Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Computational Physics 199 (2004) 260–290
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

Visibility and its dynamics in a PDE based implicit framework ☆

Y.-H.R. Tsai [a,*], L.-T. Cheng [b], S. Osher [c], P. Burchard, G. Sapiro [d]

[a] *Institute for Advanced Study, and Department of Mathematics, Princeton University, Princeton, NJ 08544, USA*
[b] *Department of Mathematics, UCSD, La Jolla, CA 92093-0112, USA*
[c] *Department of Mathematics, UCLA, Los Angeles, CA 90095-1555, USA*
[d] *Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN 55455, USA*

Received 17 June 2003; received in revised form 21 November 2003; accepted 11 February 2004
Available online 12 April 2004

Abstract

We investigate the problem of determining visible regions given a set of (moving) obstacles and a (moving) vantage point. Our approach to this problem is through an implicit framework, where the obstacles are represented by a level set function. The visibility problem is formally formulated as a boundary value problem (BVP) of a first order partial differential equation. It is based on the continuation of values along the given ray field. We propose a one-pass, multi-level algorithm for the construction of the solution on a grid. Furthermore, we study the dynamics of shadow boundaries on the surfaces of the obstacles when the vantage point moves along a given trajectory. In all of these situations, topological changes such as merging and breaking occur in the regions of interest. These are automatically handled by the level set framework proposed here. Finally, we obtain additional useful information through simple operations in the level set framework.

1. Introduction

In this paper, we consider the visibility problem described as follows: given a collection of hypersurfaces representing the boundaries of objects, called the occluders, in two- or three-dimensional space, determine

☆ Research of the first and the third author is supported by ONR N00014-97-1-0027, DARPA/NSF VIP Grant NSF DMS 9615854 and ARO DAAG 55-98-1-0323.

Research for the last author is supported by ONR, NSF, PECASE, and CAREER.

The work of the first author is partially supported by the National Science Foundation under agreement No. DMS-0111298. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

* Corresponding author. Tel.: +1-609-258-6495; fax: +1-609-258-1735.
E-mail addresses: ytsai@math.princeton.edu (Y.-H.R. Tsai), lcheng@math.ucsd.edu (L.-T. Cheng), sjo@math.ucla.edu (S. Osher), guille@ece.umn.edu (G. Sapiro).

0021-9991/$ - see front matter © 2004 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2004.02.015

the regions of space or on the surfaces visible to a given observer. In real world applications, this problem must be solved efficiently. Generalizations of the visibility problem are just as, if not more, important; this includes the case of a moving rather than static observer and the determination of regions visible for all time or invisible for all time in this situation. We begin with the basic visibility problem for simplicity, and address parts of the dynamic problem later on.

The visibility problem can be reformulated into a problem of determining light and dark regions given a point light source. Under this point of view, a more precise set of assumptions we make in the visibility problem include: a space composed of a homogeneous medium and objects with non-reflecting and non-diffracting surfaces. Furthermore, we disregard interference, assuming that the distances between objects are large compared to the wavelength of light. Under these conditions, light rays travel in straight lines and are obliterated upon contact with the surface of an object. Thus a point is called visible with respect to a vantage point, the observer, if the line segment between the point and the vantage point does not intersect any of the obstructing objects or their surfaces in space.

Even under these simplifying assumptions, the visibility problem arises as a crucial part of numerous applications in different scientific fields, including rendering [16], visualization, etching [1], the modeling of melting ice [5], surveillance, navigation, and inverse problems, to name a few. In the case of computer graphics and rendering, for example, determination of the visible portions of object surfaces allows for those portions alone to be rendered, thus significantly saving costly computation in unnecessary (invisible) regions. In some modeling problems, such as etching and melting ice that we listed above, visibility is used to find out how certain quantities of interest accumulate, given a radiating source and a (dynamic) surface configuration. There are also variational problems that minimize the corresponding energy functionals over the visible regions of the ambient space, see e.g. [17].

Currently there are numerous algorithms for solving the visibility problem using explicit surface representations. For example, the work of [11,15] uses linearity to process triangulated surfaces. A detailed review of related work on the visibility problem, especially concerning explicit surfaces, can be found in [14]. Furthermore, there are a variety of visibility algorithms from computational geometry (see, e.g. [7] that is commonly implemented by hardware and [2,3]). These algorithms often combine special data structures and related algorithms for efficient decomposition and information retrieval of the configuration space. Some are even implemented commercially in hardware to accelerate the solution.

While explicit surfaces, for example triangulated surfaces, are used in a majority of computer graphics and vision applications, implicitly represented surfaces are gaining more attention. This is partly due to the fact that in many applications, the data (i.e. surfaces) are obtained originally and naturally in an implicit form. It is also because of the fact that more and more problems are formulated and solved using the level set method [20]. Hence, it is natural to work directly with the implicit data without converting to a different explicit representation. Currently, visibility algorithms for implicit surfaces mostly consist of sending rays out from the vantage point to a point of interest (or the reverse) and testing for intersections with the surfaces of the objects using information arising from the implicit formulation.

