J Sci Comput (2016) 67:837–859
DOI 10.1007/s10915-015-0105-5

A Fast Sweeping Method for Eikonal Equations on Implicit Surfaces

Tony Wong[1] · **Shingyu Leung**[1]

Received: 21 May 2015 / Revised: 13 August 2015 / Accepted: 12 September 2015 /
Published online: 18 September 2015

Abstract We propose a computational efficient yet simple numerical algorithm to solve the surface eikonal equation on general implicit surfaces. The method is developed based on the embedding idea and the fast sweeping methods. We first approximate the solution to the surface eikonal equation by the Euclidean weighted distance function defined in a tubular neighbourhood of the implicit surface, and then apply the efficient fast sweeping method to numerically compute the corresponding viscosity solution. Unlike some other embedding methods which require the radius of the computational tube satisfies $h = O(\Delta x^{\gamma})$ for some $\gamma < 1$, our approach allows $h = O(\Delta x)$. This implies that the total number of grid points in the computational tube is optimal and is given by $O(\Delta x^{1-d})$ for a co-dimensional one surface in $\mathbb{R}^d$. The method can be easily extended to general static Hamilton–Jacobi equation defined on implicit surfaces. Numerical examples will demonstrate the robustness and convergence of the proposed approach.

Keywords Partial differential equations · Implicit surfaces · Level set method · Eikonal equations · Interface modeling

1 Introduction

We consider the eikonal equation in an isotropic medium, given by

$$\begin{aligned} |\nabla u(\mathbf{x})| &= \tfrac{1}{F(\mathbf{x})}, \quad && \mathbf{x} \in \Omega \setminus P \\ u(\mathbf{x}_s) &= 0, && \mathbf{x}_s \in P \end{aligned} \tag{1}$$

✉ Shingyu Leung
masyleung@ust.hk

Tony Wong
kwwongam@ust.hk

[1] Department of Mathematics, Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong

where $\Omega \subset \mathbb{R}^n$ is an open bounded domain, P is the set of sources, $F(\mathbf{x})$ is the wave velocity and $u(\mathbf{x})$ denotes the travel-time of the wave from the source $\mathbf{x}_s$ to a target point $\mathbf{x} \in \Omega$. This first order Hamilton–Jacobi (HJ) equation is one of the fundamental building block for the level set method [24,32] and such nonlinear partial differential equation can be found in a wide range of applications including geometrical optics, computer vision, geophysics, and etc. To numerically obtain the first arrival solution, tremendous number of efficient numerical methods have been proposed for approximating the viscosity solution [6,7] to Eq. (1). These methods include the fast sweeping method [12,13,28,29,35,40,41] and the fast marching method [27,33,36].

In this paper, we consider a slightly more general equation where Ω is a general implicit surface Σ, and the typical gradient is replaced by the surface gradient ∇_Σ, which is the gradient intrinsic to the surface. If the surface is embedded in the Euclidean space, the surface gradient is just the orthogonal projection of the usual gradient onto the tangent plane of the surface. Mathematically, if we consider a surface Σ in a scalar field u, the surface gradient is defined as

$$\nabla_\Sigma u = \nabla u - (\mathbf{n} \cdot \nabla u)\mathbf{n},$$

where $\mathbf{n}$ is a unit normal vector defined on Σ. We aim to develop an efficient and converging numerical algorithm for obtaining the viscosity solution of the surface eikonal equation given by

$$\begin{aligned} |\nabla_\Sigma u(\mathbf{x})| &= \tfrac{1}{F(\mathbf{x})}, \quad && \mathbf{x} \in \Sigma \setminus P \\ u(\mathbf{x}_s) &= 0, && \mathbf{x}_s \in P. \end{aligned} \tag{2}$$

Similar to the typical eikonal equation (1), the viscosity solution can be interpreted as the first arrival time for high frequency surface wave propagation on Σ, for example [10], and is useful in modeling constrained motions of curves/surfaces and image regularizations [18].

Numerically, on the other hand, there has not been much attention paid to the eikonal equation on surfaces (2). The first work in this direction was proposed in [14] to determine the geodesic paths on manifolds using the fast marching method. The method has extended the fast marching method on triangulated mesh by incorporating the local metric from the geometry. Similar idea can be found in [38,39]. Some other methods have been proposed for surfaces represented by an explicit parametrization, see for example [4,34,37].

All approaches discussed above require some form of explicit representation of the interface. In this paper, we are interested in developing a numerical algorithm for implicit surfaces without any explicit interface triangulation. The resulting algorithm, coupled with the level set method [24,32], will then be well-suited for applications which involve eikonal solution on dynamic surfaces, such as those biological modelings in [8,26]. One approach for solving the surface eikonal equation on surfaces has been proposed in [21] based on the fast marching method. Since the solution of the surface eikonal equation on an implicit surface is the intrinsic weighted distance function, the paper has proposed to approximate such distance function by the Euclidean weighted distance function defined in a tubular neighbourhood of the implicit surface. In other words, one replaces the surface eikonal equation (2) by the typical eikonal equation (1) defined in a narrow computation tube $\{|\phi| \leq h\}$ containing the original surface for some tube radius h. In the following context, this method will be called the embedding method for short.

Since the surface PDE is extended off the interface to a corresponding PDE in a small neighborhood of the surface on a fixed Eulerian mesh, we can first apply to the equation any

well-developed numerical technique to determine the Euclidean weighted distance function and then use it to approximate the original intrinsic distance function on the surface. Fast marching method was used in [21]. In this work, we will explore the possibility of using fast sweeping methods instead. We will discuss two numerical Hamiltonians including the Godunov Hamiltonian, which has been widely used for the eikonal equation (1) because of its convergence behavior, and also the Lax–Friedrichs (LxF) Hamiltonian, which can be easily applied to even non-convex HJ equations.

In order to observe the numerical convergence to the exact solution in the expected order, some embedding methods require that the radius of the computational tube satisfies $h = O(\Delta x^{\gamma})$ for some $\gamma < 1$, such as [21]. This implies that the total number of mesh point in such a tube will be given by $O(\Delta x^{\gamma-d})$ for a codimension-one surface in $\mathbb{R}^d$. We will give a detailed study of this behavior in Sect. 3, at least in the context of the fast sweeping method. Yet a more important criteria in designing an embedding method for the problem is to take care of the causality in designing the numerical scheme. When incorporating with the embedding method with a computational tube radius $h = O(\Delta x)$, we have observed that a simple fast sweeping method based on the Godunov Hamiltonian fails to converges to the exact solution. The main reason, as will be analyzed, is that this approach has not been able to correctly take care of the characteristic directions. We therefore propose to introduce an extension equation in an extra layer of computational tube, which helps to fix the direction of the characteristics in the numerical scheme. Even with the extra layer of the computational cells, the proposed method only requires that the overall tube radius satisfies $h = O(\Delta x)$. This is, therefore, optimal in the sense that the total number of mesh points in the computational tube is $O(\Delta x^{1-d})$ for a co-dimensional one surface in $\mathbb{R}^d$.

It is worthwhile to compare our approach to some fast sweeping approaches and embedding ideas. Qian et al. [28,29] has proposed a fast sweeping method for solving eikonal equations on surfaces explicitly represented by a triangulation. One main contribution in those work is a new ordering strategy for the fast sweeping method so that the alternating sweepings in each iteration can cover all directions of information propagation. That method has been shown to be computationally efficient and numerically convergent for a triangulated domain. Our proposed method in this paper, on the other hand, relies simply on the natural ordering provided by the underlying Cartesian mesh and the resulting sweeping strategy does not require to introduce any reference point on the surface.

In the level set framework, Bertalmio et al. [3] has proposed an embedding approach for solving *time-dependent* PDEs on surfaces. The main idea is to first express the surface gradient operator using the typical Euclidean gradient and a projection operator

$$\mathrm{P}_{\mathbf{n}} = I - \mathbf{n} \otimes \mathbf{n},$$

where $\mathbf{n}$ can be easily determined using the level set function. Related to this level set approach, Macdonald and Ruuth [19] has developed the Closest Point Method (CPM) to solve *time-dependent* PDEs on implicit surfaces. With the closest point function cp: $\mathbf{x} \to \mathbf{y}$ which maps a grid point $\mathbf{x}$ to the closest point $\mathbf{y} = \mathrm{argmin}_{\mathbf{z}\in\Sigma} \|\mathbf{x} - \mathbf{z}\|_2$, one can show

$$\nabla_{\Sigma} u(\mathbf{y}) = \nabla u(\mathbf{y}) = \nabla u(cp(\mathbf{x}))$$

so that the surface gradient can be computed using the Euclidean gradient. Not only that those works have emphasized only on *time-dependent* equations while this paper concentrates on *static* Hamilton–Jacobi equations, the main difference between the current approach and these work is that the extrinsic surface gradient is now simply *replaced* by the Euclidean gradient. Since we are interested only in a robust convergent first order scheme, such approximation

is significant for the current application and leads to a computationally efficient numerical algorithm.

The paper is organized as follows. In Sect. 2, we first summarize the fast sweeping method for HJ equations including the eikonal equation (1) and the hyperbolic conservation laws, which is the building block of our approach. In Sect. 3, we will then follow the approach in [21] but naively replace the fast marching method by the fast sweeping method. However, we are also going to show in the same section that the numerical approach will lead to a non-converging method. Based on the analysis in Sect. 3, we propose a simple yet converging numerical algorithm for Eq. (2) in Sect. 4. Some extensions will be discussed in Sect. 5. Section 6 shows various numerical examples to demonstrate the feasibility and robustness of the new formulation.

2 Fast Sweeping Methods for Eikonal Equations and the Advection Equations

In this section, we will briefly summarize the fast sweeping method for solving the HJ equations. For a complete description of the algorithm, we refer interested readers to [12, 13,28,29,35,40,41] and thereafter for general HJ equations, and [5,15–17] for extensions to advection equations and hyperbolic conservation laws.

