Fast Object Segmentation by Growing Minimal Paths from a Single Point on 2D or 3D Images

Fethallah Benmansour and Laurent D. Cohen

CEREMADE-Université Paris Dauphine,
Place du Marchal de Lattre de Tassigny, 75775 Paris Cedex 16, France
Email: benmansour, cohen @ceremade.dauphine.fr

Abstract

In this paper, we present a new method for segmenting closed contours and surfaces. Our work builds on a variant of the minimal path approach. First, an initial point on the desired contour is chosen by the user. Next, new keypoints are detected automatically using a front propagation approach. We assume that the desired object has a closed boundary. This a-priori knowledge on the topology is used to devise a relevant criterion for stopping the keypoint detection and front propagation. The final domain visited by the front will yield a band surrounding the object of interest. Linking pairs of neighboring keypoints with minimal paths allows us to extract a closed contour from a 2D image. This approach can also be used for finding an open curve giving extra information as stopping criteria. Detection of a variety of objects on real images is demonstrated. Using a similar idea, we can extract networks of minimal paths from a 3D image called Geodesic Meshing. The proposed method is applied to 3D data with promising results.

Keywords: Image Segmentation, Minimal Paths, Energy minimizing curves, Surface meshing, Object Extraction, Digital Topology, Fast Marching Method.

1 Introduction

Energy minimization techniques have been applied to a broad variety of problems in image processing and computer vision. Since the original work on snakes [12], they have notably been used for boundary detection. An active contour model, or snake, is a curve that deforms its shape in order to minimize an energy combining an internal part which smoothes the curve and an external part which guides the curve toward particular image features. For instance, the geodesic active contour model [5, 22] relies on the minimization of a geometric energy functional that deforms an initial curve toward local geodesics in a Riemannian metric derived from the image. Whereas the geodesic active contour model presents significant improvements compared to the original snake model, the energy minimization process is still prone to local minima. Consequently, results strongly depend on the model initialization.

To overcome this issue, Cohen and Kimmel [8] introduced an approach to globally minimize the geodesic active contour energy, provided that two endpoints of the curve

are initially supplied by the user. This energy is of the form $\int_\gamma \tilde{\mathcal{P}}$ where the incremental cost $\tilde{\mathcal{P}}$ is chosen to take lower values on the interesting features of the image, and γ is a path joining the two points. The solution of this minimization problem is obtained through the computation of *the minimal action map* associated to a source point. The minimal action map can be regarded as the arrival times of a front propagating from the source point with velocity $(1/\tilde{\mathcal{P}})$. It satisfies the Eikonal equation and can be numerically solved efficiently using the Fast Marching Method. Moreover, as introduced in [9] we can compute simultaneously the Euclidean path length of the minimal path for each endpoint with the *Fast Marching Method*. This technic will be detailed in section 2.2.

Still, the minimal path technique may fail if the potential is too noisy, or not enough contrasted or if the objects of interest are long thin curvy curves. In this case a portion of the minimal path can be a short cut to the starting point, leading to wrong results, like in figure 1. We propose in this paper a new approach to avoid this drawback. A first version of this method was described in [2, 4]. The idea is to detect recursively new source points (called keypoints) along the curve of interest between the two given points. These points are automatically detected using a criterion based on the Euclidian length of minimal paths and are almost equi-distributed along the curve of interest. In [9] it was proposed to use the length of the minimal path in order to find the second extremity of the path. Here we iterate this idea many times in order to obtain intermediate points along the contour. Since the front propagates faster on the features, since potential is smaller, the first point for which the length λ is reached, is located in this area (of small values of $\tilde{\mathcal{P}}$) and is a valuable choice as a keypoint. A good choice of parameter λ enables the front to propagates further in the direction of the thin curve without propagating in all directions and thus to reduce considerably the visited domain on the image.

In section 2, we give some background on minimal path, Fast Marching, and finding the Euclidian length of the minimal path. In section 3, we introduce a novel front propagation approach, based on the Fast Marching Method, to distribute a set of points on a codimension-1 manifold that is not known a priori, all starting from a single point (or more if desired) initialized on the desired object boundary. Each newly detected keypoint is immediately defined as a new source of propagation, and keypoints are detected as described before. By using the a-priori knowledge on the topology of the closed manifold, we devise a relevant criterion for stopping the keypoint detection and front propagation. For open curves, we propose in section 3.2 two different stopping criteria. The first one needs more interaction by specifying and endpoint to reach and the second one is based on an approximation of the total contour length. These criteria are general for any dimension for closed manifolds or elongated structures. In section 3.4, we explain how to extract a codimension-1 closed manifold within the image using the previous results. The main idea is to link pairs of neighboring keypoints with minimal paths via gradient descent on the minimal action map. In section 3.5, we explain briefly the influence of the path length parameter λ. In section 4, we explain how we can extend the previous ideas to 3D by using the right connectivity associated to the dimension. Segmentation results on a set of 2D and 3D images are presented. Finally, conclusions and perspectives follow in section 5.

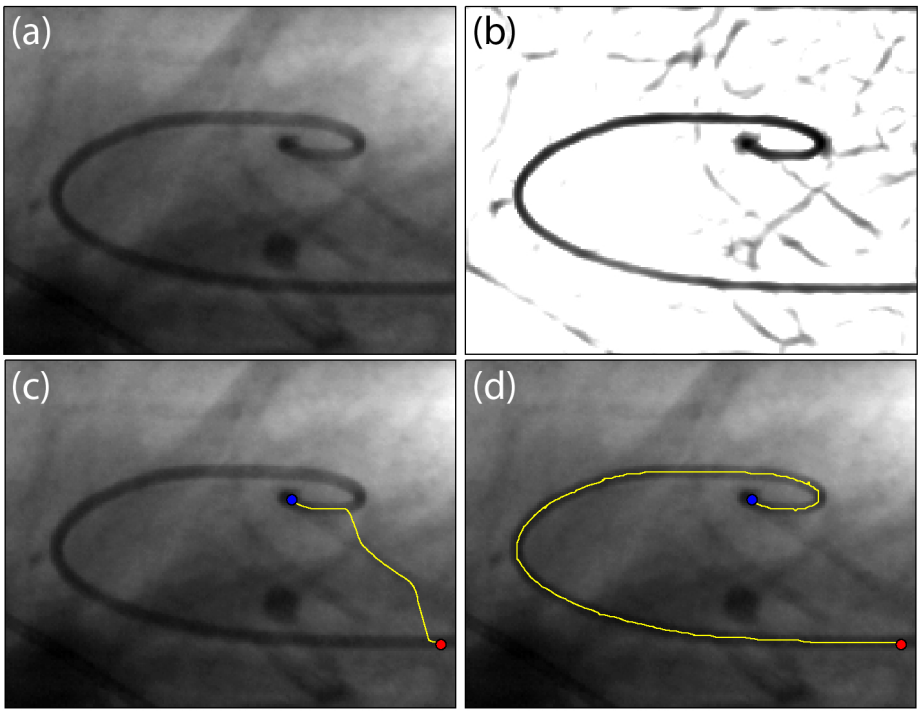

Fig. 1: (a) Original Image. (b) Build potential based on the Laplacian. (c) Short cut. (d) Fast Marching with Keypoints detection can avoid short cut.

2 Background on minimal paths

2.1 Definitions

Given a 2D image $I : \Omega \rightarrow \mathbb{R}^+$ and two points $\mathbf{p}_1$ and $\mathbf{p}_2$, the underlying idea introduced by Cohen and Kimmel [8] is to build a potential $\mathcal{P} : \Omega \rightarrow \mathbb{R}^{*+}$ which takes lower values near desired features of the image I. The choice of the potential $\mathcal{P}$ depends on the application. For example, one can define $\mathcal{P}$ as a decreasing function of $\|\nabla I\|$ to extract image edges by finding a curve that globally minimizes the energy functional $E : \mathcal{A}_{\mathbf{p}_1,\mathbf{p}_2} \rightarrow \mathbb{R}^+$

$$E(\gamma) = \int_\gamma \Big\{\mathcal{P}\big(\gamma(s)\big) + w\Big\}\mathrm{d}s = \int_\gamma \tilde{\mathcal{P}}\big(\gamma(s)\big)\mathrm{d}s, \tag{1}$$

where $\mathcal{A}_{\mathbf{p}_1,\mathbf{p}_2}$ is the set of all paths connecting $\mathbf{p}_1$ to $\mathbf{p}_2$, s is the arc-length parameter, w is a positive constant regularization term and $\tilde{\mathcal{P}} = (\mathcal{P} + w)$. A curve connecting $\mathbf{p}_1$ to $\mathbf{p}_2$ that globally minimizes the energy (1) is a *minimal path* between $\mathbf{p}_1$ and $\mathbf{p}_2$, noted $\mathcal{C}_{\mathbf{p}_1,\mathbf{p}_2}$. The solution of this minimization problem is obtained through the computation of the *minimal action map* $\mathcal{U}_1 : \Omega \rightarrow \mathbb{R}^+$ associated to $\mathbf{p}_1$. The minimal action is the

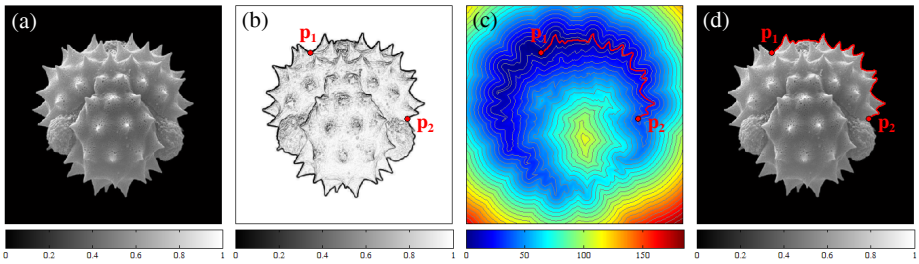

Fig. 2: Extraction of an open contour from an electron microscopy image. (**a**) Original image I. (**b**) Potential $\mathcal{P} = (\|\nabla I\| + \varepsilon)^{-3}$, where ε is a small positive constant, and user-supplied points $\mathbf{p}_1$ and $\mathbf{p}_2$. (**c**) Minimal action map $\mathcal{U}_1$ and minimal path $\mathcal{C}_{\mathbf{p}_1,\mathbf{p}_2}$ between $\mathbf{p}_1$ and $\mathbf{p}_2$. (**d**) Image I and minimal path $\mathcal{C}_{\mathbf{p}_1,\mathbf{p}_2}$.