Another idea in determining whether a point is visible to a given observer is to compare the geodesic and Euclidean distances between the observer and that point. See [26] for an example of this approach. The geodesic distance between two points is the distance in the space in the presence of obstacles, namely the objects. Let $\mathbf{x}$ represent the point of interest and $\mathbf{x}_o$ represent the observer point. The geodesic distance can thus be calculated by solving the Eikonal equation

$$H(\phi)|\nabla u| = 1$$

with condition $u(\mathbf{x}_o) = 0$, and $u = \infty$ inside of the occluders. Here, $H(x) = \chi_{[0,\infty)}(x)$ is the characteristic function of $[0, \infty)$. Thus the point $\mathbf{x}$ is occluded if and only if

$$u(\mathbf{x}) > |\mathbf{x} - \mathbf{x}_o|.$$

However, this algorithm as it was implemented is $O(N \log N)$, where N is the number of grid points (see, e.g. [31]). Furthermore, numerical implementation of the Heaviside function may cause problems for accuracy. For other related topics, we refer the readers to [28,30,31].

Our proposed method for ray casting is different from any of the above. In essence, we send out rays in an implicit manner so as to propagate the causality relation of visibility. This implicit framework for visibility offers many other advantages. For example, the visibility information can be interpreted as the solution of simple first order PDEs and [29] offers a near optimal solution method on the grid. The dynamics of the visibility with respect to moving vantage point or dynamic surfaces can be derived and tracked implicitly within the same framework. Our method retains nearly all the benefits of a level set method, including painless Boolean operations of sets, automatic resolution of surfaces as well as the incorporation of geometric information and the handling of various surface topologies afforded. By embedding the visibility in a Lipschitz continuous function, we can obtain much more information. For example, if one is in the shadow and wants to "be seen" as soon as possible, then one can simply follow the gradient of ψ. Moreover, by the continuous nature of our solution ψ, its application to generating "soft shadows" (or "diffuse shadows") in graphics might be advantageous. Furthermore, using the same framework and the well developed level set calculus and numerics, one can start solving variational problems on the visibility numerically with reasonable efficiency [9]. This will then relate to classical "guarding cameras" or "pursuer-evader" problems in computational geometry and robotics.

In our case of visibility, a real valued function ϕ of two or three dimensions, called the level set function, is introduced. The zero level set of this function represents the surfaces of the occluding objects, and the points where ϕ is negative represent the interior of the objects. Several efficient algorithms have been developed in the literature to obtain this representation. There are also methods describing level set implementation of space partitioning schemes such as octrees. Thus our level set method for visibility will use this function ϕ whenever the objects are considered.

In our approach, the visibility problem is first formally formulated as a boundary value problem for a first order PDE whose solution is constructed by correctly extending the boundary values along the ray fields. The solution of this problem is another level set function, which we called ψ. We then introduce a multi-level algorithm for this boundary value problem for a given fixed vantage point. This algorithm constructs the occlusion boundary, the interface separating visible from invisible. At each resolution level, we solve a radially defined causality relation on a given grid *in one pass*, obtaining not only a conservative estimate of the visible and invisible regions but a locally second order approximation of the occlusion boundary. Thus, our algorithm is independent of both the convexity of the occluders and the grid geometry, and its parallelization is straightforward. In comparison with the method using geodesic distance that is described above, our algorithm in a more primitive form is $\mathcal{O}(N)$, a factor of $\log(N)$ faster.

In the second part of the paper, we extend our study to the dynamic visibility problem. In this case, we consider a moving vantage point. Obviously the static visibility problem can be applied at each time to solve this problem, and our algorithm can be used to solve it efficiently enough. However, this static approach does not give us other useful information about the dynamics; for instance, how fast a point in space will become visible or invisible. In many cases, the problem can be solved even faster if the visibility at a previous time is used effectively to produce visibility at future time. Thus we study the dynamics of curves on the occluders that separate light and dark regions on the occluders. The curves in fact can be represented using a level set approach, following the work of [6,8]. We derive motion laws for all these types of curves and evolve them under the level set framework. Thus, this part of our our work complements the book of Cipolla and Giblin [10] which discusses the reconstruction of shape from the perspective (orthogonal) projection of the horizons. To complete our study of visibility dynamics, we derive an emergence-time estimate to predict an occluded object's emergence into view. Obviously, researchers in the computer vision community have also studied similar topics, which are called visual events. However, their assumptions usually involve objects which are explicitly represented

as unions of simple geometrical shapes. We stress here again that we provide a framework to work directly with implicit data that are easy to obtain nowadays without having to convert between data representations. In computer vision, our study is related to what is called visual events. We refer the readers to [13,14,18,23].

