Published in Image Processing On Line on 2019–02–24.
Submitted on 2018–04–25, accepted on 2019–02–11.
ISSN 2105–1232 © 2019 IPOL & the authors CC–BY–NC–SA
This article is available online with supplementary materials,
software, datasets and online demo at
https://doi.org/10.5201/ipol.2019.227

Hamiltonian Fast Marching: A Numerical Solver for Anisotropic and Non-Holonomic Eikonal PDEs

Jean-Marie Mirebeau[1], Jorg Portegies[2]

[1] University Paris-Sud, CNRS, University Paris-Saclay, 91405, Orsay, France
jean-marie.mirebeau@math.u-psud.fr
[2] Department of Mathematics and Computer Science, CASA, Eindhoven University of Technology, Netherlands
j.m.portegies@tue.nl

Communicated by Bertrand Kerautret *Demo edited by* Bertrand Kerautret

Abstract

We introduce a generalized Fast-Marching algorithm, able to compute paths globally minimizing a measure of length, defined with respect to a variety of metrics in dimension two to five. Our method applies in particular to arbitrary Riemannian metrics, and implements features such as second order accuracy, sensitivity analysis, and various stopping criteria. We also address the singular metrics associated with several non-holonomic control models, related with curvature penalization, such as the Reeds-Shepp's car with or without reverse gear, the Euler-Mumford elastica curves, and the Dubins car. Applications to image processing and to motion planning are demonstrated.

Source Code

This paper is related to the *HamiltonFastMarching* code, designed to solve various classes of eikonal equations, and extract the related minimal paths. The software is written in C++, but comes with interfaces for the scripting languages Python, Matlab® and Mathematica®. The codes are available at the web page of the article[1].

Supplementary Material

A series of notebooks[2] written in the Python language, present a hands on usage of the code provided in this paper, and allow to reproduce the numerical experiments. Additional notebooks[3] by the second author are written in the Mathematica® language.

Keywords: eikonal equation; geodesic; elastica curve; image segmentation; motion planning

[1] https://doi.org/10.5201/ipol.2019.227
[2] http://nbviewer.jupyter.org/github/Mirebeau/HFM_Python_Notebooks/blob/master/Summary.ipynb
[3] https://github.com/Mirebeau/HFM_Mathematica_Notebooks

1 Introduction

In this paper we provide the numerical details involved in a generalized fast marching solver, able to compute the distance from a point within a domain, and the related paths of minimal length, which by design are globally optimal. Path length is measured with respect to a given metric which may take a variety of forms and involve data-driven parameters. For instance the metric may be Isotropic (locally proportional to the Euclidean norm), or Riemannian. Combining a dimension lifting trick with the choice of an adequate singular metric, we are also able to penalize path curvature, according to the Reeds-Shepp [50, 22], Euler-Mumford [43], or Dubins [19] models. Our software is referred to as the Hamiltonian Fast Marching (HFM) library[4], because it relies on an original and specific representation of the Hamiltonian of the addressed problems (14), and uses the fast marching method (a generalization of Dijkstra's algorithm) to solve eikonal-type equations in a single pass.

The search for optimal paths appears in many applications, such as motion planning and image analysis, see the numerical section of this paper and e.g. [57, 48]. The numerical approach chosen in this paper is to first compute a distance map, which is the viscosity solution to a Partial Differential Equation (PDE) of eikonal type [2], and then extract the minimal paths, which obey an Ordinary Differential Equation (ODE) defined in terms of the distance map. In the case of isotropic metrics, the first task is essentially a solved problem [51, 61], however it becomes challenging when considering (strongly) anisotropic metrics [29, 58, 8, 1, 35, 36]. For that purpose, we rely on an original *Eulerian* and *causal* discretization of the eikonal equation [40, 39]. Two robust geodesic backtracking methods are provided for minimal path extraction. The HFM software is made available as an open-source C++ library with interfaces to Python®, Matlab®, and Mathematica®. It is fast (albeit single-threaded by nature), built for extensibility (for introducing additional classes of metrics, stopping criteria, interfaces with other programming languages, etc), features uncommon but useful methods such as reverse mode sensitivity analysis, and comes with a thorough series of introductory notebooks, see also the numerical sections 4 and 5.

1.1 Curve Optimization Via Eikonal PDEs

In order to make our contributions clear and to relate them to the literature, we need to formally state the addressed mathematical problem. The objects discussed in this section - the metric, the distance map, the geodesic flow, and the optimal paths - are illustrated in Figure 1.

Let $\mathbb{E} = \mathbb{R}^d$ be the ambient space, and let $\Omega \subset \mathbb{E}$ be a domain[5]. The geometry of the domain is described using a quasi-metric $\mathcal{F} : \overline{\Omega} \times \mathbb{E} \to [0, \infty]$, which specific form is discussed later. This object defines a measure of path length $\mathcal{L}_\mathcal{F} : \Gamma \to [0, \infty]$, where $\Gamma := \mathrm{Lip}([0, 1], \overline{\Omega})$ is the set of locally Lipschitz paths, and a related quasi-distance $d_\mathcal{F} : \overline{\Omega} \times \overline{\Omega} \to [0, \infty]$, as follows. For any path $\gamma \in \Gamma$, and any points $\mathbf{p}, \mathbf{q} \in \overline{\Omega}$

$$\mathcal{L}_\mathcal{F}(\gamma) := \int_0^1 \mathcal{F}_{\gamma(t)}(\dot\gamma(t))\mathrm{d}t, \qquad d_\mathcal{F}(\mathbf{p}, \mathbf{q}) := \inf\{\mathcal{L}_\mathcal{F}(\gamma);\ \gamma \in \Gamma, \gamma(0) = \mathbf{p}, \gamma(1) = \mathbf{q}\}. \tag{1}$$

Depending on the choice of quasi-metric $\mathcal{F}$ the quasi-distance $d_\mathcal{F}$ may *or may not* satisfy $d_\mathcal{F}(\mathbf{p}, \mathbf{q}) = d_\mathcal{F}(\mathbf{q}, \mathbf{p})$ (symmetry), $d_\mathcal{F}(\mathbf{p}, \mathbf{q}) < \infty$ (global controllability), or $d_\mathcal{F}(\mathbf{p}, \mathbf{q}) \to 0$ as $\mathbf{p} \to \mathbf{q}$ (local controllability). For readability we choose to drop in the following the prefix "quasi-", used conventionally to emphasize that a metric or a distance may lack symmetry. Note that we reserve the word *metric* for functions $\mathcal{F}$ measuring the length $\mathcal{F}_\mathbf{p}(\dot{\mathbf{p}})$ of a tangent vector $\dot{\mathbf{p}}$ at a given point $\mathbf{p}$, as in (1, left),

[4] github.com/mirebeau/HamiltonFastMarching

[5] Our numerical method requires Ω to be a bounded box, but allows to introduce obstacles in the domain, and to equip it with various kinds of periodic boundary conditions.

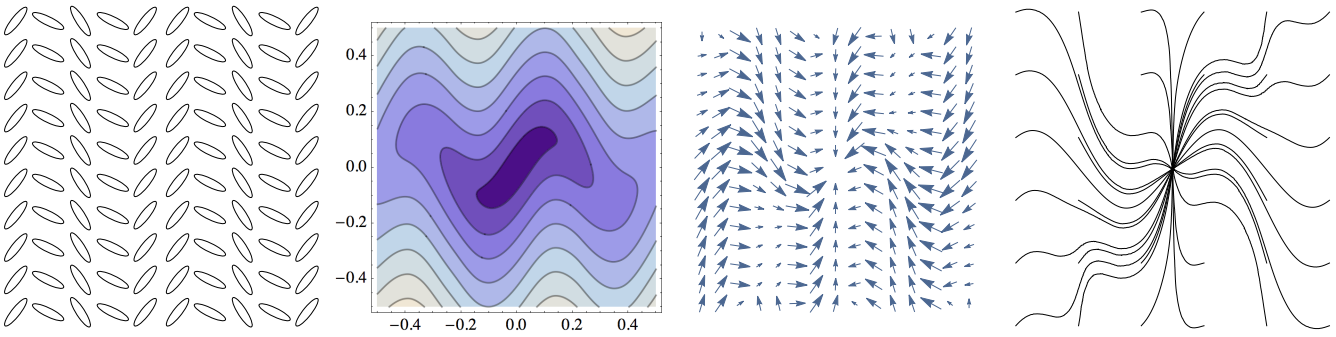

Figure 1: The four steps of minimal path computation. (I) Define a metric, here of Riemannian type and visualized as a family of ellipsoids. (II) Compute of a distance map, here from the center point, by solving the eikonal PDE (3). (III) Compute the geodesic flow direction $\mathbf{p} \in \Omega \mapsto \mathrm{d}\mathcal{F}^*_{\mathbf{p}}(\mathrm{d}u(\mathbf{p}))$. (IV) Backtrack the minimal geodesics, by solving the ODE (5).

and the word *distance* for functions such as $d_{\mathcal{F}}$ measuring the minimal path length between two points, as in (1, right).

Our objective is to compute the distance map $u : \overline{\Omega} \to]-\infty, \infty]$ from the domain boundary $\partial\Omega$, which can be characterized as follows

$$\forall \mathbf{p} \in \Omega,\ u(\mathbf{p}) = \inf_{\mathbf{q} \in \partial\Omega} u(\mathbf{q}) + d_{\mathcal{F}}(\mathbf{q}, \mathbf{p}), \qquad \forall \mathbf{p} \in \partial\Omega,\ u(\mathbf{p}) = \sigma(\mathbf{p}), \tag{2}$$

where $\sigma : \overline{\Omega} \to]-\infty, \infty]$ is a given initial delay. Note that we could merge (2) into a single equation, namely $u(\mathbf{p}) = \inf\{\sigma(\mathbf{q}) + d_{\mathcal{F}}(\mathbf{q}, \mathbf{p}); \mathbf{q} \in \partial\Omega\}$ for all $\mathbf{p} \in \overline{\Omega}$. We prefer however to formally separate the boundary condition (2, right), from the self consistency property (2, left) of the distance map u, which gives rise to the eikonal PDE stated below (3, left). The distance from e.g. a single point of interest $\mathbf{q}_* \in \partial\Omega$ is obtained by setting $\sigma(\mathbf{q}_*) = 0$ and $\sigma \equiv \infty$ on $\partial\Omega \setminus \{\mathbf{q}_*\}$. Under suitable assumptions, the function u is known to be the unique viscosity solution to the (generalized) eikonal PDE

$$\forall \mathbf{p} \in \Omega,\ \mathcal{F}^*_{\mathbf{p}}(\mathrm{d}u(\mathbf{p})) = 1, \qquad \forall \mathbf{p} \in \partial\Omega,\ u(\mathbf{p}) = \sigma(\mathbf{p}), \tag{3}$$

where "d" denotes the differentiation operator. We refer to [2] for this theory, including the concept of *discontinuous* solutions to eikonal equations, which is required for some of our problem instances [39] but is out of the scope of this paper. The eikonal PDE (3) involves the dual metric $\mathcal{F}^* : \overline{\Omega} \times \mathbb{E}^* \to [0, \infty[$, defined by

$$\mathcal{F}^*_{\mathbf{p}}(\hat{\mathbf{p}}) := \sup\{\langle \hat{\mathbf{p}}, \dot{\mathbf{p}} \rangle;\ \dot{\mathbf{p}} \in \mathbb{E},\ \mathcal{F}_{\mathbf{p}}(\dot{\mathbf{p}}) \leq 1\}, \tag{4}$$

for any given $\mathbf{p} \in \overline{\Omega}$ and $\hat{\mathbf{p}} \in \mathbb{E}^*$. Here and below, points $\mathbf{p} \in \overline{\Omega}$ are denoted with bold letters, tangent vectors $\dot{\mathbf{p}} \in \mathbb{E}$ are distinguished with dots, and co-vectors $\hat{\mathbf{p}} \in \mathbb{E}^*$ are decorated with hats. We emphasize that the supremum in (4) is taken among all tangent vectors $\dot{\mathbf{p}}$ with length $\mathcal{F}_{\mathbf{p}}(\dot{\mathbf{p}})$ smaller or equal to 1, as measured by the (primal) metric $\mathcal{F}$ at the given base point $\mathbf{p}$.

Once u is known, the minimal path joining a given $\mathbf{q} \in \Omega$ from the corresponding optimal $\mathbf{p} \in \partial\Omega$ is extracted by solving (backwards in time) the following Ordinary Differential Equation (ODE)

$$\forall t \in [T^-, T^+],\ \dot{\gamma}(t) = \mathrm{d}\mathcal{F}^*_{\gamma(t)}(\mathrm{d}u(\gamma(t))), \qquad \gamma(T^+) = \mathbf{q}, \tag{5}$$

where $T^- := \sigma(\mathbf{p})$ and $T^+ := u(\mathbf{q})$. The parametrization interval $[T_-, T_+]$ used here for the path γ differs in general from the one $[0, 1]$ appearing in the definition of path length (1), but this is not an issue since path length is invariant under non-decreasing reparametrization. We denoted by $\mathrm{d}\mathcal{F}^*_{\mathbf{p}}(\hat{\mathbf{p}})$ the differential of $\hat{\mathbf{p}} \in \mathbb{E}^* \mapsto \mathcal{F}^*_{\mathbf{p}}(\hat{\mathbf{p}})$, see [22] Appendix C.

We discuss in Section 1.2 possible choices of metric $\mathcal{F}$ that can be addressed with the HFM library, and in particular the singular metrics used to encode curvature penalization. The design

choices underlying our numerical method for solving the PDE (3) are then presented in Section 1.3. Applications of minimal path methods to image processing and motion planning are briefly surveyed in Section 1.4, where we also present the outline of the paper.

1.2 Isotropic, Anisotropic and Non-Holonomic Metrics

In the applications for which our software is intended, such as image processing and motion planning, the metric $\mathcal{F}$ encoding the geometry of the domain is usually data driven and spatially inhomogeneous. The specific structure of the expression of the metric is in principle dictated by the application, but it must also be compatible with the numerical method used for minimal path computation. We present in the following the metric models that can be addressed using our software, distinguishing three general classes: *Isotropic*, *Anisotropic*, and *Non-Holonomic*, which are illustrated in Figure 2. In each case, we provide the generic expression of the primal metric, as well as the dual metric which is defined by (4) and appears in the eikonal equation (3) and the geodesic backtracking ODE (5).

Isotropic metrics are locally proportional to the Euclidean norm, which is defined as usual on the embedding space $\mathbb{E} := \mathbb{R}^d \supseteq \Omega$. The dual metric turns out to be proportional to the Euclidean norm as well, but with an inverse ratio. For any point $\mathbf{p} \in \overline{\Omega}$, any vector $\dot{\mathbf{p}} \in \mathbb{E}$ and any co-vector $\hat{\mathbf{p}} \in \mathbb{E}^*$

$$\mathcal{F}_{\mathbf{p}}(\dot{\mathbf{p}}) := c(\mathbf{p})\|\dot{\mathbf{p}}\|, \qquad \mathcal{F}^*_{\mathbf{p}}(\hat{\mathbf{p}}) = \|\hat{\mathbf{p}}\|/c(\mathbf{p}). \tag{6}$$

The cost function $c : \overline{\Omega} \to]0, \infty[$, appearing in these expressions, is given by the user and usually data driven. It is homogeneous to the inverse of a speed, and assumed to be continuous. The HFM software uses the classical discretization scheme [51] to numerically solve isotropic eikonal equations on Cartesian grids of arbitrary dimension.

Anisotropic metrics define, at any given point, different norms for unit Euclidean vectors in different directions. Riemannian metrics are the simplest and most common instance of anisotropic metrics. They are determined by a field $\mathcal{M} : \overline{\Omega} \to \mathrm{S}^{++}(\mathbb{E})$ of positive definite tensors, and take the form

$$\mathcal{F}_{\mathbf{p}}(\dot{\mathbf{p}}) := \|\dot{\mathbf{p}}\|_{\mathcal{M}(\mathbf{p})}, \qquad \mathcal{F}^*_{\mathbf{p}}(\hat{\mathbf{p}}) = \|\hat{\mathbf{p}}\|_{\mathcal{D}(\mathbf{p})}, \tag{7}$$

where $\|\dot{\mathbf{p}}\|_M := \sqrt{\langle M\dot{\mathbf{p}}, \dot{\mathbf{p}}\rangle}$, and where $\mathcal{D}(\mathbf{p}) := \mathcal{M}(\mathbf{p})^{-1}$ denotes the dual tensor field. The HFM software uses the recently developed FM-VR1 numerical scheme [40], to solve Riemannian eikonal equations, in dimension $d \in \{2, 3\}$. Finsler metrics [64, 13], i.e. non-Riemannian anisotropic metrics, are presently not supported by the HFM software. For these we refer to the earlier open source library [37], due to the first author, which implements the (two-dimensional) FM-ASR scheme [35].

Non-Holonomic control models forbid, at some points in space $\mathbf{p} \in \Omega$, some directions of motion $\dot{\mathbf{p}} \in \mathbb{E}$. The corresponding metrics are singular, in that they associate infinite costs $\mathcal{F}_{\mathbf{p}}(\dot{\mathbf{p}}) = +\infty$ to these forbidden directions. Sub-Riemannian metrics are the simplest instances of non-holonomic metrics, and can be regarded as generalized Riemannian metrics having some infinite eigenvalues. The HFM software is able to handle strongly anisotropic Riemannian metrics, with condition numbers $\gtrsim 10$, which can approximate sub-Riemannian metrics convincingly enough for many applications. We implement in particular the Reeds-Shepp car model, defined on the configuration space $\mathbb{R}^2 \times \mathbb{S}^1$ of positions and orientations, which is adequate for modeling wheelchair-like vehicles [50] but is also fundamental in image vision [47]. Several generalisations, defined on the 5-dimensional domain $\mathbb{R}^3 \times \mathbb{S}^2$, see [22, 40], are also present.

We address the computation of planar paths globally minimizing several curvature dependent energies, following [39], which we now briefly review. For that purpose, consider again the three-dimensional state space $\mathbb{M} := \mathbb{R}^2 \times \mathbb{S}^1$ of positions and orientations, which points are denoted $\mathbf{p} = (\mathbf{x}, \theta)$. A singular quasi-metric $\mathcal{F}$ is introduced, involving a data-driven cost function $c : \mathbb{M} \to]0, \infty[$

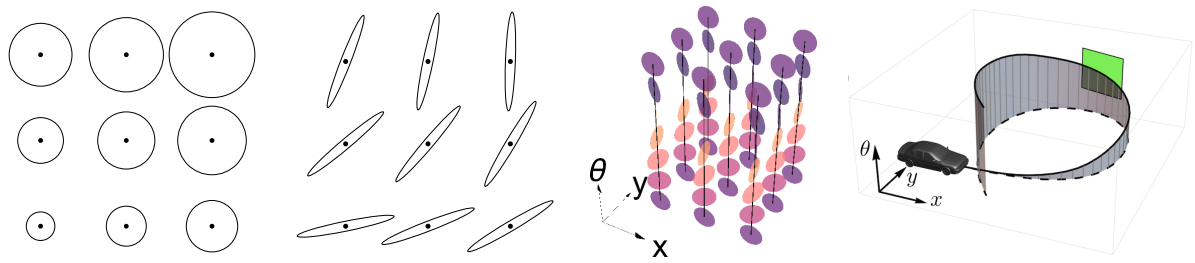

Figure 2: A metric $\mathcal{F}$ associates to each point $\mathbf{p}$ a gauge $\mathcal{F}_{\mathbf{p}}$, i.e. a generalized norm which may be non-symmetrical and may take infinite values. The three first figures display unit balls (also referred to as Tissot's indicatrix) of the gauges of successively: (i) an Isotropic metric, (ii) an Anisotropic Riemannian metric, (iii) and the non-holonomic sub-Riemannian metric associated with the Reeds-Shepp model on $\mathbb{R}^2 \times \mathbb{S}^1$. In the later case, the admissible directions of motion depend on the angular coordinate θ (vertical axis in (iii) and (iv)), reflecting the fact that a car cannot move sideways.

and a curvature penalty $\mathcal{C} : \mathbb{R} \to]0, \infty]$. For any tangent vector $\dot{\mathbf{p}} = (\dot{\mathbf{x}}, \dot{\theta})$, with unit physical velocity $\|\dot{\mathbf{x}}\| = 1$ and arbitrary angular velocity $\dot{\theta}$, and any co-tangent vector $(\hat{\mathbf{x}}, \hat{\theta})$, one has

$$\mathcal{F}_{(\mathbf{x},\theta)}(\dot{\mathbf{x}}, \dot{\theta}) := c(\mathbf{x}, \theta) \cdot \begin{cases} \mathcal{C}(\dot{\theta}) \text{ if } \dot{\mathbf{x}} = \mathbf{n}(\theta), \\ +\infty \text{ otherwise.} \end{cases} \qquad \mathcal{F}^*_{(\mathbf{x},\theta)}(\hat{\mathbf{x}}, \hat{\theta}) = c(\mathbf{x}, \theta)^{-1} \sup_{\dot{\theta} \in \mathbb{R}} \frac{\langle \hat{\mathbf{x}}, \mathbf{n}(\theta) \rangle + \hat{\theta}\dot{\theta}}{\mathcal{C}(\dot{\theta})}. \tag{8}$$

We denoted $\mathbf{n}(\theta) := (\cos\theta, \sin\theta)$, and adopted the convention that $\mathbb{S}^1 := \mathbb{R}/(2\pi\mathbb{Z})$ has tangent space $\mathbb{R}$. The quasi-metric $\mathcal{F}$ is extended by positive 1-homogeneity to the case $\|\dot{\mathbf{x}}\| \neq 1$, and the dual metric expression follows from (4), see also Appendix A. Consider a path $\eta : [0, T] \to \mathbb{R}^2$, parametrized at unit Euclidean speed, and abusively identify the unit vector $\dot{\eta}(s)$ with the corresponding direction in $\mathbb{S}^1$. Then the length of the orientation-lifted path $(\eta, \dot{\eta})$ measured w.r.t. the above metric $\mathcal{F}$ equals the following curvature penalized energy

$$\mathcal{L}_{\mathcal{F}}((\eta, \dot{\eta})) = \int_0^L c(\eta(s), \dot{\eta}(s))\, \mathcal{C}(\ddot{\eta}(s))\, \mathrm{d}s, \tag{9}$$

In addition, by construction of the metric $\mathcal{F}$, any orientation lifting $(\eta, \tilde{\eta}) : [0, T] \to \mathbb{R}^2 \times \mathbb{S}^1$ of the physical path η, distinct from the canonical one $(\eta, \dot{\eta})$, satisfies $\mathcal{L}_{\mathcal{F}}((\eta, \tilde{\eta})) = +\infty$. By minimizing the length (1, right) among arbitrary paths $\gamma = (\eta, \tilde{\eta})$, one thus automatically selects pairs of the form $\gamma = (\eta, \dot{\eta})$, and optimizes the curvature dependent energy (9).

The state dependent cost $c(\mathbf{x}, \theta)$ appearing in the metric definition (8) is arbitrary and user defined, but the curvature cost $\mathcal{C}(\dot{\theta})$ must presently be chosen among the following instances, related to the Reeds-Shepp car without reverse gear [22], the Euler-Mumford elastica curves [43], and Dubins's car [19]. For any $\kappa \in \mathbb{R}$, standing for the path curvature $\dot{\theta}$

$$\mathcal{C}^{\mathrm{RS}}(\kappa) := \sqrt{1 + (\xi\kappa)^2}, \qquad \mathcal{C}^{\mathrm{EM}}(\kappa) := 1 + (\xi\kappa)^2, \qquad \mathcal{C}^{\mathrm{D}}(\kappa) := \begin{cases} 1 & \text{if } |\xi\kappa| \leq 1, \\ +\infty & \text{else,} \end{cases} \tag{10}$$

where the parameter $\xi > 0$ modulates the amount of curvature penalization and has the dimension of a turning radius. Note that the metric (8) associated with (10, left) defines a model, referred to as the Reeds-Shepp *forward* model, which distinguishes itself from the original sub-Riemannian Reeds-Shepp model [50, 47] discussed in the previous paragraph by the absence of reverse gear, see the discussion in [22]. Finally, we also provide the more parametrizable models defined by the metrics of the form

$$\mathcal{F}_{(\mathbf{x},\theta)}(\dot{\mathbf{x}}, \dot{\theta}) = c(\mathbf{x}, \theta)\, \mathcal{C}\left(\xi(\mathbf{x}, \theta)(\dot{\theta} - \kappa(\mathbf{x}, \theta))\right) \quad \text{if } \dot{\mathbf{x}} = \mathbf{n}(\theta). \tag{11}$$

As before, the metric is extended 1-homogeneously to the case where $\mathbf{x}$ is positively collinear with $\mathbf{n}(\theta)$, and set to $+\infty$ otherwise. The fields $\xi : \mathbb{M} \to]0, \infty[$ and $\kappa : \mathbb{M} \to \mathbb{R}$, provided by the user, locally modulate the turning radius and the imbalance of the car. See Figure 17.

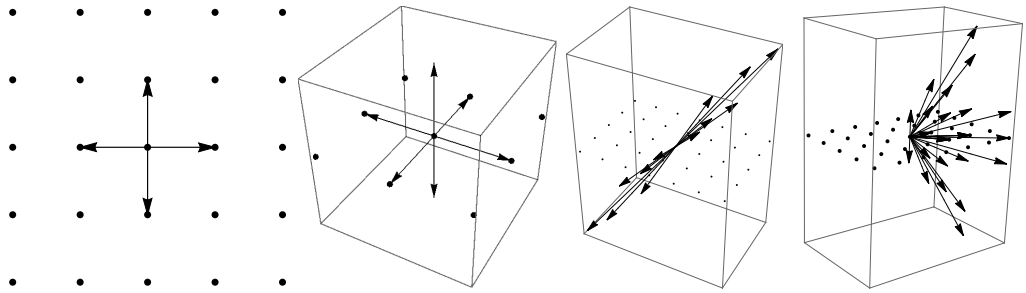

Figure 3: The stencil of a PDE discretization at a point $\mathbf{p}$, is the collection of neighbors of $\mathbf{p}$ in the discretization grid involved in the numerical approximation of the PDE operator by finite differences. We display the stencils used for the discretization of eikonal equations, associated with (left, center left) an isotropic metric in dimension 2 and 3, (center right) an anisotropic Riemannian metric, (right) the Euler-Mumford elastica model.

1.3 An Eulerian and Causal Discretization of the Eikonal Equation

The HFM software relies on an *Eulerian* and *causal* discretization of the eikonal equation (3), and on a Cartesian grid, based on papers [40, 39]. We briefly describe here these two design choices, and contrast them with the possible alternatives.