The fast sweeping method was originated in [30] in the context of computer graphics for the shape-from-shading problem. For simplicity, we present the algorithm for the two-dimensional case only. Generalization to higher dimensions is rather straight-forward. We first discretize the computation domain into a uniform mesh $\{\mathbf{x}_{i,j}\}$ with mesh size Δx and represent the solution at the mesh points by $u_{i,j}$. The fundamental idea is to design of an upwind, monotone and consistent discretization for the nonlinear term $|\nabla u|$. Furthermore, if there is a simple local solver for the discretized system, one can then obtain an efficient numerical algorithm for solving the nonlinear partial differential equation. The term *sweeping* comes from the fact that all methods in this class can be easily incorporated with the symmetric Gauss–Seidel iterations, see Algorithm 1, which may lead to an $O(N)$ algorithm with N the total number of mesh points in the computational domain.

Algorithm 1: The fast sweeping method for the eikonal equation $|\nabla u| = 1$ or the advection equation $\mathbf{v} \cdot \nabla u = (p, q) \cdot (u_x, u_y) = 0$.

Data: The set of boundary points (x_s, y_s), the mesh size Δx.
Result: $u_{i,j}$ in the computational domain.
Initialization: Assign the boundary condition. For the eikonal equation, set $u_{i,j} = 0$ if (x_i, y_j) is a point in the boundary, otherwise $u_{i,j} = \infty$;
while *not converges* **do**
  for *each of the four sweeping directions* **do**
    if $u_{i,j} \neq 0$ **then**
      update $u_{i,j}$ according to the local solution to the corresponding update formula associated to the numerical discretization;
    end
  end
end

The fast sweeping approach in Algorithm 1 can be easily adopted by the eikonal equations, advection equations or hyperbolic conservation laws. In particular, if we are solving the

eikonal equation $|\nabla u| = 1$, we have a choice in picking a numerical Hamiltonian for the discretization. For example, for each (x_i, y_j), we define $u^x_{\min} = \min(u_{i-1,j}, u_{i+1,j})$ and $u^y_{\min} = \min(u_{i,j-1}, u_{i,j+1})$. The Godunov Hamiltonian [41] has the following simple update formula, given by

$$u_{i,j} \leftarrow \begin{cases} \min\left(u^x_{\min}, u^y_{\min}\right) + \Delta x & \text{if } \left|u^x_{\min} - u^y_{\min}\right| > \Delta x \\ \frac{1}{2}\left[u^x_{\min} + u^y_{\min} + \sqrt{2\Delta x^2 - \left(u^x_{\min} - u^y_{\min}\right)^2}\right] & \text{otherwise.} \end{cases} \tag{3}$$

The corresponding local solver for the LxF Hamiltonian [13] is given by

$$u_{i,j} \leftarrow \frac{u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}}{4} + \frac{\Delta x}{2}\left[1 - \sqrt{\left(\frac{u_{i+1,j} - u_{i-1,j}}{2\Delta x}\right)^2 + \left(\frac{u_{i,j+1} - u_{i,j-1}}{2\Delta x}\right)^2}\right]. \tag{4}$$

When solving the advection equations $\mathbf{v} \cdot \nabla u = (p, q) \cdot (u_x, u_y) = 0$, we can use the simple upwind differencing to discretize the equation and derive the update formula [5,15–17]. At each point (x_i, y_j), we define $p^{\pm} = \frac{1}{2}(p_{i,j} \pm |p_{i,j}|)$ and, therefore, obtain the update formula given by

$$u_{i,j} \leftarrow \frac{p^+ u_{i-1,j} - p^- u_{i+1,j} + q^+ u_{i,j-1} - q^- u_{i,j+1}}{|p_{i,j}| + |q_{i,j}|}. \tag{5}$$

3 The First Attempt

In this section, we first directly apply the embedding idea in [21] on a computational tube of radius $O(\Delta x)$ but replace the eikonal solver by the fast sweeping method. Because the Hamiltonian is convex and simple enough, the closed-form solution to the local problem from the Godunov discretization can be explicitly determined. This makes the Godunov fast sweeping particularly attractive.

Following the embedding idea in [21], we attempt to solve the surface eikonal equation $|\nabla_\Sigma u| = 1$ not only on the interface Σ but replace it by the typical eikonal equation $|\nabla u| = 1$ defined in a computational tube Γ_h of radius h enclosing the surface Σ. In particular, we might simply update the numerical solution $u_{i,j}$ only if the grid point $(x_i, y_j) \in \Gamma_h$ while keep it to be the numerical infinity otherwise. Therefore, the Godunov Hamiltonian will not pass any information from outside Γ_h to the interior. This numerical scheme is extremely easy to code, as demonstrated in Algorithm 2. Like other fast sweeping methods, the iterative scheme converges to an approximation to the eikonal equation for a *given* Δx. But in this section, we are going to demonstrate that this proposed scheme does not converge to the exact viscosity solution to the surface eikonal equation as Δx tends to zero if we have the tube radius $\Gamma_h = O(\Delta x)$. Even so, the analysis of such a straight-forward scheme will still be important since it motivates us the proposed approach which will be discussed later in Sect. 4.

For simplicity, we analyze this scheme in the two-dimensional case and consider $\Sigma = \{x = y\}$ with the point source located at the origin and we determine the solution in $[0, 1]^2$. Therefore, the exact solution on Σ at (x, x) is simply the Euclidean distance from the origin and is given by $\sqrt{2}x$. Since the geometry of this example is very simple, it turns out that we can calculate the explicit formula of the numerical solution from the Godunov Hamiltonian

Algorithm 2: The first attempt to solve the surface eikonal equation $|\nabla_\Sigma u| = 1$ using the Godunov fast sweeping method.

Data: The source location (x_s, y_s), the mesh size Δx, the level set representation of the surface $\phi_{i,j}$ and the tube radius $h = O(\Delta x)$.
Result: $u_{i,j}$ in the computational tube.
Initialization: Set $u_{i,j} = 0$ if (x_i, y_j) is at the point source, otherwise $u_{i,j} = \infty$;
while *not converges* **do**
  for *each of the four sweeping directions* **do**
    if $|\phi_{i,j}| \le h$ *and* $u_{i,j} \neq 0$ **then**
      update $u_{i,j}$ using (3);
    end
  end
end

for some tube radius h with $\frac{h}{\Delta x} \in \left(\frac{1}{\sqrt{2}}, \frac{3}{\sqrt{2}}\right)$. To do that, we first state some essential observations.

Observation 3.1 *$u_{i,j}$ depends only on $u_{i-1,j}$ and $u_{i,j-1}$.*

Because of the special structure of our computational domain, information propagates only from the lower left part to the upper right part in the domain. The upwind Godunov Hamiltonian enforces a grid point (x_i, y_j) to choose its lower and left neighbour grid values $u_{i-1,j}$ and $u_{i,j-1}$ in the update formula. This observation is crucial since it implies that the numerical solution will converge in only one sweeping direction (from lower left to upper right).

Observation 3.2 *Assume $\frac{h}{\Delta x} \in \left[\sqrt{2}, \frac{3}{\sqrt{2}}\right)$ and consider a given level $y = y_j$ where $j \ge 3$. There are only five grid points $u_{j,j-2}, \ldots, u_{j,j+2}$ within the computation tube.*

This is due to the construction of the computational tube and is clearly demonstrated in Fig. 1.

Observation 3.3 *If $\frac{h}{\Delta x} \in \left[\sqrt{2}, \frac{3}{\sqrt{2}}\right)$, we have $u_{i+2,i} = u_{i+1,i} + \Delta x$ and $u_{i,i+2} = u_{i,i+1} + \Delta x$.*

From Observation 3.2, $u_{i+2,i}$ is a boundary point of the computational tube. Based on the Observation 3.1, the value depends only on $u_{i+1,i}$ and $u_{i+2,i-1}$. However, since $u_{i+2,i-1}$ lies outside the computational tube, its default value is set to be the numerical infinity and will not be updated. When using the Godunov Hamiltonian, we therefore have $u_{i+2,i} = u_{i+1,i} + \Delta x$. The other formula can also be obtained in a similar argument.

We now state two propositions which lead to the closed form numerical solution from the Godunov Hamiltonian under some simple configurations. The first one is about a symmetry in the numerical solution, while the second one gives us the explicit formulae of the numerical solution.

Proposition 3.1 *If $\frac{h}{\Delta x} \in \left(\frac{1}{\sqrt{2}}, \frac{3}{\sqrt{2}}\right)$, we have $u_{i,j} = u_{j,i}$.*

Proof By Observation 3.3, the proposition is equivalent to show that

$$u_{i,i+1} = u_{i+1,i} \text{ and } u_{i,i+2} = u_{i+2,i}$$

for $i \ge 3$. We prove it by induction. For case $i \le 3$, it can be shown by directly calculation. Now assume case $i \le k$ is true, where $k \ge 3$. For the case where $i = k + 1$, we use

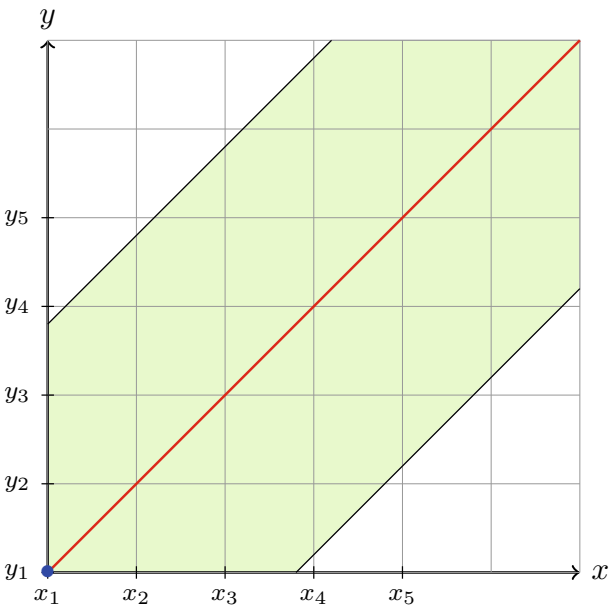

Fig. 1 The *red line* is the interface Σ given by $y - x = 0$. The *green region* indicates the computation tube Γ_h. The *blue dot* denotes the source location at the origin $(x_1, y_1) = (0, 0)$ (Color figure online)