Through out this paper, we use the following notation:

- The space in which we work will be $\mathbb{R}^d$, where $d = 2$ or 3.
- $\mathbf{x}_o$ denotes the position of the vantage point, or observer. We further assume that $\mathbf{x}_o$ never lies in the interior of the objects.
- Ω is a set of connected domains whose closure denotes the objects in question. Furthermore, let $\Gamma = \partial\Omega$.
- ϕ denotes the level set function representing the objects of interest. We may further assume that ϕ is the signed distance function to Γ. This particular level set function can be efficiently computed using fast algorithms such as the fast marching method of [31] or fast sweeping methods [30].
- We define the view direction vector pointing from $\mathbf{x}_o$ to $\mathbf{x}$ by $r(\mathbf{x}_o, \mathbf{x}) = (\mathbf{x} - \mathbf{x}_o)/|\mathbf{x} - \mathbf{x}_o|$. When the context is clear, we will drop the arguments and write simply $r(\mathbf{x})$ or r.
- Let $\mathbf{x}_1$ and $\mathbf{x}_2$ denote two points in space. We say $\mathbf{x}_1 \preceq \mathbf{x}_2$ ($\mathbf{x}_1$ is "before" $\mathbf{x}_2$) if the conditions $r(\mathbf{x}_o, \mathbf{x}_1) = r(\mathbf{x}_o, \mathbf{x}_2)$ and $|\mathbf{x}_1 - \mathbf{x}_o| \leqslant |\mathbf{x}_2 - \mathbf{x}_o|$ are satisfied. We also define the strict relation $\prec$ if the condition $|\mathbf{x}_1 - \mathbf{x}_o| \leqslant |\mathbf{x}_2 - \mathbf{x}_o|$ above is replaced by $|\mathbf{x}_1 - \mathbf{x}_o| < |\mathbf{x}_2 - \mathbf{x}_o|$.
- A point $\mathbf{y} \in \Gamma$ is called a horizon point if and only if $r(\mathbf{x}_o, \mathbf{y}) \cdot \mathbf{n}(\mathbf{y}) = 0$, where $\mathbf{n}(\mathbf{y})$ is the outer normal of Γ at $\mathbf{y}$. The horizon thus refers to the set of horizon points.
- A point $\mathbf{y} \in \Gamma$ is a terminator point if and only if there is a point $\mathbf{y}^*$ such that: (1) $\mathbf{y}^* \prec \mathbf{y}$ and (2) $\mathbf{y}^*$ is a horizon point. The terminator thus refers to the set of terminator points.
- The visible contour refers to the set of visible points of the horizons and terminators.

We remark here that what we called horizons are commonly referred to as "silhouettes" in some communities. However, after consulting a Merriam-Webster Dictionary, we decide to use the word "horizon" to emphasize that it is *a special curve on the occluder* that is defined as above, in constrast to the "silhouette", which is actually what we called a "terminator", which is the extension/projection of a horizon to a reference plane (e.g. the retinas of our eyes).

2. Implicit ray casting

We now set up the foundation of our approach and derive properties of ray casting of a single point source in an implicit framework. The motivation is that the visibility along each ray emanating from the vantage point satisfies a causality condition: if a point is occluded, then all other points farther away from the vantage point on the same ray are also occluded, i.e., if $\mathbf{x}_1$ is occluded and $\mathbf{x}_1 \preceq \mathbf{x}_2$, then $\mathbf{x}_2$ is also occluded.

We can describe the result of this causality on a sphere centered at the vantage point. Define

$$\rho(\theta) = \begin{cases} \min_{\mathbf{x}\in\mathbb{R}^d}\{|\mathbf{x} - \mathbf{x}_o| : r(\mathbf{x}_o, \mathbf{x}) = \theta, \phi(\mathbf{x}) \leqslant 0\} & \text{if exists,} \\ \infty & \text{otherwise.} \end{cases} \tag{2.1}$$

Any given point $\mathbf{x}$ is invisible if $\rho(r(\mathbf{x}, \mathbf{x}_o)) \leqslant |\mathbf{x} - \mathbf{x}_o|$. Please see Fig. 1 for an example. However, ρ is typically a piecewise continuous function with large jump discontinuity which causes some computational difficulties. In the graphics community, this is closely related to what is called the z-buffer. We defer our discussion of constructing accurate approximations of ρ in a forthcoming paper [9].

In this section, we will provide two closely related interpretations and their corresponding numerical methods. In the first interpretation, the visibility function ψ has a closed analytical expression while in the second interpretation, ψ is the solution of a boundary value problem of a first order linear PDE. The

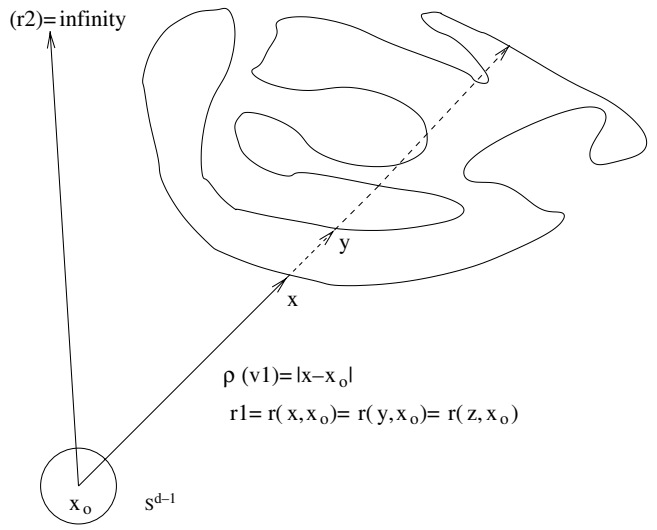

Fig. 1. This figure shows the definition of ρ.

methods to construct approximations to both formulations are closely related and we will extend this type of methods to incorporate a multi-level mesh refinement strategy for efficiency.

