SIAM J. SCI. COMPUT.
Vol. 38, No. 4, pp. A2307–A2333

A FAST-MARCHING ALGORITHM FOR NONMONOTONICALLY EVOLVING FRONTS*

ALEXANDRA TCHENG† AND JEAN-CHRISTOPHE NAVE†

Abstract. The nonmonotonic propagation of fronts is considered. When the speed function $F : \mathbb{R}^n \times [0,T] \to \mathbb{R}$ is prescribed, the nonlinear advection equation $\phi_t + F|\nabla\phi| = 0$ is a Hamilton–Jacobi equation known as the level-set equation. It is argued that a small enough neighborhood of the zero-level-set $\mathcal{M}$ of the solution $\phi : \mathbb{R}^n \times [0,T] \to \mathbb{R}$ is the graph of $\psi : \mathbb{R}^n \to \mathbb{R}$, where ψ solves a Dirichlet problem of the form $H(\mathbf{u}, \psi(\mathbf{u}), \nabla\psi(\mathbf{u})) = 0$. A fast-marching algorithm is presented where each point is computed using a discretization of such a Dirichlet problem, with no restrictions on the sign of F. The output is a directed graph whose vertices evenly sample $\mathcal{M}$. The convergence, consistency, and stability of the scheme are addressed. Bounds on the computational complexity are estimated and experimentally shown to be on par with the fast-marching method. Examples are presented where the algorithm is shown to be globally first-order accurate. The complexities and accuracies observed are independent of the monotonicity of the evolution.

Key words. front propagation, Hamilton–Jacobi equations, viscosity solutions, fast-marching method, level-set method, finite-difference schemes

AMS subject classifications. 65M06, 65M22, 65H99, 65N06, 65N12, 65N22

DOI. 10.1137/15M1017302

1. Introduction.

Front propagation is a time-dependent phenomenon occurring when the boundary between two distinct regions of space is evolving. It is possible to make the distinction between *monotone* and *nonmonotone* motion of fronts. Consider a fire propagating through a forest: the interface divides space into a burnt and an unburnt region. It evolves monotonically in that if a point $\mathbf{x} = (x_1, \ldots, x_n)$ of space belongs to the burnt region, then it cannot belong to the unburnt region at a later time [21]. In contrast, if a plate containing water is gently rocked, the front separating the dry region from the wet one may advance or recede. In this paper, we present an algorithm that dynamically builds a sampling of the subset of $\mathbb{R}^n \times [0,T)$ consisting of the surface traced out by the front as it evolves through time. The structure of the scheme is akin to the fast-marching method (FMM), yet our approach is oblivious to the monotonicity of the evolution.

Given an initial front $\mathcal{C}_0$ as a codimension-one subset of $\mathbb{R}^n$ and an advection rule, our goal is to recover the front $\mathcal{C}_t$ at later times $t > 0$. Let each point of the front evolve with a prescribed speed $F : \mathbb{R}^n \times [0,T] \to \mathbb{R}$ in the direction of the outward normal to the front $\hat{\mathfrak{n}}_t : \mathcal{C}_t \to S^n$. The evolution is nonlinear, such that even if F and $\mathcal{C}_0$ are smooth the front may become C^0 and undergo topological changes [8].

On the one hand, a robust numerical method for tracking either kind of evolution is the level-set method [21, 22, 28]. This implicit approach embeds the front as the

*Submitted to the journal's Methods and Algorithms for Scientific Computing section April 17, 2015; accepted for publication (in revised form) May 17, 2016; published electronically August 2, 2016.

http://www.siam.org/journals/sisc/38-4/M101730.html

†Department of Mathematics and Statistics, McGill University, 805 Sherbrooke Street West, Montreal, Quebec H3A 0B9, Canada (alexandra.tcheng@mail.mcgill.ca, jcnave@math.mcgill.ca). The first author was supported by the Schulich Graduate Fellowship. The second author was supported by NSERC Discovery and Discovery Accelerator Supplement grants.

zero-level-set of a function $\phi : \mathbb{R}^n \times [0,T) \to \mathbb{R}$ and solves the initial-value problem

$$\left\{ \begin{array}{rcll} \phi_t + F|\nabla\phi| & = & 0 & \text{in } \mathbb{R}^n \times (0,T), \\ \phi(\mathbf{x},0) & = & \phi_0(\mathbf{x}) & \text{on } \mathbb{R}^n \times \{0\}, \end{array} \right. \tag{1.1}$$

where ϕ_0 satisfies $\{\mathbf{x} : \phi_0(\mathbf{x}) = 0\} = \mathcal{C}_0$. Assuming the computational grid comprises N^n points, the total complexity of the first-order algorithm solving (1.1) is $\mathcal{O}(N^{n+1})$ when endowed with the usual CFL condition $\Delta t \propto 1/N$. On the other hand, the FMM [26, 27, 28, 29, 33] may be used when $F = F(\mathbf{x}) \geq \delta > 0$ and is therefore suited for monotone propagation. This approach reduces the dimensionality of the problem by building the *first arrival time function* $\hat{\psi} : \mathbb{R}^n \to \mathbb{R}$, i.e., $t = \hat{\psi}(\mathbf{x})$ gives the unique time at which the front reaches $\mathbf{x}$. This function solves the boundary-value problem

$$\left\{ \begin{array}{rcll} |\nabla\hat{\psi}| & = & \frac{1}{F} & \text{in } \mathcal{B}_0, \\ \hat{\psi}(\mathbf{x}) & = & 0 & \text{on } \mathcal{C}_0, \end{array} \right. \tag{1.2}$$

where $\mathcal{B}_0$ is the unbounded component of $\mathbb{R}^n \setminus \mathcal{C}_0$ for $F > 0$. The FMM solves (1.2) in $\mathcal{O}(N^n \log N^n)$ time using a variant of Dijkstra's algorithm [9]. Schemes derived from the FMM include an extension to the case where F depends on time [34, 35] and the generalized FMM [4]. Problem (1.2) may also be solved using the fast sweeping method [32], which is a first-order accurate method running in $\mathcal{O}(N^n)$ time. When $F \equiv 1$, higher-order methods exist, such as the one of Cheng and Tsai [5], which exploits the relation between (1.2) and the time-dependent Eikonal equation.

The generalized FMM of [4] allows F to vanish and change sign albeit at the expense of the computational time. In [31], when $n = 2$ the authors propose a scheme that can handle such speed functions *while* featuring a computational complexity comparable to that of the FMM. This approach considers the set $\mathcal{M} := \{(\mathbf{x}, t) : \mathbf{x} \in \mathcal{C}_t\}$. Each point $p \in \mathcal{M}$ may be described as $p = (x, y, \hat{\psi}(x, y))$ and/or $p = (\tilde{\psi}(y, t), y, t)$ and/or $p = (x, \bar{\psi}(x, t), t)$. In regions where $|F| \geq \delta > 0$, the algorithm builds $\hat{\psi}$ using the FMM, whereas near points where $F = 0$, it solves a PDE satisfied by either $\tilde{\psi}$ or $\bar{\psi}$. Despite a number of satisfactory results, the mechanism to change representation is cumbersome, and $\mathcal{M}$ is undersampled near regions where F vanishes.

In the present paper, rather than limiting ourselves to the $n + 1$ representations just mentioned, we now describe $\mathcal{M}$ locally with functions of the form $\psi(\mathbf{u})$ where the points $\mathbf{u} = (u_1, \ldots, u_n)$ belong to *some hyperplane* lying in $\mathbb{R}^n \times [0, T]$. Consider the orthonormal spanning set for this hyperplane consisting of $\{\hat{u}_1, \ldots, \hat{u}_n\}$, and let the normal be $\hat{u}_{n+1}$. Thus in this convention, any point $(\mathbf{u}, u_{n+1})$ of space-time $\mathbb{R}^n \times [0, T]$ may be written as $(\mathbf{u}, u_{n+1}) = u_1\hat{u}_1 + \cdots + u_n\hat{u}_n + u_{n+1}\hat{u}_{n+1}$. Section 2 shows that $\psi : \mathbb{R}^n \to \mathbb{R}$ solves a Dirichlet problem of the form

$$\left\{ \begin{array}{rcll} H(\mathbf{u}, \psi(\mathbf{u}), \nabla\psi(\mathbf{u})) & = & 0 & \text{in } \mathcal{U} \subset \mathbb{R}^n, \\ \psi(\mathbf{u}(\mathbf{x}, t)) & = & u_{n+1}(\mathbf{x}, t) & \text{on } \mathbf{x} \in \mathcal{C}_t \cap \mathcal{V} \end{array} \right. \tag{1.3}$$

for appropriate neighborhoods $\mathcal{U}$ and $\mathcal{V}$. The *Hamiltonian* function $H : \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}$ depends on ψ through the speed function F and involves constants that capture the relative orientations of the two coordinate systems in use. Section 3 presents the discretization of (1.3) using finite-differences, as well as the design of a constrained minimization problem. Section 4 discusses the different parts of the algorithm for the case $n = 2$. The protocol we propose is similar to the FMM: Points sampling

$\mathcal{M}$ first belong to the *narrow band* $\mathcal{N}$ before moving to the *accepted* set $\mathcal{A}$. When $p_a \in \mathcal{N}$ is accepted, a local $\{\hat{u}_1, \ldots, \hat{u}_{n+1}\}$-coordinate system is found. Using p_a and another point in $\mathcal{A}$, a new point is computed through the finite-difference solvers and the optimization problem. Section 5 highlights the properties of the solvers and the global scheme. In particular, convergence of the local solvers is addressed and the total computational complexity is estimated. As illustrated in section 6 with numerous examples, the output of the algorithm is a *directed graph*. Its meshsize is bounded below by a predetermined parameter h, and its vertices provide a discrete sampling of $\mathcal{M}$. Given $t \in (0, T)$, the front $\mathcal{C}_t$ can be recovered using interpolation.

Methods with a similar flavor have been explored in [25], which presents a high-order fast interface tracking method, and [17], where the authors formulate n "quasi-linear PDEs satisfied by the manifold's local parametrization" and "solve that system locally in an Eulerian framework."

For clarity of illustration, the setting of our paper is $n = 2$, so that $\mathbf{x} = (x_1, x_2) = (x, y)$, $\mathbf{u} = (u_1, u_2) = (u, v)$, and $u_3 = w$. It is worth noting, however, that most of the underlying ideas, which are presented in sections 2 and 3, easily extend to the general higher dimensional case. Current obstacles to the full generalization of the method lie in the design of a practical interface tracker, able to capture the global features of the front at all times. The successful results of sections 4, 5, and 6 when $n = 2$ provide a proof of concept that such a goal should be pursued in future work.

Our approach offers numerous advantages when $n = 2$. It extends previous work: If the local and global systems coincide, then $\psi = \hat{\psi}$, whereas if $(\hat{u}_1, \hat{u}_2) = (\hat{y}, \hat{t})$ and $\hat{u}_3 = \hat{x}$, then $\psi = \tilde{\psi}$. By construction, the transition from one representation to another is smooth, and the resolution of the sampling is regular. The initialization is almost identical to the main procedure and does not require any information away from $\mathcal{C}_0$. The accuracy of the scheme is $\mathcal{O}(h)$. If $\mathcal{C}_0$ is sampled by m points, the computational complexity is bounded by CN, where $C = \max\{\mathcal{O}(m), \mathcal{O}\left(10^{n+2}\right)\}$ with $N = \mathcal{O}(m^n)$. When run on a monotone example, for the same computational time, our method yields a more accurate solution than the FMM.

2. Hyperplane representation.

In this section, we derive problem (1.3) from the initial-value problem (1.1) after making a few assumptions. We then argue that the solution of (1.3) locally describes $\mathcal{M}$ and that a finite number of such solutions provides a covering of $\mathcal{M}$.

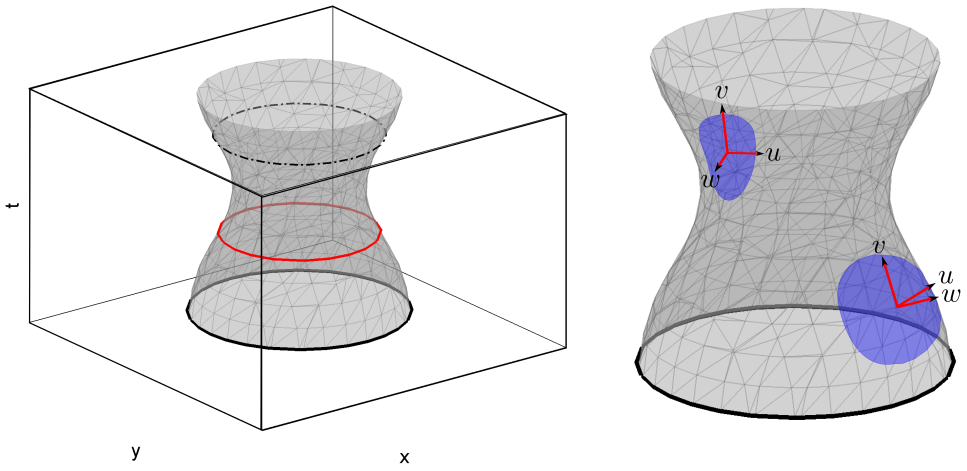
2.1. Assumptions.

