SIAM J. NUMER. ANAL.
Vol. 41, No. 1, pp. 325–363
© 2003 Society for Industrial and Applied Mathematics

ORDERED UPWIND METHODS FOR STATIC HAMILTON–JACOBI EQUATIONS: THEORY AND ALGORITHMS*

JAMES A. SETHIAN† AND ALEXANDER VLADIMIRSKY‡

Abstract. We develop a family of fast methods for approximating the solutions to a wide class of static Hamilton–Jacobi PDEs; these fast methods include both semi-Lagrangian and fully Eulerian versions. Numerical solutions to these problems are typically obtained by solving large systems of coupled nonlinear discretized equations. Our techniques, which we refer to as "Ordered Upwind Methods" (OUMs), use partial information about the characteristic directions to decouple these nonlinear systems, greatly reducing the computational labor. Our techniques are considered in the context of control-theoretic and front-propagation problems.

We begin by discussing existing OUMs, focusing on those designed for isotropic problems. We then introduce a new class of OUMs which decouple systems for general (anisotropic) problems. We prove convergence of one such scheme to the viscosity solution of the corresponding Hamilton–Jacobi PDE. Next, we introduce a set of finite-differences methods based on an analysis of the role played by anisotropy in the context of front propagation and optimal trajectory problems.

The performance of the methods is analyzed, and computational experiments are performed using test problems from computational geometry and seismology.

Key words. ordered upwind methods, fast marching methods, Dijkstra-like methods, anisotropic optimal control, dynamic programming, viscosity solution, anisotropic front propagation

AMS subject classifications. 65N12, 65N06, 49L20, 49L25, 49N90, 35F30, 35B37

PII. S0036142901392742

1. Introduction.

In this paper we present a family of noniterative methods applicable to the boundary value problem for static Hamilton–Jacobi equations of the form[1]

$$\begin{aligned} H(\nabla u, \boldsymbol{x}) &= 1, \qquad \boldsymbol{x} \in \Omega \subset R^2, \\ u(\boldsymbol{x}) &= q(\boldsymbol{x}), \quad \boldsymbol{x} \in \partial\Omega, \end{aligned} \tag{1}$$

where the Hamiltonian H is assumed to be Lipschitz-continuous, convex, and homogeneous of degree 1 in the first argument:

$$H(\nabla u, \boldsymbol{x}) = \|\nabla u\| F\left(\boldsymbol{x}, \frac{\nabla u}{\|\nabla u\|}\right) \tag{2}$$

for some function F. We will further assume that the function q is also Lipschitz-continuous, and that

$$\begin{aligned} 0 < F_1 \le F(\boldsymbol{x}, \boldsymbol{p}) \le F_2, \\ q_1 \le q(\boldsymbol{x}) \le q_2 \end{aligned}$$

*Received by the editors July 23, 2001; accepted for publication (in revised form) August 30, 2002; published electronically April 9, 2003.
http://www.siam.org/journals/sinum/41-1/39274.html

†Department of Mathematics, University of California at Berkeley, Berkeley, CA 94720 (sethian@math.berkeley.edu). The research of this author was supported in part by the Applied Mathematical Sciences, Office of Energy Research, U.S. Department of Energy, grant DE-AC03-76SF00098, and by the Division of Mathematical Sciences, National Science Foundation.

‡Department of Mathematics, Cornell University, Ithaca, NY 14853 (vlad@math.cornell.edu). The research of this author was supported in part by a National Science Foundation Postdoctoral Research Fellowship.

[1]For the sake of notational clarity we restrict our discussion to R^2; all results can be restated for R^n and for meshes on manifolds.

www.manaraa.com

for all $\boldsymbol{x} \in \Omega$ and for all the vectors of unit length $\boldsymbol{p}$.

Even for arbitrarily smooth H, q, and $\partial\Omega$, a smooth solution on Ω need not exist. In general, there are infinitely many weak Lipschitz-continuous solutions, but the unique *viscosity solution* can be defined using additional conditions on the smooth test functions (see [10, 9]).

To obtain a numerical solution, one often starts with a mesh X covering the domain Ω. Let $U_i = U(\boldsymbol{x_i})$ be the numerical solution at the mesh point $\boldsymbol{x_i} \in X$. Denote the set of mesh points adjacent to $\boldsymbol{x_i}$ as $N(\boldsymbol{x_i})$ and the set of values adjacent to U_i as $NU(\boldsymbol{x_i}) = \{U_j | \boldsymbol{x_j} \in N(\boldsymbol{x_i})\}$. Let $\overline{H}$ be a consistent discretization of H such that one can write

$$\overline{H}(U_i, NU(\boldsymbol{x_i}), \boldsymbol{x_i}) = 1. \tag{3}$$

If M is the total number of mesh points, then one needs to solve M coupled nonlinear equations simultaneously. One typical approach is to solve this nonlinear system iteratively.

Our ultimate goal is to introduce a set of "single-pass" numerical methods. By this, we mean that each U_i is recalculated at most r times, where r depends only upon the PDE (1) and the mesh structure and not upon the number of mesh points.

To construct single-pass algorithms with efficient update orderings, one can utilize the fact that the value of $u(\boldsymbol{x})$ for the first-order PDE depends only on the value of u along the characteristic(s) passing through the point $\boldsymbol{x}$. If $\boldsymbol{x_{i_1}}, \boldsymbol{x_{i_2}} \in N(\boldsymbol{x_i})$ are such that the characteristic for the mesh point $\boldsymbol{x_i}$ lies in the simplex $\boldsymbol{x_i x_{i_1} x_{i_2}}$, then it is useful to consider an *upwind* discretization of the PDE:

$$\overline{H}(U_i, U_{i_1}, U_{i_2}, \boldsymbol{x_i}) = 1. \tag{4}$$

This reduces the coupling in the system: U_i *depends* only upon U_{i_1} and U_{i_2} and not on all of the $NU(\boldsymbol{x_i})$. A recursive construction allows one to build the entire *dependency graph* for $\boldsymbol{x_i}$.

If two or more characteristics collide at the point $\boldsymbol{x}$, the solution loses smoothness. The entropy condition does not allow characteristics to be created at these collision points; hence, if $\boldsymbol{x_i}$ is far enough from these collision points, then, for a suitably chosen discretization, its dependency graph is actually a tree. If the characteristic directions of the PDE were known in advance, then the dependency-ordering of the grid points would be known as well, leading to a fully decoupled system. Formally, this construction would lead to an $O(M)$ method.

In general, characteristic directions are not known in advance due to the nonlinearity of (1). Nonetheless, single-pass methods can be devised to determine the mesh point ordering (and the characteristic directions) in the process of decoupling the system. We refer to such methods as "Ordered Upwind Methods" (OUMs) and show that they have computational complexity of $O(M \log M)$.

Since (1) can be interpreted as a description of a continuous control problem, we start in section 2 by viewing the discrete control problem and by considering Dijkstra's method as a prototype for the OUMs to be built for the continuous case.

Next, in section 3, we view the Hamilton–Jacobi PDE (1) as an anisotropic min-time optimal trajectory problem. In this control-theoretic setting, the speed of a vehicle's motion depends not only on its position, but also on the direction. The corresponding value function u is the viscosity solution of the static Hamilton–Jacobi–

www.manaraa.com

Bellman equation

$$\begin{aligned} &\max_{\boldsymbol{a}\in S_1}\{(\nabla u(\boldsymbol{x})\cdot(-\boldsymbol{a}))f(\boldsymbol{x},\boldsymbol{a})\}=1, && \boldsymbol{x}\in\Omega,\\ &u(\boldsymbol{x})=q(\boldsymbol{x}), && \boldsymbol{x}\in\partial\Omega. \end{aligned} \tag{5}$$

Here, $\boldsymbol{a}$ is the unit vector determining the direction of motion, $f(\boldsymbol{x},\boldsymbol{a})$ is the speed of motion in the direction $\boldsymbol{a}$ starting from the point $\boldsymbol{x}\in\Omega$, and $q(\boldsymbol{x})$ is the time-penalty for exiting the domain at the point $\boldsymbol{x}\in\partial\Omega$. The maximizer $\boldsymbol{a}$ corresponds to the characteristic direction for the point $\boldsymbol{x}$.

If the speed functions F and f depend only upon their first argument, both forms of the Hamilton–Jacobi PDE reduce to the Eikonal equation

$$\|\nabla u(\boldsymbol{x})\| = K(\boldsymbol{x}), \tag{6}$$

where $K(\boldsymbol{x})=\frac{1}{F(\boldsymbol{x})}=\frac{1}{f(\boldsymbol{x})}$. In this case, the characteristics of the PDE coincide with the gradient lines of its viscosity solution u. This property is the foundation for two single-pass methods for the Eikonal equation: Tsitsiklis' algorithm (1994) and Sethian's Fast Marching Method (1996), representing Dijkstra-like decoupling of a semi-Lagrangian and a fully Eulerian discretization of (1), respectively (section 4).

We then proceed to the central part of this paper. First, we show (in section 5) that neither of these single-pass Eikonal solvers can be directly applied in the general anisotropic case. Nonetheless, the underlying ideas can be used to build the OUMs which are applicable for much more general equations. Such methods hinge on two properties of the unique viscosity solution:

1. The viscosity solution $u(\boldsymbol{x})$ is strictly increasing along the characteristics of the PDE (1).

2. We can derive a precise upper bound on the maximum angle between the characteristic and the gradient of u.

In section 6, we introduce the first general OUM with computational complexity of $O(\frac{F_2}{F_1}M\log M)$ based on a semi-Lagrangian discretization; an announcement of this algorithm without details or proof was first made in [39]. The method's convergence to the viscosity solution is proven in section 7.

Next, in section 8, we reinterpret the Hamilton–Jacobi PDE (1), this time as describing an anisotropic front expansion (contraction) problem. In this context, $F(\boldsymbol{x},\boldsymbol{n})$ is interpreted as the speed of the front in the normal direction $\boldsymbol{n}$, and $\partial\Omega$ as the initial position of the front.

$$\begin{aligned} &\|\nabla u\|F\left(\boldsymbol{x},\tfrac{\nabla u}{\|\nabla u\|}\right)=1, && \boldsymbol{x}\in\Omega,\\ &u(\boldsymbol{x})=0, && \boldsymbol{x}\in\partial\Omega. \end{aligned}$$

The anisotropy is the result of the dependence of F on $\boldsymbol{n}$. The level sets of the viscosity solution u correspond to the positions of the front at different times. We consider only those front expansion (contraction) problems in which the speed F is such that the Hamiltonian H is convex.

The fully Eulerian OUMs, introduced in section 8, use the finite-difference approximations developed as a generalization of the Fast Marching Method and are based on the analysis of the role played by anisotropy in the front propagation and optimal trajectory problems. These single-pass methods also have the same computational complexity of $O(\frac{F_2}{F_1}M\log M)$. The appendix examines the relationship between the first-order semi-Lagrangian and Eulerian OUMs.

Finally, in section 9, we analyze the efficiency of the new methods and consider several anisotropic test problems from computational geometry and seismology.

www.manaraa.com

2. Dijkstra's method and discrete optimal trajectories. We begin by considering the problem of computing the shortest path on a network. (See, for example, [3] for a catalogue of available algorithms.) Computing the shortest path can be viewed as a discrete dynamic programming problem. Here, it serves as a simpler analogue for the continuous optimal trajectory problem considered in the next section.

For the case in which the network is sparsely connected and all arc-costs are positive, the heap-sort version of Dijkstra's method [12] is one of the most widely used algorithms. We will now reinterpret Dijkstra's method as a single-pass OUM since it serves as a model for building the OUMs for the continuous front propagation and optimal trajectory problems.

2.1. Shortest paths and value function. Consider a discrete network of nodes $X = \{\boldsymbol{x_1}, \dots, \boldsymbol{x_M}\}$. A vehicle starts somewhere in the network and travels from node to node until it reaches one of the *exit nodes* $\boldsymbol{x} \in Q \subset X$. A vehicle's trajectory is a sequence of nodes $(\boldsymbol{y_1}, \dots, \boldsymbol{y_r})$ such that $\boldsymbol{y_r} \in Q$ and $\boldsymbol{y_k} \notin Q$ for $k < r$. There is a *time-penalty* $K(\boldsymbol{x_i}, \boldsymbol{x_j}) = K_{ij} > 0$ for passing from $\boldsymbol{x_i}$ to $\boldsymbol{x_j}$. ($K_{ij} = +\infty$ if there is no link from $\boldsymbol{x_i}$ to $\boldsymbol{x_j}$.) For all $\boldsymbol{x} \in Q$ there is an *exit time-penalty* $q(\boldsymbol{x}) < \infty$. Thus, the total time needed for a trajectory $(\boldsymbol{y_1}, \dots, \boldsymbol{y_r})$ is

$$\text{TotalTime}(\boldsymbol{y_1}, \dots, \boldsymbol{y_r}) = \sum_{j=1}^{r} K(\boldsymbol{y_j}, \boldsymbol{y_{j+1}}) + q(\boldsymbol{y_r}). \tag{7}$$

The goal is to find the optimal trajectory for each node $\boldsymbol{x} \in X \backslash Q$.

The key idea of *dynamic programming* [5, 6] is to solve for all of the nodes at once. Instead of searching for a particular optimal trajectory, one derives an equation for the *value function* $U(\boldsymbol{x})$, defined as the minimum time to exit the network if one starts at $\boldsymbol{x}$:

$$\begin{cases} U(\boldsymbol{x}) = \min\limits_{\substack{\text{all the paths} \\ \text{starting at } \boldsymbol{x}}} \text{TotalTime}(\boldsymbol{x}, \dots), & \boldsymbol{x} \in X \backslash Q, \\ U(\boldsymbol{x}) = q(\boldsymbol{x}), & \boldsymbol{x} \in Q. \end{cases} \tag{8}$$

Bellman's optimality principle [5] shows the relationship between $U(\boldsymbol{x})$ and the values of U on the set of adjacent nodes $N(\boldsymbol{x}) = \{\boldsymbol{y} \in X \mid K(\boldsymbol{x}, \boldsymbol{y}) < \infty\}$, namely,

$$U(\boldsymbol{x}) = \min_{\boldsymbol{y} \in N(\boldsymbol{x})} \{K(\boldsymbol{x}, \boldsymbol{y}) + U(\boldsymbol{y})\} \quad \text{for all } \boldsymbol{x} \in X \backslash Q. \tag{9}$$

Equation (9) is nonlinear, and it has to hold for each node in $X \backslash Q$. Thus, if there are M such nodes, we have to solve a coupled system of M nonlinear equations.

2.2. Dijkstra's method. Dijkstra's method [12] provides a way of decoupling system (9) and is based on the following monotonicity observations.

OBSERVATION 2.1. *If* $(\boldsymbol{y_1}, \dots, \boldsymbol{y_r})$ *is an optimal trajectory for* $\boldsymbol{y_1}$, *then we have* $U(\boldsymbol{y_1}) > \cdots > U(\boldsymbol{y_r})$.

OBSERVATION 2.2. *If* $N_-(\boldsymbol{x}) = \{\boldsymbol{y} \in N(\boldsymbol{x}) \mid U(\boldsymbol{y}) < U(\boldsymbol{x})\}$, *then Bellman's equation* (9) *can be rewritten as*

$$U(\boldsymbol{x}) = \min_{\boldsymbol{y} \in N_-(\boldsymbol{x})} \{K(\boldsymbol{x}, \boldsymbol{y}) + U(\boldsymbol{y})\} \quad \textit{for all } \boldsymbol{x} \in X \backslash Q. \tag{10}$$

If the nodes were somehow sorted by the value of U, one could solve equations (10) one by one, yielding a method with an overall complexity of $O(M)$. Even though

www.manaraa.com

this ordering on X is not known in advance, Dijkstra's method reconstructs it (one node at a time) as follows.

All the nodes are divided into three classes: *Far* (no information about the correct value of U is known), *Accepted* (the correct value of U has been computed), and *Considered* (adjacent to *Accepted*). For every *Considered* $\boldsymbol{x}$ we define the set $\mathrm{NF}(\boldsymbol{x}) = \{\boldsymbol{y} \in N(\boldsymbol{x}) \,|\boldsymbol{y} \text{ is } \textit{Accepted}\}$.

1. Start with all the nodes in *Far*.
2. Move the exit nodes ($\boldsymbol{y} \in Q$) to *Accepted* ($U(\boldsymbol{y}) = q(\boldsymbol{y})$).
3. Move all the nodes $\boldsymbol{x}$ adjacent to the boundary into *Considered* and evaluate the tentative values

$$V(\boldsymbol{x}) := \min_{\boldsymbol{y} \in \mathrm{NF}(\boldsymbol{x})} \{K(\boldsymbol{x}, \boldsymbol{y}) + U(\boldsymbol{y})\}. \tag{11}$$

4. Find the node $\bar{\boldsymbol{x}}$ with the smallest value of V among all the *Considered*.
5. Move $\bar{\boldsymbol{x}}$ to *Accepted* ($U(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}})$).
6. Move the *Far* nodes adjacent to $\bar{\boldsymbol{x}}$ into *Considered*.
7. Reevaluate V for all the *Considered* $\boldsymbol{x}$ adjacent to $\bar{\boldsymbol{x}}$

$$V(\boldsymbol{x}) := \min\{V(\boldsymbol{x}), K(\boldsymbol{x}, \bar{\boldsymbol{x}}) + U(\bar{\boldsymbol{x}})\}. \tag{12}$$

8. If *Considered* is not empty, then go to 4.

The described algorithm has the computational complexity of $O(M \log(M))$; the factor of $\log(M)$ reflects the necessity to maintain a sorted list of the *Considered* values $V(\boldsymbol{x_i})$ to determine the next *Accepted* node.[2]

On a grid-like network, we can reinterpret Dijkstra's method as an upwind finite difference scheme. Consider a uniform Cartesian grid of grid size h, where the time-penalty $K_{ij} > 0$ is given for passing through each grid point $x_{ij} = (ih, jh)$. The minimal total time-to-exit U_{ij} starting from the node $\boldsymbol{x}_{ij}$ can be written in terms of the minimal total time-to-exit starting at its neighbors:

$$U_{ij} = \min(U_{i-1,j}, U_{i+1,j}, U_{i,j-1}, U_{i,j+1}) + K_{ij}. \tag{13}$$

As pointed out by Sethian in [36], the U_{ij} obtained through Dijkstra's method is formally a first-order approximation to the solution $u(x, y)$ of the differential equation

$$H(\nabla u(\boldsymbol{x}), u(\boldsymbol{x}), \boldsymbol{x}) = \max(|u_x|, |u_y|) = K(\boldsymbol{x}), \tag{14}$$

provided that the time-penalties are $K_{ij} = hK(\boldsymbol{x})$.

3. Continuous optimal trajectory problems and semi-Lagrangian discretization.

3.1. Statement of problem. Consider an optimal trajectory problem for a vehicle moving inside the domain Ω, with the speed f depending upon the direction of motion and the current position of the vehicle inside the domain. The dynamics of the vehicle is defined by

$$\begin{aligned} \boldsymbol{y}'(t) &= f(\boldsymbol{y}(t), \boldsymbol{a}(t))\boldsymbol{a}(t), \\ \boldsymbol{y}(0) &= \boldsymbol{x} \ \in \Omega, \end{aligned} \tag{15}$$

[2]This variant of Dijkstra's method is often referred to as a *heap-sort Dijkstra's method* since its implementation requires the use of a binary heap, d-heap, or Fibonacci heap to maintain the ordering of the *Considered* nodes efficiently [3].

The complexity estimate for the densely connected network would be $O(M^2 \log(M))$, but for our case, when X is a grid or a mesh, the precise complexity estimate is $O(rM \log(M))$, where r is the maximum number of nodes connected to a single node in X.

www.manaraa.com

where $\boldsymbol{y}(t)$ is the position of the vehicle at time t, $S_1 = \{\boldsymbol{a} \in R^2 \mid \|\boldsymbol{a}\| = 1\}$ is the set of *admissible control values*, and $\mathcal{A} = \{\boldsymbol{a} : R_{+,0} \mapsto S_1 \mid \boldsymbol{a}(\cdot) \text{ is measurable}\}$ is the set of *admissible controls*. We are interested in studying $\boldsymbol{y}(t)$ only while the vehicle remains inside Ω, i.e., until the exit time

$$T(\boldsymbol{x}, \boldsymbol{a}(\cdot)) = \inf\{t \in R_{+,0} | \boldsymbol{y}(t) \in \partial\Omega\}.$$

If the function $q(\boldsymbol{x}) \geq 0$ is the *time-penalty* for exiting the domain at the point $\boldsymbol{x} \in \partial\Omega$, then a *min-time* optimal trajectory problem is the task of finding an optimal control $\boldsymbol{a}(\cdot)$ which minimizes the total time:[3]

$$\text{TotalTime}\left(\boldsymbol{x}, \boldsymbol{a}(\cdot)\right) = T\left(\boldsymbol{x}, \boldsymbol{a}(\cdot)\right) + q\left(\boldsymbol{y}\left(T(\boldsymbol{x}, \boldsymbol{a}(\cdot))\right)\right).$$

We will alternatively refer to the above quantity as a *total cost* of using the control: $\text{Cost}\left(\boldsymbol{x}, \boldsymbol{a}(\cdot)\right) = \text{TotalTime}\left(\boldsymbol{x}, \boldsymbol{a}(\cdot)\right)$.

Unless otherwise explicitly specified, we will assume that both f and q are Lipschitz-continuous and that there exist constants f_1, f_2, q_1, q_2 such that

$$\begin{aligned} 0 < f_1 \leq f(\boldsymbol{x}, \boldsymbol{a}) \leq f_2 < \infty \quad &\text{for all } \boldsymbol{x} \in \Omega \text{ and } \boldsymbol{a} \in S_1, \\ 0 < q_1 \leq q(\boldsymbol{x}) \leq q_2 < \infty \quad &\text{for all } \boldsymbol{x} \in \partial\Omega. \end{aligned} \tag{16}$$

For notational convenience, we will also define the *anisotropy coefficient* $\Upsilon = \frac{f_2}{f_1}$. Strictly speaking, since f_1 and f_2 are global bounds, the coefficient Υ reflects the measure of anisotropy only in the homogeneous domain (i.e., when $f(\boldsymbol{x}, \boldsymbol{a}) = f(\boldsymbol{a})$). We will use Υ in deriving the worst-case-scenario computational complexity of the algorithms. In section 9.2, the more accurate *local anisotropy coefficient* $\Upsilon(\boldsymbol{x})$ will be defined and used for a more detailed computational complexity analysis.