Observation 3.1 and deduce that $u_{k+1,k+2}$ depends on $u_{k,k+2}$ and $u_{k+1,k+1}$. And $u_{k+2,k+1}$ depends on $u_{k+1,k+1}$ and $u_{k+2,k}$. By the induction hypothesis, $u_{k,k+2} = u_{k+2,k}$. Therefore, $u_{k+1,k+2}$ and $u_{k+2,k+1}$ both use the same set of values in their updating formula based on the Godunov Hamiltonian, and therefore their values equal. Next, by Observation 3.3, we have $u_{k+1,k+3} = u_{k+1,k+2} + \Delta x$ and $u_{k+3,k+1} = u_{k+2,k+1} + \Delta x$. Since we have just shown $u_{k+1,k+2} = u_{k+2,k+1}$. It follows that $u_{k+1,k+3} = u_{k+3,k+1}$ and it leads to the proposition by induction. □

Proposition 3.2 *Denote* $\alpha_\pm = 1 \pm \frac{1}{\sqrt{2}}$. *If* $\frac{h}{\Delta x} \in \left[\sqrt{2}, \frac{3}{\sqrt{2}}\right)$, *we have*

1. $u_{i,i} = u_{i-1,i} + \frac{\Delta x}{\sqrt{2}}$;
2. $|u_{i-1,i+1} - u_{i,i}| < \Delta x$;
3. $u_{i,i+1} = u_{i-1,i} + \frac{\Delta x}{2}\left(\alpha_+ + \sqrt{2 - \alpha_-^2}\right)$.

Proof These expressions can be explicitly constructed.

1. $u_{i,i}$ is updated based on $u_{i-1,i}$ and $u_{i,i-1}$. By the symmetry of the numerical solution, we have $u_{i-1,i} = u_{i,i-1}$. Therefore,

$$u_{i,i} = \frac{u_{i-1,i} + u_{i,i-1} + \sqrt{2\Delta x^2 - (u_{i-1,i} - u_{i,i-1})^2}}{2} = u_{i-1,i} + \frac{\Delta x}{\sqrt{2}}.$$

2. We have

$$|u_{i-1,i+1} - u_{i,i}| = |u_{i+1,i-1} - u_{i,i}| = \left|(u_{i,i-1} + \Delta x) - \left(u_{i,i-1} + \frac{\Delta x}{\sqrt{2}}\right)\right| = \left(1 - \frac{1}{\sqrt{2}}\right)\Delta x < \Delta x.$$

3. $u_{i,i+1}$ is updated using $u_{i-1,i+1}$ and $u_{i,i}$. Since $|u_{i-1,i+1} - u_{i,i}| < \Delta x$, we have

$$\begin{aligned} u_{i,i+1} &= \frac{u_{i-1,i+1} + u_{i,i} + \sqrt{2\Delta x^2 - (u_{i-1,i+1} - u_{i,i})^2}}{2} \\ &= \frac{(u_{i-1,i} + \Delta x) + \left(u_{i-1,i} + \frac{\Delta x}{\sqrt{2}}\right) + \sqrt{2\Delta x^2 - \left(1 - \frac{1}{\sqrt{2}}\right)^2 \Delta x^2}}{2} \\ &= u_{i-1,i} + \frac{\Delta x}{2}\left(\alpha_+ + \sqrt{2 - \alpha_-^2}\right). \end{aligned}$$

□

These observations and propositions provide us a recursive relation to explicitly construct the numerical solution from the Godunov Hamiltonian everywhere in the computation tube if the tube radius satisfies $\sqrt{2} \le \frac{h}{\Delta x} < \frac{3}{\sqrt{2}}$. In particular, we obtain

$$\begin{aligned} u_{i-1,i} &= u_{i-2,i-1} + \frac{\Delta x}{2}\left(\alpha_+ + \sqrt{2 - \alpha_-^2}\right) = u_{i-3,i-2} + \frac{2\Delta x}{2}\left(\alpha_+ + \sqrt{2 - \alpha_-^2}\right) \\ &= \cdots = u_{1,2} + \frac{(i-2)\Delta x}{2}\left(\alpha_+ + \sqrt{2 - \alpha_-^2}\right) \end{aligned}$$

and

$$\begin{aligned} u_{i,i} &= u_{i-1,i} + \frac{\Delta x}{\sqrt{2}} = u_{1,2} + \frac{(i-2)\Delta x}{2}\left(\alpha_+ + \sqrt{2 - \alpha_-^2}\right) + \frac{\Delta x}{\sqrt{2}} \\ &= \alpha_+ \Delta x + \frac{(i-2)\Delta x}{2}\left(\alpha_+ + \sqrt{2 - \alpha_-^2}\right). \end{aligned}$$

Now, we take $\Delta x = N^{-1}$ and consider the numerical solution at the location $(x, y) = (x_N, y_N) = (1, 1)$. This implies

$$u_{N,N} = \frac{\alpha_+}{N} + \frac{N-2}{2N}\left(\alpha_+ + \sqrt{2 - \alpha_-^2}\right).$$

We have noticed, however, that this solution converges to $\frac{1}{2}\left(\alpha_+ + \sqrt{2 - \alpha_-^2}\right)$ but not to the exact solution $\sqrt{2}$ as N goes to infinity. This simple example shows that a trivial embedding method coupled with the Godunov fast sweeping method might fail to converge to the exact solution as $\Delta x \to 0$ when choosing a tube radius $h = O(\Delta x)$.

In developing a fast method for solving the eikonal equation, it is extremely important to take care of the causality by considering the characteristic direction. The trivial embedding method developed above, however, does not provide correct information for the update formula near the boundary $\partial\Gamma_h$, as shown in Fig. 2. Consider the numerical value of a grid point $u_{i,j}$ which lies inside the computational domain $(x_i, y_j) \in \Gamma_h$ but adjacent to the boundary of the computational tube $\partial\Gamma_h$, as shown in blue in Fig. 2. The corresponding update formula

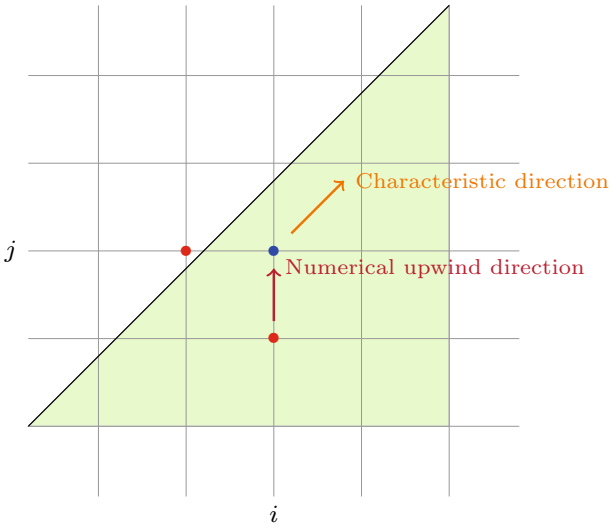

Fig. 2 The numerical upwind direction and the real characteristic direction near a boundary grid point

depends on the values at two adjacent grid points, marked in red. However, since $u_{i-1,j}$ lies outside the computation tube, only $u_{i,j-1}$ will be used. This implies that there is always a mismatch in the numerical upwind direction and the real characteristic direction at a grid point near the computational boundary, and this mismatch does not vanish as we decrease Δx. This is the main reason why the method does not converge to the correct solution.

In Fig. 3a, we have plotted several numerical solutions (in blue circles and blue solid lines) to the eikonal equation using various tube radius given by $h = O(\Delta x)$. The top blue data are obtained using the tube radius $h = 5\Delta x/\sqrt{8}$, which has been analyzed above. We clearly see that the numerical solution converges to $\frac{1}{2}\left(\alpha_+ + \sqrt{2-\alpha_-^2}\right)$, represented by the red dotted line on the top, as $\Delta x \to 0$. Using a larger ratio $h/\Delta x$ as we refine the mesh, the numerical solution does converge to a value closer to the exact solution $\sqrt{2}$, as demonstrated by different blue solid curves in the figure. Nevertheless, for any finite fixed ratio $h/\Delta x$, the approach proposed earlier in this section does not converge to the exact solution.

On the other hand, the numerical solution seems to converge to the exact solution if we have $\Delta x \to 0$ and $h/\Delta x \to \infty$ at the same time, as shown in Fig. 3b. Therefore, one possible attempt to fix this simple fast sweeping scheme is to follow a similar approach as in [21] by choosing the tube radius $h = O(\Delta x^\gamma)$ for some $\gamma \in (0, 1)$. In such a choice, the computation tube will shrink in a much smaller speed as we reduce the mesh size Δx. Their original intention is to increase the resolution of the numerical solution when approximating the Euclidean weighted distance function by having $h \to 0$ and $\frac{h}{\Delta x} \to \infty$ at the same time. We have preformed numerical experiments on the above example by choosing $\gamma = 0.8$ and $\gamma = 0.7$, respectively, and we have checked the numerical error in the numerical solution at the point $(x, y) = (1, 1)$. We have found that the solutions from these two cases seem to converge to the exact solution, but the smaller the value of γ the better the numerical convergence, as shown in Fig. 3a, c. In particular, for a relatively larger γ (=0.8), we found that the numerical solution converges slower than $O(\sqrt{\Delta x})$. For a smaller γ (=0.7), the convergence is faster than $O(\sqrt{\Delta x})$.

