Underwater Path Planing Using Fast Marching Algorithms

Clément Pêtrès[*], Yan Pailhas[†], Yvan Petillot[‡] and Dave Lane[§]
[*]School of Electrical and Physical Sciences
Heriot Watt University, Edinburgh, EH14 4AS
Email: cp23@hw.ac.uk
[†]School of Electrical and Physical Sciences
Email: y.pailhas@hw.ac.uk
[‡]School of Electrical and Physical Sciences
Email: y.r.petillot@hw.ac.uk
[§]Ocean Systems Laboratory
Email: d.m.lane@hw.ac.uk

Abstract— **In this paper, new tools for obstacle avoidance and path planning for underwater vehicles are presented. The authors technique, based on a level set formulation of the path planning problem, extracts optimal paths from *complex* and *continuous* environments in a *complete* and *consistent* manner. Fast Marching algorithm is known to be efficient for finding cost optimal path in mobile robotics because of its reliability, precision, and simple implementation. Fast Marching algorithm originally propagates a wave front to isotropically explore the space. We propose an anisotropic version of Fast Marching by adding *directional constraints* in a cost function to minimize. We then propose a path planning method able to deal with vectorial fields of force for the first time. Furthermore we explore the relation between the curvature of the optimal path and the cost function generated from scalar and vectorial constraints. This *a priori* knowledge of the influence of the environment on the final path curvature allows us to propose a solution to make sure a path is reachable by the vehicle according to its kinematics. A *multiresolution* scheme based on an adaptive mesh generation is eventually introduced to speed up the overall algorithm. Results are shown computed from real and simulated underwater environments.**

I. Introduction

In mobile robotics path planning issue is mostly addressed for wheeled robots moving on 2D surfaces fitted out with high rate communication modules. Underwater environment is fairly more exigent. It is unfriendly to communicate, prone to currents and the set of possible paths is a 3D space. Moreover underwater torpedo like vehicles are strongly nonholonomic, contrarily to wheeled robots capable to easily stop and rotate.

Literature proposes many methods such as potential field or roadmap methods (see for example [1] for a concise overview). They are either not complete or computationally expensive as soon as the environment becomes complex [2].

The method we describe in next sections is a part of the cell decomposition approach. The map of the environment is first divided into a set of cells. The adjacency of the cells is represented as a connectivity graph which is then searched to find a channel from start to goal configurations. Practically these approaches are widely used in mobile robotics because they are very suitable to sensor images decomposed in a grid of pixels. The key issue is then to use an efficient graph search algorithm to find a path through a network of cells connected by cost links.

Breadth-first search, heuristic search and hybrid search (like A*) algorithms are very popular in path planning [3]. Their principle is always to build a distance map weighted by costs of obstacles. The construction is performed by evolving a level set front from the start point (seen as a source) to the goal point to reach. The optimal path is then found by performing a descent backtracking on the distance map from the goal point to the source of exploration.

These "best-first" search algorithms are complete (they converge to a solution if one exists) but they suffer from metrication errors. Results of these discrete graph-search algorithms can be improved by taking a larger neighbourhood as a structuring element but they are still not consistent, there will always be an error in some directions that will be invariant to the grid resolution [4]. It is not the case with the Sethian's Fast Marching (FM) algorithm [5].

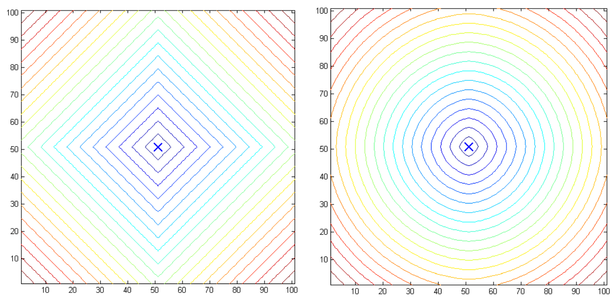

Fig. 1. On the left Euclidean distance computed with a 4-connexity breadth-first algorithm gives square level sets. On the right, the distance is computed with Fast Marching, giving circles.

