Path Planning for Autonomous Underwater Vehicles

Clément Pêtrès, Yan Pailhas, Pedro Patrón, Yvan Petillot, Jonathan Evans, and David Lane

***Abstract*—Efficient path-planning algorithms are a crucial issue for modern autonomous underwater vehicles. Classical path-planning algorithms in artificial intelligence are not designed to deal with wide continuous environments prone to currents. We present a novel Fast Marching (FM)-based approach to address the following issues. First, we develop an algorithm we call FM* to efficiently extract a 2-D continuous path from a discrete representation of the environment. Second, we take underwater currents into account thanks to an anisotropic extension of the original FM algorithm. Third, the vehicle turning radius is introduced as a constraint on the optimal path curvature for both isotropic and anisotropic media. Finally, a multiresolution method is introduced to speed up the overall path-planning process.**

***Index Terms*—Autonomous underwater vehicle (AUV), currents, Fast Marching (FM), FM* algorithm, multiresolution method, path planning, turning radius.**

I. Introduction

A. Underwater Environment and Autonomous Underwater Vehicle

In MOBILE robotics, path-planning research has focused on wheeled robots moving on surfaces equipped with high-rate communication modules. The underwater environment is much more demanding: it is difficult to communicate because of low-bandwidth channels undersea; it is prone to currents; and the workspace may be worldwide. Moreover, torpedo-like vehicles are strongly nonholonomic.

The current state of technology allows many laboratories to move forward in the development of autonomous underwater vehicles (AUVs). The need of a reliable cognition process for finding a feasible 2-D trajectory solution derived from 2-D underwater imagery is important.

B. Contributions

The main contribution of the authors is to present a Fast Marching (FM)-based method as an advanced tool for underwater path planning. With a similar complexity to classical techniques in artificial intelligence, the FM algorithm converges to a smoothed continuous solution when implemented on a sampled environment. This specificity is crucial to understanding of the other contributions of our method.

- A heuristically guided version FM* of the FM algorithm is developed. FM* combines the efficiency of the A* algorithm (described in the next section) with the accuracy of the FM algorithm.
- FM* allows the curvature of the final path to be constrained, which enables us to take the turning radius of any mobile robot into account. This property is formally proven for both isotropic and anisotropic media.
- We show that ordered-upwind-methods-based path planners enable the addition of directional constraints. We propose an anisotropic FM algorithm able to deal with smooth fields of force like underwater currents, but this concept can be generalized for any kind of directional constraint.
- We propose a multiresolution scheme to speed up the overall method in order to cope with real-time constraints of autonomous underwater path planning. This is achieved by implementing the FM algorithm on adaptive unstructured meshes.

C. Related Work

An FM-based path planner is proposed in [1] that allows dynamic replanning. No heuristic is introduced in this method to speed up the exploration process. The replanning ability counterbalances this lack of efficiency in the case of *a priori* unknown terrain. The Field D* algorithm [2] is another approach, which is close to the FM* algorithm in the sense that it is an interpolated version of the D* algorithm (described in [3]) to the continuous domain. In practice, the FM* algorithm is easier to implement and does not present the pathological cases of the Field D* algorithm when the path is extracted using a gradient descent.

There is limited literature on path planning for AUVs. Carroll *et al.* [4] used the A* algorithm. Warren [5] used a potential field method. This technique is fast, but it may be affected by local minima. Techniques such as sequential quadratic programming [6], case-based reasoning [7], and genetic algorithms [8], [9] have also been applied to the motion planning of underwater robotic vehicles.

The issue of path planning under directional constraints has been addressed by Alvarez *et al.* In [10], the authors used genetic algorithms, which is an offline technique even if they optimize it using dynamic programming. In [11], the A* algorithm is adapted to take current influence into account. This method is close in spirit to our anisotropic FM algorithm, apart from the fact that it is based on a discrete motion model for the AUV.

Manuscript received May 4, 2006; revised September 23, 2006. This paper was recommended for publication by Associate Editor G. Antonelli and Editor L. Parker upon evaluation of the reviewers' comments. This paper was presented in part at the Oceans Europe Conferences, Brest, France, June 2005, and in part at the IJCAI Workshop on Planning and Learning in *A Priori* Unknown or Dynamic Domains, Edinburgh, U.K., August 2005.

The authors are with the Ocean Systems Laboratory, Heriot-Watt University, Edinburgh EH14 4AS, U.K. (e-mail: clementpetres@yahoo.fr; y.pailhas@hw.ac.uk; p.patron@hw.ac.uk; y.r.petillot@hw.ac.uk; j.evans@hw.ac.uk; d.m.lane@hw.ac.uk).

Color versions of Figs. 1–7, 9–13, and 15–17 are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TRO.2007.895057

D. Overview of the Paper

The plan of the paper is as follows. Section II reviews the framework of our FM-based planning technique. We rigorously present it as a part of the broader class of sampling-based path-planning methods. In this section, it is shown that the FM algorithm is an extension of classical artificial-intelligence algorithms to the continuous domain. Section III briefly details the FM algorithm and presents results using the FM* algorithm introduced in this paper. Section IV presents an anisotropic version of the FM algorithm to deal with directional constraints such as currents. In Section V, the turning radius of the vehicle is introduced as a constraint on the path solution. Finally, Section VI presents a multiresolution method to quickly extract suboptimal trajectories.

II. Path-Planning Framework

A. Environment Representation

The uniform framework to study the path-planning problem among static obstacles is the *configuration space (C-space)*. The main idea of the C-space is to represent the robot as a point, called a *configuration*.

A robot configuration is a vector of parameters specifying position, orientation, and all the characteristics of the robot in the environment. The C-space is the set of all possible configurations. We denote *C-free* as the regions of C-space which are free of static obstacles. Obstacles in the workspace become *C-obstacles* in the C-space. Usually, a simple rigid body transformation [12] is used to map the real environment into the C-space.

B. Problem Statement

Given a C-space Ω, the path-planning problem is to find a curve

$$C: \quad [0,1] \rightarrow \text{C-free}, \quad s \mapsto C(s)$$

where s is the arc-length parameter of C. We focus on 2-D C-spaces in this paper; nonetheless, this framework holds for C-spaces of any dimensions.

An optimal path is a curve C that minimizes a set of internal and external constraints (time, fuel consumption, or danger, for instance). We assume in this paper that the complete set of constraints is described in a cost function τ, which can be isotropic or anisotropic.

- Isotropic case: the cost function τ depends only on the configuration x

$$\tau: \quad \Omega \rightarrow \mathbb{R}_+, \quad x \mapsto \tau(x),\ \tau(x) > 0.$$

- Anisotropic case: τ depends on the configuration x and a vector $\vec{F}$ of a field of force $\mathcal{F}$

$$\tau: \quad \Omega \times \mathcal{F} \rightarrow \mathbb{R}_+, \quad (x, \vec{F}) \mapsto \tau(x, \vec{F}),\ \tau(x, \vec{F}) > 0.$$

In Sections II and III, the C-space Ω is assumed to be isotropic. Directional constraints are introduced in Section IV.

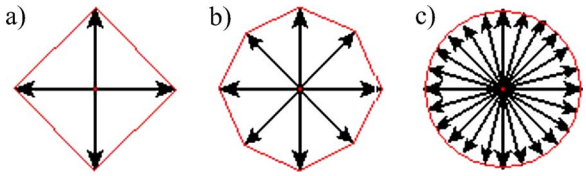

Fig. 1. Three examples of motion models. a) 4-connectivity. b) 8-connectivity. c) Continuous motion models. Cost-to-go level sets corresponding to these three motion models are, respectively, squares, octagons, and circles.

C. Sampling-Based Path-Planning Methods

The main idea in sampling-based path-planning methods is to avoid an exhaustive construction of C-obstacles by sampling the C-space (see [13] for an exhaustive study of sampling-based methods). The sampling scheme may be probabilistic (see [14] for a survey on probabilistic path planning) or may be deterministic (see [15] for a study of the relationship between the deterministic and the probabilistic approaches). We focus in this paper on the uniform Cartesian deterministic sampling scheme, although we propose a nonuniform deterministic sampling scheme further in Section VI. Cartesian sampling-based methods are widely used in mobile robotics because they are suitable for sensor images mapped into a grid of pixels. The key issue is then to use an efficient grid-search algorithm to find an optimal path in the sense of a metric.