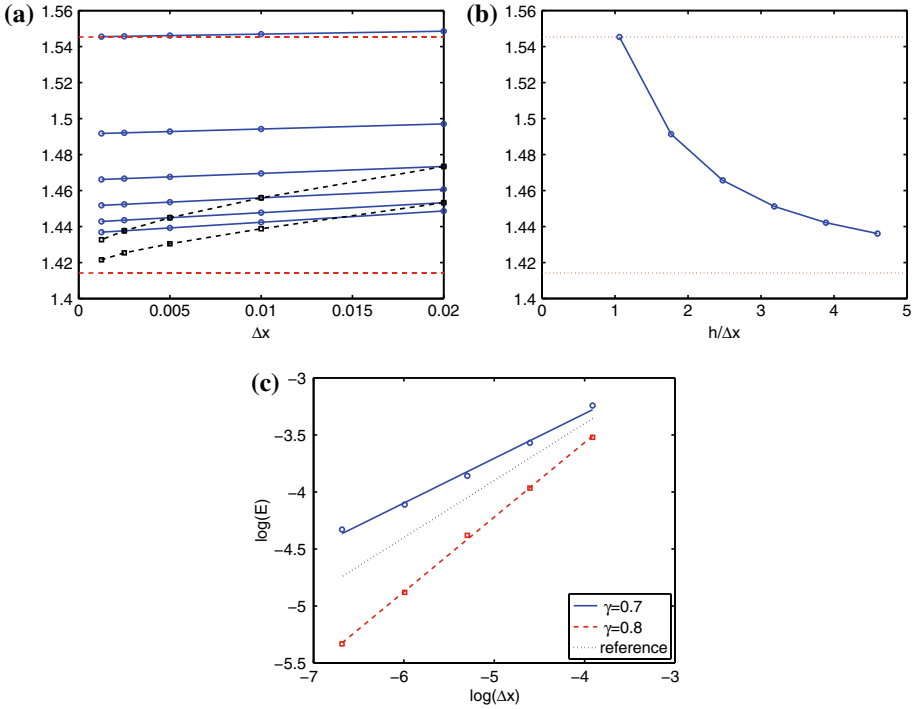

Fig. 3 Numerical solutions at $(x, y) = (1, 1)$ on different Δx and h. The exact solution $\sqrt{2}$ and $\frac{1}{2}\left(\alpha_+ + \sqrt{2-\alpha_-^2}\right)$ are plotted using the *bottom* and the *top red dotted lines* on both figures, respectively. **a** *Blue solid lines with circles* (from *top* to *bottom*) represent the numerical solutions from different Δx using a fix ratio of $h/\Delta x$ changing from $2.5/\sqrt{2}$ to $7.5/\sqrt{2}$ in an increment of $1/\sqrt{2}$. The *black dashed line with squares* represents the numerical solutions from different Δx using $h = 1.5(\Delta x)^\gamma$ for $\gamma = 0.7$ and 0.8. **b** Numerical solution as $\Delta x \to 0$ while keeping $h/\Delta x$ fixed. **c** Error at the location $(x, y) = (1, 1)$ versus the mesh size with $h = O(\Delta x^\gamma)$ and $\gamma = 0.8$ and $\gamma = 0.7$, respectively. A *black dotted reference line* has slope 0.5. The case with $\gamma = 0.7$ has a better behavior with the slope of the least squares fitting line 0.6546 (Color figure online)

Such improvement in the numerical solution can be partially explained by the constructions in the explicit numerical solution using $h = O(\Delta x)$ as demonstrated above. The smaller value of γ, the greater number of grid points lies inside the tube. As argued, the boundary grid points introduce problematic information in the characteristic direction. The further away (relative to Δx) the boundary to the interface Σ, the more the error gets diluted. But we again emphasize that it is not clear how small should γ be used in order to yield good numerical results. Also, since the number of mesh points across the computational tube is $O(h/\Delta x) = O(\Delta x^{\gamma-1})$, the total number of mesh points within Γ_h is therefore

$$O(\Delta x^{\gamma-1}) \cdot O(\Delta x^{1-d}) = O(\Delta x^{\gamma-d}).$$

As a result, this approach has a heavy trade-off between the accuracy in the solution and the overall computational cost. In the next section, we are going to propose a simple numerical treatment by incorporating the true characteristic direction to the update formula, while keeping the radius of the computational tube $h = O(\Delta x)$.

4 Our Proposed Approach

Our proposed approach is motivated by the fact that the scheme has to take care of the correct characteristic direction even for those grid points adjacent to the boundary of the computational tube. This implies that instead of keeping $u_{i,j}$ as the numerical infinity for all $(x_i, y_j) \notin \Gamma_h$, one has to also update the solution in the exterior of Γ_h if the grid location is adjacent to the computational tube.

To design a correct condition at those locations, we recall that the viscosity solution to the surface eikonal equation $|\nabla_\Sigma u| = 1$ on an implicit surface can be seen as the intrinsic distance function of the surface itself. At each point on the surface, it is geometrically clear that the surface gradient $\nabla_\Sigma u$ of the intrinsic distance function must lie on the tangent plane and it implies that the surface gradient should be orthogonal to the normal of the surface. If the level set function ϕ is the signed distance representation of the surface given by $\{\phi = 0\}$ with $|\nabla\phi| = 1$, such condition can be expressed as

$$\mathbf{n} \cdot \nabla_\Sigma u = 0 \text{ on } \Sigma \tag{6}$$

where $\mathbf{n} = \nabla\phi$ is the unit normal defined on the interface Σ. Our approach uses this expression to impose an extra boundary value condition for Γ_h. Assuming that the solution to the surface eikonal equation is not only given on Σ but also *all* level sets given by $\{|\phi| < h\}$, we have a natural extension of the solution to outside the tube using

$$\mathbf{n} \cdot \nabla u = 0. \tag{7}$$

Mathematically, this means that the function u has to be constant along the normal direction to the boundary and, therefore, such condition imposes simply a constant extrapolation of u away from Γ_h.

In this work, we propose to extend the solution to eikonal equation within Γ_h by solving (7) outside the computational tube. One simple way to implement this extension idea is to introduce a time variable and solve the following linear advection equation in the computational domain except the tube, up to steady state

$$u_t + \text{sgn}(\phi)\mathbf{n} \cdot \nabla u = 0 \text{ on } \Omega - \Sigma_h$$

where $\text{sgn}(x)$ is the signum function. Instead, we can also consider the time-independent version by solving

$$\big[\text{sgn}(\phi)\mathbf{n}\big] \cdot \nabla u = 0 \text{ on } \Omega - \Sigma_h \tag{8}$$

where the extra signum function takes care of the upwind direction in the characteristics so that all information will flow outward from the interface to the rest of the computational domain.

Recall that we require only a boundary condition at grid point adjacent to the computational tube, much computational efforts are wasted if this extension equation is solved on $\Omega - \Sigma_h$. The first step in our approach is to introduce a second layer of computational tube, as shown in Fig. 4. The inner computation layer (denoted by Γ^{in} with radius h^{in}) is the same as our previous computation tube in Sect. 3 where we impose the eikonal equation, while the outer layer is an extension layer (denoted by Γ^{out} with width h^{out}) in which we impose the constrained boundary condition (8). Numerically, the equation in each individual domain Γ^{in} and Γ^{out} can be efficiently solved by fast sweeping methods. The inner layer can be handled just like what we have discussed in Sect. 3. When we update $u_{i,j}$ in the outer layer Γ^{out} based on the advection equation (8), we apply the fast sweeping updating formula (5) by considering

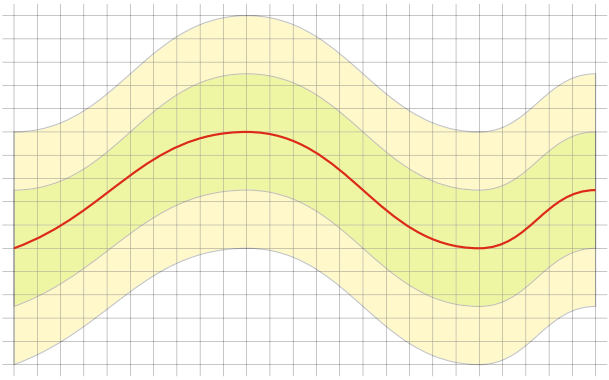

Fig. 4 The computational tube Γ consists of two separate layers, denoted by Γ^{in} and Γ^{out}. The inner layer Γ^{in} for the eikonal equation is represented by *green*, while the outer layer Γ^{out} for the extension equation is highlighted by *yellow* (Color figure online)

$$p_{i,j} = [\mathrm{sgn}(\phi)\phi_x]_{i,j} = \frac{\mathrm{sgn}(\phi_{i,j})}{2\Delta x}\left(\phi_{i+1,j} - \phi_{i-1,j}\right)$$
$$q_{i,j} = [\mathrm{sgn}(\phi)\phi_y]_{i,j} = \frac{\mathrm{sgn}(\phi_{i,j})}{2\Delta x}\left(\phi_{i,j+1} - \phi_{i,j-1}\right).$$

Since we require that Δx is small enough to resolve the interface, the level set function ϕ is differentiable in the computational tube so that it is accurate to approximate the derivatives using the central difference. The algorithm can be easily implemented and we have summarized in Algorithm 3. If high order extrapolation is necessary, one can apply the PDE based extrapolation approach proposed in [1,2].

Algorithm 3: A simple fast sweeping method for the surface eikonal equation $|\nabla_\Sigma u| = 1$.