Fast Marching algorithm is also a Dijkstra's like graph-search algorithm but is consistent in the continuous domain. The idea behind FM algorithm is to improve the update of expanded nodes by approximating the first derivative of the distance map while respecting the entropic flow. For the same

$O(n.\lg(n))$ complexity, where n is the number of nodes, FM algorithm provides better approximations for level sets as we can see on the figure 1.

The original Sethian's Fast Marching algorithm is based on an isotropic assumption of the free space. However as FM algorithm explicitly estimates the gradient of the distance map which can be interpreted as the moving direction, it is possible to take directional constraints such as winds or currents into account. This feature is interesting for path planning in underwater environment. Theory of anisotropic Fast Marching was first developed by Vladimirsky [6]. We propose in this paper a simplified implementation of his method to speed it up with regard to smooth underwater fields of force.

It has been shown [4] that the curvature of an optimal path found after a functional minimization is explicitly bounded by gradient of the initial cost function. We extend this result to our anisotropic FM method which allow us to introduce the vehicle's kinematics as a constraint to make sure the optimal path reachable by a nonholonomic underwater robot.

Finally we propose a multiresolution scheme to speed up the overall method in order to cope with hard real-time constraints of autonmous underwater path planning. The method proposed by the authors is to couple an octree decomposition with an adaptive mesh generation.

The structure of the paper is as follows. Section II presents the Sethian's Fast Marching algorithm and curvature constraints induced with this technique. We propose in section III our anisotropic version of Fast Marching and an extension of Cohen and Kimmel's results on path curvature. Section IV presents some results, our multiresolution approach and opens a discussion about limitation and future work.

II. Fast Marching Algorithm

Given a discrete map of obstacles, our goal is to find a global cost optimal path between two points. We first translate the map of obstacles into a cost function f. Then the "cost optimal" question leads to algorithms that seek for the minimal path, that is to say the path along which integration over f is minimal. Inspired by the paper of Cohen and Kimmel [4] we begin with comparing path planning problem with geometrical optics in physics.

A. Geometrical Optics

The Fermat principle in optics is the physical interpretation of minimal paths described in next sections.

The Fermat principle stipulates that the path of monochromatic light is the shortest in term of time. Into an homogeneous environment the light velocity is constant so sets of points reached at a given time are circles and minimal paths are straight lines (left subfigure 2).

Let's now consider an inhomogeneous environment separated by an interface as shown in the right part of figure 2. We assimilate in that case the cost function f to the refractive index n, $f = n$. The speed of light can be written $v = \frac{c}{n} = \frac{c}{f}$. Since speed of light is faster on the top part than on the bottom, light tends to stay above the interface and it yields the

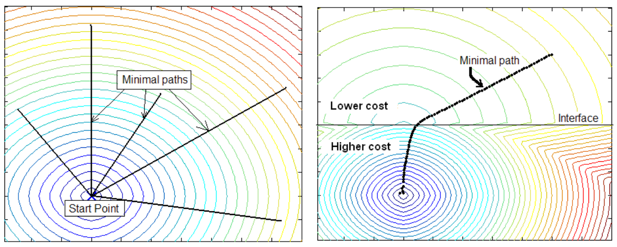

Fig. 2. Examples of optimal paths with one or two constant values for the cost function.

refractive effect well known in Snell-Descartes laws. Optimal path is the one that minimizes the time T between two points p_0 and p_1 : $T = \frac{1}{c}\int_{p_0}^{p_1} f ds$.

B. Problem Formulation

We are looking for paths along which the integral of f is minimal. The distance map U is defined as the minimal energy integrated along a path C between a starting point p_0 and any point p:

$$U(p) = \inf_{A_{p_0,p}} E(C) = \inf_{A_{p_0,p}} \left\{ \int_D f((s))ds \right\} \qquad (1)$$

where D is the image domain and $A_{p_0,p}$ is the set of all paths between p_0 and p.