As in the discrete case, the key idea of dynamic programming [5] is to define the value function $u(\boldsymbol{x})$ such that

$$\begin{cases} u(\boldsymbol{x}) = \inf_{\boldsymbol{a}(\cdot)} \text{Cost}(\boldsymbol{x}, \boldsymbol{a}(\cdot)), & \boldsymbol{x} \in \Omega \backslash \partial\Omega, \\ u(\boldsymbol{x}) = q(\boldsymbol{x}), & \boldsymbol{x} \in \partial\Omega. \end{cases} \tag{17}$$

In general, an optimal control $\boldsymbol{a}(\cdot)$ does not have to exist; therefore, when proving properties of the value function u, one uses *ϵ-suboptimal controls* $\boldsymbol{a}(\cdot)$ such that $\text{Cost}(\boldsymbol{x}, \boldsymbol{a}(\cdot)) < u(\boldsymbol{x}) + \epsilon$. To simplify the presentation, we will somewhat loosely refer to the optimal controls and trajectories. If such optimal controls do not exist, the corresponding properties can be formulated and proven for the ϵ-suboptimal controls and trajectories.

3.2. Properties of the value function. The following lemmas enumerate several well-known properties of the value function (see proofs in [46], for example). In section 7 we will prove the similar properties of the numerical approximation constructed by the OUM.

LEMMA 3.1 (Fixed-time optimality principle). *Let $d(\boldsymbol{x})$ be the minimum distance to the boundary $\partial\Omega$. Then for every point $\boldsymbol{x} \in \Omega\backslash\partial\Omega$ and for any $\tau < d(\boldsymbol{x})$*

$$u(\boldsymbol{x}) = \tau + \inf_{\boldsymbol{a}(\cdot)} \left\{u(\boldsymbol{y}(\tau))\right\}, \tag{18}$$

[3] A different optimal trajectory problem can be formulated for minimizing the total cost of moving the vehicle with a unit speed, when the running cost depends upon both the vehicle's position and the direction of motion; see [45, 15], for example. It is not hard to show the equivalence of this *min-cost* problem to the *min-time* optimal trajectory problem considered here [46].

www.manaraa.com

where $\boldsymbol{y}(\cdot)$ is a trajectory corresponding to a chosen control $\boldsymbol{a}(\cdot)$.

LEMMA 3.2 (Fixed-boundary optimality principle). *Consider a simple closed curve $\Gamma \subset \Omega \backslash \partial\Omega$ and an arbitrary point $\boldsymbol{x}$ inside Γ. For every control $\boldsymbol{a}(\cdot)$, we define $T_\Gamma(\boldsymbol{x}, \boldsymbol{a}(\cdot))$ to be the earliest time at which the corresponding trajectory $\boldsymbol{y}(\cdot)$ reaches the curve Γ. Then*

$$u(\boldsymbol{x}) = \inf_{\boldsymbol{a}(\cdot)} \left\{ T_\Gamma\left(\boldsymbol{x}, \boldsymbol{a}(\cdot)\right) + u\left(\boldsymbol{y}\left(T_\Gamma(\boldsymbol{x}, \boldsymbol{a}(\cdot))\right)\right)\right\}. \tag{19}$$

LEMMA 3.3. *The value function $u(\boldsymbol{x})$ is Lipschitz-continuous*[4] *and bounded on $\Omega \backslash \partial\Omega$. If $\boldsymbol{y}'(t) = \boldsymbol{a}(t)$ defines an optimal trajectory for a point $\boldsymbol{x}$ (i.e., $\boldsymbol{y}(0) = \boldsymbol{x}$ and $u(\boldsymbol{x}) = \mathrm{Cost}(\boldsymbol{x}, \boldsymbol{a}(\cdot))$), then the function $u(\boldsymbol{y}(t))$ is strictly decreasing for $t \in [0, T(\boldsymbol{x}, \boldsymbol{a}(\cdot))]$.*

The following two lemmas utilize the *fixed-time optimality principle* and provide the key motivation for constructing OUMs for this problem (sections 6 and 8).

LEMMA 3.4. *Consider a point $\bar{\boldsymbol{x}} \in \Omega \backslash \partial\Omega$. Then, for any constant C such that $q_2 \leq C \leq u(\bar{\boldsymbol{x}})$, the optimal trajectory for $\bar{\boldsymbol{x}}$ will intersect the level set $u(\boldsymbol{x}) = C$ at some point $\tilde{\boldsymbol{x}}$. If $\bar{\boldsymbol{x}}$ is distance d_1 away from that level set, then*

$$\|\tilde{\boldsymbol{x}} - \bar{\boldsymbol{x}}\| \leq d_1 \Upsilon. \tag{20}$$

Proof. Let $\boldsymbol{a}(\cdot)$ be an optimal control for $\bar{\boldsymbol{x}}$. The intersection point $\tilde{\boldsymbol{x}} = \boldsymbol{y}(\tau)$ exists because of the continuity of the value function and of the optimal trajectory: $u(\bar{\boldsymbol{x}}) \geq C \geq q_2 \geq u(\boldsymbol{y}(T(\bar{\boldsymbol{x}}, \boldsymbol{a}(\cdot))))$.

Therefore,

$$u(\bar{\boldsymbol{x}}) = \tau \ + \ u(\tilde{\boldsymbol{x}}) \geq \frac{\|\tilde{\boldsymbol{x}} - \bar{\boldsymbol{x}}\|}{f_2} + C.$$

There also exists some point $\hat{\boldsymbol{x}}$ on the level set such that $\|\bar{\boldsymbol{x}} - \hat{\boldsymbol{x}}\| = d_1$. Consider a control $\boldsymbol{a_1}(t) = \frac{\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}}{d_1}$, and suppose it takes time τ_1 to reach $\hat{\boldsymbol{x}}$ along the corresponding straight-line trajectory. By the optimality principle,

$$u(\bar{\boldsymbol{x}}) \leq \tau_1 \ + \ u(\hat{\boldsymbol{x}}) \leq \frac{d_1}{f_1} + C.$$

Thus, $\|\tilde{\boldsymbol{x}} - \bar{\boldsymbol{x}}\| \leq d_1 \Upsilon$. ∎

LEMMA 3.5. *Consider an unstructured (triangulated) mesh X of diameter h on Ω. Consider a simple closed curve $\Gamma \subset \Omega \backslash \partial\Omega$ with the property that for any point $\boldsymbol{x}$ on Γ there exists a mesh point $\hat{\boldsymbol{x}}$ inside Γ such that $\|\boldsymbol{x} - \hat{\boldsymbol{x}}\| < h$. Suppose a mesh point $\bar{\boldsymbol{x}}$ is such that $u(\bar{\boldsymbol{x}}) \leq u(\boldsymbol{x_i})$ for all the mesh points $\boldsymbol{x_i} \in X$ inside Γ. The optimal trajectory for $\bar{\boldsymbol{x}}$ will intersect Γ at some point $\tilde{\boldsymbol{x}}$ such that*

$$\|\tilde{\boldsymbol{x}} - \bar{\boldsymbol{x}}\| \leq h \Upsilon. \tag{21}$$

Proof. Let $\boldsymbol{a}(\cdot)$ be an optimal control for $\bar{\boldsymbol{x}}$. The intersection point $\tilde{\boldsymbol{x}} = \boldsymbol{y}(\tau)$ exists because of the continuity of Γ and of the optimal trajectory. Therefore,

$$u(\bar{\boldsymbol{x}}) = \tau \ + \ u(\tilde{\boldsymbol{x}}) \geq \frac{\|\tilde{\boldsymbol{x}} - \bar{\boldsymbol{x}}\|}{f_2} + \ u(\tilde{\boldsymbol{x}}).$$

[4] This holds in the interior of Ω even in the presence of *state constraints*: the assumption $f_1 > 0$ is sufficient for the *local controllability* near $\partial\Omega$ (as defined in [4], for example).

www.manaraa.com

Let $\hat{\boldsymbol{x}}$ be a mesh point inside Γ such that $\|\tilde{\boldsymbol{x}} - \hat{\boldsymbol{x}}\| \le h$. Consider a control $\boldsymbol{a_1}(t) = \frac{\tilde{\boldsymbol{x}}-\hat{\boldsymbol{x}}}{\|\tilde{\boldsymbol{x}}-\hat{\boldsymbol{x}}\|}$, and let τ_1 be the time required to reach $\tilde{\boldsymbol{x}}$ from $\hat{\boldsymbol{x}}$ using $\boldsymbol{a_1}(\cdot)$. Then, by the optimality principle,

$$u(\hat{\boldsymbol{x}}) \le \tau_1 \; + \; u(\tilde{\boldsymbol{x}}) \le \frac{h}{f_1} \; + \; u(\tilde{\boldsymbol{x}}).$$

To complete the proof, we recall that $u(\bar{\boldsymbol{x}}) \le u(\hat{\boldsymbol{x}})$. ∎

3.3. Hamilton–Jacobi–Bellman PDE. As in the discrete case, Bellman's optimality principle can be used to formally derive the local equation for $u(\boldsymbol{x})$ if the value function is smooth around $\boldsymbol{x}$:

$$\begin{aligned} \min_{\boldsymbol{a}\in S_1}\{(\nabla u(\boldsymbol{x})\cdot\boldsymbol{a})f(\boldsymbol{x},\boldsymbol{a})\} \; + \; 1 = 0, \quad \boldsymbol{x}\in\Omega, \\ u(\boldsymbol{x}) = q(\boldsymbol{x}), \quad \boldsymbol{x}\in\partial\Omega. \end{aligned} \tag{22}$$

The above *Hamilton–Jacobi–Bellman PDE* can be rewritten in the form $H(\nabla u, \boldsymbol{x}) = 1$, where the Hamiltonian $H = -\min_{\boldsymbol{a}\in S_1}\{(\boldsymbol{p}\cdot\boldsymbol{a})f(\boldsymbol{x},\boldsymbol{a})\} = \max_{\boldsymbol{a}\in S_1}\{(\boldsymbol{p}\cdot(-\boldsymbol{a}))f(\boldsymbol{x},\boldsymbol{a})\}$. Moreover, this Hamiltonian is convex and homogeneous of degree one in the first argument; thus, this PDE belongs to the class of problems described in section 1. We also note that the characteristics of this PDE can be formally shown to be the optimal trajectories for the corresponding min-time control problem.

In an important case of isotropic optimal speed function ($f(\boldsymbol{x},\boldsymbol{a}) = f(\boldsymbol{x})$), equation (22) reduces to the *Eikonal equation* $\|\nabla u(\boldsymbol{x})\| = \frac{1}{f(\boldsymbol{x})}$. We particularly emphasize one property of the Eikonal equations: if ∇u is defined at the point, then the minimizer is $\boldsymbol{a} = \frac{-\nabla u}{\|\nabla u\|}$. Thus, the gradient lines of $u(\boldsymbol{x})$ coincide with the characteristics of the Eikonal PDE (i.e., the optimal trajectories for the isotropic min-time control problem). This is the main reason for the following *causality property*, a foundation for the noniterative Eikonal solvers.

PROPERTY 3.6 (Causality for the Eikonal equation). *If $\nabla u(\boldsymbol{x})$ is defined and $\boldsymbol{x}\boldsymbol{x_1}\boldsymbol{x_2}$ is a sufficiently small acute simplex, which contains the characteristic for $\boldsymbol{x}$, then $u(\boldsymbol{x}) \ge \max\{u(\boldsymbol{x_1}), u(\boldsymbol{x_2})\}$.*

Unfortunately, a smooth solution to (22) might not exist even for smooth f, q, and $\partial\Omega$. Generally, this equation has infinitely many weak Lipschitz-continuous solutions, but the unique *viscosity solution* [10] can be defined using the conditions on smooth test functions [9] as follows.

A bounded, uniformly continuous function u is the *viscosity solution* of (22) if the following holds for each smooth test function[5] $\phi \in C_c^\infty(\Omega)$:

(i) if $u - \phi$ has a local minimum at $\boldsymbol{x_0} \in \Omega$, then

$$\min_{\boldsymbol{a}\in S_1}\{(\nabla\phi(\boldsymbol{x_0})\cdot\boldsymbol{a})f(\boldsymbol{x_0},\boldsymbol{a})\} + 1 \le 0; \tag{23}$$

(ii) if $u - \phi$ has a local maximum at $\boldsymbol{x_0} \in \Omega$, then

$$\min_{\boldsymbol{a}\in S_1}\{(\nabla\phi(\boldsymbol{x_0})\cdot\boldsymbol{a})f(\boldsymbol{x_0},\boldsymbol{a})\} + 1 \ge 0. \tag{24}$$

Moreover, the optimality principle (Lemma 3.1) can be used to demonstrate that the value function of the min-time optimal trajectory problem satisfies the inequalities

[5] The standard definition of the viscosity solution (see [9, 8], for example) uses the test functions $\phi \in C^1(\Omega)$. However, as shown in [9], the definition using the test functions $\phi \in C_c^\infty(\Omega)$ is equivalent. This second formulation enables us to use the upper bounds on the second derivatives of ϕ in the convergence proof in section 7.

www.manaraa.com

(24) and (23) and thus is the viscosity solution of the Hamilton–Jacobi–Bellman PDE (see [8, 7] or [16], for example[6]).

3.4. Modified definition of the viscosity solution. Define $S_1^{\phi,\boldsymbol{x}} = \{\boldsymbol{a} \in S_1 \mid \boldsymbol{a}\cdot\nabla\phi(\boldsymbol{x}) \le -\|\nabla\phi(\boldsymbol{x})\|\Upsilon^{-1}\}$. In [46] we demonstrate that using $S_1^{\phi,\boldsymbol{x_0}}$ instead of S_1 in the inequalities (24) and (23) yields an equivalent definition of the viscosity solution for (22).

Proof. We first observe that, since $f > f_1 > 0$, if the minimum is attained for some $\boldsymbol{a} = \boldsymbol{a_1}$, then $(\boldsymbol{a_1}\cdot\nabla\phi(\boldsymbol{x_0})) < 0$. Let $\boldsymbol{b} = \frac{-\nabla\phi(\boldsymbol{x_0})}{\|\nabla\phi(\boldsymbol{x_0})\|}$. Since $\boldsymbol{a_1}$ is the minimizer, we have

$$(\nabla\phi(\boldsymbol{x_0})\cdot\boldsymbol{a_1})f(\boldsymbol{x_0},\boldsymbol{a_1}) \le (\nabla\phi(\boldsymbol{x_0})\cdot\boldsymbol{b})f(\boldsymbol{x_0},\boldsymbol{b}) \le -\|\nabla\phi(\boldsymbol{x_0})\|f_1.$$

Therefore,

$$\boldsymbol{a_1}\cdot\nabla\phi(\boldsymbol{x_0}) \le -\|\nabla\phi(\boldsymbol{x_0})\|\frac{f_1}{f(\boldsymbol{x_0},\boldsymbol{a_1})} \le -\|\nabla\phi(\boldsymbol{x_0})\|\Upsilon^{-1}. \qquad \square$$

Remark 3.7. We have just established a bound on the angle between the characteristic of the PDE (22) and the gradient line of its viscosity solution. If the gradient $\nabla u(\boldsymbol{x_0})$ exists, then $\nabla u(\boldsymbol{x_0}) = \nabla\phi(\boldsymbol{x_0})$. Therefore, $\boldsymbol{a_1}\cdot\nabla u(\boldsymbol{x_0}) \le -\|\nabla u(\boldsymbol{x_0})\|\Upsilon^{-1}$. If γ is the angle between $\nabla u(\boldsymbol{x_0})$ and $(-\boldsymbol{a_1})$, then $\cos(\gamma) \ge \frac{1}{\Upsilon}$. (If the level sets of $u(\boldsymbol{x})$ were straight lines, the last inequality would trivially follow from Lemma 3.4.) We note that the above argument heavily uses the existence of a positive lower bound f_1 and, therefore, does not directly apply to more general control problems.

3.5. A semi-Lagrangian discretization for the Hamilton–Jacobi–Bellman PDE. Assume that a triangulated mesh X of diameter h is defined on Ω. For every mesh point $\boldsymbol{x} \in X$, define $S(\boldsymbol{x})$ to be a set of all the simplexes in the mesh adjacent to $\boldsymbol{x}$ (i.e., the simplexes that have $\boldsymbol{x}$ as one of their vertices). If $s \in S(\boldsymbol{x})$, we will use the notation $\boldsymbol{x_{s,1}}$ and $\boldsymbol{x_{s,2}}$ for the other vertices of the simplex s.

A simple control-theoretic discretization of (19) follows from the assumption that, as the vehicle starts to move from a mesh point $\boldsymbol{x}$ inside a simplex $s \in S(x)$, its direction of motion $\boldsymbol{a}$ does not change until the vehicle reaches the edge $\boldsymbol{x_{s,1}x_{s,2}}$ (see Figure 1). The value $u(\tilde{\boldsymbol{x}})$ at the point of intersection can be approximated[7] using the values $u(\boldsymbol{x_{s,1}})$ and $u(\boldsymbol{x_{s,2}})$.

Defining $\tau(\zeta) = \|\tilde{\boldsymbol{x}} - \boldsymbol{x}\| = \|(\zeta\boldsymbol{x_{s,1}} + (1-\zeta)\boldsymbol{x_{s,2}}) - \boldsymbol{x}\|$ and $\boldsymbol{a}_\zeta = \frac{\tilde{\boldsymbol{x}}-\boldsymbol{x}}{\tau(\zeta)}$, we can now write the equation for the numerical approximation U:

$$\begin{aligned} U(\boldsymbol{x}) &= \min_{s\in S(\boldsymbol{x})} V_s(\boldsymbol{x}), \\ V_s(\boldsymbol{x}) &= \min_{\zeta\in[0,1]} \left\{ \frac{\tau(\zeta)}{f(\boldsymbol{x},\boldsymbol{a}_\zeta)} + \zeta U(\boldsymbol{x_{s,1}}) + (1-\zeta)U(\boldsymbol{x_{s,2}}) \right\}. \end{aligned} \tag{25}$$

The above "naive" derivation is based on a direct application of Bellman's optimality principle rather than on discretization of the corresponding Hamilton–Jacobi–Bellman

[6] The control-theoretic problems discussed in these papers are slightly different. They consider infinite horizon or exit time problems with time-discounted running costs, e.g., $\mathrm{Cost}(\boldsymbol{x},\boldsymbol{a}(\cdot)) = \int_0^\infty e^{-\lambda s}K(\boldsymbol{y}(s),\boldsymbol{a}(s))ds$. Thus, the resulting PDE is also slightly different, but Kruzhkov's transform [24] can be used to obtain the mapping from one to another; see [8], [4], for example. In addition, the iterative methods in these papers are also applicable for a more general case of $f_1 = 0$.

[7] Since the interpolation has to be used to approximate the value at a non–mesh point $\tilde{\boldsymbol{x}}$, we refer to this and similar discretizations as *semi-Lagrangian*.

www.manaraa.com

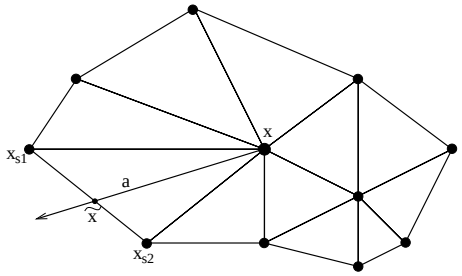


FIG. 1. *A control-theoretic discretization: each trajectory is approximated by a straight line within a simplex.*

PDE; a number of related methods, treatment of more general control problems (including the case $f_1 = 0$), and the proof of convergence can be found in [25, 26], and [18]. Similar higher-order control-theoretic numerical methods can be found in [17].

The discretized equation (25) has to be satisfied at every mesh point in X; this results in a coupled system of M nonlinear equations, which usually have to be solved simultaneously through the iterations. Due to the structure of the system, each iteration involves solving a local minimization problem for each mesh point, and even in the simplest problems the number of iterations will be proportional to the diameter of the mesh-graph. The number of iterations can be reduced using Gauss–Seidel relaxation (as in [18]), but we know of no theoretical guarantees of the rate of convergence.

4. OUMs for the isotropic case: Dijkstra-like Eikonal solvers. Until recently, the Eikonal equation, corresponding to the isotropic optimal trajectory and front propagation problems, was the only case for which single-pass methods were available. Several fast algorithms have been introduced to solve the corresponding discretized system as efficiently as Dijkstra's method solves the shortest path problems on discrete networks. These methods are based on an observation that a particular upwind discretization possesses a *causality* property similar to that of the Eikonal equation (Property 3.6).

PROPERTY 4.1 (Causality). *If s is an acute simplex in $S(\boldsymbol{x})$ and $V_s(\boldsymbol{x})$ is a value of $U(\boldsymbol{x})$ computed under the assumption that the characteristic for $\boldsymbol{x}$ lies in s, then $V_s(\boldsymbol{x}) \geq \max\{U(\boldsymbol{x_{s,1}}), U(\boldsymbol{x_{s,2}})\}$.*

Any upwind discretization possessing this property leads to equations which can be decoupled by computing the value function at the mesh points in the increasing order. Since the ordering is not known in advance, we can structure these Dijkstra-like solvers in the spirit of section 2.2 as follows.

All the mesh points are divided into three classes: *Far* (no information about the correct value of U is known), *Accepted* (the correct value of U has been computed), and *Considered* (adjacent to *Accepted*), for which V has already been computed, but it is still unclear if $V = U$. For every *Considered* $\boldsymbol{x}$, we define the set $\mathrm{NS}(\boldsymbol{x}) = \{s \in S(\boldsymbol{x}) \mid \boldsymbol{x_{s,1}}$ and $\boldsymbol{x_{s,2}}$ are *Accepted*$\}$. We will also use a set $S(\boldsymbol{x_1}, \boldsymbol{x_2})$ to denote the simplexes adjacent to both of these mesh points.

