutions than areas with large E^*.

3 Measures for paths on rough terrain

Using the roughness measure of § 2, we now construct appropriate measures for gauging the quality of candidate paths at a given level of approximation. After considering some

[3] Use of these approximations requires wavelets with good spatial localization, for instance, the wavelets in [26].

candidates and their drawbacks, we propose a new non-scalar measure, D_{smax}, suitable for rough terrain.

First, a small generalization: we will assume that we have a non-negative *terrain cost* C_l defined for each cell of data. In this paper we will usually assume that the terrain cost is the roughness measure described in § 2.3, i.e. for each level l the terrain cost of cell (i, j) is $C_l(i, j) = E^*(A, l)$, where A is the cell determined by the index pair (i, j). However, in general $E^*(A, l)$ can be combined with other non-negative measures of terrain cost, such as terrain slope, coefficient of friction, cost of crossing rivers and other non-geometric features of the terrain, etc.

A path p through a discrete terrain is a sequence $x_0 x_1 x_2 \ldots x_l$ of connected cells from a starting cell x_0 to an ending cell x_l such that each cell on the path is free. Cell connectivity is usually defined in one of two ways: 4-connectivity (two cells are connected if they share an edge) or 8-connectivity (two cells are connected if they share a vertex). We will consider, without loss of generality, 4-connected cells.

With a terrain cost C, we can now define path costs D. Some obvious candidates are:

- Total Cost D_{total}:
$$D_{\text{total}}(p) = \sum_{(i,j)\in p} C(i, j) \tag{6}$$
- Worst Cost D_{max}:
$$D_{\text{max}}(p) = \max_{(i,j)\in p} C(i, j) \tag{7}$$
- Average Cost D_{avg}:
$$D_{\text{avg}}(p) = D_{\text{total}}/\text{length}(p) \tag{8}$$

None of these costs in completely adequate for our application. The total cost D_{total} is not suitable since D_{total} implicitly involves path length and hence will choose paths that pass through regions of high cost with shorter path length. Using D_{avg}, the average cost, creates a new problem: paths wander aimlessly in regions of low cost merely to lower the average.

Using D_{max} is better from the point of view of safety since the selected path will minimize the cost of the worst region the path traverses. However, D_{max} suffers from being unable to discriminate among large sets of paths – a phenomenon we can call "the ravine effect." If there is a small region of uniformly high cost (the "ravine") that separates the start from the goal, all paths from the start to the goal have the same cost D_{max}. To take a concrete example, suppose the terrain contains a river that separates the robot from the goal (see Figure 2). Suppose, further that the river can be crossed by the robot at any point, but at a high, fixed cost. Then using D_{max} *all* paths which cross the river will have the same path cost.

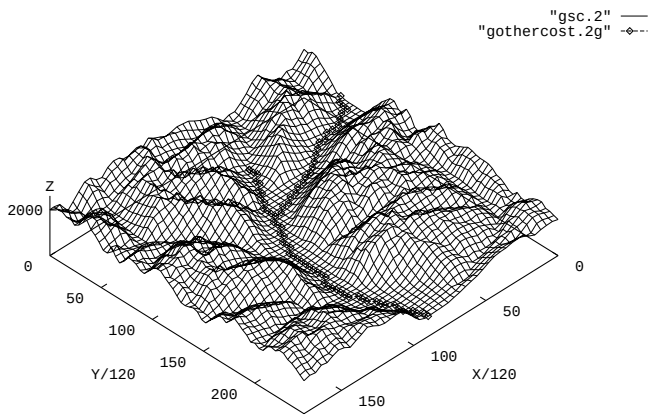


Figure 2: A river can separate the terrain with a small region of high terrain cost

We propose a new non-scalar measure called D_{smax} which addresses this problem by generalizing $D_{\max}$. For a path $p = x_0 x_1 x_2 \ldots x_l$, we define

$$\begin{aligned} D_{\mathrm{smax}}(p) \quad &= \quad \text{list of costs of } x_i \in p, i \neq 0, \\ &\qquad \text{sorted in non-increasing order.} \end{aligned} \tag{9}$$

D_{smax} is not a scalar, but has a natural, lexicographic ordering $\leq_l$. Recall that lexicographic order [1] means $\{u_1 u_2 \ldots u_l\} \leq_l \{v_1 v_2 \ldots v_m\}$ when either (1) there exists an integer j such that $u_j < v_j$ and for all $i < j, u_i = v_i$, or (2) $l \leq m$ and $u_i = v_i$ for $1 \leq i \leq p$.

This induces the following order relation on paths:

$$p_1 \leq p_2 \iff D_{\mathrm{smax}}(p_1) \leq_l D_{\mathrm{smax}}(p_2). \tag{10}$$

This is a linear order.