Numerical solvers of the time optimal control problem (2) can be of either *semi-Lagrangian* or *Eulerian* nature. Semi-Lagrangian schemes are a discretization of Bellman's optimality principle, which expresses that the characterization (2, left) of the problem solution u still holds if the global domain boundary $\partial\Omega$ is replaced with the boundary $\partial V(\mathbf{p})$ of an arbitrary (in practice chosen small) neighborhood $V(\mathbf{p}) \subset \Omega$ of the current point $\mathbf{p}$. A continued line of research has been devoted to this approach since the seminal work [61], see the paragraph on causality below. The HFM software does not follow this popular route, but relies in contrast on an Eulerian scheme, directly expressing the eikonal PDE (3) at the discrete level using finite differences, similarly to [51, 4]. This approach requires an exact or approximate representation of the dual metric in the following form: for any point $\mathbf{p} \in \overline{\Omega}$, and any co-vector $\hat{\mathbf{p}} \in \mathbb{E}^*$

$$\mathcal{F}^*_{\mathbf{p}}(\hat{\mathbf{p}})^2 \approx \sum_{1\leq i\leq I} \alpha_i(\mathbf{p}) \max\{0, \langle \hat{\mathbf{p}}, \dot{\mathbf{e}}_i\rangle\}^2, \tag{12}$$

where $\alpha_i = \alpha_i(\mathbf{p}) \geq 0$ and $\dot{\mathbf{e}}_i = \dot{\mathbf{e}}_i(\mathbf{p}) \in \mathbb{Z}^d$ are weights and offsets respectively, with index $1 \leq i \leq I$. The dependency of the offset $\dot{\mathbf{e}}_i = \dot{\mathbf{e}}_i(\mathbf{p})$ w.r.t. the base point $\mathbf{p}$ is usually omitted in equations such as (12), to alleviate notations. The design of these weights and offsets is non-trivial, see Section 2 and [40, 39], and expressions slightly more general than (12) are often needed as well (14). Figure 3 displays some for the stencils (collection of offsets) used in the numerical scheme for various isotropic and anisotropic metrics. The eikonal PDE (3), is then discretized under the following form

$$\forall \mathbf{p} \in X, \sum_{1\leq i\leq I} \alpha_i(\mathbf{p}) \max\{0, U(\mathbf{p}) - U(\mathbf{p} - h\dot{\mathbf{e}}_i)\}^2 = h^2, \qquad \forall \mathbf{p} \in \partial X,\ U(\mathbf{p}) = \sigma(\mathbf{p}). \tag{13}$$

where $h > 0$ is the discretization scale, and X and ∂X are subsets of the Cartesian grid $h\mathbb{Z}^d$ devoted to discretizing Ω and $\partial\Omega$ respectively. Note, crucially, that the numerical scheme (13, left) is a non-decreasing function of the finite differences $(U(\mathbf{p}) - U(\mathbf{p} - h\dot{\mathbf{e}}))_{\dot{\mathbf{e}}\in\mathbb{Z}^d}$. This property, referred to as monotony or degenerate ellipticity, is essential in establishing the well posedness and stability of the discretized problem, see [45]. The implementation of the HFM library is slightly more flexible than (13), allowing for e.g. axis dependent grid scales, the introduction of obstacles in the domain, and second order accuracy.

Causality is a property of the discretization scheme which reflects the deterministic nature of the original control problem, and enables it to be solved in a single pass using the fast marching

algorithm/dynamic programming principle, see Section 3.1. In the context of Eulerian schemes, causality means that the numerical scheme is a function of the *positive parts* of the finite differences $(U(\mathbf{p}) - U(\mathbf{p} - h\dot{\mathbf{e}}))_{\dot{\mathbf{e}} \in \mathbb{Z}^d}$, which is indeed the case in (13, left). See Definition 2.1 below for a formal definition, and [58] for a counterpart in the context of semi-Lagrangian schemes. At the continuous level, $u(\mathbf{p})$ can be regarded as the arrival time at $\mathbf{p} \in \Omega$ of a front originating from the domain boundary $\partial\Omega$ at the given time σ, and propagating with a speed locally dictated by the metric $\mathcal{F}$. The continuous counterpart of the causality property is the fact that the front arrival time at a given point only depends on the earlier arrival times, and not on the future ones. In the context of shortest paths on graphs, causality amounts to the positivity of the edge weights, which similarly enables Dijkstra's single pass algorithm. Note that monotone but *non-causal* discretization schemes can also be numerically solved, using a variety of iterative methods such as Fast-Sweeping [65] or (adaptive) Gauss-Seidel iterations [8, 4]. Their practical usage can however be tedious, in particular with the strongly anisotropic and inhomogeneous metrics encountered in image processing, which lead to long and unpredictable computation times [6].

Designing causal discretization schemes for anisotropic metrics is a non-trivial task, which has been the subject of a continued line of research in the semi-Lagrangian setting [29, 27, 1], following the discovery [58, 62] that it is related to a geometrical property of the discretization stencils. The works [35, 36] assume a Cartesian grid discretization and use, similarly to here but in the semi-Lagrangian setting, tools from lattice geometry to design reasonably small causal stencils for (strongly) anisotropic metrics. In the Eulerian setting, the design of causal schemes for non-isotropic metrics has in contrast been overlooked until the recent works [40, 39], which are at the foundation of the proposed software.

1.4 Applications to Image Processing and Motion Planning

Minimal path methods have numerous applications [57, 48], of which we can only give a glimpse. We focus here on applications which can directly benefit from the numerical methods provided within the proposed software, and in particular from paths globally minimizing a curvature dependent energy.

Motion planning. When preparing the motion of an automatic vehicle in a known environment, or the movement of an articulated machine such as a robotic arm, using the path globally minimizing a well chosen energy has obvious advantages [30]. A variety of methods for path optimization exist, such as local optimization from an initial guess, geodesic shooting, stochastic exploration, or other techniques based on a fine geometrical description of the domain [7]. Approaches based on the eikonal PDE have incomparable flexibility, and a guarantee of global optimality. Their main drawbacks are their cost for the complex models, due to the curse of dimensionality, and until recently the inability to handle non-holonomic constraints. The HFM library solves the latter issue, and for instance allows to introduce curvature penalization in the energy functional, so as to minimize the effort on the machine structure. Hard constraints can also be accounted for, such as vehicles with a bounded turning radius. In the proof of concept work [41], the HFM library is used to optimize a surveillance system for detecting an enemy vehicle, subject to non-holonomic constraints, in a worst case scenario.

Image Segmentation. Numerous imaging processing methods involve *paths*, which can represent the contours of two dimensional regions, the centerlines of tubular structures, or white fibers in dMRI scans of the brain. Various path constructions exist, such as fiber following methods based on ODEs, often enhanced with stochastic perturbations [53]. Another convenient selection principle is to select a path minimizing an adequate energy, which in practice is obtained by local optimization [28], or by solving an eikonal equation [16]. The latter approach, chosen in this paper, has the appealing guarantee that the globally optimal path is found, but it was historically more limited in the type of

energies that could be minimized. The HFM library intends to alleviate this constraint, by enabling the efficient computation of globally optimal Riemannian geodesics, as well as paths minimizing a family of curvature dependent energies. We refer to [48], for an in-depth overview of the uses of minimal paths in image processing, and devote the following paragraphs to works which specifically investigate the use of anisotropic (Riemannian or Finslerian) or non-holonomic (curvature penalizing) metrics in image processing.

Riemannian metrics were first applied in [6] to the segmentation of image regions and of tubular structures. The metric tensors are built in a preliminary step, based on a local filtering of the image, and are intended to guide the minimal paths along the image structures of interest; their design remains a challenging task, which is application dependent, requires expert knowledge, and remains the object of active research. Activity in the field [12, 15] was renewed after the development of fast Riemannian eikonal equation solvers [36]. *Finslerian metrics* allow path length to be measured differently depending on the direction, by e.g. augmenting a Riemannian metric with an asymmetric linear term [49]. They have important applications in image segmentation [34, 13]. An efficient discretization scheme exists for two dimensional Finslerian eikonal equations [35, 37], but for technical reasons[6] it is not presently implemented in the HFM library.

Path energies featuring second order terms, such as *curvature*, are completely natural in image segmentation [28]. Curvature penalization is for instance a good prior for region segmentation in noisy images, in particular if the objects of interest are convex [14]. Another use case is tubular structure segmentation in images of the retina, which feature complex overlays of vessels: curvature penalization helps eliminate "shortcuts" where the extracted path runs along the concatenation of several distinct intersecting vessels [4]. Finally, white fiber tractography in dMRI scans of the brain is a promising application field: the curvature penalty adds inertia to the paths, preventing them from getting lost at fiber crossings [22]. Global optimization of path energies featuring curvature and higher order tems, using dynamic programming, was first investigated in [59, 32]. We do believe that the approach used in the HFM library lies on firmer grounds, which are presented mathematically in Section 1.2, Section 2.4, and which history is briefly described below. The first step was to introduce the configuration space of all positions (in the image domain) and orientations [46]. Among other things, this allows to design the cost function(s) involved in the path energy using an orientation sensitive filtering of the image, based on e.g. wavelets [21] or Gabor filters. The second step is to relate the second order energy model of interest with a non-holonomic metric on the configuration space, see [4, 14] for the Reeds-Shepp and Euler-Mumford models respectively. The final step is to design a causal discretization scheme so as to solve the resulting eikonal PDE (possibly slightly relaxed) more efficiently, which was addressed in [52, 22, 40, 39].

Outline. We discuss in Section 2 the models that can be addressed with our numerical method, and describe in detail our discretization of the related eikonal PDEs. Section Section 3 describes the fast marching algorithm, and the related methods of geodesic backtracking and sensitivity analysis. Numerical experiments are presented in Section 4.

2 Expression of the Hamiltonian

In this section, we describe the discretization of several generalized eikonal equations, using an original and specific representation of the corresponding Hamiltonian, as announced in the introduction (12). These implementation principles are at the foundation of our numerical method, but are invisible

[6] It uses a semi-Lagrangian discretization, in contrast with the Eulerian discretization chosen for the HFM library, see Section 1.3.

from the top-level interface of the HFM library. They are of interest to curious users and to those who would like to extend the core C++ library with additional models and features.

In the rest of this paper we denote by $\mathcal{H}_{\mathbf{p}} := (\mathcal{F}^*_{\mathbf{p}})^2$, $\mathbf{p} \in \overline{\Omega}$, the squared dual metric, which is *abusively* referred to as the Hamiltonian. Note that, in classical mechanics, the Hamiltonian is defined as $\frac{1}{2}\mathcal{H}_{\mathbf{p}}$, but the multiplicative factor $1/2$ tends to introduce unnecessary clutter in both the description and the implementation of our numerical method. Our approach requires the Hamiltonian to be representable, exactly or approximately (in which case a consistency error is introduced), in the following general form, which generalizes (12). For any point $\mathbf{p} \in \Omega$ and any co-vector $\hat{\mathbf{p}} \in \mathbb{E}^*$

$$\mathcal{H}_{\mathbf{p}}(\hat{\mathbf{p}}) \approx \max_{1 \le k \le K} \left(\sum_{1 \le i \le I} \alpha_{ik} \langle \hat{\mathbf{p}}, \dot{\mathbf{e}}_{ik} \rangle_+^2 + \sum_{1 \le j \le J} \beta_{jk} \langle \hat{\mathbf{p}}, \dot{}_{jk} \rangle^2 \right), \tag{14}$$

where $a_+ := \max\{0, a\}$ for any $a \in \mathbb{R}$. The integers I, J, K are meta-parameters which are fixed for each class of metric. In contrast, the weights $\alpha_{ik}, \beta_{jk} \ge 0$ and the offsets $\dot{\mathbf{e}}_{ik}, \dot{}_{jk} \in \mathbb{Z}^d$ implicitly depend on the current point $\mathbf{p} \in X$ of the domain, where $1 \le i \le I$, $1 \le j \le J$, and $1 \le k \le K$. Let us mention that several strategies, discussed in Section 3.1.3, are used to limit the memory footprint of storing the local parameters $\alpha_{ik}, \beta_{jk}, \dot{\mathbf{e}}_{ik}, \dot{}_{jk}$ of the Hamiltonian representation (14, rhs).

The eikonal equation (3) is equivalently stated in terms of the Hamiltonian evaluated on a continuous function's differential: $\mathcal{H}_{\mathbf{p}}(\mathrm{d}u(\mathbf{p}))$ where $\mathbf{p} \in \Omega$. The finite-difference approximation $HU(\mathbf{p})$ of this quantity is obtained by inserting the following first order upwind approximations in the right-hand side of (14)

$$\langle \mathrm{d}U(\mathbf{p}), \dot{\mathbf{e}} \rangle_+^2 \approx h^{-2} \max\{0,\, U(\mathbf{p}) - U(\mathbf{p} - h\dot{\mathbf{e}})\}^2, \tag{15}$$

$$\langle \mathrm{d}U(\mathbf{p}), \dot{\mathbf{e}} \rangle^2 \approx h^{-2} \max\{0,\, U(\mathbf{p}) - U(\mathbf{p} - h\dot{\mathbf{e}}),\, U(\mathbf{p}) - U(\mathbf{p} + h\dot{\mathbf{e}})\}^2. \tag{16}$$

We denoted by $U : X \cup \partial X \to \mathbb{R}$ a discrete map, where X and ∂X are disjoint finite subsets of the Cartesian grid $h\mathbb{Z}^d$ of scale $h > 0$, devoted to approximating Ω and $\partial\Omega$. By convention, if needed, U is extended by $+\infty$ outside of its domain so as to implement outflow boundary conditions. The eikonal Equation (3) is thus discretized in the form

$$\forall \mathbf{p} \in X,\ HU(\mathbf{p}) = 1, \qquad\qquad \forall \mathbf{p} \in \partial X,\ U(\mathbf{p}) = \sigma(\mathbf{p}). \tag{17}$$

To be more specific, our implementation requires the discretization domain $X \cup \partial X$ to be box shaped. It also allows for various types of periodic boundary conditions, devoted to e.g. the commonly used manifolds $\mathbb{R}^2 \times \mathbb{S}^1$ and $\mathbb{R}^3 \times \mathbb{S}^2$, and for axis-dependent grid scales, such as a physical scale and an angular scale in the previous case. See Section D for more detail on the discretization grid conventions.

Before turning to specific models, we formally define the properties of monotony and causality, already mentioned in the introduction Section 1.3. These properties enable solving the system (17) in a single pass using the fast marching algorithm, see Section 3.1.

Definition 2.1. *A numerical scheme on a finite set Z is a map $H : Z \times \mathbb{R} \times \mathbb{R}^Z \to \mathbb{R}$. It is said*

- *Monotone iff H is non-decreasing with respect to the second and (each of the) third variables.*
- *Causal iff H only depends on the positive part of the third variable(s).*

One denotes for any $U : Z \to \mathbb{R}$ and any $\mathbf{p} \in Z$

$$HU(\mathbf{p}) := H\big(\mathbf{p}, U(\mathbf{p}), (U(\mathbf{p}) - U(\mathbf{q}))_{\mathbf{q} \in Z}\big).$$

For notations to match between Definition 2.1 and (17), one must set $Z := X \cup \partial X$, and define $HU(\mathbf{p}) := U(\mathbf{p})$ for all $\mathbf{p} \in \partial X$. By design, the properties of monotony and causality are obeyed by any numerical scheme obtained by inserting the upwind finite differences (15) in an expression of the form (14), namely

$$HU(\mathbf{p}) := h^{-2} \max_{1\le k\le K} \Big(\sum_{1\le i\le I} \alpha_{ik} \max\big\{0,\ U(\mathbf{p}) - U(\mathbf{p} - h\dot{\mathbf{e}}_{ik})\big\}^2 \tag{18}$$
$$+ \sum_{1\le j\le J} \beta_{jk} \max\big\{0,\ U(\mathbf{p}) - U(\mathbf{p} - h\dot{\imath}_{jk}),\ U(\mathbf{p}) - U(\mathbf{p} + h\dot{\imath}_{jk})\big\}^2 \Big).$$

2.1 Dijkstra's Algorithm

We show that, with a special choice of parameters, our discretization scheme (17) reduces to the classical problem of finding shortest paths on a graph with non-negative edge lengths. In addition, the fast marching algorithm reduces to Dijkstra's method in that case. Indeed, assume that $I = 1$ and $J = 0$, while $K \ge 1$ remains arbitrary. Then $HU(\mathbf{p}) = 1$ rewrites as

$$\max_{1\le k\le K} \left(\frac{U(\mathbf{p}) - U(\mathbf{p} - h\dot{\mathbf{e}}_k)}{l_k(\mathbf{p})} \right)_+^2 = 1, \qquad \text{equivalently} \quad U(\mathbf{p}) = \min_{1\le k\le K} l_k(\mathbf{p}) + U(\mathbf{p} - h\dot{\mathbf{e}}_k).$$

We denoted $l_k := 1/\sqrt{\alpha_{1k}}$ and $\dot{\mathbf{e}}_k := \dot{\mathbf{e}}_{1k}$, which both (sometimes implicitly) depend on the current point $\mathbf{p}$. Almost any optimal control problem can be addressed with Dijkstra's algorithm, by using sufficiently wide stencils. This even includes problems involving curvature or torsion dependent energies [59]. We advocate however for more clever designs of the weights and offsets, as discussed below for several metric structures of interest, so as to improve the accuracy of the numerical results.

2.2 Isotropic Metrics

We recall in this subsection the Eulerian discretization of isotropic eikonal equations, which dates back to [51]. As discussed in the general introduction, isotropic metrics are (by definition) locally proportional to the Euclidean norm $\|\cdot\|$ on the ambient space $\mathbb{E} := \mathbb{R}^d$: one has $\mathcal{F}_{\mathbf{p}}(\dot{\mathbf{p}}) = c(\mathbf{p})\|\dot{\mathbf{p}}\|$, for any point $\mathbf{p} \in \overline{\Omega}$ and any vector $\dot{\mathbf{p}} \in \mathbb{E}$, where $c : \overline{\Omega} \to]0, \infty[$ is a cost function provided by the user. The dual metric reads $\mathcal{F}^*_{\mathbf{p}}(\hat{\mathbf{p}}) = c(\mathbf{p})^{-1}\|\hat{\mathbf{p}}\|$, for any co-vector $\hat{\mathbf{p}} \in \mathbb{E}^*$.

The Hamiltonian is representable exactly in the form (14), using only d scalar products ($I = 0$, $J = d$, $K = 1$). Denoting by $\dot{\mathbf{e}}_i = (0, \cdots, 0, 1, 0, \cdots, 0)$ the unit vector directed along the i-th coordinate axis, for any $1 \le i \le d$, one has indeed

$$\mathcal{H}_{\mathbf{p}}(\hat{\mathbf{p}}) = c(\mathbf{p})^{-2}\|\hat{\mathbf{p}}\|^2 = c(\mathbf{p})^{-2} \sum_{1\le i\le d} \langle \hat{\mathbf{p}}, \dot{\mathbf{e}}_i\rangle^2. \tag{19}$$

Note that, in this special case, the offsets $(\dot{\mathbf{e}}_i)_{i=1}^d$ do not depend on the current base point $\mathbf{p} \in \Omega$. For completeness, we write the full form of the discretization scheme, specializing (18) and recovering [51]. For any $U : X \cup \partial X \to \mathbb{R}$ and any grid point $\mathbf{p} \in X$

$$HU(\mathbf{p}) = (c(\mathbf{p})h)^{-2} \sum_{1\le i\le d} \max\{0,\, U(\mathbf{p}) - U(\mathbf{p} - h\dot{\mathbf{e}}_i),\, U(\mathbf{p}) - U(\mathbf{p} + h\dot{\mathbf{e}}_i)\}^2. \tag{20}$$

Several numerical schemes often exist for discretizing a given eikonal equation. Substituting $\langle \mathbf{p}, \dot{\mathbf{e}}_i\rangle^2 = \langle \mathbf{p}, \dot{\mathbf{e}}_i\rangle_-^2 + \langle \mathbf{p}, \dot{\mathbf{e}}_i\rangle_+^2$ in (19) yields for instance the alternative discretization [56]

$$\widetilde{HU}(\mathbf{p}) = (c(\mathbf{p})h)^{-2} \sum_{1\le i\le d} \sum_{\tau\in\{-1,1\}} \max\{0,\, U(\mathbf{p}) - U(\mathbf{p} - \tau h\dot{\mathbf{e}}_i)\}^2. \tag{21}$$

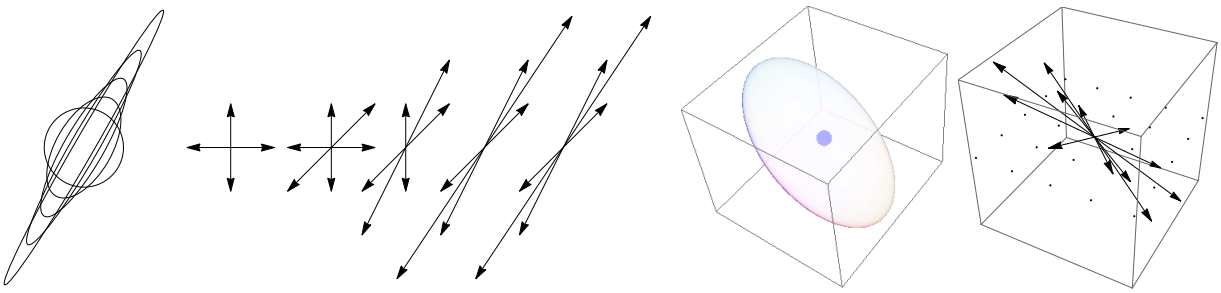

Figure 4: For a given positive definite tensor D, we display the unit ball $\{\dot{\mathbf{p}}; \|\dot{\mathbf{p}}\|_{D^{-1}} \leq 1\}$ and the collection of offsets $(\dot{\mathbf{e}}_i)_{1 \leq i \leq d'}$ appearing in Proposition 2.2. Left: several examples in dimension two. Right: a three dimensional case.

According to [56] this scheme is more diffusive than (20) near the cut locus (the points of non-differentiability of the distance map). However it has the advantage of being differentiable with respect to the entries of the discrete map U, which is welcome if the eikonal PDE solution is part of a larger optimization problem, see Section 4.2.

Diagonal metrics are an elementary generalization of isotropic metrics, in which the propagation cost is possibly distinct along each coordinate axis. The metric, and dual metric, read

$$\mathcal{F}_{\mathbf{p}}(\dot{\mathbf{p}}) = \sqrt{\sum_{1 \leq i \leq d} c_i(\mathbf{p})^2 \langle \dot{\mathbf{e}}_i, \dot{\mathbf{p}} \rangle^2}, \qquad \mathcal{F}^*_{\mathbf{p}}(\hat{\mathbf{p}}) = \sqrt{\sum_{1 \leq i \leq d} c_i(\mathbf{p})^{-2} \langle \hat{\mathbf{p}}, \dot{\mathbf{e}}_i \rangle^2}. \tag{22}$$

Thus $\mathcal{H}_{\mathbf{p}}(\hat{\mathbf{p}}) := \mathcal{F}^*_{\mathbf{p}}(\hat{\mathbf{p}})^2$ naturally has the required form (14), with again $I = 0$, $J = d$, $K = 1$.

2.3 Riemannian Metrics, and Sub-Riemannian Approximations

Riemannian metrics are the most common class of non-isotropic metrics. They are determined by a field $\mathcal{M} : \overline{\Omega} \to \mathrm{S}^{++}(\mathbb{E})$ of positive definite tensors, and take the form $\mathcal{F}_{\mathbf{p}}(\dot{\mathbf{p}}) := \|\dot{\mathbf{p}}\|_{\mathcal{M}(\mathbf{p})}$, where we recall $\|\dot{\mathbf{p}}\|_M := \sqrt{\langle M\dot{\mathbf{p}}, \dot{\mathbf{p}} \rangle}$. In differential geometry, the natural distance on a sub-manifold of $\mathbb{R}^n$ can be described by a Riemannian metric on a parametrization domain. In image segmentation, the Riemannian tensors may stem naturally from the data [25], or be creatively designed based on some local image analysis [6]. In order to discretize Riemannian eikonal equations, we introduce an adequate decomposition of positive definite tensors. The condition number of $D \in \mathrm{S}^{++}(\mathbb{E})$, where $\mathrm{S}^{++}(\mathbb{E})$ denotes the set of positive definite tensors, is defined by

$$\mathrm{Cond}(D) := \sqrt{\|D\| \|D^{-1}\|}. \tag{23}$$

Proposition 2.2. *Let $D \in \mathrm{S}^{++}(\mathbb{E})$, where $\mathbb{E} := \mathbb{R}^d$, and let $d' := d(d+1)/2$. Then there exists non-negative weights $\alpha_i \geq 0$ and integer offsets $\dot{\mathbf{e}}_i \in \mathbb{Z}^d$, where $1 \leq i \leq d'$, such that*

$$D = \sum_{1 \leq i \leq d'} \alpha_i \, \dot{\mathbf{e}}_i \otimes \dot{\mathbf{e}}_i. \tag{24}$$

Furthermore, this decomposition can be chosen such that $\|\dot{\mathbf{e}}_i\| \leq \mathrm{Cond}(D)^{\alpha}$, for all $1 \leq i \leq d'$, where $\alpha := d - 1$. In dimension $d = 3$, one has the improved estimate $\alpha = 1$.

We assume in the following that the decomposition (24) is the one provided by Voronoi's first reduction of the quadratic form $\dot{\mathbf{p}} \mapsto \langle \dot{\mathbf{p}}, D\dot{\mathbf{p}} \rangle$, which enjoys these properties.

We refer to [39] for the proof of Proposition 2.2, and to [54] for more background on Voronoi's theory and the field of additive lattice geometry. Our numerical codes devoted to Riemannian metrics

do compute the tensor decomposition (24) at each gridpoint, using a simple and efficient algorithm due to Selling [55], see Appendix B. This algorithm applies in dimension $d \in \{2,3\}$ only, but we hope to address the case $d \in \{4,5\}$ in the future using other techniques. Our discretization scheme directly involves the offsets $(\mathbf{e}_i)_{i=1}^{d'}$, hence their norm should be as small as possible. The chosen construction is proved in [38] to be optimal in this regard, in dimension $d = 2$, and sharp worst and average case estimates of the stencil radius are also established. The tensor decomposition (24) can be used to discretize PDEs other than eikonal equations, such as anisotropic diffusion [23].

Proposition 2.2 yields an exact representation of Riemannian Hamiltonians in the desired form (14), using $d' := d(d+1)/2$ scalar products ($I = 0$, $J = d'$, $K = 1$). Indeed, for any point $\mathbf{p} \in \overline{\Omega}$ and any co-vector $\hat{\mathbf{p}} \in \mathbb{E}^*$, one has denoting by $\mathcal{D}(\mathbf{p}) := \mathcal{M}(\mathbf{p})^{-1}$ the inverse of the Riemannian metric tensor

$$\mathcal{H}_{\mathbf{p}}(\hat{\mathbf{p}}) = \|\hat{\mathbf{p}}\|^2_{\mathcal{D}(\mathbf{p})} = \langle \hat{\mathbf{p}}, \mathcal{D}(\mathbf{p})\hat{\mathbf{p}} \rangle = \sum_{1 \leq i \leq d'} \alpha_i(\mathbf{p}) \, \langle \hat{\mathbf{p}}, \dot{\mathbf{e}}_i \rangle^2. \tag{25}$$

The weights $\alpha_i(\mathbf{p}) \geq 0$ and the offsets $\dot{\mathbf{e}}_i = \dot{\mathbf{e}}_i(\mathbf{p}) \in \mathbb{Z}^d$ are those appearing in the decomposition (24) of $\mathcal{D}(\mathbf{p})$. Specializing (18) yields

$$HU(\mathbf{p}) = \sum_{1 \leq i \leq d'} \alpha_i(\mathbf{p}) \max\{0, U(\mathbf{p}) - U(\mathbf{p} - h\dot{\mathbf{e}}_i), U(\mathbf{p}) - U(\mathbf{p} + h\dot{\mathbf{e}}_i)\}^2. \tag{26}$$

In comparison with the classical discretization (20) of isotropic eikonal equations, the offsets are point dependent: $\mathbf{e}_i = \mathbf{e}_i(\mathbf{p})$, and are a bit more numerous: d' instead of d. Interestingly, if the Riemannian metric tensor is proportional to the identity matrix, $\mathcal{M}(\mathbf{p}) = c(\mathbf{p})^2 \, \mathrm{Id}$ at some point $\mathbf{p} \in \Omega$, then one can show that (25) reduces to the previous numerical scheme (20) (in particular $d' - d$ of the coefficients $\alpha_i(\mathbf{p})$ vanish, while the others are equal to $c(\mathbf{p})^{-2}$). The offsets $(e_i)_{i=1}^{d'}$, introduced in Proposition 2.2 and used in our numerical scheme, are illustrated in Figure 4 for various two and three dimensional tensors.