1. Start with all the mesh points in *Far*.
2. Move the mesh points on the boundary ($\boldsymbol{y} \in \partial\Omega$) to *Accepted* ($U(\boldsymbol{y}) = q(\boldsymbol{y})$).
3. Move all the mesh points $\boldsymbol{x}$ adjacent to the boundary into *Considered* and

www.manaraa.com

evaluate the tentative values

$$V(\boldsymbol{x}) := \min_{s \in NS(\boldsymbol{x})} V_s(\boldsymbol{x}). \tag{26}$$

4. Find the mesh point $\bar{\boldsymbol{x}}$ with the smallest value of V among all the *Considered*.
5. Move $\bar{\boldsymbol{x}}$ to *Accepted* ($U(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}})$).
6. Move the *Far* mesh points adjacent to $\bar{\boldsymbol{x}}$ into *Considered*.
7. Reevaluate V for all the *Considered* $\boldsymbol{x}$ adjacent to $\bar{\boldsymbol{x}}$

$$V(\boldsymbol{x}) := \min\left\{V(\boldsymbol{x}), \min_{s \in (S(\boldsymbol{x},\bar{\boldsymbol{x}}) \bigcap \mathrm{NS}(\boldsymbol{x}))} V_s(\boldsymbol{x})\right\}. \tag{27}$$

8. If *Considered* is not empty, then go to 4.

Such Dijkstra-like methods use heap-sort data structures to achieve Dijkstra-like efficiency of $O(M \log M)$ and compute the numerical solutions converging to the viscosity solution (due to the upwinding structure of discretization).

The first Dijkstra-like method, introduced by Tsitsiklis in 1994, evolved from studying isotropic min-cost optimal trajectory problems and was based on a direct approximation of the characteristic directions at each mesh point [44, 45]. Tsitsiklis proved that Property 4.1 holds for the particular first-order semi-Lagrangian discretization (i.e., formula (25)) of the Eikonal equation, when used on a uniform Cartesian grid in R^n. The algorithm requires solving a local minimization problem to update the solution at each mesh point; however, as shown in [45], the Kuhn–Tucker optimality conditions can be used to find a quadratic equation satisfied by the minimum value instead. In the appendix we provide a more general proof that the same causality property is possessed by the discretization (25) on an arbitrary unstructured mesh and derive the corresponding quadratic equation for the minimum value.

The family of Fast Marching Methods, introduced by Sethian in [33] and extended by Sethian and several co-authors in [35, 21, 38], evolved from studying isotropic front propagation problems (see section 8.1 for the recasting of Eikonal PDE in this context). Those discretizations were based on upwinding approximations of the gradient and were all obtained in a fully Eulerian frame of reference. Sethian proved that the causality Property 4.1 holds for a wide class of upwind finite-difference discretizations. Following that approach, upwind finite-difference operators were then used to obtain higher-order Cartesian versions [35], extensions to triangulated meshes [21], and general higher-order versions for the unstructured meshes in R^n [38]. In addition, the "lifting-to-surface" technique introduced in [38] allowed the Fast Marching Method to be used to solve a limited class of non-Eikonal (elliptically anisotropic) problems. We note that these extensions are all OUMs, relying on an upwinding criterion that establishes a monotonicity-preserving update procedure. Early applications of the Fast Marching Methods included the narrow band level set method [1], photolithography [34], a comparison of a similar algorithm with volume-of-fluid techniques [19], and a fast algorithm for image segmentation [27]. More recent applications include problems in robotic navigation [22], extension velocity computation [2], visibility evaluation [35], geophysics [32, 37], and computational geometry [23]. To produce an update for each mesh point, these methods require solving a quadratic equation, which will depend on the particular upwind finite-difference operator used. The original Fast Marching Method, as defined in [33], was based on the first-order Godunov-type discretization on a uniform Cartesian grid, and the corresponding quadratic update equation coincides with the equation derived from the Kuhn–Tucker conditions in [45].

www.manaraa.com

(See the appendix for a discussion of the correspondence between the first-order semi-Lagrangian and fully Eulerian discretizations on unstructured meshes.)

Several different higher-order versions of the Fast Marching Method are available for structured and unstructured meshes, while there are currently no higher-order Dijkstra-like methods based on a semi-Lagrangian approach. The reason is the difficulty of finding the higher-order semi-Lagrangian discretization that would provably possess the causality property. Finally, when such a discretization is found, it will generally require performing a local minimization at every mesh point, since it is not obvious whether the Kuhn–Tucker conditions can be used to produce a quadratic equation in this more general case.

5. Characteristics versus gradients. In this section, we show why the Dijkstra-like methods cannot be directly applied to handle general (non-Eikonal) Hamilton–Jacobi equations. As a test problem, consider the "distance from the origin" Eikonal equation ($\|\nabla d\| = 1$, $d(0,0) = 0$), in a plane $z = c_1 x + c_2 y$ for some vector $\boldsymbol{c} = \left[\begin{smallmatrix} c_1 \\ c_2 \end{smallmatrix}\right]$. The level sets of d will be just the circles around the origin in that plane. Projecting those circles orthogonally onto the $x-y$ plane, we will see a set of concentric ellipses. As expected, the function $u(\boldsymbol{x})$, whose level sets coincide with these ellipses, may be obtained in two ways:

- *As an optimal-trajectory problem.* The function d can be considered as a value function for a vehicle moving with a unit speed in the plane $z = c_1 x + c_2 y$. As shown in [46], if one considers another vehicle which moves as a shadow of the first one in the $x-y$ plane, its value function will be the viscosity solution $u(\boldsymbol{x})$ of the Hamilton–Jacobi–Bellman equation

$$\min_{\boldsymbol{a} \in S_1} \{(\nabla u(\boldsymbol{x}) \cdot \boldsymbol{a}) f(\boldsymbol{x}, \boldsymbol{a})\} + 1 = 0, \qquad \boldsymbol{x} \in \Omega, \tag{28}$$

  and the vehicle's speed function in the direction $\boldsymbol{a} = \left[\begin{smallmatrix} a_1 \\ a_2 \end{smallmatrix}\right]$ will be given by

$$f(\boldsymbol{a}, x, y) = \left(1 + (c_1 a_1 + c_2 a_2)^2\right)^{-\frac{1}{2}}.$$

- *As a front propagation problem.* As shown in [36], this same problem can be viewed in the front propagation framework, using the speed function

$$F(\boldsymbol{x}, \boldsymbol{n}) = \sqrt{\frac{(1+c_2^2)n_1^2 + (1+c_1^2)n_2^2 - 2c_2 c_1 n_1 n_2}{1 + c_1^2 + c_2^2}}.$$

  The Hamilton–Jacobi PDE corresponding to this speed function F is

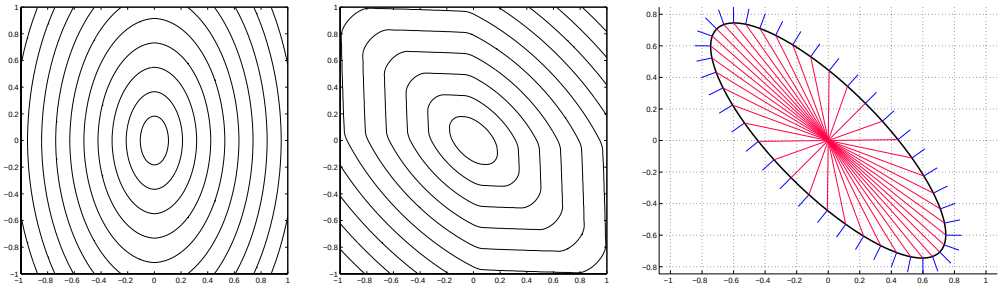
$$\sqrt{\frac{(1+c_2^2)u_x^2(\boldsymbol{x}) + (1+c_1^2)u_y^2(\boldsymbol{x}) - 2c_2 c_1 u_x(\boldsymbol{x}) u_y(\boldsymbol{x})}{1 + c_1^2 + c_2^2}} = 1. \tag{29}$$

It would appear that Tsitsiklis' algorithm (defined for the isotropic case in section 4) can be applied to this anisotropic problem without any changes at all, except that the dependence of the speed f upon the direction $\boldsymbol{a}$ will now be present in the update-from-a-single-simplex formula:

$$V_s(\boldsymbol{x}) = \min_{\zeta \in [0,1]} \left\{ \frac{\tau(\zeta)}{f(\boldsymbol{x}, \boldsymbol{a}_\zeta)} + \zeta U(\boldsymbol{x_{s,1}}) + (1-\zeta) U(\boldsymbol{x_{s,2}}) \right\}. \tag{30}$$

What happens when this algorithm is used to compute the expansion of the ellipse (as defined by (28))? In Figure 2 we show the level sets of the numerical solution

www.manaraa.com



(a) (b) (c)

FIG. 2. (a) *and* (b) *Ellipse expansion computed by Tsitsiklis' algorithm. Both computations were performed on a* 129×129 *uniform Cartesian grid.* (c) *The characteristics and the gradient directions for the second expanding ellipse.*

U obtained by this method for two different expanding ellipses. The first contour plot corresponds to the vector $\boldsymbol{c} = \begin{bmatrix} \sqrt{2} \\ 0 \end{bmatrix}$. The numerical solution converges to the value function $u(\boldsymbol{x})$ and is first-order accurate as the grid size tends to 0. (We will return to this example later when we discuss fast methods relying on a particular grid orientation in section 5.) The second contour plot corresponds to the vector $\boldsymbol{c} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$. In this case, it is obvious that $U(\boldsymbol{x})$ does not approximate the viscosity solution very well. Nor does it improve under a grid refinement.

In order to understand what is different in the second example, we recall that all Dijkstra-like methods are fundamentally dependent on the *causality property* (3.6) of the Eikonal equation. Each of these single-pass methods is based on the observation that a certain discretization also possesses a similar *causality property*. This *causality* results from the fact that the characteristics of the Eikonal equation coincide with the gradient lines of its viscosity solution u. However, for the anisotropic problems this property does not hold. When the characteristic and gradient directions are different, the simplex $\boldsymbol{x}\boldsymbol{x_j}\boldsymbol{x_k}$ may contain the characteristic for the point $\boldsymbol{x}$, even if the gradient $\nabla u(\boldsymbol{x})$ is not pointing from that simplex. Thus, no matter how small that simplex is, it is still possible that $u(\boldsymbol{x}) < u(\boldsymbol{x_j})$. This is an intrinsic problem with Dijkstra-like methods in the anisotropic case: to produce the numerical solution efficiently, these methods attempt to compute $U(\boldsymbol{x})$ in the ascending order (i.e., from the simplex containing $(-\nabla u)$), whereas, in order to maintain the upwinding, $U(\boldsymbol{x})$ has to be computed from the simplex containing the characteristic. That phenomenon is also quite obvious from comparing Figures 2(b) and 2(c): the Dijkstra-like method fails exactly at those points where the gradient line and the characteristic do not lie in the same coordinate quadrant (or, more generally, in the same simplex—the quadrants are used because the numerical solution in Figure 2 is computed on a Cartesian grid).

However, it is still possible that, for a given PDE and for a chosen discretization scheme, Dijkstra-like decoupling will produce a convergent numerical solution, provided that it is used on *a specially oriented grid* (e.g., Figure 2(a)). A criterion based on this observation was introduced by Sethian in [36] as follows.

CRITERION 5.1 (Applicability of the Fast Marching Method). *For a static Hamilton–Jacobi equation* $H(\nabla u, \boldsymbol{x}) = 0$*, if the convex Hamiltonian* H *is approximated on a Cartesian grid by a consistent difference operator*

$$H_{ij}(U_{i,j}, U_{i-1,j}, U_{i+1,j}, U_{i,j-1}, U_{i,j+1}, \boldsymbol{x_{i,j}}) = 0,$$

www.manaraa.com

and if it is known that $U_{i,j}$ depends only on the smaller *values of U at the neighboring points, then the Fast Marching Method can be used to compute $U_{i,j}$'s efficiently.*

Remark 5.2. In the context of upwinding discretizations on unstructured meshes, the above criterion is equivalent to requiring that the characteristics and the (numerically approximated) vector $(-\nabla u)$ should always lie in the same simplex. Several sufficient conditions for a class of numerical Hamiltonians to satisfy the above criterion on Cartesian grids were presented by Osher and Fedkiw in [29]. For instance, the causality property was proven in [29] for the Godunov-type upwinding discretization H_{ij}^G, provided that the original Hamiltonian $H(\nabla u, \boldsymbol{x})$ has a special form $H(\nabla u, \boldsymbol{x}) = G(u_x^2, u_y^2)$ for some function G. We note that, even for a relatively simple elliptical front propagation (29), this condition is satisfied only in the case when c_1 or c_2 is equal to zero, i.e., only when the axes of the ellipse are exactly aligned with the grid coordinate directions. This is precisely the situation illustrated by Figure 2(a).

In general, finding discretizations which satisfy Criterion 5.1 is a difficult task. We note the following problems associated with this approach:

- *Whether or not the criterion is satisfied depends upon a particular grid/mesh-orientation.* Indeed, the two test problems in Figure 2 are actually the same (modulo a rotation by 45°), yet only one of them satisfies the criterion.
- *For any anisotropic problem, there are infinitely many grid orientations such that the criterion is not satisfied.* If an angle between the characteristic and the gradient line is not zero, then any grid line lying inside that angle will violate the criterion. Correspondingly, the bigger the *anisotropy coefficient* Υ is, the harder it is to find the grid orientation satisfying the criterion.
- *The criterion is infinitely sensitive to grid perturbations.*
- *If the criterion is not satisfied, the numerical solution does not lose stability under grid refinement.* In other words, when it does not work, it is not immediately obvious.
- *If the criterion is not satisfied even at a single grid point, the numerical solution need not converge to the viscosity solution.* Criterion 5.1 is the basis for determining the order for computing the values of U. Computing even one of them from a wrong quadrant can greatly affect the ordering of the remaining computations.
- *For many anisotropic problems, the criterion cannot be satisfied for any choice of the grid directions.* Indeed, if the angle between gradient lines and the characteristics is sufficiently wide, and if the medium is substantially inhomogeneous (i.e., if the speed $f(\boldsymbol{x}, \boldsymbol{a})$ varies significantly in different parts of Ω), then any Cartesian grid might violate Criterion 5.1 for some grid point $\boldsymbol{x} \in X$.

As a result, we have chosen to concentrate on a family of robust single-pass methods, which are independent of the grid choice[8] and applicable to a wider class of control problems. Nevertheless, for the limited class of problems in which Criterion 5.1 can be analytically demonstrated for a certain choice of grid, the original Dijkstra-like solvers will perform better than the new OUMs introduced in the next section.

[8] Of course, it is just the fact of *convergence* that is independent of the grid choice for our methods; the *speed of convergence* is certainly influenced by the choice of the grid and its alignment with the shock lines.

In fact, if the computational mesh is not fixed for some application-specific reasons, the convergence of our single-pass methods can be further improved by using the computed characteristic information to dynamically add the mesh points inside the AF, wherever the shock is suspected.

www.manaraa.com

5.1. Causality in the Hamilton–Jacobi–Bellman PDE. A different (weaker) *causality property* for the more general Hamilton–Jacobi–Bellman equation results from Bellman's optimality principle (see section 3). Since the characteristics of that PDE are, in fact, the optimal trajectories of the corresponding control problem, we know that the value function u is strictly increasing along the characteristics. Our OUMs for the general anisotropic problems will be based on the fixed-boundary optimality principle (Lemma 3.2).

Let $u(\boldsymbol{x})$ be the value function for the anisotropic min-time optimal trajectory problem defined on Ω in section 3. We will use the notation $T_{\hat{\boldsymbol{x}}}(\boldsymbol{x})$ for the minimum time required to reach the point $\hat{\boldsymbol{x}}$ starting from the point $\boldsymbol{x}$. If Γ is a simple closed curve in $\Omega \backslash \partial\Omega$ and a point $\boldsymbol{x}$ is inside Γ, then Lemma 3.2 shows that

$$u(\boldsymbol{x}) = \inf_{\hat{\boldsymbol{x}} \in \Gamma} \left\{ T_{\hat{\boldsymbol{x}}}(\boldsymbol{x}) + u(\hat{\boldsymbol{x}}) \right\}.$$

Because of the properties of the speed function f and by the continuity of Γ, that infimum is actually a minimum achieved at some point $\tilde{\boldsymbol{x}}$, i.e., $u(\boldsymbol{x}) = T_{\tilde{\boldsymbol{x}}}(\boldsymbol{x}) + u(\tilde{\boldsymbol{x}})$. The point $\tilde{\boldsymbol{x}}$ can be interpreted as an intersection of the optimal trajectory for $\boldsymbol{x}$ with the curve Γ. Thus, knowing u on Γ is sufficient for evaluating u at any point inside Γ. Moreover, if Γ is a level set of $u(\boldsymbol{x})$, then, by Lemma 3.4, we know that $\|\tilde{\boldsymbol{x}} - \boldsymbol{x}\| \leq d_1 \Upsilon$, where $\Upsilon = \frac{f_2}{f_1}$ and d_1 is the distance from $\boldsymbol{x}$ to Γ (see Figure 3). The last observation[9] necessary for constructing a computational algorithm is that, if d_1 is small relative to the size of Γ, then the optimal time $T_{\tilde{\boldsymbol{x}}}(\boldsymbol{x})$ cannot be much smaller than the time required to traverse the straight line trajectory from $\boldsymbol{x}$ to $\tilde{\boldsymbol{x}}$.

6. Control-theoretic OUM. We now describe our control-theoretic OUM, which was first discussed without convergence proof in [39]. Consider an unstructured triangulated mesh X of diameter h (i.e., if the mesh points $\boldsymbol{x_j}$ and $\boldsymbol{x_k}$ are adjacent, then $\|\boldsymbol{x_j} - \boldsymbol{x_k}\| \leq h$).

Let $\boldsymbol{x_j}$ and $\boldsymbol{x_k}$ be two adjacent mesh points. Define the upwinding approximation for $U(\boldsymbol{x})$ from a "virtual simplex" $\boldsymbol{x_j x x_k}$:

$$V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x}) = \min_{\zeta \in [0,1]} \left\{ \frac{\tau(\zeta)}{f(\boldsymbol{x}, \boldsymbol{a}_\zeta)} + \zeta U(\boldsymbol{x_j}) + (1-\zeta) U(\boldsymbol{x_k}) \right\}, \tag{31}$$

where $\tau(\zeta) = \|(\zeta \boldsymbol{x_j} + (1-\zeta)\boldsymbol{x_k}) - \boldsymbol{x}\|$ and $\boldsymbol{a}_\zeta = \frac{(\zeta \boldsymbol{x_j} + (1-\zeta)\boldsymbol{x_k}) - \boldsymbol{x}}{\tau(\zeta)}$.

Remark 6.1. The above update formula is basically the same as the upwind formula for simplex s given by (25). The difference is that $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$ is defined even when $\boldsymbol{x_j}$ and $\boldsymbol{x_k}$ are not adjacent to $\boldsymbol{x}$.

Control-theoretic OUM for anisotropic problems. As before, mesh points are divided into three classes (*Far*, *Considered*, *Accepted*). The *AcceptedFront* is defined as a set of *Accepted* mesh points, which are adjacent to some not-yet-accepted (i.e., *Considered*) mesh points. Define the set AF of the line segments $\boldsymbol{x_j x_k}$, where $\boldsymbol{x_j}$ and $\boldsymbol{x_k}$ are adjacent mesh points on the *AcceptedFront*, such that there exists a *Considered* mesh point $\boldsymbol{x_i}$ adjacent to both $\boldsymbol{x_j}$ and $\boldsymbol{x_k}$. For each *Considered* mesh point $\boldsymbol{x}$ we define the "near front" as the part of AF "relevant to $\boldsymbol{x}$":

$$\mathrm{NF}(\boldsymbol{x}) = \{ \boldsymbol{x_j x_k} \in AF \mid \exists \tilde{\boldsymbol{x}} \text{ on } \boldsymbol{x_j x_k} \text{ s.t. } \|\tilde{\boldsymbol{x}} - \boldsymbol{x}\| \leq h\Upsilon \}.$$

[9] Since Γ generally is not a level set of u, the logic of the method is more subtle and cannot really be based on Lemma 3.4. Instead, it relies on Lemma 3.5, which provides a weaker version of this inequality, but for any Γ "well-resolved" by an underlying mesh X.

www.manaraa.com

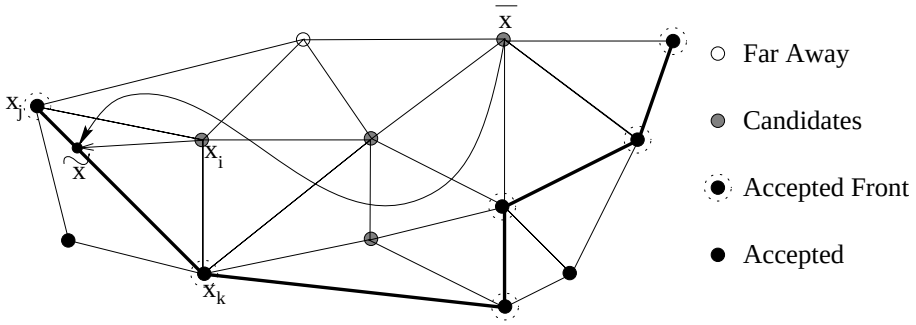


FIG. 3. *The AcceptedFront and the Considered mesh points. Segments of AF are shown in bold. The optimal trajectory for* $\bar{\boldsymbol{x}}$ *cannot intersect AF too far away from* $\bar{\boldsymbol{x}}$*, for if* $\|\tilde{\boldsymbol{x}} - \bar{\boldsymbol{x}}\| > h\Upsilon$*, then* $u(\boldsymbol{x_i}) < u(\bar{\boldsymbol{x}})$.