The set $\mathcal{C}_0$ is a closed, co-dimension-one subset of $\Omega \subset \mathbb{R}^2$ without boundary, where Ω is bounded. Moreover, $\mathcal{C}_0$ is the graph of a C^1 function. The speed function $F : \mathbb{R}^2 \times [0, T] \to \mathbb{R}$ is continuous. It may vanish and change sign. Under those assumptions, the level-set equation (LSE) in (1.1) is a Hamilton–Jacobi equation with a unique continuous viscosity solution [6, 7, 8]. It follows that $\mathcal{M}$ embeds in $\mathbb{R}^2 \times [0, T]$ as a C^0 manifold. We denote as $\hat{\nu}(\mathbf{x}, t)$ the outward normal to $\mathcal{M}$ at $(\mathbf{x}, t)$.

2.2. PDE on an arbitrary plane.

Let a normal vector $\bar{\nu} \in S^3$ be given, and consider a corresponding plane lying in xyt-space. We denote as $\hat{u}\hat{v}\hat{w}$ the unique right-handed orthonormal coordinate system satisfying the following conditions: $\hat{w} = \bar{\nu}$, the $\hat{v}\hat{w}$-plane is vertical, and the $\hat{t}$-component of $\hat{v}$ is positive. See Figure 1. We have

$$\begin{pmatrix} \hat{u} \\ \hat{v} \\ \hat{w} \end{pmatrix} = \begin{pmatrix} \alpha_1 & \alpha_2 & \alpha_3 \\ \beta_1 & \beta_2 & \beta_3 \\ \gamma_1 & \gamma_2 & \gamma_3 \end{pmatrix} \begin{pmatrix} \hat{x} \\ \hat{y} \\ \hat{t} \end{pmatrix} := M \begin{pmatrix} \hat{x} \\ \hat{y} \\ \hat{t} \end{pmatrix}, \tag{2.1}$$



(a) $\mathcal{M} \subset \mathbb{R}^2 \times [0, T]$. The xyt-coordinate system is the global one.

(b) Two neighborhoods and their uvw-coordinate systems.

FIG. 1. *Illustration of the notation for* $n = 2$.

where the α's, β's, and γ's are constants. In particular, $\alpha_3 = 0$ and $\beta_3 > 0$ (see Appendix A for details). Let us now assume that a neighborhood $\mathcal{V}$ of $(\mathbf{x}_0, t_0) \in \mathcal{M}$ may be represented as the graph of the C^1 function:

$$\psi : \mathbb{R}^2 \to \mathbb{R}, \qquad \psi : \mathbf{u} \mapsto \psi(\mathbf{u}). \tag{2.2}$$

If $\bar{\nu} \cdot \hat{\nu}(\mathbf{x}_0, t_0) > 0$, then by the implicit function theorem $\phi(\mathbf{x}, t) = w - \psi(\mathbf{u})$, where ϕ is the solution of (1.1). Using implicit differentiation, we replace the derivatives appearing in (1.1) by

$$\phi_{x_i} = \gamma_i - \alpha_i \psi_u - \beta_i \psi_v, \quad i = 1, 2, \quad \text{and} \quad \phi_t = \gamma_3 - \alpha_3 \psi_u - \beta_3 \psi_v. \tag{2.3}$$

Using the orthonormality of the $\hat{u}\hat{v}\hat{w}$-coordinate system, (1.1) can be rewritten solely in terms of α_3, β_3, and γ_3 as

$$[\gamma - (\alpha\psi_u + \beta\psi_v)] + G\sqrt{1 + \psi_u^2 + \psi_v^2 - [\gamma - (\alpha\psi_u + \beta\psi_v)]^2} = 0, \tag{2.4}$$

where the subscript $\cdot_3$ was dropped, and $G(\mathbf{u}, \psi(\mathbf{u})) := F\left(\mathbf{x}(\mathbf{u}, \psi(\mathbf{u})), t(\mathbf{u}, \psi(\mathbf{u}))\right)$ was introduced. Equivalently, (2.4) can be rearranged using the vectors $\hat{R} := (\alpha, \beta, \gamma)$ and $\nu := (-\psi_u, -\psi_v, 1)$, as

$$\nu \cdot \hat{R} + G\sqrt{\nu \cdot \nu - \left(\nu \cdot \hat{R}\right)^2} = 0. \tag{2.5}$$

Note that this formulation is independent of dimension. In the event where $\bar{\nu} = \hat{\nu}(\mathbf{x}_0, t_0)$, the point $(\mathbf{x}_0, t_0)$ is a local maximum of $\mathcal{V}$, and we may explicitly relate the orientation coefficients to the speed function:

$$\hat{R} = \left(0 \ , \ \frac{1}{\sqrt{1 + F^2(\mathbf{x}_0, t_0)}} \ , \ \frac{-F(\mathbf{x}_0, t_0)}{\sqrt{1 + F^2(\mathbf{x}_0, t_0)}} \right). \tag{2.6}$$

We arrived at the following conclusion: Suppose ψ satisfies the boundary condition

$$\psi(\mathbf{u}(\mathbf{x}, t^0)) = w(\mathbf{x}, t^0) \qquad \text{when } \mathbf{x} \in \mathcal{C}_{t^0} \cap \mathcal{V} \tag{2.7}$$

and solves (2.5) on $\mathcal{U} \subset \mathbb{R}^2$. Then letting $t^* := t(\mathbf{u}, \psi(\mathbf{u}))$ for $\mathbf{u} \in \mathcal{U}$, we have that $\mathbf{x}^* := \mathbf{x}(\mathbf{u}, \psi(\mathbf{u})) \in \mathcal{C}_{t^*}$.

2.3. Well-posedness. Equation (2.4) is a Hamilton–Jacobi equation of the form

$$H(\mathbf{u}, \psi, \nabla\psi; \hat{R}) := -[\gamma - \beta\psi_v] - G(\mathbf{u}, \psi)\sqrt{\psi_u^2 + (\beta + \gamma\psi_v)^2} = 0, \tag{2.8}$$

where we defined the *Hamiltonian* $H : \mathbb{R}^2 \times \mathbb{R} \times \mathbb{R}^2 \to \mathbb{R}$. The well-posedness of this equation is guaranteed in the class of viscosity solutions [1]. The following theorem justifies defining H as the negative of the left-hand side of (2.4).

THEOREM 2.1. *The viscosity solutions of* (1.1) *and* (2.8) *are equivalent in the sense that the zero-level-set of* ϕ *at various times* $t = t(\mathbf{u}, w)$ *is precisely the set of points* $\mathbf{x} = \mathbf{x}(\mathbf{u}, w)$ *for which* $\psi(\mathbf{u}) = w$.

Note that this equivalence is investigated in the general case in Osher and Fedkiw's paper [22].

Proof. If $\mathcal{M}$ is C^1 at a point $\bar{p}$, then the manipulations of section 2.2 hold in the strong sense. Thus, let $\mathcal{M}$ be singular at the point $\bar{p}$, and consider the second-order PDE,

$$\phi_t^\epsilon + F|\nabla\phi^\epsilon| = \epsilon\Delta\phi^\epsilon, \qquad \epsilon > 0, \tag{2.9}$$

i.e., a viscous term was added to the right-hand side of the level-set equation. Define $\psi^\epsilon := w - \phi^\epsilon(u, v)$. Following the same reasoning as in section 2.2, using the orthonormal relations, as well as the fact that $\alpha_3 = 0$, (2.9) becomes

$$[\gamma - \beta\psi_v] + G\sqrt{\psi_u^2 + (\beta + \gamma\psi_v)^2} = -\epsilon\left[\psi_{uu}^\epsilon + \left(\beta_1^2 + \beta_2^2\right)\psi_{vv}^\epsilon\right]. \tag{2.10}$$

Taking ϵ to zero, the arguments detailed for example in section 1.5 of Lions' book [18] apply to yield that the Hamilton–Jacobi equation of interest is (2.8). ∎

In light of this result, (2.8) is preferred over (2.4) in the rest of this paper.

2.4. Geometric properties. We illustrate how $\mathcal{M}$ can be globally described by a covering of ψ-functions.

ψ provides a local representation of $\mathcal{M}$. Suppose that the solution of (1.3) is such that $\mathcal{V} := \mathcal{U} \times \psi(\mathcal{U})$ is C^0. Let $t_{\min} = \inf\{t : (\mathbf{x}, t) \in \mathcal{V}\}$ and $t_{\max} = \sup\{t : (\mathbf{x}, t) \in \mathcal{V}\}$.

THEOREM 2.2. *For any given* $t^* \in (t_{\min}, t_{\max})$, *we have*

$$\{\mathbf{x} \in \mathbb{R}^2 \,:\, \psi(\mathbf{u}(\mathbf{x}, t^*)) = w(\mathbf{x}, t^*)\} = \mathcal{C}_{t^*} \cap \mathcal{V}.$$

The proof follows the same argument as the proof of Theorem 4.2 in [31], and we therefore omit it.

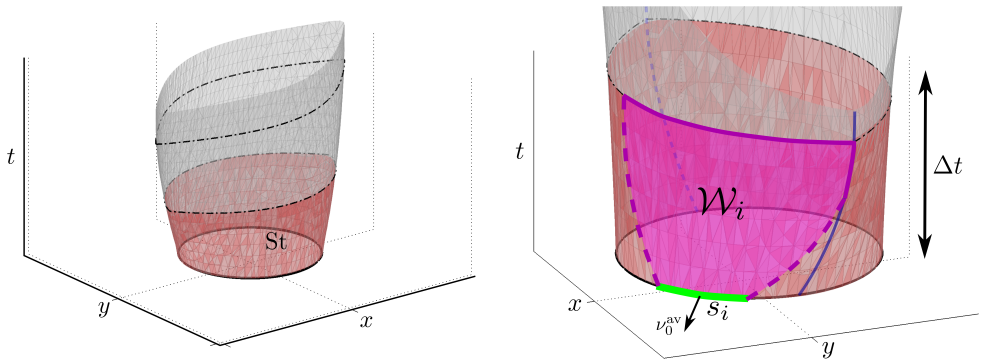


FIG. 2. *Illustration of the notation in Theorem* 2.3. $\mathcal{M}$ *is* C^0 *along the blue line.*

Covering of $\mathcal{M}$. It follows from compactness that $\mathcal{M}$ can be covered by a finite number of images of functions ψ, each solving a problem of the form (1.3). The construction presented below builds one such covering that is particularly suited for algorithmic purposes. Given $\Delta t > 0$ and $0 \le t_0 \le T - \Delta t$, consider the strip

$$\text{St} := \{(\mathbf{x}, t) \in \mathcal{M} \,:\, t_0 \le t \le t_0 + \Delta t\}. \tag{2.11}$$

THEOREM 2.3. *Suppose that* $\mathcal{C}_{t_0}$ *has codimension one in* $\mathbb{R}^n \times [0, T)$. *There exists a finite covering of* St *consisting of the graphs of functions* ψ_i, *where each* ψ_i *solves a Dirichlet problem of the form* (1.3) *with boundary conditions imposed on* $\mathcal{C}_{t_0}$.

Proof. Consider an open covering of $\mathcal{C}_{t_0}$ by open segments s with the following property: Defining

$$\nu_0^{\text{av}} := \frac{1}{|s|} \int \hat{\nu}\, ds, \quad \text{where} \quad |s| = \int ds, \tag{2.12}$$

for each $p \in s$ we have $\hat{\nu}(p) \cdot \nu_0^{\text{av}} > 0$. Since $\mathcal{C}_{t_0}$ is compact, we may extract a finite open covering, say, $\mathcal{S}_0 = \{s_i : i = 1, 2, \ldots, I\}$. Fix i, and define *the future domain of influence of* s_i as the subset of St satisfying

$$\begin{aligned}\mathcal{W}_i := \{(\mathbf{x}, t) \,:\, &\exists\, \{(\mathbf{x}_n, t_n)\}_{n=1}^{\infty} \to (\mathbf{x}, t) \text{ such that for each} \\ &(\mathbf{x}_n, t_n)\, \exists \text{ a characteristic of the LSE starting in } s_i \text{ and ending at } (\mathbf{x}_n, t_n)\}.\end{aligned}$$

(See, for example, [12] for details on the method of characteristics.) See Figure 2 for an illustration. We have $\text{St} = \cup_{i=1}^{I} \mathcal{W}_i$. Define

$$\nu_{\Delta t}^{\text{av}} := \frac{1}{|\mathcal{W}_i|} \int \hat{\nu}\, d\mathcal{W}_i, \quad \text{where} \quad |\mathcal{W}_i| = \int d\mathcal{W}_i. \tag{2.13}$$

Note that $\lim_{\Delta t \to 0} \nu_{\Delta t}^{\text{av}} = \nu_0^{\text{av}}$. Picking $\Delta t > 0$ small enough, it follows from the continuity of $\mathcal{M}$ that for each $p \in \mathcal{W}_i$, we have $\hat{\nu}(p) \cdot \nu_{\Delta t}^{\text{av}} > 0$. Consider the Dirichlet problem $H(\mathbf{u}, \psi, \nabla\psi; \hat{R}(\nu^{\text{av}})) = 0$ with boundary conditions imposed at $s_i \subset \mathcal{C}_{t_0}$. From Theorem 2.2, $\mathcal{W}_i$ is contained in the graph of ψ. ∎

A global covering may then be obtained from decomposing $\mathcal{M}$ into strips of the form (2.11) and extracting a finite subcover.