We define an "addition" of the costs of two paths p_1 and p_2 as

$$\begin{aligned} &D_{\mathrm{smax}}(p_1) \oplus D_{\mathrm{smax}}(p_2) = \\ &\quad \text{sorted merge}(D_{\mathrm{smax}}(p_1), D_{\mathrm{smax}}(p_2)). \end{aligned} \tag{11}$$

D_{smax} thus provides finer discrimination than $D_{\max}$ and is immune to the ravine effect. This is illustrated in § 5, figure 15. In addition, the ordering properties are sufficient to allow it to be used in standard path search algorithms (see § 4.3).

4 Algorithms

The proposed multiresolution solution to motion planning on rough terrain consists of four parts: terrain preprocessing, computation of terrain-dependent obstacles, path finding, and path refinement. We now describe each of these parts in detail.

4.1 Terrain preprocessing

We perform a 2-dimensional wavelet decomposition on the sampled data, giving a sequence of scaling coefficients (s_{lj}) and wavelet coefficients (w_{lj}), for each level l. The scaling coefficients give rise to approximating terrain surfaces $z = f_l(u,v) = 2^{-l/2} \sum s_{lj} \phi_{lj}$.

The underlying wavelet used can be an orthonormal wavelet or a biorthogonal wavelet pair obtained from multiresolution. The 2-dimensional wavelets are obtained from the standard multiresolution tensor product construction ([10]).[4] In this construction, the wavelet coefficient sequences consist in fact of three sequences (w^H_{lj}), (w^V_{lj}), and (w^D_{lj}). (The superscripts indicate the wavelet's propensity to distinguish between horizontal, vertical, and diagonal features.) The wavelet coefficients are obtained by a 2-dimensional version of Mallat's tree algorithm ([20]).

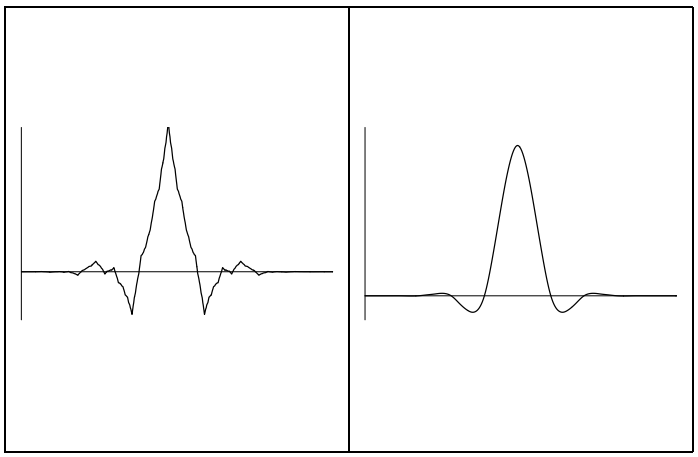
As scaling filters, we can choose filters which represent "well behaved" underlying functions, such as B-splines ([8]), or the interpolating scaling functions of [26]. The wavelets should be chosen to be concentrated in absolute value around a given point, making the use of the above approximate error measure effective. The pseudocoiflets P_4 of [26], for instance, have this property. In addition, the interpolation property for the scaling functions corresponding to these wavelets guarantees that the scaling coefficients form a sampling of the actual approximation surface; thus multiresolution planning can be performed reliably on the *scaling coefficient* surface, and not the approximation surface, which, although smoother than the original, consists of just as many points.

The specific wavelets we use are the pseudocoiflets P_4 [26], which have $2N = 4$ vanishing moments and are shown in Figure 3. The scaling filters for these wavelets are also shown.

4.2 Terrain Obstacle Computation

In general, the robot's motion is subject to several constraints depending on the specific robot and application. These can include terrain slope constraints (there is a limit on the slope of terrain that can be traversed by the robot), contact constraints (a suitable subset of feet or wheels should be in contact with the terrain), static stability constraints (the vertical projection of the robot's center of mass is inside its support polygon), and collision avoidance constraints (parts of the robot do not intersect the terrain or other parts of the robots). See also [13, 29, 30].

[4] Other constructions for 2-dimensional wavelets are possible.



(a)

$\sqrt{2}\tilde{\phi}$	$\frac{1}{\sqrt{2}}\phi$
-0.00390625	
0.0	
0.0703125	
-0.0625	-0.031250
-0.24609375	0.0
0.56250	0.281250
1.359375	0.500000
0.56250	0.281250
-0.24609375	0.0
-0.0625	-0.031250
0.0703125	
0.0	
-0.00390625	

(b)

Figure 3: Analyzing and reconstructing pseudocoiflets. (a) The case $\tilde{\phi}_{N=2}, \phi_{N=2}$ (b) Filter coefficients for $N = 2$. The left column represents the analyzing filter $\tilde{\phi}$, and the right column, the reconstructing filter ϕ

The planner must find a path from the initial position to the goal position, without violating any of the constraints imposed. We consider holonomic constraints on the terrain, i.e. constraints that can be expressed as a function of the position of the robot on the terrain. Regions that do not satisfy the constraints are considered to be virtual *obstacles*, i.e., forbidden regions of the terrain. The areas of the terrain where all constraints are satisfied are called *free*. Static stability constraints and terrain slope constraints, for example, can be formulated in this way. These terrain-dependent constraints can also be computed efficiently using multiresolution.