1. Start with all the mesh points in Far.
2. Move the mesh points on the boundary ($\boldsymbol{y} \in \partial\Omega$) to *Accepted* ($U(\boldsymbol{y}) = q(\boldsymbol{y})$).
3. Move all the mesh points $\boldsymbol{x}$ adjacent to the boundary into *Considered* and evaluate the tentative values

$$V(\boldsymbol{x}) := \min_{\boldsymbol{x_j}\boldsymbol{x_k} \in \mathrm{NF}(\boldsymbol{x})} V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x}). \tag{32}$$

4. Find the mesh point $\bar{\boldsymbol{x}}$ with the smallest value of V among all the *Considered*.
5. Move $\bar{\boldsymbol{x}}$ to *Accepted* ($U(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}})$) and update the *AcceptedFront*.
6. Move the Far mesh points adjacent to $\bar{\boldsymbol{x}}$ into *Considered* and compute their tentative values by (32).
7. Recompute the value for all the other *Considered* $\boldsymbol{x}$ such that $\bar{\boldsymbol{x}} \in \mathrm{NF}(\boldsymbol{x})$

$$V(\boldsymbol{x}) := \min\left\{V(\boldsymbol{x}), \min_{\bar{\boldsymbol{x}}\boldsymbol{x_i} \in \mathrm{NF}(\boldsymbol{x})} V_{\bar{\boldsymbol{x}},\boldsymbol{x_i}}(\boldsymbol{x})\right\}. \tag{33}$$

8. If *Considered* is not empty, then go to 4.

We note that the resulting algorithm

- is "single-pass," since it produces the numerical solution U in $O(\Upsilon^2 M \log(M))$ steps. This is because there are a total of M points to *Accept*, every time a mesh point is accepted there are at most Υ^2 *Considered* points to re-evaluate, and we must maintain an ordering of *Considered* based on V, which contributes a factor of $\log(M)$.
- produces the numerical solution U that converges to u as the diameter of the mesh tends to zero (see the proof in section 7);
- is at most first-order accurate;
- works equally well on acute and nonacute triangulated meshes;
- is applicable for a general anisotropic optimal trajectory problem described in section 3.

An extension of this method to R^n and manifolds is straightforward, since the update formula (31) can easily be generalized for these cases. The only part of the program which needs to be modified to handle a manifold-approximating mesh is the algorithm for sorting and searching the *AcceptedFront*.

Remark 6.2 (Comments on computational complexity).

1. In the above complexity analysis, the calculation of an upwind-update-from-a-single-simplex value $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$ was counted as a single operation. We note that

www.manaraa.com

the optimization problem solved to compute $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$ is *local* (i.e., $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$ can be computed independently from any other $V_{\boldsymbol{x_i},\boldsymbol{x_m}}(\boldsymbol{x_l})$) and, thus, should not be confused with the iterations necessary to solve the coupled system of nonlinear equations (25) simultaneously. More details on algorithmic efficiency can be found in section 9.

2. As we will show in section 7, AF can be considered as an approximation for a level set of U. Thus, if the mesh diameter h is sufficiently small, then the number of *Considered* points which have to be updated after each acceptance becomes closer to Υ, since the *Considered* points are immediately adjacent to AF. Thus, as h decreases, the computational complexity of the method tends to $O(\Upsilon M \log(M))$.

3. If the problem is formulated in R^n, the complexity of the corresponding algorithm is $O(\Upsilon^{n-1} M \log(M))$, where M remains the total number of mesh points.

Remark 6.3 (A comment on the rate of convergence). Our proof of convergence in section 7 does not provide an estimate for the rate of convergence. We believe that this method is first-order; this belief is based on the first-order accuracy of the approximations behind the semi-Lagrangian discretization (used to calculate $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$) and is also confirmed by the numerical evidence (see section 9.4 and [38, 46]). Based on numerical experiments, we note that for sufficiently small h, $\|U-u\|_\infty$ is at worst $O(\Upsilon h)$. This is not surprising, since Υh is the largest distance over which the first-order accurate approximation might be performed when $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$ is computed.

Remark 6.4 (A comment on mesh degeneracy). The fast Eikonal solvers described in section 4 rely on the *causality property*, which holds only for the acute simplexes. An additional "splitting section" construction is required to handle the nonacute case [21, 38]. Not surprisingly, the acuteness of simplexes in X is not required for the OUM introduced here. After all, the algorithm uses the mesh connectivity only to determine what becomes *Considered*, when a new mesh point is *Accepted*. All of the upwind-update-from-a-single-simplex values $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$ are computed from the simplexes defined by the position of *AcceptedFront* rather than from the simplexes present in X.

Nevertheless, in order to prove the convergence of $U(\boldsymbol{x})$ to the viscosity solution, we will have to assume that the mesh X cannot be arbitrarily degenerate. Namely, we will assume that if h is the diameter of X and h_{min} is the smallest triangle height in X, then the ratio $\eta = h/h_{min}$ is bounded for all sufficiently small h. See the proof of Lemma 7.5 for details.

Remark 6.5 (A comment on the order of *Acceptance*). Unlike in Sethian's Fast Marching Methods or in Tsitsiklis' algorithm, in the above method the mesh points *are not Accepted* in the order of increasing U. As was pointed out in section 5, for the anisotropic optimal trajectory problems the fact that the characteristic for $\boldsymbol{x}$ lies inside the simplex $\boldsymbol{x x_j x_k}$ does not mean that the gradient is pointing from that simplex. Thus, it is entirely possible that $U(\boldsymbol{x}) \le V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x}) < U(\boldsymbol{x_j})$. Nevertheless, we will show in Lemma 7.3 that the order of *Acceptance* is monotone, albeit in a much weaker sense than for the single-pass Eikonal solvers.

Remark 6.6 (Decoupling of the "extended semi-Lagrangian scheme"). Define the extended set of neighbors

$$N_K(\boldsymbol{x}) = \{\boldsymbol{x_1 x_2} \in X \mid \boldsymbol{x_1} \text{ and } \boldsymbol{x_2} \text{ are adjacent and } \exists \tilde{\boldsymbol{x}} \text{ on } \boldsymbol{x_1 x_2} \text{ s.t. } \|\tilde{\boldsymbol{x}} - \boldsymbol{x}\| \le h\Upsilon\}.$$

Note that if we replace $\mathrm{NF}(\boldsymbol{x})$ by $N_K(\boldsymbol{x})$, the formula (32) becomes

$$U(\boldsymbol{x}) = \min_{\boldsymbol{x_1 x_2} \in N_K(\boldsymbol{x})} V_{\boldsymbol{x_1 x_2}}(\boldsymbol{x}). \tag{34}$$

www.manaraa.com

This is an "extended" version of the semi-Lagrangian scheme (25), and it is easy to show that its solution U converges to the viscosity solution u. Equation (34) can be solved by successive approximation techniques described in [18], for example. However, a single-pass algorithm cannot be used to find U since we need to consider all possible directions of motion for the vehicle starting at the point $\boldsymbol{x}$ (i.e., $U(\boldsymbol{x})$ might potentially depend upon $U(\boldsymbol{y})$ for all $\boldsymbol{y} \in N_K(\boldsymbol{x})$, including the values $U(\boldsymbol{y}) > U(\boldsymbol{x})$). Therefore, the formula (32) can be interpreted as an upwinding analogue of (34).

The above comparison is just an analogue—not an equivalence. The numerical values produced by executing the above OUM will be different from those obtained by solving the coupled system (34); thus, the convergence of the OUM has to be proven separately.

7. Proof of convergence. In this section we prove the convergence of the above algorithm to the viscosity solution.[10]

We will assume that the numerical solution $U(\boldsymbol{x})$ is computed for each $x \in X$, using the OUM described in section 6. For the points $\boldsymbol{x} \in \Omega \backslash X$, $U(\boldsymbol{x})$ is defined by linear interpolation as follows.

If $\boldsymbol{x}$ is inside Ω but is not a mesh point, then it lies in some simplex $\boldsymbol{x_1 x_2 x_3}$. In that case, there exist $\zeta_1, \zeta_2, \zeta_3 \geq 0$ such that

$$\zeta_1 + \zeta_2 + \zeta_3 = 1, \qquad \zeta_1 \boldsymbol{x_1} + \zeta_2 \boldsymbol{x_2} + \zeta_3 \boldsymbol{x_3} = \boldsymbol{x}. \tag{35}$$

The value at $\boldsymbol{x}$ is defined to be $U(\boldsymbol{x}) = \zeta_1 U(\boldsymbol{x_1}) + \zeta_2 U(\boldsymbol{x_2}) + \zeta_3 U(\boldsymbol{x_3})$.

Suppose that h_{min} is the smallest triangle height in the mesh X. We will use the constant $\eta = \frac{h}{h_{min}}$ to characterize the degree of "degeneracy" of the mesh X.

7.1. Properties of the numerical solution. The following lemmas demonstrate several properties of the numerical solution U, which are necessary to prove the convergence and also mirror the properties of the value function for the optimal trajectory problem (section 3.2).

7.1.1. Is NF($\boldsymbol{x}$) big enough? Suppose that the mesh point $\bar{\boldsymbol{x}}$ is about to be *Accepted* (hence, $V(\bar{\boldsymbol{x}}) = \min_{\boldsymbol{x} \in Considered} V(\boldsymbol{x})$).

LEMMA 7.1. *For every Considered mesh point* $\boldsymbol{x}$ *define*

$$W(\boldsymbol{x}) = \min_{\boldsymbol{x_1 x_2} \in AF} \min_{\zeta \in [0,1]} \left\{ \frac{\tau(\zeta)}{f(\boldsymbol{x}, \boldsymbol{a})} + (\zeta U(\boldsymbol{x_1}) + (1-\zeta) U(\boldsymbol{x_2})) \right\}, \tag{36}$$

where $\tau(\zeta) = \|(\zeta \boldsymbol{x_1} + (1-\zeta)\boldsymbol{x_2}) - \boldsymbol{x}\|$ *and* $\boldsymbol{a} = \frac{\zeta(\boldsymbol{x_1} - \boldsymbol{x}) + (1-\zeta)(\boldsymbol{x_2} - \boldsymbol{x})}{\tau(\zeta)}$. *If* $\bar{\boldsymbol{x}}$ *is about to be* Accepted, *then* $U(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}}) = W(\bar{\boldsymbol{x}})$.

Proof. First, $U(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}})$ simply because $\bar{\boldsymbol{x}}$ is about to be *Accepted*.

Recall that $V(\boldsymbol{x})$ for every *Considered* mesh point $\boldsymbol{x}$ is computed by formula (32) as follows:

$$V(\boldsymbol{x}) = \min_{\boldsymbol{x_1 x_2} \in \mathrm{NF}(\boldsymbol{x})} \min_{\zeta \in [0,1]} \left\{ \frac{\tau(\zeta)}{f(\boldsymbol{x}, \boldsymbol{a})} + (\zeta U(\boldsymbol{x_1}) + (1-\zeta) U(\boldsymbol{x_2})) \right\},$$

[10] As of now, we do not know of any natural discretized version of the Hamilton–Jacobi–Bellman PDE that would be exactly satisfied by the numerical solution $U(\boldsymbol{x})$ produced by the OUM in section 6. Since $U(\boldsymbol{x})$ is defined constructively (i.e., by an algorithm to compute it), we cannot rely on properties of a discretized equation for the proof of convergence; thus, the proof in this section has to rely on the properties of the algorithm itself.

www.manaraa.com

where $\mathrm{NF}(\boldsymbol{x})$ is the part of the *AcceptedFront* "relevant to $\boldsymbol{x}$": $\mathrm{NF}(\boldsymbol{x}) = \{\boldsymbol{x_1x_2} \in AF \mid \exists \tilde{\boldsymbol{x}}$ on the line segment $\boldsymbol{x_1x_2}$ s.t. $\|\tilde{\boldsymbol{x}} - \boldsymbol{x}\| \leq h\Upsilon\}$. Since $\mathrm{NF}(\boldsymbol{x}) \subset AF$, we immediately see that for any *Considered* mesh point $\boldsymbol{x}$

$$V(\boldsymbol{x}) \geq W(\boldsymbol{x}). \tag{37}$$

Let $\boldsymbol{x_1x_2} \in AF$ and $\zeta \in [0,1]$ be such that the minimum in formula (36) is attained; i.e., if $\hat{\boldsymbol{x}} = (\zeta \boldsymbol{x_1} + (1-\zeta)\boldsymbol{x_2})$, then

$$W(\bar{\boldsymbol{x}}) = \frac{\|\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}\|}{f(\bar{\boldsymbol{x}}, \frac{\hat{\boldsymbol{x}}-\bar{\boldsymbol{x}}}{\|\hat{\boldsymbol{x}}-\bar{\boldsymbol{x}}\|})} + (\zeta U(\boldsymbol{x_1}) + (1-\zeta)U(\boldsymbol{x_2}))\,. \tag{38}$$

Let $\boldsymbol{x_3}$ be the *Considered* mesh point adjacent to both $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$. Then $U(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}}) \leq V(\boldsymbol{x_3})$ since $\bar{\boldsymbol{x}}$ is about to be accepted. $V(\boldsymbol{x_3})$ is also computed by formula (32); thus,

$$V(\boldsymbol{x_3}) \leq \frac{\|\hat{\boldsymbol{x}} - \boldsymbol{x_3}\|}{f(\boldsymbol{x_3}, \frac{\hat{\boldsymbol{x}}-\boldsymbol{x_3}}{\|\hat{\boldsymbol{x}}-\boldsymbol{x_3}\|})} + (\zeta U(\boldsymbol{x_1}) + (1-\zeta)U(\boldsymbol{x_2})) \leq \frac{h}{f_1} + (\zeta U(\boldsymbol{x_1}) + (1-\zeta)U(\boldsymbol{x_2}))\,.$$

Combining this with the inequalities (37) and(38), we obtain

$$\begin{aligned}\frac{\|\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}\|}{f_2} + (\zeta U(\boldsymbol{x_1}) + (1-\zeta)U(\boldsymbol{x_2})) \leq W(\bar{\boldsymbol{x}}) \leq V(\bar{\boldsymbol{x}}) &\leq V(\boldsymbol{x_3}) \\ &\leq \frac{h}{f_1} + (\zeta U(\boldsymbol{x_1}) + (1-\zeta)U(\boldsymbol{x_2}))\,,\end{aligned}$$

which implies $\|\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}\| \leq h\Upsilon$. Therefore, $\boldsymbol{x_1x_2} \in \mathrm{NF}(\bar{\boldsymbol{x}})$ and $W(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}})$. ∎

7.1.2. Uniform upper bound.

LEMMA 7.2. *If Ω is convex and $d(\boldsymbol{x})$ is the distance from $\boldsymbol{x} \in \Omega$ to the boundary $\partial\Omega$, then*

$$U(\boldsymbol{x}) \leq \frac{d(\boldsymbol{x})}{f_1} + q_2. \tag{39}$$

Proof. If $\boldsymbol{x} \in \partial\Omega$, then the inequality holds trivially since $0 \leq q(\boldsymbol{x}) \leq q_2$.

If $\boldsymbol{x}$ is a mesh point inside Ω, we prove the lemma by induction: assume that the inequality (39) holds for all the mesh points that are on the *AcceptedFront* just before $\boldsymbol{x} = \bar{\boldsymbol{x}}$ is *Accepted*. Consider a (possibly nonunique) shortest path from $\bar{\boldsymbol{x}}$ to the boundary. By the properties of the distance function $d(\cdot)$, that shortest path is a straight line. Moreover, suppose that line intersects the segment of *AcceptedFront* $\boldsymbol{x_1x_2} \in AF$ at the point $\hat{\boldsymbol{x}} = (\zeta \boldsymbol{x_1} + (1-\zeta)\boldsymbol{x_2})$. It is trivial to show that $d(\bar{\boldsymbol{x}}) = d(\hat{\boldsymbol{x}}) + \|\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}\|$. Using Lemma 7.1,

$$U(\bar{\boldsymbol{x}}) \;=\; W(\bar{\boldsymbol{x}}) \;\leq\; \frac{\|\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}\|}{f(\bar{\boldsymbol{x}}, \frac{\hat{\boldsymbol{x}}-\bar{\boldsymbol{x}}}{\|\hat{\boldsymbol{x}}-\bar{\boldsymbol{x}}\|})} + (\zeta U(\boldsymbol{x_1}) + (1-\zeta)U(\boldsymbol{x_2}))\,.$$

Based on the assumption of induction,

$$U(\bar{\boldsymbol{x}}) \;\leq\; \frac{\|\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}\|}{f_1} + \zeta\left(\frac{d(\boldsymbol{x_1})}{f_1} + q_2\right) + (1-\zeta)\left(\frac{d(\boldsymbol{x_2})}{f_1} + q_2\right).$$

www.manaraa.com

By the convexity of Ω, the distance-to-boundary function $d(\boldsymbol{x})$ is concave and $d(\hat{\boldsymbol{x}}) \geq \zeta d(\boldsymbol{x_1}) + (1-\zeta) d(\boldsymbol{x_2})$. Therefore,

$$U(\bar{\boldsymbol{x}}) \;\leq\; \frac{1}{f_1}(\|\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}\| + d(\hat{\boldsymbol{x}})) + q_2 \;=\; \frac{d(\bar{\boldsymbol{x}})}{f_1} + q_2,$$

which completes the proof by induction. (The base of induction is obvious, since only the mesh points on the boundary $\partial\Omega$ are already *Accepted* when the algorithm starts.)

If $\boldsymbol{x}$ is inside Ω but is not a mesh point, then it lies in some simplex $\boldsymbol{x_1 x_2 x_3}$ and there exist $\zeta_1, \zeta_2, \zeta_3 \geq 0$ such that

(40)
$$\zeta_1 + \zeta_2 + \zeta_3 = 1, \quad \boldsymbol{x} = \zeta_1 \boldsymbol{x_1} + \zeta_2 \boldsymbol{x_2} + \zeta_3 \boldsymbol{x_3}, \quad U(\boldsymbol{x}) = \zeta_1 U(\boldsymbol{x_1}) + \zeta_2 U(\boldsymbol{x_2}) + \zeta_3 U(\boldsymbol{x_3}).$$

Once again, using the concavity of the distance function,

$$\begin{aligned} U(\boldsymbol{x}) &\leq \zeta_1 \left(\frac{d(\boldsymbol{x_1})}{f_1} + q_2 \right) + \zeta_2 \left(\frac{d(\boldsymbol{x_2})}{f_1} + q_2 \right) + \zeta_3 \left(\frac{d(\boldsymbol{x_3})}{f_1} + q_2 \right) \\ &\leq q_2 + \frac{1}{f_1} \left(\zeta_1 d(\boldsymbol{x_1}) + \zeta_2 d(\boldsymbol{x_2}) + \zeta_3 d(\boldsymbol{x_3}) \right) \;\leq\; \frac{d(\boldsymbol{x})}{f_1} + q_2. \qquad \square \end{aligned}$$

The obtained bound is "uniform" since it is independent of the diameter h of the mesh X. We also note that a uniform upper bound on U can be derived even for a nonconvex Ω, assuming that η remains bounded and the boundary $\partial\Omega$ is adequately represented by the mesh as h tends to zero.

7.1.3. Relaxed monotonicity of the *Accepted*. In contrast with Dijkstra-like Eikonal solvers, the OUM introduced in section 6 is not computing (and accepting) the values in a monotone fashion: "$\boldsymbol{x_i}$ is *Accepted* after $\boldsymbol{x_j}$" does not imply "$U(\boldsymbol{x_i}) \geq U(\boldsymbol{x_j})$" (see Remark 6.5). However, a weaker monotonicity property can still be formulated, based on the evolution of AF during the computation. Recall that AF is defined as the set of the line segments $\boldsymbol{x_j x_k}$, where $\boldsymbol{x_j}$ and $\boldsymbol{x_k}$ are adjacent mesh points on the *AcceptedFront* such that there exists a *Considered* mesh point $\boldsymbol{x_i}$ adjacent to both $\boldsymbol{x_j}$ and $\boldsymbol{x_k}$. Define U^{AF}_{min} (and U^{AF}_{max}) as the min (max) value of U on the set AF. Note that, since U is defined by the linear interpolation, both U^{AF}_{min} and U^{AF}_{max} are attained at the mesh points.

The following definitions are useful for discussing the evolution of *AcceptedFront*:

— $AF_{\bar{\boldsymbol{x}}}$ is the state of AF *immediately before* $\bar{\boldsymbol{x}}$ is *Accepted*.

— $U^{AF_{\bar{\boldsymbol{x}}}}_{min}$ and $U^{AF_{\bar{\boldsymbol{x}}}}_{max}$ are the minimum and maximum values of U on $AF_{\bar{\boldsymbol{x}}}$.

— $AF^{\bar{\boldsymbol{x}}}$ is the state of AF *immediately after* $\bar{\boldsymbol{x}}$ is *Accepted*.

— $U^{AF^{\bar{\boldsymbol{x}}}}_{min}$ and $U^{AF^{\bar{\boldsymbol{x}}}}_{max}$ are the minimum and maximum values of U on $AF^{\bar{\boldsymbol{x}}}$.

LEMMA 7.3 (Monotonicity of AF's evolution). *Suppose that h_{min} is the smallest triangle height in the triangulated mesh X on Ω. Then the following weak monotonicity results hold for the numerical solution U:*

(i)

$$U^{AF_{\bar{\boldsymbol{x}}}}_{min} + \frac{h_{min}}{f_2} \;\leq\; U(\bar{\boldsymbol{x}}) \;\leq\; U^{AF_{\bar{\boldsymbol{x}}}}_{max} + \frac{h}{f_1}. \tag{41}$$

(ii)

$$U^{AF_{\bar{\boldsymbol{x}}}}_{min} \;\leq\; U^{AF^{\bar{\boldsymbol{x}}}}_{min}. \tag{42}$$

www.manaraa.com

(iii) *If* $\boldsymbol{x_i}$ *is Accepted before* $\boldsymbol{x_j}$*, then* $U_{min}^{AF_{\boldsymbol{x_i}}} \leq U_{min}^{AF_{\boldsymbol{x_j}}}$.

(iv) *If* $U_{max}^{AF_{\bar{\boldsymbol{x}}}} \leq U_{min}^{AF_{\bar{\boldsymbol{x}}}} + \frac{h}{f_1}$*, then* $U_{max}^{AF^{\bar{\boldsymbol{x}}}} \leq U_{min}^{AF^{\bar{\boldsymbol{x}}}} + \frac{h}{f_1}$.