Sub-Riemannian metrics can be regarded as degenerate Riemannian metrics, which metric tensors have some infinite eigenvalues. The inverse tensor $\mathcal{D} : \overline{\Omega} \to \mathrm{S}^+(\mathbb{E}^*)$ is well defined, positive semi-definite, but rank deficient. They define *non-holonomic* control models, in the sense that, at some points in space $\mathbf{p} \in \overline{\Omega}$, some directions of motion $\dot{\mathbf{p}} \in \mathbb{E}$ are forbidden, namely those outside of the span of $\mathcal{D}(\mathbf{p})$. Proper sub-Riemannian metrics should also obey a local controllability property, expressed in terms of commutators of vector fields, which is out of the scope of this paper, see [42]. The HFM library addresses sub-Riemannian metrics using a relaxation approach, involving a family $\mathcal{F}^\varepsilon$ of Riemannian metrics which tensors $\mathcal{M}_\varepsilon$ explode as some parameter $\varepsilon \to 0$. This technique is viable numerically thanks to the good behavior of our numerical scheme with strongly anisotropic metrics. In practice, tensors of condition number $\mathrm{Cond}(\mathcal{M}_\varepsilon) \approx 10$ offer a good compromise between (a) the sharpness of the sub-Riemannian relaxation, and (b) the size of the discretization stencils, see Proposition 2.2.

The Reeds-Shepp car model (with reverse gear) [50] is perhaps the most notorious example of a sub-Riemannian metric, also appearing in Petitot's description of the visual cortex V1 [47]. This model is posed on the manifold $\mathbb{M} := \mathbb{R}^2 \times \mathbb{S}^1$ of positions and orientations, and is described here for a positive relaxation parameter $\varepsilon > 0$. For any point $(\mathbf{x}, \theta) \in \mathbb{M}$ and any tangent vector $(\dot{\mathbf{x}}, \dot{\theta}) \in T_{(\mathbf{x},\theta)}\mathbb{M} = \mathbb{R}^2 \times \mathbb{R}$ we define following [52]

$$\mathcal{F}^\varepsilon_{(\mathbf{x},\theta)}(\dot{\mathbf{x}}, \dot{\theta})^2 = \|(\dot{\mathbf{x}}, \dot{\theta})\|^2_{\mathcal{M}_\varepsilon(\mathbf{x},\theta)} := c(\mathbf{x}, \theta)^2 \left(\langle \mathbf{n}(\theta), \dot{\mathbf{x}} \rangle^2 + \varepsilon^{-2} \langle \mathbf{n}(\theta)^\perp, \dot{\mathbf{x}} \rangle^2 + (\xi \dot{\theta})^2 \right), \tag{27}$$

where $c : \overline{\Omega} \to]0, \infty[$ is a given cost function, and $\xi > 0$ is a parameter homogeneous to a radius of curvature. We denoted by $\mathbf{n}(\theta) := (\cos\theta, \sin\theta)$ the unit vector of orientation $\theta \in \mathbb{S}^1$, and by $\mathbf{n}(\theta)^\perp$ the clockwise orthogonal vector. Lateral physical motions, in the direction of $\mathbf{n}(\theta)^\perp$, see their cost

strongly penalized by the ε^{-2} factor in (27), and are completely excluded in the limit sub-Riemannian model for which $\varepsilon = 0$. Our software also implements the higher dimensional generalization of this model on $\mathbb{R}^3 \times \mathbb{S}^2$, first considered in [22], as well as a "dual" variant introduced in [40] which penalizes non-planarity (akin to a torsion penalization) instead of curvature.

We can make explicit the tensor field $\mathcal{M}_\varepsilon : \mathbb{R}^2 \times \mathbb{S}^1 \to \mathrm{S}^{++}(\mathbb{R}^3)$ such that $\mathcal{F}^\varepsilon_{(\mathbf{x},\theta)}(\dot{\mathbf{x}}, \dot{\theta}) = \|(\dot{\mathbf{x}}, \dot{\theta})\|_{\mathcal{M}_\varepsilon(\mathbf{x},\theta)}$, for all $(\mathbf{x}, \theta) \in \mathbb{R}^2 \times \mathbb{S}^1$. The matrix $\mathcal{M}_\varepsilon(\mathbf{x}, \theta)$ is block-diagonal with structure $(2 \times 2,\ 1 \times 1)$, and reads

$$\mathcal{M}_\varepsilon(\mathbf{x}, \theta) = c(\mathbf{x}, \theta)^2 \left(\begin{array}{c|c} \mathbf{n}(\theta) \otimes \mathbf{n}(\theta) & \\ + \varepsilon^{-2}\, \mathbf{n}(\theta)^\perp \otimes \mathbf{n}(\theta)^\perp & \\ \hline & \xi^2 \end{array} \right).$$

The eigenvectors are $(\mathbf{n}(\theta), 0)$, $(\mathbf{n}(\theta)^\perp, 0)$, and $(\mathbf{0}, 1)$, with corresponding eigenvalues $c(\mathbf{x}, \theta)^2(1, \varepsilon^{-2}, \xi^2)$. Computing the inverse tensors $\mathcal{D}_\varepsilon(\mathbf{x}, \theta) := \mathcal{M}_\varepsilon(\mathbf{x}, \theta)^{-1}$, we obtain the following expression of the Hamiltonian: for any co-vector $(\hat{\mathbf{x}}, \hat{\theta}) \in T^*_{(\mathbf{x},\theta)}\mathbb{M}$

$$\mathcal{H}^\varepsilon_{(\mathbf{x},\theta)}(\hat{\mathbf{x}}, \hat{\theta}) = \|(\hat{\mathbf{x}}, \hat{\theta})\|^2_{\mathcal{D}_\varepsilon(\mathbf{x},\theta)} := c(\mathbf{x}, \theta)^{-2} \left(\langle \hat{\mathbf{x}}, \mathbf{n}(\theta) \rangle^2 + \varepsilon^2 \langle \hat{\mathbf{x}}, \mathbf{n}(\theta)^\perp \rangle^2 + (\hat{\theta}/\xi)^2 \right). \tag{28}$$

From this point, the HFM library applies the previous Riemannian discretization strategy (25), with a positive relaxation parameter. In practice, choosing $\varepsilon := 0.1$ yields good results. Note that the matrix $\mathcal{D}^\varepsilon_{(\mathbf{x},\theta)}$ of the sub-Riemannian Hamiltonian is rank deficient when $\varepsilon = 0$, hence Proposition 2.2 is not directly applicable to the limit model.

2.4 Curvature Penalized Models

This section is devoted to the computation of paths globally minimizing a curvature dependent energy, following [39], which is one of the main novelties brought by the HFM library. For that purpose, the curvature dependent energy of a planar path is reformulated as a the length of a lifted path in the three dimensional domain $\mathbb{M} := \mathbb{R}^2 \times \mathbb{S}^1$, defined with respect to a singular metric $\mathcal{F} : T\mathbb{M} \to [0, \infty]$, see Section 1.2. Said otherwise, the introduction of a second order path derivative in the cost function comes at the price of (i) an extra dimension $\mathbb{S}^1$, and (ii) a strongly (formally infinitely) anisotropic metric $\mathcal{F}$. This approach shares many similarities with the sub-Riemannian Reeds-Shepp model (27), the main difference being that the models considered in this section lack the ability to shift into reverse gear. We describe below the computation of the dual metric, and its approximation in the form (14) which is required for our discretization strategy.

The dual metric $\mathcal{F}^*$ to the models of interest is described in (8) as the solution to a one dimensional optimization problem. More precisely, one has

$$\mathcal{F}^*_{(\mathbf{x},\theta)}(\hat{\mathbf{x}}, \hat{\theta}) = c(\mathbf{x}, \theta)^{-1} f(\hat{x}, \hat{\theta}),$$

where $\hat{x} := \langle \hat{\mathbf{x}}, \mathbf{n}(\theta) \rangle \in \mathbb{R}$, and where $f : \mathbb{R}^2 \to [0, \infty[$ is defined as the following maximum

$$f(\hat{x}, \hat{\theta}) := \sup_{\dot{\theta} \in \mathbb{R}} \frac{\hat{x} + \hat{\theta}\dot{\theta}}{\mathcal{C}(\dot{\theta})}, \qquad \text{extremal when} \quad \hat{\theta}\mathcal{C}(\dot{\theta}) = (\hat{x} + \hat{\theta}\dot{\theta})\mathcal{C}'(\dot{\theta}). \tag{29}$$

We specialize, in the next equation, the optimality condition (29, right) for the curvature costs $\mathcal{C}^{\mathrm{RS}}$, $\mathcal{C}^{\mathrm{EM}}$ and $\mathcal{C}^{\mathrm{D}}$ associated to the Reeds-Shepp, Euler-Mumford, and (non-smooth) Dubins models, see (10). After simplification this condition respectively reads

$$\dot{\theta}\hat{x}\xi^2 = \hat{\theta}, \qquad \dot{\theta}^2\xi^2\hat{\theta} + 2\dot{\theta}\xi^2\hat{x} - \hat{\theta} = 0, \qquad \dot{\theta}\xi \in \{-1,\ 1,\ \infty\}.$$

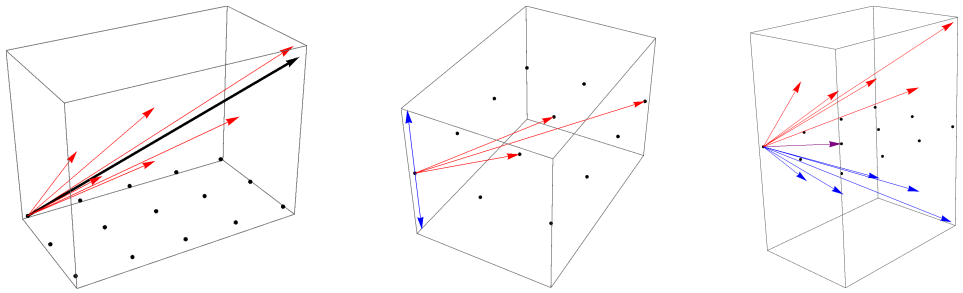

Figure 5: Left: Applying Proposition 2.3 to a vector (black) with $\varepsilon = 0.1$, in dimension 3, yields the red offsets. Center: Discretization stencil for the Reeds-Shepp model, red offsets obtained by applying Proposition 2.3 in dimension $d = 2$, blue offsets related to the angular dimension. Right: Stencil for the Dubins model, red and blue offsets obtained by applying Proposition 2.3 in dimension $d = 3$ to two distinct vectors.

Recall that $\xi > 0$ is a parameter, homogenous to a radius of curvature, which modulates the intensity of curvature penalization. Solving for $\dot\theta$, and inserting the optimal value in (29, left), one obtains the following expression of $f(\hat x, \hat\theta)$ respectively

$$\sqrt{\hat x_+^2 + (\hat\theta/\xi)^2}, \qquad \hat x + \sqrt{\hat x^2 + (\hat\theta/\xi)^2}, \qquad \max\{0,\, \hat x + \hat\theta/\xi,\, \hat x - \hat\theta/\xi\} \tag{30}$$

We refer to Appendix A for the details of the computations, and to [39] for a different proof.

Let us focus on the case (30, left) of the Reeds-Shepp *forward* model, which squared dual metric thus reads

$$\mathcal{H}^{\mathrm{RS}}_{(\mathbf{x},\theta)}(\hat{\mathbf{x}}, \hat\theta) = c(\mathbf{x},\theta)^{-2} \left(\langle \hat{\mathbf{x}}, \mathbf{n}(\theta)\rangle_+^2 + (\hat\theta/\xi)^2 \right). \tag{31}$$

The main difference with the classical Reeds-Shepp model (with reverse gear), which Hamiltonian is obtained by setting $\varepsilon = 0$ in (28), is the positive part in the physical contribution $\langle \hat{\mathbf{x}}, \mathbf{n}(\theta)\rangle_+^2$ to the Hamiltonian (as opposed to the angular contribution $(\hat\theta/\xi)^2$). This difference accounts for the lack of reverse gear in the present model. The discretization of this term is addressed in the next proposition, proved in [39], which introduces (similarly to the reversible case) a relaxation parameter $\varepsilon > 0$. We denote by $\mathrm{P}_{\mathbf{n}}$ the orthogonal projection onto the hyperplane orthogonal to a unit vector $\mathbf{n}$

$$\mathrm{P}_{\mathbf{n}} := \mathrm{Id} - \mathbf{n} \otimes \mathbf{n}.$$

Proposition 2.3. *Let $\mathbf{n} \in \mathbb{E} := \mathbb{R}^d$ and let $\varepsilon > 0$. Consider non-negative weights and offsets $(\alpha_i, \dot{\mathbf{e}}_i)_{i=1}^I \in (\mathbb{R}_+ \times \mathbb{E})^I$, obtained e.g. by Proposition 2.2, such that for all $\hat{\mathbf{x}} \in \mathbb{E}^*$ one has*

$$\langle \hat{\mathbf{x}}, \mathbf{n}\rangle^2 + \varepsilon^2 \| \mathrm{P}_{\mathbf{n}}\, \hat{\mathbf{x}}\|^2 = \sum_{1\le i\le I} \alpha_i \langle \hat{\mathbf{x}}, \dot{\mathbf{e}}_i\rangle^2.$$

Assume that $\langle \mathbf{n}, \dot{\mathbf{e}}_i\rangle \ge 0$ (otherwise replace $\dot{\mathbf{e}}_i$ with its opposite), for all $1 \le i \le I$. Then

$$\langle \hat{\mathbf{x}}, \mathbf{n}\rangle_+^2 \le \sum_{1\le i\le I} \alpha_i \langle \hat{\mathbf{x}}, \dot{\mathbf{e}}_i\rangle_+^2 \le \langle \hat{\mathbf{x}}, \mathbf{n}\rangle_+^2 + \varepsilon^2 \| \mathrm{P}_{\mathbf{n}}\, \hat{\mathbf{x}}\|^2.$$

The offsets (e_i) of corresponding to Proposition 2.3, as well as to the Reeds-Shepp and Dubins model discussed below, are illustrated in Figure 5.

Denote by $\alpha_i^\varepsilon(\theta)$ and $\mathbf{e}_i^\varepsilon(\theta)$ the weights and offsets obtained by applying Proposition 2.3 to the vector $\mathbf{n}(\theta) \in \mathbb{R}^2$ and a suitably small $\varepsilon > 0$, where $1 \le i \le I$ and $I = d' = d(d+1)/2 = (2\times 3)/2 = 3$,

see Proposition 2.2. Then we approximate the Hamiltonian (31) of the Reeds-Shepp forward model in the form (14) with parameters $I = 3$, $J = 1$, $K = 1$, as follows

$$\mathcal{H}^{\mathrm{RS}}_{(\mathbf{x},\theta)}(\hat{\mathbf{x}},\hat{\theta}) \approx c(\mathbf{x},\theta)^{-2}\Big(\sum_{1\le i\le I} \alpha_i^\varepsilon(\theta)\langle\hat{\mathbf{x}},\dot{\mathbf{e}}_i^\varepsilon(\theta)\rangle_+^2 + (\hat{\theta}/\xi)^2\Big).$$

For completeness, we write the full discretization scheme for this model, specializing (18). Let $h > 0$ be the gridscale and let $\mathbb{M}_h := h\mathbb{Z}^2 \times (h\mathbb{Z}/2\pi\mathbb{Z})$. We assume here that $2\pi/h$ is an integer[7], so that the discretization of the periodic circle $\mathbb{S}^1$ makes sense. For any $U : \mathbb{M}_h \to \mathbb{R}$ and any $(\mathbf{x},\theta) \in \mathbb{M}_h$

$$\begin{aligned} H^{\mathrm{RS}}_\varepsilon U(\mathbf{x},\theta) :=& (c(\mathbf{x},\theta)h)^{-2}\Big(\sum_{1\le i\le I} \alpha_i^\varepsilon(\theta)\max\{0,\, U(\mathbf{x},\theta) - U(\mathbf{x}-h\dot{\mathbf{e}}_i^\varepsilon(\theta),\theta)\}^2 \\ &+ \xi^{-2}\max\{0,\, U(\mathbf{x},\theta) - U(\mathbf{x},\theta-h),\, U(\mathbf{x},\theta) - U(\mathbf{x},\theta-h)\}^2\Big). \end{aligned}$$

The Dubins Hamiltonian is discretized using a similar procedure. In view of (30, right) one has indeed

$$\mathcal{H}^{\mathrm{D}}_{(\mathbf{x},\theta)}(\hat{\mathbf{x}},\hat{\theta}) = c(\mathbf{x},\theta)^{-2}\max\{(\langle\hat{\mathbf{x}},\mathbf{n}(\theta)\rangle + \hat{\theta}/\xi)_+^2,\, (\langle\hat{\mathbf{x}},\mathbf{n}(\theta)\rangle - \hat{\theta}/\xi)_+^2\}. \tag{32}$$

Apply Proposition 2.3 to the three dimensional vectors $\mathbf{v}_+(\theta) := (\mathbf{n}(\theta),\xi^{-1})$ and $\mathbf{v}_-(\theta) := (\mathbf{n}(\theta),-\xi^{-1})$, and denote by $\alpha^\varepsilon_{+i}(\theta)$, $\alpha^\varepsilon_{-i}(\theta)$, $\dot{\mathbf{e}}^\varepsilon_{+i}(\theta)$ and $\dot{\mathbf{e}}^\varepsilon_{-i}(\theta)$ the corresponding weights and offsets, where $1 \le i \le I$ and $I = (3\times 4)/2 = 6$. Then we approximate (32) in the form (14) with parameters $I = 6$, $J = 0$, $K = 2$, as follows

$$\mathcal{H}^{\mathrm{D}}_{(\mathbf{x},\theta)}(\hat{\mathbf{x}},\hat{\theta}) \approx c(\mathbf{x},\theta)^{-2}\max\Big\{\sum_{1\le i\le I}\alpha^\varepsilon_{+i}(\theta)\,\langle\hat{\mathbf{p}},\dot{\mathbf{e}}^\varepsilon_{+i}(\theta)\rangle_+^2,\ \sum_{1\le i\le I}\alpha^\varepsilon_{-i}(\theta)\,\langle\hat{\mathbf{p}},\dot{\mathbf{e}}^\varepsilon_{-i}(\theta)\rangle_+^2\Big\}.$$

The final PDE discretization scheme is as usual obtained by specializing (18). The Euler-Mumford elastica model is addressed using a similar procedure, for which we refer to [39] due to space constraints.

3 Implementation

In this section, we provide the implementation details for the Fast-Marching algorithm, used to numerically solve our Eulerian and causal discretization (17) of the eikonal PDE. We strongly rely on the specific form (14) of the Hamiltonian approximation. We also discuss geodesic backtracking, and sensitivity analysis.

In order to keep notations simple, we assume that the Hamiltonian $\mathcal{H}$ of the model of interest is representable in the form (14) with parameters $J = 0$ and $K = 1$, whereas I remains arbitrary. The following arguments can be easily adapted to the general case where J and K are arbitrary as well. For any discretization point $\mathbf{p} \in X$, any co-vector $\hat{\mathbf{p}} \in \mathbb{E}^*$, and any discrete map U, the Hamiltonian and finite differences scheme are thus assumed to read

$$\mathcal{H}_{\mathbf{p}}(\hat{\mathbf{p}}) = \sum_{1\le i\le I}\alpha_i\langle\hat{\mathbf{p}},\dot{\mathbf{e}}_i\rangle_+^2, \qquad HU(\mathbf{p}) := h^{-2}\sum_{1\le i\le I}\alpha_i\left(U(\mathbf{p}) - U(\mathbf{p}-h\dot{\mathbf{e}}_i)\right)_+^2. \tag{33}$$

Recall that $a_+^2 := \max\{0,a\}^2$. The weights $\alpha_i = \alpha_i(\mathbf{p}) \ge 0$ and offsets $\dot{\mathbf{e}}_i = \dot{\mathbf{e}}_i(\mathbf{p}) \in \mathbb{Z}^d$, where $1 \le i \le I$, depend on the current point $\mathbf{p} \in X$. Our objective is numerically solve the eikonal

[7] In the HFM software, two distinct scales h_x and h_θ are used, and the latter is specified via the integer $n_\theta := 2\pi/h_\theta$.

equation (3). For that purpose, we rely on the fast marching algorithm to compute $U : X \cup \partial X \to]-\infty, \infty]$ obeying

$$\forall \mathbf{p} \in X,\ HU(\mathbf{p}) = 1, \qquad \forall \mathbf{p} \in \partial X,\ U(\mathbf{p}) = \sigma(\mathbf{p}), \tag{34}$$

where X and ∂X are finite subsets of $h\mathbb{Z}^d$, and where the boundary data $\sigma : \partial X \to]-\infty, \infty]$ is given. In the following, we denote by $N := \#(X \sqcup \partial X)$ the cardinality of the discrete domain.

3.1 Fast Marching

The fast marching algorithm is a generalization of Dijkstra's algorithm [18] for the computation of shortest paths on graphs. Said otherwise, it is a specialization of the dynamic programming principle. The general structure is unchanged, see Algorithm 1, and in particular each point of the domain $X \cup \partial X$ is tagged as Accepted only once, after what the corresponding value of the computed solution U is frozen. This algorithm is implemented in the function `void HamiltonFastMarching<T>::Run()` of the provided source code, located in file:
`HamiltonFastMarching-master/Headers/Base/HamiltonFastMarching.hxx`

Algorithm 1 Fast marching

Pre-compute the reversed stencils $(V[\mathbf{p}])_{\mathbf{p} \in X \cup \partial X}$, defined in (36).
Tag points of ∂X as Trial, and other points as Far.
Initialize $U : X \cup \partial X \to]-\infty, \infty]$ to the value $+\infty$ on X, and the boundary condition on ∂X.
While there remains Trial points.
    Find a Trial point $\mathbf{q}$ minimizing U.
    Tag $\mathbf{q}$ as Accepted, and call Post-process($\mathbf{q}$).
    For each neighbor $\mathbf{p} \in V[\mathbf{q}]$, either Far or Trial
        If $\mathbf{p}$ is Far, **then** tag it as Trial, and call Pre-process($\mathbf{p}$).
        Update $U(\mathbf{p})$, taking into account the value $U(\mathbf{q})$ of the last Accepted point.

The collection of points tagged as Trial can be regarded as a propagation front. Two additional sub-routines, appearing as Pre-process($\mathbf{p}$) and Post-process($\mathbf{q}$) in Algorithm 1, and referred to as pre-processing and post-processing, are called when a point enters and leaves the front respectively. By default, some memory management is performed at these points, see Section3.1.2 and Section 3.1.3, but additional tasks may optionally be plugged in as well. At the post-processing stage one may for instance check for stopping criteria, or slightly alter the value $U(\mathbf{p})$ before it is frozen using the (formally) second order HAFMM numerical scheme [57].

3.1.1 Stencils and Reversed Stencils

Our algorithm uses point dependent, adaptive stencils, hence we need to cautiously determine the dependency graph underlying the discretized problem (34). For that purpose, we introduce the forward stencil $\mathcal{V}(\mathbf{p}) \subset X \cup \partial X$, of an arbitrary interior point $\mathbf{p} \in X$, and its counterpart the reversed stencil $V[\mathbf{q}] \subset X$, for all $\mathbf{q} \in X \cup \partial X$. The forward stencil $\mathcal{V}(\mathbf{p})$ at $\mathbf{p} \in X$, collects all points $\mathbf{q} \in X \cup \partial X$ whose value $U(\mathbf{q})$ is involved in the definition $HU(\mathbf{p})$, see (33, right).

$$\mathcal{V}(\mathbf{p}) := \{\mathbf{q} = \mathbf{p} - h\dot{\mathbf{e}}_i(\mathbf{p});\ 1 \le i \le I,\ \mathbf{q} \text{ is visible from } \mathbf{p}\} \cap (X \cup \partial X), \tag{35}$$

The intersection with $X \cup \partial X$ is meant to exclude all points $\mathbf{q}$ which fall outside of the (box shaped) discrete domain. Outside of this domain, the unknown function U is by convention extended by $+\infty$, which amounts to implement outflow boundary conditions. The *visibility* constraint is a test ensuring

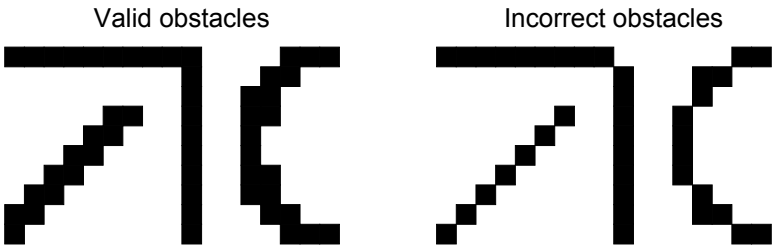

Figure 6: Obstacles are specified to the HFM software as boolean images. They must be "water tight": contiguous obstacle voxels must share a full $(d-1)$-dimensional face. Otherwise the numerical scheme will leak through, and the extracted minimal paths may jump across walls.

that the segment $[\mathbf{p}, \mathbf{q}]$ does not intersect any of the obstacles that may optionally be introduced within the domain. This condition is required because our adaptive stencils are often 5 to 10 pixels wide, hence they would otherwise jump over thin obstacles, such as walls in building maps which are usually drawn 1 pixel wide. See Figure 6.

The reversed stencil $V[\mathbf{q}] \subseteq X$ at $\mathbf{q} \in X \cup \partial X$, is defined by inverting the stencil connectivity graph

$$V[\mathbf{q}] := \{\mathbf{p} \in X;\ \mathbf{q} \in \mathcal{V}(\mathbf{p})\}. \tag{36}$$

Thus $V[\mathbf{q}]$ collects all points $\mathbf{p} \in X$ such that $HU(\mathbf{p})$ depends on $U(\mathbf{q})$. Forward and reverse stencils coincide in the classical discretization of isotropic fast marching (2.2), but differ in most other cases.

The numerical cost of constructing the stencils and the reversed stencils can be neglected in models for which these sets are independent of the current point (e.g. isotropic metrics Section 2.2), or depend only on a strict subset of its coordinates (e.g. the angular coordinate for curvature penalized models Section 2.4), by exploiting redundancy as discussed in Section 3.1.3 below. In other cases (e.g. Riemannian metrics Section2.3), the construction of the direct stencils $\mathcal{V}(\mathbf{p})$ has complexity[8] $\mathcal{O}(I)$ for each $\mathbf{p} \in X$, thus $\mathcal{O}(NI)$ overall. Once this is done, the reversed stencils are computed by sorting the pairs $\{(\mathbf{p}, \mathbf{q});\ \mathbf{p} \in X, \mathbf{q} \in \mathcal{V}(\mathbf{p})\}$ according to their second element, which costs $\mathcal{O}(IN \ln N)$.