Since this depends on the specific application, we do not discuss it in detail here. However we do present a concrete example of a terrain-dependent obstacle below in § 5. We consider the case of a robot which is small relative to the resolution of the terrain data and hence can be considered as a point for the sake of motion planning. This is reasonable, for instance, for navigation of human-scale mobile vehicles with terrain data obtained from satellite imaging.

We impose obstacles due to limitations on maximum terrain slope. The slope is computed from the gradient obtained by applying the 3×3 Sobel gradient operators to the scaling coefficients[5]. Regions of slope greater the maximum slope are marked as obstacles. Figure 4 shows the slope obstacles for a maximum slope of 0.6 (about 31 degrees) and 0.4 (about 22 degrees) at level 2. In § 5 we show the effect this has on paths.

4.3 Path finding

The basic algorithm is a variation of Dijkstra's single source shortest path algorithm [1] and is shown in Figure 4.3. It finds a path from a start cell s to a goal cell g or reports that there is no such path. The algorithm assumes that free space cells (i.e. the cells that are not obstacles) are marked FREE during the obstacle computation step and their costs $v.D_{\mathrm{smax}}$ are initialized to $+\infty$[6].

We sketch the proof of correctness of the algorithm here. The key point is that the cost measure D_{smax} under the "addition" operation $\oplus$ behaves like a non-negative number under addition. In particular it is easy to show

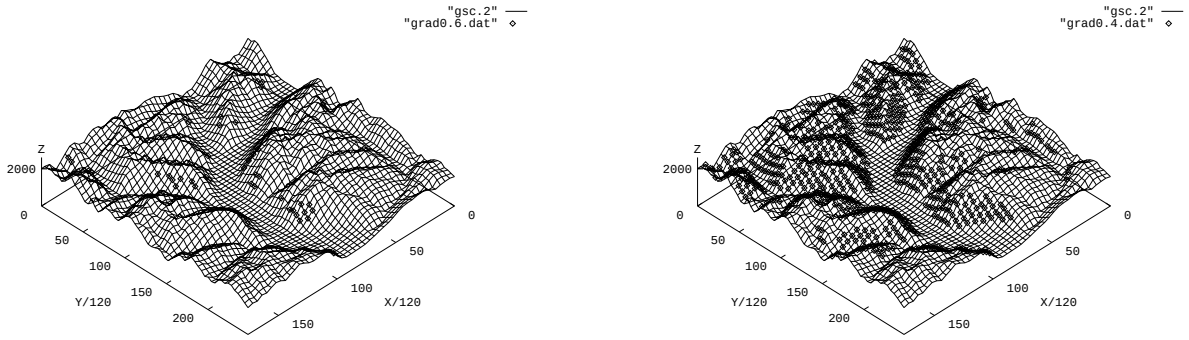
Proposition 1. Let L_i denote a sorted list of costs and L be the singleton list $\{l\}$. Then

$$L_1 \leq L_2 \Rightarrow L_1 \leq L_2 \oplus L.$$

We denote by $u.D_{\mathrm{smax}}$ the cost of a path from the start s to u, and $u.D_{\mathrm{smax}}{}^{v}$ is the cost of any path from v to u. The previous result then implies:

[5]Note that at poorly approximated regions, the computed slope will be poorly approximated as well. The wavelet coefficients can be used to estimate bounds on the derivatives as well, but we do not deal with this issue here.

[6]i.e. a sufficiently large number, e.g. the maximum terrain cost computed on the data



slope $< 0.6 (\approx 31°)$ slope $< 0.4 (\approx 22°)$

Figure 4: Obstacles due to maximum slope constraint

Proposition 2. $w.D_{\mathbf{smax}} \leq x.D_{\mathbf{smax}} \Rightarrow w.D_{\mathbf{smax}} \leq x.D_{\mathbf{smax}} \oplus w.D_{\mathbf{smax}}{}^{x}$, for all x and for all path costs $w.D_{\mathbf{smax}}{}^{x}$.

Thus we show that at each iteration of the **while** loop in Figure 4.3, $w.D_{\mathbf{smax}}$ is the cost of the optimal path to w and $v.D_{\mathbf{smax}}$ is the cost of the best path through VISITED cells.

4.4 Path refinement

The hierarchical path planner begins with a suitable initial level l. The path is then planned in the free terrain areas on level l, according to the algorithm of § 4.3.

We then employ a simple path refinement strategy similar to, for example, [16]. First the path p is grown into *channel* P by a given margin m, i.e. if p is a level l path, a cell with index $(u, v) \in P$ if there exists a cell $(i, j) \in p$ and $|(u - i)| \leq m$ and $|(v - j)| \leq m$.

At the next finer level of resolution, $l - 1$, cells which are not refinements of the channel P are marked as obstacles[7]. Other obstacles at level $l - 1$ can also be incorporated in this stage, and the path finding algorithm of § 4.3 is repeated, using terrain cost C_{l-1} on this restricted area.