Proof. (i) Let $\boldsymbol{x_1}$ be a mesh point on $AF_{\bar{\boldsymbol{x}}}$ such that $U(\boldsymbol{x_1}) = U_{min}^{AF_{\bar{\boldsymbol{x}}}}$. Since it is on *AcceptedFront* immediately before $\bar{\boldsymbol{x}}$ is *Accepted*, there exists at that time a *Considered* mesh point $\boldsymbol{x_2}$ adjacent to $\boldsymbol{x_1}$. Thus,

$$V(\boldsymbol{x_2}) \leq U(\boldsymbol{x_1}) + \frac{\|\boldsymbol{x_2} - \boldsymbol{x_1}\|}{f(\boldsymbol{x_2}, \frac{\boldsymbol{x_2}-\boldsymbol{x_1}}{\|\boldsymbol{x_2}-\boldsymbol{x_1}\|})}.$$

Since $\bar{\boldsymbol{x}}$ is about to be *Accepted*, $U(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}}) \leq V(\boldsymbol{x_2}) \leq U_{min}^{AF_{\bar{\boldsymbol{x}}}} + \frac{h}{f_1}$. On the other hand,

$$U(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}}) = U(\tilde{\boldsymbol{x}}) + \frac{\|\tilde{\boldsymbol{x}} - \bar{\boldsymbol{x}}\|}{f(\bar{\boldsymbol{x}}, \frac{\tilde{\boldsymbol{x}}-\bar{\boldsymbol{x}}}{\|\tilde{\boldsymbol{x}}-\bar{\boldsymbol{x}}\|})}$$

for some $\tilde{\boldsymbol{x}} \in AF_{\bar{\boldsymbol{x}}}$. Thus, $U(\bar{\boldsymbol{x}}) \geq U_{min}^{AF_{\bar{\boldsymbol{x}}}} + \frac{h_{min}}{f_2}$.

(ii) As $\bar{\boldsymbol{x}}$ is *Accepted*, several mesh points might be removed from the *AcceptedFront*, but the only point possibly added to the *AcceptedFront* is $\bar{\boldsymbol{x}}$ itself. ($\bar{\boldsymbol{x}}$ will be added if there still is a not-yet-*Accepted* mesh point adjacent to it.) Since $U_{min}^{AF_{\bar{\boldsymbol{x}}}} \leq U(\bar{\boldsymbol{x}})$, it follows that $U_{min}^{AF_{\bar{\boldsymbol{x}}}} \leq U_{min}^{AF^{\bar{\boldsymbol{x}}}}$.

(iii) This point follows trivially by induction from the inequality (42).

(iv) Since $\bar{\boldsymbol{x}}$ is the only point possibly added to the *AcceptedFront*,

$$U_{max}^{AF^{\bar{\boldsymbol{x}}}} \leq \max\left(U_{max}^{AF_{\bar{\boldsymbol{x}}}}, U(\bar{\boldsymbol{x}})\right) \leq U_{min}^{AF_{\bar{\boldsymbol{x}}}} + \frac{h}{f_1} \leq U_{min}^{AF^{\bar{\boldsymbol{x}}}} + \frac{h}{f_1}. \qquad \square$$

Remark 7.4. It immediately follows from the above Lemma that if $q_2 \leq q_1 + h/f_1$, then $U_{max}^{AF} \leq U_{min}^{AF} + h/f_1$ at all times. Thus, if the exit time-penalty q is approximately constant on $\partial\Omega$, then the AF will be approximately a level set of U throughout the computation. Moreover, even if q is not approximately constant, the AF will still approximate a level set of U as soon as U_{min}^{AF} becomes bigger than $(q_2 - h/f_1)$.

7.1.4. Uniform Lipschitz-continuity.

LEMMA 7.5. (i) *Let* $L_1 = \eta/f_1$*. If* $\boldsymbol{x_1}$ *and* $\boldsymbol{x_2}$ *are two adjacent mesh points inside* Ω*, then*

$$|U(\boldsymbol{x_1}) - U(\boldsymbol{x_2})| \leq L_1\|\boldsymbol{x_1} - \boldsymbol{x_2}\|. \tag{43}$$

(ii) *Let* $L_2 = \eta L_1$*. If* $\nabla U(\boldsymbol{x})$ *is defined for some* $\boldsymbol{x} \in \Omega\backslash\partial\Omega$ *such that* $d(\boldsymbol{x}) > h$ *(i.e.,* $\boldsymbol{x}$ *is not in a simplex immediately adjacent to* $\partial\Omega$*), then*

$$\|\nabla U(\boldsymbol{x})\| \leq L_2. \tag{44}$$

(iii) *Finally, for arbitrary points* $\boldsymbol{x_1}, \boldsymbol{x_2} \in \Omega$*,*

$$|U(\boldsymbol{x_1}) - U(\boldsymbol{x_2})| \leq L_2\|\boldsymbol{x_1} - \boldsymbol{x_2}\|. \tag{45}$$

Proof. (i) Suppose that $\boldsymbol{x_1}, \boldsymbol{x_2} \in \Omega\backslash\partial\Omega$ are two adjacent mesh points. Without loss of generality, assume that $\boldsymbol{x_1}$ was *Accepted* before $\boldsymbol{x_2}$. Thus, immediately before $\boldsymbol{x_2}$ is *Accepted*, $\boldsymbol{x_1}$ will still be on the *AcceptedFront* and

$$U(\boldsymbol{x_2}) \leq \frac{\|\boldsymbol{x_1} - \boldsymbol{x_2}\|}{f(\boldsymbol{x_2}, \frac{\boldsymbol{x_1}-\boldsymbol{x_2}}{\|\boldsymbol{x_1}-\boldsymbol{x_2}\|})} + U(\boldsymbol{x_1}) \leq \frac{\|\boldsymbol{x_1} - \boldsymbol{x_2}\|}{f_1} + U(\boldsymbol{x_1}) \leq L_1\|\boldsymbol{x_1} - \boldsymbol{x_2}\| + U(\boldsymbol{x_1}).$$

www.manaraa.com

Since U's are not necessarily *Accepted* in the ascending order, it is not generally true that $U(\boldsymbol{x_2}) \geq U(\boldsymbol{x_1})$, but from Lemma 7.3,

$$\begin{aligned} U(\boldsymbol{x_1}) &\leq U_{min}^{AF_{\boldsymbol{x_1}}} + \frac{h}{f_1} \leq U_{min}^{AF_{\boldsymbol{x_2}}} + \frac{h}{f_1} \leq U(\boldsymbol{x_2}) + \frac{h}{f_1} \\ &= U(\boldsymbol{x_2}) + \eta \frac{h_{min}}{f_1} \leq U(\boldsymbol{x_2}) + L_1 \|\boldsymbol{x_1} - \boldsymbol{x_2}\|, \end{aligned}$$

which concludes the proof of inequality (43).

(ii) Let $\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_3}$ be the vertices of the simplex in the mesh X which contains $\boldsymbol{x}$. Since $d(\boldsymbol{x}) > h$, we know that $\boldsymbol{x_1}$, $\boldsymbol{x_2}$, and $\boldsymbol{x_3}$ are also inside Ω, i.e., not on the boundary. Inside each simplex, U is defined by the linear interpolation, and ∇U is a constant. Whatever the direction of ∇U, a straight line parallel to it passes through one of the vertices and intersects the opposite side of the triangle. Without loss of generality, assume that that line passes through $\boldsymbol{x_1}$ and intersects the side $\boldsymbol{x_2 x_3}$ at the point $\boldsymbol{x_4}$. Since $\boldsymbol{x_4}$ lies on $\boldsymbol{x_2 x_3}$, either $(\boldsymbol{x_2} - \boldsymbol{x_1}) \cdot (\boldsymbol{x_4} - \boldsymbol{x_1}) \geq \|\boldsymbol{x_4} - \boldsymbol{x_1}\|^2$ or $(\boldsymbol{x_3} - \boldsymbol{x_1}) \cdot (\boldsymbol{x_4} - \boldsymbol{x_1}) \geq \|\boldsymbol{x_4} - \boldsymbol{x_1}\|^2$. Without loss of generality, assume the latter. Since $\|\boldsymbol{x_4} - \boldsymbol{x_1}\| \geq h_{min}$,

$$\|\nabla U\| h_{min} \leq \|\nabla U\| \|\boldsymbol{x_4} - \boldsymbol{x_1}\| = |U(\boldsymbol{x_4}) - U(\boldsymbol{x_1})| \leq |U(\boldsymbol{x_3}) - U(\boldsymbol{x_1})| \leq L_1 h.$$

Thus, $\|\nabla U\| \leq L_1 \frac{h}{h_{min}} = L_2$.

(iii) This point is obvious, since U is piecewise linear, with the slope bounded by L_2 in every simplex. □

Remark 7.6. Better estimates of L_1 and L_2 can be derived if f and q are smooth and h is sufficiently small. However, to prove the uniform convergence of U to the value function u, it is just necessary to show that some such L_2 independent of h does indeed exist. The dependence of L_2 upon η is not dangerous: if the triangulated mesh X_r does not become more and more "degenerate" as $h_r \to 0$, then η_r will be bounded.

7.2. Convergence to a viscosity solution.

THEOREM 7.7. *Consider a sequence of meshes* $\{X_r\}$ *such that* $h_r \to 0$ *but* $\eta^r = \frac{h_r}{h_{r_{min}}} < \eta$ *as* $r \to \infty$. *Let* U^r *be the approximate solution obtained on the mesh* X_r *by the algorithm described in section* 6. *As* $h_r \to 0$, U^r *uniformly converges to the viscosity solution of* (22) *(defined by the inequalities* (23) *and* (24) *in section* 3.3*).*

Proof. Since $\{U^r\}$ are bounded and uniformly Lipschitz-continuous, by the Arzela–Ascoli theorem, there exists a subsequence $\{X_p\}$ of the sequence $\{X_r\}$ such that $h_p \to 0$ as $p \to \infty$, and a function u such that $U^p \to u$ uniformly as $p \to \infty$. Boundedness and uniform continuity of u immediately follow from the properties of U^p.

(i) Consider any function $\phi \in C_c^\infty(\Omega)$, and suppose that $(u - \phi)$ has a *strict local minimum* at $\boldsymbol{x_0} \in \Omega$. Define B_δ to be the closed ball of radius δ around $\boldsymbol{x_0}$. Then there exists some $\delta > 0$ such that $B_\delta \subset \Omega$ and $\boldsymbol{x} \in B_\delta$ implies

$$(u - \phi)(\boldsymbol{x_0}) < (u - \phi)(\boldsymbol{x}). \tag{46}$$

If $D_2(\boldsymbol{x})$ is the matrix of second derivatives of $\phi(\boldsymbol{x})$, then there exists $\mu > 0$ such that $\|D_2(\boldsymbol{x})\|_2 \leq \mu$ for all $\boldsymbol{x} \in B_\delta$. Now let $\boldsymbol{x_0}^p$ be a minimum point for $(U^p - \phi)$ over B_δ; from (46) and from the uniform convergence of U^p's it follows that

$$\lim_{p \to \infty} \boldsymbol{x_0}^p = \boldsymbol{x_0}. \tag{47}$$

www.manaraa.com

If $\boldsymbol{x_0}^p$ is not a mesh point of X_p and lies in the interior of some simplex s, we define $\boldsymbol{x_1}^p$ to be the vertex of s closest to it. If $\boldsymbol{x_0}^p$ lies on the edge of a simplex, we take $\boldsymbol{x_1}^p$ to be the closest endpoint of that edge; finally, we use $\boldsymbol{x_1}^p = \boldsymbol{x_0}^p$ if $\boldsymbol{x_0}^p$ is a mesh point. In any case, since ϕ is smooth and U^p is linear on every simplex, we know that $U^p(\boldsymbol{x_1}^p) - U^p(\boldsymbol{x_0}^p) = \nabla\phi(\boldsymbol{x_0}^p)\cdot(\boldsymbol{x_1}^p - \boldsymbol{x_0}^p)$ and

$$(U^p - \phi)(\boldsymbol{x_1}^p) \le (U^p - \phi)(\boldsymbol{x_0}^p) + \frac{\mu {h_p}^2}{2}. \tag{48}$$

Moreover, using inequality (48), we obtain for every $\boldsymbol{x} \in B_\delta$

$$\begin{aligned}\phi(\boldsymbol{x}) - \phi(\boldsymbol{x_1}^p) &= (\phi(\boldsymbol{x}) - \phi(\boldsymbol{x_0}^p)) + (\phi(\boldsymbol{x_0}^p) - \phi(\boldsymbol{x_1}^p)) \\ &\le (U^p(\boldsymbol{x}) - U^p(\boldsymbol{x_0}^p)) + \left(U^p(\boldsymbol{x_0}^p) - U^p(\boldsymbol{x_1}^p) + \frac{\mu {h_p}^2}{2}\right) = U^p(\boldsymbol{x}) - U^p(\boldsymbol{x_1}^p) + \frac{\mu {h_p}^2}{2}.\end{aligned} \tag{49}$$

Since $\|\boldsymbol{x_0}^p - \boldsymbol{x_1}^p\| \le h_p$, it is also clear that $\lim_{p\to\infty} \boldsymbol{x_1}^p = \boldsymbol{x_0}$. So, for big enough p, $h_p\Upsilon \le \delta$; thus, by the update formula (31), there exists $\tilde{\boldsymbol{x}}^{\boldsymbol{p}} \in AF_{\boldsymbol{x_1}^p} \bigcap B_\delta$ such that

$$U^p(\boldsymbol{x_1}^p) = \frac{\tau_p}{f(\boldsymbol{x_1}^p, \boldsymbol{a}^p)} + U^p(\tilde{\boldsymbol{x}}^{\boldsymbol{p}}), \tag{50}$$

where $\tau_p = \|\tilde{\boldsymbol{x}}^p - \boldsymbol{x_1}^p\|$ and $\boldsymbol{a}^p = \frac{\tilde{\boldsymbol{x}}^p - \boldsymbol{x_1}^p}{\|\tilde{\boldsymbol{x}}^p - \boldsymbol{x_1}^p\|}$.

Using the smoothness of ϕ, the inequality (49), and the equality (50), we obtain

$$\begin{aligned}\nabla\phi(\boldsymbol{x_1}^p)\cdot\boldsymbol{a}^p + \frac{1}{f(\boldsymbol{x_1}^p, \boldsymbol{a}^p)} &\le \frac{\phi(\boldsymbol{x_1}^p + \tau_p\boldsymbol{a}^p) - \phi(\boldsymbol{x_1}^p)}{\tau_p} + \frac{1}{f(\boldsymbol{x_1}^p, \boldsymbol{a}^p)} + \tau_p\mu \\ &\le \frac{U^p(\boldsymbol{x_1}^p + \tau_p\boldsymbol{a}^p) - U^p(\boldsymbol{x_1}^p) + (\mu {h_p}^2/2)}{\tau_p} + \frac{1}{f(\boldsymbol{x_1}^p, \boldsymbol{a}^p)} + \tau_p\mu \\ &= \frac{U^p(\tilde{\boldsymbol{x}}^{\boldsymbol{p}}) - U^p(\boldsymbol{x_1}^p) + \tau_p/f(\boldsymbol{x_1}^p, \boldsymbol{a}^p)}{\tau_p} + \frac{\mu {h_p}^2}{2\tau_p} + \tau_p\mu = \frac{\mu {h_p}^2}{2\tau_p} + \tau_p\mu.\end{aligned} \tag{51}$$

Since $\tilde{\boldsymbol{x}}$ lies on the $AF_{\boldsymbol{x_1}^p}$ and $\tau_p = \|\tilde{\boldsymbol{x}}^p - \boldsymbol{x_1}^p\|$, it is at least as big as the minimal triangle height in the mesh X_p, i.e., $\tau_p \ge \frac{h_p}{\eta}$. On the other hand, $\tau_p \le h_p\Upsilon$ because $\tilde{\boldsymbol{x}}^p \in \mathrm{NF}(\boldsymbol{x_1}^p)$. Combining these bounds with inequality (51), we see that

$$\nabla\phi(\boldsymbol{x_1}^p)\cdot\boldsymbol{a}^p + \frac{1}{f(\boldsymbol{x_1}^p, \boldsymbol{a}^p)} \le \mu\left(\frac{\eta}{2} + \Upsilon\right)h_p. \tag{52}$$

The sequence $\{\boldsymbol{a}^p\}$ has to have a subsequence converging to some vector $\boldsymbol{b} \in S_1$; we restrict our attention to that subsequence, but will still use the subscript p to avoid further cluttering of the notation. Now we can use the continuity of f, the smoothness of ϕ, and the uniformity of convergence of U^p to pass to a limit as $p \to \infty$ in the inequality (52):

$$\nabla\phi(\boldsymbol{x_0})\cdot\boldsymbol{b} + \frac{1}{f(\boldsymbol{x_0}, \boldsymbol{b})} \le 0 \implies (\nabla\phi(\boldsymbol{x_0})\cdot\boldsymbol{b})f(\boldsymbol{x_0}, \boldsymbol{b}) + 1 \le 0,$$

which completes the first half of the proof, since

$$\min_{\boldsymbol{a}\in S_1}\{(\nabla\phi(\boldsymbol{x_0})\cdot\boldsymbol{a})f(\boldsymbol{x_0}, \boldsymbol{a})\} \le (\nabla\phi(\boldsymbol{x_0})\cdot\boldsymbol{b})f(\boldsymbol{x_0}, \boldsymbol{b}).$$

www.manaraa.com

(ii) Consider any function $\phi \in C_c^\infty(\Omega)$, and suppose that $(u-\phi)$ has a *strict local maximum* at $\boldsymbol{x_0} \in \Omega$. Define B_δ to be the closed ball of radius δ around $\boldsymbol{x_0}$. Then there exists some $\delta > 0$ such that $B_\delta \subset \Omega$ and $\boldsymbol{x} \in B_\delta$ implies

$$(u-\phi)(\boldsymbol{x_0}) > (u-\phi)(\boldsymbol{x}). \tag{53}$$

If $\nabla\phi(\boldsymbol{x_0}) = 0$, then the inequality (24) is trivially satisfied. Thus, we will further assume that $\|\nabla\phi(x)\| \geq \nu > 0$ for all $\boldsymbol{x} \in B_\delta$. If $D_2(\boldsymbol{x})$ is the matrix of second derivatives of $\phi(\boldsymbol{x})$, then there exists $\mu > 0$ such that $\|D_2(\boldsymbol{x})\|_2 \leq \mu$ for all $\boldsymbol{x} \in B_\delta$. Now let ${\boldsymbol{x_0}}^p$ be a maximum point for $(U^p - \phi)$ over B_δ; from (53) and from the uniform convergence of U^p's it follows that

$$\lim_{p\to\infty} {\boldsymbol{x_0}}^p = \boldsymbol{x_0}. \tag{54}$$

If ${\boldsymbol{x_0}}^p$ is not a mesh point of X_p and lies in the interior of some simplex s, we define ${\boldsymbol{x_1}}^p$ to be the vertex of s closest to it. If ${\boldsymbol{x_0}}^p$ lies on the edge of a simplex, we take ${\boldsymbol{x_1}}^p$ to be the closest endpoint of that edge; finally, we use ${\boldsymbol{x_1}}^p = {\boldsymbol{x_0}}^p$ if ${\boldsymbol{x_0}}^p$ is a mesh point. In any case, since ϕ is smooth and U^p is linear on every simplex, we know that $U^p({\boldsymbol{x_1}}^p) - U^p({\boldsymbol{x_0}}^p) = \nabla\phi({\boldsymbol{x_0}}^p)\cdot({\boldsymbol{x_1}}^p - {\boldsymbol{x_0}}^p)$ and

$$(U^p-\phi)({\boldsymbol{x_1}}^p) \geq (U^p-\phi)({\boldsymbol{x_0}}^p) - \frac{\mu {h_p}^2}{2}. \tag{55}$$

Moreover, using the inequality (55), we obtain for every $\boldsymbol{x} \in B_\delta$,

$$\begin{aligned}\phi(\boldsymbol{x}) - \phi({\boldsymbol{x_1}}^p) &= (\phi(\boldsymbol{x}) - \phi({\boldsymbol{x_0}}^p)) + (\phi({\boldsymbol{x_0}}^p) - \phi({\boldsymbol{x_1}}^p)) \\ &\geq (U^p(\boldsymbol{x}) - U^p({\boldsymbol{x_0}}^p)) + \left(U^p({\boldsymbol{x_0}}^p) - U^p({\boldsymbol{x_1}}^p) - \frac{\mu {h_p}^2}{2}\right) = U^p(\boldsymbol{x}) - U^p({\boldsymbol{x_1}}^p) - \frac{\mu {h_p}^2}{2}.\end{aligned} \tag{56}$$

Since $\|{\boldsymbol{x_0}}^p - {\boldsymbol{x_1}}^p\| \leq h_p$, it is also clear that $\lim_{p\to\infty} {\boldsymbol{x_1}}^p = \boldsymbol{x_0}$. As proven in section 3.4,

$$\min_{\boldsymbol{a}\in S_1}\{(\nabla\phi(\boldsymbol{x_0})\cdot\boldsymbol{a}) f(\boldsymbol{x_0},\boldsymbol{a})\} = \min_{\boldsymbol{a}\in S_1^{\phi,\boldsymbol{x_0}}}\{(\nabla\phi(\boldsymbol{x_0})\cdot\boldsymbol{a}) f(\boldsymbol{x_0},\boldsymbol{a})\}. \tag{57}$$

Thus, to prove (24), we need to consider only $\boldsymbol{a} \in S_1^{\phi,\boldsymbol{x_0}}$, i.e., only $\boldsymbol{a}$ such that

$$\boldsymbol{a}\cdot\nabla\phi(\boldsymbol{x_0}) \leq -\Upsilon^{-1}\|\nabla\phi(\boldsymbol{x_0})\| \leq -\nu\Upsilon^{-1}. \tag{58}$$

Suppose that a particular $\boldsymbol{a} \in S_1^{\phi,\boldsymbol{x_0}}$ was chosen. We would like to show that, for sufficiently small h_p,

(*) *if we start at ${\boldsymbol{x_1}}^p$ and go some distance $\tau_p = O(h_p)$ in the direction $\boldsymbol{a}$, then we will have to intersect the $AF_{{\boldsymbol{x_1}}^p}$.*

If the local maximum were attained at the mesh point (i.e., the case ${\boldsymbol{x_1}}^p = {\boldsymbol{x_0}}^p$) and the test function ϕ were linear, then (*) would be almost obvious: ϕ would be linearly decreasing in the direction $\boldsymbol{a}$, and so would U^p, because of the local maximum condition, and, as we know from Lemma 7.3,

$$U^p(x) \geq U_{min}^{AF_{{\boldsymbol{x_1}}^p}} + \frac{h_p}{\eta f_2}$$

www.manaraa.com

for every mesh point $x \in X_p$ *Accepted* after the point $\boldsymbol{x_1}^p$. Since ϕ is generally not linear and $\boldsymbol{x_1}^p \neq \boldsymbol{x_0}^p$, we will have to be more careful.