minimal energy integrated along a path between $\mathbf{p}_1$ and any point $\mathbf{x}$ of the domain Ω :

$$\forall\, \mathbf{x} \in \Omega,\ \mathcal{U}_1(\mathbf{x}) = \min_{\gamma \in \mathcal{A}_{\mathbf{p}_1,\mathbf{x}}} \left\{ \int_\gamma \tilde{\mathcal{P}}\big(\gamma(s)\big)\mathrm{d}s \right\}. \tag{2}$$

The values of $\mathcal{U}_1$ may be regarded as the arrival times of a front propagating from the source $\mathbf{p}_1$ with velocity $(1/\tilde{\mathcal{P}})$. $\mathcal{U}_1$ satisfies the Eikonal equation

$$\begin{cases} \|\nabla \mathcal{U}_1(\mathbf{x})\| = \tilde{\mathcal{P}}(\mathbf{x}) \text{ for } \mathbf{x} \in \Omega, \\ \quad \mathcal{U}_1(\mathbf{p}_1) = 0. \end{cases} \tag{3}$$

The map $\mathcal{U}_1$ has only one local minimum, the point $\mathbf{p}_1$, and its flow lines satisfy the Euler-Lagrange equation of functional (1) (see [1]). Thus, the minimal path $\mathcal{C}_{\mathbf{p}_1,\mathbf{p}_2}$ can be retrieved with a simple gradient descent on $\mathcal{U}_1$ from $\mathbf{p}_2$ to $\mathbf{p}_1$ (see Fig. 2), solving the following ordinary differential equation with standard numerical methods like Heun's or Runge-Kutta's :

$$\begin{cases} \dfrac{\mathrm{d}\mathcal{C}_{\mathbf{p}_1,\mathbf{p}_2}}{\mathrm{d}s}(s) = -\nabla \mathcal{U}_1\big(\mathcal{C}_{\mathbf{p}_1,\mathbf{p}_2}(s)\big), \\ \mathcal{C}_{\mathbf{p}_1,\mathbf{p}_2}(0) = \mathbf{p}_2. \end{cases} \tag{4}$$

Let us extend the definitions given so far to the case of multiple sources and introduce other definitions which will be useful hereinafter. These definitions hold in dimension 2 and higher. The *minimal action map* associated to the potential $\tilde{\mathcal{P}} : \Omega \to \mathbb{R}^{*+}$ and the set of n sources $\mathcal{S} = \{\mathbf{p}_1, \ldots, \mathbf{p}_n\}$ is the function $\mathcal{U} : \Omega \to \mathbb{R}^+$ defined by

$$\begin{aligned} \forall\, \mathbf{x} \in \Omega,\ \ \mathcal{U}(\mathbf{x}) &= \min_{1 \le j \le n} \big\{\mathcal{U}_j(\mathbf{x})\big\}, \\ \text{where } \ \mathcal{U}_j(\mathbf{x}) &= \min_{\gamma \in \mathcal{A}_{\mathbf{p}_j,\mathbf{x}}} \left\{ \int_\gamma \tilde{\mathcal{P}}\big(\gamma(s)\big)\mathrm{d}s \right\}. \end{aligned} \tag{5}$$

The map $\mathcal{U}$ is a weighted distance map to the set of sources $\mathcal{S}$, and it satisfies the Eikonal equation

$$\begin{cases} \|\nabla \mathcal{U}(\mathbf{x})\| = \tilde{\mathcal{P}}(\mathbf{x}) \text{ for } \mathbf{x} \in \Omega, \\ \quad \mathcal{U}(\mathbf{p}_j) = 0 \qquad \text{for } \mathbf{p}_j \in \mathcal{S}. \end{cases} \tag{6}$$

The *Voronoi region* associated to the source $\mathbf{p}_j \in \mathcal{S}$, noted $\mathcal{R}_j$, is the locus of points of the domain Ω which are closer (in the sense of the weighted distance) to $\mathbf{p}_j$ than to any other source of $\mathcal{S}$:

$$\mathcal{R}_j \;=\; \Big\{ \mathbf{x} \in \Omega;\, \mathcal{U}_j(\mathbf{x}) \leq \mathcal{U}_i(\mathbf{x}),\; \forall\, i \in \{1, \ldots, n\},\; i \neq j \Big\}. \tag{7}$$

The region $\mathcal{R}_j$ is a connected subset of the domain Ω, and its boundary is noted $\partial\mathcal{R}_j$. The union of Voronoi regions and its complementary set, the *Voronoi diagram*, leads to a tessellation of the domain Ω, called the *Voronoi partition*.

The *Voronoi index map* is the function $\mathcal{V} : \Omega \to \{1, \ldots, n\}$ that assigns to any point of the domain Ω the index of its Voronoi region :

$$\forall\, \mathbf{x} \in \mathcal{R}_j,\; \mathcal{V}(\mathbf{x}) = j. \tag{8}$$

If two Voronoi regions $\mathcal{R}_i$ and $\mathcal{R}_j$ are adjacent (i.e. if $\partial\mathcal{R}_i \cap \partial\mathcal{R}_j$ is a non-empty set), then the minimal path $\mathcal{C}_{\mathbf{p}_i,\mathbf{p}_j}$ passes through the point of $\partial\mathcal{R}_i \cap \partial\mathcal{R}_j$ which has the smallest $\mathcal{U}$ value. This point, noted $\mathbf{m}_{i|j}$, is the *midpoint* of the minimal path $\mathcal{C}_{\mathbf{p}_i,\mathbf{p}_j}$ since it is equidistant to $\mathbf{p}_i$ and $\mathbf{p}_j$ in the sense of the weighted distance.

The *Euclidean path length map* is the function $\mathcal{L} : \Omega \to \mathbb{R}^+$ that assigns to any point $\mathbf{x}$ of the domain Ω the Euclidean length of the minimal path between $\mathbf{x}$ and the source which is the closest in the sense of the weighted distance :

$$\forall\, \mathbf{x} \in \mathcal{R}_j,\; \mathcal{L}(\mathbf{x}) = \int_{\mathcal{C}_{\mathbf{p}_j,\mathbf{x}}} \mathrm{d}s. \tag{9}$$

Note that if $\tilde{\mathcal{P}}(\mathbf{x}) = 1$ for all $\mathbf{x} \in \Omega$, then the maps $\mathcal{U}$ and $\mathcal{L}$ are equal and both correspond to the Euclidean distance map to the set of sources $\mathcal{S}$.

Introduced first as a boundary detection method, minimal path techniques have been successfully applied to sundry problems (see [7] for a review), such as path planning [14], contour completion [6], tubular surface extraction [9], motion tracking [3], or remeshing of triangulated manifolds [15].

2.2 Fast Marching Method

The *Fast Marching Method* (FMM) is a numerical method introduced by Sethian in [17–19] and Tsitsiklis in [20] for efficiently solving the isotropic Eikonal equation on a cartesian grid. In equation (6), the values of $\mathcal{U}$ may be regarded as the arrival times of wavefronts propagating from each point of $\mathcal{S}$ with velocity $(1/\tilde{\mathcal{P}})$. The central idea behind the FMM is to visit grid points in an order consistent with the way wavefronts propagate, i.e. with the Huygens principle. It leads to a single-pass algorithm for solving equation (6) and computing the maps $\mathcal{U}$, $\mathcal{V}$ and $\mathcal{L}$ in a common computational framework (see Table 1).

Table 1 : *Fast Marching Method for solving equation (6).*

- **Notation.**
  $\mathcal{N}_{\text{M}}(\mathbf{x})$ is the set of M neighbors of a grid point $\mathbf{x}$, where $\text{M} = 4$ in 2D and $\text{M} = 6$ in 3D.
- **Initialization.**
  For each grid point $\mathbf{x}$, do
    Set $\mathcal{U}(\mathbf{x}) := +\infty$, $\mathcal{V}(\mathbf{x}) := 0$ and $\mathcal{L}(\mathbf{x}) := +\infty$.
    Tag $\mathbf{x}$ as *Far*.
  For each source $\mathbf{p}_j \in \mathcal{S}$, do
    Set $\mathcal{U}(\mathbf{p}_j) := 0$, $\mathcal{V}(\mathbf{p}_j) := j$ and $\mathcal{L}(\mathbf{p}_j) := 0$.
    Tag $\mathbf{p}_j$ as *Trial*.
- **Marching loop**.
  While the set of *Trial* points is non-empty, do
    Find $\mathbf{x}_{\text{min}}$, the *Trial* point with the smallest $\mathcal{U}$ value.
    Tag $\mathbf{x}_{\text{min}}$ as *Alive*.
    For each grid point $\mathbf{x}_{\text{n}} \in \mathcal{N}_{\text{M}}(\mathbf{x}_{\text{min}})$ which is not *Alive*, do
      $\{u, v, \ell\}$:=`UpdateSchemeFMM` $\left(\mathbf{x}_{\text{n}}, \mathcal{N}_{\text{M}}(\mathbf{x}_{\text{n}})\right)$ (see text).
      Set $\mathcal{U}(\mathbf{x}_{\text{n}}) := u$, $\mathcal{V}(\mathbf{x}_{\text{n}}) := v$ and $\mathcal{L}(\mathbf{x}_{\text{n}}) := \ell$.
      If $\mathbf{x}_{\text{n}}$ is *Far*, tag $\mathbf{x}_{\text{n}}$ as *Trial*.

Ordered sweeping of grid points.

The FMM is a front propagation approach that computes the values of $\mathcal{U}$ in increasing order, and the structure of the algorithm is almost identical to Dijkstra's algorithm for computing shortest paths on graphs [10]. In the course of the algorithm, each grid point is tagged as either *Alive* (point for which $\mathcal{U}$ has been computed and frozen), *Trial* (point for which $\mathcal{U}$ has been estimated but not frozen) or *Far* (point for which $\mathcal{U}$ is unknown). The set of *Trial* points forms an interface between the set of grid points for which $\mathcal{U}$ has been frozen (the *Alive* points) and the set of other grid points (the *Far* points). This interface may be regarded as a set of fronts expanding from each source until every grid point has been reached.