Growing the path in this way provides a simple way to control the amount of level $l - 1$ data to be searched: m is a small positive integer; if $m = 0$, only the cells corresponding

[7]Only the boundaries of the channel need be marked.

```
PriortyQueue Snext(φ);
Snext.insert(s);
while (Snext ≠ φ ) do
   w := Snext.delete( Snext.min );
   mark w as VISITED;
   for each neighbor v of w do
      if v is FREE and not VISITED then
         if v.D_smax > w.D_smax⊕ { v.TerrainCost } then
            v.D_smax := w.D_smax⊕ { v.TerrainCost };
            v.PreviousCell := w;
            if v ∈ Snext then
               Snext.delete(v);
            end if;
         end if;
         Snext.insert(v);
      end if;
   end for;
end while;
if goal g is marked VISITED then
   read off path by following
   the PreviousCell pointers;
else
   report failure;
end if
```

Figure 5: Path finding algorithm

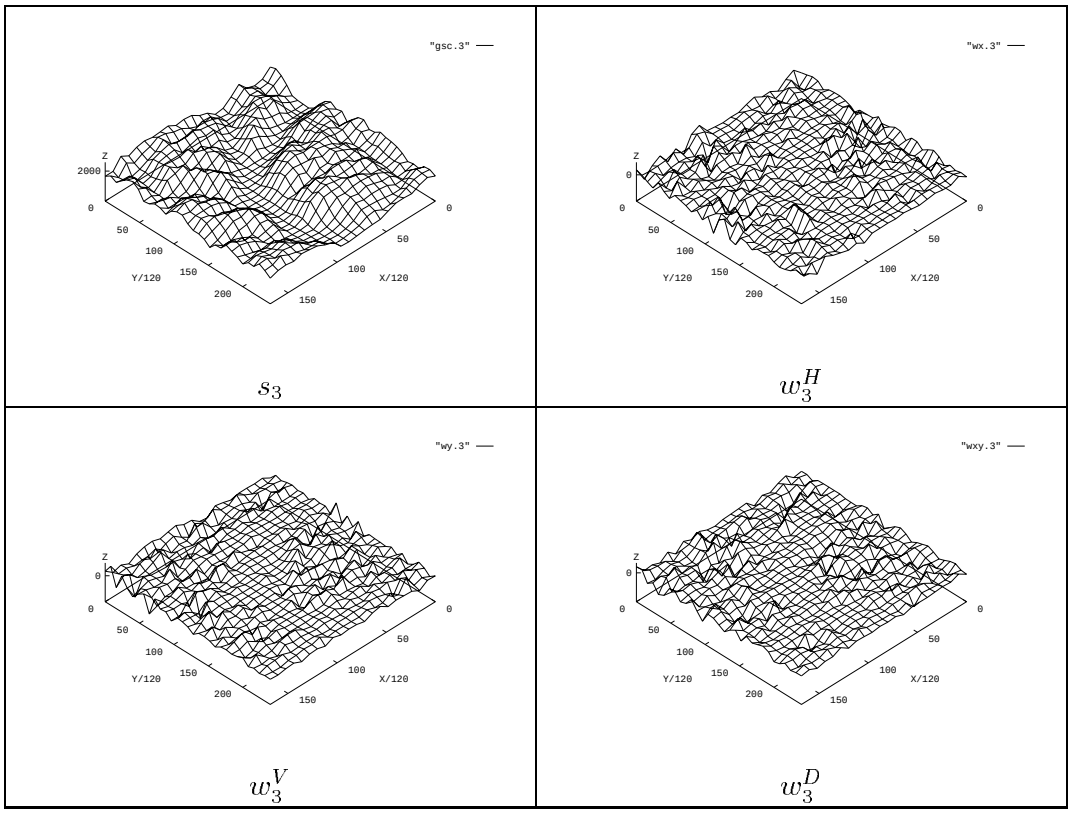


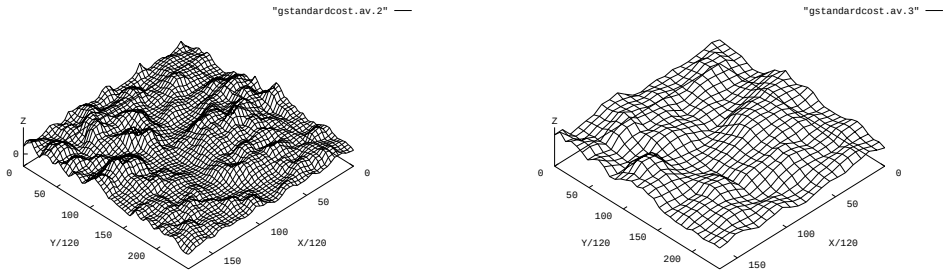
Figure 6: Wavelet decomposition of terrain data; s and w are the scaling and wavelet coefficients, respectively.

to the level l path are searched; as m increases, larger neighborhoods of the level l path are searched.

5 Experimental results

We have implemented a motion planner for rough terrain based on the above algorithms, in `C++`. The planner has been tested on both synthetic and real terrain data. We describe the results for the St. Mary Lake terrain data shown in Figure 1.