Finally, let us mention that Equations (35) and (36) do not reflect our implementation entirely faithfully, for two reasons. First, we actually store the offsets instead of the points, e.g. $\dot{\mathbf{e}}_i(\mathbf{p}) \in \mathbb{Z}^d$ instead of $\mathbf{p} - h\dot{\mathbf{e}}_i(\mathbf{p})$ in (35). Second, the visibility and out of domain tests are performed while dynamic programming is running, and not during the initial stencil construction as suggested in (35).

3.1.2 Elementary Update

Consider a point $\mathbf{p} \in X$ tagged Trial, and which value $U(\mathbf{p})$ must be updated with respect to the Accepted points, as specified in the last line of Algorithm 1. In view of the Hamiltonian's expression (33), this means that the currently stored value $U(\mathbf{p})$ must be replaced with the largest solution $\lambda \in \mathbb{R}$ to the following univariate quadratic equation

$$\sum_{i \in \mathfrak{I}} \alpha_i(\mathbf{p}) \left(\lambda - U(\mathbf{q}_i)\right)^2 = 1, \qquad \text{where } \mathfrak{I} := \{1 \leq i \leq I;\ \mathbf{q}_i \in \mathcal{V}(\mathbf{p}),\ \mathbf{q}_i \text{ is Accepted}\}. \tag{37}$$

The neighbor points $\mathbf{q}_i := \mathbf{p} - h\dot{\mathbf{e}}_i(\mathbf{p})$, where $1 \leq i \leq I$, are those appearing in the numerical scheme $HU(\mathbf{p})$ at $\mathbf{p}$, see (33, right). Note that the index set $\mathfrak{I}$ excludes points that are out of the domain $X \cup \partial X$, are not visible from $\mathbf{p}$, or are tagged Trial or Far. One can show that, by construction of the fast marching algorithm[9], Equation (37) has two real roots, the smallest of which is a numerical

[8] We rely on basis reduction techniques to compute the tensor decomposition of Proposition 2.2, see Appendix B. Strictly speaking, their cost depends on the tensor condition number, but it grows so slowly that it can be regarded as constant for the applications of interest, see [44] for a complexity analysis of similar methods.

[9] Using the fact that points are tagged as Accepted sequentially, in the order of increasing values of U.

artifact, while the largest denoted λ^* satisfies by construction $\lambda^* \geq U(\mathbf{q}_i)$ for all $i \in \mathfrak{I}$, and defines the updated value of $U(\mathbf{p})$.

Some of our discretization schemes involve a rather large number of neighbors, for instance $I = 12$ for three dimensional Riemannian metrics, and $I = 27$ for the Euler-Mumford elastica model (choosing $K = 5$ in [39]). The latter figure may increase if the user requests more accuracy, see the discretization details in [39]. In order to maintain a $\mathcal{O}(1)$ complexity of the elementary update, we cache for each Trial point $\mathbf{p} \in X$ the coefficients (a, b, c) of the second degree polynomial $a\lambda^2 - 2b\lambda + c$ defined by (37). This cache is initialized as $(0, 0, -1)$ when Pre-process($\mathbf{p}$) is called. It is increased (component-wise) by $\alpha_i(\mathbf{p})\,(1, U(\mathbf{q}_i), U(\mathbf{q}_i)^2)$ each time a new neighbor $\mathbf{q}_i$ is Accepted. Finally, it is deleted when Post-process($\mathbf{p}$) is called.

The fast marching algorithm, similarly to Dijkstra's algorithm, must also maintain a queue of all Trial points, sorted by increasing values of U. For each point $\mathbf{p} \in X \sqcup \partial X$, queue maintenance operations include the insertion of the new key (when $\mathbf{p}$ is first tagged as Trial), a local re-ordering when the attached value $U(\mathbf{p})$ is modified (at most I times), and eventually the key removal (when $\mathbf{p}$ is tagged as Accepted). The overall complexity, using a heap based sorted data structure, is thus

$$\mathcal{O}(IN \ln N).$$

3.1.3 Memory Usage

The memory footprint of our algorithm is linear with respect to the number N of discretization points. The proportionality constant does matter however, since the image resolution of e.g. dMRI medical data is often huge. In addition, several applications introduce extra domain dimensions, accounting for e.g. the orientation, grayscale, or radius of some tubular structure to be extracted [15, 46], which multiplies the number of discretization points. For this reason, the HFM software implements several memory optimizations, discussed in this paragraph. We use the $\mathcal{O}(\cdot)$ notation[10] to denote space complexity, counted in bytes.

Our generalized fast marching solver uses wide stencils, which weights and offsets take significant memory space if stored indiscriminately. By default[11] we use weights $\alpha_i \in \mathbb{R}$ of "double" type, hence occupying 8 bytes[12], and offsets $\dot{\mathbf{e}}_i \in \mathbb{Z}^d$ with "signed char" components, hence occupying d bytes, where d is the domain dimension, and where $1 \leq i \leq I$. The storage cost of the forward and the reversed stencils is thus respectively

$$\mathcal{O}(IN_1(8+d)) \qquad\qquad \mathcal{O}(IN_2 d)$$

where N_1 and N_2 are the number of actually stored stencils. An extensive storage ($N_1 = N_2 = N$) of the numerical scheme stencils thus costs $\mathcal{O}(I(8+2d)N)$, whereas the memory footprint of the input speed function and of the output value function is $\mathcal{O}(2 \times 8N)$. For the Euler-Elastica model, which uses quite large stencils ($I = 27$ and $d = 3$), the space complexity $\mathcal{O}(378N)$ of our numerical method would thus vastly exceed the incompressible external cost $\mathcal{O}(16N)$ of input and output. Fortunately, we have developed two data management strategies for stencil storage which make the former cost negligible for this specific model, and severely limit it for others.

- *Data sharing.* The stencil offsets $\dot{\mathbf{e}}_i(\mathbf{p})$ of some models only depend on a strict subset $\overline{\mathbf{p}}$ of the coordinates of the current point $\mathbf{p} \in X$, e.g. none ($\overline{\mathbf{p}} = \emptyset$) for isotropic metrics Section 2.2, or the angular coordinate only ($\overline{\mathbf{p}} = \theta$ where $\mathbf{p} = (\mathbf{x}, \theta)$) for curvature penalized models Section 2.4 such as the Euler-Mumford elasticae. In addition, the weights are proportional to (the inverse

[10] This choice of notation is slightly abusive here, since there is *no* hidden proportionality constant.
[11] The C++ code is templated over these types.
[12] A Byte, the unit of computer memory capacity, consists of 8 bits.

square of) some user provided cost functions $c(\mathbf{p}, l)$, $\mathbf{p} \in \overline{\Omega}$, $1 \leq l \leq L$, where L is one or is a small positive integer. In summary, the discretization stencil structure takes the form

$$\dot{\mathbf{e}}_i = \dot{\mathbf{e}}_i(\overline{\mathbf{p}}), \qquad \alpha_i = c(\mathbf{p}, l_i(\overline{\mathbf{p}}))^{-2}\alpha_i(\overline{\mathbf{p}}). \tag{38}$$

We take advantage of this structure by storing only the offsets $\dot{\mathbf{e}}_i(\overline{\mathbf{p}}) \in \mathbb{Z}^d$, weights $\alpha_i(\overline{\mathbf{p}}) \geq 0$ and cost index $l_i(\overline{\mathbf{p}}) \in [\![1, L]\!]$ (omitted if $L = 1$) associated with the points of the lower dimensional domain $\overline{\mathbf{p}} \in \overline{X}$. We denoted $[\![1, L]\!] := [1, L] \bigcap \mathbb{N}$. Thus $N_1 = N_2 = \#(\overline{X}) \ll N$.

For instance, in the Euler-Mumford elastica model, $N_1 = N_2$ is the number of angular directions, which is typically around 60. Thus the huge footprint $\mathcal{O}(378N)$ of extensive stencil storage for this model, mentioned above, is with our strategy reduced to a negligible constant cost $\mathcal{O}(378 \times 60)$.

- *Recomputation.* For a number of models, either Riemannian or involving generalized curvature costs (11), the numerical scheme stencils lack redundancy, hence the previous data sharing strategy is not applicable. We thus implement an alternative strategy for memory usage reduction, which is to only store the direct stencils $\mathcal{V}(\mathbf{p})$ of the points currently tagged as Trial. Stencils must then be computed twice: (i) initially, so as to construct the reversed stencils $V[\mathbf{p}]$, and (ii) when a point label changes from Far to Trial. The storage cost of the forward stencils then becomes negligible, since $N_1 \approx N^{1-1/d} \ll N$. The additional computational complexity introduced by these recomputations is moderate - actually, it is dominated by the incompressible cost of maintaining the priority queue of all Trial points. The reversed stencils $V[\mathbf{p}]$ still must be stored all, thus $N_2 = N$, but fortunately they use only a fraction of the memory space of the forward ones (namely $d/(8 + d)$ where typically $d \in \{2, 3\}$).

  For the Euler-Mumford elastica model ($I = 27$ and $d = 3$), generalized in the sense of (11), the cost of extensive stencil storage $\mathcal{O}(378N)$ is reduced with this recomputation strategy to $\mathcal{O}(81N + 297N^{\frac{2}{3}})$. In comparison, the cumulated cost of the input parameter fields $\xi, \kappa, c : X \to \mathbb{R}$, and of the output, is $\mathcal{O}(32N)$.

Our algorithm also stores the coefficients of the second degree polynomials (37) associated with the Trial points, in addition to the stencils, which incurs the memory cost $\mathcal{O}(4N + 24N^{1-1/d})$ with the chosen implementation[13]. Finally, we chose to store the set $\mathfrak{I}$ of active neighbors of each point, defined in (37), which costs $\mathcal{O}(\lceil \ln_2 I \rceil N)$, for use in geodesic backtracking and sensitivity analysis.

3.2 Geodesic Extraction

Geodesic backtracking is done by solving an ODE (5) involving the distance map, which is numerically approximated in a preliminary step by solving the discretized eikonal PDE (34). Despite the apparent simplicity of numerical ODE integration, this task deserves caution. Naive implementations of ODE geodesic backtracking indeed suffer from artifacts such as paths (i) interrupted nearby obstacles before they reach the seed points, (ii) going past the seed points, or (iii) endlessly oscillating close to singularities of the distance map. These difficulties are particularly marked when using metrics which are (a) strongly inhomogeneous, (b) strongly anisotropic, leading to wide discretization stencils, and (c) non-holonomic and/or non-locally controllable, often leading to discontinuous (in addition to being non-differentiable) front arrival times.

The proposed software implements two robust numerical methods for geodesic backtracking. They produce similar results in practice, but are based on distinct principles that we discuss below. These

[13]The three floating point coefficients of this quadratic polynomial are stored in an array sized according to the number $\approx N^{1-\frac{1}{d}}$ of trial points. A table of unsigned integers stores, for each point $\mathbf{p} \in X \cup \partial X$, the corresponding array index if $\mathbf{p}$ is tagged Trial, or a dummy value otherwise.

methods presently lack a proper convergence analysis, in particular for non-holonomic models, hence they may be regarded as partly heuristic.

3.2.1 Modified Euler Method using Upwind Gradients

Our first approach to geodesic backtracking is to solve the ODE (5) using an explicit second order integration scheme, and an upwind approximation of the geodesic flow direction. More precisely, consider the differential equation

$$\dot\gamma(t) = V(\gamma(t)), \qquad \text{where } V(\mathbf{p}) := \mathrm{d}\mathcal{F}^*_{\mathbf{p}}(\mathrm{d}u(\mathbf{p})), \tag{39}$$

for all $\mathbf{p} \in \overline{\Omega}$. We use the integration scheme[14] $\gamma(t+\delta) \approx \gamma(t) + \delta V\left(\gamma(t) + \frac{1}{2}\delta V(\gamma(t))\right)$, often called the modified Euler method, the midpoint rule, or the Runge-Kutta-2 method. The time-step $\delta > 0$ is locally adjusted so that $|\delta V(\gamma(t))|$ equals a fraction of the gridscale, by default $h/4$. The stopping criterion involves, for robustness, a safety radius around the seed points.

The main difficulty for implementing this method lies in the (approximate) evaluation of the geodesic flow direction $V(\mathbf{p})$. For instance, naively approximating the gradient $\mathrm{d}u(\mathbf{p})$ using centered finite differences often yields unstable results. Instead, we use an upwind approximation of $V(\mathbf{p})$, derived from the numerical scheme, namely

$$V(\mathbf{p}) \approx \widetilde{V}(\mathbf{p}) := h^{-1} \sum_{1\le i\le I} \alpha_i \left(U(\mathbf{p}) - U(\mathbf{p} - h\dot{\mathbf{e}}_i)\right)_+ \dot{\mathbf{e}}_i. \tag{40}$$

Let us formally justify this expression. On the one hand one has

$$\mathrm{d}\mathcal{H}_{\mathbf{p}}(\mathrm{d}u(\mathbf{p})) = 2\mathcal{F}^*_{\mathbf{p}}(\mathrm{d}u(\mathbf{p}))\,\mathrm{d}\mathcal{F}^*_{\mathbf{p}}(\mathrm{d}u(\mathbf{p})) = 2V(\mathbf{p}),$$

where we used first the identity $\mathcal{H}_{\mathbf{p}}(\hat{\mathbf{p}}) = \mathcal{F}^*_{\mathbf{p}}(\hat{\mathbf{p}})^2$, and second the eikonal equation $\mathcal{F}^*_{\mathbf{p}}(\mathrm{d}u(\mathbf{p})) = 1$. One obtains on the other hand

$$\mathrm{d}\mathcal{H}_{\mathbf{p}}(\hat{\mathbf{p}}) = 2\sum_{1\le i\le I} \alpha_i \langle \hat{\mathbf{p}}, \dot{\mathbf{e}}_i\rangle_+ \dot{\mathbf{e}}_i, \qquad \mathrm{d}_{\mathbf{p}}\mathcal{H}(\mathrm{d}u(\mathbf{p})) \approx 2h^{-1}\sum_{1\le i\le I} \alpha_i \left(u(\mathbf{p}) - u(\mathbf{p} - h\dot{\mathbf{e}}_i)\right)_+ \dot{\mathbf{e}}_i,$$

where we used (left) the Hamiltonian expression $\mathcal{H}_{\mathbf{p}}(\hat{\mathbf{p}}) = \sum_{i=1}^I \alpha_i \langle \hat{\mathbf{p}}, \dot{\mathbf{e}}_i\rangle_+^2$, and (right) a first order Taylor expansion. Substituting the PDE solution u with its numerical approximation U, one obtains (40) as announced. It is well known that the geodesics on a manifold move at constant speed, a property which in our context reads $\mathcal{F}_{\mathbf{p}}(V(\mathbf{p})) = 1$ (under some differentiability assumptions), and can be derived from (39, right) and Legendre-Fenchel duality. The following proposition, never published elsewhere, further motivates (40) by establishing a discrete counterpart of this property. It is stated in more formal and abstract terms, and proved, in Appendix C.

Proposition 3.1. *Under the assumptions of our discretization, one has* $\mathcal{F}_{\mathbf{p}}(\widetilde{V}(\mathbf{p})) \le 1$ *for any* $\mathbf{p} \in X$. *In addition, equality holds if* U *coincides with an affine function over the stencil* $\{\mathbf{p}\} \cup \{\mathbf{p} - h\dot{\mathbf{e}}_i\}_{1\le i\le I}$.

The HFM library sets $\widetilde{V}(\mathbf{p}) = 0$ on boundary points $\mathbf{p} \in \partial X$, and extends $\widetilde{V}$ to the continuous domain $\overline{\Omega} \supset X \cup \partial X$ by bilinear interpolation. Outlier points, inside walls or attached to inconsistently large[15] values of U, are for robustness excluded from the interpolation.

[14] The ODE is in fact solved backwards in time, but we omit this detail here so as to alleviate notations.

[15] Such inconsistent values typically arise when the front arrival times u are discontinuous in the domain interior Ω, with e.g. the Dubins model.

3.2.2 Diffuse Geodesics and Reverse Algorithmic Differentiation

Our second backtracking method is based on the following principle. Consider a smooth field $\zeta : \overline{\Omega} \to \mathbb{R}$, that should be regarded as a perturbation of the front speed. Define $u_\varepsilon : \overline{\Omega} \to]-\infty, \infty]$, for any sufficiently small $\varepsilon \in \mathbb{R}$, by

$$\forall \mathbf{p} \in \Omega,\ \mathcal{F}^*_{\mathbf{p}}(\mathrm{d}u_\varepsilon(\mathbf{p})) = 1 + \varepsilon\xi(\mathbf{p}), \qquad \forall \mathbf{p} \in \partial\Omega,\ u_\varepsilon(\mathbf{p}) = \sigma(\mathbf{p}).$$

Note that $u_0 = u$ is the solution to the original unperturbed shorted path problem. Under suitable assumptions, the front arrival time $u_\varepsilon(\mathbf{p}_*)$ at a point of interest $\mathbf{p}_* \in \Omega$ can be differentiated with respect to the parameter ε, and denoting by γ_* the minimal geodesic associated with $u_0(\mathbf{p}_*)$ one has at first order

$$u_\varepsilon(\mathbf{p}_*) = u_0(\mathbf{p}_*) + \varepsilon \int_0^1 \mathcal{F}_{\gamma_*(t)}(\dot{\gamma}_*(t))\, \xi(\gamma_*(t))\, \mathrm{d}t + o(\varepsilon).$$

This derivation follows for the expression of the distance $d_{\mathcal{F}}$ as an infimum on the set of paths (1), and from the envelope theorem, see [5]. Our second geodesic backtracking method takes advantage of a discrete counterpart of this property. Consider the discrete solution $U_\varepsilon : X \sqcup \partial X \to]-\infty, \infty]$ to

$$\forall \mathbf{p} \in X,\ HU_\varepsilon(\mathbf{p}) = (1 + \varepsilon\xi(\mathbf{p}))^2, \qquad \forall \mathbf{p} \in \partial X,\ U_\varepsilon(\mathbf{p}) = \sigma(\mathbf{p}).$$

Generically, U_ε is differentiable with respect to the parameter ε, and one has at first order

$$U_\varepsilon(\mathbf{p}_*) = U(\mathbf{p}_*) + \varepsilon \sum_{\mathbf{q} \in X} \rho(\mathbf{q})\xi(\mathbf{q}) + o(\varepsilon),$$

for some coefficients $\rho : X \to \mathbb{R}$. The assumption underlying our second backtracking method is that the bulk of the coefficients ρ is supported along the minimal geodesic γ_* of the continuous problem. We use sensitivity analysis by reverse accumulation to compute ρ, see Section 3.3. This diffuse support is summarized into a unique path using an averaging procedure. However, path extraction is restarted if the support of ρ is spread over an excessively large domain. This happens if e.g. there is no uniqueness of the minimal geodesic, in which case the support of ρ is split into several components, which is detected using an ad-hoc criterion, see Figure 11 (right). The computation of the weights ρ is described in the following subsection.

3.3 Sensitivity Analysis

Sensitivity analysis is a generic name for (semi-)automatic (meta-)programming techniques aimed at computing the first order differential of a function defined algorithmically as the composition of several elementary functions. This approach combines the *formal* differentiation of the elementary functions, with the *numerical* propagation of the so-called sensitivities, using the composition rules for derivatives. Denoting by $F : \mathbb{R}^m \to \mathbb{R}^n$ and $\mathbf{p} \in \mathbb{R}^m$ the given map and point of interest, the objective is to evaluate the Jacobian matrix $\mathrm{d}F_{\mathbf{p}}$ of size $m \times n$. Interestingly, sensitivity analysis comes in two flavors, in which time and space complexities differ. They are referred to as *forward* and *reverse*, and are mostly adequate when $m \gg n$ and when $m \ll n$ respectively.

The HFM library implements sensitivity analysis "by hand", since this is particularly simple and efficient for the addressed problem, instead of relying on meta-programming automatic differentiation techniques [26]. Our approach allows enhancements in performance and convenience, but also has a few limitations: differentiation is only possible with respect to certain parameters, and some variants of the numerical scheme are unsupported (i.e. slightly inexact results will be obtained with the second order accurate method, or using time-dependent parameter fields).

To our knowledge, previous literature only considered *forward* differentiation of *isotropic* fast marching [5]. Our contribution is thus two fold: we generalize these previous works to anisotropic fast marching, and we implement *reverse* sensitivity analysis for the first time in this context. The algorithms presented in this subsection are used in [41] to solve two player zero-sum games, where the first player places a surveillance system and the second player wants to visit a target undetected.

3.3.1 Inputs and Outputs of the Differentiation Methods

The Hamiltonian appearing in the eikonal equation is represented internally in the HFM software in a sum of squares form (33), involving weights and offsets, denoted by $\alpha_i(\mathbf{p}) \geq 0$ and $\dot{\mathbf{e}}_i(\mathbf{p}) \in \mathbb{Z}^d$, where $\mathbf{p} \in X$ and $1 \leq i \leq I$. Our differentiation techniques assume that the offsets remain constant, but that the weights are proportional to (the inverse square of) one or several[16] user provided cost functions $c(\mathbf{p}, l)$, $1 \leq l \leq L$, which themselves are subject to a linear perturbation $\varepsilon\xi(\mathbf{p}, l)$. See the discussion (38) on the shared stencil data structure. We also assume a linear perturbation $\varepsilon\zeta(\mathbf{q})$ of the boundary conditions $\sigma(\mathbf{q})$, $q \in \partial X$. Summarizing, the perturbed cost functions and boundary conditions read

$$c_\varepsilon(\mathbf{p}, l) := c(\mathbf{p}, l) + \varepsilon\xi(\mathbf{p}, l), \qquad \sigma_\varepsilon(\mathbf{q}) := \sigma(\mathbf{q}) + \varepsilon\zeta(\mathbf{q}), \tag{41}$$

for all $\mathbf{p} \in X$, all $\mathbf{q} \in \partial X$, and all $1 \leq l \leq L$. Following (38), the weights of the perturbed Hamiltonian read $\alpha_i^\varepsilon(\mathbf{p}) = \alpha_i(\overline{\mathbf{p}})c_\varepsilon(\mathbf{p}, l_i(\overline{\mathbf{p}}))^{-2}$. We assume that the discretized problem solution U_ε is indeed differentiable at the parameter $\varepsilon = 0$.

Forward mode. The procedure input consists of two fields $\xi : X \times [\![1, L]\!] \to \mathbb{R}$ and $\zeta : \partial X \to \mathbb{R}$. The output is the first order derivative $\mu = U_0'$, which obeys for all $\mathbf{p} \in X \cup \partial X$

$$U_\varepsilon(\mathbf{p}) = U_0(\mathbf{p}) + \varepsilon\mu(\mathbf{p}) + o(\varepsilon).$$

Reverse mode. The procedure input is a point $\mathbf{p}_* \in X \cup \partial X$, and the output consists of two families of weights $\rho : X \times [\![1, L]\!] \to \mathbb{R}$ and $\pi : \partial X \to \mathbb{R}$ such that one has at first order, for any perturbation fields ξ and ζ

$$U_\varepsilon(\mathbf{p}_*) = U_0(\mathbf{p}_*) + \varepsilon\left(\sum_{\mathbf{p}\in X, l\in[\![1,L]\!]} \rho(\mathbf{p}, l)\xi(\mathbf{p}, l) + \sum_{\mathbf{q}\in\partial X} \pi(\mathbf{q})\zeta(\mathbf{q})\right) + o(\varepsilon).$$

One may also request the Taylor expansion of a weighted sum of front arrival times, $\sum_{\mathbf{p}\in X\cup\partial X} \phi(\mathbf{p})U_\varepsilon(\mathbf{p})$ for some given $\phi : X \cup \partial X \to \mathbb{R}$, instead of a single value $U_\varepsilon(\mathbf{p}_*)$.

3.3.2 Algorithmic Strategy

In order to describe the application of sensitivity analysis to discretized eikonal equations, we introduce slightly modified notations. The perturbed Hamiltonian is rewritten as

$$\mathcal{H}_\mathbf{p}^\varepsilon(\dot{\mathbf{p}}) \approx h^{-2} \sum_{\mathbf{q}\in\mathcal{V}(\mathbf{p})} \exp\left(-2\alpha_\varepsilon(\mathbf{p}, \mathbf{q})\right) \langle \mathbf{p} - \mathbf{q}, \dot{\mathbf{p}}\rangle_+^2, \tag{42}$$

where $\varepsilon \in \mathbb{R}$ is a small parameter. This description is made equivalent to (33, left) by introducing the discretization stencils $\mathcal{V}(\mathbf{p}) := \{\mathbf{p} - h\dot{\mathbf{e}}_i(\mathbf{p});\ 1 \leq i \leq I\}$ (see (35) for boundary conditions and

[16] $L = 1$ for isotropic or curvature penalized models, but $L = d$ for diagonal models, see Section 2.2.

obstacles), and defining $\alpha_\varepsilon(\mathbf{p},\mathbf{q}) := -\frac{1}{2}\ln \alpha_i^\varepsilon(\mathbf{p})$ when $\mathbf{q} = \mathbf{p} - h\dot{\mathbf{e}}_i(\mathbf{p})$. Denote by $U_\varepsilon : X \cup \partial X \to \mathbb{R}$ the unique solution to the perturbed and discretized eikonal PDE

$$\forall \mathbf{p} \in X,\ h^{-2} \sum_{\mathbf{q}\in\mathcal{V}(\mathbf{p})} \exp(-2\alpha_\varepsilon(\mathbf{p},\mathbf{q}))(U_\varepsilon(\mathbf{p}) - U_\varepsilon(\mathbf{q}))_+^2 = 1, \qquad \forall \mathbf{p} \in \partial X,\ U_\varepsilon(\mathbf{p}) = \sigma_\varepsilon(\mathbf{p}). \tag{43}$$

Our objective is to relate the first order Taylor expansions of the boundary conditions σ_ε, weights α_ε, and solution U_ε. Before we begin, let us mention that the simplified numerical scheme (33) considered in this section makes U_ε differentiable, but that it is only almost everywhere differentiable in the general case (18). That is because $a \in \mathbb{R} \mapsto \max\{0,a\}^2$ is everywhere differentiable, but not $(a,b) \in \mathbb{R}^2 \to \max\{a,b\}^2$.