1) Metric Space: A metric ρ defines the distance between two configurations in the C-space, which becomes a metric space (see [13] for a rigorous definition of a metric space).

In this paper, the metric ρ we refer to is defined as

$$\rho(x, x') = \int_{[0,1]} \tau\left(C_{x,x'}(s)\right) ds \tag{1}$$

where $C_{x,x'}$ is a path between two configurations x and x', and τ is a strictly positive cost function.

This metric can be seen as the "cost-to-go" for a specific robot to reach x' from x. At a configuration x, $\tau(x)$ can be interpreted as the cost of one step from x to its neighbors. If a C-obstacle in some region S is impenetrable, then $\tau(S)$ will be infinite. τ is supposed to be strictly positive for an obvious physical reason: $\tau(x) = 0$ would mean that free transportation from some configuration x is possible.

2) Motion Models: Metric ρ is defined for a C-space assuming a continuous-motion model. However, since the C-space is partitioned into a Cartesian grid, grid-search algorithms commonly use a 4- or a 8-connectivity discrete motion model for the robot, see Fig. 1.

A discrete approximation ρ_d of metric ρ is defined as

$$\rho_d(x, x') = \sum_{i=1}^{n} \tau(x_i) \tag{2}$$

where $x_1 = x$ and $x_n = x'$, and transitions between x_i and x_{i+1} are governed by a discrete motion model.

3) Grid-Search Principle: A grid-search algorithm is an optimization technique that successively performs an *exploration* and an *exploitation* process. The exploration process builds a "minimum cost-to-go" map, called *distance function*, from the

start to the goal configuration. The exploitation process is a *backtracking* from the goal to the start configuration.

The distance function $u : \Omega^2 \rightarrow \mathbb{R}_+$ is the solution of the functional minimization problem, defined as follows:

$$\begin{aligned} u(x_{\text{start}}, x) &= \inf_{\{C(x_{\text{start}},x)\}} \rho(x_{\text{start}}, x) \\ u(x_{\text{start}}) &= 0 \end{aligned} \tag{3}$$

where $\{C(x_{\text{start}}, x)\}$ is the set of all curves between the source x_{start} and the current configuration x. For the sake of notational simplicity, and assuming that the source of exploration x_{start} is fixed, we note $u(x_{\text{start}}, x) = u(x)$.

The distance function u can be related to the *value function* concept in reinforcement learning. The difference lies only in the fact that value functions are refined in an iterative process (called learning), whereas the distance function is built from scratch. In the path-planning literature, one can find other names for the distance function, such as navigation function [13], multivalued distance map [16], or convex map [17].

Once the distance function is found through the goal configuration, the optimal path is the one which follows the steepest descent over the distance function from the goal to the start configuration. This backtracking technique is reliable, as no local minima are exhibited during the exploration process.

4) Grid-Search Algorithms: Grid-search algorithms rely on a partitioning of the C-space in three sets: *Accepted* configurations for which the distance function u has been computed; *Considered* configurations for which u has been estimated; and the remaining *Far* configurations for which u is unknown.

The set of *Considered* configurations is stored in a priority queue. On top of this queue, the configuration with the highest priority is called the *trial*. At each iteration of the exploration process, the trial configuration is moved from *Considered* to *Accepted*, and its *Far* neighbors are updated and moved from *Far* to *Considered*. The exploration process expands from the start configuration and ends when the goal configuration is eventually set to *Accepted*.

According to the priority assignment of the *Considered* set, we distinguish two classes of grid-search algorithms (see [18] for a survey of grid-search algorithms).

1) *Breadth-first (BF) search*: BF algorithms give the highest priority to the *Considered* configuration x with the lowest estimate of the distance function u. $u(x)$ does not depend on the goal configuration. This is why the distance function is built symmetrically around the start configuration [see Fig. 2(a)].
2) *Hybrid search (HS)*: HS algorithms give the highest priority to the *Considered* configuration x, with the lowest $f(x)$ defined as $f(x) = (1-\alpha)u(x) + \alpha h(x, x_{\text{goal}})$. Here $h(x, x_{\text{goal}})$ is a heuristic that estimates the residual distance between the configuration x and the goal configuration x_{goal}, and α is a constant $(0 < \alpha \leq 1)$. If $\alpha = 0.5$, this popular HS algorithm is called the A* algorithm. Instead of exploring around the start configuration, HS algorithms focus the search towards the goal [see Fig. 2(b)].

5) Heuristic: When only sparse convex C-obstacles are present, heuristically guided searches are the most efficient techniques. These are the most commonly used heuristics. Let $x_{i,j}$ be the point of coordinates (i, j) on a grid.

- 4-connectivity heuristics

$$\begin{aligned} h_4\left(x_{i,j}, x'_{i',j'}\right) &= m\left(|i'-i| + |j'-j|\right) \\ h_{4_{\max}}\left(x_{i,j}, x'_{i',j'}\right) &= m\left(\max\left\{|i'-i|, |j'-j|\right\}\right). \end{aligned}$$

- Euclidean heuristic

$$h_e\left(x_{i,j}, x'_{i',j'}\right) = m\left((i'-i)^2 + (j'-j)^2\right)^{\frac{1}{2}} \tag{4}$$

where $m = \min_\Omega\{\tau\}$.

6) Computational Complexity: When a *Far* configuration is updated and moved to *Considered*, a routine is called to sort the *Considered* set in such a way that the configuration with the highest priority remains first in the queue. For optimal efficiency, the priority queue is stored in a heap structure [19].

Assume that the *Considered* list is ordered. When a new element is inserted, the price of reordering the list is $O(\log n)$, where n is the number of elements in the list. At each iteration of the exploration process, the *Considered* list is sorted at most three times (the four trial neighbors, excluding at least one adjacent *Accepted* configuration). Considering that the exploration process is iterated at most N times, where N is the total number of configurations in the grid, the computational complexity of grid-search algorithms is $O(N \log(N))$.

7) Conclusions: First, resulting paths from grid-search algorithms are discontinuous and not unique, because the robot is supposed to follow a discrete motion model. Further, even when

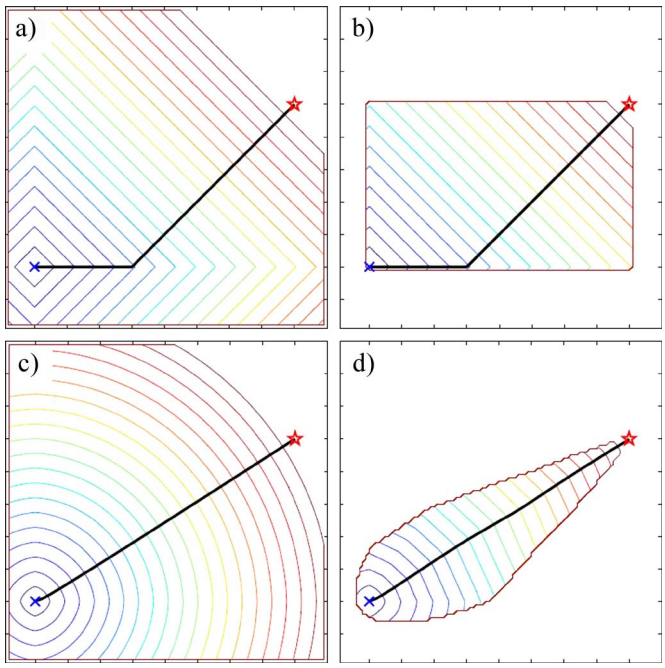

Fig. 2. Examples of distance maps and paths computed over a constant 100×100 cost map ($\tau = 1$) using a 4-connectivity. a) BF. b) A* using the "Manhattan distance" h_4 as a heuristic. c) FM. d) FM*. Paths computed using FM-based algorithms are continuous. FM* exploration is well focused towards the goal point because of the natural choice of the Euclidean heuristic h_e in this case.

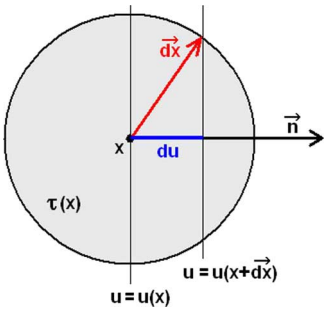

Fig. 3. On a small surface $d\Omega$ around a point x with a radius dx, one can approximate the distance function u as a plane wave, for which the level sets are parallel and perpendicular to the gradient ∇u of u. These observations allow the translation of the functional minimization problem (3) into the Eikonal equation (5).

these paths are post optimized (see [20] for a survey of trajectory-optimization techniques), they can still cause the vehicle to execute expensive trajectories.