In the terrain preprocessing phase, the terrain was decomposed for 4 refinements levels. Figure 6 shows the decomposition at level 3. The surfaces depicted in Figures 10, 9 and 8 are the scaling coefficients of the terrain at levels 2, 3 and 4 respectively. Figure 7 plots the



Level 2 Level 3

Figure 7: Terrain Costs C_l at levels of refinement

terrain cost functions C_l computed from the wavelet coefficients.

In all cases below, the boundary of the data set is marked as an obstacle to keep the robot within the mapped terrain. For illustration, we consider the terrain at approximation levels 2,3 and 4 (higher resolution terrain is difficult to display clearly at the resolution of printed paper). We shall call a path of type $(a \succ b)$ if the path was found by starting at the coarser level b and refined to a level a path.

Figure 8 shows a $(4 \succ 4)$-path found at level 4. Figures 9 and 10 shows the refinement of this path to $(3 \succ 4)$ and $(2 \succ 4)$ respectively. In this example, the path was grown into a channel by margin $m = 3$.

We also computed the $(2 \succ 2)$ path, i.e. found at level 2 using our $D_{\mathbf{smax}}$ cost, but without multiresolution. As hoped for, this turns out to be indentical to the $(2 \succ 4)$ path. This behavior is common but not necessary.

For a different set of end points, Figure 11 shows a $(2 \succ 4)$-path, and the corresponding $(2 \succ 2)$-path is shown in figure 12. Keep in mind that in these examples, for illustration, we are using only the terrain roughness measure as the terrain cost; other costs can be incorporated as indicated in §3. In this case the paths are different, illustrating that the globally optimal level-2 path need not be found. However, see Figure 13, which depicts the $D_{\mathbf{smax}}$ path costs for the $(2 \succ 2)$ and $(2 \succ 4)$ paths. However, we observe that in this case the the suboptimal $(2 \succ 4)$ path is not very bad at all: in fact it is only worse than the optimal $(2 \succ 2)$ path in the thirteenth largest cost along the path and generally passes through lower cost terrain. This is to be expected since the $(2 \succ 4)$ path passes through terrain that is low in cost at all lower resolution levels, and not just at level 2.

We note that the paths stay on the large smooth regions whenever possible. In the above examples, the paths were found that avoid regions of slope greater than 0.4 (about 22 degrees) by marking such regions as obstacles. In Figure 14 the $(2 \succ 4)$ path is found

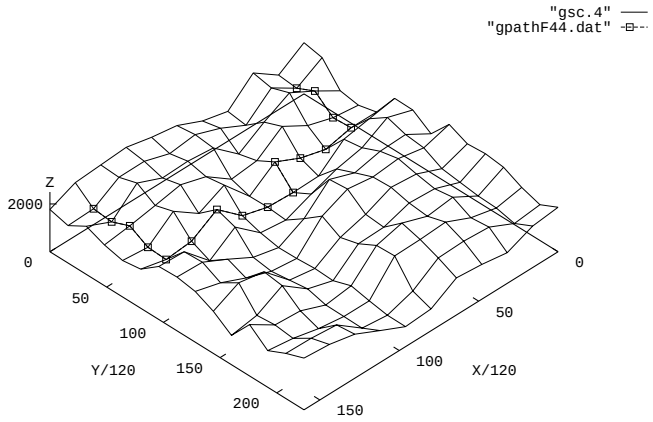


Figure 8: $(4 \succ 4)$ path

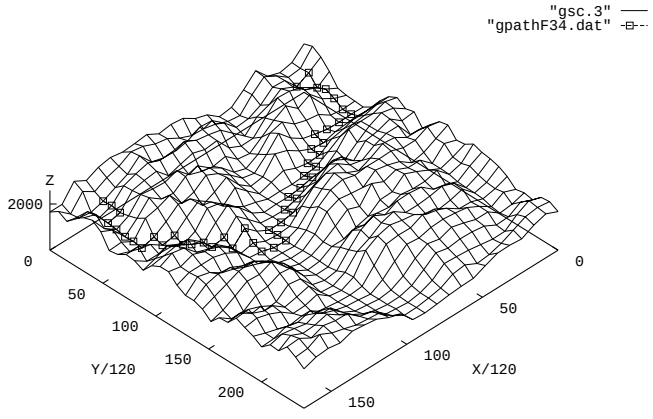


Figure 9: $(3 \succ 4)$ path

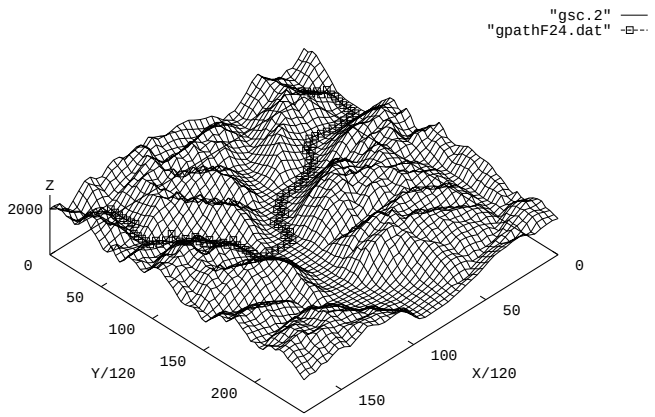


Figure 10: $(2 \succ 4)$ path (the $(2 \succ 2)$ path is same).