For all $\mathbf{p} \in \partial X$, the boundary condition (43, right) yields the simple relation

$$U'_\varepsilon(\mathbf{p}) = \sigma'_\varepsilon(\mathbf{p}),$$

where the prime denotes differentiation with respect to the parameter ε. For interior points $\mathbf{p} \in X$, we obtain differentiating (43, left) that

$$\sum_{\mathbf{q}\in\mathcal{V}(\mathbf{p})} \omega_\varepsilon(\mathbf{p},\mathbf{q})\Big(U'_\varepsilon(\mathbf{p}) - U'_\varepsilon(\mathbf{q}) - (U_\varepsilon(\mathbf{p}) - U_\varepsilon(\mathbf{q}))\,\alpha'_\varepsilon(\mathbf{p},\mathbf{q})\Big) = 0,$$

where $\omega_\varepsilon(\mathbf{p},\mathbf{q}) := \exp(-2\alpha_\varepsilon(\mathbf{p},\mathbf{q}))(U_\varepsilon(\mathbf{p}) - U_\varepsilon(\mathbf{q}))_+$. Note, crucially, that $\omega_\varepsilon(\mathbf{p},\mathbf{q}) = 0$ whenever $U_\varepsilon(\mathbf{p}) \leq U_\varepsilon(\mathbf{q})$. This property directly comes from the causality of the PDE discretization (43, left), i.e. the fact that it only involves positive parts of finite differences. Thanks to this, $U'_\varepsilon(\mathbf{p})$ can be expressed in terms of those $U'_\varepsilon(\mathbf{q})$ associated to points $\mathbf{q} \in \mathcal{V}(\mathbf{p})$ reached by the front propagation strictly *earlier* than $\mathbf{p}$, in other words such that $U_\varepsilon(\mathbf{p}) > U_\varepsilon(\mathbf{q})$. More precisely one has

$$U'_\varepsilon(\mathbf{p}) = \sum_{\mathbf{q}\in\mathcal{V}(\mathbf{p})} \overline{\omega}_\varepsilon(\mathbf{p},\mathbf{q})\,(U'_\varepsilon(\mathbf{q}) + (U_\varepsilon(\mathbf{p}) - U_\varepsilon(\mathbf{q}))\alpha'_\varepsilon(\mathbf{p},\mathbf{q}))\,, \tag{44}$$

where $\overline{\omega}_\varepsilon(\mathbf{p},\mathbf{q}) := \omega_\varepsilon(\mathbf{p},\mathbf{q}) / \sum_{\mathbf{q}\in\mathcal{V}(\mathbf{p})} \omega_\varepsilon(\mathbf{p},\mathbf{q})$, for any $\mathbf{p} \in X$ and any $\mathbf{q} \in \mathcal{V}(\mathbf{p})$.

Let us reformulate this procedure in linear algebra terms. Let $\mu := (U'_0(\mathbf{p}_n))_{n=1}^N$ collect the derivatives of the solution, where the points of $X \cup \partial X$ are sorted by increasing values $U_0(\mathbf{p}_1) \leq \cdots \leq U_0(\mathbf{p}_N)$. We regard the perturbations ξ and ζ, of the cost functions and the boundary conditions, as column vectors. Thus ζ collects the values $(\sigma'_0(\mathbf{p}))_{\mathbf{p}\in\partial X}$, and ξ is linearly related to $(\alpha'_0(\mathbf{p},\mathbf{q}))_{\mathbf{q}\in\mathcal{V}(\mathbf{p})}^{\mathbf{p}\in X}$. Therefore (44) at the parameter $\varepsilon = 0$ can be rewritten in the form

$$\mu = L\mu + A\xi + B\zeta,$$

where L is a strictly upper triangular matrix (in view of (44) and recalling that $\overline{\omega}_\varepsilon(\mathbf{p},\mathbf{q}) > 0$ requires $U_\varepsilon(\mathbf{p}) > U_\varepsilon(\mathbf{q})$), and A and B are known matrices. Forward automatic differentiation computes $\mu := (\mathrm{Id} - L)^{-1}(A\xi + B\zeta)$, given ξ and ζ. Reverse automatic differentiation is the adjoint procedure, computing $\rho := \phi(\mathrm{Id} - L)^{-1}A$ and $\pi := \phi(\mathrm{Id} - L)^{-1}B$, given $\phi : X \cup \partial X \to \mathbb{R}$. The triangular structure of $\mathrm{Id} - L$ is of course leveraged for fast inversion, by recursive substitution, and actually the matrices A, B and L are never explicitly assembled by the HFM software.

4 Numerical Experiments

This section is devoted to numerical experiments, which are reproducible using the series of notebooks that come alongside this publication. Part of these are Jupyter notebooks written in the Python®

language[17], while others are designed for the Mathematica® software[18]. The original source code[19] also features additional examples.

We illustrate the different metric models introduced in Section 2, such as isotropic, Riemannian, or non-holonomic metrics, as well as the algorithmic techniques presented in Section 3, such as forward and reverse automatic differentiation.

The numerical examples discussed in this section were designed with the intent to illustrate the functionalities of the HFM library. In contrast, Section 5 shifts the focus from algorithmics to modeling, and involves examples and test cases coming from applications.

4.1 Base Functionalities, with an Isotropic Metric

This paragraph serves as an introduction to using the HFM library, on a problem involving a basic isotropic metric, and that could thus be addressed using a variety of other software packages. Features more specific to our software are considered in the next subsections. We address here a two-dimensional shortest path problem involving an isotropic metric on a domain $\Omega \subset \mathbb{R}^2$, with the following value function

$$u(\mathbf{p}) := \min_{\substack{\gamma(1)=\mathbf{p} \\ \gamma(0)\in\partial\Omega}} \sigma(\gamma(0)) + \int_0^1 c(\gamma(t))\|\gamma'(t)\| \, \mathrm{d}t. \tag{45}$$

Such optimization problems are specified to the HFM library using a dictionary, which entries are in this paper denoted by **key**:value. The exact syntax for constructing such an object depends in practice on the language used for interfacing, e.g. Python, Matlab® or Mathematica®. This dictionary is then fed to the adequate executable, for instance MatlabHFM_Isotropic2, where the prefix denotes the external interface ("Matlab" could be replaced with "Python", "Mathematica", or "File" for the the stand-alone file based executable), and the suffix denotes the metric model, here an isotropic metric on a two-dimensional domain, which corresponds to (45).

The next step is to define a Cartesian grid, in our example of size $2n \times n$ where $n = 100$, and a box domain $\Omega_0 = [-1, 1] \times [0, 1]$, enclosing the PDE domain Ω, and containing the seed points where finite boundary conditions are imposed, and possibly some obstacles. For that purpose we provide the keys **dims**:$(2n, n)$, **origin**:$(-1, 0)$ (the bottom left corner), and **gridscale**:h, where $h := 1/n$. A few adjustments to this construction might unfortunately be necessary, depending on the visualization software, see the technical discussion in Appendix D.

We next introduce some starting points for the front propagation, by e.g. the key pair **seeds**:$[(-0.5, 0.3), (0.5, 0.8)]$, and some boundary conditions at these points, e.g. **seedValues**:[0,0.5]. We also need to define the cost function, e.g. the constant **cost**:1, and opt in or out of the second order enhancement to the fast marching method, e.g. **sndOrder**:1. At this point, we have gathered enough data to run the fast marching algorithm, but the HFM library must also be instructed what to export in return. We here choose to request minimal geodesics towards a family of points, e.g. **tips**:[(0,0.6),(-0.9,0.5),(0.8,0.8)], as well as the numerical solution U to the eikonal equation and the upwind geodesic flow vector field (40), as specified by the key pairs **exportValues**:1 and **exportGeodesicFlow**:1. The outputs of this program are illustrated in Figure 7.

In order to enrich the problem, we may introduce some obstacles in the domain and a position dependent cost function. They are specified using the key pairs **walls**:arrBool and **cost**:arrFloat (replacing the former **cost**:1), where arrBool and arrFloat are arrays of boolean and of positive floating point values, sized according to the discretization grid. See Figure 8.

[17] github.com/Mirebeau/HFM_Python_Notebooks
[18] github.com/Mirebeau/HFM_Mathematica_Notebooks
[19] github.com/Mirebeau/HamiltonFastMarching

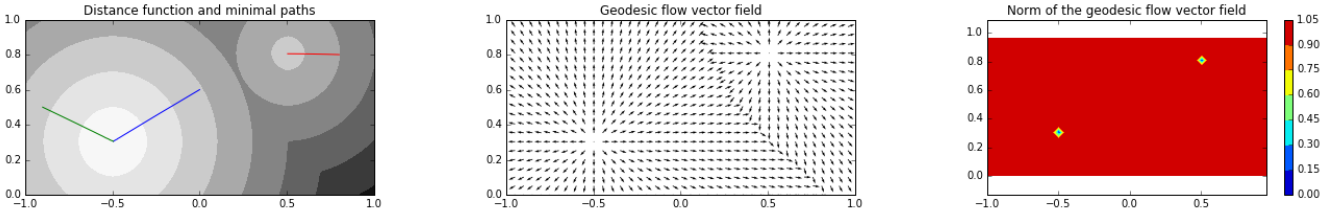

Figure 7: Left: distance map $u(\mathbf{p}) := \min\{\|\mathbf{p}-\mathbf{p}_1\|, \sigma_2+\|\mathbf{p}-\mathbf{p}_2\|\}$, where $\mathbf{p}_1 = (-0.5, 0.3)$, $\mathbf{p}_2 = (0.5, 0.8)$ and $\sigma_2 = 0.5$. The function u was numerically computed using isotropic fast marching with second order enhancement. Center: geodesic flow $V(x) := \nabla u(x)$ obtained numerically using (40). Right: verification that $|\nabla V(x)| = 1$, at all points of differentiability.

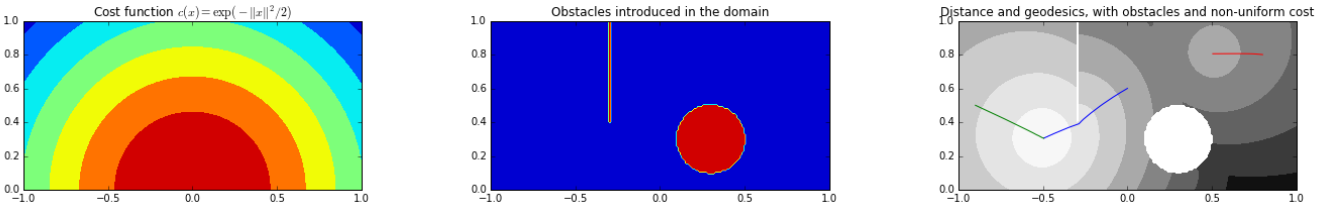

Figure 8: Left: level sets of a non-constant cost function. Center: obstacles introduced in the domain. Right: level sets of the distance map from two seed points, obtained using isotropic fast marching.

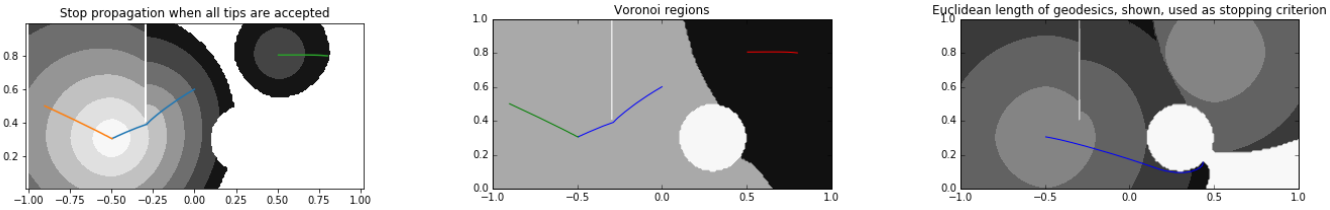

Figure 9: Examples of stopping criteria. Left: front propagation is stopped when all geodesics tips are accepted, which is enough for backtracking. Center: Voronoi regions associated with two seeds. Right: the Euclidean length $l(\mathbf{p})$, of the minimal path from each point $\mathbf{p} \in \Omega$, is here computed simultaneously with the distance map $u(\mathbf{p})$ and used as a stopping criterion.

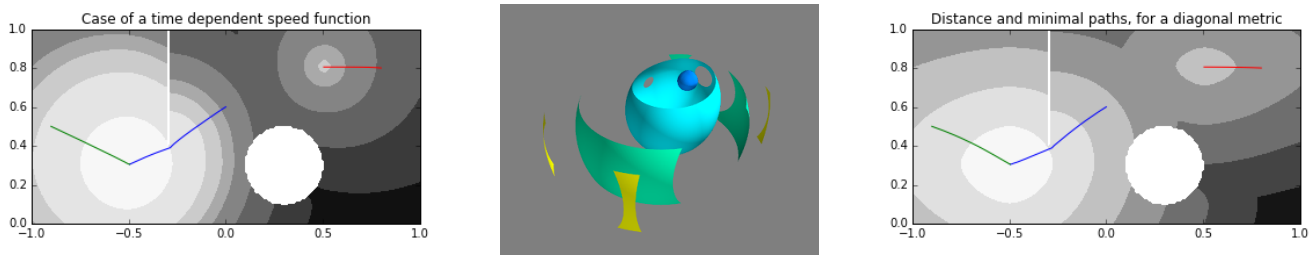

Figure 10: Distance map associated with, respectively: a time dependent cost function (left), a three dimensional isotropic metric (center), or a diagonal metric (right).

The front propagation can be stopped early in many applications, for instance as soon as the geodesic(s) of interest can be backtracked, saving substantial computation time: often up to 80% in applications to tubular structure segmentation, see Section 4.5 . Fortunately, the single pass nature of the fast marching algorithm makes it particularly simple to define and implement termination criteria. Recall that the points of the discretization grid are successively "accepted", and the corresponding values frozen, in the order $U(\mathbf{p}_{\sigma(1)}) \leq \cdots \leq U(\mathbf{p}_{\sigma(n)})$ of increasing values of the numerical solution U. We may instruct the HFM library to stop the front propagation when all or any of a set of points are accepted, which is enough for geodesic backtracking, using the keys **stopWhenAllAccepted** or

stopWhenAnyAccepted respectively. See Figure 9 (left). One may also put a threshold on the number of accepted points, or on the values of the solution U.

Several side products may be computed simultaneously while the fast marching algorithm is running, for use within additional stopping criteria and/or for being returned to the user. For instance, the computation of the Voronoi diagram associated with the different seeds is triggered by assigning labels of them, with e.g. the pair **seedFlags**:[0,1]. See Figure 9. Identical labels may be assigned to several seeds, in which case the regions are merged. Optionally, front propagation can be stopped when the Voronoi regions meet, using **VoronoiStoppingCriterion**:'RegionsMeeting'. In that case the minimal geodesic from the seeds of the meeting regions is returned, in the form of two halves joining at the regions meeting point. Another side product of interest is the Euclidean length of the minimal geodesics, see Figure 9 (right).

The use of a time dependent cost function $c = c(\mathbf{p}, t)$, giving rise to the eikonal equation $c(\mathbf{p}, u(\mathbf{p}))\|\nabla u(\mathbf{p})\| = 1$, is illustrated in Figure 10. We use an explicit scheme, which is simpler than [63] but introduces an $\mathcal{O}(h^2)$ overall error in the solution (expected to be dominated by the discretization error, since the numerical scheme used is at best second order). Other classical models implemented in the HFM library are also illustrated in Figure 10, namely two and three dimensional, isotropic and diagonal metrics, corresponding to the executables ending with 'Isotropic3', 'Diagonal2', or 'Diagonal3'. Finally, we recall that the features presented in this subsection are transversal and applicable to any minimal path model implemented in the HFM library.

4.2 Automatic Differentiation

Automatic, or algorithmic, differentiation, is a meta-programming technique for evaluating the Jacobian of a numerical function. We refer to Section 3.3 for details on these methods, and only recall here their basic intent. Denote by $u = F(c, \sigma)$ the mapping which associates the numerical PDE solution u to the numerical cost function c and boundary conditions σ. Denote by J the Jacobian matrix of F at some fixed input (c, σ), assuming it is indeed differentiable there. The matrix J is usually non-sparse and excessively large. Fortunately, the technique of forward (resp. backward) differentiation lets us evaluate at a very reasonable cost any matrix-vector product $\mu = J \cdot (\xi, \zeta)$ with the Jacobian (resp. $(\rho, \pi) = J^T \cdot \phi$ with its adjoint), where ξ, ζ and ϕ are arbitrary.

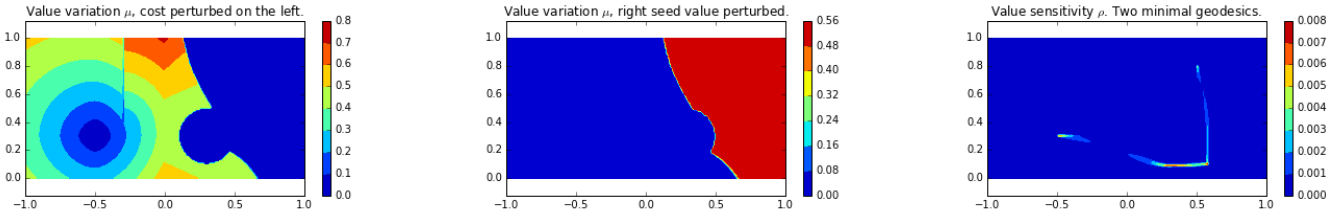

Figure 11: Illustration of forward and backward automatic differentiation. Same setting as Figure 8. Left: first order term μ in the Taylor expansion $u_\varepsilon = u_0 + \varepsilon\mu + o(\varepsilon)$ of the distance map associated with the perturbed cost $c_\varepsilon = c_0 + \varepsilon\chi_{x<0}$. Center: likewise, for a perturbation of the boundary condition $\sigma_{2,\varepsilon} = \sigma_2 + \varepsilon$ at the seed $\mathbf{p}_2 = (0.5, 0.8)$. Right: sensitivity ϕ, with respect to the cost function, of the distance $u(\mathbf{p}_*)$ from $\mathbf{p}_* = (0.6, 0.1)$ to the seed points. The support of ϕ approximates the minimal geodesic(s) from $\mathbf{p}_*$ to the closest seed point(s). Two geodesics are visible here since $\mathbf{p}_*$ was chosen in the cut locus.

In the context of the fast marching algorithm, the first reported uses of forward and backward differentiation respectively appear in [5] and [41]. For simplicity, these functionalities are illustrated here in the context of isotropic metrics, but they are applicable to all the minimal path models implemented in the HFM library, including those featuring dimension lifting and curvature penalization techniques [41].

Forward differentiation is used in Figure 11 to compute the first order perturbation in the distance map u, constructed in Section 4.1, subject to a given perturbation of the input cost function (left), or of the boundary conditions (center). Conversely, backward differentiation yields the sensitivity of a specific pointwise value $u(\mathbf{p}_*)$ to arbitrary modifications in the cost and boundary conditions, which reveals the support of the minimal geodesic(s), as discussed in Section 3.2.2 and illustrated in Figure 11 (right).

We next present an optimization problem in which one aims to maximize the distance between two points $\mathbf{p}_0$ and $\mathbf{p}_1$, by adjusting the cost function $c : \Omega \to]0, \infty[$ of an isotropic metric. See [6, 41] for problems of similar nature, some of them more complex. The cost function is constrained by upper and lower bounds, and subject to an integral penalty term, all determined by positive constants α, β, γ. The problem reads as follows

$$\max_{\alpha \leq c \leq \beta} \min_{\substack{\gamma(0)=\mathbf{p}_0 \\ \gamma(1)=\mathbf{p}_1}} \int_0^1 c(\gamma(t))\|\gamma'(t)\|\mathrm{d}t - \gamma \int_\Omega c(x)\mathrm{d}x. \tag{46}$$

The minimum over all paths γ is, obviously, numerically computed using the fast marching algorithm. We use the same domain and obstacles as in the previous section, on a 100×200 Cartesian grid, but switch to the differentiable discretization scheme (21) for isotropic eikonal PDEs. This defines a concave function of the cost, which is then maximized, using the LBFGS algorithm[20] numerically which takes about 20 seconds. For comparison, solving similar instances reportedly took hours in [5]. The lower complexity of our implementation mainly comes from the use of backward differentiation, instead of forward differentiation in [5]. The optimal cost function c for (46) has several interesting features. For instance it creates "barriers" in the space between close obstacles. Another peculiarity is the fact that there are uncountably many distinct paths of minimal cost, w.r.t. c, joining the endpoints $\mathbf{p}_0$ and $\mathbf{p}_1$. See Figure 12.

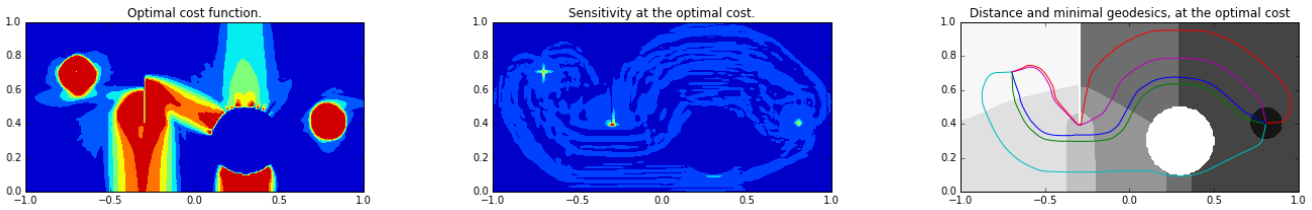

Figure 12: Left: cost function maximizing the distance between the seed $(-0.7, 0.7)$ and the tip $\mathbf{p}_* = (0.8, 0.4)$, subject to bound constraints and an integral penalty, see (46). Same obstacles as in Figure 8. Center: Sensitivity of $u(\mathbf{p}_*)$ with respect to the cost function, at the optimum. Right: distance map u from the seed, backtracked geodesics from the tip $\mathbf{p}_*$ and nearby pixels, at the optimum.

4.3 Riemannian Metrics

Riemannian metrics are the most widely occurring class of non-isotropic metrics in applications. Extending the fast marching algorithm to this context is a non-trivial task, which has been the subject of a continued line of research since the 2000's. Many of the proposed approaches have significant drawbacks, such as the use of excessively large stencils [29, 58, 1], the pre-condition that the condition number of the metric tensors be mild [27], or the fact that they give up on the causality property [8].

[20] It turns out empirically that the LBFGS algorithm still works with the classical non-differentiable scheme (20), although it requires twice as many iterations to reach the same accuracy.

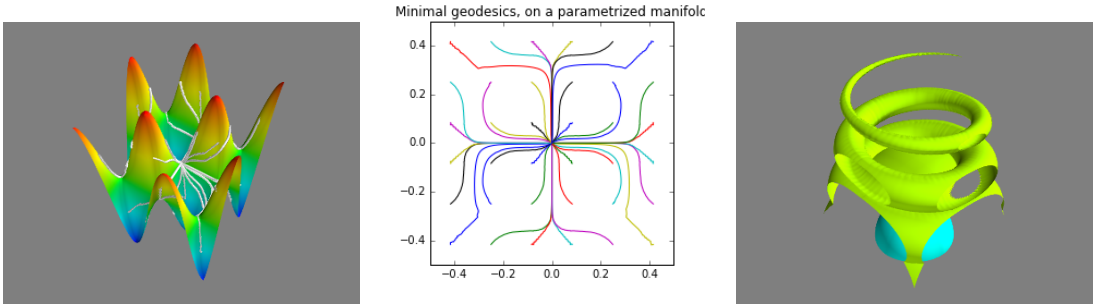

Figure 13: Left: minimal geodesics on a parametrized surface in $\mathbb{R}^3$. Center: top view of the same geodesics. Right: level sets of the distance map, for a strongly anisotropic three dimensional test case inspired by tubular structure segmentation.

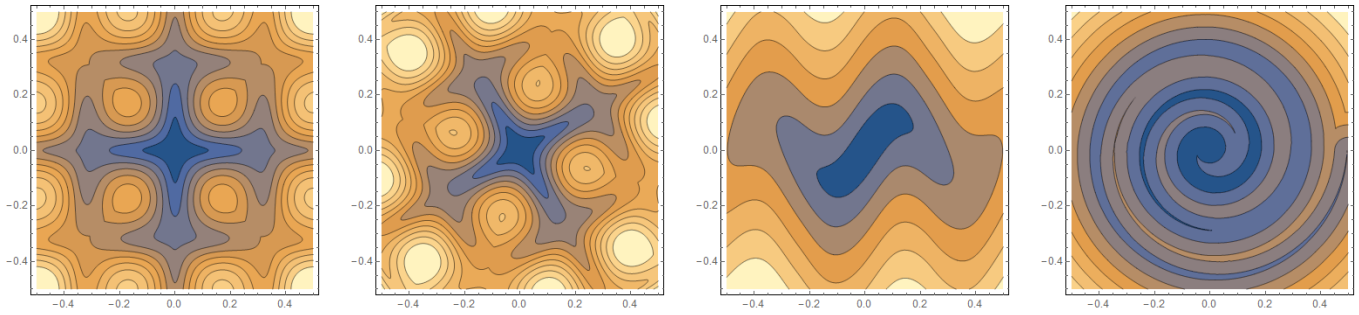

Figure 14: Level sets of the four anisotropic Riemannian two dimensional examples described in Section 4.3.

The author has introduced efficient and specific techniques for discretizing *Riemannian* eikonal PDEs on *Cartesian grids*. Two flavors were developed, in the semi-Lagrangian and Eulerian framework respectively, referred to as[21] the FM-LBR and FM-VR1. They are introduced in [36] and [40] respectively, with open source code distribution and reproducible experiments in [37] and the present manuscript respectively. For those readers who may hesitate between the two approaches, let us say that the FM-VR1 is faster, especially in dimension 3, and more accurate if the second order enhancement of the numerical scheme is activated, but slightly less accurate otherwise, see [40] for a more in-depth comparison. From a practical point of view, the FM-VR1 code is more feature packed, and has interfaces to more scripting languages, while the FM-LBR is tightly integrated within the Insight ToolKit (ITK®) library.

We repeat in this subsection the numerical experiments presented in [36, 40], which can now be easily reproduced using the provided Python and Mathematica® notebooks. The test cases are all synthetic, but are inspired from problems arising in geometry processing, seismic imaging and tubular segmentation. In practice, instructing the HFM software to numerically solve a Riemmanian eikonal equation is done by selecting the appropriate executable, ending with 'Riemann2' or 'Riemann3', depending on the dimension[22], and providing the key-value pair **metric**:arrTensors, where "arrTensors" is an array containing the Riemannian metric tensors. (Alternatively, the inverse tensors may be supplied as **dualMetric**:arrTensors.)

A series of two dimensional test cases, considered in [58, 36, 40], are described below. Level sets of each example are displayed in Figure 14. We recall that the condition number of a symmetric tensor is defined by $\mathrm{Cond}(M) := \sqrt{\|M\|\|M^{-1}\|}$. The numerical techniques implemented in the HFM library are primarily intended for Riemannian metrics which tensors have condition number $\lesssim 10$, although more degenerate instances are often handled successfully.

[21] Fast-Marching using Lattice Basis Reduction, and Fast-Marching using Voronoi's First Reduction
[22] The current numerical implementation is indeed limited to dimensions 2 and 3.

1. (Geometry processing inspired) We compute the distance from the origin on a two-dimensional manifold, embedded in $\mathbb{R}^3$ and defined by the following height map, where $x, y \in [-1, 1]^2$

$$z(x, y) := (3/4) \sin(3\pi x) \sin(3\pi y).$$

   The Riemannian metric is given by $\mathcal{M}(x, y) = I + \nabla z(x, y) \nabla z(x, y)^T$, and has condition number $\lesssim 6$. See Figures 13 and 14. A particularity of this test case is that the eigenvectors of the metric are quite often almost aligned with the coordinate axes. We thus rotate them by $\frac{\pi}{6}$, in a second test case, to show that our numerical method is not sensitive to this bias, contrary to several others, see the discussion in [36].

2. (Seismic imaging inspired) We consider the distance-to-origin problem for the metric tensor $\mathcal{M}(x, y)$ with eigenvalue 0.8^{-2} associated with the eigenvector $(1, \frac{\pi}{2} \cos(4\pi x))$, and eigenvalue 0.2^{-2} associated with a perpendicular eigenvector, on the domain $[-1, 1]^2$. The condition number of the Riemannian metric is thus bounded by 4.