Data: The source location (x_s, y_s), the mesh size Δx, the level set representation of the surface $\phi_{i,j}$, the inner tube radius h^{in} and the outer tube radius h^{out}.
Result: $u_{i,j}$ in the computational tube.
Initialization: Set $u_{i,j} = 0$ if (x_i, y_j) is at the point source, otherwise $u_{i,j} = \infty$;
while *not converges* **do**
  for *each of the four sweeping directions* **do**
    if $|\phi_{i,j}| \le h^{in}$ *and* $u_{i,j} \neq 0$ **then**
      update $u_{i,j}$ using (3) for the Godunov Hamiltonian or (4) for the LxF Hamiltonian;
    else if $h^{in} < |\phi_{i,j}| \le h^{out}$ **then**
      update $u_{i,j}$ using (5) with $\mathbf{v} = (p, q) = \mathrm{sgn}(\phi)\mathbf{n}$;
    end
  end
end

Now, all grid points in the inner computational tube have correct characteristic information for updating its latest value. Returning to our previous simple example as demonstrated in Fig. 2, in particular, we find that $u_{i,j}$ can now update its value using $u_{i-1,j}$ which thus provides a better approximation to the characteristic direction. Furthermore, we also mention that constructing an extra extension layer is necessary anyway if we are using the LxF Hamiltonian instead of the Godunov Hamiltonian. In the original work [13], higher order

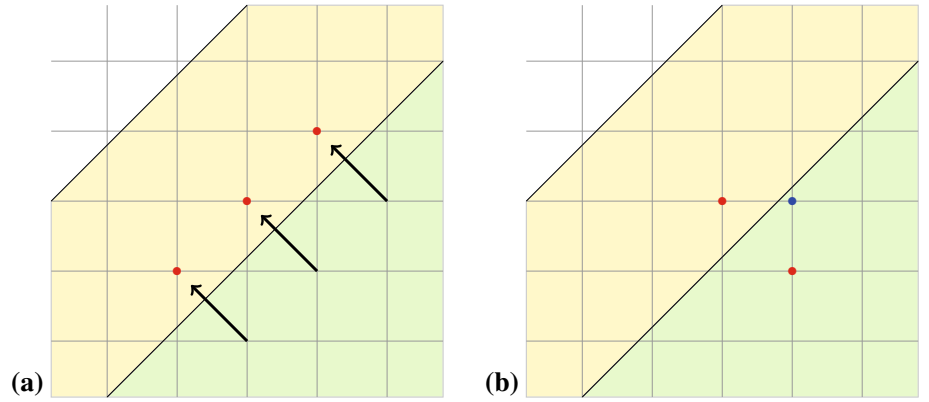

Fig. 5 **a** The numerical values in the outer layer are constantly extrapolated from those in the inner layer. The *arrows* in the figure show the unit outward normal directions, which are also the extrapolation direction. In the ideal situation, the same numerical value is found along the cross section in the outer layer. **b** The boundary point, represented by the *blue dot*, can now take the numerical value of its left neighbour point which lies in the extension layer. This extra point corrects the numerical upwind direction (Color figure online)

extrapolation technique has been coupled with a maximization–minimization condition to determine a proper value for grid locations outside the computational boundary. On the other hand, we found that a constant extrapolation condition is already good enough to provide a convergent and robust algorithm for determining the values in the outer layer (Fig. 5).

In the current approach, we are able to use a tube radius h^{in} and h^{out} both of $O(\Delta x)$. Nevertheless, the tube radius has to be wide enough for resolving both the eikonal equation in inner layer and the extension equation in outer layer. Using the same lower bound proposed in [21], both layers should have at least $h\sqrt{d}$ in their width which makes a lower bound of the whole tube h^{out} to be $\sqrt{d}\Delta x + \sqrt{d}\Delta x = 2\sqrt{d}\Delta x$. In our numerical experiments, we choose $2d\Delta x$. Even though such choice is slightly larger than the theoretical lower bound, the total number of grid points increases only linearly as $\Delta x \to 0$.

In the following, we prove the numerical convergence of the LxF fast sweeping method on the two dimension eikonal equations $|\nabla u| = R$, where $R = R(\mathbf{x})$ is some positive function.

Proposition 4.1 *Suppose that the initial values of numerical solution are all set to be positive in both the inner and the outer layers (except the source points). Then the numerical solution always stays positive.*

Proof We split the discussion into two cases. First, we consider a location (x_i, y_j) in the inner layer. Take $\Delta x = \Delta y$, $\sigma_x = \sigma_y = 1$, the Hamiltonian for the eikonal equation is $H(p, q) = \sqrt{p^2 + q^2}$. The LxF fast sweeping has the following update formula,

$$u^*_{i,j} = \frac{R_{i,j}\Delta x}{2} + \frac{1}{4}\Big[u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - \sqrt{(u_{i+1,j} - u_{i-1,j})^2 + (u_{i,j+1} - u_{i,j-1})^2}\Big]$$

It is clear that the sign of $u^*_{i,j}$ solely depends on the second term of the above expression. Note that

$$\begin{aligned}&(u_{i+1,j}+u_{i-1,j}+u_{i,j+1}+u_{i,j-1})^2-\left[\sqrt{(u_{i+1,j}-u_{i-1,j})^2+(u_{i,j+1}-u_{i,j-1})^2}\right]^2\\&\quad=4u_{i+1,j}u_{i-1,j}+2u_{i+1,j}u_{i,j+1}+2u_{i+1,j}u_{i,j-1}+2u_{i-1,j}u_{i,j+1}\\&\qquad+2u_{i-1,j}u_{i,j-1}+4u_{i,j+1}u_{i,j-1}.\end{aligned}$$

The second term will always keep its positive sign if the values of $u_{i\pm1,j}$ and $u_{i,j\pm1}$ are all positive. So if the numerical solution starts with the positive initial values everywhere, the value of $u^*_{i,j}$ always stays positive at the following iterations.

If (x_i, y_j) is in the outer layer, we have

$$u^*_{i,j}=\frac{p^+u_{i-1,j}-p^-u_{i+1,j}+q^+u_{i,j-1}-q^-u_{i,j+1}}{|p|+|q|},$$

where (p, q) is the unit outward normal vector at (x_i, y_j), and $p^\pm := (p\pm|p|)/2$. Since, $u^*_{i,j}$ is just a convex combination of $u_{i\pm1,j}$ and $u_{i,j\pm1}$ with $p^+, -p^-, q^+, -q^- \geq 0$, $u^*_{i,j}$ stays positive as long as $u_{i\pm1,j}$ and $u_{i,j\pm1}$ do so. Its positiveness again inherits from the initial values of the numerical solution. □

We recall a simple fact in analysis, a decreasing and bounded below sequence converges. We then have the following corollary.

Corollary 1 *The LxF fast sweeping method on eikonal equations $|\nabla\phi| = R(x)$ converges numerically.*

Proof Recall that we impose large values to the initial guess in the fast sweeping algorithm, the numerical solution based on the Godunov Hamiltonian is guaranteed to be decreasing. Even though there is no such property in the LxF Hamiltonian, on the other hand, we can explicitly impose this constraint in the iteration by updating the numerical solution only when the updated value is less than its old value. Now, since the numerical solution always approximates to the correct viscosity solution from above, the numerical solution at each grid point forms a decreasing sequence. And, because each of these sequences is bounded below by zero based on the above proposition, they converge as the iteration goes. □

It does not seems to be trivial to rigorously prove that the numerical solution on Σ converges to the viscosity solution to the surface eikonal equation as Δx goes to zero. Intuitively, as explained earlier, one has to provide the correct characteristic information in designing a local update formula. For the surface eikonal equation, such information might come from outside the inner computational domain. The extension step we proposed does give a way to bring the necessary information to the inner tube. To prove that the numerical solution indeed converges to the viscosity solution, however, one difficulty is that the computational domain depends explicitly on Δx. Also, one has to consider at the same time the coupling of the extension equation in the outer layer and the numerical solution to the eikonal equation in the inner tube. Nevertheless, the numerical examples in Sect. 6 have shown that the proposed numerical scheme does converge to the correct viscosity solution without any difficulty.

5 Extensions

The proposed approach can be easily extended to various equations and on general implicit surfaces. In this section, we discuss various straight-forward extensions and applications.

5.1 General Static HJ Equations on Surfaces

The proposed approach can be easily extended to other static HJ equations on surfaces. Suppose that we are given a HJ equation

$$H(\nabla_\Sigma u) = 0,$$

defined only on the surface Σ. Following the same idea in the previous section, we replace the intrinsic gradient $\nabla_\Sigma u$ directly by the extrinsic gradient ∇u and consider the equation

$$H(\nabla u) = H(u_x, u_y) = 0$$

in a small neighborhood of the surface Σ. Numerically, we first construct a two-layered computational tube that encloses the surface Σ. In the inner layer Γ^{in}, we solve the HJ equation $H(u_x, u_y) = 0$ by replacing the Hamiltonian with the LxF Hamiltonian

$$H\left(\frac{D_x^+ u_{i,j} + D_x^- u_{i,j}}{2}, \frac{D_y^+ u_{i,j} + D_y^- u_{i,j}}{2}\right) - \sigma_x \left(\frac{D_x^+ u_{i,j} - D_x^- u_{i,j}}{2}\right) - \sigma_y \left(\frac{D_y^+ u_{i,j} - D_y^- u_{i,j}}{2}\right),$$

where

$$\sigma_x \geq \max \left|\frac{\partial H}{\partial u_x}\right| \quad \text{and} \quad \sigma_y \geq \max \left|\frac{\partial H}{\partial u_y}\right|$$

and $D_x^\pm$ and $D_y^\pm$ are the forward and the backward difference operators along the x- and the y-direction, respectively. Numerically, we can follow the same approach as in Algorithm 3 but only need to replace the iterative update formula at each grid point (x_i, y_j) within Γ^{in} by

$$u_{i,j} \leftarrow \left[\frac{\sigma_x}{\Delta x} + \frac{\sigma_y}{\Delta y}\right]^{-1} \left[\sigma_x \frac{u_{i+1,j} + u_{i-1,j}}{2\Delta x} + \sigma_y \frac{u_{i,j+1} + u_{i,j-1}}{2\Delta y} - H\left(\frac{u_{i+1,j} - u_{i-1,j}}{2\Delta x}, \frac{u_{i,j+1} - u_{i,j-1}}{2\Delta y}\right)\right].$$

In the outer layer Γ^{out}, we keep the typical extension equation as before.