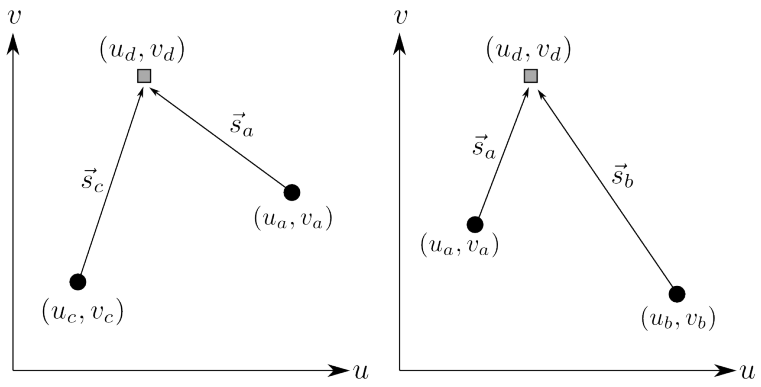


FIG. 3. *The points $p_a = (u_a, v_a, w_a)$, $p_b = (u_b, v_b, w_b)$, and $p_c = (u_c, v_c, w_c)$ belong to $\mathcal{A}$. Given $\mathbf{u}_d = (u_d, v_d)$, we wish to compute w_d.*

3. Discretization. Throughout this section, assume that the uvw-coordinate system is *fixed*. Suppose we are given two points belonging to $\mathcal{M}$ of the form $p_a = (\mathbf{u}_a, w_a)$ and $p_b = (\mathbf{u}_b, w_b)$ and wish to compute $w_d = \psi(\mathbf{u}_d)$ at a location $\mathbf{u}_d = (u_d, v_d)$. We will refer to p_a and p_b as the *parents* of the *child* point $p_d = (\mathbf{u}_d, w_d)$. In section 3.1, we first assume that $\mathbf{u}_d$ is *given* and propose two different solvers that take as input p_a, p_b, and $\mathbf{u}_d$ and return a value w_d. Then in section 3.2, we turn to the question of how $\mathbf{u}_d$ should be determined. The answer takes the form of a *constrained minimization problem*, for which we propose two different methods.

Section 3 thus provides a framework for solving the problem locally. Global issues also need to be addressed—see section 4.4.

3.1. Solving the PDE. As illustrated in Figure 3 define

$$\vec{s}_a := \mathbf{u}_d - \mathbf{u}_a, \qquad \vec{s}_b := \mathbf{u}_d - \mathbf{u}_b, \qquad \vec{s}_c := \mathbf{u}_d - \mathbf{u}_c \tag{3.1}$$

and let $\hat{s}_i = ((s_i)_u, (s_i)_v)$, where $i = a, b, c$, be the corresponding *unit* vectors. Then

$$\psi_i = \frac{w_d - w_i}{|\vec{s}_i|} \qquad \text{where } i = a, b, c, \tag{3.2}$$

provides first-order approximations of the directional derivatives of ψ at $\mathbf{u}_d$. Making use of the calculus identity $\psi_k = \nabla\psi \cdot \hat{k}$ for any unit vector $\hat{k}$, we have

$$\begin{pmatrix} \psi_u \\ \psi_v \end{pmatrix} = \begin{pmatrix} (s_a)_u & (s_a)_v \\ (s_i)_u & (s_i)_v \end{pmatrix}^{-1} \begin{pmatrix} \psi_a \\ \psi_i \end{pmatrix} =: B^{-1} \begin{pmatrix} \psi_a \\ \psi_i \end{pmatrix}, \text{ where } i = b, c, \tag{3.3}$$

provided that $\hat{s}_a$ and $\hat{s}_i$ are not colinear. We write

$$\tilde{\nu} = -\vec{M}\, w_d - \vec{N} \quad \text{with} \quad \vec{M} := (m_u, m_v, 0) \quad \text{and} \quad \vec{N} := (n_u, n_v, -1), \tag{3.4}$$

where the constants m_u, m_v, n_u, and n_v depend on $(\mathbf{u}_a, w_a)$, $(\mathbf{u}_i, w_i)$, where $i = b$ or c, and $\mathbf{u}_d$. The quantity $\tilde{\nu}$ provides a first-order approximation of ν.

3.1.1. Direct solver. Consider approximating G by setting it equal to $G_0 = G(p_a)$. Note that this is independent of w_d, which implies that the relation

$$\tilde{\nu} \cdot \hat{R} + G_0 \sqrt{\tilde{\nu} \cdot \tilde{\nu} - \left(\tilde{\nu} \cdot \hat{R}\right)^2} = 0 \tag{3.5}$$

can be rearranged as a *quadratic* in w_d. Indeed, first rewriting (3.5) as

$$\left(\tilde{\nu}\cdot\hat{R}\right)^2\left(1+G_0^2\right)=G_0^2\left(\tilde{\nu}\cdot\tilde{\nu}\right) \tag{3.6}$$

and then making use of (3.4), we arrive at

$$\left(\tilde{\nu}\cdot\hat{R}\right)^2=k_1w_d^2+2k_2w_d+k_3 \qquad \text{and} \qquad \tilde{\nu}\cdot\tilde{\nu}=k_4w_d^2+2k_5w_d+k_6, \tag{3.7}$$

where

$$\begin{array}{lll} k_1=\left(\hat{R}\cdot\vec{M}\right)^2, & k_2=\left(\hat{R}\cdot\vec{M}\right)\left(\hat{R}\cdot\vec{N}\right), & k_3=\left(\hat{R}\cdot\vec{N}\right)^2, \\ k_4=\vec{M}\cdot\vec{M}, & k_5=\vec{M}\cdot\vec{N}, & k_6=\vec{N}\cdot\vec{N}. \end{array} \tag{3.8}$$

Rearranging further yields

$$\left(k_1+G_0^2(k_1-k_4)\right)w_d^2+2\left(k_2+G_0^2(k_2-k_5)\right)w_d+\left(k_3+G_0^2(k_3-k_6)\right)=0$$

. The discriminant of this quadratic reads

$$0+\rho_1G_0^2+\rho_2G_0^4=G_0^2\left(\rho_1+\rho_2G_0^2\right), \tag{3.9}$$

where $\rho_1=-2k_2k_5+k_1k_6+k_3k_4$ and $\rho_2=\rho_1+k_5^2-k_4k_6$. We set

$$w_d=\frac{-\left(k_2+G_0^2(k_2-k_5)\right)+G_0\sqrt{\rho_1+\rho_2G_0^2}}{\left(k_1+G_0^2(k_1-k_4)\right)}. \tag{3.10}$$

Since $\text{sign}(\gamma)=-\text{sign}(G_0)$, this choice of root minimizes $t_d=\alpha u_d+\beta v_d+\gamma w_d$, where $\alpha=0$ and $\beta>0$. This is consistent with the control theoretical approach to this problem [11, 13, 33, 34]. If $0=k_1+G_0^2(k_1-k_4)$, the quadratic relation degenerates to a linear one with solution:

$$w_d=-\frac{k_3+G_0^2(k_3-k_6)}{2\left(k_2+G_0^2(k_2-k_5)\right)}. \tag{3.11}$$

The outward normal to $\mathcal{M}$ at p_d can then be computed as $\hat{\nu}_d=\tilde{\nu}/|\tilde{\nu}|$.

Remark. As mentioned in section 2.3, when $\hat{R}=(0,0,\pm1)$, the PDE reduces to the Eikonal equation. If working on a Cartesian grid, formulas (3.10) and (3.11) can be verified to agree with the solvers used in the traditional FMM.

3.1.2. Iterative solver. An alternative to approximating the speed term G as in section 3.1.1 is to solve (2.5) iteratively in pseudotime τ, i.e., Let

$$\tilde{\nu}^n:=-\vec{M}\,w_d^n-\vec{N} \qquad \text{and} \qquad G^n:=F(\,\mathbf{x}(\mathbf{u}_d,w_d^n),t(\mathbf{u}_d,w_d^n)\,) \tag{3.12}$$

and iterate

$$\tilde{\nu}^n\cdot\hat{R}+G^n\sqrt{\tilde{\nu}^n\cdot\tilde{\nu}^n-\left(\tilde{\nu}^n\cdot\hat{R}\right)^2}=\frac{w_d^{n+1}-w_d^n}{\Delta\tau} \tag{3.13}$$

until $|w_d^{n+1}-w_d^n|$ is below some predetermined tolerance. The direct solver can be used to initialize w_d^0. The size of the pseudotime step is determined based on Definition 5 of [19]—see Appendix B for details.

3.2. Optimal sampling. Suppose that a set $\mathcal{A}$ of points sampling a part of $\mathcal{M}$ is available. Given p_a, $p_i \in \mathcal{A}$, we now consider *where* to compute a new point p_d. Constraints are heuristically imposed so as to efficiently build a regular sampling of $\mathcal{M}$. Rigorous justifications of these choices are provided in section 5, where the properties of the scheme are analyzed.

3.2.1. Constrained minimization problem.

Accuracy. According to (3.2), the accuracy of either solver increases as $|\vec{s}_a|$ and $|\vec{s}_i|$ become smaller. We combine those requirements into a single function to be minimized:

$$f(\mathbf{u}_d) := |\vec{s}_a|^2 + |\vec{s}_i|^2.$$

The result is that the child point p_d should lie close to its parents. However, we now argue that it should not be *too* close.

Evenness of the sampling. Repulsion between p_d and its parents is introduced to avoid oversampling parts of $\mathcal{M}$. The minimum three-dimensional (or $(n+1)$-dimensional) distance between p_d and any other point in $\mathcal{A}$ should be at least h, where h is a predetermined parameter. As will be clear in section 6, h bounds the meshsize of the resulting graph from below.

Characteristic structure. Last, we make sure that p_d does not violate the characteristic structure of the solution: w_d must be real, and in addition p_d has to satisfy $t_d \geq \max\{t_a, t_i\}$ for causality to hold. We impose the more stringent condition:

$$t_d \geq \max\{t_a, t_i\} + \Delta t, \quad \text{where} \quad \Delta t = \frac{h}{\sqrt{G^2(\mathbf{u}_j) + 1}},$$

with $j = a$ if $\max\{t_a, t_i\} = t_a$ and $j = i$ otherwise. This choice of Δt speeds up computations and simplifies our analysis, as will be evident in section 5.1.1, where $\Delta t = \beta h$.

Optimal sampling—summary. Given a pair of points p_a, $p_i \in \mathcal{A}$, we wish to minimize the objective function

$$f(\mathbf{u}_d) = |\vec{s}_a|^2 + |\vec{s}_i|^2 \tag{A}$$

subject to the constraints

$$g_1(\mathbf{u}_d) = \Im(w_d) = 0, \tag{V1}$$

$$g_2(\mathbf{u}_d) = t_d - \max\{t_a, t_i\} \geq \Delta t, \tag{V2}$$

$$g_3(\mathbf{u}_d) = \min_{p \in \mathcal{A}}\{\|p_d - p\|\} \geq h. \tag{E}$$

3.2.2. Solving the constrained minimization problem. Although the objective function is simple, the constraints are nonlinear functions of $\mathbf{u}_d$. Unlike the other two, constraint (V1) does not require evaluating w_d. If using the direct solver, it amounts to checking the sign of the discriminant. Constraint (V2) requires conversion of the data to $\mathbf{x}t$-coordinates. Constraint (E) can be very costly to verify if $\mathcal{A}$ is large. This is why it is preferable to work with a predetermined small subset of $\mathcal{A}$. To be efficient, a scheme solving this problem should not require too many evaluations of the constraints. We briefly discuss two possible methods and relegate the details of the procedure to Appendix C.

The grid method. This simple yet efficient approach is the one we use in practice. The problem is solved using grids sampling a neighborhood of p_a and p_i in the uv-plane. The objective function f is computed at those points where all three constraints are satisfied. The location of the minimum of f is found, and a finer grid centered at this point is defined. The procedure is repeated until convergence, e.g., the change in the value of the minimum is below some predefined tolerance.

The Lagrangian method. Following [3] (sections 2.1, 2.2, and 3.1), the augmented Lagrangian L is defined by adding the three constraints to f along with Lagrange multipliers μ_i and penalty coefficients $c_i > 0$, where $i = 1, 2, 3$. The following procedure is iterated until convergence, e.g., the change in the value of the minimum is below some predefined tolerance. First, for some values of the c_i and μ_i, the unconstrained minimum of L is found. This can be done using BFGS along with line minimization [24]. Then the c_i's are increased and the μ_i's are updated. Although convergence is guaranteed, this method requires a large number of evaluations of the constraints, which makes it too slow for our purposes.

4. Algorithms.

We present the algorithms used to generate the results in section 6.

4.1. Initialization.

Little information is required to initialize the algorithm. The input is a sampling of $\mathcal{C}_0$ consisting of m points and $m-1$ undirected segments. The normal $\hat{\nu}$ at each point and a final time must also be provided.

Store those pieces of information into a list, say, $\mathcal{L}$. In the following, we will assume that each row of $\mathcal{L}$ contains data pertaining to one point, e.g., if $p = (\mathbf{x}, t) = (x_1, x_2, \ldots, t)$ has normal $\hat{\nu}(\mathbf{x}, t)$, then $\mathcal{L}$ might initially look like

	x_1	x_2	$\ldots$	t	$\hat{\nu}_1(\mathbf{x},t)$	$\hat{\nu}_2(\mathbf{x},t)$	$\ldots$	$\hat{\nu}_{n+1}(\mathbf{x},t)$
row 1	$-.1625$	0.0080	$\ldots$	0.0342×10^{-15}	$-.1634$	0.301	$\ldots$	-0.5668
row 2	$-.1705$	0.0099	$\ldots$	0.0398×10^{-15}	$-.1706$	0.385	$\ldots$	-0.5909
				$\vdots$				

The variables $M_{\mathcal{N}}$ and $M_{\mathcal{A}}$ help monitor the number of points in $\mathcal{N}$ and $\mathcal{A}$, respectively. See Algorithm 1.