Let us denote by $\mathcal{N}_{\text{M}}(\mathbf{x})$ the set of M neighbors of a grid point $\mathbf{x}$, where $\text{M} = 4$ if Ω is a 2D domain and $\text{M} = 6$ if Ω is a 3D domain. Initially, all grid points are tagged as *Far*, except the points of $\mathcal{S}$ that are tagged as *Trial*. At each iteration of the FMM one chooses the *Trial* point with the smallest $\mathcal{U}$ value, denoted by $\mathbf{x}_{\text{min}}$ in Table 1. Then, $\mathbf{x}_{\text{min}}$ is tagged as *Alive* and the values of $\mathcal{U}$, $\mathcal{V}$ and $\mathcal{L}$ are updated for each point of the set $\mathcal{N}_{\text{M}}(\mathbf{x}_{\text{min}})$ which is either *Trial* or *Far*. In order to satisfy a causality condition, the way $\mathcal{U}$, $\mathcal{V}$ and $\mathcal{L}$ are updated in the vicinity of $\mathbf{x}_{\text{min}}$ requires special care. The iteration ends by tagging every *Far* point of the set $\mathcal{N}_{\text{M}}(\mathbf{x}_{\text{min}})$ as *Trial*. The algorithm automatically stops when all grid points are *Alive*.

The key to the speed of the FMM is the use of a priority queue to quickly find the *Trial* point with the smallest $\mathcal{U}$ value. If *Trial* points are ordered in a min-heap data structure, the computational complexity of the FMM is $\mathcal{O}(N\log N)$, where N is the total number of grid points. Some authors [21, 11, 13] have proposed various versions

of the fast marching in order to get an $\mathcal{O}(N)$ complexity. These approaches are less accurate on the result or assuming some hypotheses that are not met in our method.

Update scheme for the Fast Marching Method.

Here, we present a way to estimate $\mathcal{U}$, $\mathcal{V}$ and $\mathcal{L}$ for a grid point $\mathbf{x}_\text{n}$, i.e. a way to compute the output of routine `UpdateSchemeFMM` in Table 1. We limit ourselves to the 2D case, though extending the scheme to higher dimensions is straightforward. Adopting standard notation, we denote by $\mathcal{U}_{i,j}$ the value of $\mathcal{U}$ at the grid vertex (i, j) associated to the point $\mathbf{x}_\text{n}$ with coordinates $(i\ h_x, j\ h_y)$, where h_x and hy are grid spacings in the x and y directions.

A discretized version of (6) is solved in order to compute $\mathcal{U}_{i,j}$. For the Eikonal equation, classic finite difference schemes tend to overshoot and are unstable. Although with a different approach, Rouy and Tourin [16] showed that the correct viscosity solution for $\mathcal{U}_{i,j}$ is given by the following first order accurate scheme :

$$\left(\frac{\max\{(\mathcal{U}_{i,j}-\mathcal{U}_{i-1,j}),(\mathcal{U}_{i,j}-\mathcal{U}_{i+1,j}),0\}}{h_x}\right)^2 + \left(\frac{\max\{(\mathcal{U}_{i,j}-\mathcal{U}_{i,j-1}),(\mathcal{U}_{i,j}-\mathcal{U}_{i,j+1}),0\}}{h_y}\right)^2 = (\tilde{\mathcal{P}}_{i,j})^2\ . \tag{10}$$

This is an upwind scheme : the forward and backward differences are chosen to follow the direction of the flow of information. Since the action can only grow due to the quadratic nature of equation (10), information is propagating *upwards* (from smaller to larger values of $\mathcal{U}$).

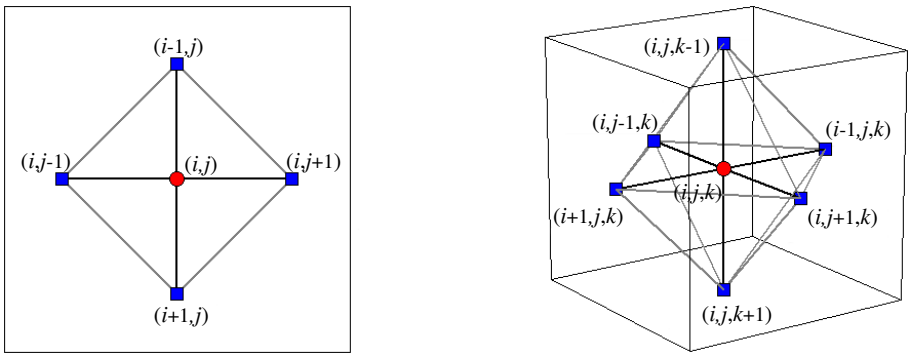

Fig. 3: Connecting a grid point $\mathbf{x}_\text{n}$ and the points of $\mathcal{N}_\text{M}(\mathbf{x}_\text{n})$ with virtual edges forms four triangles on a 2D grid and eight tetrahedrons on a 3D grid.

Connecting with virtual edges $\mathbf{x}_\text{n}$ and the points of $\mathcal{N}_\text{M}(\mathbf{x}_\text{n})$, i.e. $\mathbf{x}_\text{n}$ and the four nearby grid points, forms four triangles (see Fig. 3). For each triangle, we attempt to solve two quadratic equations to get estimations of $\mathcal{U}_{i,j}$, $\mathcal{V}_{i,j}$ and $\mathcal{L}_{i,j}$. These estimations

are respectively noted u, v and ℓ. As an example, consider the triangle with vertices $\{(i-1,j),(i,j),(i,j-1)\}$. Four cases may occur :

- If neither $(i-1,j)$ nor $(i,j-1)$ are *Alive*, information is not propagating through this triangle. Then, $u=+\infty$, $v=0$ and $\ell=+\infty$.
- If $(i-1,j)$ is *Alive* but $(i,j-1)$ is not, then $v=\mathcal{V}_{i-1,j}$. Furthermore, u and ℓ are the largest solutions of quadratic equations

$$\left(\frac{u-\mathcal{U}_{i-1,j}}{h_x}\right)^2=(\tilde{\mathcal{P}}_{i,j})^2 \quad \text{and} \quad \left(\frac{\ell-\mathcal{L}_{i-1,j}}{h_x}\right)^2=1.$$

- If $(i,j-1)$ is *Alive* but $(i-1,j)$ is not, then $v=\mathcal{V}_{i,j-1}$. Furthermore, u and ℓ are the largest solutions of quadratic equations

$$\left(\frac{u-\mathcal{U}_{i,j-1}}{h_y}\right)^2=(\tilde{\mathcal{P}}_{i,j})^2 \quad \text{and} \quad \left(\frac{\ell-\mathcal{L}_{i,j-1}}{h_y}\right)^2=1.$$

- If both $(i-1,j)$ and $(i,j-1)$ are *Alive*, then u is the largest real solution of quadratic equation

$$\left(\frac{u-\mathcal{U}_{i-1,j}}{h_x}\right)^2+\left(\frac{u-\mathcal{U}_{i,j-1}}{h_y}\right)^2=(\tilde{\mathcal{P}}_{i,j})^2,$$

  ℓ is the largest real solution of quadratic equation

$$\left(\frac{\ell-\mathcal{L}_{i-1,j}}{h_x}\right)^2+\left(\frac{\ell-\mathcal{L}_{i,j-1}}{h_y}\right)^2=1,$$

  and v is chosen according to the smallest value of $\mathcal{U}$:

$$v=\begin{cases}\mathcal{V}_{i-1,j} & \text{if } \mathcal{U}_{i-1,j}\leq\mathcal{U}_{i,j-1},\\ \mathcal{V}_{i,j-1} & \text{otherwise.}\end{cases}$$

For each of four triangles, we get a triplet $\{u,v,\ell\}$. Finally, we choose the one with the smallest u (this is the triplet returned by the routine `UpdateSchemeFMM`). Note that if one needs to approximate $\nabla\mathcal{U}$, computing the derivatives of $\mathcal{U}$ in the triangle used to estimate $\mathcal{U}_{i,j}$ gives a consistent approximation of $\nabla\mathcal{U}(\mathbf{x}_{\mathrm{n}})$. On a 3D grid, the neighborhood of a grid point is divided into eight tetrahedra (see Fig. 3), but the 3D update scheme is very similar to the 2D one.

3 Distribution of a set of keypoints on a 2D curve

3.1 Keypoints detection and local update

First, we consider the case where the domain Ω is a 2D domain. We assume that we are given an initial set $\mathcal{S}=\{\mathbf{p}_1,\ldots,\mathbf{p}_n\}$ of points on a curve along which a potential $\tilde{\mathcal{P}}:\Omega\rightarrow\mathbb{R}^{*+}$ takes lower values. Note that the set $\mathcal{S}$ may contain only one point and this will be the case in our examples below.

We propose in this paper a new approach. The idea is to detect recursively new source points (called keypoints) along the curve of interest. These points are automatically detected using a criterion based on the Euclidian length of minimal paths and are almost equi-distributed along features of interest. The method is called **Minimal Path method With Keypoint Detection** (MPWKD, see Table 2). A front is propagated from each source point of $\mathcal{S}$ with velocity $(1/\tilde{\mathcal{P}})$. Keypoints are sequentially detected, on the curve of interest, during the front propagation and stored in a set $\mathcal{S}^* = \{\mathbf{p}^*_{n+1}, \ldots, \mathbf{p}^*_{n+m}\}$. Each newly detected keypoint is immediately defined as a new source of propagation, and keypoints are detected with a criterion based on the Euclidean length of minimal paths. This criterion depends on only one parameter, denoted λ. Initially, fronts are propagated from each point of $\mathcal{S}$ with velocity $(1/\tilde{\mathcal{P}})$, until a grid point $\mathbf{x}$ such that $\mathcal{L}(\mathbf{x}) \geq \lambda$ is tagged as *Alive*. This point is then defined as the first keypoint, denoted $\mathbf{p}^*_{n+1}$ (see Fig. 6). Such a criterion has already been used in [9] to find a minimal path given only one endpoint. Assuming that the point $\mathbf{p}^*_{n+1}$ belongs to the Voronoi region $\mathcal{R}_j$ when it is detected, this criterion ensures that the minimal path $\mathcal{C}_{\mathbf{p}_j,\mathbf{p}^*_{n+1}}$ minimizes the integral of $\tilde{\mathcal{P}}$ (along itself) over all open curves with Euclidean lengths greater than or equal to λ and with endpoints in $\mathcal{S}$. Therefore, $\mathbf{p}^*_{n+1}$ is likely to belong to the curve along which the values of $\tilde{\mathcal{P}}$ are low.