Second, BF and HS algorithms are indeed efficient ($O(N\log(N))$ complexity) but they suffer from biases. Results from these discrete-search algorithms can be improved by taking a larger neighborhood for the motion model, like an 8- or 16-connectivity motion model, giving better approximations ρ_d of ρ in diagonal directions. However, there will always be an error in some directions that will be invariant to the grid resolution. These limitations can be overcome using the FM algorithm, presented in the next section.

III. FM-Based Path Planning

The FM algorithm is a level-set method introduced in image processing by Sethian [21]. In path planning, the main interest (developed in [22] and [23]) of the FM algorithm relies on the fact that, with the same complexity as classical grid-search algorithms, it extracts an accurate solution u of the functional minimization problem of (3).

A. Eikonal Equation

Before introducing the FM algorithm itself, we start from the observation that the functional minimization problem of (3) is equivalent to solving the Eikonal equation

$$\|\nabla u\| = \tau. \tag{5}$$

We give here a geometrical intuition of how to convert (3) into (5). It is inspired by a level set formulation of the Eikonal equation in [24]; nonetheless, a formal proof can be found in [25].

We start from the fact that the gradient ∇u of u is normal to its level sets. Let $\vec{n} = \nabla u/\|\nabla u\|$ be the outwards unit normal vector to level sets of u located in x. Express a variation du of u according to a variation $\vec{dx}$ of the position x (Fig. 3)

$$u(x+\vec{dx}) = u(x) + \langle \nabla u, \vec{dx}\rangle \Rightarrow du(x) = \langle \nabla u, \vec{dx}\rangle \tag{6}$$

where $\langle , \rangle$ is the standard dot product in $\mathbb{R}^2$.

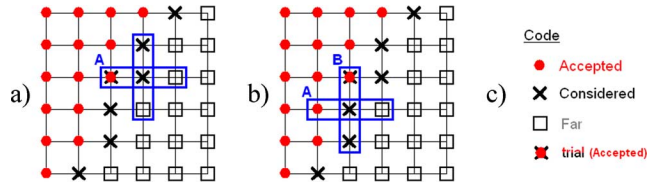

Fig. 4. To update a neighbor x_k (which is *Far* or *Considered*, but not *Accepted*) of the trial point, we examine its 4-connectivity neighborhood. a) Only one *Accepted* point around x_k, then we apply Case 1. b) Two nonopposite *Accepted* points around x_k, then we apply Case 2.

Within the small region $d\Omega$ of Ω centered on x with a radius dx, we can assimilate τ as a constant $\forall p \in d\Omega$, $\tau(p) = \tau(x) = \tau$. Within $d\Omega$ level sets of u are seen as straight lines

$$du(x) = \tau\langle \vec{n}, \vec{dx}\rangle. \tag{7}$$

From (6) and (7), we get $\langle \nabla u, \vec{dx}\rangle = \tau\langle \nabla u, \vec{dx}\rangle/\|\nabla u\|$, which leads to the Eikonal equation (5).

B. Upwind Schemes and Numerical Approximations

The FM algorithm uses a first-order numerical approximation of the Eikonal equation (5) based on the following operators. Suppose a function u is given with values $u_{i,j} = u(x_{i,j})$ on a Cartesian grid with grid spacing h.

- Forward operator (direction x): $D^{+x}_{i,j} = (u_{i+1,j} - u_{i,j})/h$.
- Backward operator (direction x): $D^{-x}_{i,j} = (u_{i,j} - u_{i-1,j})/h$.

Forward and backward operators in direction y are similar.

The following upwind scheme (due to Godunov [26]) is used to estimate the gradient ∇u in two dimensions

$$\begin{aligned}\|\nabla u_{i,j}\|^2 \approx &\left[\max\left\{D^{-x}_{i,j}u, -D^{+x}_{i,j}u, 0\right\}^2\right.\\ &\left.+\max\left\{D^{-y}_{i,j}u, -D^{+y}_{i,j}u, 0\right\}^2\right]\\ =&\,\tau^2_{i,j}\end{aligned} \tag{8}$$

where $\tau_{i,j} = \tau(x_{i,j})$.

It is proven in [21] that this numerical scheme converges to a correct continuous solution.[1]

C. FM Method

The FM algorithm belongs to the class of BF algorithms. At each iteration, the trial configuration to be moved from *Considered* to *Accepted* is the one with the lowest estimate of the distance function u. Contrary to classical BF algorithms, the trial's neighbors may be updated more than once using the numerical scheme (8).

1) Algorithm for 2-D Isotropic FM: We give here an insight of the update procedure of the *Far* or *Considered* trial's neighbors $\{x_k\}$, because we slightly modify it in our anisotropic version in Section IV. For more implementation details, see [21] and [24]. One or two *Accepted* points are used to solve (8), see Fig. 4.

[1]This numerical scheme actually converges to the viscosity solution in the sense of Crandall and Lions [27].

- Case 1: only one *Accepted* point A or one pair $\{A_1, A_2\}$ of opposite *Accepted* points around x_k [see Fig. 4(a)]. Note $u_A = u(A)$ or $u_A = \min\{u(A_1), u(A_2)\}$. In this case, (8) is equivalent to

$$(u_k - u_A)^2 = \tau_k^2$$

which leads to

$$u_k = u_A + \tau_k.$$

- Case 2: at least two nonopposite *Accepted* points around x_k belong to two crossing pairs $A = \{A_1, A_2\}$ and $B = \{B_1, B_2\}$ [see Fig. 4(b)]. Note $u_A = \min\{u(A_1), u(A_2)\}$ and $u_B = \min\{u(B_1), u(B_2)\}$. In this case, (8) is equivalent to

$$(u_k - u_A)^2 + (u_k - u_B)^2 = \tau_k^2. \tag{9}$$

Based on the discriminant test of (9), one or two *Accepted* points are used to solve it: if $\tau_k > |u_A - u_B|$

$$u_k = \frac{1}{2}\left(u_A + u_B + \sqrt{2\tau_k^2 - (u_A - u_B)^2}\right)$$

else

$$u_k = \min(u_A, u_B) + \tau_k.$$

2) FM:* The original FM algorithm uses the same priority assignment as BF algorithms. We can introduce the Euclidean heuristic defined in (4) to speed up the exploration process. We call this heuristically guided FM "FM*" because it can be seen as the equivalent to the A* algorithm in the continuous domain.

Note that if the cost function $\tau(x)$ is the inverse of the AUV velocity, then the distance function is equivalent to the first arrival travel time t of the vehicle. Using a time-dependent cost function $\tau(x, t)$ makes the FM* easy to extend for planning paths among moving obstacles.

3) Backtracking: The FM algorithm computes a first-order estimate of the gradient of the distance function. The optimal path is naturally extracted from the goal to the start configuration by performing a gradient descent backtracking.

D. Isotropic FM Applied to Underwater Path Planning

The isotropic FM and the FM* algorithms are validated here using simulated and real underwater environments.

1) Nonconvex Obstacles: Nonconvex obstacles are a problem for embedded path-planning algorithms using potential field methods [12]. Figs. 5 and 6 show that the FM algorithm associated with a gradient descent naturally deals with the concavity, since the start configuration is the only global minimum of the distance function u.

Note that to increase the safety distance between the path and the C-obstacles, a simple dilatation of the cost function is required.

2) Complex C-Spaces: The computational complexity of FM-based path planning is $(O(N \log N)$, where N is the number of pixels in the image. This efficiency is not affected by the complexity (i.e., number and shapes of C-obstacles) of the C-space. Fig. 7 shows a harbor simulation in which the main entrance is obstructed by a net. FM exploration naturally finds the little back entrance, contrary to probabilistic path-planning methods, which may not return any solution to this "narrow passage" problem [28].

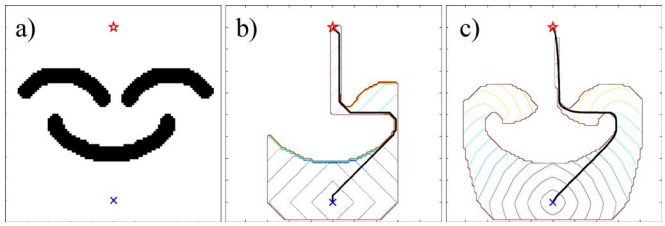

Fig. 5. a) 100×100 binary cost map. b) Exploration and optimal path found using A* algorithm. c) Exploration and optimal path found using FM* algorithm.