3. (Tubular segmentation inspired) We illustrate the robustness of our numerical scheme by choosing a Riemannian metric which is both discontinuous and extremely anisotropic. More precisely, the metric is isotropic Euclidean sufficiently far from a predefined curve, in this case a spiral, and close to the curve it is highly anisotropic with eigenvalues $(1, 100^{-2})$. The tangent to the curve is the eigenvector corresponding to the small eigenvalue.

Three dimensional counterparts of the "seismic imaging" and "tubular segmentation" inspired test cases are also presented in Figure 13 (right), and in the Python notebooks.

4.4 Planar Curvature Penalization

An important innovation of the HFM library is the ability to find planar curves globally minimizing an energy involving their *curvature*. For that purpose, the problem is lifted in the configuration space $\mathbb{R}^2 \times \mathbb{S}^1$ of positions and orientations, following an idea introduced in [46]. A suitable non-holonomic metric is then introduced on this set, as proposed in [4, 14] for the Reeds-Shepp and Euler-Mumford models. Lastly, a specific generalization of the fast marching method is required for efficiently solving these problems numerically, see Section 1.2, Section 2.4 and [40, 39] which are at the foundation of our software, or related but distinct methods [52, 22].

The HFM library implements four curvature penalized path models, whose distinctive features are listed below. See Section 1.2 and Section 2.4 for a formal definition.

- The classical *Reeds-Shepp* sub-Riemannian model, encodes the motion constraints of a wheelchair, which is able to move both forward and backward. Distinctive feature: the trajectories feature *cusps*, at places where the vehicle changes from forward to reverse gear.

- The *Reeds-Shepp forward* variant is mathematically similar, except that the vehicle cannot go backwards. Distinctive feature: the trajectories feature *in-place rotations*, in particular at their endpoints and around obstacles.

- The *Euler-Mumford* elastica model describes the rest position of an elastic bar[23]. Distinctive feature: the trajectories are smooth, and "feel natural".

- The *Dubins* car model has a bounded turning radius. Distinctive feature: the optimal trajectories are concatenations of straight lines and of arcs of circle.

[23] Assuming that the cost function c is constant.

These models are illustrated in Figure 15 in the special case of a constant cost function $c \equiv 1$. In Figure 16 we consider both a position dependent cost function $c = c(\mathbf{x})$, which is a common case in applications, and an orientation dependent cost function $c = c(\theta)$, which in particular is suitable for modeling a sailboat. The executable ending with 'ReedsShepp2', 'ReedsSheppForward2', 'Elastica2' or 'Dubins2' must be selected to solve these instances. The parameter ξ, homogeneous to a radius and modulating the intensity of curvature penalization see (10), must be assigned a value, such as **xi**:1. The cost function is set with the pair **cost**:arr where 'arr' is either a constant, a one-dimensional array if $c = c(\theta)$, a two-dimensional array if $c = c(\mathbf{x})$, or a three dimensional array in the general case where $c = c(\mathbf{x}, \theta)$, of suitable dimensions.

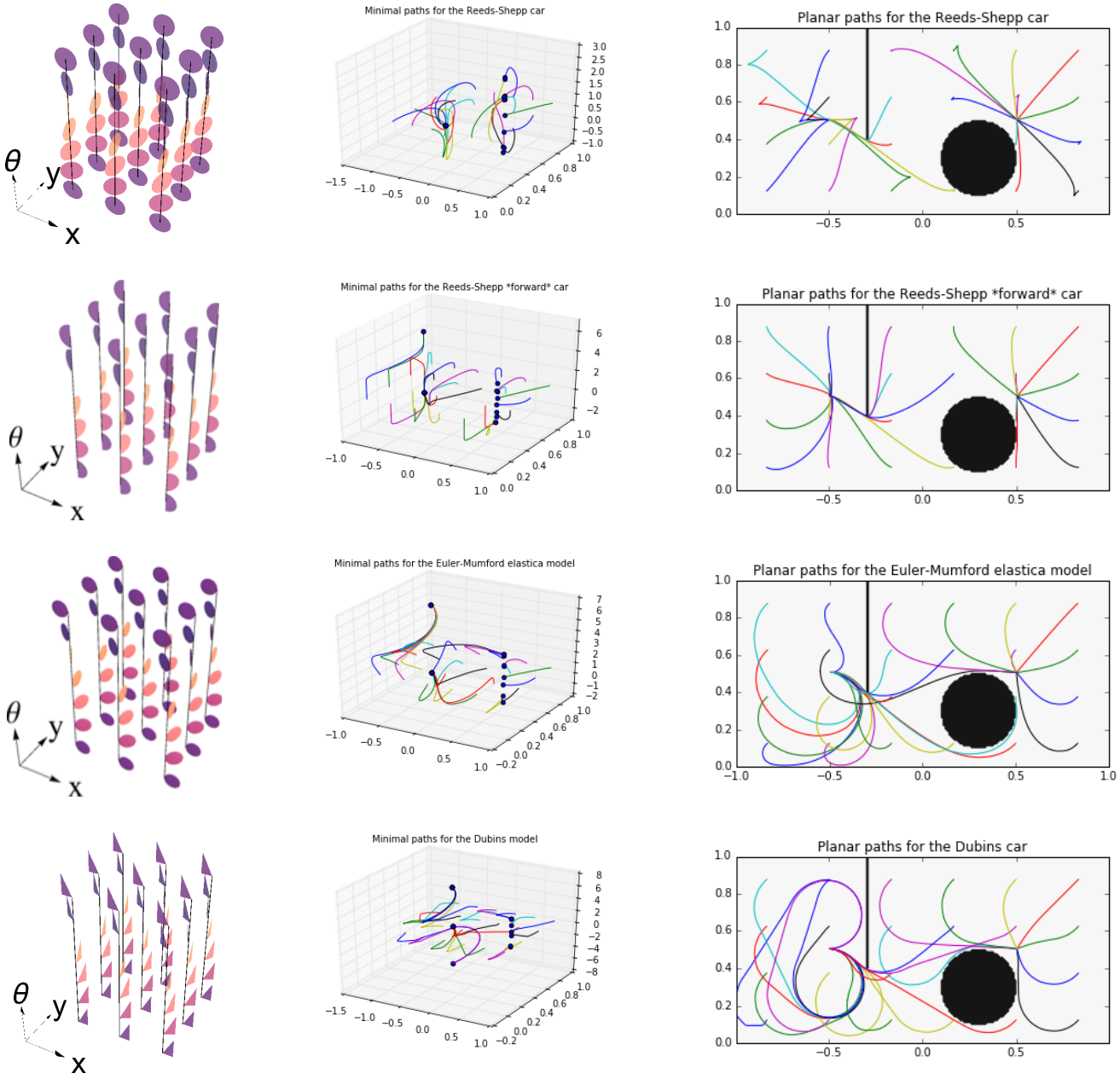

Figure 15: From top to bottom: Reeds-Shepp, Reeds-Shepp forward, Euler-Mumford elastica, and Dubins car models. Left: control sets, i.e. unit balls of the local metric. Center: minimal paths in $\mathbb{R}^2 \times \mathbb{S}^1$. Right: planar projection of the minimal paths.

The HFM software also implements generalizations of the above four models, in which the

curvature penalization parameter may vary over the domain $\xi = \xi(\mathbf{x}, \theta)$, and an additional term $\kappa = \kappa(\mathbf{x}, \theta)$, homogeneous to a curvature, introduces some asymmetry in the treatment of right and left turns. See (11) and Figure 17.

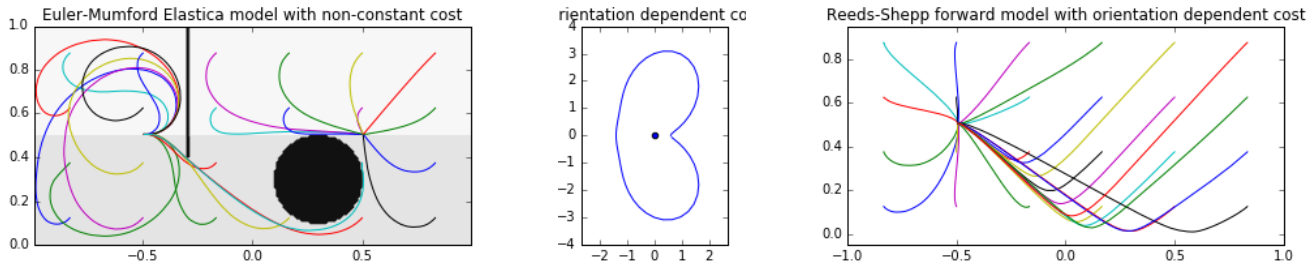

Figure 16: Left: Planar projection of Euler-Mumford elastica paths, for a cost function $c = c(\mathbf{x})$ depending on the physical position $\mathbf{x} \in \mathbb{R}^2$ (smaller cost in the upper part of the domain). Center: polar plot of a cost function $c = c(\theta)$, related with sailboat navigation, depending only on the orientation $\theta \in \mathbb{S}^1$. Right: minimal paths for the Reeds-Shepp forward model with the later cost function. The paths *tack*, as an actual sailboat would do, instead of going straight against the wind.

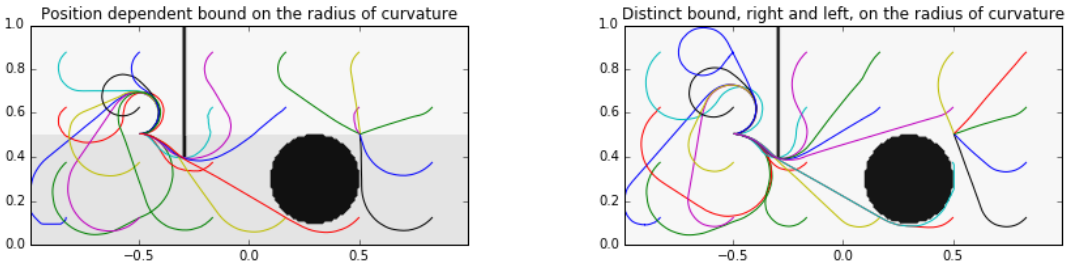

Figure 17: Minimal paths for variants of the Dubins car. Left: the car has a position dependent turning radius $\xi = \xi(x) = 1 + \chi_{y<1/2}$, smaller in the upper part of the domain. Right: the car has a distinct turning radius on the right and left.

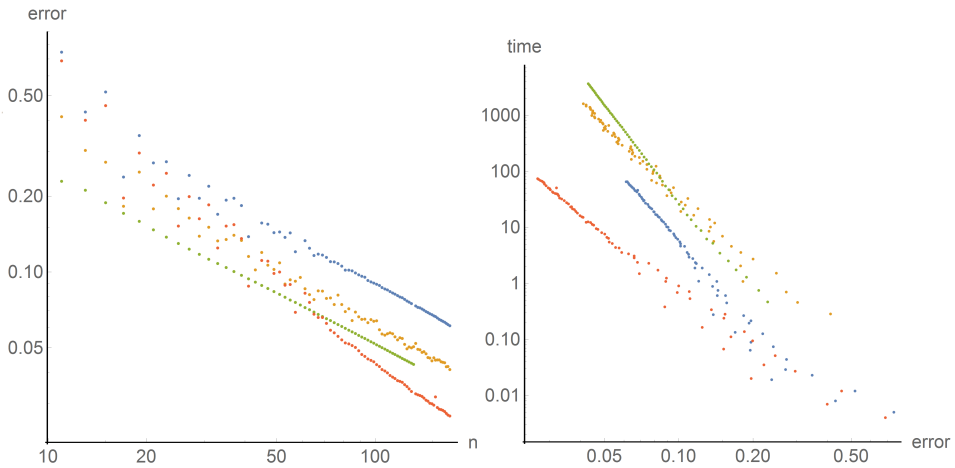

Figure 18: Comparison of the resolution/accuracy (left) and accuracy/computation time (right) compromise of several numerical methods for the eikonal equation of the Reeds-Shepp model. Log-log scale. In both figures we compare an iterative approach (green), the FM-LBR (yellow), the FM-VR1 implemented in the HFM-library (blue), and the same with second order enhancement (red). The latter turns out to be the most efficient, see Section 4.4 for a discussion.

Computational efficiency is an important aspect of numerical techniques for minimal path extraction, which severely constrains their range of applications (response times of different orders of

magnitude are required for e.g. real time control of a dynamical system, convenient interaction with a human within an image processing software, or offline data processing). In Figure 18, we compare the accuracy and execution time of four numerical methods for the classical Reeds-Shepp model, extending the work [52]. The graphics display the results of the following methods: the HFM library with (red) or without (blue) second order enhancement, the semi-Lagrangian FM-LBR implementation (yellow) [52], and a non-causal scheme solved using an iterative numerical method (green)[4]. A Cartesian grid of size $(4n, 4n, 2n)$ is used to discretize the domain $[-2\pi, 2\pi]^2 \times \mathbb{S}^1$, and the cost function is set constant $c \equiv 1$. The exact solution to the eikonal equation, computed using a geodesic shooting method, is used for reference.

The first three compared numerical methods involve a relaxation parameter, here set to the default value $\varepsilon = 0.1$. Therefore one could expect the numerical error to plateau at sufficiently high grid scales n. However this phenomenon is not observed in our experiment, which only shows straight lines in log-log scale. This suggests that the discretization error introduced by the relaxation is not a limiting factor for our numerical scheme, at reasonable grid sizes.

At a given image resolution n, the most accurate numerical method is the FM-VR1 scheme with second order enhancement, which is implemented in the HFM library. The (slow) iterative method comes next, and is best among the first order schemes, followed by the FM-LBR scheme, and finally (a bit disappointingly) the first order instantiation of the FM-VR1. Given a computation time budget, the FM-VR1 with second order enhancement offers the best accuracy. The first order FM-VR1 comes second, compensating its lower accuracy resolution-wise with respect to the alternative methods by its much better complexity. The FM-LBR comes third, and the non-causal iterative method comes last due to its much longer computation times, which are superlinear with respect to the number of grid points.

The HFM library can extract three dimensional curves globally minimizing a curvature dependent energy. For that purpose, the curves are lifted in the five dimensional configuration space $\mathbb{R}^3 \times \mathbb{S}^2$, and the second order path energy functional is rewritten in terms of a suitable non-holonomic metric, similarly to the planar case albeit in higher dimension. Numerically, an adequate eikonal PDE is solved, after what the optimal curves are backtracked. Incidentally, the workflow that we just described involves numerically solving a non-linear PDE with degenerate anisotropy on a five dimensional manifold with non-trivial topology, which may look like a fairly challenging problem. Convincing results are nevertheless obtained in our examples, which run within $5s$ to $40s$ on a standard laptop.

Three minimal path models are considered in this section, which are higher dimensional generalizations and variants of the Reeds-Shepp model. The original sub-Riemannian model distinguishes itself by the presence of *cusps*, where the path orientation is reversed. The forward-only variant lacks cusps, but often features *in place rotations*, in particular at the minimal path endpoints, see [22] for a discussion. Finally, a dual variant favors paths which may be non-smooth, but are embedded in smooth manifolds. It can be related with torsion penalization, rather than curvature penalization, see [40]. See also [59] for a distinct attempt to implement genuine torsion penalization. Note that the three dimensional analogues of the Euler-Mumford and Dubins models, considered in the planar case Section 4.4, are not implemented in the HFM library at the time of writing.

The non-holonomic metrics considered involve a data-driven cost function $c : \mathbb{R}^3 \times \mathbb{S}^2 \to]0, \infty[$, a parameter $\xi > 0$ homogeneous to a radius of curvature, and in our case a relaxation parameter[24] $\varepsilon > 0$. For any point $(\mathbf{x}, \mathbf{n}) \in \mathbb{R}^3 \times \mathbb{S}^2$, and any tangent vector $(\dot{\mathbf{x}}, \dot{\mathbf{n}})$, the metric associated with the

[24]The formula numerically implemented in the HFM library is intermediate between the original singular model ($\varepsilon = 0$) and the relaxed model (with the given $\varepsilon > 0$), in the spirit of Proposition 2.3.

Reeds-Shepp model, the forward variant, and the dual variant, respectively read

$$\begin{aligned}
\mathcal{F}^{\varepsilon}_{(\mathbf{x},\mathbf{n})}(\dot{\mathbf{x}},\dot{\mathbf{n}})^2 &= c(\mathbf{x},\mathbf{n})^2 \left(\langle \mathbf{n}, \dot{\mathbf{x}} \rangle^2 + \varepsilon^{-2} \|\mathbf{n} \times \dot{\mathbf{x}}\|^2 + \xi^2 \|\dot{\mathbf{n}}\|^2 \right), \\
\mathcal{F}^{\varepsilon,+}_{(\mathbf{x},\mathbf{n})}(\dot{\mathbf{x}},\dot{\mathbf{n}})^2 &= c(\mathbf{x},\mathbf{n})^2 \left(\langle \mathbf{n}, \dot{\mathbf{x}} \rangle_+^2 + \varepsilon^{-2} \langle \mathbf{n}, \dot{\mathbf{x}} \rangle_-^2 + \varepsilon^{-2} \|\mathbf{n} \times \dot{\mathbf{x}}\|^2 + \xi^2 \|\dot{\mathbf{n}}\|^2 \right), \\
\mathcal{F}^{\varepsilon,\times}_{(\mathbf{x},\mathbf{n})}(\dot{\mathbf{x}},\dot{\mathbf{n}})^2 &= c(\mathbf{x},\mathbf{n})^2 \left(\varepsilon^{-2} \langle \mathbf{n}, \dot{\mathbf{x}} \rangle^2 + \|\mathbf{n} \times \dot{\mathbf{x}}\|^2 + \xi^2 \|\dot{\mathbf{n}}\|^2 \right).
\end{aligned}$$

We denoted by $\mathbf{n} \times \dot{\mathbf{x}}$ the cross product of the three dimensional vectors $\mathbf{n} \in \mathbb{S}^2$ and $\dot{\mathbf{x}} \in \mathbb{R}^3$. In the limit as $\varepsilon \to 0$, the angular orientation $\mathbf{n}$ and the physical velocity $\dot{\mathbf{x}}$ are respectively constrained to be collinear, positively collinear, and orthogonal if the path cost remains bounded, due to ε^{-2} penalization terms.

4.5 Reeds-Shepp Models in $\mathbb{R}^3 \times \mathbb{S}^2$

Our implementation relies on a parametrization of the sphere $\mathbb{S}^2$ using two Euler angles $\theta \in [0, \pi]$ and $\phi \in [0, 2\pi]$, with appropriate boundary conditions

$$\mathbf{n}(\theta, \phi) := (\cos\theta, \sin\theta\cos\phi, \sin\theta\sin\phi).$$

The Reeds-Shepp model and the dual variant, but not the forward variant, also make sense if $\mathbf{n}$ belongs to the projective orientation space $\mathbb{P}^2 = \mathbb{S}^2/\{-1, 1\}$, which is parametrized as above except that $\theta \in [0, \pi/2]$ only. The same angular spacing $h_\phi = 2\pi/n_\phi$ is used for discretizing the angles θ and ϕ, hence the Cartesian grid devoted to the angular $\mathbb{S}^2$ (resp. $\mathbb{P}^2$) space has dimension $(n_\phi/2, n_\phi)$ (resp. $(n_\phi/4, n_\phi)$). See Figure 19.

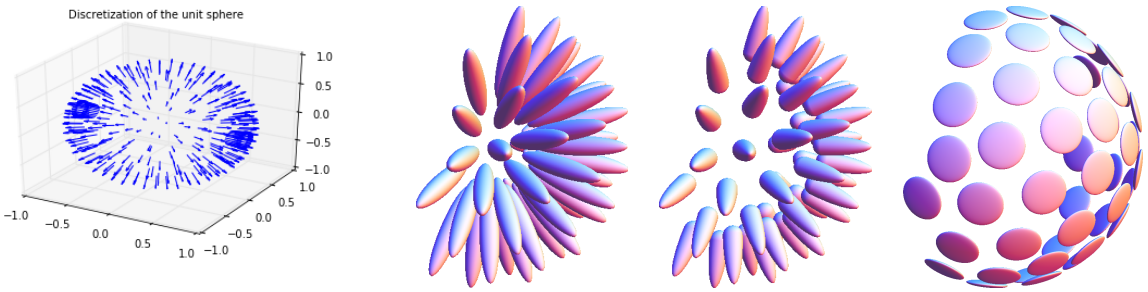

Figure 19: Left: discretization of the unit sphere $\mathbb{S}^2$, using 24×12 points, used for the presented numerical experiments in $\mathbb{R}^3 \times \mathbb{S}^2$. The next three pictures show the sets $\{\dot{\mathbf{x}} \in \mathbb{R}^3; \mathcal{F}^{\varepsilon}_{(0,\mathbf{n})}(\dot{\mathbf{x}}, 0) \leq 1\}$, with the relaxation parameter $\varepsilon = 0.2$ and for several orientations $\mathbf{n} \in \mathbb{S}^2$, where $\mathcal{F}$ is successively the metric of the Reeds-Shepp model (needles), of the forward variant (half needles), and of the dual variant (plates).

The HFM library executables ending with 'ReedsShepp3' or 'ReedsSheppForward3' must be selected to solve these problems. In the case of the Reeds-Shepp model, the projective angular space is imposed by the key-value pair **projective**:1, and the dual model is activated by **dual**:1. The curvature penalization and the relaxation parameter need to be set, e.g. **xi**:1 and **eps**:0.2. In the examples presented, the discretization grid size is defined as **dims**:$[60, 40, 25, n_\theta, 24]$, where $n_\theta = 6$ for the projective angular space $\mathbb{P}^2$, and $n_\theta = 12$ in the case of $\mathbb{S}^2$, see Figure 19. The cost function is specified by **cost**:arr, where arr is either a constant, a two-dimensional array ($c = c(\mathbf{n})$), a three dimensional array ($c = c(\mathbf{x})$), or a five-dimensional array ($c = c(\mathbf{x}, \mathbf{n})$), of suitable dimensions.

A test case inspired by tubular structure segmentation is presented in Figure 20. The cost function is small in the neighborhood of two curves, which respectively have low curvature and low torsion, and is large elsewhere. The common endpoints of the curves are respectively used as the seed of the front propagation, and the tip from which to backtrack a geodesic. The minimal paths selected

by the Reeds-Shepp and the Reeds-Shepp dual model go along the low curvature and low torsion curve respectively, as was expected and desired. A second test case, inspired by motion planning, is displayed in Figure 21. The backtracked curves are in that case smooth and with well distributed curvature, up to the occasional and expected singularities: cusps for the Reeds-Shepp model, and in-place rotations for the forward variant. Computation times[25] are respectively around $5s$ and $40s$ in the tubular segmentation and motion planning related experiments, despite the similar grid sizes, because the front propagation can be aborted earlier in the first case.

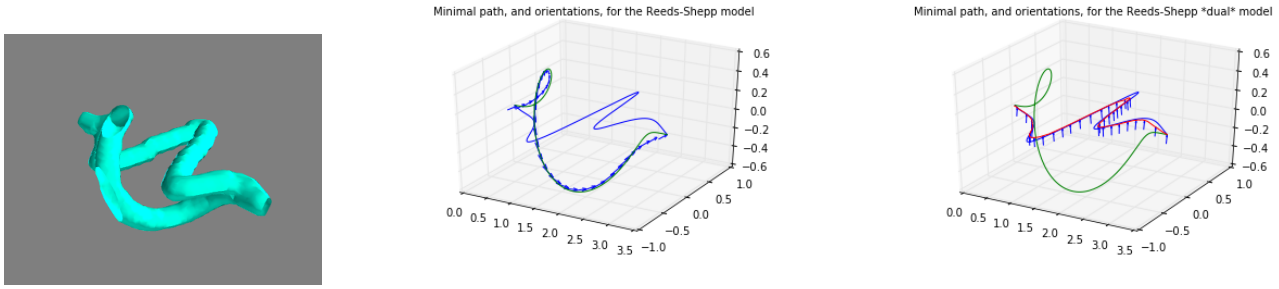

Figure 20: Illustration of the three dimensional Reeds-Shepp model and of its dual variant. Left: level set of the cost function $c = c(\mathbf{x})$, which is small in the neighborhood of two curves, with respectively low torsion and low curvature. Center: the minimal path for the Reeds-Shepp model follows the low curvature curve. The path $\gamma(t) = (\mathbf{x}(t), \mathbf{n}(t))$ exists in $\mathbb{R}^3 \times \mathbb{S}^2$, and its points are displayed as small arrows with origin $\mathbf{x}(t)$ and direction $\mathbf{n}(t)$. Right: the minimal path for the Reeds-Shepp *dual* model follows the low torsion curve.

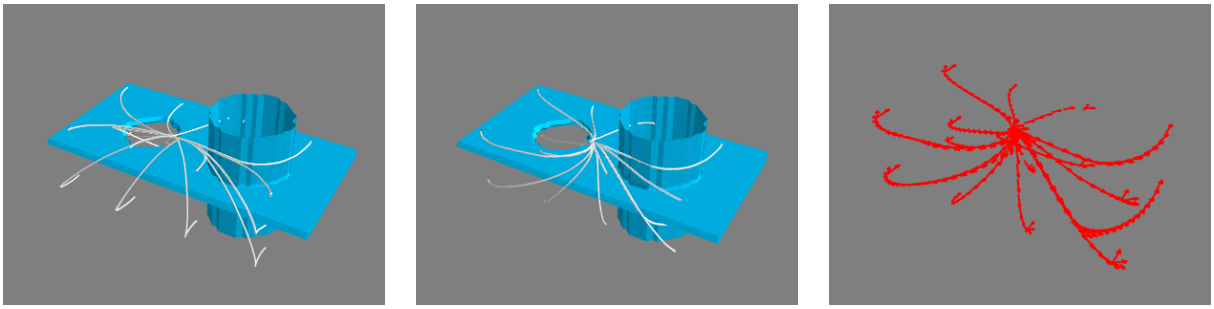

Figure 21: Minimal paths for the the three dimensional Reeds-Shepp model (left), and of its forward variant (center), in a domain with obstacles and a constant cost function. Note the *cusps* (left), where the vehicle orientation is reversed, for the original model. Right: the forward model features *in place rotations* at some of the path endpoints.

5 Selected Applications

In this section, we present selected applications of the HFM library. The difference with the numerical experiments discussed in the previous section lies in the intent and focus: modeling aspects are more emphasized here, and the datasets come from other application fields (although some are synthetic), instead of being specifically designed so as to illustrate the functionalities of the HFM library.

5.1 An Interpretation of Poggendorff's Visual Illusions

We present two visual illusions due to Poggendorff, and their explanation according to the work [24]. This research builds on the works [47, 9], which have shown that the first layer V1 of the visual

[25] On a 2.7GHz processor using a single core.