Once the first keypoint has been detected, it is defined as a new source of propagation. Next, the process in iterated and assuming ℓ keypoints $\mathbf{p}^*_{n+1}, \ldots, \mathbf{p}^*_{n+\ell}$ have been added, front propagation from this set and $\mathcal{S}$ is made with velocity $1/\tilde{\mathcal{P}}$. Front propagation is continued until a grid point $\mathbf{x}$ such that $\mathcal{L}(\mathbf{x}) \geq \lambda$ is tagged as *Alive*. This point is defined as the $\ell+1$ keypoint, denoted $\mathbf{p}^*_{n+\ell+1}$, and is added to the set of sources. Afterward, front propagation is continued, and so on. Thus, during the front propagation, keypoints are sequentially detected on the curve along which $\tilde{\mathcal{P}}$ takes low values (see Figures 4, 5 and 6,). At each iteration of the process, it is not necessary to start again the whole computation from scratch since values of $\mathcal{U}$, $\mathcal{V}$ and $\mathcal{L}$ which have already been estimated would not differ in the vicinity of initial sources (i.e. in the vicinity of points of $\mathcal{S}$). In order to limit the computational cost, one just needs to update the value $\mathcal{U}$ and go on computation till criterion is reached. For this update, we reinitialize $\mathcal{U}$, $\mathcal{V}$ and $\mathcal{L}$ in the following manner :

$$\mathcal{U}(\mathbf{p}^*_{n+\ell}) := 0, \quad \mathcal{V}(\mathbf{p}^*_{n+\ell}) := n+\ell, \quad \mathcal{L}(\mathbf{p}^*_{n+\ell}) := 0,$$

tag $\mathbf{p}^*_{n+\ell}$ as *Trial* and continue front propagation. However, without any additional modification of the original FMM, final values of $\mathcal{U}$, $\mathcal{V}$ and $\mathcal{L}$ would be incorrect for grid points which are tagged as *Alive* when $\mathbf{p}^*_{n+\ell}$ is detected and closer (in the sense of a weighted distance) to $\mathbf{p}^*_{n+\ell}$ than to the initial sources. These points cannot be updated solely due to the fact that, in the original FMM, values of $\mathcal{U}$, $\mathcal{V}$ and $\mathcal{L}$ are frozen for *Alive* points. An easy way to avoid this problem is just to let an *Alive* point be tagged as *Trial* again if it is closer to the new source of propagation than to initial sources. This algorithmic trick enables the local update of $\mathcal{U}$, $\mathcal{V}$ and $\mathcal{L}$ in the neighborhood of $\mathbf{p}^*_{n+\ell}$. Notice that in [6], in a different context, there was also an algorithmic trick to update the minimal action map in order to save computation time.

Table 2 : *Minimal Path method With Keypoint Detection.*

- **Notation.**
  $\mathcal{N}_{\mathrm{M}}(\mathbf{x})$ is the set of M neighbors of a grid point $\mathbf{x}$, where $\mathrm{M} = 4$ in 2D and $\mathrm{M} = 6$ in 3D. $\mathcal{N}_{\mathrm{M}^+}(\mathbf{x})$ is the set of M^+ neighbors of a grid point $\mathbf{x}$, where $\mathrm{M}^+ = 8$ in 2D and $\mathrm{M}^+ = 26$ in 3D.
- **Initialization.**
  For each grid point $\mathbf{x}$, do
    Set $\mathcal{U}(\mathbf{x}) := +\infty$, $\mathcal{V}(\mathbf{x}) := 0$ and $\mathcal{L}(\mathbf{x}) := +\infty$.
    Tag $\mathbf{x}$ as *Far*.
  For each source $\mathbf{p}_j \in \mathcal{S}$, do
    Set $\mathcal{U}(\mathbf{p}_j) := 0$, $\mathcal{V}(\mathbf{p}_j) := j$ and $\mathcal{L}(\mathbf{p}_j) := 0$.
    Tag $\mathbf{p}_j$ as *Trial* and as *Boundary*.
  $m := 1$, *StopDetection*:=*FALSE*.
- **Marching loop**.
  While the set of *Trial* points is non-empty, do
    Find $\mathbf{x}_{\text{min}}$, the *Trial* point with the smallest $\mathcal{U}$ value.
    If $\big($*StopDetection*=*FALSE*$\big)$ and $\big(\mathcal{L}(\mathbf{x}_{\text{min}}) \geq \lambda\big)$, do
      Here, $\mathbf{x}_{\text{min}}$ *is defined as the keypoint* $\mathbf{p}^*_{n+m}$.
      Set $\mathcal{U}(\mathbf{x}_{\text{min}}) := 0$, $\mathcal{V}(\mathbf{x}_{\text{min}}) := n + m$ and $\mathcal{L}(\mathbf{x}_{\text{min}}) := 0$.
      $m := m + 1$.
    Else, do
      Tag $\mathbf{x}_{\text{min}}$ as *Alive*.
      For each grid point $\mathbf{x}_{\text{n}} \in \mathcal{N}_{\mathrm{M}}(\mathbf{x}_{\text{min}})$, do
        If $\mathbf{x}_{\text{n}}$ is not *Alive*, do
          $\{u, v, \ell\}$:=`UpdateSchemeFMM` $\big(\mathbf{x}_{\text{n}}, \mathcal{N}_{\mathrm{M}}(\mathbf{x}_{\text{n}})\big)$.
          Set $\mathcal{U}(\mathbf{x}_{\text{n}}) := u$, $\mathcal{V}(\mathbf{x}_{\text{n}}) := v$ and $\mathcal{L}(\mathbf{x}_{\text{n}}) := \ell$.
          If $\big($*StopDetection*=*FALSE*$\big)$ and $\big(\mathbf{x}_{\text{n}}$ is *Far*$\big)$, do
            Tag $\mathbf{x}_{\text{n}}$ as *Trial* and as *Boundary*.
        Else if $\mathcal{V}(\mathbf{x}_{\text{n}}) \neq \mathcal{V}(\mathbf{x}_{\text{min}})$, do (*local correction of the maps if needed*)
          $\{u, v, \ell\}$:=`UpdateSchemeFMM` $\big(\mathbf{x}_{\text{n}}, \mathcal{N}_{\mathrm{M}}(\mathbf{x}_{\text{n}})\big)$.
          If $u < \mathcal{U}(\mathbf{x}_{\text{n}})$, do
            Set $\mathcal{U}(\mathbf{x}_{\text{n}}) := u$, $\mathcal{V}(\mathbf{x}_{\text{n}}) := v$ and $\mathcal{L}(\mathbf{x}_{\text{n}}) := \ell$.
            Tag $\mathbf{x}_{\text{n}}$ as *Trial*.
      If $\mathbf{x}_{\text{min}}$ is *Boundary*, do
        Tag $\mathbf{x}_{\text{min}}$ as *Interior*.
        If *StopDetection*=*FALSE*, do
          StopDetection:=`StoppingCriterion` $\big(\mathbf{x}_{\text{min}}\big)$.

3.2 Stopping criterion on an open curve

In this section we propose two different stopping criteria for keypoint detection and front propagation on an open curve. The first simple case is when the user could provide more than a single source point, for example a source point and a destination. One can stop the algorithm as soon as the destination is reached (as was shown in figure 1). The user has to provide at least two points, a starting point and a destination (two simple

clicks), and the path length parameter λ to avoid short cut. This approach needs minimal user interaction and might be used for many different applications on 2D images.

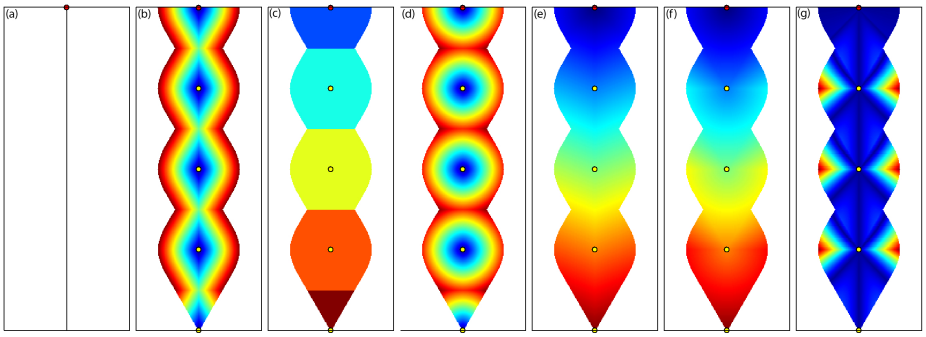

Fig. 4: Approximation of the total minimal path length. (**a**) A line potential with contrast equal to 2 : the velocity on the line is twice the velocity on the background. (**b**) The minimal action map $\mathcal{U}$ associated to some distributed keypoints. (**c**) The Voronoi map $\mathcal{V}$. (**d**) The minimal path length map $\mathcal{L}$. (**e**) The real total minimal path length map to the red source. (**f**) The approximated total minimal path length map to the red source as explained on section 3.2. (**g**) Approximation error of the total minimal path length.

We also propose another stopping criterion based on the *total path length*. Here we suppose that only one single point is provided by the user, i.e $\mathcal{S} = \{\mathbf{p}_1\}$ and the idea is to detect automatically the keypoints until a total minimal path length is reached. Such a criterion has already been used in [9] to find a minimal path given only one endpoint. In section 3.1, we explained how one can compute, on a common framework, the minimal action map $\mathcal{U}$ and the minimal path length map $\mathcal{L}$ with respect to the set $\mathcal{S} \cup \mathcal{S}^*$. A good way to compute an approximation of the total minimal path length $\mathcal{L}_T$ consists on solving the same Euclidean Eikonal equation : $\|\nabla \mathcal{L}_T\| = 1$ but without the local update as done is section 3.1, since this local update gives the minimal path length map to the set of points $\mathcal{S} \cup \mathcal{S}^*$, and one wants the minimal path length with respect to the source point $\mathbf{p}_1$. This computation gives us a good approximation of the total path length on elongated structures (see figure 4). The error is maximal on the perpendicular direction of the elongated structure and almost nil along the structure. On figure 5, the user provides a single source point, the path length parameter λ and a total length parameter to reach. Thus one can find features of interest by a minimal interaction consisting in a single click and two parameters.