Algorithm 1. Initialization.

1: $\mathcal{A} \leftarrow \mathcal{L}$, $M_{\mathcal{A}} \leftarrow m + 1$
2: $\mathcal{N} \leftarrow \mathcal{L}$, $M_{\mathcal{N}} \leftarrow m + 1$
3: $h \leftarrow (\min_{p,\, q \in \mathcal{L}} \|p - q\|)\,/2$.

4.2. Get a local representation.

This routine is called when a point p_a is accepted to go from a global to a local representation. See Algorithm 2.

Algorithm 2. Get a local representation.

1: Find the L closest neighbors of p_a among $\tilde{\mathcal{A}}$ and form the list $\mathcal{S}$. Set $\tilde{\mathcal{A}} = \mathcal{S}$.
2: Find $\bar{\nu}$ such that $\bar{\nu} \cdot \hat{\nu}(p_i) > 0$ for each $p_i \in \mathcal{S}$. The default value is $\bar{\nu} = \hat{\nu}_a$.
3: Compute the uvw-coordinates of the points in $\mathcal{S}$ using relation (2.1).
4: Find the point $p_b \in \mathcal{S}$ lying closest to p_a with $u_b - u_a > h/2$.
5: **Return:** $\tilde{\mathcal{A}}$, the change of coordinates matrix M, and p_b.

4.3. Computing a new point. Once the minimum $\mathbf{u}_d$ is found using the direct solver, the iterative solver is run to improve the accuracy of the solution w_d. See Algorithm 3.

Algorithm 3. Compute a new point.

1: $\Delta t = h/\sqrt{G^2(\mathbf{u}_a)+1}$, $p_d = \mathbf{0}$, $\hat{\nu}_d = \mathbf{0}$.
2: Solve the constrained optimization problem defined by p_a, p_i, $G(\mathbf{u}_a)$, h, and Δt to get $\mathbf{u}_d$. See section 3.2 and Appendix C for details.
3: **if** a solution to the optimization problem was found **then**
4: Compute w_d associated with $\mathbf{u}_d$ using the iterative solver detailed in section 3.1.2.
5: Compute the normal $\hat{\nu}_d$ at $(\mathbf{u}_d, w_d)$.
6: Get p_d and $\hat{\nu}_d$ in $(\mathbf{x}, t)$-coordinates using the change of coordinates matrix M.
7: **end if**
8: **Return:** $[p_d; \hat{\nu}_d]$.

4.4. Global constraints. The global features of the manifold are monitored with the help of the structure *Book*, which consists of a list of segments. For example, if $\mathcal{N}$ has the form

$$\mathcal{N} = [p_1; p_2; \ldots; p_{M_\mathcal{N}}]^T. \quad \text{then} \quad Book = [p_2 - p_5;\ p_{11} - p_{23};\ \ldots;\ p_{14} - p_8]^T. \tag{4.1}$$

Segments are not directed, i.e., $[p_1 - p_2]$ is the same as $[p_2 - p_1]$. Initially, the collection of segments lies in the xy-plane and provides a piecewise linear approximation of $\mathcal{C}_0$. We impose the following constraints on *Book*.

- *Multiplicity:* Each point $p \in \mathcal{N}$ appears exactly twice in *Book*.
- *Loops:* A segment cannot start and end at the same point. A segment can appear only once in *Book*.
- *Intersections:* Segments cannot intersect.
- *Spikes:* The acute angle formed by two segments sharing an endpoint cannot be less than $\approx 0.2\pi$.

The situations are illustrated with simple examples in Figures 4–7. The framed subfigures are obtained after connecting *hanging nodes*, which are nodes that appear only once in *Book*. This is done as follows.

Stitching hanging nodes. Given a hanging node $p_1 = (\mathbf{x}_1, t_1)$, let $p_2 = (\mathbf{x}_2, t_2)$ be the hanging node that minimizes the xy-distance $\|\mathbf{x}_1 - \mathbf{x}_2\|$. Then add the segment $[p_1 - p_2]$ to *Book*. Repeat until there are no hanging nodes left.

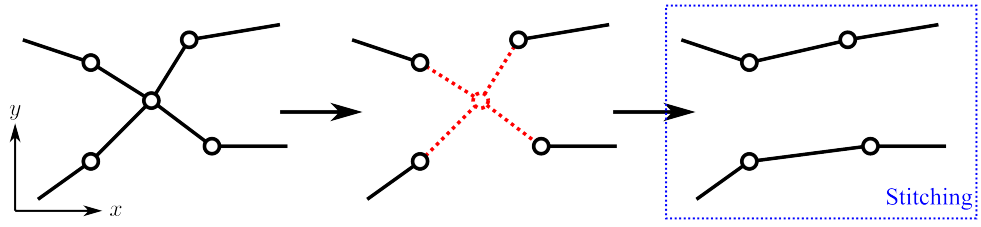


FIG. 4. *Multiplicity.*

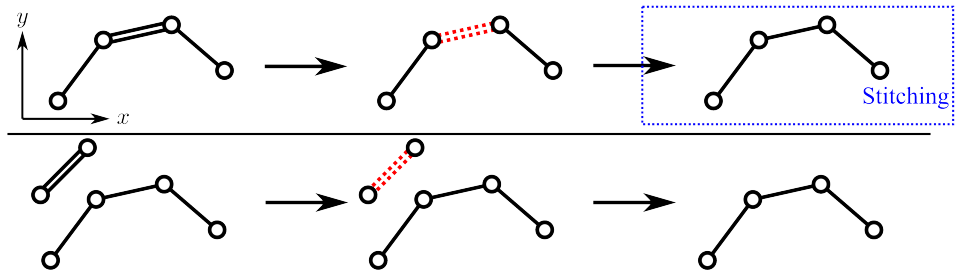


FIG. 5. *Loops. Once the double-loop is removed, the nodes are either stitched (top situation) or removed from $\mathcal{N}$ (bottom situation).*

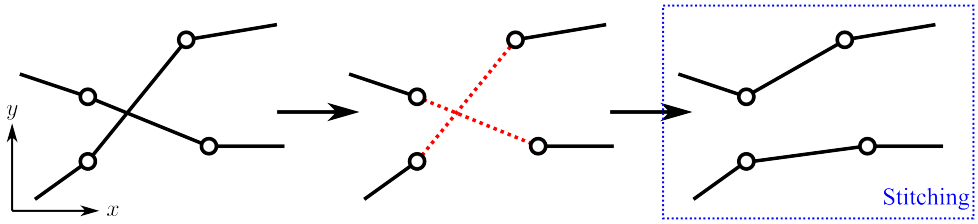


FIG. 6. *Intersections. The final configuration must differ from the original one.*

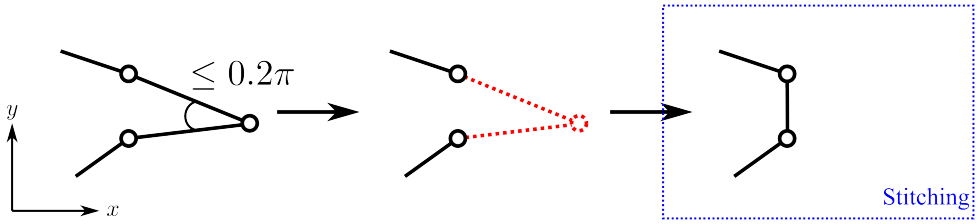


FIG. 7. *Spikes.*

Those global constraints are imposed to $\mathcal{N}$ in Algorithm 4. Note that we omit the details of the updating procedure of *Book*, which consists in relabeling nodes and possibly deleting some segments. See [30] for details.

Remark. When a new point is computed by Algorithm 3, the checks performed at lines 7 and 8 need only involve the segments attached to that new point, which requires an $\mathcal{O}(1)$ number of operations.

4.5. Main loop. A couple of remarks on Algorithm 5 are in order. Lines 5–7 are ignored during the first m iterations of the main loop: This can be considered as the end of the initialization procedure. The fact that the algorithm does not compute a new value when $t_a >$ final time implies that $\mathcal{N}$ is eventually empty, at which point the algorithm naturally ends. If Algorithm 3 fails, next-to-nearest neighbors are used (line 14). Since $|\tilde{\mathcal{A}}| \leq L$, this occurs at most L times. Here, we present the algorithms with $i = b$. The procedure is unchanged if $i = c$.

Algorithm 4. Book keeping of $\mathcal{N}$.

if $M_{\mathcal{N}} > 1$ **then**
  if a new point was computed by Algorithm 3 **then**
    Update *Book*.
  else
    Update and clean *Book*.
  end if
  Check intersections, and clean *Book* if a point was removed from $\mathcal{N}$.
  Check spikes, and clean *Book* if a point was removed from $\mathcal{N}$.
end if

---

procedure CLEAN *Book*.
  do
    Multi = 0; Loop = 0; Inter = 0;
    Check multiplicity, and set Multi to 1 if a point was removed from $\mathcal{N}$.
    Check loops, and set Loop to 1 if a point was removed from $\mathcal{N}$.
    Check intersections, and set Inter to 1 if a point was removed from $\mathcal{N}$.
  while Multi+Loop+Inter > 0
end procedure

Algorithm 5. Main loop.

counter = 1
while $\mathcal{N}$ is not empty **do**
  Find $p \in \mathcal{N}$ with the smallest t value, call it $p_a = (\mathbf{x}_a, t_a)$.
  Remove $[p_a; \hat{\nu}_a]$ from $\mathcal{N}$. $M_{\mathcal{N}} \leftarrow M_{\mathcal{N}} - 1$
  if counter$> m$ **then**
    Add $[p_a; \hat{\nu}_a]$ to $\mathcal{A}$ in the $M_{\mathcal{A}}$-th row. $M_{\mathcal{A}} \leftarrow M_{\mathcal{A}} + 1$
  end if
  if $t_a <$ Final time **then**
    Set $\tilde{\mathcal{A}} = \mathcal{A}$ and get a local representation using Algorithm 2.
    Run Algorithm 3 with the pair p_a - p_b to get p_d, $\hat{\nu}_d$.
    if Algorithm 3 was successful at finding a new point **then**
      Add $[p_d; \hat{\nu}_d]$ to $\mathcal{N}$ in the $M_{\mathcal{N}}$-th row. $M_{\mathcal{N}} \leftarrow M_{\mathcal{N}} + 1$
    else
      Remove $[p_b; \hat{\nu}_b]$ from $\tilde{\mathcal{A}}$, find a new $p_b \in \tilde{\mathcal{A}}$, and go back to line 10.
    end if
  end if
  Run Algorithm 4 to impose the global constraints on $\mathcal{N}$.
  counter $\leftarrow$ counter+1
end while

5. Properties of the scheme. We first discuss local properties of the scheme by approximating the solution of the optimization problem and asserting the convergence of the solvers. We then turn to global properties.

5.1. Local properties. Let us assume that p_a and $p_i \in \mathcal{A}$ are exact, i.e., p_a, $p_i \in \mathcal{M}$ and that $\|p_a - p_i\| = \mathcal{O}(h)$. Moreover, consider that we are working on a neighborhood where ψ is C^1, so that $\bar{\nu} = \hat{\nu}_a$ and $w_i = w_a + o(h^2)$. We investigate the

properties of the algorithms used to compute p_d. We show that the constraints of the optimization problem are such that the discretizations of (2.5) are monotone.

5.1.1. Optimization. The objective function can be rewritten as

$$f(\mathbf{u}_d) = 2\left(\|\mathbf{u}_d - \mathbf{u}_{\min}\|^2 - \mathbf{u}_a \cdot \mathbf{u}_i^T\right), \quad \text{where} \quad \mathbf{u}_{\min} = \left(\frac{u_a + u_i}{2}, \frac{v_a + v_i}{2}\right), \tag{5.1}$$

i.e., f is a paraboloid which reaches its minimum at $\mathbf{u}_{\min}$. It follows from the convexity of f that the solution of our optimization problem either is $\mathbf{u}_{\min}$ or lies on the boundary of the feasibility set. In the limit where $h \to 0$, we expect $w_{\min} \approx (w_a + w_i)/2$. Recalling the relation $t = \beta v + \gamma w$, if γ is small we may have $t_{\min} \leq \max\{t_a, t_b\}$, which violates constraint (V2). As a result, in general $\mathbf{u}_{\min}$ does not belong to the feasibility set.

Turning to constraint (V1) requires the study of the sign of $\rho_1 + \rho_2 G_0^2$. It can be shown that

$$\rho_1 \geq \|M\|^2 \|N\|^2 \epsilon \qquad \text{for some } \epsilon = \epsilon(\hat{R}, \vec{M}, \vec{N}) > 0. \tag{5.2}$$

Picking $\delta > 0$ such that $\epsilon \geq \delta^2/(1+\delta^2)$ gives $\rho_1 + \rho_2 G_0^2 \geq 0$ when $|G_0| < \delta$. When $|G_0| \geq \delta > 0$, using the approximation $w_i = w_a$ yields the following estimate:

$$\rho_1 + G_0^2 \rho_2 \;\approx\; \left(\gamma^2 - \beta^2 G_0^2\right) m_u^2 + m_v^2 \;=\; m_v^2 \geq 0. \tag{5.3}$$

Next, we momentarily focus on the solution to the following subproblem: Minimize (A) subject only to constraint (V2). The constraint is equivalent to

$$\beta\,(v_d - v_a) + \gamma\,(w_d - w_a) \geq \beta h \implies (v_d - v_a) + o(h^2) \geq h \tag{5.4}$$

so that it primarily constrains v_d. Thus, as $h \to 0$, the solution tends to

$$u_d = u_{\min}, \qquad v_d = \max\{h + v_a, v_{\min}\}. \tag{5.5}$$

The interplay between (V2) and (E) must be addressed. If $p_j \in \mathcal{A}$ since $t_a \geq t_j$,

$$\beta\,(v_d - v_j) + \gamma\,(w_d - w_j) \geq \beta h \implies (v_d - v_j) + \mathcal{O}(h^2) \geq h \tag{5.6}$$

and so $\|p_d - p_i\| \geq h$ as $h \longrightarrow 0$.