2.1. Embedding the visibility by a continuous function

Our first interpretation of this causality condition is to define a continuous visibility function

$$\psi(\mathbf{x}) := \min_{\mathscr{L}(\mathbf{x}_o,\mathbf{x})} \phi(\xi), \tag{2.2}$$

where $\mathscr{L}(\mathbf{x}_o, \mathbf{x})$ is the line segment connecting $\mathbf{x}_o$ and $\mathbf{x}$. Thus if $\psi(\mathbf{x})$ is negative, then $\mathbf{x}$ is occluded. We can approximate the value of $\psi(\mathbf{x})$ by $\psi^h(\mathbf{x})$ as follows:

$$\psi^h(\mathbf{x}) = \min(\psi^h(\mathbf{x}'), \phi(\mathbf{x})), \tag{2.3}$$

where $\mathbf{x}'$ is some point "immediately before" $\mathbf{x}$ in the ray direction. Here $\mathbf{x}'$ depends on the given grid structure and $\psi^h(\mathbf{x})$ can be found by linear interpolation in 2D or bilinear interpolation in 3D. See Fig. 2. We defer the details of the interpolation step to Appendix A. Denote by $G_{\mathbf{x}}$ the set of grid points needed to find $\psi^h(\mathbf{x})$ and $\mathbb{X}$ the set of grid points in which the value of ψ is already determined. Our algorithm imposes the condition that $G_{\mathbf{x}}$ is a subset of $\mathbb{X}$. This means that as long as the values of $\psi(\mathbf{x}')$ are computed ahead of the computation of $\psi(\mathbf{x})$, our algorithm will be valid. Due to the hypothesis that the lines-of-sight are straight lines emanating from $\mathbf{x}_o$. The region bounded by $\mathbb{X}$ is star-shaped with respect to the vantage point. Naturally, the first point to update ψ is then $\mathbf{x}_o$. For example, consider the case where the vantage point lies on the origin of the coordinate system. The grid points in the first quadrant can be updated row by row, starting from the positive part of the x-axis, in an increasing order of their y-coordinate component. The grid points in each row are updated from left to right, in increasing order of their x-coordinate. We can thus generalize this approach: in 3D, we consider the vantage point as the origin and approximate ψ in each octant separately, and inside each octant, we employ similar approach to what is described above.

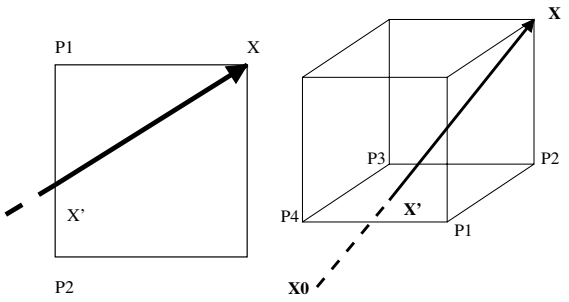

Fig. 2. A demonstration of 2D and 3D interpolation.

We write down our basic algorithm as follows:

Algorithm 2.1 (Basic visibility sweeping).
1. Set $\psi(\mathbf{x}_o) = \phi(\mathbf{x}_o)$.
2. Do a star-shaped [1] updating sequence on the grid.
3. For each grid point $\mathbf{x}$, choose $\mathbf{x}'$ depending on the grid geometry.
4. Compute the value of $\psi(\mathbf{x})$ via (2.3).

For clarity of exposition, we defer a detailed description to the next subsection, and also Appendix A. Fig. 3 shows what ψ should look like in a one dimension setting.

Before we move on to our multi-level implementation of the above algorithm, we show below that the solution ψ is no steeper than the level set function ϕ that embeds the occluders. This means that if L is a Lipschitz constant for ϕ, then it is also a Lipschitz constant for ψ.

Lemma 2.2. *For every* $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, *iff* $|\phi(\mathbf{x}) - \phi(\mathbf{y})| \leqslant L|\mathbf{x} - \mathbf{y}|$, *then* $|\psi(\mathbf{x}) - \psi(\mathbf{y})| \leqslant L|\mathbf{x} - \mathbf{y}|$.

Proof. This can be shown directly from the definition:

$$|\psi(\mathbf{x}) - \psi(\mathbf{y})| = \left| \min_{\mathscr{L}(\mathbf{x},\mathbf{x}_o)} \phi - \min_{\mathscr{L}(\mathbf{y},\mathbf{x}_o)} \phi \right| \leqslant |(\phi(\mathbf{x}_o) - L|\mathbf{x} - \mathbf{x}_o|) - (\phi(\mathbf{x}_o) + L|\mathbf{y} - \mathbf{x}_o|)| \leqslant L|\mathbf{x} - \mathbf{y}|. \quad \square$$

2.2. PDE interpretation and extensions

We just describe an algorithm for constructing our "solution", Eq. (2.2), to the single vantage point visibility problem. Here, we propose yet another approximation scheme to (2.2) using upwind finite differencing technique for solving PDEs.

Our algorithm is the following: Set $\psi = \phi$. At grid point $\mathbf{x}_{i,j}$, $\psi_{i,j}$ is computed by:
1. Solve for $\psi_{i,j}$ by upwinding:

$$\nabla\psi_{i,j} \cdot r(\mathbf{x}_{i,j}) = 0. \tag{2.4}$$

2. Update

$$\psi_{i,j} = \min(\phi_{i,j}, \psi_{i,j}).$$

[1] See Appendix A.2.