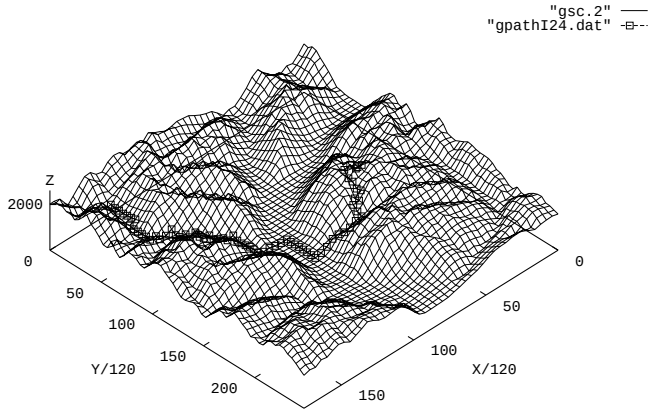


Figure 11: $(2 \succ 4)$ path, with different endpoints

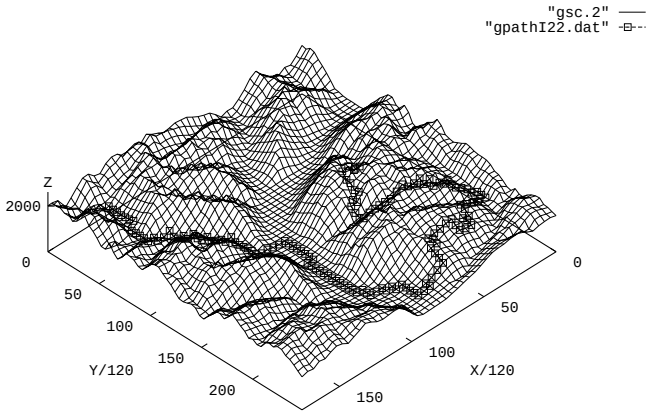


Figure 12: Corresponding $(2 \succ 2)$ path

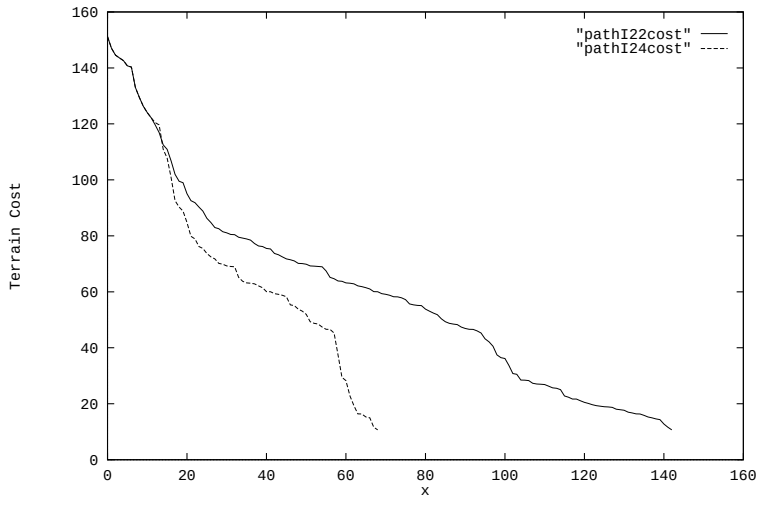


Figure 13: D_{smax} path costs for the $(2 \succ 2)$ (dashed line) and $(2 \succ 4)$ (solid line) paths