Suppose we start moving from $\boldsymbol{x_1}^p$ in the direction $\boldsymbol{a}$ for some time t_p. Using the inequality (56), the smoothness of ϕ, and the inequality (58), we obtain

$$\begin{aligned} U^p(\boldsymbol{x_1}^p + t_p\boldsymbol{a}) - U^p(\boldsymbol{x_1}^p) &\leq (\phi(\boldsymbol{x_1}^p + t_p\boldsymbol{a}) - \phi(\boldsymbol{x_1}^p)) + \frac{\mu {h_p}^2}{2} \\ &\leq \left(t_p(\nabla\phi(\boldsymbol{x_1}^p)\cdot\boldsymbol{a}) + \frac{\mu {t_p}^2}{2}\right) + \frac{\mu {h_p}^2}{2} \leq -\nu t_p \Upsilon^{-1} + \mu\frac{{h_p}^2 + {t_p}^2}{2}. \end{aligned} \tag{59}$$

In order to prove (*) we will need the following inequality to be satisfied:

$$-\nu t_p \frac{f_1}{f_2} + \mu\frac{{h_p}^2 + {t_p}^2}{2} \leq -\frac{h_p}{f_1}. \tag{60}$$

Let $t_p = Ah_p$. We will now show that the constant A can be chosen such that (60) holds for small enough h_p. Indeed, (60) can be rewritten as

$$h_p^2\left(\mu\frac{1+A^2}{2}\right) + h_p\left(\frac{1}{f_1} - \nu A\frac{f_1}{f_2}\right) \leq 0. \tag{61}$$

If A is such that $(\frac{1}{f_1} - \nu A\frac{f_1}{f_2}) > 0$, then we also have that (60) is satisfied for all $h_p \in [0, (\nu A\frac{f_1}{f_2} - f_2)\frac{2}{\mu(1+A^2)}]$. Thus, choosing any $A > (\frac{f_2^2}{\nu f_1})$, we ensure that (60) is satisfied for the sufficiently small h_p. Combining this with the inequality (59) and using the monotonicity result in Lemma 7.3, we see that

$$U^p(\boldsymbol{x_1}^p + t_p\boldsymbol{a}) \leq U^p(\boldsymbol{x_1}^p) - \frac{h_p}{f_1} \leq U_{min}^{AF_{\boldsymbol{x_1}^p}},$$

i.e., the point $(\boldsymbol{x_1}^p + t_p\boldsymbol{a})$ cannot be inside the $AF_{\boldsymbol{x_1}^p}$. Since $\boldsymbol{x_1}^p$ is inside $AF_{\boldsymbol{x_1}^p}$, that means that (*) holds: there exists some $\tau_p \in [0, t_p]$ such that

$$\tilde{\boldsymbol{x}}^{\boldsymbol{p}} = (\boldsymbol{x_1}^p + \tau_p\boldsymbol{a}) \in AF_{\boldsymbol{x_1}^p}.$$

By Lemma 7.1,

$$U^p(\boldsymbol{x_1}^p) = W^p(\boldsymbol{x_1}^p) \leq \frac{\tau_p}{f(\boldsymbol{x_1}^p, \boldsymbol{a})} + U^p(\tilde{\boldsymbol{x}}^{\boldsymbol{p}}). \tag{62}$$

The remainder of the proof is similar to what we have done to prove (i).

Using the smoothness of ϕ, the inequality (56), and the inequality (62), we obtain

$$\begin{aligned} \nabla\phi(\boldsymbol{x_1}^p)\cdot\boldsymbol{a} + \frac{1}{f(\boldsymbol{x_1}^p, \boldsymbol{a})} &\geq \frac{\phi(\boldsymbol{x_1}^p + \tau_p\boldsymbol{a}) - \phi(\boldsymbol{x_1}^p)}{\tau_p} + \frac{1}{f(\boldsymbol{x_1}^p, \boldsymbol{a})} - \tau_p\mu \\ &\geq \frac{U^p(\boldsymbol{x_1}^p + \tau_p\boldsymbol{a}) - U^p(\boldsymbol{x_1}^p) - (\mu {h_p}^2/2)}{\tau_p} + \frac{1}{f(\boldsymbol{x_1}^p, \boldsymbol{a})} - \tau_p\mu \\ &= \frac{U^p(\tilde{\boldsymbol{x}}^{\boldsymbol{p}}) - U^p(\boldsymbol{x_1}^p) + \tau_p/f(\boldsymbol{x_1}^p, \boldsymbol{a})}{\tau_p} - \frac{\mu {h_p}^2}{2\tau_p} - \tau_p\mu \geq -\frac{\mu {h_p}^2}{2\tau_p} - \tau_p\mu. \end{aligned} \tag{63}$$

Since $\tilde{\boldsymbol{x}}$ lies on the $AF_{\boldsymbol{x_1}^p}$ and $\tau_p = \|\tilde{\boldsymbol{x}}^p - \boldsymbol{x_1}^p\|$, it is at least as big as the minimal triangle height in the mesh X_p, i.e., $\tau_p \geq \frac{h_p}{\eta}$. On the other hand, $\tau_p \leq t_p = Ah_p$,

www.manaraa.com

where A is a constant chosen for this particular ϕ, but independent of h_p. Combining these bounds with inequality (63), we see that

$$\nabla\phi(\boldsymbol{x_1}^p)\cdot\boldsymbol{a}+\frac{1}{f(\boldsymbol{x_1}^p,\boldsymbol{a})}\geq-\mu\left(\frac{\eta}{2}+A\right)h_p. \tag{64}$$

We can now use the continuity of f, the smoothness of ϕ, and the uniformness of convergence of U^p to pass to a limit as $p\to\infty$ in the inequality (64):

$$\nabla\phi(\boldsymbol{x_0})\cdot\boldsymbol{a}+\frac{1}{f(\boldsymbol{x_0},\boldsymbol{a})}\geq 0\implies(\nabla\phi(\boldsymbol{x_0})\cdot\boldsymbol{a})f(\boldsymbol{x_0},\boldsymbol{a})+1\geq 0,$$

which completes the proof of inequality (24), since $\boldsymbol{a}$ was chosen to be an arbitrary vector in $S_1^{\phi,\boldsymbol{x_0}}$.

In this proof we have several times passed to a subsequence. If some other subsequence of U^r converges to a different limit $\bar{u}$, the above argument could be repeated for that subsequence to prove that $\bar{u}$ also satisfies (23) and (24). The uniqueness of the viscosity solution (proven in [10, 9]) implies $u=\bar{u}$; thus, the entire sequence U^r converges to u uniformly as $r\to\infty$. ∎

7.3. Additional comments. First, the above proof of convergence, as well as the preceding lemmas, can be easily repeated for the corresponding method in higher dimensions. Moreover, if the update formula (31) in the description of the method is replaced by any other update-from-a-single-simplex formula such that

1. the update formula is consistent (converges to the PDE as $h\to 0$),
2. the update formula is upwinding (the update is computed/accepted only from the simplex, which contains the characteristic direction), and
3. the update formula is stable (there exists a uniform bound for U),

then the resulting numerical solutions should converge to the viscosity solution of the Hamilton–Jacobi–Bellman PDE. This conjecture is the basis for the methods based on finite-difference discretization discussed in section 8.2.

Second, the above proof uses the continuity of the speed function f, but not the Lipschitz-continuity. Thus, if the viscosity solution can be defined for $f\in C(\Omega)$, then the above proof will still be valid. The majority of the control-theoretic papers to which we are referring in this work require the speed of motion f (or, equivalently, the running cost K) to be Lipschitz-continuous. Nevertheless, the value function u can be defined for a much broader class of control problems, including those for which f is discontinuous. Some examples of our method applied to such problems can be found in section 9.4. Numerical evidence confirms that the method described above works correctly in that more general case. This is not surprising, since Bellman's optimality principle is valid even when the speed $f(\boldsymbol{x},\boldsymbol{a})$ is very ill-behaved, and our numerical methods merely mimic the logic of that principle.

8. Front propagation problems and OUMs. Anisotropic aspects of front propagation have been studied in several different contexts, including geometric optics, geophysics, tomography, and crystal growth; our primary emphasis is on an application-neutral analysis, concentrating on the properties of the particular class of static Hamilton–Jacobi PDEs. We begin with the correspondence between anisotropic optimal trajectory problems and a class of anisotropic front expansion (contraction) problems described in section 8.1. Our goal is to determine a set of the anisotropic front expansion (contraction) problems, which can be solved efficiently by our semi-Lagrangian OUM, and to construct a family of OUMs based on fully Eulerian discretizations.

www.manaraa.com

8.1. Front propagation problems: Static Hamilton–Jacobi approach. Consider a simple curve Γ_t moving in R^2 in the direction normal to itself with some speed $F(\boldsymbol{x}, \boldsymbol{n})$, where $\boldsymbol{n}$ is an "outwards pointing" unit vector normal to the curve as it passes through the point $\boldsymbol{x}$. If the curve is not smooth, then $\boldsymbol{n}$ is not defined, but a geometric construction based on a variant of Huygens' principle can still be used to define the evolution of Γ_t. An important subclass of the front propagation problems consists of applications in which the speed function F never changes sign. If the function F is strictly positive (or negative), then the front always expands (or contracts). This implies that the front passes through each point only once. Thus, we can define $u(\boldsymbol{x})$ to be the arrival time: $u(\boldsymbol{x}) = t \iff \boldsymbol{x} \in \Gamma_t$. If F is always nonnegative, the outwards unit normal vector can be expressed as $\boldsymbol{n}(\boldsymbol{x}) = \frac{\nabla u(\boldsymbol{x})}{\|\nabla u(\boldsymbol{x})\|}$ and, assuming that $\boldsymbol{n}(\boldsymbol{x})$ is always defined, it is straightforward to show that the arrival time u satisfies the PDE

$$\|\nabla u(\boldsymbol{x})\| F\left(\boldsymbol{x}, \frac{\nabla u(\boldsymbol{x})}{\|\nabla u(\boldsymbol{x})\|}\right) = 1, \tag{65}$$

$$u = 0 \text{ on } \Gamma_0.$$

This is a static Hamilton–Jacobi PDE of the form $H(\nabla u, \boldsymbol{x}) = 1$, where the Hamiltonian H is homogeneous of degree one in the first argument. To interpret (65) where ∇u does not exist, one normally uses the unique viscosity solution, as defined in [9, 10]. As follows from the results in [14] and [41], the level sets of the viscosity solution u of (65) will correspond to the evolution of Γ_0 defined by Huygens' principle.

We note that, in general, the Hamiltonian $H(\nabla u, \boldsymbol{x}) = \|\nabla u\| F(\boldsymbol{x}, \frac{\nabla u}{\|\nabla u\|})$ is not convex. As shown in [14], such $H(\nabla u, \boldsymbol{x})$ can always be considered as a result of a differential game model. Several iterative numerical schemes are based on this approach (see [4] or [16], for example). However, if H is convex, it can be alternatively considered as a product of a dual min-time optimal control problem [41]. Using two interpretations of the Hamiltonian, we can show that the speeds F and f are related by a homogeneous Legendre transform:

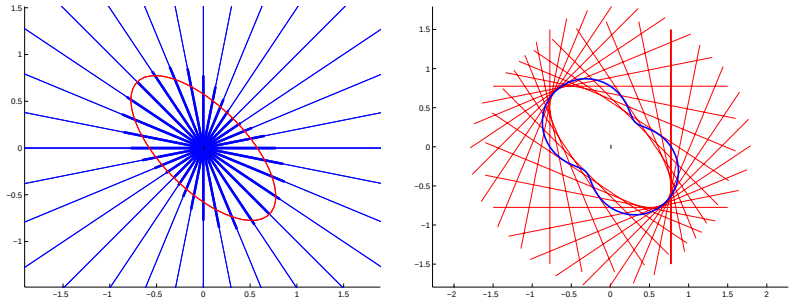
$$F(\boldsymbol{x}, \boldsymbol{n}) = \max_{\boldsymbol{a} \in S_1} \{(\boldsymbol{n} \cdot (-\boldsymbol{a})) f(\boldsymbol{x}, \boldsymbol{a})\}, \tag{66}$$

$$f(\boldsymbol{x}, \boldsymbol{a}) = \min_{\boldsymbol{n} \in S_1,\, (\boldsymbol{n} \cdot \boldsymbol{a}) < 0} \left\{ \frac{F(\boldsymbol{x}, \boldsymbol{n})}{(-\boldsymbol{n} \cdot \boldsymbol{a})} \right\}. \tag{67}$$

Remark 8.1. We note that $F(\boldsymbol{x}, \boldsymbol{n})$ is the speed of the front's movement in the direction normal to itself (here, $\boldsymbol{n} = \frac{\nabla u(\boldsymbol{x})}{\|\nabla u(\boldsymbol{x})\|}$), whereas $f(\boldsymbol{x}, \boldsymbol{a})$ is the speed of the vehicle's motion in the direction $\boldsymbol{a}$. Correspondingly, the correct $\boldsymbol{n}$ is fully determined by the gradient direction of the function $u(\boldsymbol{x})$, while the optimal $\boldsymbol{a} \in S_1$ is determined by the direction of the characteristic passing through the point $\boldsymbol{x}$ and, therefore, is a function of the particular Hamilton–Jacobi–Bellman equation. In the isotropic case, however, there is no difference since (66) yields $f(\boldsymbol{x}) = F(\boldsymbol{x})$.

Define the vehicle's speed profile $S_f(\boldsymbol{x}) = \{\boldsymbol{a} f(\boldsymbol{x}, \boldsymbol{a}) \mid \boldsymbol{a} \in S_1\}$, its flipped (center symmetry applied) version $S_{-f}(\boldsymbol{x}) = \{-\boldsymbol{a} f(\boldsymbol{x}, \boldsymbol{a}) \mid \boldsymbol{a} \in S_1\}$, and the front propagation speed profile $S_F(\boldsymbol{x}) = \{\boldsymbol{n} F(\boldsymbol{x}, \boldsymbol{n}) \mid \boldsymbol{n} \in S_1\}$. The formulas (66) and (67) can now be

www.manaraa.com



(a) (b)

FIG. 4. (a) *Using S_f to construct S_F;* (b) *using S_F to construct S_f.*

interpreted geometrically:[11]

— $F(\boldsymbol{x}, \boldsymbol{n})$ can be obtained by projecting $S_{-f}(\boldsymbol{x})$ onto a unit vector $\boldsymbol{n}$ and then by taking the maximum of this (signed) projection (see Figure 4(a));

— $S_{-f}(x)$ can be obtained as an envelope of lines perpendicular to $\boldsymbol{n}$ drawn at every point of $S_F(x)$ (see Figure 4(b)).

By the above construction, S_f is always convex. Thus, different optimal trajectory problems will yield the same Hamilton–Jacobi–Bellman equation, provided that the speed profiles have the same convex hull.[12] See [30] and the references therein for an additional discussion of Wulff shapes and mutual properties of functions related by the homogeneous Legendre transform.

Finally, we note that the correspondence between these two types of anisotropic problems can be used to build an alternative definition of Huygens' principle: using scaled speed profiles S_{-f} at each point of the wavefront instead of the circles of front-direction-dependent radius; see [14], [40], or [46].

8.2. OUMs for Eulerian discretizations. Given $0 < F_1 \le F(\boldsymbol{x}, \boldsymbol{n}) < F_2$ for all $\boldsymbol{x}$ and $\boldsymbol{n}$, we can use the formula (67) to prove that $0 < F_1 = f_1 \le f(\boldsymbol{x}, \boldsymbol{a}) < f_2 = F_2$ for all $\boldsymbol{x}$ and $\boldsymbol{a}$. Thus, the corresponding control problem can be treated by the OUMs with the semi-Lagrangian update formula described in section 6.

We now proceed to construct the OUMs for the fully Eulerian approximation of $\|\nabla u\| F(\boldsymbol{x}, \frac{\nabla u}{\|\nabla u\|}) = 1$. The key idea is that any consistent upwind finite difference discretization can be used to compute an update-from-a-single-simplex $V_{\boldsymbol{x_j}, \boldsymbol{x_k}}(\boldsymbol{x})$. Our derivation of such discretizations generalizes the approach used in defining the Fast Marching Method on unstructured meshes given in [38].

[11] In wave physics, $F(\boldsymbol{x}, \boldsymbol{n})$ corresponds to the *"phase velocity"* if $\boldsymbol{n}$ is the direction normal to the wavefront [11]. In crystalline variational problems, $F(\boldsymbol{x}, \boldsymbol{n})$ corresponds to the *"surface free energy"* if $\boldsymbol{n}$ is the direction normal to the surface [43]. Additionally, $f(\boldsymbol{x}, \boldsymbol{a})$ corresponds to the *"group velocity,"* i.e., the speed with which a blob of energy is moving in the direction $\boldsymbol{a}$ [11]. Finally, this speed profile is often referred to as a *"ray surface"* or *"impulse-response surface"* [11, 31]. The corresponding object in crystalline variational problems is the *"Wulff shape"*—the shape which minimizes the free surface energy for a fixed volume with no additional constraints [43].

[12] Similar geometric construction is common in tomography; the formulas (67) and (66) are related to the *inverse Radon transform* [20]. Analytic expressions for F in terms of f and for f in terms of F can be easily derived for R^2 (see [30, 46], for example). Similar formulas expressing the relationship between the *group speed* and the *phase speed* were known in wave physics [28] at least as early as 1837.

www.manaraa.com

8.2.1. Upwind finite-difference discretization. Consider an unstructured triangulated mesh X of diameter h (i.e., if the mesh points $\boldsymbol{x_j}$ and $\boldsymbol{x_k}$ are adjacent, then $\|\boldsymbol{x_j} - \boldsymbol{x_k}\| \leq h$). Let $\boldsymbol{x_j}$ and $\boldsymbol{x_k}$ be two adjacent mesh points and choose some other mesh point $\boldsymbol{x} \in \Omega \backslash \partial\Omega$. Define the unit vectors $\boldsymbol{P_1} = \frac{\boldsymbol{x}-\boldsymbol{x_j}}{\|\boldsymbol{x}-\boldsymbol{x_j}\|}$ and $\boldsymbol{P_2} = \frac{\boldsymbol{x}-\boldsymbol{x_k}}{\|\boldsymbol{x}-\boldsymbol{x_k}\|}$. Assume that $\boldsymbol{P_1}$ and $\boldsymbol{P_2}$ are linearly independent, and consider the 2×2 nonsingular matrix P having $\boldsymbol{P_1}$ and $\boldsymbol{P_2}$ as its rows. Let $v_r(\boldsymbol{x})$ be the value of the directional derivative for the direction $\boldsymbol{P_r}$ evaluated at the point $\boldsymbol{x}$. Assuming that the function u is differentiable at $\boldsymbol{x}$, we have $P\nabla u(\boldsymbol{x}) = \boldsymbol{v}(\boldsymbol{x})$, where $\boldsymbol{v}(\boldsymbol{x}) = \begin{bmatrix} v_1(\boldsymbol{x}) \\ v_2(\boldsymbol{x}) \end{bmatrix}$. Recall that the front propagation equation (65) can be written as $\|\nabla u(\boldsymbol{x})\|^2 F^2(\boldsymbol{x}, \frac{\nabla u(\boldsymbol{x})}{\|\nabla u(\boldsymbol{x})\|}) = 1$, which can be restated in terms of $\boldsymbol{v}(\boldsymbol{x})$:

$$\boldsymbol{v}(\boldsymbol{x})^T (PP^T)^{-1} \boldsymbol{v}(\boldsymbol{x}) F^2 \left(\boldsymbol{x}, \frac{P^{-1}\boldsymbol{v}(\boldsymbol{x})}{\|P^{-1}\boldsymbol{v}(\boldsymbol{x})\|} \right) = 1. \tag{68}$$

To obtain the discretized equation, we now replace each v_r with a difference approximation: $v_r(\boldsymbol{x}) \approx w_r \equiv a_r U + b_r$, where the b_r's linearly depend on the values of U (and possibly of ∇U for higher-order schemes) at the mesh points $\boldsymbol{x_j}$ and $\boldsymbol{x_k}$.

Remark 8.2. In general, the choice of the difference approximation will depend upon the structure of the mesh and will also affect the rate of convergence of the method. The simplest first-order finite-difference approximation is obtained by choosing

$$a_1 = \frac{1}{\|\boldsymbol{x} - \boldsymbol{x_j}\|}, \quad b_1 = \frac{-U(\boldsymbol{x_j})}{\|\boldsymbol{x} - \boldsymbol{x_j}\|}, \quad a_2 = \frac{1}{\|\boldsymbol{x} - \boldsymbol{x_k}\|}, \quad b_2 = \frac{-U(\boldsymbol{x_k})}{\|\boldsymbol{x} - \boldsymbol{x_k}\|}.$$

Higher-order accurate operators can be built using the computed value for ∇U at the mesh points (see [38, 46] for further details).