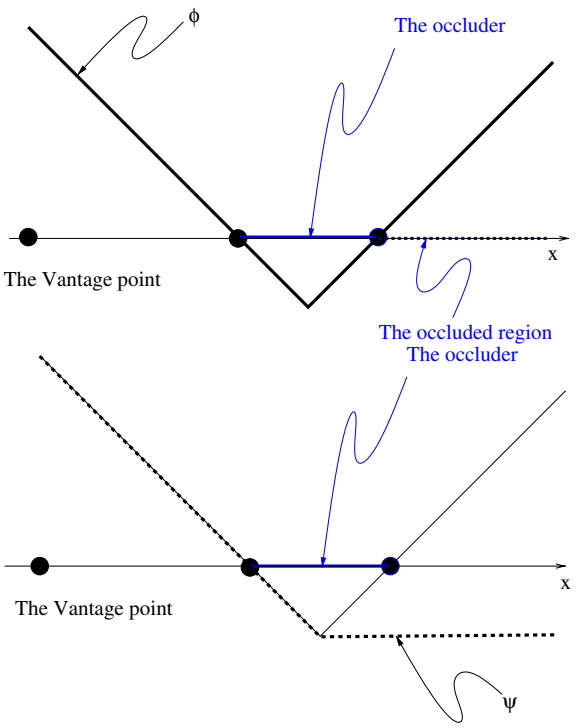

Fig. 3. A demonstration of the motivation of our implicit ray casting algorithm in one dimension.

To faciliate the description of solving Step 1, we first make the following definition.

Definition 2.3 (Finite difference operators). We define the backward finite difference operators D_x^-, D_y^- by

$$D_x^- \psi_{i,j} := -\frac{\psi_{i-1,j} - \psi_{i,j}}{\Delta x},$$

and

$$D_y^- \psi_{i,j} := -\frac{\psi_{i,j-1} - \psi_{i,j}}{\Delta y}.$$

The forward difference operators D_x^+, D_y^+ by

$$D_x^+ \psi_{i,j} := \frac{\psi_{i+1,j} - \psi_{i,j}}{\Delta x},$$

and

$$D_y^+ \psi_{i,j} := \frac{\psi_{i,j+1} - \psi_{i,j}}{\Delta y}.$$

For simplicity, we will assume that $\mathbf{x}_o = (0,0)$, $\mathbf{x} = (x,y)$, $r(\mathbf{x}) = (r_1, r_2)$, and shall introduce an upwind scheme in the first quadrant $\Omega_{+,+} = \{x \geqslant 0, y \geqslant 0, xy \neq 0\}$. In this region, Eq. (2.4) is discretized by regular upwind scheme

$$r_1(\mathbf{x}_{i,j})D_x^-\psi_{i,j} + r_2(\mathbf{x}_{i,j})D_y^-\psi_{i,j} = 0.$$

For convenience, we define $G_{+,+}$ to be the function that computes the solution for the above equation; i.e.,

$$\psi_{i,j} = \left(\frac{1}{r_1(\mathbf{x}_{i,j}) + r_2(\mathbf{x}_{i,j})}\right)^{-1}\left(r_1(\mathbf{x}_{i,j})\psi_{i-1,j} + r_2(\mathbf{x}_{i,j})\psi_{i,j-1}\right) \tag{2.5}$$

$$:= G_{+,+}(\psi_{i-1,j}, \psi_{i,j-1}). \tag{2.6}$$

Correspondingly, we can define the solution "operators" $G_{+,-}$ for the subdomain $\Omega_{+,-} = \{x \geqslant 0, y \leqslant 0, xy \neq 0\}$, $G_{-,+}$ for $\Omega_{-,+} = \{x \leqslant 0, y \geqslant 0, xy \neq 0\}$, and $G_{-,-}$ for $\Omega_{-,-} = \{x \leqslant 0, y \leqslant 0, xy \neq 0\}$, respectively, by

$$r_1(\mathbf{x}_{i,j})D_x^-\psi_{i,j} + r_2(\mathbf{x}_{i,j})D_y^+\psi_{i,j} = 0, \quad \mathbf{x}_{i,j} \in \Omega_{+,-},$$

$$r_1(\mathbf{x}_{i,j})D_x^+\psi_{i,j} + r_2(\mathbf{x}_{i,j})D_y^-\psi_{i,j} = 0, \quad \mathbf{x}_{i,j} \in \Omega_{-,+},$$

$$r_1(\mathbf{x}_{i,j})D_x^+\psi_{i,j} + r_2(\mathbf{x}_{i,j})D_y^+\psi_{i,j} = 0, \quad \mathbf{x}_{i,j} \in \Omega_{-,-}.$$

Thus, the algorithm can be written as:

Algorithm 2.4.

for $i = 0{:}n$
for $j = 0{:}m$ (i or $j \neq 0$)
$\psi_{i,j} = \min(G_{+,+}(\psi_{i-1,j}, \psi_{i,j-1}), \phi_{i,j})$.
for $i = 0{:}n$
for $j = 0{:}-1{:}-m$ (i or $j \neq 0$)
$\psi_{i,j} = \min(G_{+,-}(\psi_{i-1,j}, \psi_{i,j+1}), \phi_{i,j})$.
for $i = 0{:}-1{:}-n$
for $j = 0{:}m$ (i or $j \neq 0$)
$\psi_{i,j} = \min(G_{-,+}(\psi_{i+1,j}, \psi_{i,j-1}), \phi_{i,j})$.
for $i = 0{:}-1{:}-n$
for $j = 0{:}-1{:}-m$ (i or $j \neq 0$)
$\psi_{i,j} = \min(G_{-,-}(\psi_{i+1,j}, \psi_{i,j+1}), \phi_{i,j})$.