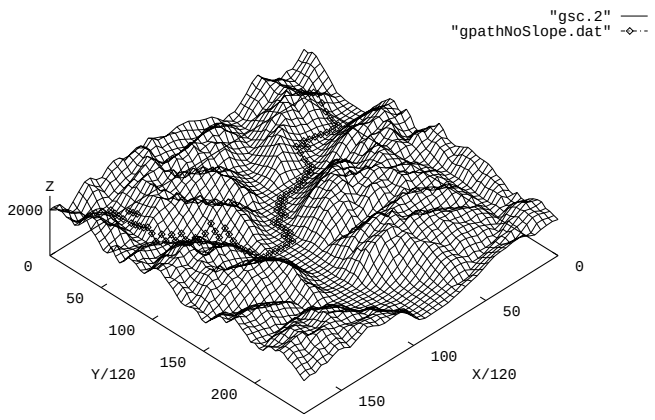


Figure 14: $(2 \succ 4)$ path without slope constraints

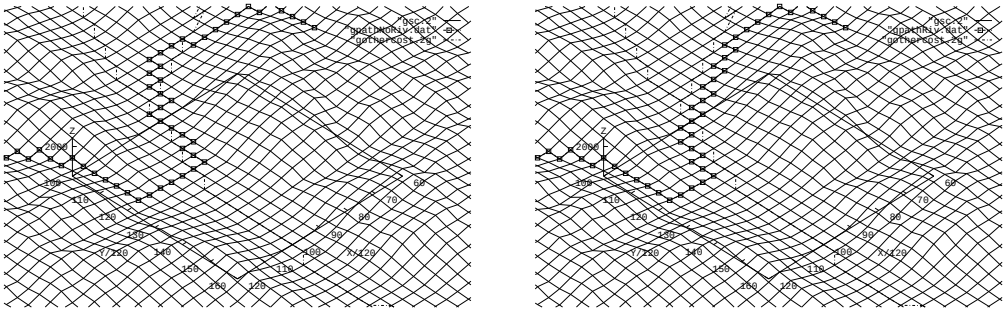
without any additional obstacles (constraints) imposed. Observe that as may be expected, the path prefers smoothest regions even when these are the sides of mountains, and avoids the less steep, albeit high variation (and hence high cost) valleys.

Finally, we illustrate the robustness of our path cost measure $D_{\mathbf{smax}}$ to the "ravine"-effect. Consider the case where a river (of very high cost) crosses the terrain and separates the goal from the robot (see figure 2).

We observe that usually (e.g., the $(2 \succ 4)$ path) the path is identical to the path found without no river present: almost all paths from start to goal have a the same high cost of crossing the river appended to the front of the $D_{\mathbf{smax}}$ cost, and hence the ordinal relationship among paths is unchanged. In some cases, if the original path crosses the river several times, then the new path (see figure 15) is slightly perturbed to reduce the number of crossings to the minimum possible. This is shown clearly in Figure 15. We see $(2 \succ 2)$ paths in a close up of the region around the river fork, both without taking the river into account and with the river cost included.

6 Conclusions

We have developed, implemented and tested a new motion planning algorithm for navigating mobile robots on natural, non-homogeneous terrain. Multiresolution terrain representation is used and leads to a fast algorithm; this is important due to the large size of the terrain data and the global search required. Significantly, multiresolution permits motion planning



$(2 \succ 2)$ path, no river cost $(2 \succ 2)$ path, with river cost

Figure 15: Behavior of paths that cross the river: immunity to the "Ravine effect"

to be *anytime* since a coarse level solution is obtained extremely fast and is successively refined to higher resolution levels.

A key contribution is the use of a "terrain roughness measure" to guide the search for paths at each level of resolution. This brings two benefits: First, the paths are encouraged to traverse well approximated regions, so that the refinement of the traversed terrain is likely to be similar at finer resolutions. Second, the paths are intuitively appealing since they prefer large, smooth sections of the terrain, rather than relying on small features in the data.

We use wavelet multiresolution to compute both the smoothed terrain as well the roughness measure. Wavelets allows the degree of smoothing performed by wavelet filtering to be well quantified; the wavelet coefficients are also used calculate the roughness measure. By using interpolating pseudocoiflets, the multiresolution planning can be performed directly on the *scaling coefficient* surface, rather than on the approximation surface.

Another feature is a new non-scalar path cost measure, $D_{\mathbf{smax}}$, for paths on rough terrain. It is based on terrain costs along the path, sorted in non-increasing order. The measure retains the safety property of the maximum-cost measure, i.e. the selected path always has best value of the worst terrain cost along the path. However, it provides finer discrimination among alternate paths that may have the same worst cost, i.e. it is immune to the "ravine effect."

Finally, our solution method is quite general, consisting of four parts: terrain preprocessing; optional computation of additional, terrain-dependent obstacles; path finding; and path refinement. We believe it is easily extensible to problem-specific constraints. In future work, we plan to incorporate constraints due to foot placement, stability in the presence of frictional contacts, etc., for navigating our Platonic Beast legged robot over natural terrain.

References

[1] A. Aho, J. Hopcroft, and J. Ullman. *The Design and Analysis of Computer Algorithms*, Addison-Wesley, 1974.

[2] J. Bares, M. Hebert, T. Kanade, E. Krotkov, T. Mitchell, R. Simmons, and W. Whittaker. Ambler, an autonomous robot for planetary exploration. *IEEE Computer*, 22(6):18–26, June 1989..

[3] R. Barman, S. Kingdon, J. J. Little, A. K. Mackworth, D. K. Pai, M. Sahota, H. Wilkinson, and Y. Zhang. Dynamo: real-time experiments with multiple mobile robots. In *Proceedings of the IEEE Intelligent Vehicles Conference*, pages 261–266, Tokyo, Japan, July 1993.

[4] J. Barraquand and J.C. Latombe. On non-holonomic mobile robots and optimal maneuvering. *Revue d'Intelligence Artificielle* 3(2), 1989.

[5] R. A. Brooks and T. Lozano-Pérez. A Subdivision Algorithm in Configuration Space for Findpath with Rotation, IJCAI, 1983.

[6] P. Burt and E. Adelson. The Laplacian pyramid as a compact image code. *IEEE Trans. Comm.*, 31, 482–540, 1983.