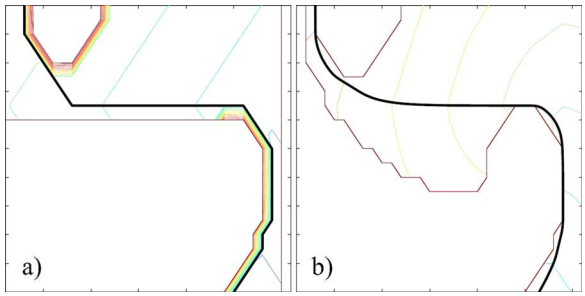

Fig. 6. Close-up of Fig. 5(b) and (c). One can see that FM* algorithm gives a smoother solution (b) than the A* algorithm (a). This is thanks to the better FM* estimate of the distance function in the continuous domain.

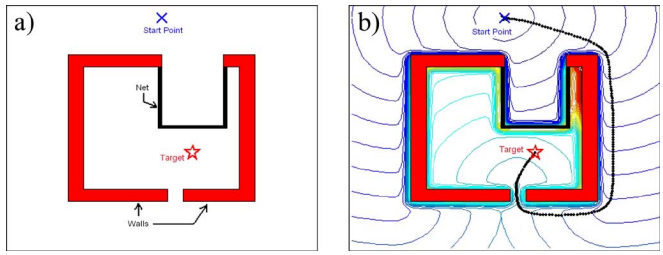

Fig. 7. 100×100 simulated cost map of a harbor obstructed by a net (a) and the optimal path found using the FM algorithm (b).

3) Real Environment: A C++ implementation of the 2-D FM-based path-planning method has been applied to real sonar images, see Fig. 8. These images have been processed inline in the simulator [23] of the Ocean Systems Laboratory.

IV. Directional Constrained Path Planning

The FM method is a particular case of *ordered upwind methods* (OUMs), which can be applied to isotropic or anisotropic media. It is shown in [29] that the computational complexity of OUMs is $O(\Upsilon N \log N)$, where N is the number of configurations, and Υ is the anisotropic coefficient which can be much greater than one. In this case, OUMs can no longer be considered fast.

The idea behind our anisotropic FM method is to simplify OUMs by considering the gradient ∇u as a good estimate of the wavefront propagation of the distance function u. This is equivalent to assuming that the field of force $\mathcal{F}$ is smooth. This

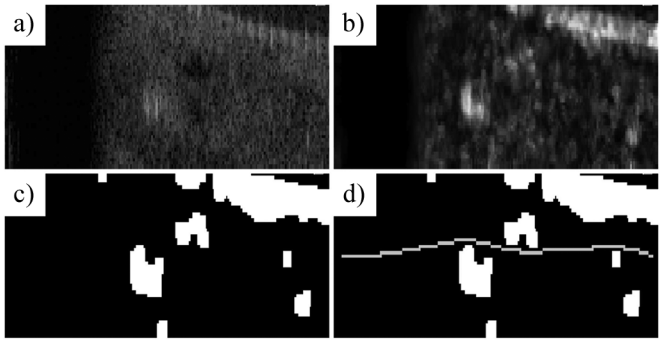

Fig. 8. a) Original 200×50 sonar image. b) After processing using a salient filter. c) After thresholding. d) Optimal path. Computation time is approximately 10 ms for the FM algorithm and 1 ms for the gradient descent algorithm.

way, the $O(N \log N)$ complexity of the original FM algorithm is preserved in the anisotropic version. One can note a similar attempt in [30], but the authors do not analyze the complexity of their method.

A. Anisotropic Cost Function

The original FM algorithm is fast thanks to the resolution of the simple quadratic equation (8) for u. Since τ appears as a square in this equation, our idea is to build a new cost function $\tilde{\tau}$ linearly dependent on u. We split the cost function in two parts: $\tilde{\tau} = \tau_{\text{obst}} + \tau_{\text{vect}}$. τ_{obst} is linked to the obstacles as previously, and τ_{vect} is defined as follows:

$$\forall x_{i,j} \in \Omega, \tau_{\text{vect}}(i,j) = \alpha \left(1 - \frac{\langle \nabla u_{i,j}, \vec{F_{i,j}} \rangle}{Q_{i,j}}\right) \geq 0 \quad (10)$$

where $Q_{i,j} = (\tau_{\text{obst}}(i,j) + 2\alpha) \sup_\Omega\{||\vec{F}||\}$ is a normalization term, so that $|\langle \nabla u, \vec{F} \rangle / Q| \leq 1$, and α is a positive gain.

τ_{vect} models the external current forces applied to the vehicle. It is equivalent to saying that a force favors the vehicle when both force and vehicle are pointing in the same direction.

Note that the concept can be generalized to path planning in any field of forces, like path planning for sailing applications, where $\mathcal{F}$ is a wind field, for instance.

B. Anisotropic FM Method

The anisotropic version of the FM algorithm is similar to the isotropic version. A new quadratic upwind scheme is used to estimate ∇u with the new cost function $\tilde{\tau}$

$$\begin{aligned} ||\nabla u_{i,j}||^2 \approx & \left[\max\left\{D_{i,j}^{-x}u, -D_{i,j}^{+x}u, 0\right\}^2 \right. \\ & \left. + \max\left\{D_{i,j}^{-y}u, -D_{i,j}^{+y}u, 0\right\}^2\right] \\ = & \tilde{\tau}_{i,j}^2. \end{aligned} \quad (11)$$

1) Algorithm for 2-D Anisotropic FM: The trial's neighbors $\{x_k\}$ are updated in 4-connectivity (see Fig. 4). Let $\{A_i\}$ be the *Accepted* points around x_k, and let $\vec{F_k} = F_x \vec{u_x} + F_y \vec{u_y}$ be the vectorial force in x_k.

- Case 1: one *Accepted* point
  First, suppose that x_k has only one adjacent *Accepted* point, $\{A_i\} = \{A\}$. (11) becomes $(u_k - u(A))^2 = \tilde{\tau}_k^2$.
  In this case, $\nabla u_k = \pm(u_k - u(A))\overrightarrow{u_{x \text{ or } y}}$, then
  $$u_k = u(A) + \frac{\tau_{\text{obst}}(k) + \alpha}{1 + \frac{\alpha}{Q_k} F_{x \text{ or } y}}.$$
- Case 2: two consecutive *Accepted* points
  Suppose that x_k has two consecutive adjacent *Accepted* points, $\{A_i\} = \{A_1, A_2\}$. (11) becomes $(u_k - u(A_1))^2 + (u_k - u(A_2))^2 = \tilde{\tau}_k^2$, which leads to the following conditions: if $|(u(A_2) - Lb) - (u(A_1) - La)| \leq L\sqrt{U}$
  $$u_k = \frac{-V + \sqrt{\Delta}}{2U}$$
  else $u_k = \infty$, where
  $$\begin{cases} a = \frac{\alpha}{Q} F_x \\ b = \frac{\alpha}{Q} F_y \\ \lambda = \tau_{\text{obst}}(k) + \alpha \\ L = \frac{\lambda}{1-(a^2+b^2)} > 0 \\ \Delta = \frac{4\lambda}{L}\left[UL^2 - [(u(A_2) - Lb) - (u(A_1) - La)]^2\right] \\ U = 2 - (a+b)^2 > 0 \\ V = 2u(A_1)(a^2 + ab - 1) + 2u(A_2)(b^2 + ab - 1) + 2\lambda(a+b). \end{cases}$$
  This is a generalized formulation of the isotropic FM equations. Conditions about $u(A_1)$ and $u(A_2)$ are, respectively, translated by La and Lb (related to F_x and F_y).

2) Computation of the Distance Function: For two or more *Accepted* points around a trial's neighbor x_k, the computation of u_k needs three steps.

- We apply Case 1 for each adjacent *Accepted* point and we store the values $\{u_{k_i}\}$.
- We apply Case 2 for each couple of two consecutive adjacent *Accepted* points and we store the values $\{u_{k_j}\}$.
- Finally, $u_k = \min\{\{u_{k_i}\}, \{u_{k_j}\}\}$ gives the correct viscosity solution for the distance function in x_k.