The minimal path between p_0 and any point p_1 in the image can be easily deduced from the distance map. Assuming that cost function f is *stricly positive*, the action map will have only one local minimum which is the starting point p_0, and the minimal path will be found by a simple backtracking on the distance map. This characteristic makes Fast Marching algorithm very reliable which is important for an autonomous oriented underwater path planner.

C. Fast Marching Resolution

In order to compute the distance map U, a front-propagation equation related to equation 1 is solved:

$$\frac{\partial C}{\partial t} = \frac{1}{f}\vec{n} \qquad (2)$$

where $\vec{n}$ is the normal to the front. This Eulerian formulation for curve evolution was introduced by Osher and Sethian [7] to overcome numerical difficulties and handle topological changes. A front starting from an infinitesimal circle shape around p_0 is evolved until the goal point is assigned a value for U.

Fast Marching technique, introduced by Sethian [5], was addressed by Cohen and Kimmel [4] who noticed that map U satisfies the Eikonal equation:

$$\|\nabla U\| = f \;\; and \;\; U(p_0) = 0 \qquad (3)$$

Classic finite differences schemes for this equation tend to overshoot and are unstable. Sethian proposed to use the

Godunov Hamiltonian which is a one-sided derivative. It looks to the up-wind direction of the moving front, and thereby avoids the over-shooting of finite differences. At each pixel (i,j), the unknown u satisfies:

$$(\max\{u-U_{i-1,j},u-U_{i+1,j},0\})^2+ \\ (\max\{u-U_{i,j-1},u-U_{i,j+1},0\})^2=f_{i,j}^2 \qquad (4)$$

yielding the correct viscosity solution u for $U_{i,j}$.

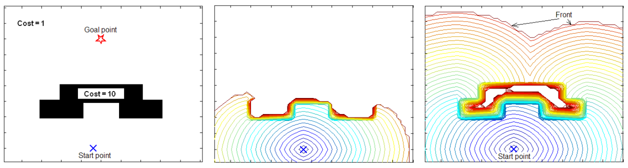

Fig. 3. A front propagation relative to the cost function on the left.

The originality of Fast Marching is to introduce an order in the selection of grid points. This order is based on the fact that information is propagating outward, because distance can only grow due to the quadratic equation 4. Fast Marching algorithm is detailed in table 1.

TABLE I
ALGORITHM FOR 2D FASTMARCHING

```
1) Definitions
   • Alive is the set of all grid points at which the distance value $\mathcal{U}$
     has been reached and will not be changed;
   • Trial is the set of next grid points to be examined and for which
     an estimate U of $\mathcal{U}$ has been computed using equation 4 only
     from Alive points;
   • Far is the set of all other grid points, for which there is not yet
     an estimate for $\mathcal{U}$
2) Initialization
   • Alive set is confined to the starting point $p_0$, $U(p_0)=\mathcal{U}(p_0)=0$;
   • Trial - the initial front is confined to the neighbors of $p_0$ with
     initial values $U(p)=f(p)$;
   • Far is the set of all other grid points, $U=\infty$;
3) Loop
   • Let $p=(i_{min},j_{min})$ be the Trial point with the smallest
     distance U;
   • Move it from the Trial set to the Alive set (i.e. $U_{i_{min},j_{min}}=\mathcal{U}_{i_{min},j_{min}}$ is frozen);
   • For each neighbor (i, j) (4-connexity in 2D) of $(i_{min},\ j_{min})$:
     a) If (i, j) is Far, add it to the Trial set and compute a first
        estimate U of $\mathcal{U}$ using equation 4
     b) If (i, j) is Trial, update the distance $U_{i,j}$ using equation 4
```

D. Algorithm for 2D Up-Wind Scheme

Notice that only Alive points are considered to solve equation 4. We examine neighbors of point (i, j) in 4-connexity, see figure 4.