[7] A. Cohen, I. Daubechies, and P. Vial. Wavelets and fast wavelet transform on the interval. *AT&T Bell Laboratories preprint*, 1992.

[8] A. Cohen, I. Daubechies, and J.C. Feauveau. Biorthogonal bases of compactly supported wavelets. *Comm. Pure and Appl. Math.* 45, pp. 485–560, 1992.

[9] B. Dacre-Wright and T. Simeon. Free space representation for a mobile robot moving on a rough terrain. *Proc. IEEE Int. Conf. on Intelligent Robots and Systems*, pp. 37 – 43, 1993.

[10] I. Daubechies, *Ten Lectures on Wavelets. CBMS–NSF Regional Conference Series in Applied Mathematics* 61, SIAM, 1992.

[11] R.A. DeVore, B. Jawerth, and V. Popov. Compression of wavelet decompositions, American Journal of Mathematics, 114, pp. 737–785, 1992.

[12] E. Gat, M.G. Slack, D. P. Miller, R. J. Firby. Path Planning and Execution Monitoring for a Planetary Rover *Proceedings of the IEEE International Conference on Robotics and Automation,* pp. 20–25, 1990.

[13] M Iagolnitzer, F. Richard, J. F. Samson, and P. Tournassoud. Locomotion of an all-terrain mobile robot. *Proceedings of the IEEE International Conference on Robotics and Automation,* pp. 104–109, 1992.

[14] S. Jaffard, Exposants de Hölder en des points donnés et coéfficients d'ondelettes. *C.R. Acad. Sci. Paris*, 308, Série 1, 1989, pp. 79-81.

[15] S. Kambhampati and L. Davis. Multiresolution path planning for mobile robots. *IEEE Journal on Robotics and Automation.* RA-2 (3) pp. 135–145, 1986.

[16] M. D. Kelly, "Edge detection in pictures by computer using planning," Machine Intelligence, v. 6, pp. 397–409, 1971.

[17] J.C. Latombe. *Robot Motion Planning*, Kluwer, 1991.

[18] T. A. Linden and J. Glicksman "Contingency planning for an Autonomous Land Vehicle", IJCAI 87, pp. 1047–1054, 1987.

[19] M. D. Levine, "A knowledge-based computer vision system," in *Computer Vision Systems*, Hanson and Riseman, eds., Academic Press, pp. 335–354, 1978.

[20] S. Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Trans. PAMI* 11, 1989, pp. 674–693.

[21] S. Mallat and W.L. Hwang. Singularity detection and processing with wavelets. *IEEE Trans. Inform. Theory* 38, 1991, pp. 617–643.

[22] F. Mokhtarian and A.K. Mackworth, Scale–based description and recognition of planar curves and two-dimensional shapes, IEEE PAMI, 8, pp. 34–43, 1986.

[23] H. Noborio, T. Naniwa, and S. Arimoto. A quadtree-based path-planning algorithm for a mobile robot. *J. Robotic Systems*, 7(4):555–574, 1990.

[24] D. K. Pai, R. Barman, S. Ralph. "Platonic Beasts: a new family of multilimbed robots," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1019–1025, San Diego, May 1994.

[25] J. K. Peterson. "Obstacle avoidance using hierarchical dynamic programming", in *IEEE Southeastern Symposium on System Theory*, pp. 192 –196, 1991.

[26] L.-M. Reissell. Multiresolution Geometric Algorithms Using Wavelets: Representation for Parametric Curves and Surfaces. Department of Computer Science, UBC, Technical Report TR 93–17, 1993.

[27] Rosenfeld, A. (Ed.) "Multiresolution Image Processing and Analysis", Springer-Verlag, 1982.

[28] E. Schalit. ARCANE: Towards Autonomous Navigation on Rough Terrains. *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 2568–2574, 1992.

[29] Z. Shiller, J. C. Chen. Optimal Motion Planning of Autonomous Vehicles in Three Dimensional Terrain *Proceedings of the IEEE International Conference on Robotics and Automation,* pp. 198–203, 1990.

[30] T. Simeon and B. Dacre-Wright. A practical motion planner for all-terrain mobile robots. *Proc. IEEE Int. Conf. on Intelligent Robots and Systems*, pp. 1357–1363, 1993.

[31] G. Strang. Wavelets and dilation equations: a brief introduction. *SIAM Review* 31(4), 1989, pp. 614–627.

[32] D. Terzopoulos, "Multiresolution Computation of Visible-Surface Representations", Ph.D. thesis, MIT, 1984.

[33] S. Tanimoto and T. Pavlidis. A hierarchical data structure for picture processing. *CGIP* 4, 2, 104–119, 1975.

[34] A. Witkin. Scale space filtering. *Proc. Internal. Joint Conf. Artificial Intelligence*, 1983.

[35] D. Zhu and J. C. Latombe. New heuristic algorithms for efficient hierarchical path planning, *IEEE Transactions on Robotics and Automation.* 7 (1) pp. 9–20, 1991.