For convenience, let $Q = (PP^T)^{-1}$ and use $\boldsymbol{v}(\boldsymbol{x}) \approx \boldsymbol{w}(\boldsymbol{x}) = V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})\boldsymbol{a} + \boldsymbol{b}$. Then the discretized version of (68) can be used as an equation for the upwind-update-from-a-single-simplex $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$:

$$\left((\boldsymbol{a}^T Q \boldsymbol{a})(V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x}))^2 + (2\boldsymbol{a}^T Q \boldsymbol{b}) V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x}) + (\boldsymbol{b}^T Q \boldsymbol{b}) \right) F^2 \left(\boldsymbol{x}, \frac{P^{-1}\boldsymbol{w}}{\|P^{-1}\boldsymbol{w}\|} \right) = 1. \tag{69}$$

Remark 8.3. In the isotropic case, the analogous equation was just a quadratic (see the appendix and [38]). Equation (69) is a more complex nonlinear equation since $\boldsymbol{w}(\boldsymbol{x})$ also depends on $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$. In general, this equation will have to be solved approximately, and the overall efficiency of the method will also depend on the iterative numerical method used to solve (69). Since these iterations are generally unavoidable, we will consider solving this equation as a single operation in the further analysis of computational complexity. We note that the iterative zero-finding required to compute $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$ is *local* (i.e., $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$ can be computed independently from any other $V_{\boldsymbol{x_i},\boldsymbol{x_l}}(\boldsymbol{x_m})$) and thus should not be confused with the iterations necessary to solve a coupled system of nonlinear equations (such as (25)) simultaneously.

8.2.2. Upwinding criterion and combined update formula. We need to ensure that the value of $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$ computed from (69) is truly upwind, i.e., that the characteristic for the mesh point $\boldsymbol{x}$ lies inside the simplex $\boldsymbol{x}\boldsymbol{x_j}\boldsymbol{x_k}$. The approximate gradient $P^{-1}(V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})\boldsymbol{a} + \boldsymbol{b})$ can be used to compute an approximation to the characteristic direction $\boldsymbol{a}(\boldsymbol{x})$. For R^n, the requirement that the characteristic direction

www.manaraa.com

should point into the simplex $\boldsymbol{x x_1 \dots x_n}$ is equivalent to the condition that all the elements of the vector $(P^T)^{-1}\boldsymbol{a}(\boldsymbol{x})$ should be positive.

The unfortunate feature of this upwinding criterion is that it is based on the approximate rather than the exact characteristic direction. Due to the approximation error, it is possible that an upwinding criterion will not be satisfied even though the true characteristic for the mesh point $\boldsymbol{x}$ lies inside the simplex $\boldsymbol{x x_j x_k}$. If that simplex is small enough, this can happen only when one of the elements of the vector $(P^T)^{-1}\boldsymbol{a}(\boldsymbol{x})$ is close to zero, i.e., only when the characteristic direction almost coincides with $(-\boldsymbol{P_1})$ or $(-\boldsymbol{P_2})$. That corresponds to the situation in which $U(\boldsymbol{x})$ can be computed based on either $U(\boldsymbol{x_j})$ or $U(\boldsymbol{x_k})$. Thus, we define the "one-sided-update" formula in a manner consistent with the control-theoretic perspective:

$$V_{\boldsymbol{x_i}}(\boldsymbol{x}) = \frac{\|\boldsymbol{x_i} - \boldsymbol{x}\|}{f(\boldsymbol{x}, \frac{\boldsymbol{x_i}-\boldsymbol{x}}{\|\boldsymbol{x_i}-\boldsymbol{x}\|})} + U(\boldsymbol{x_i}). \tag{70}$$

Therefore, the final formula for the upwind-update-from-a-single-simplex becomes

$$V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x}) = \begin{cases} \text{solution of (69)} & \text{if } \boldsymbol{P_1} \text{ and } \boldsymbol{P_2} \text{ are linearly independent} \\ & \text{and the upwinding criterion is satisfied,} \\ \min(V_{\boldsymbol{x_j}}(\boldsymbol{x}),\, V_{\boldsymbol{x_k}}(\boldsymbol{x})) & \text{otherwise.} \end{cases} \tag{71}$$

Using the finite-difference update formula (71) instead of the formula (31) in the algorithm described in section 6, we obtain a new OUM for solving the front expansion problem (Hamilton–Jacobi equation (65)). In fact, this defines a whole family of such methods, since different upwind finite-difference operators can be used to approximate $w_r(\boldsymbol{x})$ in (68).

We note that the resulting methods

- are single-pass and have the same computational complexity as the semi-Lagrangian OUM introduced in section 6,
- work equally well on acute and nonacute triangulated meshes,
- are applicable for a general anisotropic optimal trajectory problem described in section 3,
- can be easily extended to R^n and manifolds. (The generalizations of the mapping $\boldsymbol{n} \mapsto \boldsymbol{a}$, of (68), and of the upwinding criterion are obvious.)

Remark 8.4 (Convergence). In the appendix, we show the connection between a particular first-order Eulerian OUM (based on Remark 8.2) and the first-order semi-Lagrangian OUM (introduced in section 6); in this case, the convergence to the viscosity solution follows from section 7. However, as of right now, we do not have a proof of convergence for the general (higher-order) OUMs based on the Eulerian discretization. We rely on general convergence considerations (see the remarks following the proof of Theorem 7.7) and on the numerical evidence (section 9.4 and [39, 46]). In all of our numerical experiments the numerical solution U produced by these methods converges to the viscosity solution of the original PDE. The rate of convergence depends on the particular finite-difference operators used to approximate $w_r(\boldsymbol{x})$ in (68).

9. Implementation and numerical results.

9.1. Implementation notes. An efficient implementation of the described numerical methods for the anisotropic optimal-trajectory and front-propagation problems requires dealing with several algorithmic issues. Storing and sorting the current

www.manaraa.com

AcceptedFront, for example, has to be implemented rather carefully to enable efficient search for the "*AcceptedFront* neighborhood" $\mathrm{NF}(\boldsymbol{x})$ for every *Considered* point $\boldsymbol{x}$. The inverse operation (searching for all *Considered* $\boldsymbol{x}$ such that $\bar{\boldsymbol{x}} \in \mathrm{NF}(\boldsymbol{x})$) is another major component of the implementation. Efficient use of data structures allows us to construct an algorithm with the computational complexity of $O(\Upsilon M \log M)$.

The connection between a particular class of anisotropic front-propagation and optimal-trajectory problems allows us to build both semi-Lagrangian and fully Eulerian single-pass methods. On a fixed mesh X, the computational complexity of these methods will be the same. However, the overall efficiency of each program will be affected by the chosen upwind-update-from-a-single-simplex formula. The optimal choice depends on the particular speed functions F and f and on the details of implementation: the semi-Lagrangian method requires performing a local minimization at each mesh point (using (31)), whereas the finite-differences upwind update formula requires finding the roots of the nonlinear equation (69). Generally, both the minimization and the root-finding have to be done approximately, and the overall efficiency depends on the particular numerical method used to compute the approximate update. Our implementations used the "golden section search" and the Newton–Raphson method to numerically resolve the control-theoretic and finite-difference update formulas, respectively; the implementation details of these and other numerical minimization and zero-finding techniques can be found in [42].

We note that the above complexity and efficiency discussion is limited to finding a numerical solution on a fixed grid. The speed of convergence (of the numerical approximation $U(\boldsymbol{x})$ to the viscosity solution $u(\boldsymbol{x})$ as the grid is refined) is a separate issue. Thus, the availability of the higher-order accurate upwind update formulas is a significant advantage of the Eulerian approach.

9.2. Using a local anisotropy coefficient. So far we have always used the global bounds on the speed function $0 < F_1 \le F(\boldsymbol{x}, \boldsymbol{p}) \le F_2$ for all $\boldsymbol{p}$ and $\boldsymbol{x}$. We now define the local bounds on F,

$$F_1(\boldsymbol{x}) = \min_{\boldsymbol{p} \in S_1} F(\boldsymbol{x}, \boldsymbol{p}), \qquad F_2(\boldsymbol{x}) = \max_{\boldsymbol{p} \in S_1} F(\boldsymbol{x}, \boldsymbol{p}),$$

and the local anisotropy coefficient $\Upsilon(\boldsymbol{x}) = \frac{F_2(\boldsymbol{x})}{F_1(\boldsymbol{x})}$.

We note that many of the lemmas stated in section 3 for $u(\boldsymbol{x})$ in terms of F_1 and F_2 can be restated in terms of $F_1(\boldsymbol{x})$ and $F_2(\boldsymbol{x})$. Most importantly, this is true for Remark 3.7, which establishes a bound on the angle between the characteristic and gradient directions. Thus, it is also possible to build the numerical method using $\Upsilon(\boldsymbol{x})$ instead of Υ in the definition of $\mathrm{NF}(\boldsymbol{x})$. Moreover, if F is smooth and the maximum/minimum in defining $F_1(\boldsymbol{x})$ and $F_2(\boldsymbol{x})$ are taken not just at the point but over some closed ball B centered at $\boldsymbol{x}$, then the resulting algorithm provably converges to the viscosity solution. (Indeed, for small enough h, $\mathrm{NF}(\boldsymbol{x}) \subset B$ even if $NF(\boldsymbol{x})$ were defined using the global anisotropy coefficient Υ.)

This observation leads to a substantially more efficient algorithm since the global anisotropy coefficient Υ can be much larger than $\sup_{\boldsymbol{x} \in \Omega} \Upsilon(\boldsymbol{x})$ for the front propagating in a strongly inhomogeneous medium.

9.3. Heuristic: Relaxing the update procedure. In the algorithm described in section 6, there are two different situations in which the tentative value $V(\boldsymbol{x})$ is recomputed for a *Considered* point $\boldsymbol{x}$:

- $V(\boldsymbol{x})$ is first computed using the entire $\mathrm{NF}(\boldsymbol{x})$ at the moment when $\boldsymbol{x}$ is added to *Considered*;

www.manaraa.com

- $V(\boldsymbol{x})$ is then recalculated from at most two simplexes every time the newly *Accepted* mesh point $\bar{\boldsymbol{x}}$ belongs to $\mathrm{NF}(\boldsymbol{x})$.

If the boundary condition for the PDE is nearly constant (i.e., if $q_2 \le q_1 + h/f_1$, where h is the diameter of the triangulated mesh), Lemma 7.3 shows that the AF will also approximate the level set throughout the execution of the algorithm. On the other hand, Lemma 3.4 shows that the optimal trajectory for $\boldsymbol{x}$ intersects a level set at some point $\tilde{\boldsymbol{x}}$ such that $\|\tilde{\boldsymbol{x}} - \boldsymbol{x}\| \le d_1 \Upsilon$, where d_1 is the distance from $\boldsymbol{x}$ to that level set. This means that if AF were exactly the level set, the initial evaluation of $V(\boldsymbol{x})$ would capture all the necessary information about all the potential characteristic directions for $\boldsymbol{x}$; thus, the further reevaluations of $V(\boldsymbol{x})$ would not be necessary. Since AF is only approximating the level set, capturing all the necessary directions requires "widening" the set $\mathrm{NF}(\boldsymbol{x})$. Carefully combining Lemmas 7.3 and 3.4, and assuming $U \approx u$ on AF, we can show that all the characteristic directions are still covered if $\mathrm{NF}(\boldsymbol{x})$ is taken to be two times "wider":

$$\widehat{\mathrm{NF}}(\boldsymbol{x}) \;=\; \{\boldsymbol{x_j x_k} \in AF \;\mid \exists \tilde{\boldsymbol{x}} \text{ on } \boldsymbol{x_j x_k} \text{ s.t. } \|\tilde{\boldsymbol{x}} - \boldsymbol{x}\| \le 2h\Upsilon(\boldsymbol{x})\}\,.$$

This still leads to the total of roughly 2Υ evaluations of V_s for each mesh point, but it is no longer necessary to search for all *Considered* $\boldsymbol{x}$ such that $\bar{\boldsymbol{x}} \in \mathrm{NF}(\boldsymbol{x})$ each time a new $\bar{\boldsymbol{x}}$ is *Accepted*.[13]

Furthermore, an additional update relaxation can be used with the Eulerian discretization-based methods if the boundary condition for the PDE is nearly constant. In the initial computation of $V(\boldsymbol{x})$ it is often not necessary to consider the entire $\mathrm{NF}(\boldsymbol{x})$: we can stop as soon as we find $\boldsymbol{x_j x_k} \in \mathrm{NF}(\boldsymbol{x})$ such that $V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x})$ satisfies the upwinding conditions (see section 8.2.2). The viscosity solution u of the Hamilton–Jacobi PDE is Lipschitz-continuous, and therefore ∇u exists almost everywhere. As shown in [46], if u is differentiable at the point $\boldsymbol{x}$ and the vehicle's speed profile $S_f(\boldsymbol{x})$ is strictly convex, then there exists a unique optimizing control $\boldsymbol{a}(\boldsymbol{x})$. Thus, away from the shocks, there should not be multiple simplexes in $\mathrm{NF}(\boldsymbol{x})$ producing updates which satisfy the upwinding criteria.

For a fixed grid X, the numerical evidence suggests that the "relaxation" significantly improves efficiency of the program. As the grid is refined, the numerical solution obtained by the "relaxed" scheme converges to the viscosity solution—sometimes slower, but often (depending on the type of anisotropy) even faster than the solution computed by the "full-update" scheme. However, the *asymptotic order of accuracy* of the "relaxed" and "full-update" schemes seems to be the same in all of our numerical experiments (e.g., see section 9.4.1).

9.4. Numerical experiments. In this section we consider several test problems, each of which can be described by a non-Eikonal (anisotropic) Hamilton–Jacobi PDE. In each case, the speed function is assumed to be known from the characterization of a particular application domain. For example, in the optimal-trajectory test problem, $f(\boldsymbol{x}, \boldsymbol{a})$ reflects the assumptions about the speed of the controlled vehicle, while in the seismic imaging test problem, the front-expansion speed F is derived using the assumptions about the elliptical nature of the "impulse-response surface" for the anisotropic medium.

9.4.1. Geodesic distances on manifolds. The first test problem is to find the geodesic distance on a manifold $z = g(x, y)$. As described in [21] and [36], this can

[13] The numerical experiments indicate that a much smaller "widening" of *NF* is sufficient in practice.

www.manaraa.com

be accomplished by approximating the manifold with a triangulated mesh and then solving the distance equation $\|\nabla u\| = 1$ on that mesh. Since the latter equation is Eikonal, the Fast Marching Method can be used to solve it efficiently. However, if one desires to formulate the problem in the $x - y$ plane instead of the intrinsic manifold coordinates, then the corresponding equation for u is not Eikonal. Indeed, in the $x - y$ plane, the manifold's geodesic distance function u has to satisfy (65) with the speed function F defined as

$$F(x, y, \omega) = \sqrt{\frac{1 + g_y^2 \cos^2(\omega) + g_x^2 \sin^2(\omega) - g_x g_y \sin(2\omega)}{1 + g_x^2 + g_y^2}}, \tag{72}$$

where ω is the angle between $\nabla u(x, y)$ and the positive direction of the x-axis. The degree of anisotropy in this equation is substantial, since the dependence of F upon ω can be pronounced when ∇g is relatively large.[14]

As shown in section 8.1, u can also be considered as a value function for the corresponding min-time optimal-trajectory problem and must, therefore, satisfy (22). The vehicle's speed function $f(x, y, \boldsymbol{a})$ can be defined by applying (67) to the speed of front propagation $F(x, y, \omega)$. However, it is even easier to obtain f from the control-theoretic considerations. If the vehicle moving with speed $f(x, y, \boldsymbol{a})$ in the x-y plane is just a shadow of another vehicle moving with a unit speed on the manifold, then this vehicle's speed profile is just an orthogonal projection of a unit circle from the manifold's tangent plane onto the x-y plane, i.e.,

$$f(x, y, \boldsymbol{a}) = (1 + (\nabla g(x, y) \cdot \boldsymbol{a})^2)^{-\frac{1}{2}}, \tag{73}$$

where $\boldsymbol{a}$ is a vector of unit length and f is the control-theoretic speed of motion in the direction $\boldsymbol{a}$ (see section 3.1 and section 8.1 for details).

As an example, we consider the manifold $g(x, y) = .9 \sin(2\pi x) \sin(2\pi y)$ and compute the geodesic distance on it from the origin. The *anisotropy coefficient* for this problem is $\Upsilon = \frac{F_2}{F_1} = \sqrt{100 + 324\pi^2}/10 \approx 5.7$. Since the analytical solution is not available, we use the results of the tested method on the mesh with 385×385 mesh points as an estimate of the "true" value function. This estimate is used to perform the convergence analysis on coarser regular meshes in the x-y plane for the following:

M1: iterative solution to the first-order semi-Lagrangian scheme (25),
M2: OUM based on the first-order semi-Lagrangian scheme (section 6),
M3: same as M2, but with the "relaxed update" (section 9.3),
M4: OUM based on the first-order finite-differences scheme (section 8).

See Figure 5 for the level sets of the value function and for the table of error estimates.

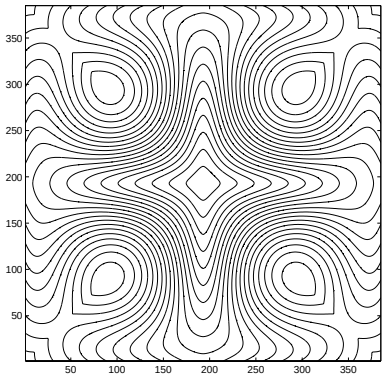
9.4.2. First arrivals in inhomogeneous anisotropic medium. Finally, we include an example of the first arrival travel times computation with applications to seismic imaging. We start with a computational domain which suggests material layering under a sinusoidal profile. The computational domain is the square $[-a, a] \times [-a, a]$, with layer shapes

$$C(x) = A \sin\left(\frac{m\pi x}{a} + \beta\right), \tag{74}$$

where A is the amplitude of the sinusoidal profile, m is the number of periods, and β is the phase offset. The domain is split into n layers by the curves $y_i(x) = C(x) + b_i$, where $i = 1, \ldots, (n - 1)$.

[14] The algorithm presented in [21] using the manifold-approximating mesh is certainly more efficient for this problem; here, it serves as a convenient test problem for the general anisotropic OUMs.

www.manaraa.com



L_∞ Error	25^2	49^2	97^2	193^2
M1	0.15474	0.05902	0.02619	0.00866
M2	0.36131	0.25581	0.13021	0.04195
M3	0.37647	0.25679	0.13070	0.04327
M4	0.36052	0.25940	0.13042	0.04174

L_2 Error	25^2	49^2	97^2	193^2
M1	0.05503	0.02281	0.01073	0.00381
M2	0.13918	0.09901	0.04876	0.01416
M3	0.14022	0.09848	0.04766	0.01374
M4	0.13749	0.09659	0.04626	0.01374

FIG. 5. *The geodesic distance from the origin on the surface* $z = .9\sin(2\pi x)\sin(2\pi y)$ *computed on the square* $[-.5,.5]\times[-.5,.5]$ *in the x-y plane. The error estimates were produced on refined meshes taking the corresponding method on a* 385^2 *mesh to be the true solution (hence the seemingly higher-than-first-order rate at the end).*

In each layer, the anisotropic speed profile S_f is given at every point (x, y) by an ellipse with the bigger axis (of length $2F_2$) tangential to the curve $C(x)$ and the smaller axis (of length $2F_1$) normal to the curve. F_1 and F_2 are constants in each layer. Thus, the ellipse's orientation and shape depend on (x, y).

This leads to an anisotropic Hamilton–Jacobi equation of the form

$$\|\nabla u(x,y)\|F = 1, \qquad u(0,0) = 0, \tag{75}$$

where the front propagation speed at every point (x, y) is given by the formula

$$F(x,y,u_x,u_y) = F_2\left(\frac{(1+q^2)u_x^2 + (1+p^2)u_y^2 - 2pqu_xu_y}{(1+p^2+q^2)(u_x^2+u_y^2)}\right)^{1/2}, \tag{76}$$

with

$$\begin{bmatrix} p \\ q \end{bmatrix} = \frac{\sqrt{(\frac{F_2}{F_1})^2 - 1}}{\sqrt{1 + \left(\frac{dC}{dx}(x)\right)^2}} \begin{bmatrix} \frac{dC}{dx}(x) \\ -1 \end{bmatrix}.$$

Formula (76) is derived using the the elliptical shape of $S_f(x, y)$ and applying formula (66) of section 8.1.

These calculations are performed using the OUMs with the control-theoretic and finite-difference formulas for computing an update-from-a-single-simplex. Both methods produce numerical solutions converging to the value function of the corresponding min-time optimal trajectory problem.

The equi-arrival curves shown in Figure 6 are obtained on a 193×193 regular mesh using the following parameter values:

$$a = .5, \quad A = .1225, \quad m = 2, \quad \beta = 0, \quad \text{and layer offsets } b_i = (-.25, 0, 0.25).$$

The max/min speed pair (F_2, F_1) for each layer is given in the figures. We note that in one of these examples the global *anisotropy coefficient* $\Upsilon = \frac{3}{.2} = 15$.

www.manaraa.com

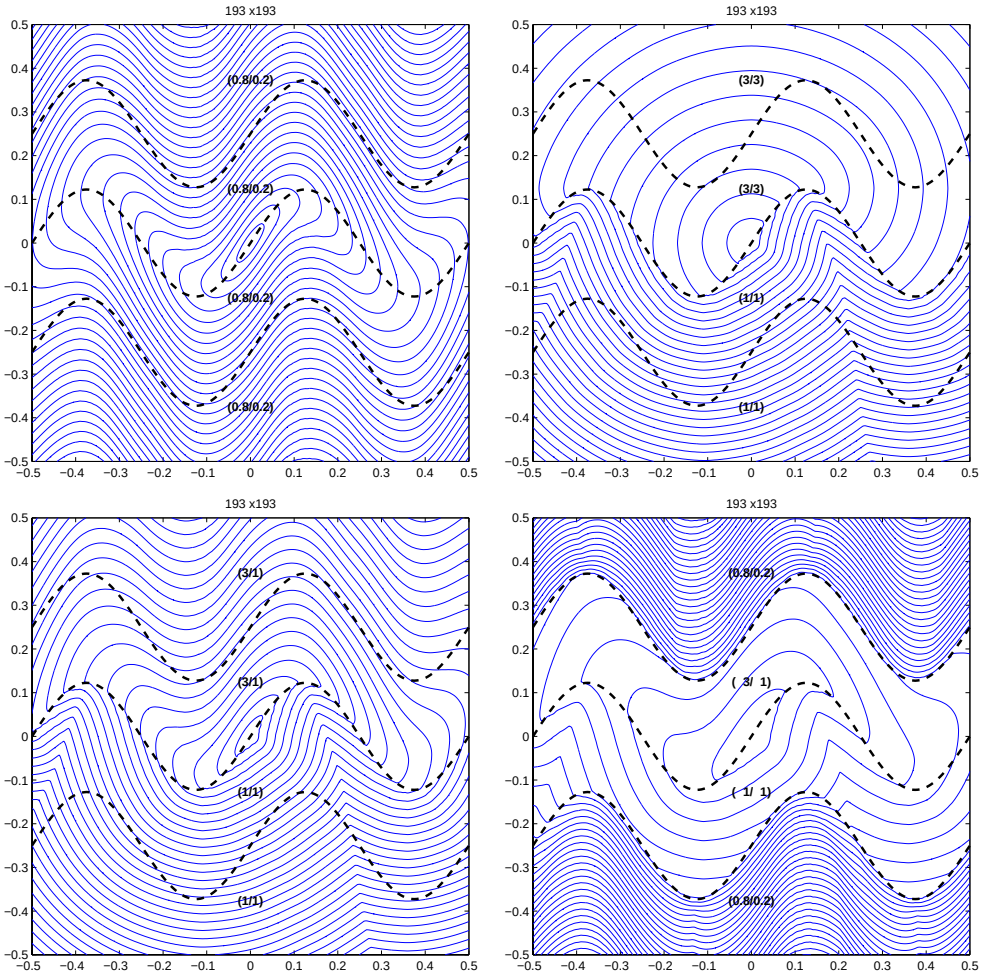


FIG. 6. *Seismic imaging test problem: equi-arrival curves in an inhomogeneous, multilayer medium.*

Remark 9.1. Since the speed function F is discontinuous across the layer boundaries, the standard viscosity solution results for the Hamilton–Jacobi–Bellman equation [10, 9] are not directly applicable. Thus, our proof of convergence in section 7 is not valid in this case either. Nevertheless, the produced numerical solutions seem to converge to the true value function of the corresponding control problem. This is not surprising since our methods are based on approximating Bellman's optimality principle, which is valid for a value function u under much more general assumptions about the speed (or the cost) of motion.