3.3 Stopping criterion for a closed curve.

In order to prevent the algorithm from distributing keypoints over the whole domain Ω, one needs to stop the keypoint detection as soon as the domain visited by the fronts contains the curve of interest. Note that even if this curve is unknown, we assume that it

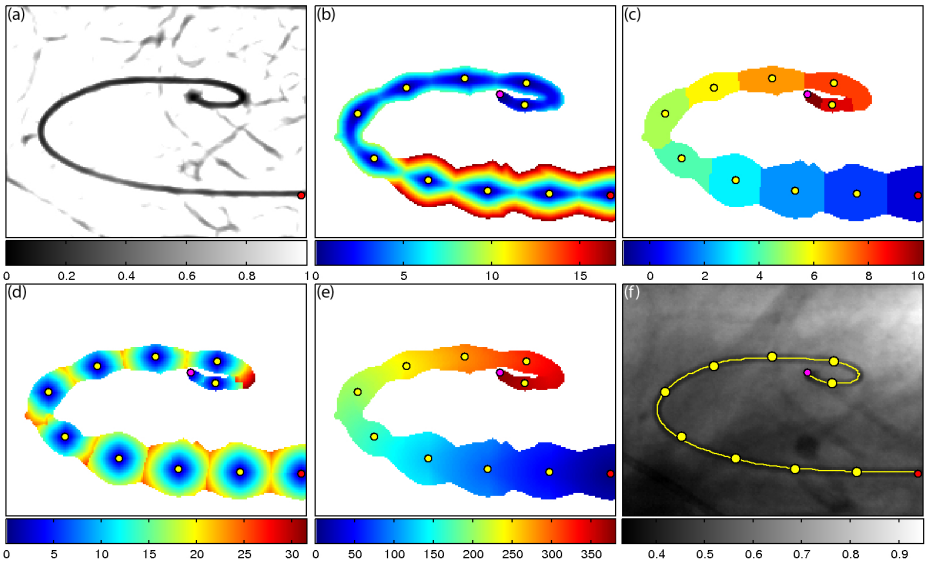

Fig. 5: Stopping criteria based on the total length. (**a**) An image potential build from the image of figure 1 (**b**) The minimal action map $\mathcal{U}$ associated to some distributed keypoints with $\lambda = 30$. (**c**) The Voronoi map $\mathcal{V}$. (**d**) The minimal path length map $\mathcal{L}$. (**e**) The approximated total minimal path length map to the red source. (**f**) Detected keypoints and minimal paths.

is closed. This topological assumption is used to devise a relevant criterion for stopping keypoint detection and front propagation.

A first strategy is to take into account the Voronoi partition, and to stop keypoint detection as soon as each Voronoi region is adjacent to at least two other Voronoi regions (i.e. as soon as there exists a cycle of Voronoi regions). This strategy, although correct, is limited to the 2D case. To get a scheme which may be extended to higher dimensions, a second strategy is proposed. Let us denote by Ω_{F} the domain visited by the propagating fronts, defined as the set of grid points which are not *Far* (i.e. the set of grid points which are either *Alive* or *Trial*). In the MPWKP, keypoint detection is stopped as soon as the border of Ω_{F} is delimited by exactly two connected subsets. The set Ω_{F} may be divided into two subsets : the set of interior points, denoted $\text{int}(\Omega_{\text{F}})$, and the set of boundary points, denoted $\partial\Omega_{\text{F}}$. In the original FMM, $\text{int}(\Omega_{\text{F}})$ and $\partial\Omega_{\text{F}}$ respectively correspond to the set of *Alive* points and the set of *Trial* points. This is no longer true in the MPWKP because of the local correction of $\mathcal{U}$, $\mathcal{V}$ and $\mathcal{L}$ in the neighborhood of a keypoint. That is why a second labelling is introduced in the MPWKP : each grid point which is not *Far*, in addition to being tagged as *Alive* or *Trial*, is also tagged as *Interior* or *Boundary* depending on whether it belongs to $\text{int}(\Omega_{\text{F}})$ or $\partial\Omega_{\text{F}}$. In the case of one source point, it is clear that the visited domain Ω_{F} surrounds the object of interest when its boundary $\partial\Omega_{\text{F}}$ splits into exactly two connected subsets. In the case of multiple sources, the number of connected boundaries of $\partial\Omega_{\text{F}}$ first decreases then increases when the band surrounding

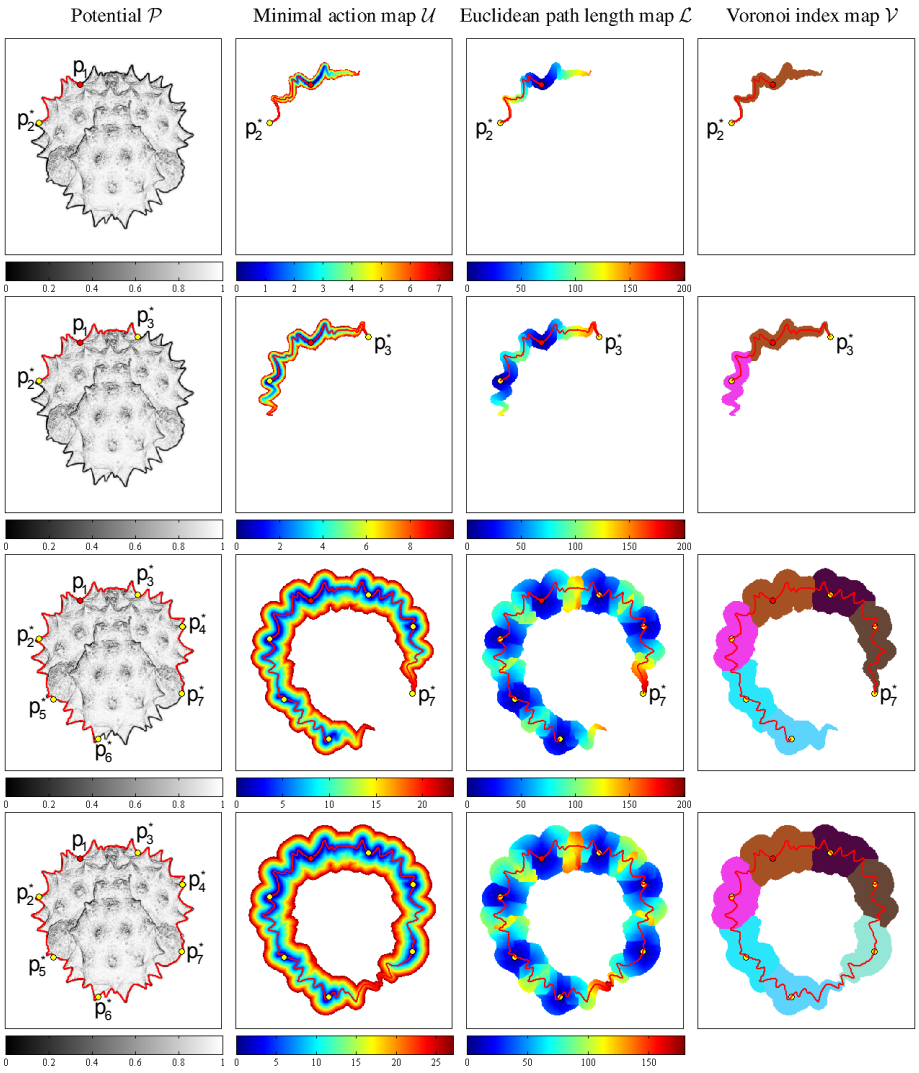

Fig. 6: Intermediate and final results for the MPWKP applied to the 2D potential of the Figure 2.**b**, with $\mathcal{S} = \{\mathbf{p}_1\}$ and $\lambda = 200$. The first, second and third rows show intermediate results obtained when are detected, respectively, $\mathbf{p}_2^*$ (the first keypoint), $\mathbf{p}_3^*$ (the second keypoint) and $\mathbf{p}_7^*$ (the last keypoint). The last row shows final results with 7 keypoints.

the object of interest is obtained. Thus, we just need to monitor the topological changes of $\partial\Omega_{\mathrm{F}}$.

In the algorithm detailed in Table 2, the stopping criterion for keypoint detection is satisfied as soon as the routine `StoppingCriterion` returns *TRUE*. This routine is called after the grid point $\mathbf{x}_{\text{min}}$ is moved from the set of *Trial* points to the set of *Alive* points, once some of the $\text{M} = 4$ neighbors of $\mathbf{x}_{\text{min}}$ have been tagged as *Boundary*. The routine `IsBoundarySplit` returns *TRUE* if both of the following tests are satisfied :