Let $\{A_1,A_2\}$ and $\{B_1,B_2\}$ are the two opposite couples with $U(A_1)\le U(A_2)$, $U(B_1)\le U(B_2)$, and $U(A_1)\le U(B1)$. Considering that $u\ge U(B_1)\ge U(A_1)$ equation 4 becomes:

$$(u-U(A_1))^2+(u-U(B_1))^2=f_{i,j}^2 \qquad (5)$$

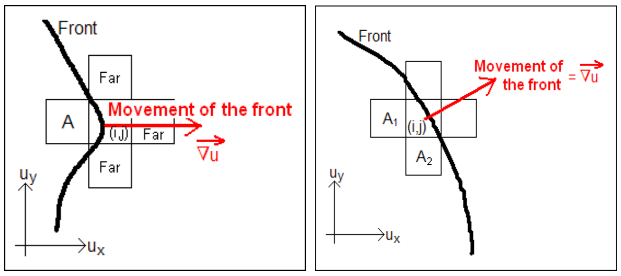

Fig. 4. On the left example of a new Alive point only surrounded by one other Alive point, on the right a new Alive point surrounded by at least two other Alive points.

Based on the discriminant test delta of equation 5, one or two neighbors are used to solve it:

- if $f_{i,j}>U(B_1)-U(A_1)$ then $u=\frac{U(A_1)+U(B_1)+\sqrt{2f_{i,j}^2-(U(B_1)-U(A_1))^2}}{2}$
- else $u=U(A_1)+f_{i,j}$

E. 3D Fast Marching

An extension of Fast Marching algorithm to three dimensions is easy to develop. Similarly to the 2D case distance U is defined as follows:

$$U(p)=\inf_{A_{p_0,p}}\left\{\int_D f(C(s))\right\} \qquad (6)$$

where $A_{p_0,p}$ is the set of all paths between p_0 and p inside the 3D domain D.

Given a start point p_0 a front evolves starting from an infinitesimal spherical shape around p_0. The 2D numerical scheme of equation 4 is extended to 3D:

$$(\max\{u-U_{i-1,j,k},u-U_{i+1,j,k},0\})^2+ \\ (\max\{u-U_{i,j-1,k},u-U_{i,j+1,k},0\})^2+ \\ (\max\{u-U_{i,j,k-1},u-U_{i,j,k+1},0\})^2=f_{i,j,k}^2 \qquad (7)$$

giving the correct viscosity solution u for $U_{i,j,k}$.

F. Kinematic constraints

In this section we explain the influence of a cost function f on the curvature of the final optimal path. Following the paper of Caselles, Kimmel and Sapiro on geodesic active contours [8], the curvature radius r along the geodesic minimizing the functional $\int_D f((s))ds$ is bounded by:

$$\boxed{r\ge\frac{\inf_D\{f\}}{\sup_D\{\|\nabla f\|\}}} \qquad (8)$$

This result gives us a nice interpretation of the connection between cost function f and curvature along the resulting optimal path. It is useful because, given any map of obstacles, we can know *a priori* if the optimal path will be reachable or not by the vehicle. We just have to compare the turning radius R of the vehicle with the limit of the curvature radius r_{lim} of the path:

- if $R < r_{lim}$, it is certain that the minimal path will be reachable.
- if $R \geq r_{lim}$ there is a risk of collision. In that case we just have to smooth f to increase the curvature limit until $r_{lim} > R$.

III. Anisotropic Fast Marching Algorithm

A. Introduction

The theory of anisotropic Fast Marching was first developed by Vladimirsky [6]. The principle is to make the cost function f dependent not only from scalar cost of obstacles but also dependent on vectorial cost of forces. Vladimirsky formally demonstrates how the characteristic of the distance map can be used for this purpose.

In this section we propose a simplified implementation of his method by considering the gradient explicitly defined in expression 5 as an approximation of the characteristic. This is equivalent to assume that the field of force $\vec{F}$ is quite smooth. Original Sethian's FM rapidity relies on the resolution of the simple quadratic equation 5 for u. Since f appears as a square in this equation our idea is to build a new cost function linearly dependent on u.