C. Limitation and Results

Note that we define $\tau_{\text{vect}} = \alpha(1 - (\langle \nabla u, \vec{F} \rangle / Q))$ for τ_{vect} to be linear for u. This choice allows fast computation, but most underwater vehicles have a more complex behavior than a linear reaction to currents. In these cases, an acceptable first-order approximation of the AUV behavior must be found.

Figs. 9 and 10 show the influence of fields of force in underwater path planning ($\alpha = 1$, $\sup_\Omega\{||\vec{F}||\} = 1$). The gain on the total cost of the paths computed using the anisotropic FM algorithm is about 10%, compared with the paths computed using isotropic FM for both figures. Processing time on these 100×100 figures is less than 1 s.

V. Curvature-Constrained Path Planning

In this section, the influence of the cost function τ on the smoothness of a path C is demonstrated. Here C is the solution of the functional minimization problem

$$\begin{aligned} \Omega^2 &\to \mathcal{C}(\Omega) \\ (x, x') &\mapsto C = \text{argmin}_{\{C(x,x')\}} \rho(x, x') \end{aligned} \quad (12)$$

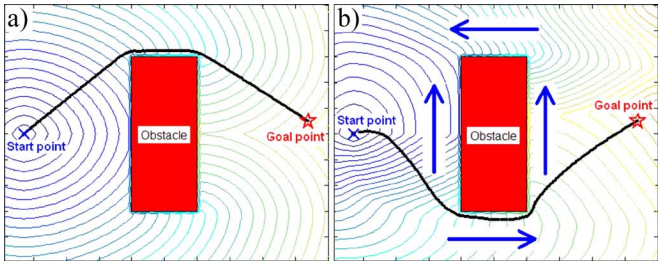

Fig. 9. a) Isotropic FM. b) Anisotropic version with currents symbolized with arrows. The path computed using the anisotropic FM leads to a 10% gain over the total cost of the trajectory.

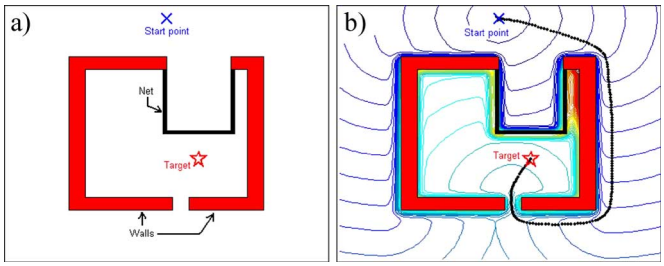

Fig. 10. A path found: a) in absence of currents, b) in presence of currents. The path computed using the anisotropic FM leads to a 8% gain over the total cost of the trajectory in this case.

where $\mathcal{C}(\Omega)$ is the set of all curves in Ω, and ρ is the continuous metric

$$\rho(x,x') = \int_{[0,1]} \tau\left(C_{x,x'}(s)\right) ds. \tag{13}$$

Both isotropic and anisotropic FM methods compute a continuous solution C associated with the continuous metric ρ. Therefore, tools from differential geometry can be used to examine the curvature properties of C.

A. Problem Statement

Let us define the kinematic constraints considered here.

- Curvature magnitude of a curve C: $k(C) = \partial^2 C/\partial^2 s$.
- Curvature radius of a curve C: $R(C) = 1/|k(C)|$.
- Lower bound on the curvature radius along a curve C: $R_{\min}(C) = \inf_{s\in[0,1]} R(C(s))$.
- Turning radius of a vehicle v: $r(v)$.

Our goal in this section is to express a formal link between the cost function τ and the lower bound $R_{\min}$. In other words, we want to find a lower bound $R_{\min}(\tau)$ before computing the distance function u (and before extracting any path).

B. Lower Bound on the Curvature Radius

1) Isotropic Case: Using the differential geometry framework, it is shown in [31] that the Euler-Lagrange equation associated with the functional minimization (12) is

$$\tau k\vec{N} - \langle \nabla\tau, \vec{N}\rangle \vec{N} = 0 \tag{14}$$

where $\vec{N}$ is the normal unit vector to a curve C and $\langle , \rangle$ is the standard dot product in $\mathbb{R}^2$.

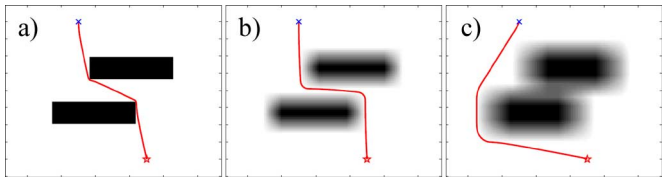

Fig. 11. Influence of smoothing the cost function. a) Binary 100×100 cost function τ (τ(C-free) $=1$ and τ(C-obst) $=11$) and the related optimal path C_a, $R_{\min}(C_a) = 332$ (in arbitrary unit). b) τ after smoothing using an 11×11 average filter, $R_{\min}(C_b) = 1216$. c) τ after smoothing using a 21×21 average filter, $R_{\min}(C_c) = 1377$.

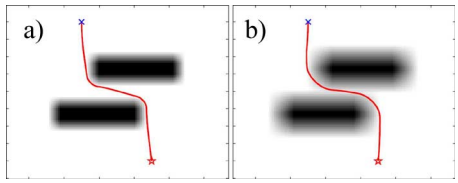

Fig. 12. Influence of both smoothing and adding an offset. The original cost function τ is similar to the one in Fig. 11(a). a) Offset $=5$, average filter 7×7, $R_{\min}(C_a) = 1977$. b) Offset $=5$, average filter 15×15, $R_{\min}(C_b) = 2787$.

From (14), it is deduced in [24] that the curvature magnitude k is bounded along any path C minimizing ρ. The lower bound $R_{\min}$ is then

$$R_{\min} \geq \frac{\inf_\Omega \tau}{\sup_\Omega \{||\nabla\tau||\}}. \tag{15}$$

The conclusion is that to increase the lower bound on the curvature radius $R_{\min}(C)$ of an optimal path C, two choices are possible (see Figs. 11 and 12):

1) smoothing the cost function to decrease $\sup_\Omega\{||\nabla\tau||\}$;
2) adding an offset to the cost function to increase the numerator $\inf_\Omega \tau$ without affecting the denominator.

2) Anisotropic Case: With our anisotropic cost function $\tilde{\tau}$, we show in the Appendix that the Euler–Lagrange equation becomes

$$\langle \nabla\tau_{\text{obst}}, \vec{N}\rangle \vec{N} + \frac{\alpha}{Q}\left(\frac{\partial F_x}{\partial y} - \frac{\partial F_y}{\partial x}\right)\vec{N} - \tilde{\tau} k \vec{N} = 0. \tag{16}$$

From (16), we derive the following lower bound on the path curvature radius:

$$R_{\min} \geq \frac{\inf_\Omega\{\tau_{\text{obst}}\}}{\sup_\Omega\{||\nabla\tau_{\text{obst}}||\} + \frac{2\alpha}{\inf_\Omega\{Q\}}||J_F||_\infty} \tag{17}$$

where J_F is the Jacobian of $\vec{F}$ on Ω, and $||\cdot||_\infty$ is the L_∞ norm.

Similar to the isotropic case, there are three parameters to tune in order to increase the lower bound on the curvature radius $R_{min}(C)$ of an optimal path C:

1) smoothing τ_{obst};
2) adding an offset to $\tilde{\tau}$;
3) or smoothing the field of force $\vec{F}$ to decrease $||J_F||_\infty$.

VI. Multiresolution Path Planning

Multiresolution methods start with the idea that it is not necessary to represent the C-space in a uniform way. Some regions

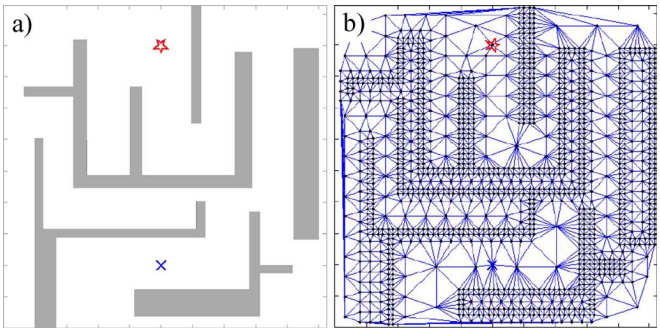

Fig. 13. a) Original 1000×1000 image. b) The mesh computed from the quadtree decomposition using a Delaunay triangulation (1400 vertices). The links between the vertices of this mesh and their neighbors are well defined, so that the propagation of the distance function from one vertex to its neighborhood is straightforward.

may be of more interest than others from a path-planning point of view. For example, a vast empty region may not need to be described as precisely as a region which contains many C-obstacles. This leads us to consider the partitioning of the environment as a crucial issue to optimize a path-planning method.