- **Local test for detecting a front collision**.
  First, we check if some fronts collide in the vicinity of $\mathbf{x}_{\text{min}}$. Let us denote by $\mathcal{N}_{\text{M}^+}(\mathbf{x}_{\text{min}})$ the set of $\text{M}^+ = 8$ neighbors of $\mathbf{x}_{\text{min}}$, and by $\partial\Omega_{\text{F}} \cap \mathcal{N}_{\text{M}^+}(\mathbf{x}_{\text{min}})$ the set of points of $\mathcal{N}_{\text{M}^+}(\mathbf{x}_{\text{min}})$ which are tagged as *Boundary*. The local test simply relies on the computation of the number of 8-connected components of $\partial\Omega_{\text{F}} \cap \mathcal{N}_{\text{M}^+}(\mathbf{x}_{\text{min}})$, denoted $\#C\big(\partial\Omega_{\text{F}} \cap \mathcal{N}_{\text{M}^+}(\mathbf{x}_{\text{min}})\big)$. Most of the time, $\mathbf{x}_{\text{min}}$ is a simple point of $\text{int}(\Omega_{\text{F}})$, and $\#C\big(\partial\Omega_{\text{F}} \cap \mathcal{N}_{\text{M}^+}(\mathbf{x}_{\text{min}})\big) = 1$ (see Fig. 7.a). The local test is satisfied if $\#C\big(\partial\Omega_{\text{F}} \cap \mathcal{N}_{\text{M}^+}(\mathbf{x}_{\text{min}})\big) > 1$, i.e. when there is a shock between some propagating fronts (see Fig. 7.b).
- **Global test for detecting a topological change of** $\partial\Omega_{\text{F}}$.
  When the local test is satisfied, we need to check if the different components of $\partial\Omega_{\text{F}} \cap \mathcal{N}_{\text{M}^+}(\mathbf{x}_{\text{min}})$ are also disconnected at a global scale. The global test is satisfied if the front collision has split an 8-connected component of $\partial\Omega_{\text{F}}$ into several 8-connected components.
  Such a test is easy to implement. For instance, consider the case where $\#C\big(\partial\Omega_{\text{F}} \cap \mathcal{N}_{\text{M}^+}(\mathbf{x}_{\text{min}})\big) = 2$. Let $\mathbf{x}_1$ and $\mathbf{x}_2$ be two grid points such that $\mathbf{x}_1$ belongs to the first component of $\partial\Omega_{\text{F}} \cap \mathcal{N}_{\text{M}^+}(\mathbf{x}_{\text{min}})$ and $\mathbf{x}_2$ to the second. We just have to visit all grid points which belong to the same 8-connected component of $\partial\Omega_{\text{F}}$ as $\mathbf{x}_1$, and assign to each visited point a temporary label. Then, the global test is satisfied if $\mathbf{x}_2$ has not been labeled.

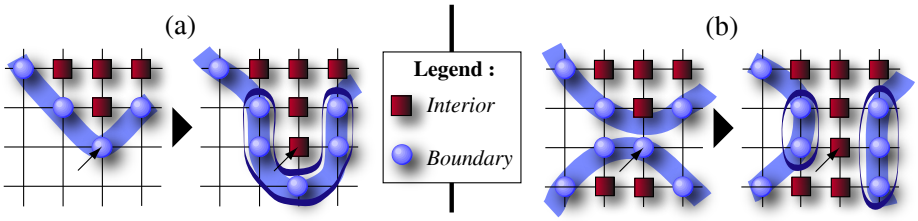

Fig. 7: Local test applied in the vicinity of a grid point $\mathbf{x}_{min}$ *(the point marked with an arrow) to detect a front collision.* (**a**) $\mathbf{x}_{min}$ *is a simple point of* $\text{int}(\Omega_{\text{F}})$ *and* $\#C\big(\partial\Omega_{\text{F}} \cap \mathcal{N}_{\text{M}^+}(\mathbf{x}_{min})\big) = 1$. (**b**) *Two fronts have collided in the neighborhood of* $\mathbf{x}_{min}$ *and* $\#C\big(\partial\Omega_{\text{F}} \cap \mathcal{N}_{\text{M}^+}(\mathbf{x}_{min})\big) = 2$.

Since the scheme used to detect the iteration at which the keypoint detection has to be stopped mainly requires tests at a local scale, it is considerably less computationally expensive than globally counting the number of connected components of $\text{int}(\Omega_{\text{F}})$ and $\partial\Omega_{\text{F}}$ at each iteration of the marching loop. Moreover, note that special care is required

to deal with the fact that a propagating front may reach the border of the domain Ω. We suggest adding virtual points along each border of the discrete grid and tagging as *Boundary* every virtual point in the neighborhood of an *Interior* point lying on the border of the grid. This ensures that any connected component of $\text{int}(\Omega_{\text{F}})$ is completely delimited by a connected set of *Boundary* points.

Once the keypoint stopping criterion is satisfied, no more grid points are moved from the set of *Far* points to the set Ω_{F}, and computations are continued until correct values of $\mathcal{U}$, $\mathcal{V}$ and $\mathcal{L}$ have been assigned to each point of Ω_{F}. The front propagation is thus limited to a band surrounding the curve of interest.

3.4 Extraction of a contour from a 2D image

In the previous sections we gave a way to obtain a set of keypoints that describe a contour curve, given an initial set of points. Our initial goal is to obtain a contour as a set of minimal paths that link pairs of points. We explain in this section how this process takes place. The MPWKD may be used to extract a closed on an open contour from a 2D image I given a single contour point $\mathbf{p}_1$ in an easy and fast manner. Once a potential $\tilde{\mathcal{P}}$ has been derived from the image I such that $\tilde{\mathcal{P}}$ takes low values along the contour of interest, applying the MPWKD with $\mathcal{S} = \{\mathbf{p}_1\}$ gives a set of points $\mathcal{S} \cup \mathcal{S}^*$, but also the maps $\mathcal{U}$ and $\mathcal{V}$ associated to $\mathcal{S} \cup \mathcal{S}^*$ and defined on a domain $\Omega_{\text{F}} \subset \Omega$. Linking pairs of points in the set $\mathcal{S} \cup \mathcal{S}^*$ by minimal paths finally enables the extraction of the desired contour.

Here, we explain first how to exploit the maps $\mathcal{U}$ and $\mathcal{V}$ to link two neighboring sources of $\mathcal{S} \cup \mathcal{S}^*$ with a minimal path. Then, we describe how to correctly choose the pairs of points of the set $\mathcal{S} \cup \mathcal{S}^*$ that should be linked, in order to build a cyclic sequence of minimal paths that follows the contour of interest.

Linking a pair of neighboring sources with a minimal path.

Consider a source $\mathbf{p}_i \in \mathcal{S} \cup \mathcal{S}^*$. The Voronoi region $\mathcal{R}_i$ associated to $\mathbf{p}_i$ may be deduced from the map $\mathcal{V} : \mathcal{R}_i = \{\mathbf{x} \in \Omega_{\text{F}};\ \mathcal{V}(\mathbf{x}) = i\}$. If there exists a Voronoi region $\mathcal{R}_j$ and a couple of grid points $(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{R}_i \times \mathcal{R}_j$ such that $(\mathbf{x}_i, \mathbf{x}_j)$ are 8-connected neighbors, then the Voronoi regions $\mathcal{R}_i$ and $\mathcal{R}_j$ are adjacent and the sources $\mathbf{p}_i$ and $\mathbf{p}_j$ are neighboring. In this case, the midpoint of the minimal path $\mathcal{C}_{\mathbf{p}_i,\mathbf{p}_j}$ may be approximated by a couple of grid points $(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j) \in \mathcal{R}_i \times \mathcal{R}_j$. Among all pairs of grid points $(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{R}_i \times \mathcal{R}_j$ which are 8-connected neighbors, $(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j)$ is the one that minimizes the accumulated energy $\Sigma\mathcal{U}$ defined by

$$\Sigma\mathcal{U}(\mathbf{x}_i, \mathbf{x}_j) = \mathcal{U}(\mathbf{x}_i) + \mathcal{U}(\mathbf{x}_j) + \frac{h(\mathbf{x}_i, \mathbf{x}_j)}{2}\Big(\tilde{\mathcal{P}}(\mathbf{x}_i) + \tilde{\mathcal{P}}(\mathbf{x}_j)\Big),$$

where $h(\mathbf{x}_i, \mathbf{x}_j)$ denotes the spacing between the grid points $\mathbf{x}_i$ and $\mathbf{x}_j$. Note that $\Sigma\mathcal{U}(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j)$ is the energy integrated along $\mathcal{C}_{\mathbf{p}_i,\mathbf{p}_j}$, i.e. the minimal weighted distance between the sources $\mathbf{p}_i$ and $\mathbf{p}_j$. Once the grid points $\tilde{\mathbf{x}}_i$ and $\tilde{\mathbf{x}}_j$ have been found, the minimal paths $\mathcal{C}_{\tilde{\mathbf{x}}_i,\mathbf{p}_i}$ and $\mathcal{C}_{\tilde{\mathbf{x}}_j,\mathbf{p}_j}$ may be retrieved by performing two gradient descents on $\mathcal{U}$, respectively from $\tilde{\mathbf{x}}_i$ to $\mathbf{p}_i$ and from $\tilde{\mathbf{x}}_j$ to $\mathbf{p}_j$. Since the two paths $\mathcal{C}_{\tilde{\mathbf{x}}_j,\mathbf{p}_i}$ and $\mathcal{C}_{\tilde{\mathbf{x}}_j,\mathbf{p}_j}$ are the two halves of $\mathcal{C}_{\mathbf{p}_i,\mathbf{p}_j}$, connecting them to each other finally gives the minimal path $\mathcal{C}_{\mathbf{p}_i,\mathbf{p}_j}$ that links the neighboring sources $\mathbf{p}_i$ and $\mathbf{p}_j$.

Building a cyclic sequence of minimal paths to extract a closed contour.

Assuming that the set $\mathcal{S} \cup \mathcal{S}^*$ contains at least three points, linking each source of $\mathcal{S} \cup \mathcal{S}^*$ to the two closest neighboring sources (in the sense of a weighted distance) via minimal paths gives a cyclic sequence of minimal paths that follow the desired closed contour (see Fig. 8). Note that finding the two closest neighboring sources of a given source only relies on the minimal weighted distance between a pair of neighboring sources $(\mathbf{p}_i, \mathbf{p}_j)$, which is given by the accumulated energy $\Sigma\mathcal{U}(\mathbf{m}_{\underline{i}|j}, \mathbf{m}_{i|\underline{j}})$ defined above.

Results on 2D data.

In figure 8, we show the segmentation results on microscopy images, found on the ANSP Algae Image Database from the Phycology Section, Patrick Center for Environmental Research, The Academy of Natural Sciences. For these images, we used a 1.6Ghz PC with 512 MB of RAM to obtain this segmentation in under a second.