For $\mathbf{x_0} \in [0,\Delta x] \times [0,\Delta y]$ not lying on a grid point, we can either replace the constraint (i or $j \neq 0$) by (i or $j \notin [0,\Delta x] \times [0,\Delta y]$) in algorithm 2.4, or we can use a slightly different discretization. For example for $\mathbf{x}_{1,j}$, $1 \geqslant j \geqslant n$,

$$D_y^-\psi_{i,j} = -\frac{r_1(\mathbf{x}_{i,j})}{r_2(\mathbf{x}_{i,j})}D_x^-\psi_{i,j-1}. \tag{2.7}$$

This corresponds to the interpolations described in Appendix A for (2.3). Extension to 3D cases follows in a straightforward manner. Fig. 4 [2] provides a result of the above algorithm applied to a real city model.

[2] The authors thank Prof. John Steinhoff and Yonghu "Tiger" Wenren for their help in obtaining this data.

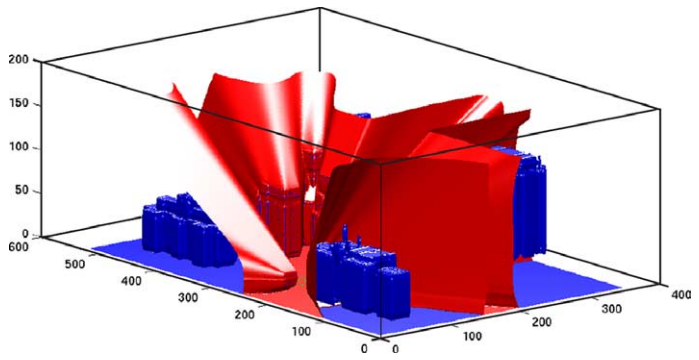

Fig. 4. A visibility result applied to a city grid. The blue surface represents the occluders, and the red surface represents the boundary between visible and invisible regions with respect to the vantage point at the position $(100, 210)$. (For interpretation of the reference to colour in this figure legend, the reader is referred to the web version of this article.)

To demonstrate the flexibility of our formulation, we consider the general case, in which the integral curves of $r(\mathbf{x})$ are more complicated than straight lines. Assume that $\psi > 0$ on a set $O \subsetneq \Omega$ and $r(\mathbf{x})$ be a differentiable vector fields such that for the integral curves starting from every interior point of Ω traces back to O. For example, our main focus in this paper has been the case in which $r(\mathbf{x}) = (\mathbf{x} - \mathbf{x}_o)/|\mathbf{x} - \mathbf{x}_o|$, and $O = \{\mathbf{x}_o\}$. In this slightly more general setting, formula (2.2) is generalized to

$$\psi(\mathbf{x}) = \min_{\mathbf{y} \in \tilde{\mathscr{L}}(O,\mathbf{x})} \phi(\mathbf{y}),$$

where $\tilde{\mathscr{L}}(O, \mathbf{x}) = \{\text{the flow line connecting } \mathbf{x} \text{ to } O\}$. We can use a variant of the fast sweeping method [30] to solve this problem. See Fig. 5 for a result under this kind of ray field, and the corresponding result computed this way.