5.2 Implicit Surfaces in the Closest Point Representation

In above sections, all implicit surfaces are represented by the zero level set of some signed distance functions. Some implicit surfaces, however, cannot be defined using any signed distance function. Typical examples include open surfaces and non-orientable surfaces. Nevertheless, if an implicit surface is not pathological, it usually admits a closest point representation in an open set that contains the implicit surface. It is, therefore, not difficult to extend our proposed algorithm to surfaces with the closest point representation. For a point $\mathbf{x}$ sufficiently close to a given surface Σ, it is possible to find the unique projection of $\mathbf{x}$ on Σ which gives the shortest Euclidean distance. Following the same notation as in the closest point method [19,20,31], we denote this unique projection $cp(\mathbf{x})$, representing the closest point on Σ to $\mathbf{x}$. Then the surface Σ can be represented by the zero level set of the function $F(\mathbf{x}) := |\mathbf{x} - cp(\mathbf{x})|$. Using this function, we can similarly define the inner and outer layer by $\Gamma^{in} = \{F(\mathbf{x}) \leq h^{in}\}$ and $\Gamma^{out} = \{h^{in} < F(\mathbf{x}) \leq h^{out}\}$, respectively.

Unlike the signed distance function as in the level set method, unit outward normal may not be well defined in surfaces with the closest point representation. For example, non-orientable surfaces like the Mobius band do not have any consistent outward normal at any point. It seems to cause trouble in our constant outward extrapolation in Γ^{out}. But we do not actually need the consistent outward normal on the surface everywhere. For a point $\mathbf{x}$ in Γ^{out}, all we need is to extrapolate the numerical value at $\mathbf{x}$ away the surface orthogonally. And this can be done by extrapolate along the unit vector

$$\text{sgn}(\phi)\mathbf{n}(\mathbf{x}) = \frac{\mathbf{x} - cp(\mathbf{x})}{|\mathbf{x} - cp(\mathbf{x})|}.$$

Since the closest point $cp(\mathbf{x})$ has the shortest distance to $\mathbf{x}$ on Σ, this can only happen iff $\mathbf{x} - cp(\mathbf{x})$ is orthogonal to the tangent plane at $cp(\mathbf{x})$. So the unit vector $\mathbf{n}(\mathbf{x})$ is orthogonal to Σ and pointing away from it.

6 Numerical Examples

6.1 Eikonal Equations

In this section, we first investigate the convergence behavior of the proposed algorithm by solving the eikonal equation $|\nabla u| = 1$ on some simple two-dimensional and three-dimensional implicit surfaces for which the exact solutions can be analytically found. Then we extend the approach to various complicated surfaces to demonstrate the effectiveness of the method.

In this first example, we consider solving the surface eikonal equation on the six-folded star-shape curve given by

$$r = r_0 + \epsilon \sin^2(3\pi\theta),$$

with $r_0 = 0.5, \epsilon = 0.2$ and the point source boundary condition $u = 0$ at $(x, y) = (0.5, 0)$, as shown in Fig. 6a. The exact solution at a given point on the curve is simply the arclength from the source point. The infinity norm error on the solution using various meshes are plotted in

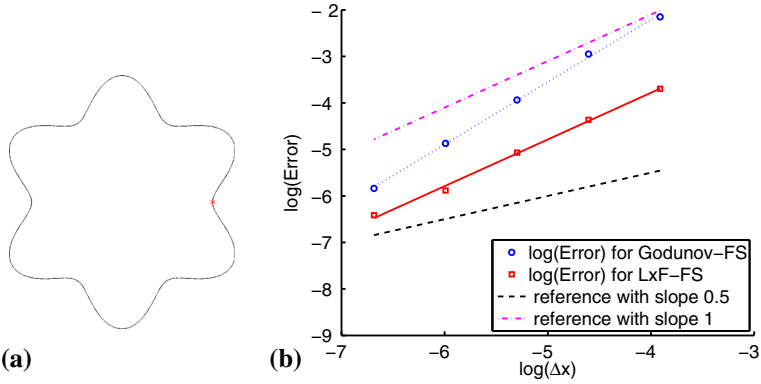

Fig. 6 (A six-folded star shaped curve) **a** The set up of the example. The point source is located at (0.5, 0) plotted using a *red dot*. **b** The slope of both linear fitting lines is clearly greater 1. The corresponding convergence order using the Godunov and the LxF Hamiltonians are 1.3413 and 1.0041, respectively (Color figure online)

Table 1 (6-Folded star shaped curve) Errors ($\times 10^{-2}$) in the numerical solution on various meshes

N	101	201	401	801	1601
(a)	11.6423	5.2356	1.9493	0.7665	0.2914
(b)	2.4838	1.2756	0.6288	0.2785	0.1638

N denotes the number of grid points in each physical direction
(a) Godunov–FS, (b) LxF–FS

Table 2 (6-Folded star shaped curve) The number of sweepings required to obtain the numerical solution

N	101	201	401	801	1601
(a)	22	34	54	91	161
(b)	58	96	166	301	559

(a) Godunov–FS, (b) LxF–FS

Fig. 6b and are shown in Table 1. In Fig. 6b, we have plotted also two reference curves with slope equals 0.5 and 1, respectively. Even though the solutions from the LxF fast sweeping is in general more accurate than that using the Godunov fast sweeping, it is clear that our proposed approach using any of these numerical Hamiltonian converges to the exact solution with an order of approximately 1.

In Table 2, we summarize the number of sweepings required to reach the converged solutions using the Godunov hamiltonian and the LxF hamiltonian, respectively. The total number of iterations for any of two numerical Hamiltonians is approximately linear (0.7187 and 0.8213 for two Hamiltonians respectively) in the number of grid points in each physical direction. We have the following observations. Unlike the Godunov–FS scheme for the eikonal equation (1) which can be shown to converge in 2^d iterations in $\mathbb{R}^d$, the proposed Godunov–FS method for the surface eikonal equation (2) takes more iterations to converge and the number of iterations now depends on N. This can be explained by the fact that in a part of the computational domain, we are solving an extension equation and the resulting solution is coupled with that from the eikonal equation. Therefore, the proof in, for example, Zhao [41] does not directly go through. Secondly, the LxF–FS requires a larger number of iterations than the Godunov–FS to converge. This is expected due to the excessive numerical dissipation and viscosity in the LxF hamiltonian. The same observation has been commonly reported in the fast sweeping community. Nevertheless, for a complicated Hamilton–Jacobi equation when the local solver for the Godunov hamiltonian cannot be easily constructed, the LxF hamiltonian still provides a simple strategy for designing a fast sweeping algorithm.

Next, we consider a sphere with radius of 0.5, where the radius of the inner computation tube and the extension tube are $2\Delta x$ and $4\Delta x$, respectively. The point source is set at $(-0.5, 0, 0)$, and the infinity norm error is measured along a circle where the sphere intersects with the y–z plane. The numerical errors for various mesh size are given in Table 3. Like the low dimensional case, the convergence rate is again approximately 1, as shown in Fig. 7.

We also plot some eikonal solutions on other two dimensional manifolds represented implicitly using a level set function in Fig. 8. We use the same radius in the inner layer as the sphere, i.e. $2\Delta x$. Except the Stanford bunny in which we choose a slightly larger outer tube ($5\Delta x$), all other examples use the same radius $4\Delta x$.

Table 3 (Sphere) Errors in the numerical solution to the eikonal equation by the LxF fast sweeping scheme on various meshes

N	81	101	121	141	161	181	201
Error ($\times 10^{-1}$)	1.1927	0.9795	0.8389	0.7313	0.6489	0.5853	0.5323

N denotes the number of grid points in each physical direction

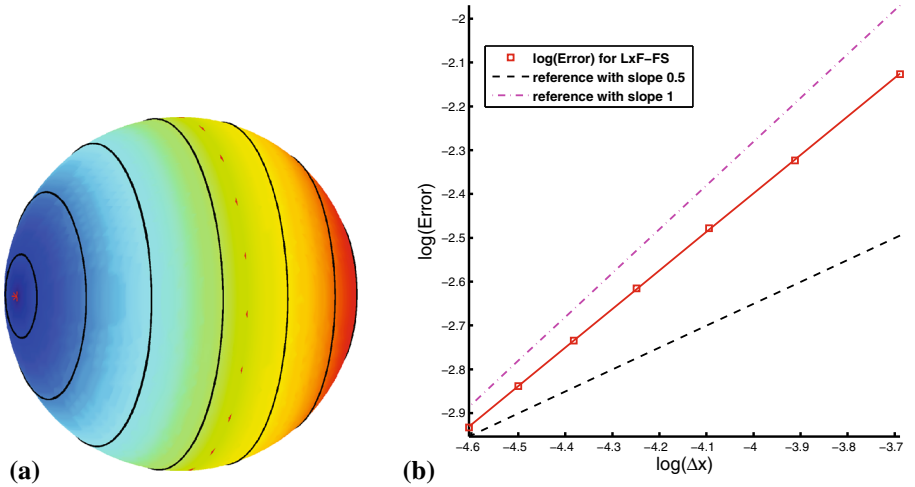

Fig. 7 (Sphere) **a** The solution to eikonal equation on the sphere. The point source and the circle where we measure the error are represented by a *red dot* and a *black dotted circle* along the sphere respectively. **b** The error in the solution to the eikonal equation on a sphere. The convergence order using the LxF fast sweeping is approximately 1 (Color figure online)

6.2 Implicit Surfaces in the Closest Point Representation

In this section, we apply our algorithm to some implicit surfaces with closest point representations. We consider the upper hemisphere and also the Mobius band. We first obtain the viscosity solution in a small neighborhood of the implicit surface defined using a Cartesian mesh, then we interpolate the solution on the corresponding triangulate surface, as shown in Fig. 9.