In conclusion, under the assumption that ψ is locally C^1, in the limit where $h \longrightarrow 0$, we have that (V1) holds, and satisfying constraint (V2) implies satisfying constraint (E). It follows that the solution of the optimization problem is given by the minimum of (5.1) subject to the constraint (5.4). Note that in the event where the assumption on the regularity of ψ is dropped, neither existence nor uniqueness can easily be asserted.

5.1.2. Direct solver. Given a location $\mathbf{u}_d$, the direct solver returns a value w_d^{dir} using either (3.10) or (3.11). We first show that this discretization is degenerate elliptic in the sense of [19], i.e., letting

$$\bar{H}(\psi_i, \psi_a) := -\left[\gamma - \beta\psi_v(\psi_a, \psi_i)\right] - G_0\sqrt{\left(\beta + \gamma\psi_v(\psi_a, \psi_i)\right)^2 + \psi_u^2(\psi_a, \psi_i)} \tag{5.7}$$

we show that $\bar{H}$ is nondecreasing in each variable. Recall relation (3.3):

$$\psi_u(\psi_a, \psi_i) = \frac{1}{\det B}\left((s_i)_v\psi_a - (s_a)_v\psi_i\right), \quad \psi_v(\psi_a, \psi_i) = \frac{1}{\det B}\left(-(s_i)_u\psi_a + (s_a)_u\psi_i\right).$$

From the previous subsection, we expect $|\vec{s}_a|$ and $|\vec{s}_i|$ to be $\mathcal{O}(h)$, as well as

$$-\frac{(s_i)_u}{\det B} = c_1 \geq 0 \quad \text{and} \quad \frac{(s_a)_u}{\det B} = c_2 \geq 0. \tag{5.8}$$

Moreover, given our assumption that $\bar{\nu} = \hat{\nu}(p_a)$, we have that $\nabla\psi(\mathbf{u}_a) = \vec{0}$, and

$$\nabla\psi(\mathbf{u}_d) = \nabla\psi(\mathbf{u}_a) + \vec{\mathfrak{h}} = \vec{\mathfrak{h}}, \tag{5.9}$$

where $\vec{\mathfrak{h}}$ is such that $|\vec{\mathfrak{h}}| = \mathcal{O}(h)$. We have

$$\frac{\partial \bar{H}}{\partial \psi_a} = \beta\frac{\partial \psi_v}{\partial \psi_a} - G_0\frac{(\beta+\gamma\psi_v)\gamma\frac{\partial\psi_v}{\partial\psi_a} + \psi_u\frac{\partial\psi_u}{\partial\psi_a}}{\sqrt{(\beta+\gamma\psi_v)^2+\psi_u^2}} \tag{5.10}$$

$$= \beta\frac{\partial \psi_v}{\partial \psi_a} - G_0\frac{\gamma\frac{\partial\psi_v}{\partial\psi_a} + \left(\gamma^2\psi_v\frac{\partial\psi_v}{\partial\psi_a} + \psi_u\frac{\partial\psi_u}{\partial\psi_a}\right)/\beta}{\sqrt{1+\left(2\beta\gamma\psi_v+\gamma^2\psi_v^2+\psi_u^2\right)/\beta^2}} \tag{5.11}$$

$$= \beta c_1 - G_0\frac{\gamma c_1 + \mathcal{O}(h)}{\sqrt{1+\mathcal{O}(h)}} \tag{5.12}$$

$$= (\beta - G_0\gamma)\, c_1 + \mathcal{O}(h) \tag{5.13}$$

$$= \sqrt{1+G_0^2}\, c_1 + \mathcal{O}(h) \;\geq\; 0 \qquad \text{for } h \text{ small enough.} \tag{5.14}$$

The argument that $\partial\bar{H}/\partial\psi_i \geq 0$ is completely analogous. We then show that the solver is stable in the sense of Barles and Souganidis [2]. Consider again the quadratic

$$\left((G_0^2+1)k_1 - G_0^2k_4\right)w_d^2 + 2\left((G_0^2+1)k_2 - G_0^2k_5\right)w_d + \left((G_0^2+1)k_3 - G_0^2k_6\right) = 0.$$

Using (2.6), (3.4), (3.8) and letting $|\vec{s}_i| = c_i h$, where c_i is $\mathcal{O}(1)$, we may rewrite it as

$$aw_d^2 + 2(b_1+hb_2)w_d + (c_1+hc_2) = 0, \quad \text{where } a, b_1, b_2, c_1, c_2 \text{ are } \mathcal{O}(1). \tag{5.15}$$

Assuming that $h < 1$, it is easily shown that the solution can be bounded by a bound independent of h. Verifying the consistency of the scheme reveals that it is first-order accurate with respect to h. Locally uniform convergence follows from [2].

5.1.3. Iterative solver. Given a location $\mathbf{u}_d$, the iterative solver returns a value w_d^{iter} using (3.13). The initial guess is provided by the direct solver. We let

$$\tilde{H}(w_d^n, \psi_a^n, \psi_i^n) := -\left[\gamma - \beta\psi_v(\psi_a^n, \psi_i^n)\right] - G^n\sqrt{\left(\beta+\gamma\psi_v(\psi_a^n,\psi_i^n)\right)^2 + \psi_u^2(\psi_a^n,\psi_i^n)},$$

where $\psi_i^n = (w_d^n - w_i)/|\vec{s}_i|$ for $i = a, b, c$, and $G^n := F(\,\mathbf{x}(\mathbf{u}_d, w_d^n), t(\mathbf{u}_d, w_d^n))$. Without loss of generality, we assume that the scheme is proper in the sense of [19], considering $\tilde{H} + \tilde{\epsilon}w^n$ if necessary. From the previous section, it follows that $\tilde{H}$ is a degenerate elliptic scheme. Theorem 8 of [19] guarantees convergence to the unique solution for arbitrary initial data.

5.2. Global properties.

5.2.1. The sets $\mathcal{A}$ and $\mathcal{N}$. We demonstrate some properties of these sets.

LEMMA 5.1. *The size of $\mathcal{N}$ cannot exceed m, the number of points sampling $\mathcal{C}_0$.*

Proof. Initially, the size of $\mathcal{N}$ is m. Each iteration of the main loop successively removes exactly one point from $\mathcal{N}$ (line 4); adds at most one point to $\mathcal{N}$ (line 12); may remove points from $\mathcal{N}$ (Algorithm 4). ∎

The following lemma justifies promoting the point in $\mathcal{N}$ with the smallest time value to the set $\mathcal{A}$.

LEMMA 5.2. *Subsequent iterations of the Main Loop will not affect the value of the point in $\mathcal{N}$ with the smallest time value.*

Proof. Suppose that during the k-th iteration of the main loop, $p_a \in \mathcal{N}$ is promoted to the set $\mathcal{A}$, and let us denote by p_* the point in $\mathcal{N}$ with the smallest time value. It follows from constraint (V2) that if p_a has a child point, its time value satisfies $t_d \geq t_a + \Delta t > t_a \geq t_*$. Therefore p_d cannot belong to the past domain of influence of p_*. ∎

Remark. It is not possible to closely mimic the proofs of Sethian [26] or Tsitsiklis [33]. Indeed, those proofs assume the existence of a graph where the values of ψ (resp., V) are to be found. In constrast, in our framework, the choice of which point gets promoted to the set $\mathcal{A}$ directly influences the shape of the graph sampling $\mathcal{M}$.

5.2.2. Global convergence. When discussing the global convergence of the scheme, we distinguish between different cases based on the regularity and topological properties of $\mathcal{M}$.

- *$\mathcal{M}$ is C^1.* Then the local solvers may be used anywhere on $\mathcal{M}$, since the correspondence between the LSE and the Dirichlet problem highlighted in section 2.2 holds in the strong sense. As $h \to 0$, the assumptions listed in the beginning of section 5.1 hold and imply that the local solvers are convergent. Considering the relation $\Delta t = h/\sqrt{G^2(\mathbf{u}_a) + 1}$ as a CFL condition ensures the stability of the global scheme. Global convergence is thus expected.
- *$\mathcal{M}$ is C^0 and the singularities arise from topological changes.* Our argument is that the global constraints imposed on $\mathcal{N}$ are such that the local solvers are used only in regions of $\mathcal{M}$ that are smooth. Indeed, topological changes imply that the segments monitored by *Book* intersect. Such a situation should be prevented by Algorithm 4. It is therefore expected that the scheme is convergent in those situations as well—although no proof is available at this point.
- *$\mathcal{M}$ is C^0.* Should the speed F be merely continuous, $\mathcal{M}$ may develop singularities without undergoing topological changes. In those situations, it is unclear whether the scheme globally converges. Let us note that the iterative scheme in section 5.1.3 should converge regardless of the regularity of the Hamiltonian. Nonetheless, the size of the pseudotime step $\Delta\tau$ required for convergence may be hard to determine. For that reason, the examples considered in the next section only involve speeds that are C^1 everywhere along the curve.

5.2.3. Computational time. Recall that, in what follows, $n = 2$ and m is the number of points sampling $\mathcal{C}_0$. We first bound the number of operations in one iteration of the main loop. Finding the minimum among $\mathcal{N}$ can be done in $\mathcal{O}(\log m)$ iterations using a binary heap [26]. Finding the L closest neighbors of p_a among $\mathcal{A}$, where L is typically $\sim$10 requires $\sim mL$ number of operations. (Since $\mathcal{A}$ is ordered in time, it is only necessary to look at the tail of the list.)

The grid method performs at most five iterations on an s^n grid, where s is typically $\sim$10. Computing w and evaluating constraints (V1) and (V2) at each point requires $\sim$10 operations. Evaluating constraint (E) requires $\sim L$ operations. We arrive at a total of $5s^n \times 10 \times L = \mathcal{O}(10^{n+2})$ operations to find $\mathbf{u}_d$. The complexity of the iterative solver is hard to bound a priori. Indeed, it is currently unclear how many operations may be required to compute the CFL condition. However, when $F \in C^1(\mathbb{R}^n \times [0,T])$, we find that devoting $\sim 10^2$ operations to this task results in a solver that only requires $\mathcal{O}(10)$ iterations to converge. Updating *Book* in Algorithm 4 requires $\mathcal{O}(m)$ operations, and as noted earlier, the other procedures usually require $\mathcal{O}(1)$ operations but may require up to $\mathcal{O}(m)$ (e.g., when topological changes occur). We arrive at the conclusion that the cost of one iteration is bounded by $\max\{\mathcal{O}(m), \mathcal{O}\left(10^{n+2}\right)\}$.

The parameter m is determined by the user and directly influences the total number of points N computed by the algorithm. We have $h \approx \text{length}(\mathcal{C}_0)/(2m)$ so that $h = \mathcal{O}(1/m)$.

When $F \equiv C$ or $F = F(t)$, then $\mathcal{M}$ is roughly decomposed into I strips comprising $\mathcal{O}(m)$ points. We have $I \leq T_{\text{F}}/(\Delta t)_{\min}$, where $(\Delta t)_{\min} = h/\sqrt{\max G^2 + 1} = \mathcal{O}(h)$, which yields a reliable estimate for the total number of points:

$$N = (\#\text{ of strips}) \times (\#\text{ points per strip}) = \frac{T_{\text{F}}}{\mathcal{O}(h)} \times \mathcal{O}(m) = \mathcal{O}(m^2). \tag{5.16}$$

When $F = F(\mathbf{x}, t)$, then $\mathcal{M}$ no longer exhibits a stratified structure. However, in general, we still expect a strip of average height $\Delta t = \mathcal{O}(h)$ to contain $\mathcal{O}(m)$ points, and therefore we arrive at the same estimate for N.

In summary, the total cost of the algorithm is CN, where

$$C = \max\{\mathcal{O}(m), \mathcal{O}\left(10^{n+2}\right)\} \quad \text{and} \quad N = \mathcal{O}(m^2). \tag{5.17}$$

Remark. Although the number of operations required to find $\mathbf{u}_d$ is admittedly large, they can be performed fairly efficiently. In MATLAB for example, the entire procedure can be vectorized. Empirically, it is found to take no more than 33% of the total computational time. (See Appendix D.)

6. Results. We illustrate the properties of the algorithm through various examples. More details about the tests procedures can be found in Appendix D.

6.1. Examples. With the exception of example (c), $\mathcal{C}_0$ consists of a circle centered at the origin. The corresponding manifolds $\mathcal{M}$ are presented in Figure 8. The exact normal is used to initialize the algorithm.

(a) *The expanding circle.* Using the constant speed $F \equiv 1$ yields a manifold $\mathcal{M}$ that consists of a truncated cone.