B. New cost function

We split the cost function f in two parts by defining a new cost function $\tilde{f} = f_{obst} + f_{vect}$. f_{obst} remains linked to obstacles as previously and f_{vect} is defined as follows:

$$\boxed{f_{vect}(i,j) = \alpha\left(1 - \frac{\langle \nabla u_{i,j} \cdot \vec{F_{i,j}} \rangle}{Q_{i,j}}\right) \geq 0} \quad (9)$$

where α is a positive gain and Q is a normalization term so that $\forall (i,j) \in D \left\| \frac{\langle \nabla u_{i,j} \cdot \vec{F_{i,j}} \rangle}{Q_{i,j}} \right\| \leq 1$.

It is equivalent to say that a force favours the vehicle when both force and vehicle are pointing in the same direction.

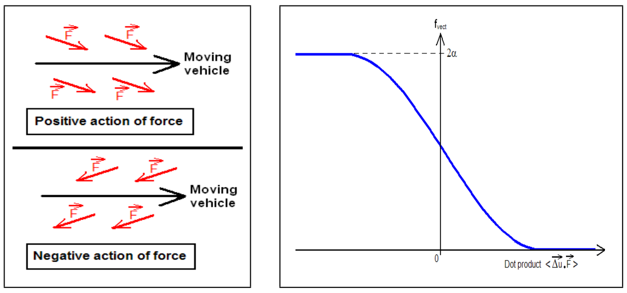

Fig. 5. On the left positive and negative actions of force applied to a mobile vehicle, on the right appearance of f_{vect}.

C. Anisotropic problem formulation

The problem for anisotropic FM becomes the same than for isotropic FM. We are looking for paths along which the integral of $\tilde{f} = f_{obst} + f_{vect}$ is minimal. Distance U is defined as the minimal energy integrated along a path between a starting point p_0 and any point p :

$$U(p) = \inf_{A_{p_0,p}} E(C) = \inf_{A_{p_0,p}} \left\{ \int_D \tilde{f}(C(s))ds \right\} \quad (10)$$

where $A_{p_0,p}$ is the set of all paths between p_0 and p on the image domain D.

Resolution of our anisotropic version of Fast Marching algorithm has just to be slightly modified to compute more precisely a first estimate of Far points. But the overall resolution is exactly the same than the one for isotropic Fast Marching by replacing f with $\tilde{f}$ (see figure 6 for illustration).

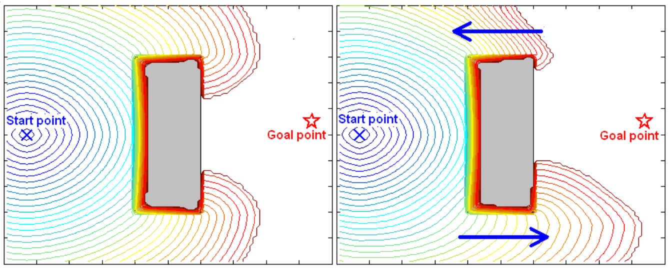

Fig. 6. On the left an isotropic Fast Marching, on the right our anisotropic version (currents are symbolized with arrows).

D. Kinematic constraints

For isotropic Fast Marching the curvature radius r along the geodesic is bounded. In this section we extend this relevant result to our anisotropic Fast Marching version.

We start from the previous result of section II.F. Given a cost function f, the curvature radius $r = \left\| \frac{\partial^2 C}{\partial s^2} \right\|^{-1}$ along the geodesics minimizing $\int_D f((s))ds$ is bounded by $\frac{\inf_D\{f\}}{\sup_D\{\|\nabla f\|\}}$.