A. Unstructured Mesh Framework

The method proposed by the authors is to couple a quadtree decomposition of the C-space with a Delaunay triangulation.

1) Quadtree Decomposition: The quadtree decomposition (or octree decomposition in more than two dimensions) is based on a recursive decomposition of a uniform grid into blocks. The size of blocks can depend either on the information in them [32] or on their distance from the robot [33].

Quadtrees allow efficient partitioning of the environment, since single blocks can be used to encode large empty regions. However, two main drawbacks remain. First, paths generated by quadtrees are suboptimal because they are constrained to segments between centers of blocks. The framed-quadtree technique [34] improves the situation, but it is only applicable for sparse environments. Second, since the initial C-space is transformed in a tree data structure, it is not easy to define the spatial neighborhood of each block. A simple solution is to test all the neighbors of each block [35].

2) Delaunay Triangulation: The method proposed by the authors is to couple the quadtree decomposition with an adaptive mesh generation. The Delaunay triangulation is a good candidate as fast and robust implementations exist.

The input of the Delaunay mesh generation is the set of nodes q with their cost $\tau(q)$ given by the quadtree decomposition. The output is a net of vertices linked to their neighbors by edges, see Fig. 13. Versions of BF and FM algorithms on this kind of unstructured mesh [36] are implemented in the following section.

B. FM Algorithm on Unstructured Meshes

The original FM algorithm computes a distance function over a C-space sampled in a rectangular orthogonal mesh Ω in $O(N_{\text{points}} \log N_{\text{points}})$ steps, where N_{points} is the total number of grid points. The idea in this section is to perform the FM algorithm over the same C-space partitioned into a mesh using far fewer nodes. It is shown in [36] that an extension of the FM

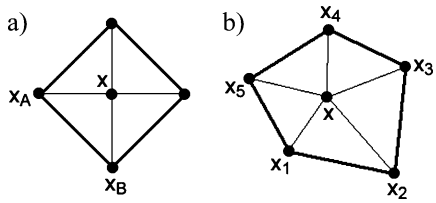

Fig. 14. Original FM algorithm is implemented on a Cartesian grid (a), the update for the point x is computed from the simplex xx_Ax_B with the smallest values u_A and u_B. When implemented on an unstructured mesh (b), the update for the point x is computed from a pair of adjacent neighbors.

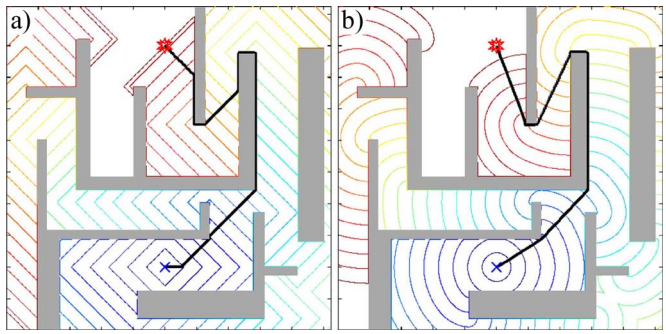

Fig. 15. Paths computed on the regular 1000×1000 grid on the original image (runtime is about 100 s). a) Using a 4-connectivity BF. b) Using a 4-connectivity FM.

algorithm on unstructured meshes with N_{nodes} nodes has the same $O(N_{\text{nodes}} \log N_{\text{nodes}})$ complexity.

Because there is no natural choice of the coordinate system for an unstructured mesh (see Fig. 14), for each simplex $xx_1, \ldots, x_n$ we compute the gradient ∇u as a linear combination of the n directional derivatives along the unit vectors $T_i = (x - x_i)/\|x - x_i\|$, for $i = 1, \ldots, n$.

Let T be the nonsingular matrix having vectors T_i as its columns. Define $a = 1/\|x - x_i\|$, $b = -(u_i/\|x - x_i\|)$, and $Q = (T^{\mathrm{T}}T)^{-1}$. Eikonal equation (5) becomes the following quadratic equation to solve for u (see [36] for further details):

$$(a^T Qa)u^2 + (2a^T Qb)u + (b^T Qb) = \tau^2. \qquad (18)$$

C. Results

Fig. 15 depicts the solution computed with BF and FM algorithms on the grid of Fig. 13(a). Fig. 16 depicts the solution computed with BF and FM algorithms on the mesh of Fig. 13(b).

On the scene of Fig. 15, both BF- and FM-based methods implemented on the mesh (including the mesh generation) are approximately 1000 times faster than BF and FM algorithms implemented on the grid.

However, the multiresolution method gives only suboptimal paths. We partially overcome the problem of suboptimal paths by interpolating the distance function (computed on mesh vertices) on the entire underlying grid and by performing a gradient descent backtracking on this grid.

Surprisingly, BF gives similar-looking paths to FM on the mesh. One could expect a better behavior of FM on the mesh, thanks to its first-order estimate of the distance function over the mesh. Further development would benefit from a tool designed

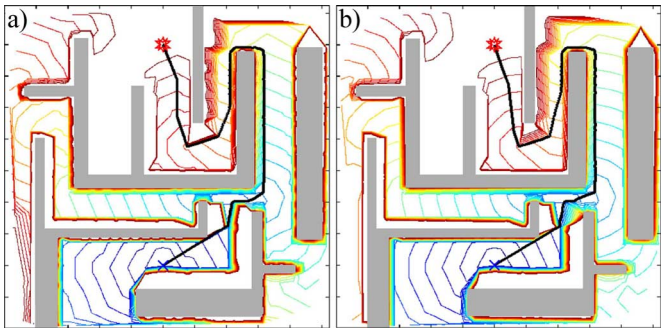

Fig. 16. Paths computed on the 1400 vertices mesh. a) Using BF. b) Using FM algorithm adapted on meshes. Both paths are suboptimal, but they are computed in 0.1 s.

to compare the quality of the paths. To the knowledge of the authors, no general criteria exist in the literature to objectively estimate the acceptability of a path for a given vehicle.

VII. Conclusion and Future Work

The underwater world is a very demanding environment for path-planning algorithms. Great efforts are currently being made to develop autonomous systems as underwater technology becomes more mature. Several key issues for underwater path planning have been addressed in this paper.

Reliability of path planners has been improved by developing an efficient algorithm called FM*, which is a continuous version of A* based on the FM algorithm. First, this algorithm has been proven to find a continuous path when it is implemented on a discretized perception of the world. Secondly, a practical implementation of anisotropic FM has been proposed to adapt our path-planning method to underwater currents. Thirdly, novel work has been presented to take the vehicle kinematics into account for both isotropic and anisotropic environments. It has been shown that smoothing the map of the environment leads to greater path curvature radius. Finally, a mesh conversion of the input data has been used to drastically reduce the computation load by reducing the input data set of the path-search algorithm.

It is noted that the data reduction produces a loss of information that can affect the optimality of the resulting path. This highlights the need for future work to find an analytical tool to measure path acceptability.

In all this work, we assume that the world is static and can be described *a priori* in a cost function. The authors are currently developing a dynamic version of the FM* in order to improve its replanning capacity in *a priori* unknown real environments.

Appendix
Proof of the Upper Bound of the Curvature Magnitude for the Anisotropic Case

Given two points x and x' in a domain $\Omega \subset \mathbb{R}^2$, we consider the functional

$$E(C) = \int_{[0,1]} \tilde{\tau}\left(C(s)\right) ds \tag{19}$$

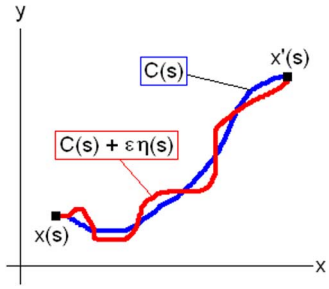

Fig. 17. Curve η is a small deformation of the curve C minimizing E.

to minimize over the following set of curves:

$$C \in \mathcal{C}^2 : \quad \begin{array}{l} [0,1] \to \Omega \\ s \mapsto [x(s), y(s)] \end{array}$$

where $\tilde{\tau} = \tau_{\text{obst}} + \tau_{\text{vect}} = \tau_{\text{obst}} + \alpha(1 - (\langle \nabla C, \vec{F} \rangle / Q))$ is the anisotropic strictly positive cost function.[2] We denote $\mathcal{C}^2$ as the set of curves derivable twice, and s as the arc-length parameter of C between x and x' ($C(0) = x$ and $C(1) = x'$).