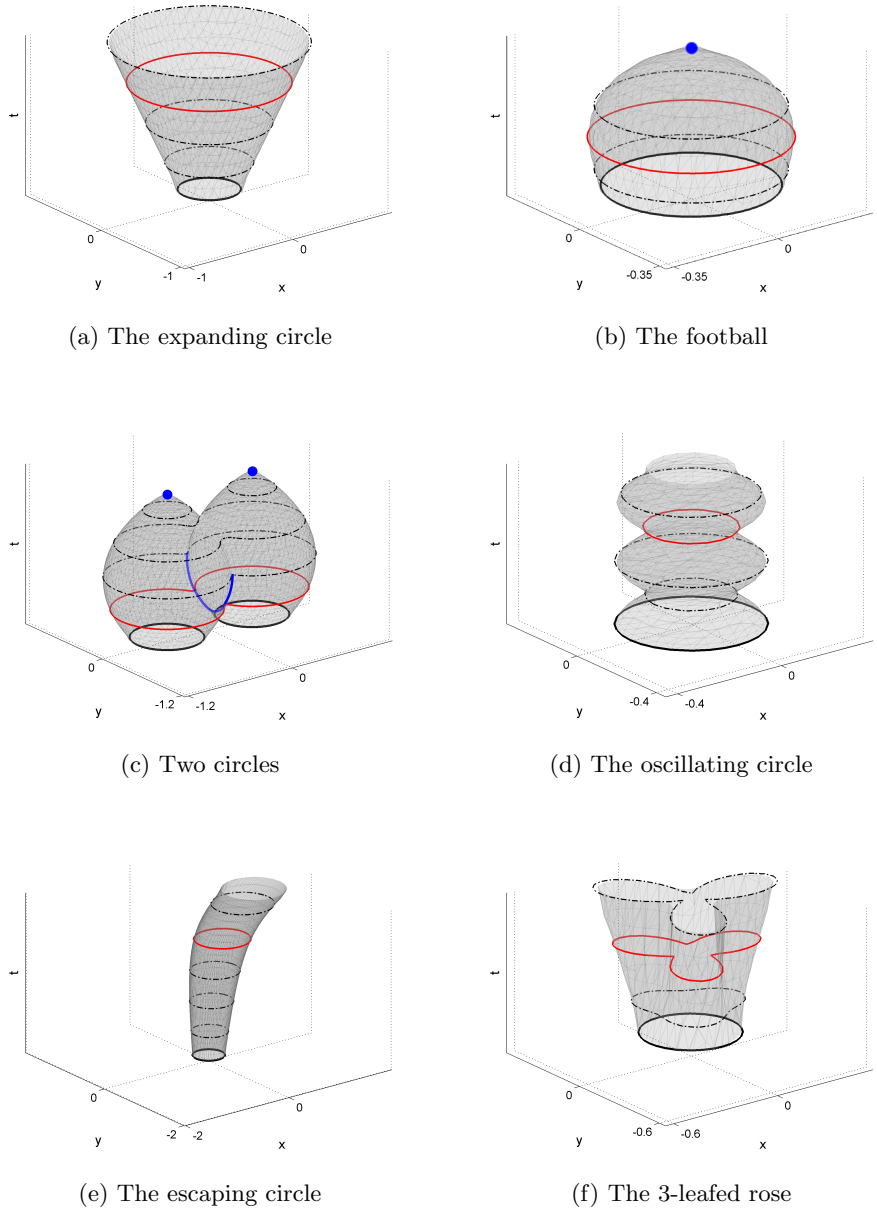
(b) *The football.* The speed is set to $F(t) = 1 - e^{10t-1}$, resulting in a circle that first expands and then contracts. $\mathcal{M}$ then resembles an American football.

(c) *Two circles.* As in the previous example, the time-dependent speed $F(t) = 1 - 2t$ changes sign. However, the fact that $\mathcal{C}_0$ now consists of two disjoint circles implies that topological changes occur during the evolution.

(d) *The oscillating circle.* The speed is set equal to $F = a\sin(b(t+c))$ for some constants a, b, and c.

(e) *The escaping circle.* The speed is such that the circle first expands and then moves in the positive x-direction while growing.

(f) *The 3-leafed rose.* F depends on the polar angle in such a way that $\mathcal{C}_t$ eventually takes the shape of a 3-leafed rose.



(a) The expanding circle

(b) The football

(c) Two circles

(d) The oscillating circle

(e) The escaping circle

(f) The 3-leafed rose

FIG. 8. *The manifold $\mathcal{M}$ in xyt-space. C^0 points are featured in blue.*

6.2. Local properties. In the following, the error associated to each point is measured as $E_p := |\phi(p)| = |\phi(\mathbf{x}, t)|$, where ϕ is the exact solution to the LSE (1.1).

6.2.1. Optimization problem. We start with an analysis of the grid method used to solve the optimization problem. Convergence results are presented on Figure 9 for examples (a), (d), and (f). The errors associated with the solutions of the direct and the iterative solver are both recorded. Second-order convergence with respect to the repulsion parameter h is clear. This result is consistent with the test procedure:

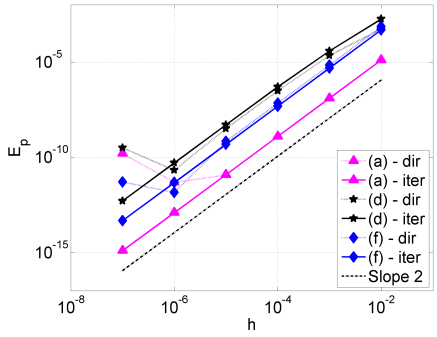
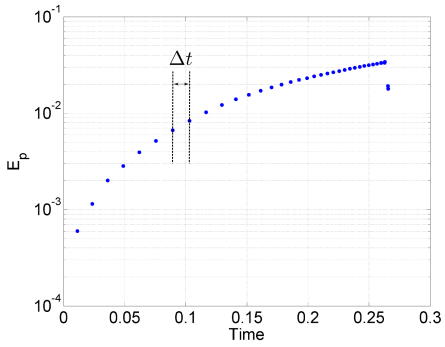


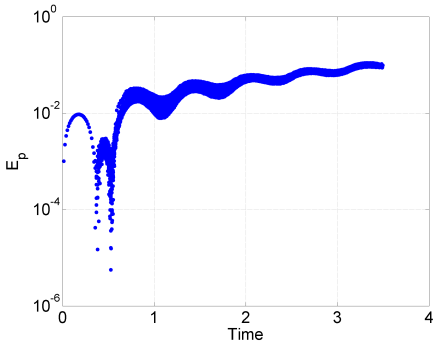
FIG. 9. *Local convergence results for examples* (a), (d), *and* (f).

TABLE 1
Number of iterations.

h	(a)	(d)	(f)
10^{-2}	1	10	9
10^{-3}	1	9	9
10^{-4}	1	9	8
10^{-5}	4	8	7
10^{-6}	6	7	6
10^{-7}	8	10	7



(a) The football, ex. (b)



(b) The oscillating circle, ex. (d) (6 periods).

FIG. 10. *Error versus time.*

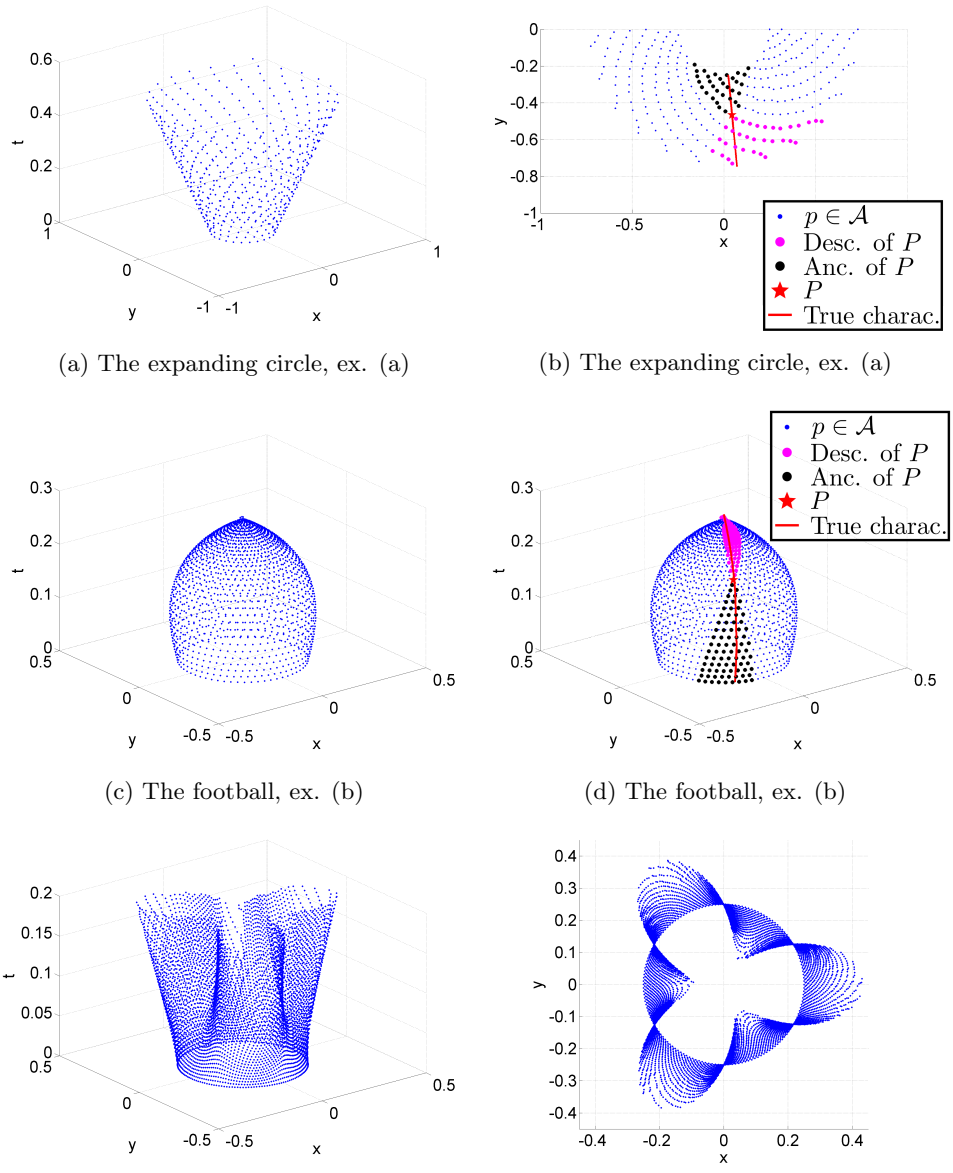
As h gets smaller, the child point moves closer to its parents, but the distance between the two parent points also decreases. For the most general case represented by example (f), the iterative solver slightly increases the accuracy of the solution. The number of iterations required to reach the tolerance is $\mathcal{O}(1)$, as recorded in Table 1.

6.2.2. Error propagation. We now discuss data obtained from running a full simulation. Convergence results for examples (b) and (d) are presented in Figure 10. In Figure 10(a), the error is seen to behave nicely even near the "tip" of $\mathcal{M}$. Note that the code naturally stops once $\mathcal{C}_t = \varnothing$. Moreover, the stratified structure alluded to in section 5.2.3 is evident. The symmetry of example (d) gives rise to cancellations as the circle contracts. The error appears to increase at a slow pace.

6.3. Global properties. Define

$$L_1(E) = h^n \sum_{p\in\mathcal{A}} E_p, \qquad L_2(E) = \sqrt{h^n \sum_{p\in\mathcal{A}} E_p^2}, \qquad L_\infty(E) = \max_{p\in\mathcal{A}} E_p \tag{6.1}$$

with $n = 2$, as well as the Haussdorff distance between the reconstructed and the exact curve, denoted by L_H. We first consider qualitative features of the manifold $\mathcal{M}$ before turning to global convergence results and speed tests.



(a) The expanding circle, ex. (a)

(b) The expanding circle, ex. (a)

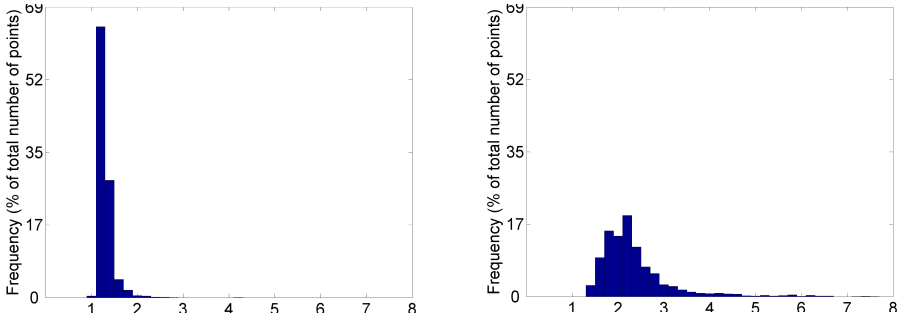
(c) The football, ex. (b)

(d) The football, ex. (b)

(e) The 3-leafed rose, ex. (f)

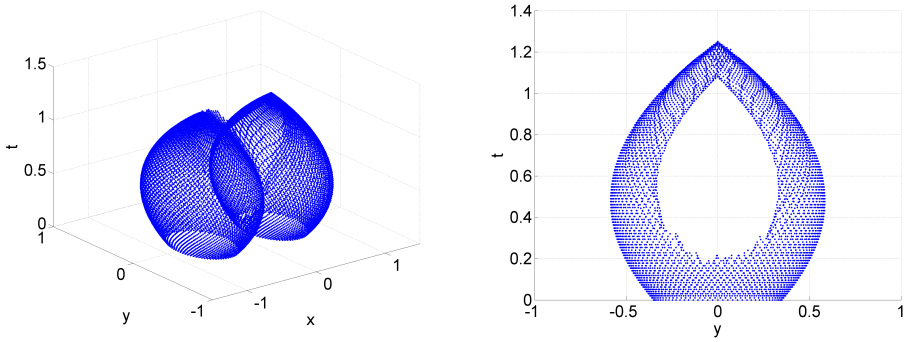
(f) The 3-leafed rose, ex. (f)

FIG. 11. *Sampling of $\mathcal{M}$ returned by the the algorithm.*



(a) Distance to Parent a (b) Distance to Parent b

FIG. 12. *Three-dimensional distance from a child to each of its parents, in units of h.*



(a) Full set (b) Side view of those points with $|x| < 0.5$

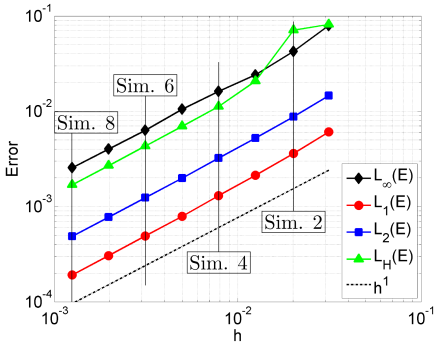
FIG. 13. *Sampling of $\mathcal{M}$ returned by the algorithm for example* (c)*, the two circles.*

6.3.1. Evenness of the sampling. As can be seen from Figure 11, the sampling of $\mathcal{M}$ is regular. This is confirmed when the distance from a child to its parents is monitored in example (b); see Figure 12.