With our new cost function $\tilde{f}$ one can show that:

$$\boxed{r \geq \frac{\inf_D\{f_{obst}\}}{\sup_D\{\|\nabla f_{obst}\|\} + \frac{2\alpha}{\inf_D\{Q\}}\|J_F\|_\infty}} \quad (11)$$

where J_F is the Jacobian of $\vec{F}$ on D and $\|\cdot\|_\infty$ is the L-infinity norm.

The conclusion is that to increase the curvature radius r we have two main choices:

- Smoothing f_{obst} to decrease $\sup_D \{ \|\nabla f_{obst}\| \}$.
- Smoothing the vectorial field of force $\vec{F}$ to decrease $\|J_F\|_\infty$.

IV. Applications in Underwater Robotics

We present in this section some results from real and simulated 2D underwater environments.

A. Isotropic Fast Marching

1) Non-convex obstacle: Fast Marching algorithm associated with a gradient descent algorithm naturally overcomes local minima (which may be induced by concavities) since start point is the only global minimum (see figure 7).

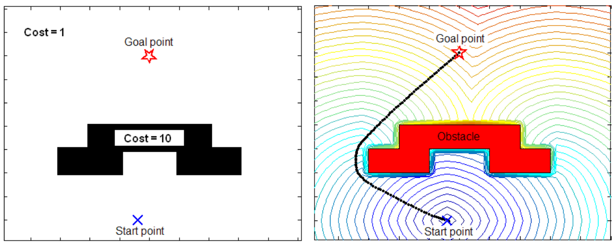

Fig. 7. On the right the optimal path corresponding to the map on the left.

2) Complex map: Fast Marching algorithm is efficient with complex obstacle maps such as the one illustrated figure on 8.

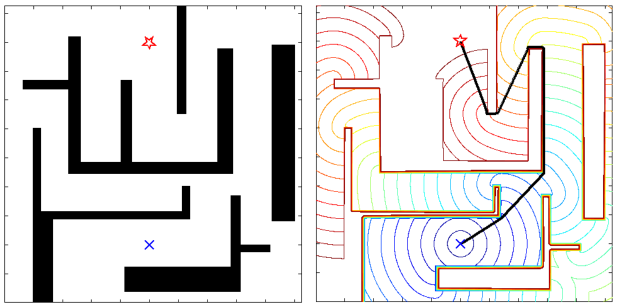

Fig. 8. The optimal path on the right corresponds to the obstacle map (1000x1000 pixels) on the left.

Minimal paths are necessarily close to obstacles. To increase the safety distance of paths from obstacles, we just have to enlarge the highest values of cost function. A single morphological operation is required.

3) Real application: A C++ implementation of 2D Fast Marching method has been performed in the laboratory applied to real sonar images (figure 9).

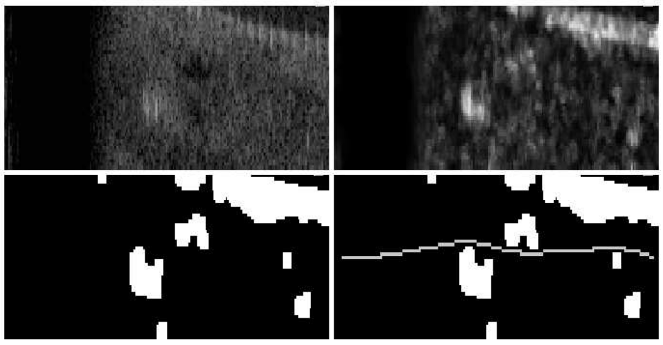

Fig. 9. From up to down and left to right, original sonar image, processed images and optimal path.

Sizes of these real images are 200x50 pixels. Time computations are about 10 ms for Fast Marching and 1 ms for gradient descent.

B. Anisotropic Fast Marching

We present here a simple example of our anisotropic Fast Marching implementation. Figure 10 shows how currents can influence optimal paths.