9.5. Conclusions. The methods presented in this paper are applicable for the static Hamilton–Jacobi–Bellman PDEs with convex Hamiltonian and finite speed function bounded away from zero (see [46] for additional background information and details of the proofs). We are currently working on extending these OUMs to a

www.manaraa.com

wider class of problems including treatment of nonconvex Hamiltonians, discontinuous speed functions, degenerate speed profiles (i.e., $f_1 = 0$), and stochastic control. We also note that parallelizable single-pass methods based on the same upwinding techniques may be built extending the ideas behind Dial's algorithm for the shortest path on the network.

We believe that similar decoupling techniques hold much promise for the nonlinear problems for which the notion of "information propagation" is well defined.

Appendix. We present a proof of the causality property for the semi-Lagrangian discretization of the Eikonal equation on an unstructured mesh with acute simplexes. We begin by restating the discretization formula (25) for the n-dimensional simplex s with the vertices at $\boldsymbol{x}, \boldsymbol{x_1}, \ldots, \boldsymbol{x_n}$. If $\tilde{\boldsymbol{x}}$ is a point on the $\boldsymbol{x_1} \ldots \boldsymbol{x_n}$ face of the simplex, we will use $\zeta = (\zeta_1, \ldots, \zeta_n)$ for its barycentric coordinates:

$$\zeta_i \in [0,1] \text{ for all } i, \qquad \sum_{i=1}^{n} \zeta_i = 1, \qquad \sum_{i=1}^{n} \zeta_i \boldsymbol{x_i} = \tilde{\boldsymbol{x}}.$$

Using Ξ to denote the set of all possible barycentric coordinates, we can write the isotropic version of the upwinding update formula (25):

$$V_s(\boldsymbol{x}) = \min_{\zeta \in \Xi} \left\{ \frac{\tau(\zeta)}{f(\boldsymbol{x})} + \sum_{i=1}^{n} \zeta_i U(\boldsymbol{x_i}) \right\}, \tag{77}$$

where $\tau(\zeta) = \|\tilde{\boldsymbol{x}} - \boldsymbol{x}\|$. Define the unit directional vectors $\boldsymbol{P_i} = \frac{\boldsymbol{x}-\boldsymbol{x_i}}{\|\boldsymbol{x}-\boldsymbol{x_i}\|}$. To justify using Dijkstra-like decoupling with this discretization, we prove the following version of Property (4.1).

PROPERTY A.1 (Causality). *If $(P_j \cdot P_k) \geq 0$ for all j and k, and if $\zeta = (\zeta_1, \ldots, \zeta_n)$ is the minimizer in formula* (77)*, then "$\zeta_i > 0$" implies "$V_s > U(\boldsymbol{x_i})$."*

Proof. Suppose $\zeta_i > 0$ for all $i \in \{1, \ldots, n\}$. (If that is not the case, the same argument can be repeated for the lower-dimensional simplex on which all barycentric coordinates are positive.)

Noting that $\frac{\partial \tau}{\partial \zeta_i}(\zeta) = \frac{(\boldsymbol{x}-\boldsymbol{x_i})\cdot(\boldsymbol{x}-\tilde{\boldsymbol{x}})}{\tau(\zeta)}$, we can write the Kuhn–Tucker optimality conditions for ζ as follows:

$$\frac{(\boldsymbol{x} - \boldsymbol{x_i}) \cdot (\boldsymbol{x} - \tilde{\boldsymbol{x}})}{\tau(\zeta) f(\boldsymbol{x})} + U(\boldsymbol{x_i}) = \lambda \qquad \text{for all } i \in \{1, \ldots, n\}, \tag{78}$$

where λ is the Lagrange multiplier. We note that

$$\lambda - U(\boldsymbol{x_i}) = \frac{\|\boldsymbol{x} - \boldsymbol{x_i}\| \sum_{j=1}^{n} \zeta_j \|\boldsymbol{x} - \boldsymbol{x_j}\| (\boldsymbol{P_i} \cdot \boldsymbol{P_j})}{\tau(\zeta) f(\boldsymbol{x})} > 0, \tag{79}$$

by the acuteness of simplex s and since $\zeta_i > 0$. Next, we note that, multiplying (78) by ζ_i and summing over all i's,

$$\lambda = \lambda \left(\sum_{i=1}^{n} \zeta_i \right) = \frac{\left(\sum_{i=1}^{n} \zeta_i (\boldsymbol{x} - \boldsymbol{x_i})\right) \cdot (\boldsymbol{x} - \tilde{\boldsymbol{x}})}{\tau(\zeta) f(\boldsymbol{x})} + \sum_{i=1}^{n} \zeta_i U(\boldsymbol{x_i}) = \frac{\tau(\zeta)}{f(\boldsymbol{x})} + \sum_{i=1}^{n} \zeta_i U(\boldsymbol{x_i}) = V_s. \tag{80}$$

Thus, $V_s > U(\boldsymbol{x_i})$ follows from (79). ∎

www.manaraa.com

We note that the proof holds on an arbitrary "acute" unstructured mesh in R^n and on manifolds. The "splitting section" techniques developed for the Fast Marching Method can also be used to implement Dijkstra-like decoupling of the above discretization on meshes with obtuse simplexes; see [21, 38] for details.

Finally, to explore the connection between semi-Lagrangian and finite-difference schemes, we further consider a column vector $\boldsymbol{w}$, with the entries $w_i = (V_s - U(\boldsymbol{x_i}))/\|\boldsymbol{x} - \boldsymbol{x_i}\|$. Using formula (79) and a matrix P whose rows are $\boldsymbol{P_i}$'s, we see that

$$\boldsymbol{w} = \frac{1}{\tau(\zeta) f(x)} P(\boldsymbol{x} - \tilde{\boldsymbol{x}}) \;\Rightarrow\; P^{-1}\boldsymbol{w} = \frac{\boldsymbol{x} - \tilde{\boldsymbol{x}}}{\tau(\zeta) f(x)} \;\Rightarrow\; \boldsymbol{w}^T \left(PP^T\right)^{-1} \boldsymbol{w} = \frac{\|\boldsymbol{x} - \tilde{\boldsymbol{x}}\|^2}{(\tau(\zeta) f(x))^2} = \frac{1}{f^2(x)}.$$

The latter is a quadratic equation in V_s and coincides with the particular first-order finite-difference upwind formula chosen in [38] for the isotropic front-propagation problems.

Furthermore, the analogous correspondence can be demonstrated for the first-order schemes in the anisotropic case. Starting from the general first-order semi-Lagrangian formula

$$V_s(\boldsymbol{x}) = \min_{\zeta \in \Xi} \left\{ \frac{\tau(\zeta)}{f(\boldsymbol{x}, \frac{\tilde{\boldsymbol{x}} - \boldsymbol{x}}{\tau(\zeta)})} + \sum_{i=1}^{n} \zeta_i U(\boldsymbol{x_i}) \right\}, \tag{81}$$

where $\tau(\zeta)$, $\tilde{\boldsymbol{x}}$, $\boldsymbol{P_i}$, and w_i are defined as above, we note that

$$\begin{aligned} 0 &= \min_{\zeta \in \Xi} \left\{ \frac{\tau(\zeta)}{f(\boldsymbol{x}, \frac{\tilde{\boldsymbol{x}} - \boldsymbol{x}}{\tau(\zeta)})} + \sum_{i=1}^{n} \zeta_i \left(U(\boldsymbol{x_i}) - V_s(\boldsymbol{x})\right) \right\} \\ &= \max_{\zeta \in \Xi} \left\{ \sum_{i=1}^{n} \zeta_i \left(V_s(\boldsymbol{x}) - U(\boldsymbol{x_i})\right) - \frac{\tau(\zeta)}{f(\boldsymbol{x}, \frac{\tilde{\boldsymbol{x}} - \boldsymbol{x}}{\tau(\zeta)})} \right\}. \end{aligned}$$

Moreover, since

$$\begin{aligned} \sum_{i=1}^{n} \zeta_i \left(V_s(\boldsymbol{x}) - U(\boldsymbol{x_i})\right) &= \sum_{i=1}^{n} \left(\zeta_i \|\boldsymbol{x} - \boldsymbol{x_i}\|\right) w_i \\ &= \sum_{i=1}^{n} \left(\zeta_i (\boldsymbol{x} - \boldsymbol{x_i})^T P^{-1}\right) w_i = (\boldsymbol{x} - \tilde{\boldsymbol{x}})^T \left(P^{-1}\boldsymbol{w}\right), \end{aligned}$$

we have

$$\max_{\zeta \in \Xi} \left\{ \frac{\tau(\zeta)}{f(\boldsymbol{x}, \frac{\tilde{\boldsymbol{x}} - \boldsymbol{x}}{\tau(\zeta)})} \left[\left(\left(-\frac{\tilde{\boldsymbol{x}} - \boldsymbol{x}}{\tau(\zeta)} \right) \cdot \frac{P^{-1}\boldsymbol{w}}{\|P^{-1}\boldsymbol{w}\|} \right) \|P^{-1}\boldsymbol{w}\| f\left(\boldsymbol{x}, \frac{\tilde{\boldsymbol{x}} - \boldsymbol{x}}{\tau(\zeta)}\right) - 1 \right] \right\} = 0.$$

Since both functions τ and f are strictly positive, the above is equivalent to

$$\max_{\zeta \in \Xi} \left\{ \left[\left(-\frac{\tilde{\boldsymbol{x}} - \boldsymbol{x}}{\tau(\zeta)} \right) \cdot \frac{P^{-1}\boldsymbol{w}}{\|P^{-1}\boldsymbol{w}\|} \right] \|P^{-1}\boldsymbol{w}\| f\left(\boldsymbol{x}, \frac{\tilde{\boldsymbol{x}} - \boldsymbol{x}}{\tau(\zeta)}\right) - 1 \right\} = 0$$

or, more conveniently,

$$\|P^{-1}\boldsymbol{w}\| \max_{\zeta \in \Xi} \left\{ \left[\left(-\frac{\tilde{\boldsymbol{x}} - \boldsymbol{x}}{\tau(\zeta)} \right) \cdot \frac{P^{-1}\boldsymbol{w}}{\|P^{-1}\boldsymbol{w}\|} \right] f\left(\boldsymbol{x}, \frac{\tilde{\boldsymbol{x}} - \boldsymbol{x}}{\tau(\zeta)}\right) \right\} = 1. \tag{82}$$

www.manaraa.com

Let $\boldsymbol{n} = \frac{P^{-1}\boldsymbol{w}}{\|P^{-1}\boldsymbol{w}\|}$. We note that, as ζ varies, $\boldsymbol{a} = \frac{\tilde{\boldsymbol{x}}-\boldsymbol{x}}{\tau(\zeta)}$ covers all directions within this simplex; moreover, the ζ minimizing (81) will also be the maximizer for (82). If all the ζ_i's are positive, then the corresponding $\boldsymbol{a}$ yields a local maximum of the expression $(-\boldsymbol{a}\cdot\boldsymbol{n})f(\boldsymbol{x},\boldsymbol{a})$. As discussed in section 8.1, such a local maximum is unique if S_f is convex; thus, by the formula (66), equation (82) is equivalent to

$$\|P^{-1}\boldsymbol{w}\|F\left(\boldsymbol{x}, \frac{P^{-1}\boldsymbol{w}}{\|P^{-1}\boldsymbol{w}\|}\right) = 1.$$

Finally, the square of this expression is a variant of the finite-difference formula (69) obtained for the first-order Eulerian discretization in section 8.2.1. The upwinding criterion required for the latter scheme is equivalent to verifying that all ζ_i's are positive.

As of right now, we are unaware of any such connections between the higher-order semi-Lagrangian and Eulerian schemes for the Eikonal or general Hamilton–Jacobi–Bellman equations.

Acknowledgments. The authors would like to thank L. C. Evans for his comments and suggestions on the general structure of the proof. The authors would also like to thank O. Hald, M. Falcone, R. Kohn, A. Spitkovsky, and A. Dukhovny.

REFERENCES

[1] D. Adalsteinsson and J.A. Sethian, *A fast level set method for propagating interfaces*, J. Comput. Phys., 118 (1995), pp. 269–277.
[2] D. Adalsteinsson and J.A. Sethian, *The fast construction of extension velocities in level set methods*, J. Comput. Phys., 148 (1999), pp. 2–22.
[3] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin, *Network Flows: Theory, Algorithms, and Applications*, Prentice–Hall, Englewood Cliffs, NJ, 1993.
[4] M. Bardi, M. Falcone, and P. Soravia, *Fully discrete schemes for the value function of pursuit-evasion games*, in Advances in Dynamic Games and Applications (Geneva, 1992), Ann. Internat. Soc. Dynam. Games 1, Birkhäuser-Boston, Cambridge, MA, 1994, pp. 89–105.
[5] R. Bellman, *Dynamic Programming*, Princeton University Press, Princeton, NJ, 1957.
[6] R. Bellman, *Introduction to the Mathematical Theory of Control Processes*, Academic Press, New York, 1967.
[7] I. Capuzzo Dolcetta and M. Falcone, *Discrete dynamic programming and viscosity solutions*, Ann. Inst. H. Poincaré Anal. Non Linéaire, 6 (1989), pp. 161–183.
[8] I. Capuzzo Dolcetta, *On a discrete approximation of the Hamilton–Jacobi equation of dynamic programming*, Appl. Math. Optim., 10 (1983), pp. 367–377.
[9] 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.
[10] M.G. Crandall and P.-L. Lions, *Viscosity solutions of Hamilton-Jacobi equations*, Trans. Amer. Math. Soc., 277 (1983), pp. 1–43.
[11] J.A. Dellinger, *Anisotropic Seismic Wave Propagation*, Ph.D. Dissertation, Department of Geophysics, Stanford University, Stanford, CA, 1991.
[12] E.W. Dijkstra, *A note on two problems in connection with graphs*, Numer. Math., 1 (1959), pp. 269–271.
[13] L.C. Evans, *Partial Differential Equations*, American Mathematical Society, Providence, RI, 1998.
[14] L.C. Evans and P.E. Souganidis, *Differential games and representation formulas for solutions of Hamilton-Jacobi-Isaacs equations*, Indiana Univ. Math. J., 33 (1984), pp. 773–797.
[15] M. Falcone, *The minimum time problem and its applications to front propagation*, in Motion by Mean Curvature and Related Topics, Proceedings of the International Conference at Trento, 1992, Walter de Gruyter, New York, 1994, pp. 70–88.
[16] M. Falcone, *A numerical approach to the infinite horizon problem of deterministic control theory*, Appl. Math. Optim., 15 (1987), pp. 1–13; corrigenda, 23 (1991), pp. 213–214.
[17] M. Falcone and R. Ferretti, *Discrete time high-order schemes for viscosity solutions of Hamilton-Jacobi-Bellman equations*, Numer. Math., 67 (1994), pp. 315–344.

www.manaraa.com

[18] R. Gonzales and E. Rofman, *On deterministic control problems: An approximate procedure for the optimal cost* I. *The stationary problem*, SIAM J. Control Optim., 23 (1985), pp. 242–266.
[19] J. Helmsen, E.G. Puckett, P. Colella, and M. Dorr, *Two new methods for simulating photolithography development in three dimensions*, in Proceedings of the SPIE 1996 International Symposium on Microlithography, Santa Clara, CA, SPIE 2726, SPIE, Bellingham, WA, 1996, pp. 253–261.
[20] A. Jain, *Fundamentals of Digital Image Processing*, Prentice-Hall, Englewood Cliffs, NJ, 1989.
[21] R. Kimmel and J.A. Sethian, *Fast marching methods on triangulated domains*, Proc. Nat. Acad. Sci. USA, 95 (1998), pp. 8341–8435.
[22] R. Kimmel and J.A. Sethian, *Fast Marching Methods for Robotic Navigation with Constraints*, Center for Pure and Applied Mathematics Report, University of California, Berkeley, 1996.
[23] R. Kimmel and J.A. Sethian, *Fast Voronoi diagrams and offsets on triangulated surfaces*, in Curve and Surface Design, P.J. Laurent, P. Sablonniere, and L.L. Schumaker, eds., Vanderbilt University Press, Nashville, TN, 2000, pp. 193–202.
[24] S.N. Kruzhkov, *Generalized solutions of the Hamilton-Jacobi Equations of the Eikonal type*, Math. USSR-Sb., 27 (1975), pp. 406–445.
[25] H.J. Kushner, *Probability Methods for Approximations in Stochastic Control and for Elliptic Equations*, Springer-Verlag, New York, 1977.
[26] H.J. Kushner and P.G. Dupuis, *Numerical Methods for Stochastic Control Problems in Continuous Time*, Academic Press, New York, 1992.
[27] R. Malladi and J.A. Sethian, *An $O(NlogN)$ algorithm for shape modeling*, Proc. Nat. Acad. Sci. USA, 93 (1996), pp. 9389–9392.
[28] J. McGulagh, *Geometrical propositions applied to the wave theory of light*, Trans. Royal Irish Acad., 17 (1837), pp. 241–263.
[29] S. Osher and R.P. Fedkiw, *Level set methods: An overview and some recent results*, J. Comput. Phys., 169 (2001), pp. 463–502.
[30] D. Peng, S. Osher, B. Merriman, and H-K. Zhao, *The geometry of Wulff crystal shapes and its relations with Riemann problems*, in Nonlinear Partial Differential Equations, Contemp. Math. 238, G.-Q. Chen and E. di Benedetto, eds., AMS, Providence, RI, 1999, pp. 251–303.
[31] G.W. Postma, *Wave propagation in a stratified medium*, Geophysics, 20 (1955), pp. 780–806.
[32] J.A. Sethian, *Fast Marching Methods for Computing Seismic Travel Times*, manuscript.
[33] J.A. Sethian, *A fast marching level set method for monotonically advancing fronts*, Proc. Nat. Acad. Sci. USA, 93 (1996), pp. 1591–1595.
[34] J.A. Sethian, *Fast marching level set methods for three-dimensional photolithography development*, in Proceedings of the SPIE 1996 International Symposium on Microlithography, Santa Clara, CA, SPIE 2726, SPIE, Bellingham, WA, 1996, pp. 262–272.
[35] J.A. Sethian, *Fast marching methods*, SIAM Rev., 41 (1999), pp. 199–235.
[36] J.A. Sethian, *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision and Materials Sciences*, Cambridge University Press, Cambridge, UK, 1996.
[37] J.A. Sethian and M. Popovici, *Three dimensional traveltimes computation using the fast marching method*, Geophysics, 64 (1999), pp. 516–523.
[38] J.A. Sethian and A. Vladimirsky, *Fast methods for the Eikonal and related Hamilton–Jacobi equations on unstructured meshes*, Proc. Nat. Acad. Sci. USA, 97 (2000), pp. 5699–5703.
[39] J.A. Sethian and A. Vladimirsky, *Ordered upwind methods for static Hamilton-Jacobi equations*, Proc. Nat. Acad. Sci. USA, 98 (2001), pp. 11069–11074.
[40] T.P. Schulze and R.V. Kohn, *A geometric model for coarsening during spiral-mode growth of thin films*, Phys. D, 132 (1999), pp. 520–542.
[41] P. Soravia, *Generalized motion of a front propagating along its normal direction: A differential games approach*, Nonlinear Anal., 22 (1994), pp. 1247–1262.
[42] W. Press, B. Flannery, S. Teukolsky, and W. Vetterling, *Numerical Recipes in C*, Cambridge University Press, Cambridge, UK, 1988.
[43] J.E. Taylor, *Crystalline variational problems*, Bull. Amer. Math. Soc., 84 (1978), pp. 568–588.
[44] J.N. Tsitsiklis, *Efficient algorithms for globally optimal trajectories*, in Proceedings of the IEEE 33rd Conference on Decision and Control, Lake Buena Vista, FL, 1994, pp. 1368–1373.
[45] J.N. Tsitsiklis, *Efficient algorithms for globally optimal trajectories*, IEEE Trans. Automat. Control, 40 (1995), pp. 1528–1538.
[46] A. Vladimirsky, *Fast Methods for Static Hamilton-Jacobi Partial Differential Equations*, Ph.D. Dissertation, Department of Mathematics, University of California, Berkeley, CA, 2001.

www.manaraa.com