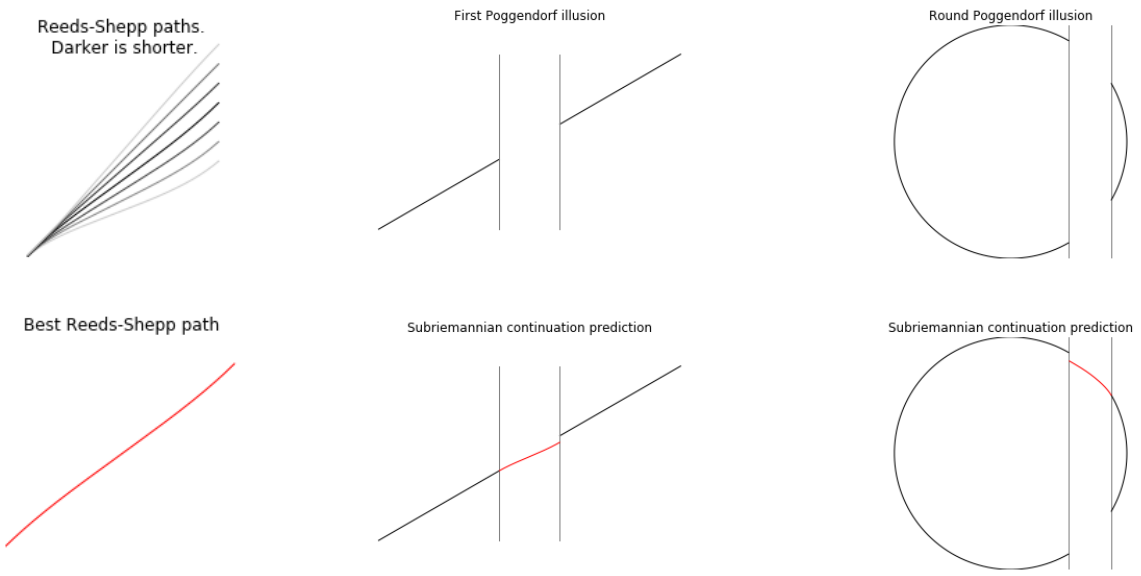

Figure 22: Poggendorff's illusions and their interpretation. Left: consider a seed point $\mathbf{p}$ in the configuration space $\mathbb{M} := \mathbb{R}^2 \times \mathbb{S}^1$, and a family of tips $\mathbf{q}_1, \cdots, \mathbf{q}_n \in \mathbb{M}$. According to the Petitot Citti-Sarti model, the visual cortex $V1$ will infer a connection between the seed and the closest tip w.r.t. the sub-Riemannian Reeds-Shepp model (with parameter ξ to be determined). Center and right: this principle explains two visual illusions due to Poggendorff, where the two parts of an occluded straight line or circle appear to mismatch.

cortex could be regarded as a biological implementation of the manifold $\mathbb{R}^2 \times \mathbb{P}^1$, equipped with the sub-Riemannian structure defining the Reeds-Shepp model.

Consider an image displaying a curve ending at a point $\mathbf{x}_* \in \mathbb{R}^2$ with tangent orientation θ_*, and a curve starting at another point $\mathbf{x}^*$ with tangent orientation θ^*. Under some conditions, a human presented with this image will associate the two curves and infer a connection between them [20]. In everyday experience, a similar phenomenon helps us guess the shape of objects which boundary is partially occluded. The curve reconstructed by the visual system is, according to the works cited above, the (planar projection of the) minimal path joining the oriented endpoints $\mathbf{p}_* := (\mathbf{x}_*, \theta_*)$ and $\mathbf{p}^* := (\mathbf{x}^*, \theta^*)$ with respect to the Reeds-Shepp model. The parameter ξ, which determines the amount of curvature penalization in this model, depends on the scale at which the image is displayed, and is adjusted by hand in our experiments. The cost function is chosen constant $c \equiv 1$ in our experiments, although a more complex data-driven construction is considered in the original work [24].

Assume now that several endpoints $(\mathbf{x}_i^*, \theta_i^*)_{i \in I}$ are present in the image, instead of a single one. Then the visual system infers a curve joining the source point $(\mathbf{x}_*, \theta_*)$ to the closest endpoint, again with respect to the distance defined by the Reeds-Shepp model, a process called perceptual grouping [3]. The images created by Poggendorff put this physiological process in evidence by displaying a partially occluded straight line or circle. To the eye, the true endpoints of the geometrical figure seem bizarrely misaligned, see Figure 22. This is a visual illusion, highlighting the difference between (a) the Reeds-Shepp geodesic toward the closest point across the occluded region (with the correct orientation), and (b) the continuation by a straight segment or an arc of circle that would be "logical" in the presented image.

5.2 Motion Planning

We illustrate in Figure 23 a two dimensional motion planning problem, namely finding the optimal path towards the exit of a building[26]. The Isotropic, Reeds-Shepp, Reeds-Shepp forward, Elastica and Dubins models are compared, and their distinctive features are clearly visible, see Section 4.4.

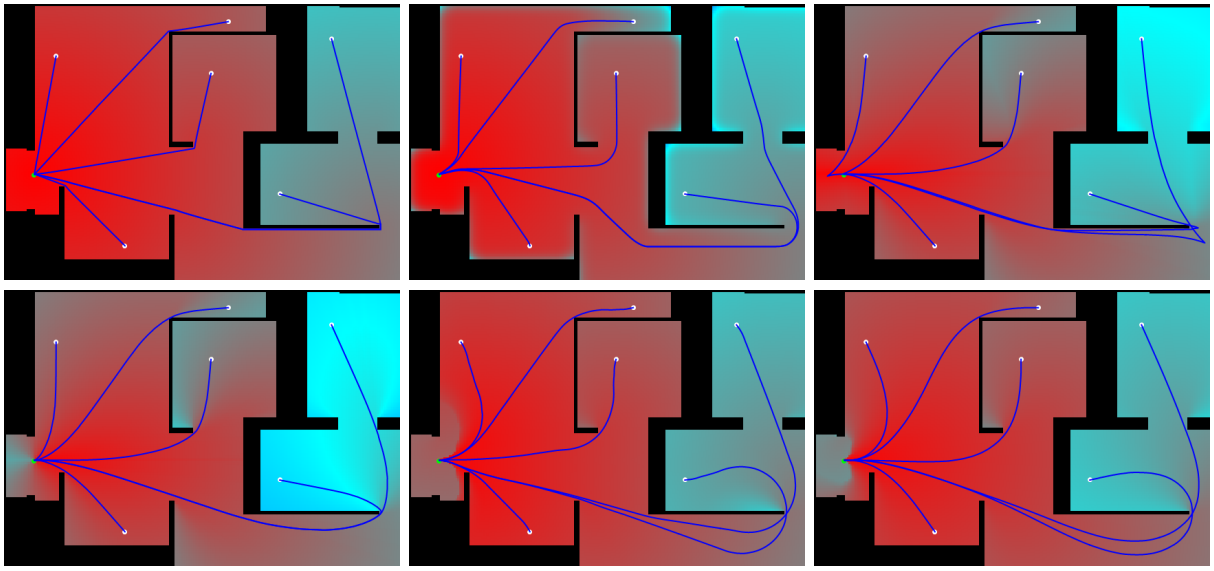

Figure 23: Optimal paths coming from various models, towards the exit of a building. The green point is the source, the white points are endpoints, black parts indicate the walls and the color indicates the distance (if necessary minimized over orientation). Top row, left to right: isotropic, isotropic with cost based on Gaussian blurred image, Reeds-Shepp model. Bottom row, left to right: forward-only Reeds-Shepp model, Dubins's model, Euler-Mumford elastica model.

The authors readily acknowledge that shortest paths (with respect to any reasonable metric) are a quite questionable approximation of natural paths, in view of the potential tradeoffs with safety, comfort or computational effort. For instance, the optimal computed trajectories are tangent to the wall corners, which would be hazardous in practice in case someone comes in the opposite direction. These trajectories also anticipate obstacles before they enter the line of sight. Paths closer to e.g. natural pedestrian motion are obtained by penalizing motion close to the walls, see again Figure 23. See [10] and references therein on the topic of pedestrian crowd motion models based on eikonal equations. Optimal paths make most sense when the environnement is well known and path length (measured with the appropriate metric) is the dominant criterion for success: for instance if one is playing a race, or trying to avoid a surveillance system [41].

5.3 Tubular Structure Segmentation

Tubular structure segmentation in medical images is one of the main applications intended for the HFM library. Early versions, and related software of the first author, are used for that purpose in [12, 52, 15, 22, 14, 33, 3]. Vessel extraction in images of the retina is illustrated in Figure 24, using two popular models involving respectively radius and orientation lifting. Let us briefly recall the principles underlying this approach.

Consider an image, e.g. in grayscale and represented by a function $I : U \to [0, 1]$, displaying a family of possibly overlaid tubular structures. Segmentation methods based on minimal paths attempt to extract the centerlines of these tubular structures as geodesics with respect to a suitable metric, joining endpoints which are either prescribed by the user or automatically detected by another

[26] The experiment uses a partial map of Museum Georges Pompidou in Paris

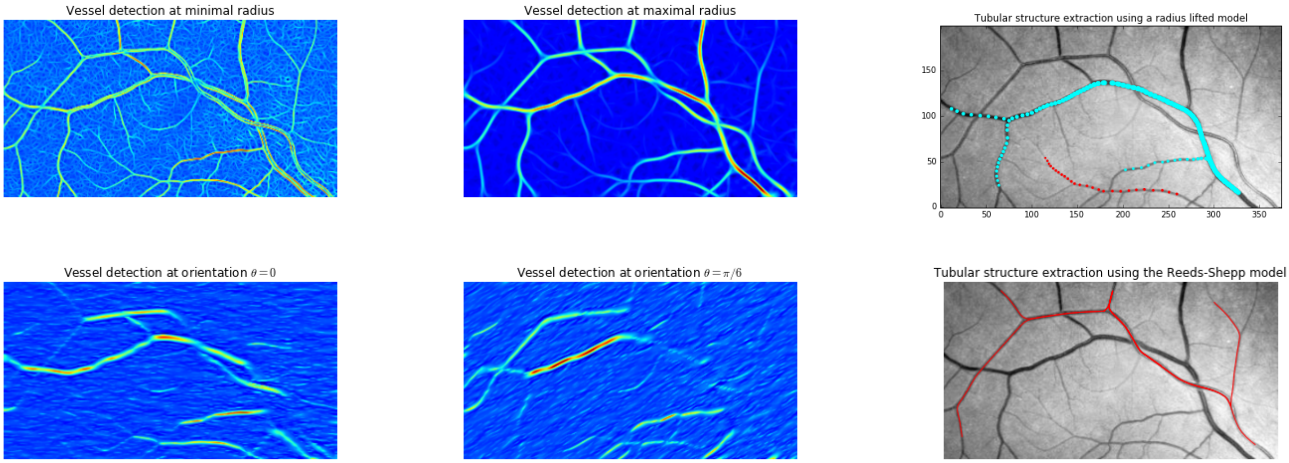

Figure 24: Vessel segmentation in an image of the retina, using two dimension lifting techniques. Top: in a radius-lifted model, a preliminary image filtering is used to detect vessels of small (left) and large (center) scale. The backtracked path contains information about the vessel radii (right). Bottom: in an orientation lifted model, a preliminary image filtering detects vessel bits of different orientations (left and center). Vessel extraction using the Reeds-Shepp model (right).

method. They are frequently combined with *dimension lifting* techniques: a cost function $c : U \times A \to]0, \infty[$ is defined over the product of the image domain U with an abstract parameter space A. In practice, the latter often represents a range of radii $A = [r, R]$, of grayscale intensities $A = [0, 1]$, or of angular orientations $A = \mathbb{S}^1$, see e.g. [31, 46]. The local cost $c(\mathbf{x}, a)$ is data-driven and built during a preliminary filtering of the input image, see e.g. [21]. The quantity $c(\mathbf{x}, a)$ should be small iff a tubular structure having feature a is likely present at the position $\mathbf{x}$ in the image.

The purpose of dimension lifting is two-fold. First: robustly extract the additional feature of interest, in addition to the vessel centerline, for applications in medical diagnostic. Vessel radius and curvature, inferred from orientation in the latter case, are particularly important in this respect. Second: disentangle the overlaid tubular structures, by separating them in three dimensional space, based on the additional feature parameter a. This helps avoid or reduce the "shortcut" and "leak" type artifacts, which often plague minimal path methods for tubular structure extraction. Figure 24 illustrates radius lifting, combined with an isotropic metric model, and orientation lifting, combined with the Reeds-Shepp model.

5.4 Segmentation in 3D DMRI Data

We illustrate three dimensional tubular structure segmentation in (simulated) dMRI data. This medical imaging technique measures diffusivity at each point $\mathbf{x} \in \mathbb{R}^3$, and in each direction $\mathbf{n} \in \mathbb{S}^2$. The input data is therefore intrinsically defined over the configuration space $\mathbb{R}^3 \times \mathbb{S}^2$, in contrast with Section 5.3 where dimension lifting was an artificially introduced image processing technique.

We use a digital phantom dataset that was constructed, using the dMRI simulation method [11], for the ISBI 2013 reconstruction challenge, where it was used as a benchmark for tracking methods [17]. The dataset itself consists of a fairly challenging configuration of 27 simulated white matter bundles, inside a spherical volume. The entire volume has a size of $50 \times 50 \times 50$ voxels. In the same volume, there are three spherical regions with isotropic diffusion.

We use a constrained spherical deconvolution approach [60] to estimate from the raw simulated dMRI data a fiber orientation distribution (FOD), that in each voxel indicates the probability of having a fiber in that direction. The density of this FOD is denoted by $f : \mathbb{R}^3 \times \mathbb{S}^2 \to \mathbb{R}^+$. Then following the approach in [22], we use the FOD to construct the cost function as follows: for any

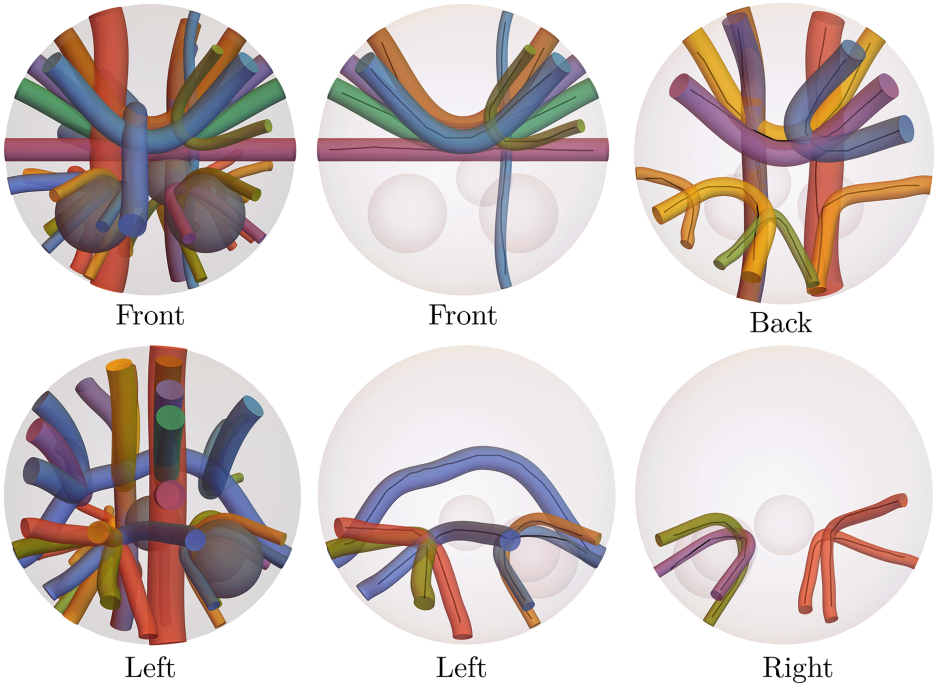

Figure 25: Visualization of the ISBI dataset from different viewpoints. The two left-most images show the full dataset, the other four images show a selection of the bundles. The black lines inside these bundles indicate the geodesics obtained with fast marching.

$\mathbf{p} = (\mathbf{x}, \mathbf{n}) \in \mathbb{R}^3 \times \mathbb{S}^2$

$$c(\mathbf{p}) = \left(1 + \sigma \left| \frac{f(\mathbf{p})}{||f||_\infty} \right|^p \right)^{-1}, \qquad \sigma, p \geq 1. \tag{47}$$

The parameter σ influences how strongly the data should affect the optimal path, and the exponent p influences how 'sharp' the FOD profiles are. As a proof-of-concept experiment on this dataset, we consider the case where the seed point and end point of a bundle are given and we compute the shortest path connecting the two. The path should then at least stay inside the bundle. We use the Reeds-Shepp forward model, with parameters $\sigma = 3$, $p = 1$, $\xi = 0.5$, $\varepsilon = 0.1$, $n_\theta = 10$ and $n_\phi = 20$, resulting in 200 points in the discretization of the sphere. The geodesics (in black) stay in almost all cases entirely inside the volume of the corresponding bundle, as is displayed in Figure 25.

6 Conclusion

In this paper, we introduced the Hamiltonian Fast Marching (HFM) library, a state-of-the-art software for global path optimization with respect to non-isotropic metrics. Our numerical methods apply in particular to Riemannian metrics in dimensions two and three, possibly strongly anisotropic, using an adaptive discretization scheme based on lattice geometry techniques. We also globally optimize second order energies depending on the path curvature, by introducing a dimension lifted space equipped with a non-holonomic metric, suitably relaxed in the implementation. The classical models due to Reeds-Shepp, Euler-Mumford, and Dubins, are implemented, as well as numerous generalizations and variants which all have specific qualitative properties and use cases. A number of transversal methods are provided, such as backward algorithmic differentiation which is novel in this context. This software is provided in the form of a unified open source C++ code, with convenient interfaces for several major scripting languages.

The HFM library has already found several applications in the study of visual illusions, motion

planning, and image segmentation. We hope that this publication, and the series of introductory notebooks that are provided alongside, will help popularize these techniques and their present and future applications. The HFM library is part of an ongoing research effort and will continue to evolve. Considered additions include higher dimensional Riemannian metrics and a better enforcement of non-holonomy constraints.

A Dual Metric for Curvature Penalization

This appendix is devoted to formally deriving the expression of the dual metric associated with the Reeds-Shepp forward, Euler-Mumford, and Dubins model. It is announced in Section 1.2 and Section 2.4. We use a unified and systematic presentation that easily generalizes to e.g. the asymmetric variants (11), or the higher dimensional instantiations. See [39] for another approach.

For notational simplicity, let us fix a point $(\mathbf{x}, \theta) \in \mathbb{R}^2 \times \mathbb{S}^1$ of the configuration space of these models, and denote $F := \mathcal{F}_{(\mathbf{x},\theta)}$, $c := c(\mathbf{x}, \theta)$ and $\mathbf{n} := (\cos\theta, \sin\theta)$. Our first step is to justify the semi-explicit dual metric expression (8, right). Indeed, one has

$$\begin{aligned} F^*(\hat{\mathbf{x}}, \hat{\theta}) &= \sup_{(\dot{\mathbf{x}},\dot{\theta})\neq 0} \frac{\langle \hat{\mathbf{x}}, \dot{\mathbf{x}}\rangle + \hat{\theta}\dot{\theta}}{F(\dot{\mathbf{x}}, \dot{\theta})} = \sup_{\substack{(\dot{x},\dot{\theta})\neq 0 \\ \dot{x}\geq 0}} \frac{\langle \hat{\mathbf{x}}, \dot{x}\mathbf{n}\rangle + \hat{\theta}\dot{\theta}}{F(\dot{x}\mathbf{n}, \dot{\theta})} = \sup_{\dot{\theta}\in\mathbb{R}} \frac{\langle \hat{\mathbf{x}}, \mathbf{n}\rangle + \hat{\theta}\dot{\theta}}{c\,\mathcal{C}(\dot{\theta})} = \\ &= \frac{f(\langle \hat{\mathbf{x}}, \mathbf{n}\rangle, \hat{\theta})}{c}. \end{aligned}$$

We used successively (i) the definition (4) of the dual metric, (ii) the fact that $F(\dot{\mathbf{x}}, \dot{\theta}) = \infty$ unless the physical velocity is a non-negative multiple of the current orientation $\mathbf{n}$, (iii) the 1-homogeneity of F, allowing to assume[27] that $\dot{x} = 1$, and its explicit expression (8, left), and (iv) the definition (29) of the function f.

Our next objective is to compute the supremum, denoted $f(\hat{x}, \hat{\theta})$, of

$$\dot{\theta} \in \mathbb{R} \mapsto (\hat{x} + \hat{\theta}\dot{\theta})/\mathcal{C}(\dot{\theta}). \tag{48}$$

For that purpose we recall the specific instantiations $\mathcal{C}^{\mathrm{RS}}$, $\mathcal{C}^{\mathrm{EM}}$ and $\mathcal{C}^{\mathrm{D}}$ of the cost function $\mathcal{C}$ penalizing curvature, see (10). For notational simplicity, and up to rescaling $\dot{\theta}$, we assume that $\xi = 1$.

$$\mathcal{C}^{\mathrm{RS}}(\dot{\theta}) = \sqrt{1 + \dot{\theta}^2}, \qquad \mathcal{C}^{\mathrm{EM}}(\dot{\theta}) = 1 + \dot{\theta}^2, \qquad \mathcal{C}^{\mathrm{D}}(\dot{\theta}) = \begin{cases} 1 & \text{if } \dot{\theta} \leq 1, \\ \infty & \text{else.} \end{cases}$$

In the Dubins case, the map (48) is linear on the interval $[-1, 1]$, and vanishes elsewhere, hence its maximum is $f(\hat{x}, \hat{\theta}) := \max\{\hat{x} - \hat{\theta}, \hat{x} + \hat{\theta}, 0\}$. In the Reeds-Shepp and Euler-Mumford cases, the supremum of (48) is attained either as $\dot{\theta} \to \pm\infty$, or for a finite value $\dot{\theta} \in \mathbb{R}$ obeying the optimality condition

$$\hat{\theta}\mathcal{C}(\dot{\theta}) = (\hat{x} + \hat{\theta}\dot{\theta})\mathcal{C}'(\dot{\theta}). \tag{49}$$

For the Reeds-Shepp curvature cost, observing that $\mathcal{C}^{\mathrm{RS}}(\dot{\theta})' = \dot{\theta}/\sqrt{1 + \dot{\theta}^2}$, relation (49) yields

$$\hat{\theta}\sqrt{1 + \dot{\theta}^2} = (\hat{x} + \hat{\theta}\dot{\theta})\frac{\dot{\theta}}{\sqrt{1 + \dot{\theta}^2}} \quad \Leftrightarrow \quad \hat{\theta}(1 + \dot{\theta}^2) = (\hat{x} + \hat{\theta}\dot{\theta})\dot{\theta} \quad \Leftrightarrow \quad \hat{\theta} = \hat{x}\dot{\theta}.$$

[27] We need not consider the case $\dot{x} = 0$, since F was extended to the line $\{(\mathbf{0}, \dot{\theta}); \dot{\theta} \in \mathbb{R}\}$ by its lower semi continuous envelope.

Thus $\dot\theta = \hat\theta/\hat x$, provided $\hat x \neq 0$, and by (48) therefore

$$f(\hat x, \hat\theta) = (\hat x + \hat\theta\frac{\hat\theta}{\hat x})/\sqrt{1+(\frac{\hat\theta}{\hat x})^2} = \sqrt{\hat x^2+\hat\theta^2},$$

as announced (since $\theta \to \pm\infty$ is sub-optimal). In addition

$$f(0,\hat\theta) = \sup_{\dot\theta\in\mathbb{R}} \hat\theta\dot\theta/\sqrt{1+\dot\theta^2} = |\hat\theta|,$$

in the limit $\dot\theta \to \mathrm{sign}(\hat\theta)\infty$, hence the previous expression still holds in the case $\hat x = 0$.

For the Euler-Mumford curvature cost, observing that $\mathcal{C}^{\mathrm{EM}}(\dot\theta)' = 2\dot\theta$ we rewrite (49) as

$$\hat\theta(1+\dot\theta^2) = (\hat x + \hat\theta\dot\theta)2\dot\theta \quad\Leftrightarrow\quad \dot\theta^2 + 2\dot\theta\frac{\hat x}{\hat\theta} - 1 = 0 \quad\Leftrightarrow\quad \dot\theta = \dot\theta_\pm = -\frac{\hat x}{\hat\theta} \pm \sqrt{1+(\frac{\hat x}{\hat\theta})^2}, \tag{50}$$

assuming $\hat\theta \neq 0$. Inserting these roots in (48) we obtain

$$\frac{\hat x + \hat\theta\dot\theta_\pm}{\mathcal{C}^{\mathrm{EM}}(\dot\theta_\pm)} = \frac{\hat\theta}{\mathcal{C}^{\mathrm{EM}}(\dot\theta_\pm)'} = \frac{\hat\theta}{2\dot\theta_\pm} = \frac{-\hat\theta\hat\theta_\mp}{2} = \frac{\hat x \mp \sqrt{\hat x^2+\hat\theta^2}}{2}, \tag{51}$$

where we used successively (i) the identity (49) defining the roots, (ii) the expression $\mathcal{C}^{\mathrm{EM}}(\dot\theta)' = 2\dot\theta$, (iii) the fact that $\dot\theta_+\dot\theta_- = -1$, which follows from (50, center) and (iv) the explicit expression of $\dot\theta_\pm$. Choosing the "+" sign in (51, right) yields the largest value, hence $f(\hat x,\hat\theta) = (\hat x + \sqrt{\hat x^2+\hat\theta^2})/2$ (since $\theta\to\pm\infty$ is sub-optimal). In the limit case $\hat\theta = 0$, this expression becomes $f(\hat x, 0) = (\hat x + |\hat x|)/2$, which is still correct, and is attained for $\dot\theta = 0$ if $\hat x > 0$, and as $\dot\theta \to \infty$ otherwise.

B Selling's Algorithm

Our discretization of Riemannian eikonal equations strongly relies on a particular decomposition of positive definite tensors, see Proposition 2.2. In this Appendix, we describe its algorithmic computation in dimension $d \in \{2,3\}$, as it is implemented in the HFM library. For that purpose, we need to introduce a few concepts from lattice geometry.

Definition B.1. *A superbase of $\mathbb{Z}^d$ is a $(d+1)$-tuple $(\mathbf{e}_0,\cdots,\mathbf{e}_d) \in (\mathbb{Z}^d)^{d+1}$ such that $|\det(\mathbf{e}_1,\cdots,\mathbf{e}_d)| = 1$ and $\mathbf{e}_0+\cdots+\mathbf{e}_d = 0$.*

Any superbase can be used to decompose any 2-tensor D, as follows

$$D = -\sum_{0\leq i<j\leq d} \langle \mathbf{e}_i, D\mathbf{e}_j\rangle\, \mathbf{v}_{ij}\otimes\mathbf{v}_{ij}, \tag{52}$$

where $\mathbf{v}_{ij} := \mathbf{e}_k^\perp$ in dimension $d=2$ and with $\{i,j,k\} = \{0,1,2\}$, and $\mathbf{v}_{ij} := \mathbf{e}_k\times\mathbf{e}_l$ in dimension $d=3$ and with $\{i,j,k,l\} = \{0,1,2,3\}$. See Lemma 4.4 in [39] for a proof. We are interested in tensor decompositions with non-negative coefficients (24), which in the case of (52) is equivalent to a geometrical property of the superbase.