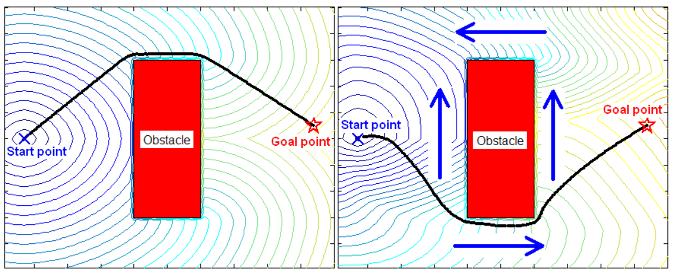

Fig. 10. On the left an isotropic Fast Marching, on the right our anisotropic Fast Marching with currents symbolized with arrows.

C. Kinodynamic motion planning

We have seen in sections II.F. and III.D. that given a map of obstacles (and possibly a map of currents) we are able to know if the minimal path will be reachable or not by the vehicle. If not the cost function f (or $\tilde{f}$) has to be smoothed. Figure 11 gives an example of the influence of the curvature limit to optimal paths.

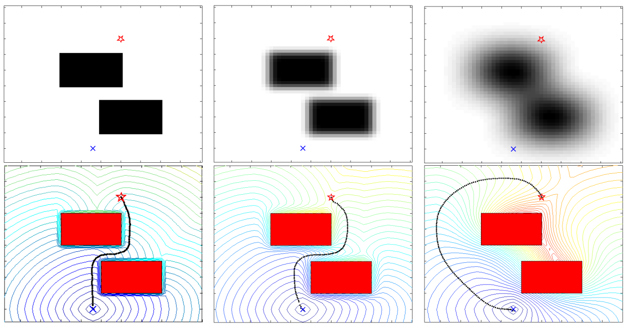

Fig. 11. On the top, from left to right, cost function corresponding respectively to $r_{lim} = 14, 34, 140$. On the down part, the related optimal paths.

D. Multiresolution path planning

A multiresolution method starts with the idea that it is not necessary to represent the entire grid with a high uniform resolution. We propose in this section a combined method to make it usable for real-time 3D underwater applications.

1) Octree decomposition: The octree decomposition is one of the most popular multiresolution approach. Although the computational cost saving of this method is obvious, see for example [9], the initial image is transformed in a tree data structure which is not very convenient to keep a clear knowledge of the spatial neighbourhood of each block.

2) Fast Marching on meshes: The method proposed by the authors is to couple the octree decomposition with an adaptive mesh generation. The Delaunay triangulation is a good candidate as fast and robust implementations exist. Input of this mesh generation is the set of nodes with their cost links given by the octree decomposition, output is the net of vertices linked to their neighbors by cost edges.

Inspired by works of Sethian and Vladimirsky [10] we implemented Fast Marching algorithm on multiresolution unstructured meshes. Figure 12 illustrates our method.

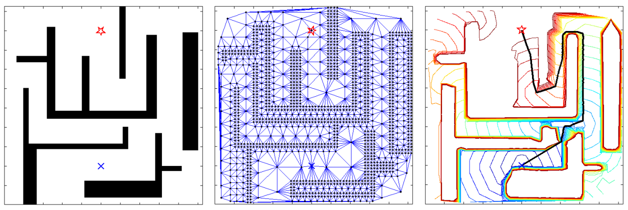

Fig. 12. On the left the original 1000x1000 image, on the middle the adaptive mesh with only 1400 vertices, on the right the optimal path found by backtracking on the interpolated distance map.

The overall multiresolution method requires five steps. First an octree decomposition of the orignal image, secondly a Delaunay mesh generation, thirdly a Fast Marching on the mesh (instead of performing it on the grid), an interpolation of the distance map (computed on mesh vertices) on grid points and a gradient descent on the interpolated distance map.

This method is approximately 1000 times faster than a brute Fast Marching on uniform cartesian grids for similar looking paths (see figures 8 and 12 to compare paths).

E. Limitations and future work

1) Noisy conditions: If Fast Marching algorithm is applied to noisy images its convergence to the optimal solution is not guaranteed any more. Figure 13 illustrates the problem.