3.5 Influence of the parameter λ on the results of the MPWKD

The influence of the parameter λ on the final results of the MPWKD is twofold (see Fig. 9). On the one hand, tuning λ is a direct way to control the density of keypoints. Since λ corresponds to the spacing between two successive keypoints along the curve of interest, the final number of keypoints is inversely proportional to the value given to λ. On the other hand, tuning λ is an indirect way to control the width of the band in which the front propagation is limited, i.e. an indirect way to control the aera of the domain Ω_{F}. The way the MPWKD is built ensures that λ is an upper bound of the Euclidean path length map $\mathcal{L}$ whenever a new keypoint is detected. Nevertheless, if λ is too large, short cuts may occur, and if λ is smaller than the width of the structure one wants to extract, many keypoints will be detected on it. Thus, the smaller the value given to λ is, the smaller the number of grid points visited during the front propagation is and the faster the algorithm is. In a sense, the MPWKD may be regarded as a way to limit the front propagation to a small neighborhood around the manifold of interest.

4 Distribution of a set of point on a surface or on a 3D elongated structure

4.1 Distribution of a set of points on a closed surface

Now, we consider the case where the domain Ω is a 3D domain, and we assume that we are given a few initial points $\mathcal{S} = \{\mathbf{p}_1, \ldots, \mathbf{p}_n\}$ distributed on a closed surface along which a potential $\tilde{\mathcal{P}} : \Omega \to \mathbb{R}^{*+}$ takes lower values.

The MPWKD, as it has been introduced in section 3.1, may be straightforwardly extended from a 2D to a 3D framework, in order to distribute a set of points on a closed surface. The overall algorithm (see Table 2) is similar in 2D and 3D, with the difference that 4-connectivity and 8-connectivity on a 2D grid, becomes 6-connectivity and 26-connectivity on a 3D grid.

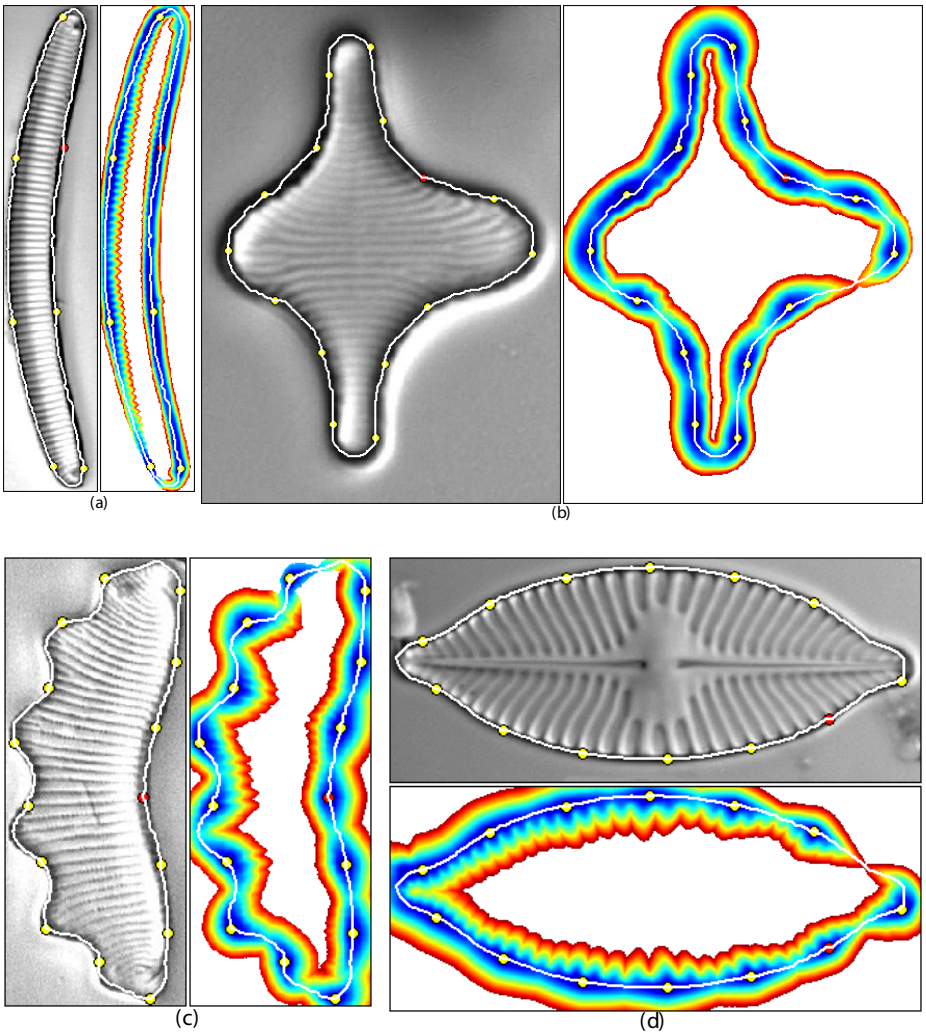

Fig. 8: Extraction of a closed contour from a 2D microscopy image. Potential $\mathcal{P}$, set of sources $\mathcal{S} \cup \mathcal{S}^*$, Minimal action map and cyclic sequence of minimal paths. (**a**) Image size 101×521, $\lambda = 180$. (**b**) 385×532, $\lambda = 80$. (**c**) 153×380, $\lambda = 60$. (**d**) 1032×435, $\lambda = 160$.

Therefore, fronts are propagated from each point of $\mathcal{S}$ with velocity $(1/\tilde{\mathcal{P}})$ and, during the front propagation, keypoints are sequentially detected on the closed surface along which $\tilde{\mathcal{P}}$ takes low values. Keypoint detection and front propagation are led until a front collision splits a 26-connected component of $\partial\Omega_{\mathrm{F}}$ into several 26-connected components. When the algorithm ends, we finally get a set of points $\mathcal{S} \cup \mathcal{S}^*$ distributed on the closed surface (see Fig. 10), but also the maps $\mathcal{U}$, $\mathcal{V}$ and $\mathcal{L}$ associated to the set of sources $\mathcal{S} \cup \mathcal{S}^*$. These maps are defined on a domain Ω_{F}, such that Ω_{F} is a

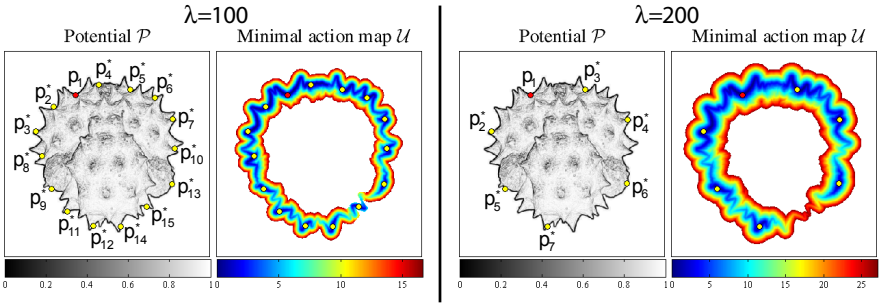

Fig. 9: Influence of the parameter λ on the results of the MPWKD applied to the 2D potential of the Figure 2.**b** with $\mathcal{S} = \{\mathbf{p}_1\}$. The value given to λ influences the number of detected keypoints and the aera of the domain Ω_F.

connected subset of Ω delimited by two connected surfaces (an inner boundary and an outer boundary). We will see in section 4.3 how to use these points.

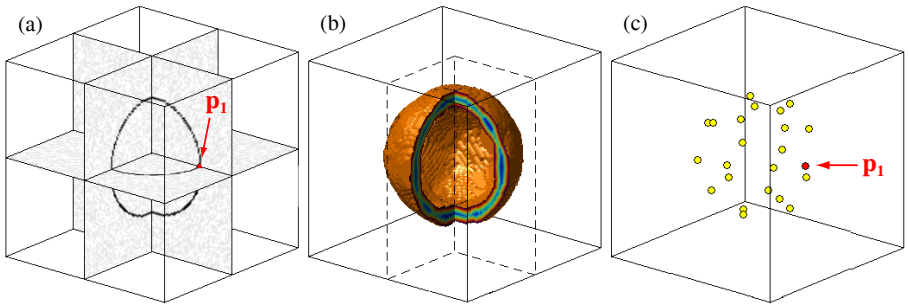

Fig. 10: Distribution of a set of points on a sphere, by applying the MPWKD to a 3D synthetic potential with $\mathcal{S} = \{\mathbf{p}_1\}$. (**a**) Cut view of Ω showing the initial point $\mathbf{p}_1$ and values of the potential $\mathcal{P}$. (**b**) Cut view showing values of the minimal action map $\mathcal{U}$ inside Ω_F. (**c**) Set of points $\mathcal{S} \cup \mathcal{S}^*$.

4.2 Distribution of a set of keypoints on a 3D elongated structure

Ideas presented on section 3.2 can be extended to 3D straightforwardly. Then, one can use this algorithm to detect 3D elongated structure by using the proposed criteria. The first criterion needs more interaction, so the user must provide a source point a destination point and path length parameter λ. For the second criterion, the user provides at least one source point, the parameter λ and a total path length to reach, which corresponds to the maximum length of arteries for example (see figure 11). We conclude that

using this method, with minimal interaction, one can have a first good segmentation on elongated structures like arteries.

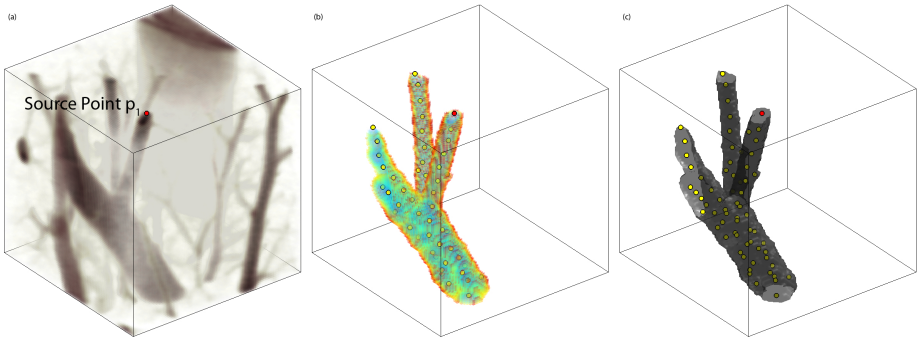

Fig. 11: Distribution of a set of points on a tubular structure. (**a**) Original image used as potential. (**b**) Values of the minimal action map $\mathcal{U}$ on the visited domain. (**c**) The front of the visited domain, with detected keypoints (yellow) and the source point (red).