Definition B.2. *A superbase $(\mathbf{e}_0,\cdots,\mathbf{e}_d)$ of $\mathbb{Z}^d$ is said D-obtuse, where $D\in \mathrm{S}^{++}(\mathbb{R}^d)$, iff $\langle D\mathbf{e}_i,\mathbf{e}_j\rangle \leq 0$ for all $0\leq i<j\leq d$.*

In dimension $d\in\{2,3\}$, Algorithm 2 due to Selling [55] takes a positive definite tensor D as input, and outputs a D-obtuse superbase. This algorithm is implemented in function:

```
void BasisReduction<TS,TD,VD>::
ObtuseSuperbase(const SymmetricMatrixType & m, SuperbaseType & sb)
```

located in file: `JMM_CPPLibs-master/LinearAlgebra/LinearAlgebra/BasisReduction.hpp`

Algorithm 2 Selling's algorithm [55], $d \in \{2,3\}$

Input: A positive definite tensor $D \in \mathrm{S}^{++}(\mathbb{R}^d)$, an arbitrary superbase $b = (\mathbf{e}_0, \cdots, \mathbf{e}_d)$ of $\mathbb{Z}^d$.
Output: A D-obtuse superbase.
While there exists $0 \leq i < j \leq d$ such that $\langle \mathbf{e}_i, D\mathbf{e}_j \rangle > 0$ **do**
    If $d = 2$ **then** $b \leftarrow (-\mathbf{e}_i, \mathbf{e}_j, \mathbf{e}_i - \mathbf{e}_j)$,
    If $d = 3$ **then** $b \leftarrow (-\mathbf{e}_i, \mathbf{e}_j, \mathbf{e}_k + \mathbf{e}_i, \mathbf{e}_l + \mathbf{e}_i)$.

Termination guarantee. Consider the energy $\mathcal{E}(\mathbf{e}_0, \cdots, \mathbf{e}_d)$, defined by $\sum_{i=0}^{2} \|\mathbf{e}_i\|_D^2$ in dimension $d = 2$ and by $\sum_{i=0}^{3} \|\mathbf{e}_i\|_D^2 + \frac{1}{2}\sum_{i<j} \|\mathbf{e}_i + \mathbf{e}_j\|_D^2$ in dimension $d = 3$. This quantity decreases by $4\langle \mathbf{e}_{i_*}, D\mathbf{e}_{j_*} \rangle > 0$ at each iteration, where i_* and j_* are the indices appearing in the While condition, as can easily be checked by expressing the present and next superbase energy in terms of the scalar products $(\langle \mathbf{e}_i, \mathbf{e}_j \rangle)_{i,j=0}^{d}$. Since there exists only finitely many superbases which energy $\mathcal{E}$ is below any given bound, the algorithm must terminate.

C Norm of the Geodesic Flow

This section is devoted to the proof of Proposition 3.1. Our first step is to rewrite the announced result in a more abstract framework, keeping only the minimal assumptions.

Proposition C.1. *Let $I \geq 1$ and let $H : \mathbb{R}^I \to [0, \infty[$ be convex and positively 2-homogeneous. Let $\mathbb{E}$ be a finite dimensional vector space, and let $\dot{\mathbf{e}}_1, \cdots, \dot{\mathbf{e}}_I \in \mathbb{E}$. Let $\mathcal{H} : \mathbb{E}^* \to [0, \infty[$ and $\mathcal{F} : \mathbb{E} \to [0, \infty]$ be defined by*

$$\mathcal{H}(\hat{\mathbf{p}}) := H(\langle \hat{\mathbf{p}}, \dot{\mathbf{e}}_1 \rangle, \cdots, \langle \hat{\mathbf{p}}, \dot{\mathbf{e}}_I \rangle), \qquad \frac{1}{2}\mathcal{F}(\dot{\mathbf{p}})^2 := \sup_{\hat{\mathbf{p}} \in \mathbb{E}^*} \langle \hat{\mathbf{p}}, \dot{\mathbf{p}} \rangle - \frac{1}{2}\mathcal{H}(\hat{\mathbf{p}}). \tag{53}$$

Consider $\delta = (\delta_1, \cdots, \delta_n) \in \mathbb{R}^n$, and assume that $H(\delta) = 1$ and that H is differentiable at δ. Then denoting

$$\dot{\mathbf{v}} := \frac{1}{2} \sum_{1 \leq i \leq I} \partial_i H(\delta) \dot{\mathbf{e}}_i, \tag{54}$$

one has $\mathcal{F}(\dot{\mathbf{v}}) \leq 1$. In addition, equality holds if there exists $\hat{\mathbf{v}} \in \mathbb{E}^*$ such that $\delta_i = \langle \hat{\mathbf{v}}, \dot{\mathbf{e}}_i \rangle$ for all $1 \leq i \leq I$.

Before turning to the proof, we make the connection with Proposition C.1, which assumptions are scattered over the paper. The assumptions of positivity, convexity, and two homogeneity of the function H, and the equality (53) (left) follow from the general expression (14) of the Hamiltonian (understood, obviously, with an $=$ sign, instead of the informal $\approx$). The element $\delta \in \mathbb{R}^n$ gathers the finite differences $\delta_i = U(\mathbf{p}) - U(\mathbf{p} - h\dot{\mathbf{e}}_i)$ of the map U in Proposition C.1, and the identity $H(\delta) = 1$ expresses that U is a solution to the discretized PDE (34) (left). Finally, the vector $\dot{\mathbf{v}}$ defined by (54) generalizes the expression (40) which is limited to Hamiltonians of the specific form (33).

Proof of Proposition C.1. The result follows from a rather direct computation, presented below. Before that, we observe that (53, right, rhs) is always non-negative, by choosing $\hat{\mathbf{p}} = 0$ and observing that $H(0) = 0$ by homogeneity. The function $\mathcal{F}$ is thus well defined, as its square root, with values

in $[0,\infty]$. Let $\hat{\mathbf{p}} \in \mathbb{E}^*$ be arbitrary, and let $\alpha = (\langle \hat{\mathbf{p}}, \dot{\mathbf{e}}_i \rangle)_{i=1}^I \in \mathbb{R}^I$. Then one has

$$\begin{aligned} 2\langle \hat{\mathbf{p}}, \dot{\mathbf{v}} \rangle - \mathcal{H}(\hat{\mathbf{p}}) &= \sum_{1 \le i \le I} \partial_i H(\delta) \langle \hat{\mathbf{p}}, \dot{\mathbf{e}}_i \rangle - H(\langle \hat{\mathbf{p}}, \dot{\mathbf{e}}_1 \rangle, \cdots, \langle \hat{\mathbf{p}}, \dot{\mathbf{e}}_I \rangle) \\ &= \langle \mathrm{d}H(\delta), \alpha \rangle - H(\alpha) \\ &= \langle \mathrm{d}H(\delta), \alpha - \delta \rangle + 2H(\delta) - H(\alpha) \\ &\le H(\alpha) - H(\delta) + 2H(\delta) - H(\alpha) \\ &= H(\delta) = 1. \end{aligned} \tag{55}$$

We use in each line successively: (i) definitions (53, left) and (54), of $\dot{\mathbf{v}}$ and $\mathcal{H}$, (ii) the definition of $\alpha \in \mathbb{R}^n$, (iii) Euler's identity $\langle \mathrm{d}H(\delta), \delta \rangle = 2H(\delta)$ which follows from the 2-homogeneity of H, (iv) the convexity inequality $\langle \mathrm{d}H(\delta), \alpha - \delta \rangle \le H(\alpha) - H(\delta)$, (v) the assumption $H(\delta) = 1$.

Taking the supremum of $\hat{\mathbf{p}} \in \mathbb{E}^*$ we obtain $\mathcal{F}(\dot{\mathbf{v}})^2 \le 1$, thus $\mathcal{F}(\dot{\mathbf{v}}) \le 1$ as announced. Finally, under the additional assumption, one can test against the co-vector $\hat{\mathbf{p}} := \hat{\mathbf{v}}$, so that $\alpha = \delta$. In that case (55) becomes an equality and therefore $\mathcal{F}^*(\dot{\mathbf{v}})^2 \ge 1$, thus $\mathcal{F}^*(\dot{\mathbf{v}}) = 1$ as announced. □

D Axes Ordering and Pixel Area

Software visualization packages have introduced an unfortunately large number of incompatible ways to display a two or three dimensional array of values. In this appendix, we discuss the conventions chosen in the HFM library, and how to interface with other common tools.

We assume that the HFM library is provided with the following parameters: **dims**:$(n_1, \cdots, n_d)$ the size of the discretization grid, **origin**:$(p_1, \cdots, p_d)$ the bottom left corner of the domain, and **gridscale**:h the length of a pixel side. This defines a box-shaped domain, which we split in an almost everywhere disjoint family of cubes.

$$\overline{\Omega} = \prod_{1 \le i \le d} [p_i, p_i + h n_i] = \bigcup_{\mathbf{0} \le \mathbf{m} < \mathbf{n}} (\mathbf{p} + h\mathbf{m} + hC).$$

On the right-hand side, we denoted $\mathbf{p} := (p_1, \cdots, p_d)$, $\mathbf{n} := (n_1, \cdots, n_d)$, and introduced the unit cube $C = [0,1]^d$. The inequality $\mathbf{m} < \mathbf{n}$ is understood componentwise, i.e. $m_i < n_i$ for all $1 \le i \le d$, and the notation $\mathbf{q} + hC$ means that the cube C is rescaled by the factor h and then translated by the vector $\mathbf{q}$.

An array of suitable dimensions $U : \prod_{i=1}^d [\![0, n_i[\![\to \mathbb{R}$ defines a piecewise constant function $u : \overline{\Omega} \to \mathbb{R}$ on the cube partition, except on the common facets of neighbor cubes. In other words $u_{|K_{\mathbf{m}}} := U(\mathbf{m})$ on each $K_{\mathbf{m}} := \mathbf{p} + h\mathbf{m} + h\,\mathrm{int}(C)$, $\mathbf{0} \le \mathbf{m} < \mathbf{n}$.

Pixel area. The HFM library assumes, as shown above and in dimension two for simplicity, that the "pixel" of coordinates (i,j) occupies the square with corners $(i,j)h$ and $(i+1,j+1)h$. However, some visualization software define the "pixel" of index (i,j) as the square of side h centered at $(i,j)h$ or at $(i+1,j+1)h$. This raises (minor) issues when geodesic paths are overlaid on images, and can be adjusted for by shifting the **origin** by $(-0.5,-0.5)h$ or $(0.5,0.5)h$.

Array ordering. When displaying the data contained in a two or three-dimensional array, some software choose to assign the array indices (i,j,k) to the coordinate axes (X,Y,Z) in a transposed manner: $i \to Y$, $j \to X$, $k \to Z$. In addition, the array may internally be stored in row-major or column-major format. The **arrayOrdering** field accounts for these specificities, and must be set to 'RowMajor' (default value) for Python Mayavi®, 'YXZ_RowMajor' for Python PyPlot, or 'YXZ_ColumnMajor' for Matlab®.

Acknowledgements

This work was partly supported by ANR research grant MAGA, ANR-16-CE40-0014.

Image Credits

Figure 24 based on data from the High-Resolution Fundus (HRF) Image Database. Figure 25 based on data from the ISBI 2013 reconstruction challenge. Other images produced by the authors.

References

[1] K. Alton and I.M. Mitchell, *An Ordered Upwind Method with Precomputed Stencil and Monotone Node Acceptance for Solving Static Convex Hamilton-Jacobi Equations*, Journal of Scientific Computing, 51 (2011), pp. 313–348. https://doi.org/10.1007/s10915-011-9512-4.

[2] M. Bardi and I. Capuzzo-Dolcetta, *Optimal Control and Viscosity Solutions of Hamilton-Jacobi-Bellman Equations*, Systems & Control: Foundations & Applications, Birkhäuser Boston, 1997. https://doi.org/10.1007/978-0-8176-4755-1_1.

[3] E.J. Bekkers, D. Chen, and J.M. Portegies, *Nilpotent Approximations of Sub-Riemannian Distances for Fast Perceptual Grouping of Blood Vessels in 2D and 3D*, Journal of Imaging and Vision, (2017). http://dx.doi.org/10.1007/s10851-018-0787-z.

[4] E. Bekkers, R. Duits, A. Mashtakov, and G. Sanguinetti, *A PDE Approach to Data-Driven Sub-Riemannian Geodesics in SE(2)*, SIAM Journal of Imaging Sciences, 8 (2015), pp. 2740–2770. https://doi.org/10.1137/15M1018460.

[5] F. Benmansour, G. Carlier, G. Peyré, and F. Santambrogio, *Derivatives with respect to metrics and applications: subgradient marching algorithm*, Numerische Mathematik, 116 (2010), pp. 357–381. https://doi.org/10.1007/s00211-010-0305-8.

[6] F. Benmansour and L.D. Cohen, *Tubular Structure Segmentation Based on Minimal Path Method and Anisotropic Enhancement*, International Journal of Computer Vision, 92 (2010), pp. 192–210. http://dx.doi.org/10.1007/s11263-010-0331-0.

[7] J-D. Boissonnat, A. Cérézo, and J. Leblond, *Shortest paths of bounded curvature in the plane*, Journal of Intelligent and Robotic Systems, 11 (1994), pp. 5–20. https://doi.org/10.1007/BF01258291.

[8] F. Bornemann and C. Rasch, *Finite-element Discretization of Static Hamilton-Jacobi Equations based on a Local Variational Principle*, Computing and Visualization in Science, 9 (2006), pp. 57–69. http://dx.doi.org/10.1007/s00791-006-0016-y.

[9] W.H. Bosking, Y. Zhang, B. Schofield, and D. Fitzpatrick, *Orientation selectivity and the arrangement of horizontal connections in tree shrew striate cortex*, Journal of Neuroscience, 17 (1997), pp. 2112–2127. http://dx.doi.org/10.1523/JNEUROSCI.17-06-02112.1997.

[10] E. Cartee and A. Vladimirsky, *Anisotropic Challenges in Pedestrian Flow Modeling*, Communications in Mathematical Sciences, (2018). http://dx.doi.org/10.4310/CMS.2018.v16.n4.a7.

[11] E. Caruyer, A. Daducci, M. Descoteaux, J-C. Houde, J-P. Thiran, and R. Verma, *Phantomas: a flexible software library to simulate diffusion MR phantoms*, May 2014.

[12] D. Chen, L. D. Cohen, and J. M. Mirebeau, *Vessel extraction using anisotropic minimal paths and path score*, in IEEE International Conference on Image Processing (ICIP), 2014, pp. 1570–1574. https://doi.org/10.1109/ICIP.2014.7025314.

[13] D. Chen, J-M. Mirebeau, and L.D. Cohen, *Finsler Geodesics Evolution Model for Region based Active Contours.*, in Proceedings of the British Machine Vision Conference (BMVC), 2016. https://dx.doi.org/10.5244/C.30.22.

[14] D. Chen, J-M. Mirebeau, and Laurent D. Cohen, *A New Finsler Minimal Path Model With Curvature Penalization for Image Segmentation and Closed Contour Detection*, Computer Vision and Pattern Recognition (CVPR), (2016), pp. 355–363. http://dx.doi.org/10.1109/CVPR.2016.45.

[15] D. Chen, J-M. Mirebeau, and L. D. Cohen, *Vessel tree extraction using radius-lifted keypoints searching scheme and anisotropic fast marching method*, Journal of Algorithms & Computational Technology, 10 (2016), pp. 224–234. https://doi.org/10.1177/1748301816656289.

[16] L.D Cohen and R. Kimmel, *Global minimum for active contour models: A minimal path approach*, International Journal of Computer Vision, 24 (1997), pp. 57–78. http://dx.doi.org/10.1109/CVPR.1996.517144.

[17] A. Daducci, E. J. Canales-Rodriguez, M. Descoteaux, E. Garyfallidis, Y. Gur, Y. C. Lin, M. Mani, S. Merlet, M. Paquette, A. Ramirez-Manzanares, M. Reisert, P. R. Rodrigues, F. Sepehrband, E. Caruyer, J. Choupan, R. Deriche, M. Jacob, G. Menegaz, V. Prčkovska, M. Rivera, Y. Wiaux, and J. P. Thiran, *Quantitative Comparison of Reconstruction Methods for Intra-Voxel Fiber Recovery From Diffusion MRI*, IEEE Transactions on Medical Imaging, 33 (2014), pp. 384–399. https://doi.org/10.1109/TMI.2013.2285500.

[18] E.W. Dijkstra, *A short introduction to the art of programming*, vol. 4, Technische Hogeschool Eindhoven Eindhoven, 1971.

[19] L. E. Dubins, *On Curves of Minimal Length with a Constraint on Average Curvature, and with Prescribed Initial and Terminal Positions and Tangents*, American Journal of Mathematics, 79 (1957), pp. 497–516. https://doi.org/10.2307/2372560.

[20] R. Duits, U. Boscain, F. Rossi, and Y. Sachkov, *Association Fields via Cuspless Sub-Riemannian Geodesics in SE(2)*, Journal of Mathematical Imaging and Vision, 49 (2013), pp. 384–417. https://doi.org/10.1007/s10851-013-0475-y.

[21] R. Duits, M. Felsberg, G. Granlund, and B. ter Haar Romeny, *Image Analysis and Reconstruction using a Wavelet Transform Constructed from a Reducible Representation of the Euclidean Motion Group*, International Journal of Computer Vision, 72 (2007), pp. 79–102. https://doi.org/10.1007/s11263-006-8894-5.

[22] R. Duits, S.P.L. Meesters, J-M. Mirebeau, and J.M. Portegies, *Optimal Paths for Variants of the 2D and 3D Reeds-Shepp Car with Applications in Image Analysis*, arXiv:1612.06137 [math], (2016). arXiv: 1612.06137.

[23] J. Fehrenbach and J-M. Mirebeau, *Sparse non-negative stencils for anisotropic diffusion*, Journal of Mathematical Imaging and Vision, 49 (2014), pp. 123–147. http://dx.doi.org/10.1007/s10851-013-0446-3.

[24] B. Franceschiello, A. Mashtakov, G. Citti, and A. Sarti, *Modelling of the Poggendorff Illusion via Sub-Riemannian Geodesics in the Roto-Translation Group*, in International Conference on Image Analysis and Processing, Springer, 2017, pp. 37–47. http://dx.doi.org/10.1007/978-3-319-70742-6_4.

[25] A. Fuster, T. Dela Haije, A. Tristán-Vega, B. Plantinga, C-F. Westin, and L. Florack, *Adjugate Diffusion Tensors for Geodesic Tractography in White Matter*, Journal of Mathematical Imaging and Vision, 54 (2016), pp. 1–14. https://doi.org/10.1007/s10851-015-0586-8.

[26] A. Griewank and A. Walther, *Evaluating Derivatives*, Principles and Techniques of Algorithmic Differentiation, Society for Industrial and Applied Mathematics, 2008. https://doi.org/10.1137/1.9780898717761.bm.

[27] S. Jbabdi, P. Bellec, R. Toro, J. Daunizeau, M. Pélégrini-Issac, and H. Benali, *Accurate anisotropic fast marching for diffusion-based geodesic tractography*, Journal of Biomedical Imaging, 2008 (2008), pp. 2–12. https://doi.org/10.1155/2008/320195.

[28] M. Kass, A. Witkin, and D. Terzopoulos, *Snakes: Active contour models*, International Journal of Computer Vision, 1 (1988), pp. 321–331. https://doi.org/10.1007/BF00133570.

[29] R. Kimmel and J.A. Sethian, *Computing geodesic paths on manifolds*, Proceedings of the National Academy of Sciences, 95 (1998), pp. 8431–8435. http://dx.doi.org/10.1073/pnas.95.15.8431.

[30] J-C. Latombe, *Robot motion planning*, vol. 124, Springer Science and Business Media, 2012. http://dx.doi.org/10.1007/978-1-4615-4022-9.

[31] H. Li and A. Yezzi, *Vessels as 4-D curves: Global minimal 4-D paths to extract 3-D tubular surfaces and centrelines*, IEEE Transactions on Medical Imaging, 26 (2007), pp. 1213–1223. http://dx.doi.org/10.1109/TMI.2007.903696.

[32] W. Liao, K. Rohr, and S. Wörz, *Globally Optimal Curvature-Regularized Fast Marching For Vessel Segmentation*, 2013, pp. 550–557. http://dx.doi.org/10.1007/978-3-642-40811-3_69.

[33] A. Mashtakov, R. Duits, Yu Sachkov, E. J. Bekkers, and I. Beschastnyi, *Tracking of Lines in Spherical Images via Sub-Riemannian Geodesics in SO(3)*, Journal of Mathematical Imaging and Vision, (2017), pp. 1–26. https://doi.org/10.1007/s10851-017-0705-9.

[34] J. Melonakos, E. Pichon, S. Angenent, and A. Tannenbaum, *Finsler Active Contours*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 30 (2008), pp. 412–423. https://doi.org/10.1109/TPAMI.2007.70713.

[35] J-M. Mirebeau, *Efficient fast marching with Finsler metrics*, Numerische Mathematik, 126 (2013), pp. 515–557. https://doi.org/10.1007/s00211-013-0571-3.

[36] ——, *Anisotropic Fast-Marching on Cartesian Grids Using Lattice Basis Reduction*, SIAM Journal on Numerical Analysis, 52 (2014), pp. 1573–1599. https://doi.org/10.1137/120861667.

[37] ——, *Anisotropic Fast Marching in ITK*, The Insight Journal, (2015). http://hdl.handle.net/10380/3518.

[38] ——, *Minimal stencils for discretizations of anisotropic PDEs preserving causality or the maximum principle*, SIAM Journal on Numerical Analysis, 54 (2016), pp. 1582–1611. http://dx.doi.org/10.1137/16M1064854.

[39] ——, *Fast Marching methods for Curvature Penalized Shortest Paths*, Journal of Mathematical Imaging and Vision, (2017). http://dx.doi.org/10.1007/s10851-017-0778-5.

[40] ——, *Riemannian fast-marching on cartesian grids using Voronoi's first reduction of quadratic forms*. 2017.

[41] J-M. Mirebeau and J. Dreo, *Automatic differentiation of non-holonomic fast marching for computing most threatening trajectories under sensors surveillance*, in Geometrical Science of Information, 2017. http://dx.doi.org/10.1007/978-3-319-68445-1_91.

[42] R. Montgomery, *A Tour of Subriemannian Geometries, Their Geodesics and Applications*, American Mathematical Society, Aug. 2006. http://dx.doi.org/10.1090/surv/091.

[43] D. Mumford, *Elastica and computer vision*, in Algebraic Geometry and its Applications, Springer, 1994, pp. 491–506. http://dx.doi.org/10.1007/978-1-4612-2628-4_31.

[44] P.Q. Nguyen and D. Stehlé, *Low-dimensional lattice basis reduction revisited*, in ANTS, Springer, 2004, pp. 338–357. http://dx.doi.org/10.1007/978-3-540-24847-7_26.

[45] A.M. Oberman, *Convergent Difference Schemes for Degenerate Elliptic and Parabolic Equations: Hamilton-Jacobi Equations and Free Boundary Problems*, SIAM Journal on Numerical Analysis, 44 (2006), pp. 879–895. https://doi.org/10.1137/S0036142903435235.

[46] M. Pechaud, R. Keriven, and G. Peyré, *Extraction of tubular structures over an orientation domain*, in IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops), IEEE, 2009, pp. 336–342. https://doi.org/10.1109/CVPR.2009.5206782.

[47] J. Petitot, *The neurogeometry of pinwheels as a sub-Riemannian contact structure*, Journal of Physiology-Paris, 97 (2003), pp. 265–309. http://dx.doi.org/10.1016/j.jphysparis.2003.10.010.

[48] G. Peyré, M. Péchaud, R. Keriven, and L.D. Cohen, *Geodesic Methods in Computer Vision and Graphics*, CGV, 5 (2010), pp. 197–397. https://doi.org/10.1561/0600000029.

[49] G. Randers, *On an Asymmetrical Metric in the Four-Space of General Relativity*, Physical Review, 59 (1941), pp. 195–199. https://doi.org/10.1103/PhysRev.59.195.

[50] J. A. Reeds and L. A. Shepp, *Optimal paths for a car that goes both forwards and backwards.*, Pacific Journal of Mathematics, 145 (1990), pp. 367–393. http://dx.doi.org/10.2140/pjm.1990.145.367.

[51] E. Rouy and A. Tourin, *A Viscosity Solutions Approach to Shape-From-Shading*, SIAM Journal on Numerical Analysis, 29 (1992), pp. 867–884. http://dx.doi.org/10.1137/0729053.

[52] G.R. Sanguinetti, E.J. Bekkers, R. Duits, M.H.J. Janssen, A. Mashtakov, and J-M. Mirebeau, *Sub-Riemannian Fast Marching in SE(2)*, in Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications, no. 9423 in Lecture Notes in Computer Science, Springer International Publishing, 2015, pp. 366–374. http://dx.doi.org/10.1007/978-3-319-25751-8_44.

[53] M. Schober, N. Kasenburg, A. Feragen, P. Hennig, and S. Hauberg, *Probabilistic Shortest Path Tractography in DTI. Using Gaussian Process ODE Solvers*, in Medical Image Computing and Computer-Assisted Intervention – MICCAI, Springer, Cham, Cham, Sept. 2014, pp. 265–272. https://doi.org/10.1007/978-3-319-10443-0_34.

[54] A. Schürmann, *Computational geometry of positive definite quadratic forms*, University Lecture Series, 49 (2009). http://dx.doi.org/10.1090/ulect/048.

[55] E. Selling, *Ueber die binären und ternären quadratischen Formen.*, Journal fur die Reine und Angewandte Mathematik, 77 (1874), pp. 143–229. https://eudml.org/doc/148235.

[56] J.A. Sethian, *A fast marching level set method for monotonically advancing fronts*, Proceedings of the National Academy of Sciences, 93 (1996), pp. 1591–1595. http://dx.doi.org/10.1073/pnas.93.4.1591.

[57] ——, *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science ... on Applied and Computational Mathematics)*, Cambridge University Press, Cambridge, U.K. ; New York, 2 edition ed., June 1999. ISBN 978-0-521-64557-7.

[58] J.A. Sethian and A. Vladimirsky, *Ordered upwind methods for static Hamilton–Jacobi equations*, PNAS, 98 (2001), pp. 11069–11074. https://doi.org/10.1073/pnas.201222998.

[59] P. Strandmark, J. Ulen, F. Kahl, and L. Grady, *Shortest Paths with Curvature and Torsion*, in IEEE International Conference on Computer Vision (ICCV), IEEE, 2013, pp. 2024–2031. https://doi.org/10.1109/ICCV.2013.253.

[60] J-D. Tournier, F. Calamante, and A. Connelly, *Robust determination of the fibre orientation distribution in diffusion MRI: Non-negativity constrained super-resolved spherical deconvolution*, NeuroImage, 35 (2007), pp. 1459–1472. https://doi.org/10.1016/j.neuroimage.2007.02.016.

[61] J. N. Tsitsiklis, *Efficient algorithms for globally optimal trajectories*, IEEE Transactions on Automatic Control, 40 (1995), pp. 1528–1538. https://doi.org/10.1109/9.412624.

[62] A. Vladimirsky, *Label-setting methods for Multimode Stochastic Shortest Path problems on graphs*, Mathematics of Operations Research, 33 (2008), pp. 821–838. http://dx.doi.org/10.1287/moor.1080.0321.

[63] A. Vladimirsky and C.i Zheng, *A fast implicit method for time-dependent Hamilton-Jacobi PDEs*, arXiv.org, (2013). http://arxiv.org/abs/1306.3506.

[64] C. Zach, L. Shan, and M. Niethammer, *Globally Optimal Finsler Active Contours.*, in DAGM-Symposium, Springer, 2009, pp. 552–561. http://dx.doi.org/10.1007/978-3-642-03798-6_56.

[65] H. Zhao, *A fast sweeping method for eikonal equations*, Mathematics of Computation, (2005). http://dx.doi.org/10.1090/S0025-5718-04-01678-3.