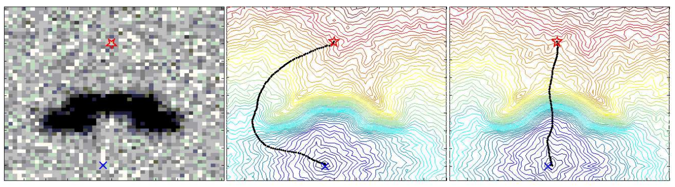

Fig. 13. From left to right, a noisy (Gaussian white noise) map and two possible paths.

Image pre-processing is required before any Fast Marching computation on real data. This issue depends on the sensors used but for most of the cases it can probably be achieved with probabilistic methods (like Markov fields or Bayesian methods).

2) Anisotropic Fast Marching: We defined $f_{vect} = \alpha\left(1 - \frac{\langle\nabla u\cdot\vec{F}\rangle}{Q}\right) \geq 0$ for f_{vect} to be linear for u. This choice allows small runtimes but most of underwater vehicles have a more complex behaviour than a linear reaction to currents. In that case an acceptable first order approximation need to be known. Sophisticated vehicle's models can be used for performing an anisotropic Fast Marching by building complex hamiltonians [6] but the method is not fast any more!

3) Multiresolution path planning: We have seen in section IV.D. that our multiresolution method is much faster than building a distance map on all the uniform grid of an image. Results look similar however we need at this stage an analytical tool to compare quality of resulting optimal paths.

V. Conclusion

This paper contributes to improve four key issues for underwater path planning.

The reliability and autonomy abilities of path planners to cope with low bandwidth channels in the water are improved by introducing a complete and consistent algorithm called Fast Marching.

We proposed a practical implementation of anisotropic Fast Marching to make our path planning method robust to hostile underwater currents.

Path adequation with underwater robot kinematics was addressed in a discussion about influence of the cost function on the path curvature for both isotropic and anisotropic Fast Marching algorithms.

High computation costs due to the extra third dimension are drastically reduced thanks to a multiresolution approach.

The overall method is validated on real and simulated 2D sensor inputs. These tests let some questions open for future work. They especially highlight the need of an objective tool to compare quality of "optimal" paths.

References

[1] A. B. Doyle, *Algorithms and Computational Techniques for Robot Path Planning*. University of Wales, Bangor: PhD thesis, 1995.

[2] S. M. LaVallee, *Planning Algorithms*, University of Illinois, 2005.

[3] S. Koenig, M. Likhachev, Y. Liu, and D. Furcy, "Incremental heuristic search in artificial intelligence," *Artificial Intelligence Magazine*, 2004.

[4] L. Cohen and R. Kimmel, "Global minimum for active contour models: A minimal path approach," *Internationnal Journal of Computer Vision*, vol. 24, no. 1, pp. 57–78, 1997.

[5] J. A. Sethian, *Level Set Methods and Fast Marching Methods*. Cambridge, Massachusetts: Cambridge University Press, 1999.

[6] A. Vladimirsky, "Ordered upwind methods for static hamilton-jacobi equation: Theory and algorithms," *SIAM Journal of Numerical Analysis*, vol. 41, no. 1, pp. 325–363, 2003.

[7] J. A. Sethian and S. J. Osher, "Front propagation with cuvature dependent speed: algorithms based on hamilton-jacobi formulations," *Journal of Computational Physics*, vol. 79, pp. 12–49, 1988.

[8] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic active contours," in *Proceedings of 5th IEEE International Conference on Computer Vision (ICCV'95)*, Cambridge, USA, 1995.

[9] S. Behnke, "Local multiresolution path planning," in *Proceedings of 7th RoboCup International Symposium*, Padua, Italy, 2004.

[10] J. A. Sethian and A. Vladimirsky, "Fast methods for the eikonal and related hamilton-jacobi equations on ustructured meshes," *Applied Mathematics*, vol. 97, no. 11, pp. 5699–5703, May 23 2000.