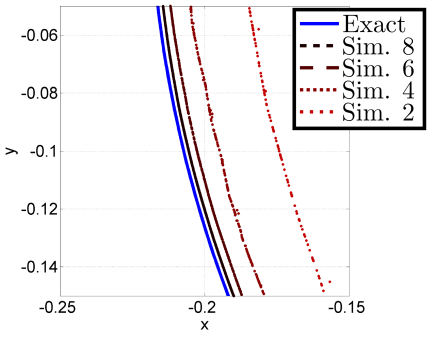
6.3.2. Domain of influence. Figure 11(a) may give the impression that the data propagate by spiralling outward. However, this is not the case, as can be seen from tracking a point's ancestry and descendance, as in Figures 11(b) and 11(d). It is of particular interest to note that the true characteristic going through the point does lie in the numerical past and future domains of influence.

6.3.3. Topological changes. The two-circles example illustrates the ability of the algorithm to deal with topological changes. As can be seen from Figure 13, the code naturally stops computing points when it reaches the y-axis. As a result, the circles appropriately merge. Their separation is also well-captured.

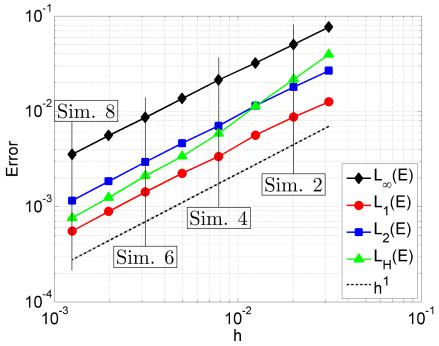
6.3.4. Convergence results. Convergence results are presented in Figure 14, along with the exact and the reconstructed curves obtained for some simulations. Example (e) is used to illustrate the robustness of the algorithm when $\mathcal{M}$ is smooth. Qualitatively similar results are obtained for examples (d), (e), and (f), namely, first-order convergence with respect to h is observed in all the norms considered. Similar



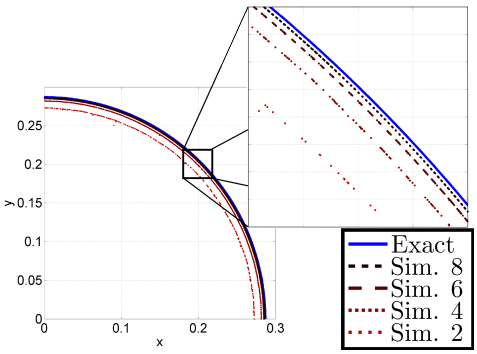
(a) The escaping circle, ex. (e)



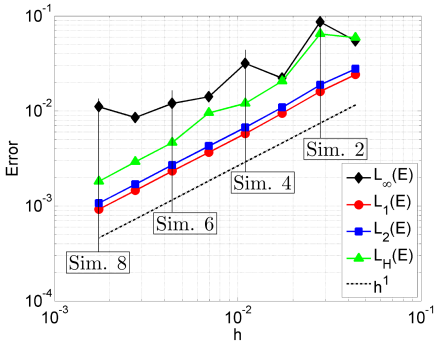
(b) The escaping circle, ex. (e)



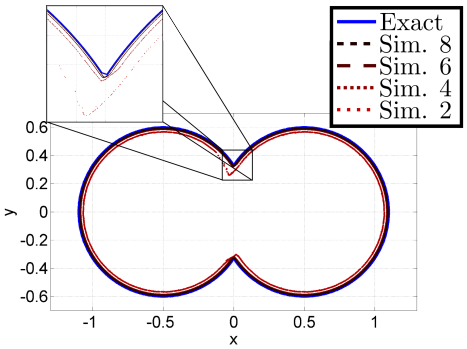
(c) The football, ex. (b)



(d) The football, ex. (b)



(e) The two circles, ex. (c) (up to $T = 0.5$)

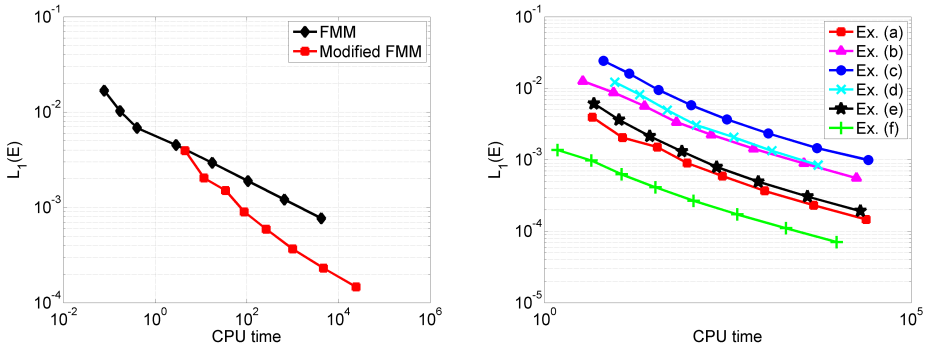


(f) The two circles, ex. (c)

FIG. 14. *Left: Global convergence results. Right: Exact and reconstructed curves.*

results are obtained for the football example, despite the fact that $\mathcal{M}$ is only C^0. The two-circles example also converges with first-order accuracy in the L_1, L_2, and L_H norms. Convergence in the L_∞ norm is not as clear.

6.3.5. Speed tests. Our method is now compared to the standard first-order FMM; see, for, example [26]. The solution to example (a) is computed on the set $|\mathbf{x}| < 0.75$ for various gridsizes—see Appendix D. The CPU times are presented in Figure 15(a). Note that the vertical axis is *the* L_1 *norm of the error* associated with



(a) The expanding circle, ex. (a) (b) Results obtained using our method

FIG. 15. *Accuracy of the solution versus CPU times.*

the sampling of $\mathcal{M}$. It is apparent that for higher accuracies, our method is faster than the standard FMM for this example. Figure 15(b) also presents CPU times obtained for nonmonotone examples: The trend is found to be similar to the monotone case.

Remark (see Appendices B and C). We point out a few changes that can be made to improve the computational speed of the code. As noted at the end of section 5.1.1, constraints (E) and (V2) are not necessarily equivalent. However, since in practice (V2) $\Longrightarrow$ (E) for an overwhelming number of points, the verification of constraint (E) may be taken out of the grid method and checked a posteriori. Based on Figure 9, running the iterative solver does not necessarily improve the accuracy of the solution. Making the stopping criterion depend on h, e.g., $|w_d^{n+1} - w_d^n| \leq 10h\Delta\tau$, may avoid unnecessary computations.

Conclusion. We presented a scheme that describes the nonmonotone propagation of fronts while featuring a numerical complexity comparable to that of the standard FMM. Local convergence was demonstrated and verified. Evidence of global convergence was supported by several examples where $F \in C^1$. The most general case, i.e., $F \in C^0$, requires further theoretical investigation and will therefore be the subject of subsequent papers. The theory presented in sections 2 and 3 trivially extends to higher dimensions. Nonetheless, some practical obstacles currently prevent the design of an algorithm when $n > 2$. The most prominent one is to properly monitor the global features of the manifold, as in Algorithm 4—see [10, 15, 16] for the case $n = 3$. Moreover, choosing an appropriate triplet of parents still requires some thought.

As it stands, the algorithm is first-order accurate. It may be extended to higher order using filtered schemes [14, 20]. Allowing h to depend on $(\mathbf{x}, t)$ would increase the accuracy in regions with high mean curvature and avoid unecessary computations in other regions. The precise adaptivity criteria must be carefully addressed. A different approach is to resort to *reseeding* by introducing points to the narrow band in regions of the front that are undersampled. We conclude by remarking that the novel ideas presented in this paper may apply to other evolution equations, such as linear advection, anisotropic propagation, or mean curvatuve flow.

Appendix A. *uvw*-coordinate system.

A.1. Tilted plane. Let a unit normal vector $\bar{\nu} = (n_1, n_2, n_3) \in S^3$ define a plane through the origin of the form $t = ax + by$, where $a = -n_1/n_3$ and $b = -n_2/n_3$. Define $\mu = \sqrt{1 + a^2 + b^2}$, as well as $\hat{w} = \pm(a, b, -1)/\mu$, where the sign is chosen such that $\text{sign}(\hat{w}_3) = \text{sign}(n_3)$. The following coordinate system can be verified to satisfy the conditions listed in section 2:

$$\text{(A.1)} \qquad \begin{cases} u &= \pm\frac{1}{\sqrt{a^2+b^2}}\,(\ b\ ,\ -a\ ,\ 0\ )\,, \\ v &= \frac{1}{\sqrt{a^2+b^2}\,\mu}\,(\ a\ ,\ b\ ,\ a^2+b^2\ )\,, \\ w &= \pm\frac{1}{\mu}\,(\ -a\ ,\ -b\ ,\ 1\ )\,. \end{cases}$$

A.2. Vertical plane. If $n_3 = 0$, then $\bar{\nu} \in S^3$ describes a vertical plane. The orthonormal coordinate system we use is then

$$\text{(A.2)} \qquad \begin{cases} u &= (\ -n_2\ ,\ n_1\ ,\ 0\ )\,, \\ v &= (\ 0\ ,\ 0\ ,\ 1\ )\,, \\ w &= (\ n_1\ ,\ n_2\ ,\ 0\ )\,. \end{cases}$$

A.3. $\bar{\nu}$ is the outward normal to $\mathcal{M}$. Suppose that $\mathcal{M}$ is C^1 at some point p_a and that $\bar{\nu} = \hat{\nu}(p_a)$. If $|F(p_a)| > 0$, then a neighborhood of p_a is locally described by $\psi(\mathbf{x})$, where ψ satisfies [26]:

$$\text{(A.3)} \qquad |\nabla\psi(\mathbf{x}_a)| = \frac{1}{|F(\mathbf{x}_a)|}.$$

Then $\phi(\mathbf{x}, t) = \text{sign}(F(p_a))\,(\psi(\mathbf{x}) - t)$, and we have

$$\text{(A.4)} \qquad \hat{\nu} \;=\; \text{sign}(F(p_a))\frac{(\psi_x, \psi_y, -1)}{\sqrt{\psi_x^2 + \psi_y^2 + 1}}.$$

Using the PDE (A.3) and the results from section A.2 with $a = \psi_x$ and $b = \psi_y$, we get

$$\text{(A.5)} \qquad \beta_3 = \frac{1}{\sqrt{1 + F^2(p_a)}} \qquad \text{and} \qquad \gamma_3 = \frac{-F(p_a)}{\sqrt{1 + F^2(p_a)}}.$$

If $F(p_a) = 0$, then $\hat{\nu}(p_a)$ describes a vertical plane, and $\beta_3 = 1$ and $\gamma_3 = 0$.

Appendix B. The iterative solver. The size of the pseudotime step used in the iterative solver is determined as follows. Defining the left-hand side of (3.13) as $-\tilde{H}(w^n)$, we set $\Delta\tau = \frac{9}{10}\,\frac{Q}{2\epsilon}$, where

$$\begin{aligned} &|\tilde{H}(w^1) - \tilde{H}(w^2)| \\ &\quad \le |\left(\tilde{\nu}^2 - \tilde{\nu}^1\right) \cdot \hat{R}| + |G^1| \left| \sqrt{\tilde{\nu}^1 \cdot \tilde{\nu}^1 - \left(\tilde{\nu}^1 \cdot \hat{R}\right)^2} - \sqrt{\tilde{\nu}^2 \cdot \tilde{\nu}^2 - \left(\tilde{\nu}^2 \cdot \hat{R}\right)^2} \right| \\ &\qquad + |G^1 - G^2| \sqrt{\tilde{\nu}^2 \cdot \tilde{\nu}^2 - \left(\tilde{\nu}^2 \cdot \hat{R}\right)^2} \;=:\; Q, \end{aligned}$$

$\epsilon = h/10$ and w_1, $w_2 \in (w_d^0 - \epsilon, w_d^0 + \epsilon)$. The neighborhood is sampled with 10 points.

Appendix C. The grid method: Pseudocode. This is the method we use in practice. Note that lines 6–10 can be completely vectorized.

Algorithm 6. Constrained optimization: The grid method.