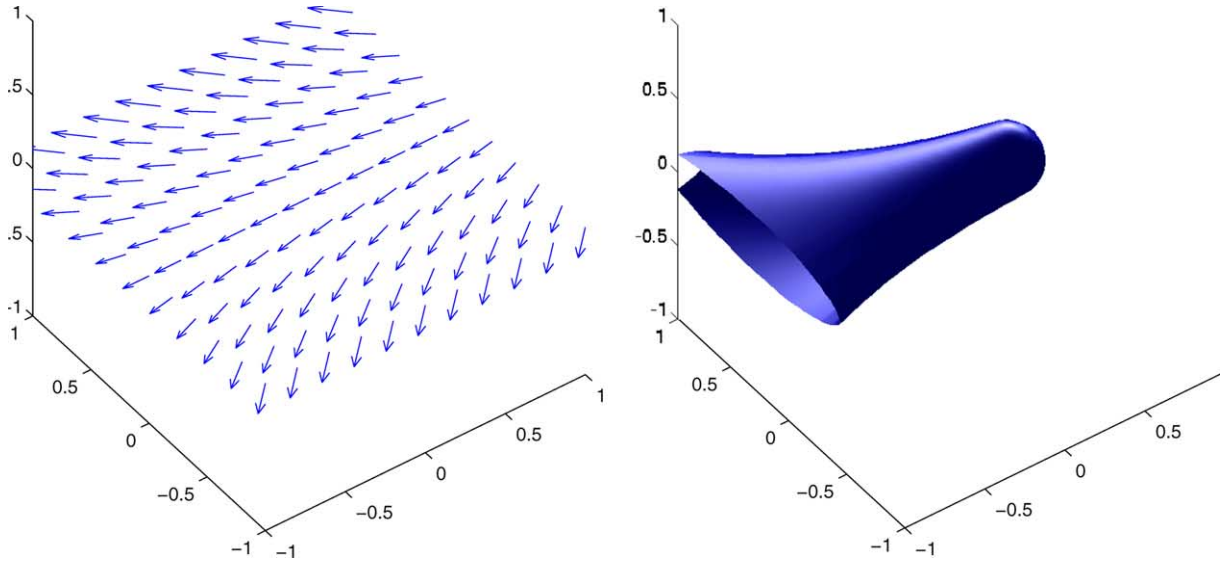

Fig. 5. Visibility under a bending ray field.

An alternative interpretation of the causality condition stated in the beginning of this section is to formulate it as a boundary value problem for the following first order PDE:

$$\nabla\psi \cdot r(\mathbf{x}) = \min(H(\psi - \phi)\nabla\phi \cdot r(\mathbf{x}), 0), \qquad \psi(\mathbf{y}) = \phi(\mathbf{y}) \quad \forall \mathbf{y} \in O, \tag{2.8}$$

where $H(z) = \chi_{[0,\infty)}(z)$ is the characteristic function of $[0,\infty)$. Consequently, we have an the a priori estimate

$$|\nabla\psi| = \max_{|r|=1} \nabla\psi \cdot r \leqslant \max_{|r|=1} \nabla\phi \cdot r = |\nabla\phi| \leqslant L,$$

i.e. Lemma 2.2 holds for solutions of (2.8). We point out, however, that much study is needed for this kind of PDEs having discontinuous coefficients.

2.3. Estimates of visibility change using Lipschitz constant

Our multi-level approach to the visibility problem requires the skipping of large regions which are determined a priori to be either visible or invisible. This hinges upon the ability of determining whether any given voxel is completely "inside" or "outside" of the objects, and can be done conservatively with the help of the Lipschitz constant of the embedding level set function ϕ.

If we assume that we know the values $\phi(\mathbf{y})$, $\phi(\mathbf{z}) > 0$, and let $\mathbf{x}$ be a point on the line segment joining $\mathbf{y}$ and $\mathbf{z}$. We would like to know if $\phi(\mathbf{x})$ can be negative. Let C be the Lipschitz constant of ϕ. It dictates the maximal decrease in values from $\mathbf{y}$ to $\mathbf{x}$ to be $L|\mathbf{y} - \mathbf{z}|$, therefore,

$$\phi(\mathbf{x}) \geqslant \phi(\mathbf{y}) - L|\mathbf{y} - \mathbf{x}|$$

and, similarly

$$\phi(\mathbf{x}) \geqslant \phi(\mathbf{z}) - L|\mathbf{z} - \mathbf{x}|.$$

Hence, if $0 \leqslant \phi(\mathbf{y}) - L|\mathbf{y} - \mathbf{x}|$ and $0 \leqslant \phi(\mathbf{z}) - L|\mathbf{z} - \mathbf{x}|$, we can conclude that the values of ϕ along the line segment connecting $\mathbf{y}$ and $\mathbf{z}$ always stay above zero.

We thus generalize the above observation to determine the sign of ϕ or ψ over a rectangular or a cubic region that we call a voxel. Let $\mathbf{x}_c$ be the center point and $\mathbf{x}_i$ the vertices of the given voxel V. If

$$\phi(\mathbf{x}_i) + L|\mathbf{x}_c - \mathbf{x}_i| < 0 \quad \forall i, \tag{2.9}$$

then we know $\phi|_V < 0$ ($V \subset \{\phi < 0\}$). Conversely, if

$$\phi(\mathbf{x}_i) + L|\mathbf{x}_c - \mathbf{x}_i| > 0 \quad \forall i, \tag{2.10}$$

then we know $\phi|_V > 0$. Since ϕ is the signed distance function, as we pointed out in the previous subsection, a Lipschitz constant of ψ can be taken to be 1.

Let ψ^h denote the approximation of ψ constructed using a grid with mesh size h. If we can obtain an L_∞ error bound $\mathscr{E}(V;\psi^h)$ of ψ^h to the analytic solution ψ on the given voxel V, we can conservatively estimate both the visible and invisible regions from our approximation ψ^h, i.e.

Case 1. If $(\psi^h(\mathbf{x}_i) + \mathscr{E}(V;\psi^h)) + L|\mathbf{x}_c - \mathbf{x}_i| < 0\ \forall i$, then $\psi|_V < 0$. (V is definitely invisible.)
Case 2. If $(\psi^h(\mathbf{x}_i) - \mathscr{E}(V;\psi^h)) + L|\mathbf{x}_c - \mathbf{x}_i| > 0\ \forall i$, then $\psi|_V > 0$. (V is definitely visible.)