We want to show that the Euler–Lagrange equation of this functional minimization problem is

$$\langle \nabla \tau_{\text{obst}}, \vec{N} \rangle \vec{N} + \frac{\alpha}{Q} \left(\frac{\partial F_x}{\partial y} - \frac{\partial F_y}{\partial x} \right) \vec{N} - \tilde{\tau} k \vec{N} = 0$$

where $\vec{N}$ is the normal unit vector to a curve C.

One can rewrite $E(C) = \int_{[0,1]} \tilde{\tau}(C(s)) \|C'(s)\| ds$. Note that it is always possible to parameterize a curve $C \in \mathcal{C}^2$ so that $\forall s \in [0,1]$, $\|C'(s)\| = \|(dC/ds)(s)\| = 1$.

We consider the curve C, which minimizes E. Let $\eta \in \mathcal{C}^2$ be a small deformation of C with the limit conditions $\eta(x) = \eta(x') = 0$ (Fig. 17).

We define $\phi(\epsilon) = E(C + \epsilon\eta)$, where $\epsilon > 0$. Then $d\phi/d\epsilon = 0$ when ϵ tends to 0. We suppose τ derivable on Ω, then

$$\frac{d\phi}{d\epsilon} = \int_{[0,1]} \frac{d}{d\epsilon} \left[\tilde{\tau}(C + \epsilon\eta) \|C' + \epsilon\eta'\| \right] ds.$$

First

$$\frac{d}{d\epsilon} \left[\tau_{\text{obst}}(C + \epsilon\eta) + \alpha \left(1 - \frac{\left\langle (C' + \epsilon\eta'), \vec{F}(C + \epsilon\eta) \right\rangle}{Q} \right) \right] = \langle \nabla \tau_{\text{obst}}, \vec{\eta} \rangle - \frac{\alpha}{Q} \frac{d}{d\epsilon} \left\langle C' + \epsilon\eta', \vec{F}(C + \epsilon\eta) \right\rangle.$$

Second

$$\lim_{\epsilon \to 0} \left\{ \frac{d}{d\epsilon} \left\langle C' + \epsilon\eta', \vec{F}(C + \epsilon\eta) \right\rangle \right\} = \langle \vec{F}, \vec{\eta'} \rangle + (\vec{T}^{\mathrm{T}} \cdot J_F) \cdot \vec{\eta}$$

[2]Note that we replaced ∇u of (10) by $\nabla C = \vec{T}$, because C is the curve found after a gradient descent on u.

where $\vec{T}^{\mathrm{T}}$ is the transpose of the tangent vector to C, and J_F is the Jacobian of $\vec{F}$ along the x and y coordinates.

Third

$$\lim_{\epsilon\to 0}\left\{\frac{d\phi}{d\epsilon}\right\} = \int_{[0,1]} \langle \nabla\tau_{\mathrm{obst}}, \vec{\eta}\rangle - \frac{\alpha}{Q}\langle \vec{F}, \vec{\eta'}\rangle - \frac{\alpha}{Q}(\vec{T}^{\mathrm{T}}\cdot J_F)\cdot\vec{\eta} + \tilde{\tau}\langle \vec{T}, \vec{\eta'}\rangle ds.$$

We integrate by part to get

$$\int_{[0,1]} \tilde{\tau}\langle \vec{T}, \vec{\eta'}\rangle ds = -\int_{[0,1]} \tilde{\tau} k \langle \vec{N}, \vec{\eta}\rangle ds$$

and to get

$$\int_{[0,1]} \langle \vec{F}, \vec{\eta'}\rangle ds = -\int_{[0,1]} \left[(J_F\cdot\vec{T})^{\mathrm{T}}\cdot\vec{\eta}\right] ds.$$

Hence

$$\lim_{\epsilon\to 0}\left\{\frac{d\phi}{d\epsilon}\right\} = \int_{[0,1]} \left\langle \left(\nabla\tau_{\mathrm{obst}} + \frac{\alpha}{Q}\left(J_F^{\mathrm{T}} - J_F\right)^{\mathrm{T}}\cdot\vec{T} - \tilde{\tau}k\vec{N}\right), \vec{\eta}\right\rangle ds.$$

Noticing that $\forall\vec{\eta}$, $d\phi/d\epsilon = 0$, and that

$$\left(J_F^{\mathrm{T}} - J_F\right)^{\mathrm{T}}\cdot\vec{T} = \left(\frac{\partial F_x}{\partial y} - \frac{\partial F_y}{\partial x}\right)\vec{N}$$

we get the Euler–Lagrange equation

$$\langle \nabla\tau_{\mathrm{obst}}, \vec{N}\rangle\vec{N} + \frac{\alpha}{Q}\left(\frac{\partial F_x}{\partial y} - \frac{\partial F_y}{\partial x}\right)\vec{N} - \tilde{\tau}k\vec{N} = 0. \quad (20)$$

Finally, we derive an upper bound of the curvature magnitude

$$|k| \le \frac{\sup_\Omega\{\|\nabla\tau_{\mathrm{obst}}\|\} + \frac{2\alpha}{\inf_\Omega\{Q\}}\|J_F\|_\infty}{\inf_\Omega\{\tau_{\mathrm{obst}}\}}. \quad (21)$$

REFERENCES

[1] R. Philippsen and R. Siegwart, "An interpolated dynamic navigation function," in *Proc. IEEE Conf. Robot. Autom.*, 2005, pp. 3782–3789.

[2] D. Ferguson and A. Stentz, "Using interpolation to improve path planning: The Field D* algorithm," *J. Field Robot.*, vol. 23, no. 2, pp. 79–101, 2006.

[3] A. Stentz, "Optimal and efficient path planning for partially-known environment," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 1994, pp. 3310–3317.

[4] K. P. Carroll, S. R. McClaran, E. L. Nelson, D. M. Barnett, D. K. Friesen, and G. N. Williams, "AUV path planning: An A* approach," in *Proc. Symp. AUV Technol.*, 1992, pp. 3–8.

[5] C. W. Warren, "A technique for autonomous underwater vehicle route planning," *IEEE J. Ocean. Eng.*, vol. 15, no. 3, pp. 199–204, Jul. 1990.

[6] Y. Petillot, I. T. Ruiz, and D. Lane, "Underwater vehicle obstacle avoidance and path planning using a multi-beam forward looking sonar," *IEEE J. Ocean. Eng.*, vol. 26, no. 2, pp. 240–251, Apr. 2001.

[7] C. Vasudevan and K. Ganesan, "Case-based path planning for autonomous underwater vehicles," *Auton. Robots*, vol. 3, no. 2-3, pp. 79–89, 1996.

[8] K. Sugihara and J. Yuh, "GA-based motion planning for underwater robotic vehicle," in *Proc. 10th Int. Symp. Unmanned Untethered Submersible Technol.*, 1997, pp. 406–415.

[9] R. Fox, A. Garcia, and M. L. Nelson, "A three dimensional path planning algorithm for autonomous vehicles," in *Proc. 11th Int. Symp. Unmanned Untethered Submersible Technol.*, 1999, pp. 546–556.

[10] A. Alvarez, A. Caiti, and R. Onken, "Evolutionary path planning for autonomous underwater vehicles in a variable ocean," *IEEE J. Ocean. Eng.*, vol. 29, no. 2, pp. 418–429, Apr. 2004.

[11] B. Garau, A. Alvarez, and G. Oliver, "Path planning of autonomous underwater vehicles in current fields with complex spatial variability: An A* approach," in *Proc. IEEE Conf. Robot. Autom.*, Barcelona, Spain, Apr. 2005, pp. 546–556.

[12] J.-C. Latombe, *Robot Motion Planning*. Norwell, MA: Kluwer, 1991.

[13] S. M. LaValle, *Planning Algorithms*. Urbana, IL: Univ. Illinois, 2006.

[14] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. Cambridge, MA: MIT Press, 2005.