$\bar{h} \leftarrow h/2$, $u_0 \leftarrow u_{\min}$, $v_0 \leftarrow h + v_a$,
$Q = 1$, tol $= 10^{-15}$, $i = 1$, $f_0 = 10^6$.
while $Q >$tol **do**
  Build a grid of $s \times s$ points, with meshsize h, centered at (u_0, v_0)
  On the grid, initialize $f = +\infty$.
  for each point (u, v) on the grid **do**
    Compute w using the direct solver (using $G(\mathbf{u}_a)$).
    Compute g_1, g_2 and g_3.
    If all three constraints are satisfied, compute $f(u, v)$.
  end for
  Let the minimum of f be f_i and occur at (u_d, v_d).
  $Q = |f_i - f_{i-1}|$
  $\bar{h} \leftarrow \bar{h}/2$, $u_0 \leftarrow u_d$, $v_0 \leftarrow v_d$, $i \leftarrow i + 1$
  if $i ==$ 5 or $f_i == +\infty$ **then** $Q = 0$ **end if**
end while
Return: (u_d, v_d).

Appendix D. Examples and tests procedures.

D.1. Examples. With the exception of the third example, $\mathcal{C}_0$ always consists of a single circle of radius r_0 centered at the origin.

The expanding circle. The speed is $F \equiv 1$ and the signed distance function that solves the LSE is $\phi(x, y, t) = \sqrt{x^2 + y^2} - t - r_0$ with $r_0 = 0.25$.

The football. The speed is $F(t) = 1 - e^{10t-1}$ and the signed distance function that solves the LSE is $\tilde{\phi}(x, y, t) = \sqrt{x^2 + y^2} - (r_0 - (e^{ct} - 1)/(ce) + t)$ with $r_0 = 0.25$.

Two circles. The speed is $F(t) = 1 - 2t$ and the signed distance function that solves the LSE up to time $T = 0.5$ is $\phi(x, y, t) = \sqrt{(x - \text{sign}(x)0.5)^2 + y^2} - \left(r_0 + t - t^2\right)$ with $r_0 = 0.35$.

The oscillating circle. The speed is $F(t) = a\sin(b(t + c))$ with $a = 0.7$, $b = 10$, and $c = 0.3$, and $\phi(x, y, t) = \sqrt{x^2 + y^2} - (r_0 + (a/b)(\cos(bc) - \cos(b(t + c))))$ is the signed distance function that solves the LSE with $r_0 = 0.25$.

The escaping circle. The speed is $F = (x - gt)(g't + g)/\sqrt{(x - gt)^2 + y^2} + c$, where $g(t) = \arctan(b(t - 0.5)) + \frac{\pi}{2}$, and the signed distance function that solves the LSE is $\phi(x, y, t) = \sqrt{(x - gt)^2 + y^2} - (r_0 + ct)$ with $r_0 = 0.25$, $b = 10$, and $c = 0.5$.

The 3-leafed rose. The speed is $F = \cos(l\theta)/\sqrt{1 + (lt/r)^2(\sin(l\theta))^2}$ and a solution of the LSE is $\phi(x, y, t) = r - (t\cos(l\theta) + r_0)$, where $l = 3$ is the number of petals and $r_0 = 0.25$. Note that this is not the signed distance function.

D.2. Tests procedures.

Time for computing $\mathbf{u}_d$. The estimate of the time taken to compute $\mathbf{u}_d$ provided in the remark at the end of section 5.2.3 is based on example (f), run with $m = 150$.

Optimization problem. Each point is obtained as follows. The parent points are exact: p_a, $p_b \in \mathcal{M}$. The point p_a is fixed, and p_b is such that $(x_b, y_b) = (x_a, y_a) + (-3, 4)h/8$. The exact normal at p_a is used to get the local coordinate frame. The optimization problem is solved using the direct solver and returns p_d^{dir}. Then the iterative solver is initialized with $w_d^0 = w_d^{\text{dir}}$ and run until $|w_d^{n+1} - w_d^n| < 10^{-10}\Delta\tau$.

Error propagation. Figures 10(a) and 10(b) were produced using $m = 60$.

Global properties. The results presented in Figure 11 were obtained using $M = 25$ for the expanding circle, $M = 60$ for the football, and $M = 150$ for the 3-leafed rose. The histograms of Figure 12 correspond to the football simulation. The binwidth is 0.2 and the total number of points N is 6229. In Figure 13, there are initially 80 points on each circle. The data used to get the results presented in Figure 14 are as follows: example (e), final T = 0.4, $t_H = 0.35$; example (b), $t_H = 0.1$; example (c), final T = 0.5, $t_H = 0.45$.

Hausdorff distance. To get the Hausdorff distance between the exact and the reconstructed curves, samplings of each set were first obtained. For the exact one, we used the exact solution to the LSE and the MATLAB `contour` function. For the reconstructed one, local Delaunay triangulations were obtained using appropriate subsets of $\mathcal{A}$. Let the resulting two cloud of points be respectively $\mathcal{C}\ell_{\rm rec}$ and $\mathcal{C}\ell_{\rm ex}$. Then the Hausdorff distance between those sets is

$$L_{\rm H}(\mathcal{C}\ell_{\rm rec}, \mathcal{C}\ell_{\rm ex}) = \max\left\{ \sup_{\mathbf{x}\in\mathcal{C}\ell_{\rm rec}} \inf_{\mathbf{y}\in\mathcal{C}\ell_{\rm ex}} d(\mathbf{x},\mathbf{y}),\ \sup_{\mathbf{y}\in\mathcal{C}\ell_{\rm ex}} \inf_{\mathbf{x}\in\mathcal{C}\ell_{\rm rec}} d(\mathbf{x},\mathbf{y}) \right\}, \tag{D.1}$$

where d is the Euclidean distance function. To get the first term in braces, the exact signed distance function is used, since for a given $\mathbf{x} \in \mathcal{C}\ell_{\rm rec}$, we have $\inf_{\mathbf{y}\in\mathcal{C}\ell_{\rm ex}} d(\mathbf{x},\mathbf{y}) = \phi(\mathbf{x}, t_H)$. To get the second one, given $\mathbf{y} \in \mathcal{C}\ell_{\rm ex}$, the nearest point $\mathbf{x} \in \mathcal{C}\ell_{\rm rec}$ is found.

Speed tests. To get the results labeled "Modified FMM" in Figure 15(a), our method is run. The points labeled "FMM" are obtained using the first-order FMM with different gridsizes. Points with $|\mathbf{x}| < 0.25 + 2dx$ are initialized with exact values. The solution to the FMM is computed on the set $|\mathbf{x}| < 0.75 + dx$ (the neighbors of those accepted points with $|\mathbf{x}| > 0.75$ were not added to the narrow band). The procedure used to sort and update the narrow band is the same in both methods:

- The point with the smallest time value is found using the MATLAB `min` command.
- This point is removed from the narrow band by deleting the corresponding row (using $\mathcal{N}(I,:) = [\,]$).
- Each new point is added at the end of the narrow band.

The CPU time was evaluated using the MATLAB `cpu` command. The final times of the simulations are: (a) 0.5, (c) 0.5, (d) 1.5 period, (e) 0.4, (f) 0.19.

Acknowledgments. The authors would like to thank Prof. Bruce Shepherd and Dr. Jan Feys for helpful discussions about the optimization problem. We also thank the anonymous referees for their very constructive comments.

REFERENCES

[1] Y. Achdou, G. Barles, H. Ishii, G. Litvinov, P. Loreti, and N. Tchou, *Hamilton-Jacobi Equations: Approximations, Numerical Analysis and Applications: Cetraro, Italy* 2011, P. Loreti and N. A. Tchou, eds., Lecture Notes in Mathematics/C.I.M.E. Foundation Subseries, Springer, Berlin, 2013.

[2] G. Barles and P. E. Souganidis, *Convergence of approximation schemes for fully nonlinear second order equations*, Asymptot. Anal., 4 (1991), pp. 271–283.

[3] D. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods*, Athena Scientific, Belmont, MA, 1996.

[4] E. Carlini, M. Falcone, N. Forcadel, and R. Monneau, *Convergence of a generalized fast-marching method for an eikonal equation with a velocity-changing sign*, SIAM J. Numer. Anal., 46 (2008), pp. 2920–2952.

[5] L.-T. Cheng and Y.-H. Tsai, *Redistancing by flow of time dependent eikonal equation*, J. Comput. Phys., 227 (2008), pp. 4002–4017.

[6] M. G. Crandall, L. C. Evans, and P.-L. Lions, *Some properties of viscosity solutions of Hamilton-Jacobi equations*, Trans. Amer. Math. Soc., 282 (1984), pp. 487–502.
[7] M. G. Crandall, H. Ishii, and P.-L. Lions, *User's guide to viscosity solutions of second order partial differential equations*, Bull. Amer. Math. Soc., 27 (1992), pp. 1–67.
[8] M. G. Crandall and P.-L. Lions, *Viscosity solutions of Hamilton-Jacobi equations*, Trans. Amer. Math. Soc., 277 (1983), pp. 1–42.
[9] E. Dijkstra, *A note on two problems in connexion with graphs*, Numer. Math., 1 (1959), pp. 269–271.
[10] J. Du, B. Fix, J. Glimm, X. Jia, X. Li, Y. Li, and L. Wu, *A simple package for front tracking*, J. Comput. Phys., 213 (2006), pp. 613–628.
[11] L. Evans, *Partial Differential Equations*, Grad. Stud. Math., AMS, Providence, RI, 2010.
[12] L. C. Evans, *An Introduction to Mathematical Optimal Control Theory*, Version 0.2, Lectures Notes, 1983; also available online from https://math.berkeley.edu/~evans/control.course.pdf.
[13] M. Falcone, *The minimum time problem and its applications to front propagation*, in Motion by Mean Curvature and Related Topics, De Gruyter, Berlin, 1994, pp. 70–88.
[14] B. D. Froese and A. M. Oberman, *Convergent filtered schemes for the Monge–Ampre partial differential equation*, SIAM J. Numer. Anal., 51 (2013), pp. 423–444.
[15] J. Glimm, J. W. Grove, X. L. Li, K.-M. Shyue, Y. Zeng, and Q. Zhang, *Three dimensional front tracking*, SIAM J. Sci. Comput., 19 (1995), pp. 703–727.
[16] J. Glimm, J. W. Grove, X. L. Li, and D. C. Tan, *Robust computational algorithms for dynamic interface tracking in three dimensions*, SIAM J. Sci. Comput., 21 (2000), pp. 2240–2256.
[17] J. Guckenheimer and A. Vladimirsky, *A fast method for approximating invariant manifolds*, SIAM J. Appl. Dynam. Syst., 3 (2004), pp. 232–260.
[18] P.-L. Lions, *Generalized Solutions of Hamilton-Jacobi Equations*, Res. Notes Math. 69, Pitman, Boston, 1982.
[19] A. M. Oberman, *Convergent difference schemes for degenerate elliptic and parabolic equations: Hamilton–Jacobi equations and free boundary problems*, SIAM J. Numer. Anal., 44 (2006), pp. 879–895.
[20] A. M. Oberman, and T. Salvador, *Filtered schemes for Hamilton-Jacobi equations: A simple construction of convergent accurate difference schemes,* J. Comput. Phys., 284 (2015), pp. 367–388.
[21] S. Osher, *A level set formulation for the solution of the Dirichlet problem for Hamilton–Jacobi equations*, SIAM J. Math. Anal., 24 (1993), pp. 1145–1152.
[22] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*, Appl. Math. Sci. 153, Springer-Verlag, New York, 2003.
[23] S. Osher and J. Sethian., *Fronts propagating with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulations*, J. Comput. Phys., 79 (1988)), pp. 12–49.
[24] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes: The Art of Scientific Computing*, 3rd ed., Cambridge University Press, New York, 2007.
[25] O. Runborg, *Analysis of high order fast interface tracking methods*, Numer. Math., 128 (2014), pp. 339–375.
[26] J. Sethian, *A fast marching level set method for monotonically advancing fronts*, Proc. Natl. Acad. Sci., USA, 93 (1996), pp. 1591–1595.
[27] J. Sethian, *Fast marching methods*, SIAM Rev., 41 (1999), pp. 199–235.
[28] J. Sethian, *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*, Cambridge Monogr. Appl. Comput. Math., Cambridge University Press, Cambridge, UK, 1999.
[29] J. A. Sethian, *Numerical methods for propagating fronts*, in Variational Methods for Free Surface Interfaces, P. Concus and R. Finn, eds., Springer New York, 1987, pp. 155–164.
[30] A. Tcheng, *Fast Marching Algorithms for Non-monotonically Evolving Fronts*, Ph.D. thesis, McGill University, 2015.
[31] A. Tcheng and J.-C. Nave, *A low-complexity algorithm for non-monotonically evolving fronts*, J. Sci. Comput. (2016), pp. 1–27, DOI:10.1007/s10915-016-0231-8.
[32] Y.-H. R. Tsai, L.-T. Cheng, S. Osher, and H.-K. Zhao, *Fast sweeping algorithms for a class of Hamilton-Jacobi equations*, SIAM J. Numer. Anal., 41 (2003), pp. 673–694.
[33] J. N. Tsitsiklis, *Efficient algorithms for globally optimal trajectories*, IEEE Trans. Automat. Control, 40 (1995), pp. 1528–1538.
[34] A. Vladimirsky, *Static PDEs for time-dependent control problems*, Interfaces Free Bound., 8 (2006), pp. 281–300.
[35] A. B. Vladimirsky, *Fast Methods for Static Hamilton-Jacobi Partial Differential Equations*, Ph.D. thesis, Lawrence Berkeley National Laboratory, Berkeley, CA, 2001.