6.3 An Anisotropic Eikonal Equation

We extend our proposed approach to solve an anisotropic eikonal equation on a moving medium [25], given by

$$|\nabla u(\mathbf{x})| = \frac{|1 - \mathbf{v}(\mathbf{x}) \cdot \nabla u(\mathbf{x})|}{F(\mathbf{x})}.$$

The equation has a wide range of applications in geophysics [11] and atmospherical sciences like noise propagation modeling [9] or underwater acoustics. If we interpret this eikonal equation in the context of wave propagation, u is the first arrival time that the wave propagates from the point source. $\mathbf{v}(\mathbf{x})$ and $F(\mathbf{x})$ are the ambient velocity field of the surrounding moving medium and the speed of the wave propagation, respectively. Various numerical methods

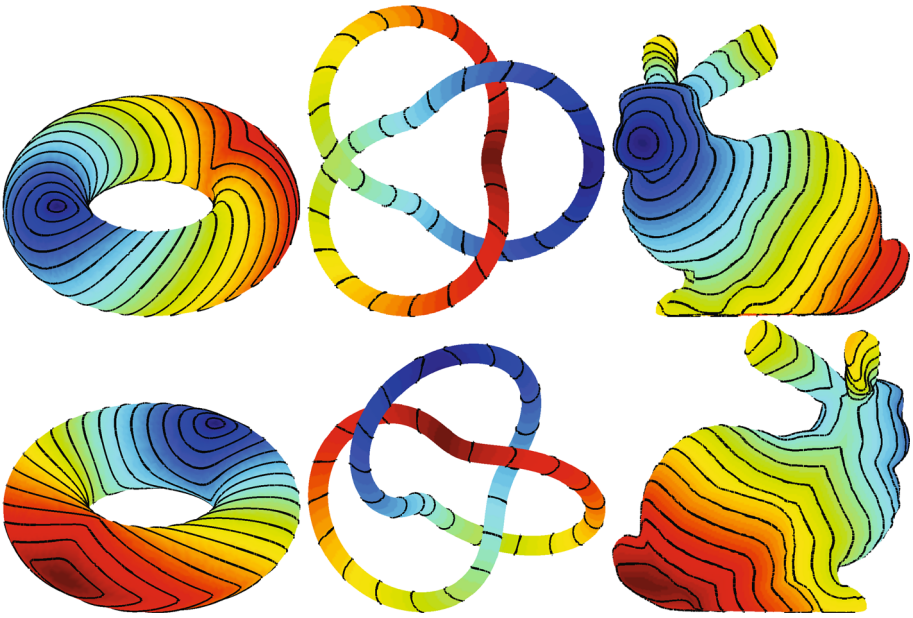

Fig. 8 Solutions to the eikonal equations on various implicit surfaces

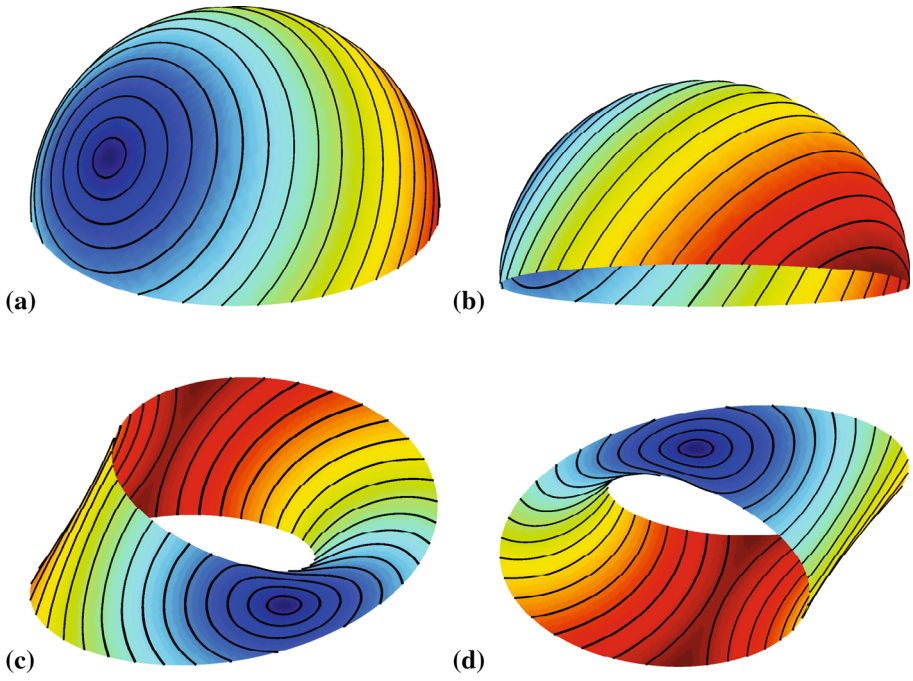

Fig. 9 The eikonal solutions on **a**, **b** upper hemisphere and **c**, **d** the Mobius band

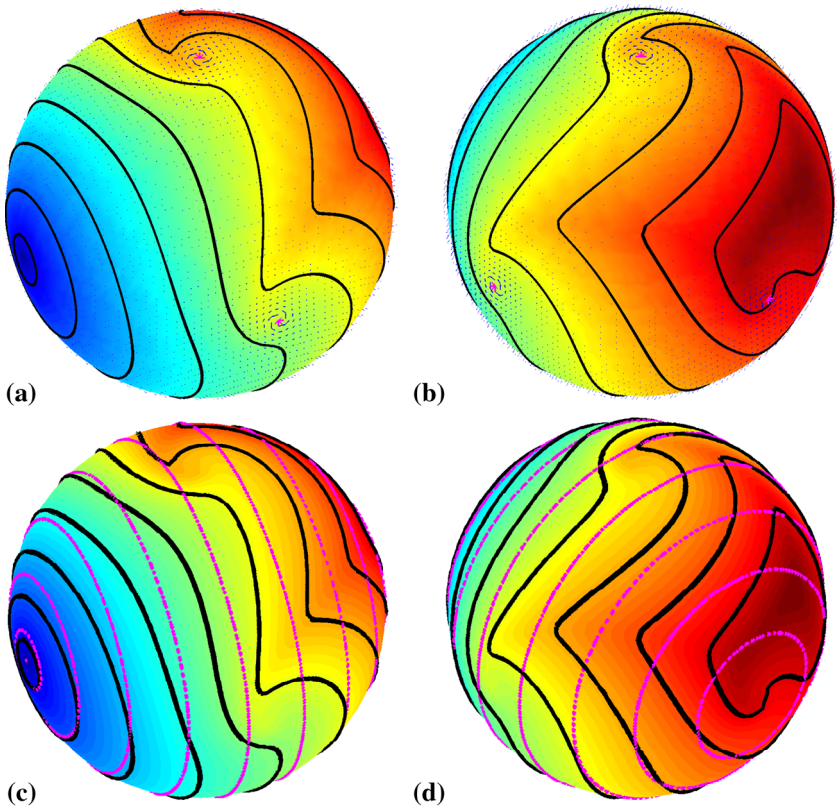

Fig. 10 The anisotropic eikonal equation in the three-vortex flow. The source point is set at (0.5, 0, 0), and the centers of vortex flow are set at (0, 0, 0.5), (0, −0.5, 0) and (0.5, 0, 0) respectively. Contour lines of our computed solution are plotted in *black solid line* on top of **a**, **b** the flow field and **c**, **d** the contour lines of the unperturbed first arrivaltime in *magenta dotted lines* (Color figure online)

might be used to solve the equation, but they are limited to a typical Euclidean space. In this part of the paper, we will solve this generalized eikonal equation on a sphere given by

$$|\nabla_\Sigma u(\mathbf{x})| = \frac{|1 - \mathbf{v}(\mathbf{x}) \cdot \nabla u(\mathbf{x})|}{F(\mathbf{x})}$$

where the ambient velocity $\mathbf{v} \in \mathbb{R}^3$ is orthogonal to the surface ($\mathbf{v} \cdot \mathbf{n} = 0$) and has a small magnitude comparing to F (such that $|\mathbf{v}| \ll F$). One simple flow field is given by the following N-point vortex flow [22,23]

$$\mathbf{v}(\mathbf{x}) = \frac{1}{2\pi} \sum_{j=1}^{N} \frac{s_j(\mathbf{x}_j \times \mathbf{x})}{|\mathbf{x}_j - \mathbf{x}|^2},$$

where $\mathbf{x}_j$ is the center of a point vortex with the index j and s_i is the corresponding strength of the vortex. In the following examples, we take $F = 3$, the radius of the sphere as 0.5, $s_j = 4$ for all j and the point source is always located at $(-0.5, 0, 0)$. Figure 10 shows our solution to the three-point vortex flow ($N = 3$) with the centers of the point vortex are located at $(0, 0, 0.5)$, $(0, -0.5, 0)$ and $(0.5, 0, 0)$, which are highlighted by red stars in

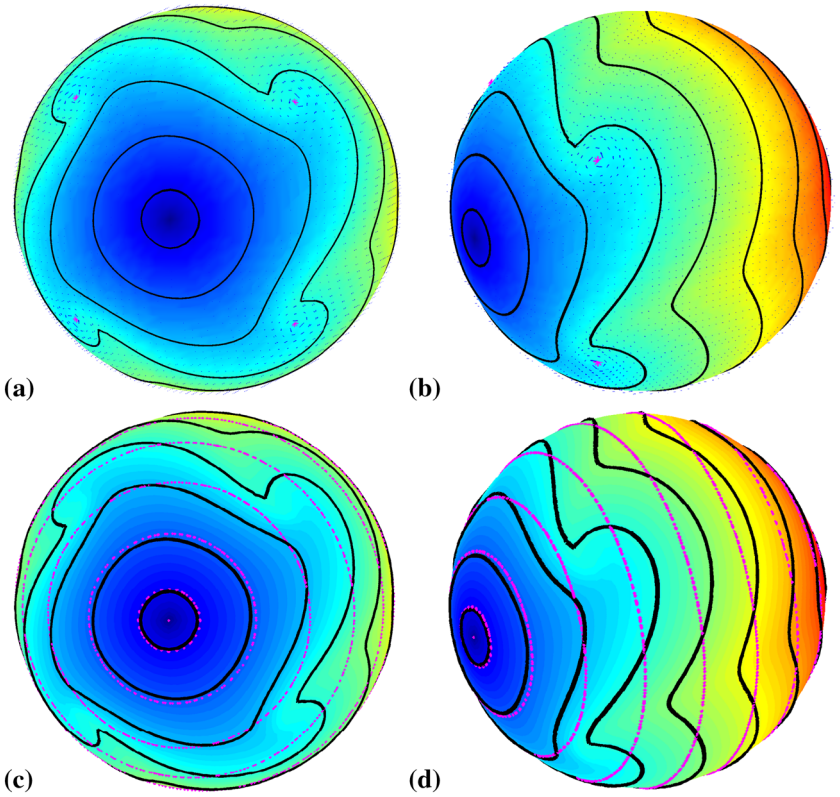

Fig. 11 The anisotropic eikonal equation in the four-vortex flow. The source point is kept at $(-0.5, 0, 0)$. Contour lines of our computed solution are plotted in *black solid line* on top of **a**, **b** the flow field and **c**, **d** the contour lines of the unperturbed first arrivaltime in *magenta dotted lines* (Color figure online)