4.3 Geodesic meshing of a closed surface from a 3D image

Once we obtain a set of keypoints that describe the desired surface, using minimal paths that link pairs of points as in section 3.4 does not lead to a surface but to a mesh of surface. Consider a potential $\tilde{\mathcal{P}}$, derived from the image I which takes lower values along a closed surface, and a single surface point $\mathbf{p}_1$. Applying the MPWKD with $\mathcal{S} = \{\mathbf{p}_1\}$ generates a cloud of points $\mathcal{S} \cup \mathcal{S}^*$ distributed on the whole surface of interest. Moreover, the MPWKD enables the computation of the maps $\mathcal{U}$ and $\mathcal{V}$ associated to the set $\mathcal{S} \cup \mathcal{S}^*$ and defined in a domain Ω_{F} that surrounds the underlying surface.

Then, it is quite straightforward to devise an algorithm for meshing the surface of interest, interpreting the cloud of points $\mathcal{S} \cup \mathcal{S}^*$ as a set of mesh vertices and the Voronoi partition of Ω_{F} as a way to derive the mesh connectivity. As in the 2D case, two points of the set $\mathcal{S} \cup \mathcal{S}^*$ may be considered as neighboring sources if their respective Voronoi regions are adjacent, and two neighboring sources may be linked with a minimal path by performing two gradient descents on $\mathcal{U}$ from the path midpoint. Linking each pair of neighboring sources with a minimal path finally gives a *geodesic mesh* that describes the underlying surface (see Fig. 12 and Fig. 13).

As seen in Section 3.4, it is straightforward to extract a closed contour from a 2D image. One may only get a mesh of minimal paths on a closed surface. Future work includes a new step based on a recent implicit method by Ardon et. al. [1], to obtain a complete closed surface.

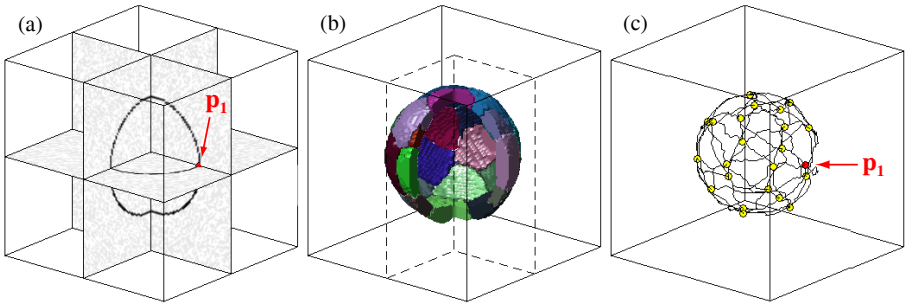

Fig. 12: Geodesic meshing of a sphere from a 3D synthetic potential and a single surface point $\mathbf{p}_1$. (**a**) Cut view of Ω showing the initial point $\mathbf{p}_1$ and values of the potential $\mathcal{P}$. (**b**) Cut view showing values of the Voronoi index map $\mathcal{V}$ inside Ω_{F}. (**c**) Set of points $\mathcal{S} \cup \mathcal{S}^*$ and geodesic mesh.

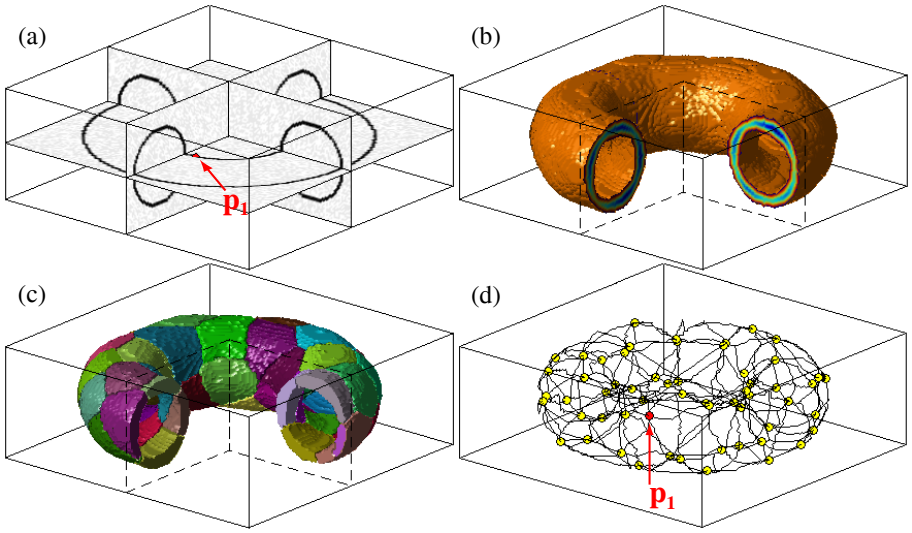

Fig. 13: Geodesic meshing of a torus from a 3D synthetic potential and a single surface point $\mathbf{p}_1$. (**a**) Cut view of Ω showing the initial point $\mathbf{p}_1$ and values of the potential $\mathcal{P}$. (**b**) Cut view showing values of the minimal action map $\mathcal{U}$ inside Ω_{F}. (**c**) Cut view showing values of the Voronoi index map $\mathcal{V}$ inside Ω_{F}. (**d**) Set of points $\mathcal{S} \cup \mathcal{S}^*$ and geodesic mesh.

5 Conclusion

In this paper we have proposed a new method to segment 2D object boundaries or to segment partially 3D object boundaries. Our method needs minimal interaction : a single source point and one or two real parameters are required, depending on the object topology. From a set of provided source points, which might be reduced to a single

source point, a front is propagated and new keypoints are iteratively detected on the boundary. The detection of these keypoints is based on the Euclidean length of minimal path to the set of source points with respect to a given metric depending on the image. A tricky algorithm has been introduced allowing local correction of the maps. Thus, computation time is considerably reduced. We proposed different stopping criteria. The first one is based on the total Euclidean path length from the sources. This criterion might be applied for 2D open object boundaries or 3D elongated structures and requires a total path length parameter to reach provided by the user. The second criterion is very simple. An endpoint is required , and the front propagation stops when this endpoint is reached. Finally, a specific stopping criterion for 2D or 3D closed object boundaries is proposed. This criterion is based on the front topology. The keypoints detection and front propagation is stopped when the visited domain surrounds the object of interest. Then, a few parameters as a source point and one or two real values can be enough to generate a coherent object boundary segmentation.

References

1. Roberto Ardon, Laurent D. Cohen, and Anthony Yezzi. A new implicit method for surface segmentation by minimal paths in 3d images. *Applied Mathematics and Optimization*, 55:127–144(18), March 2007.
2. F. Benmansour, S. Bonneau, and L.D. Cohen. Finding a closed boundary by growing minimal paths from a single point on 2d or 3d images. In *MMBIA07*, pages 1–8, 2007.
3. S. Bonneau, M. Dahan, and L. D. Cohen. Single quantum dot tracking based on perceptual grouping using minimal paths in a spatiotemporal volume. *IEEE Transactions on Image Processing*, **14**:1384–1395, 2005.
4. Stéphane Bonneau. *Chemins minimaux en Analyse d'images : Nouvelles contributions et applications à l'imagerie biologique*. PhD thesis, Université Paris-Dauphine, France, 2006.
5. V. Caselles, R. Kimmel, and G. Sapiro. Geodesic active contours. *International Journal of Computer Vision*, **22**:61–79, 1997.
6. L. D. Cohen. Multiple contour finding and perceptual grouping using minimal paths. *Journal of Mathematical Imaging and Vision*, **14**:225–236, 2001.
7. L. D. Cohen. Minimal paths and fast marching methods for image analysis. In Nikos Paragios, Yunmei Chen, and Olivier Faugeras, editors, *Mathematical Models in Computer Vision: The Handbook*. Springer, 2005.
8. L. D. Cohen and R. Kimmel. Global minimum for active contour models: a minimal path approach. *International Journal of Computer Vision*, **24**:57–78, 1997.
9. T. Deschamps and L. D. Cohen. Fast extraction of minimal paths in 3D images and applications to virtual endoscopy. *Medical Image Analysis*, **5**:281–299, 2001.
10. E. W. Dijkstra. A note on two problems in connection with graphs. *Numerische Mathematic*, **1**:269–271, 1959.
11. A. X. Falcão, J. K. Udupa, and F. K. Miyazawa. An ultra-fast user-steered image segmentation paradigm: Live-wire-on-the-fly. *IEEE Trans. on Medical Imaging*, 19(1):55–62, Jan 2000.
12. M. Kass, A. Witkin, and D. Terzopoulos. Snakes: active contour models. *International Journal of Computer Vision*, **1**:321–331, 1988.
13. S. Kim. An $\mathcal{O}$(N) level set method for Eikonal equations. *SIAM Journal on Scientific Computing*, **22**:2178–2193, 2001.
14. R. Kimmel and J. A. Sethian. Optimal algorithm for shape from shading and path planning. *Journal of Mathematical Imaging and Vision*, **14**:237–244, 2001.

15. G. Peyré and L. D. Cohen. Geodesic remeshing using front propagation. *International Journal of Computer Vision*, **69**:145–156, 2006.
16. E. Rouy and A. Tourin. A viscosity solution approach to shape from shading. *SIAM Journal on Numerical Analysis*, **29**:867–884, 1992.
17. J. A. Sethian. A fast marching level set for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*, **93**:1591–1595, 1996.
18. J. A. Sethian. Fast marching methods. *SIAM Review*, **41**:199–235, 1999.
19. J. A. Sethian. *Level Set Methods and Fast Marching Methods*. Cambridge University Press, 1999.
20. J. N. Tsitsiklis. Efficient algorithms for globally optimal trajectories. *IEEE Transactions on Automatic Control*, **40**:1528–1538, 1995.
21. L. Yatziv, A. Bartesaghi, and G. Sapiro. $\mathcal{O}$(N) implementation of the fast marching algorithm. *Journal of Computational Physics*, **212**:393–399, 2006.
22. A. Yezzi, S. Kichenassamy, A. Kumar, P. Olver, and A. Tannenbaum. A geometric snake model for segmentation of medical imagery. *IEEE Transactions on Medical Imaging*, **16**:199–209, 1997.