[15] S. M. LaValle, M. S. Branicky, and S. R. Lindemann, "On the relationship between classical grid search and probabilistic roadmaps," *Int. J. Robot. Res.*, vol. 23, no. 7, pp. 673–692, Jul.–Aug. 2004.

[16] R. Kimmel, N. Kiryati, and A. Bruckstein, "Multi-valued distance maps in finding shortest paths between moving obstacles," *IEEE Trans. Robot. Autom.*, vol. 14, no. 3, pp. 427–436, Jun. 1998.

[17] P. Melchior, B. Orsoni, O. Lavialle, A. Poty, and A. Oustaloup, "Consideration of obstacle danger level in path planning using A* and fast-marching optimization: Comparative study," *Signal Process.*, vol. 83, no. 11, pp. 2387–2396, 2003.

[18] R. E. Korf, "Artificial intelligence search algorithms," in *CRC Handbook of Algorithms and Theory of Computation*. Boca Raton, FL: CRC, 1998, pp. 36-1–36-20.

[19] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. Cambridge, MA: MIT Press, 2001.

[20] J. T. Betts, "Survey of numerical methods for trajectory optimization," *J. Guid., Control, Dyn.*, vol. 21, no. 2, pp. 193–207, Mar.–Apr. 1998.

[21] J. A. Sethian, *Level Set Methods and Fast Marching Methods*. Cambridge, U.K.: Cambridge Univ. Press, 1999.

[22] C. Petres, Y. Pailhas, J. Evans, Y. Petillot, and D. Lane, "Underwater path planning using fast marching algorithms," in *Proc. Oceans Eur. Conf.*, Brest, France, Jun. 2005, vol. 2, pp. 814–819.

[23] C. Petres and P. Patron, "Path planning for unmanned underwater vehicles," in *Proc. IJCAI Workshop Planning, Learning in A Priori Unknown or Dynamic Domains*, Edinburgh, U.K., Aug. 2005, pp. 47–54.

[24] L. Cohen and R. Kimmel, "Global minimum for active contour models: A minimal path approach," *Int. J. Comput. Vis.*, vol. 24, no. 1, pp. 57–78, 1997.

[25] A. M. Bruckstein, "On shape from shading," *Comput. Vis., Graphics and Image Process.*, vol. 44, pp. 139–154, 1988.

[26] E. Rouy and A. Tourin, "A viscosity solution approach to shape-from-shading," *SIAM J. Numer. Anal.*, vol. 29, pp. 867–884, 1992.

[27] M. G. Crandall, L. C. Evans, and P. L. Lions, "Some properties of viscosity solutions of Hamilton-Jacobi equations," *Trans. Amer. Math. Soc.*, no. 282, pp. 487–502, 1984.

[28] P. Svestka and M. Overmars, "Probabilistic path planning," in *Lecture Notes in Control and Information Sciences 229*. New York: Springer, 1998.

[29] A. Vladimirsky, "Ordered upwind methods for static Hamilton-Jacobi equation: Theory and algorithms," *SIAM J. Numer. Anal.*, vol. 41, no. 1, pp. 325–363, 2003.

[30] Q. Lin, "Enhancement, extraction, and visualization of 3D volume data," Ph.D. dissertation, Linköpings Universitet, Linköping, Sweden, 2003.

[31] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic active contours," in *Proc. 5th IEEE Int. Conf. Comput. Vis.*, 1995, pp. 694–699.

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

[33] S. Behnke, "Local multiresolution path planning," in *Proc. Robot World Cup Soccer, Rescue Competitions, Conf.*, Padua, Italy, 2004, pp. 332–343.

[34] A. Yahja, A. Stentz, S. Singh, and B. Brumitt, "Framed-quadtree path planning for mobile robots operating in sparse environments," in *Proc. IEEE Conf. Robot. Autom.*, Leuven, Belgium, May 1998, vol. 1, pp. 650–655.

[35] D. Ferguson and A. Stentz, "Multi-resolution field D*," in *Proc. Int. Conf. Intell. Auton. Syst.*, 2006, pp. 67–74.

[36] J. A. Sethian and A. Vladimirsky, "Fast methods for the Eikonal and related Hamilton-Jacobi equations on ustructured meshes," *Appl. Math.*, vol. 97, no. 11, pp. 5699–5703, May 23, 2000.

Clément Pêtrès received the Engineering degree in electrical engineering from the Ecole Nationale Superieure de Physique de Strasbourg, Strasbourg, France, and the M.Sc. degree in photonics, image, and cybernetics from the Universite Louis Pasteur de Strasbourg, Strasbourg, France, in 2004. He is currently working toward the Ph.D. degree in artificial intelligence in the Ocean Systems Laboratory, Heriot-Watt University, Edinburgh, U.K.

He is particularly interested in trajectory planning for autonomous robots. He also carried out some research activities in image processing for retinal images and some consultancy in machine learning for the NATO Undersea Research Centre, La Spezia, Italy, in 2006.

Yan Pailhas received the M.Sc. degree in signal and image processing from the Ecole Nationale Supérieure de Cachan, Cachan, France, in 2003. He also received two Engineering degrees in telecommunications with a specialization in image and signal processing from the Ecole Nationale Supérieure des Telecommunications, Paris, France, and from the Politecnico di Torino, Torino, Italy.

He has been a Research Associate in the Ocean Systems Laboratory, Heriot-Watt University, Edinburgh, U.K., since 2004, where he is currently carrying out research activities in bioacoustic signals and sensors, signal processing for detection and classification, and numerical simulations.

Pedro Patrón received the "Ingeniero en Informatica" in computer science engineering from the Universidad de Oviedo, Oviedo, Spain, in 2001, and the Diplome d'Etudes Approfondies (DEA) in computer vision, image processing and robotics from the Institut National Polytechnique de Grenoble, Grenoble, France, in collaboration with the INRIA Rhone-Alpes, in 2006.

He is currently a Research Engineer in the Ocean Systems Laboratory, Heriot-Watt University, Edinburgh, U.K., in three different research projects: AUTOTRACKER (EU-Seebyte); BAUUV (MoD); and DTC (MoD). In some of these projects, his collaboration includes some consultancy time with the spin out of the laboratory, Seebyte Ltd. His interests include autonomous decision making, replanning and plan repair, obstacle avoidance, and multirobot systems for unmanned vehicles.

Mr. Patrón received an EU Erasmus Fellowship in 2006.

Yvan Petillot received the Ph.D. degree from the Universite de Bretagne Occidentale, Brest, France, in real-time pattern recognition using optical processors, the M.Sc. degree in optics and signal processing, and the Engineering degree in telecommunications with a specialization in image and signal processing.

He is currently a Lecturer at Heriot-Watt University, Edinburgh, U.K., carrying out research and teaching activities in image processing including texture and shape analysis, segmentation, and classification of sonar and video images, simultaneous mapping and localization, and navigation. He also has a particular interest in mine detection and classification using image-based statistical techniques. He is the Principal Investigator on two EPSRC Research projects (GR/S68088/01 and GR/S16980/01) on weapons detection using mm-wave imaging, and on multimodal sensor fusion for seabed classification, respectively. He is also a Co-Investigator of the Framework V AUTOTRACKER project (obstacle avoidance and acoustic pipe tracking using an AUV) and the Framework V Amason project (data fusion of sonar data products for seabed classification and habitat mapping). He is also a reviewer for IEE and IEEE journals.

Jonathan Evans received the Ph.D. in advanced self-learning navigation of autonomous robots from the University of Liverpool, Liverpool, U.K., in 1997. He also received the M.Eng. degree (Hons) in electronic systems engineering from University of York, York, U.K., in 1993.

He has been working on research and development projects for ROVs and AUVs since the mid-1990s. He is currently the Head of Engineering for SeeByte Ltd., a spin-out company productizing the technology of the Heriot-Watt University's Ocean Systems Laboratory, Edinburgh, U.K., where he is also a Senior Research Fellow.

He is a Fellow of the Society of Underwater Technology (FSUT), a Member of the Institute of Engineering & Technology (MIET) and a Chartered Engineer (C.Eng). He is a Fellow of the Society for Underwater Technology (SUT), and the current Chair of the SUT's expert group on underwater robotics.

David Lane has over 20 years experience in advanced subsea technology research and development for ROV's and AUV's. He has a broad experience of several technologies, including control/navigation, sonar/video data processing, and advanced robotics. He has led numerous multinational R&D projects at the European level, and previously worked in the U.K. Defence Industry as a Development Engineer, and in the offshore industry in the North Sea.