Fig. 10a, b. We plot the contours of our computed first arrival time on sphere in black solid lines. In Fig. 11, we repeat the same example but consider having four point vortices located at $(-1, \pm 1, \pm 1)/2\sqrt{3}$. To demonstrate the effect of the moving medium, we plot also the unperturbed traveltime corresponding to $|\mathbf{v}| = 0$ in magenta dotted lines in Figs. 10 and 11c, d. We can clearly see the perturbation in the arrival time due to the ambient flow on the surface.

7 Conclusion and Future Work

In this paper, we have proposed a simple, computationally efficient and convergent numerical algorithm to solve surface eikonal equations on implicit surfaces. The approach is developed based on the embedding method so no surface triangulation is necessary. We have carefully studied a simple approach and have explained why we cannot simply solve the eikonal equation using the Godunov fast sweeping method in a small neighborhood of the implicit surface. Our proposed method only requires that the overall tube radius satisfies $h = O(\Delta x)$, which is optimal in the sense that the total number of mesh points in the computational tube is $O(\Delta x^{1-d})$ for a co-dimensional one surface in $\mathbb{R}^d$.

In the future, we plan to couple the level set method to extend the algorithm to simulate evolution of surfaces in which the motion law is modeled by the viscosity solution of a certain HJ equation. In addition, we also plan to develop higher order accurate schemes to HJ equations on implicit surfaces.

Acknowledgments The work of Leung was supported in part by the Hong Kong RGC Grants 605612 and 16303114.

References

1. Aslam, T.: A partial differential equation approach to multidimensional extrapolation. J. Comput. Phys. **193**, 349–355 (2004)
2. Aslam, T., Luo, S., Zhao, H.: A static PDE approach to multi-dimensional extrapolations using fast sweeping methods. SIAM J. Sci. Comput. **36**(6), A2907–A2928 (2014)
3. Bertalmio, M., Cheng, L.-T., Osher, S., Sapiro, G.: Variational problems and partial differential equations on implicit surfaces. J. Comput. Phys. **174**, 759–780 (2001)
4. Bronstein, A.M., Bronstein, M.M., Kimmel, R.: Weighted distance maps computation on parametric three-dimensional manifolds. J. Comput. Phys. **225**, 771–784 (2007)
5. Chen, W., Chou, C.S., Kao, C.Y.: Lax–Friedrichs fast sweeping methods for steady state problems for hyperbolic conservation laws. J. Comput. Phys. **234**, 452–471 (2013)
6. Crandall, M.G., Evans, L.C., Lions, P.L.: Some properties of viscosity solutions of Hamilton–Jacobi equations. Trans. Am. Math. Soc. **282**, 487–502 (1984)
7. Crandall, M.G., Lions, P.L.: Viscosity solutions of Hamilton–Jacobi equations. Trans. Am. Math. Soc. **277**, 1–42 (1983)
8. Drake, T., Vavylonis, D.: Model of fission yeast cell shape driven by membrane-bound growth factors and the cytoskeleton. PLoS Comput. Biol. **9**(10), e1003287 (2013)
9. Embleton, T.F.W.: Tutorial on sound prpagation outdoors. J. Acoust. Soc. Am. **100**, 31–48 (1996)
10. Grimshaw, R.: Propagation of surface waves at high frequencies. IMA J. Appl. Math. **4**(2), 174–193 (1968)
11. Hjelle, O., Petersen, S.A.: A Hamilton–Jacobi framwork for modeling folds in structural geology. Math. Geosci. **43**, 741–761 (2011)
12. Kao, C.Y., Osher, S.J., Tsai, Y.-H.: Fast sweeping method for static Hamilton–Jacobi equations. SIAM J. Num. Anal. **42**, 2612–2632 (2005)
13. Kao, C.Y., Osher, S.J., Qian, J.: Lax–Friedrichs sweeping schemes for static Hamilton–Jacobi equations. J. Comput. Phys. **196**, 367–391 (2004)
14. Kimmel, R., Sethian, J.A.: Computing geodesic paths on manifolds. Proc. Natl. Acad. Sci. USA **95**, 8431–8435 (1998)
15. Leung, S., Qian, J.: An adjoint state method for 3d transmission traveltime tomography using first arrival. Commun. Math. Sci. **4**, 249–266 (2006)
16. Li, W., Leung, S.: A fast local level set adjoint state method for first arrival transmission traveltime tomography with discontinuous slowness. Geophys. J. Int. **195**(1), 582–596 (2013)
17. Li, W.B., Leung, S., Qian, J.: A level-set adjoint-state method for crosswell transmission–reflection traveltime tomography. Geophs. J. Int. **199**(1), 348–367 (2014)
18. Liu, J., Leung, S.: A splitting algorithm for image segmentation on manifolds represented by the grid based particle method. J. Sci. Comput. **56**(2), 243–266 (2013)
19. Macdonald, C.B., Ruuth, S.J.: Level set equations on surfaces via the closest point method. J. Sci. Comput. **35**, 219–240 (2008)
20. Macdonald, C.B., Ruuth, S.J.: The implicit closest point method for the numerical solution of partial differential equations on surfaces. SIAM J. Sci. Comput. **31**, 4330–4350 (2009)
21. Memoli, F., Sapiro, G.: Fast computation of weighted distance functions and geodesics on implicit hypersurfaces. J. Comput. Phys. **173**, 730–764 (2001)
22. Newton, P.K.: The N-Vortex Problem: Analytical Techniques. Springer, Berlin (2001)
23. Newton, P.K., Ross, S.D.: Chaotic advection in the restricted four-vortex problem on a sphere. Phys. D **223**, 36–53 (2006)
24. Osher, S.J., Fedkiw, R.P.: Level Set Methods and Dynamic Implicit Surfaces. Springer, New York (2003)
25. Pierce, A.D.: Acoustics: An Introduction to Its Physical Principles and Applications. Acoustical Society of America, New York (1989)

26. Poirier, C.C., Ng, W.P., Robinson, D.N., Iglesias, P.A.: Deconvolution of the cellular force-generating subsystems that govern cytokinesis furrow ingression. PLoS Comput. Biol. **8**(4), e1002467 (2012)
27. Popovici, A.M., Sethian, J.A.: Three-dimensional traveltime computation using the fast marching method. In: 67th Annual International Meeting, Society of Exploration Geophysicists, Expanded Abstracts, pp. 1778–1781. Society of Exploration Geophysicists (1997)
28. Qian, J., Zhang, Y.-T., Zhao, H.-K.: Fast sweeping methods for eikonal equations on triangulated meshes. SIAM J. Numer. Anal. **45**, 83–107 (2007)
29. Qian, J., Zhang, Y.-T., Zhao, H.-K.: Fast sweeping methods for static Hamilton–Jacobi equations triangulated meshes. J. Sci. Comput. **31**, 237–271 (2007)
30. Rouy, E., Tourin, A.: A viscosity solutions approach to shape-from-shading. SIAM J. Numer. Anal. **29**, 867–884 (1992)
31. Ruuth, S.J., Merriman, B.: A simple embedding method for solving partial differential equations on surfaces. J. Comput. Phys. **227**, 1943–1961 (2008)
32. Sethian, J.A.: Level Set Methods. Cambridge University Press, Cambridge (1996)
33. Sethian, J.A.: Fast marching methods. SIAM Rev. **41**, 199–235 (1999)
34. Spira, A., Kimmel, R.: An efficient solution to the eikonal equation on parametric manifolds. Interface Free Bound. **6**, 315–327 (2004)
35. Tsai, R., Cheng, L.T., Osher, S., Zhao, H.K.: Fast sweeping method for a class of Hamilton–Jacobi equations. SIAM J. Numer. Anal. **41**, 673–694 (2003)
36. Tsitsiklis, J.N.: Efficient algorithms for globally optimal trajectories. IEEE Tran. Autom. Control **40**, 1528–1538 (1995)
37. Weber, O., Devir, Y.S., Bronstein, A.M., Bronstein, M.M., Kimmel, R.: Parallel algorithms for approximation of distance maps on parametric surfaces. ACM Trans. Graph. **27**, 104 (2008)
38. Xu, S.G., Zhang, Y.X., Yong, J.H.: A fast sweeping method for computing geodesics on triangular manifolds. IEEE Trans. Pattern Anal. Mach. Intell. **32**, 231–241 (2010)
39. Yoo, S.W., Seong, J.K., Sung, M.H., Shin, S.Y., Cohen, E.: A triangulation-invariant method for anisotropic geodesic map computation on surface meshes. IEEE Trans. Vis. Comput Graph. **18**, 1664–1677 (2012)
40. Zhang, Y.T., Zhao, H.K., Qian, J.: High order fast sweeping methods for static Hamilton-Jacobi equations. J. Sci. Comput. 29(1), 25–56 (2006)
41. Zhao, H.K.: Fast sweeping method for eikonal equations. Math. Comput. **74**, 603–627 (2005)