2.4. A multi-resolution algorithm

We offer a mesh refinement strategy to further accelerate our one-pass implicit ray casting algorithms. Essentially, these are all applications of the method of characteristics for first order PDEs.

Assuming the ability of determining whether a given voxel lies completely inside or outside of an occluder, [3] we propagate these voxels along the rays from the vantage point, and obtained a set of voxels over which the visibility status does not change. We then refine our grid over the remaining region. This is in similar spirit to what is reported in [25].

Given a grid at resolution $2h$, we use $\mathscr{V}_{2h}^{+}$ and $\mathscr{V}_{2h}^{-}$ to represent the subsets of Ω over which the analytic solution ψ is determined to remain positive and negative, respectively. Furthermore, let $\mathscr{V}_{2h} = \mathscr{V}_{2h}^{+} \cup \mathscr{V}_{2h}^{-}$ and let $\partial\mathscr{V}_{2h}^{\pm}$ denote that boundary of $\mathscr{V}_{2h}^{\pm}$. We refine the grid over the region $\Omega_h = \Omega_{2h} \setminus \mathscr{V}_{2h}$ and subsequently determine the sets $\mathscr{V}_{h/2}^{\pm}$.

One way to find $\mathscr{V}_{h/2}^{\pm}$ is to solve ψ^h for the same PDE with different boundary conditions:

$$\begin{cases} \nabla\psi^h \cdot r(x) = \min(H(\psi - \phi)\nabla\phi \cdot r(x), 0) & \text{for } x \in \Omega_h^{(0)}, \text{ the interior of } \Omega_h, \\ BC : \psi^h(x) = \begin{cases} \phi(x) & \text{if } x \in \partial\mathscr{V}_{2h}^{+} \text{ and } r(x) \cdot n_\Omega(x) > 0, \\ \psi_{\text{int}}^{2h}(x) & \text{if } x \in \partial\mathscr{V}_{2h}^{-} \text{ and } r(x) \cdot n_\Omega(x) > 0, \end{cases} \end{cases} \tag{2.11}$$

where n_Ω is the inner normal of $\partial\Omega_h$, and ψ_{int}^{2h} is a linear interpolant of the grid function ψ^{2h}. We then determine $\mathscr{V}_h$ using the analysis shown in the previous section. Hence, we can repeat this procedure until the desired resolution is reached. This approach, however, requires either a priori or a posteriori estimates of the error to the viscosity solution of the given problem. It will be reported in a forthcoming paper by the authors.

The second way is the following: At each resolution level, we construct another function M such that for each voxel V in the domain, $M_h(x) = (\tilde{V}, \sigma, \bar{\phi})$, where the first component $\tilde{V}$ is the "source" voxel of V (see Footnote 3), the second component is the visibility of V ($\sigma = 1$ if visible, -1 if invisible, 0 inconclusive), and the third will be the constant continuation of the values of ϕ along $r(x)$ from the "source" voxels.

We first identify those V_j on which the embedding level set function ϕ is negative; these are the "source" voxels. Set $M(x) = (V_j, -1, \bar{\phi})$. Find V_o that contains $\mathbf{x}_o$. If ϕ is positive over V_o, set $M(x) = (V_o, 1, \bar{\phi})$. Let $\Omega_h = \Omega_{2h} \setminus ((\bigcup_j V_j) \cup V_o)$. We then "solve" the following problem by method of characteristics:

$$\begin{cases} \vec{\nabla} M_h \cdot r(x) = 0 & \text{for } x \in \Omega_h^{(0)}, \\ BC : M_h(x) = M_{2h}(x) & \text{if } x \in \partial\mathscr{V}_{2h}^{+} \cup \partial\mathscr{V}_{2h}^{-} \text{ and } r(x) \cdot n_\Omega(x) > 0. \end{cases} \tag{2.12}$$

In a 2D setting, at each vertex x of V, $M_h(x) = \theta M(x_j) + (1-\theta)M(x_k)$ for some upwind neighbors x_j and x_k, and some $\theta \in [0,1]$ determined by $r(x)$. However, we modified $M_h(x)$ after the update formula by rounding the second component of M_h that is neither 1 nor -1 to 0. Finally, a voxel V is determined to be in $\mathscr{V}_h^{-}$, or $\mathscr{V}_h^{+}$, if, on V, the second component of M_h is -1, or 1, and the voxels referred to by M_h are all immediate neighbors to each other. Please see Fig. 6 for a demonstration of this algorithm.

As for complexity, the operation count for our multi-level algorithms is $O(N^{d-1}\log N)$. Here $N = 1/h$ where h is the smallest spatial stepsize used in the multiresolution framework and d is the dimension of the space. The N^{d-1} part of the complexity comes from the fact that a codimension one hypersurface in d-dimensional space is being generated under fast sweeping and the $\log N$ part comes from multiresolution. The memory allocation of our algorithm is also $O(N^{d-1}\log N)$, with the $\log N$ part once again due to multiresolution.

We point out here that for applications in which the occluders are triangulated and one is only interested in the visibility information projected on an image plane, there are already many specialized algorithms and hardware designed available. The purpose of our algorithm is to work with implicit data and find visibility information in the whole ambient space without the costly operations of changing representations.

[3] Later on, we will call these voxels the "source" voxels.

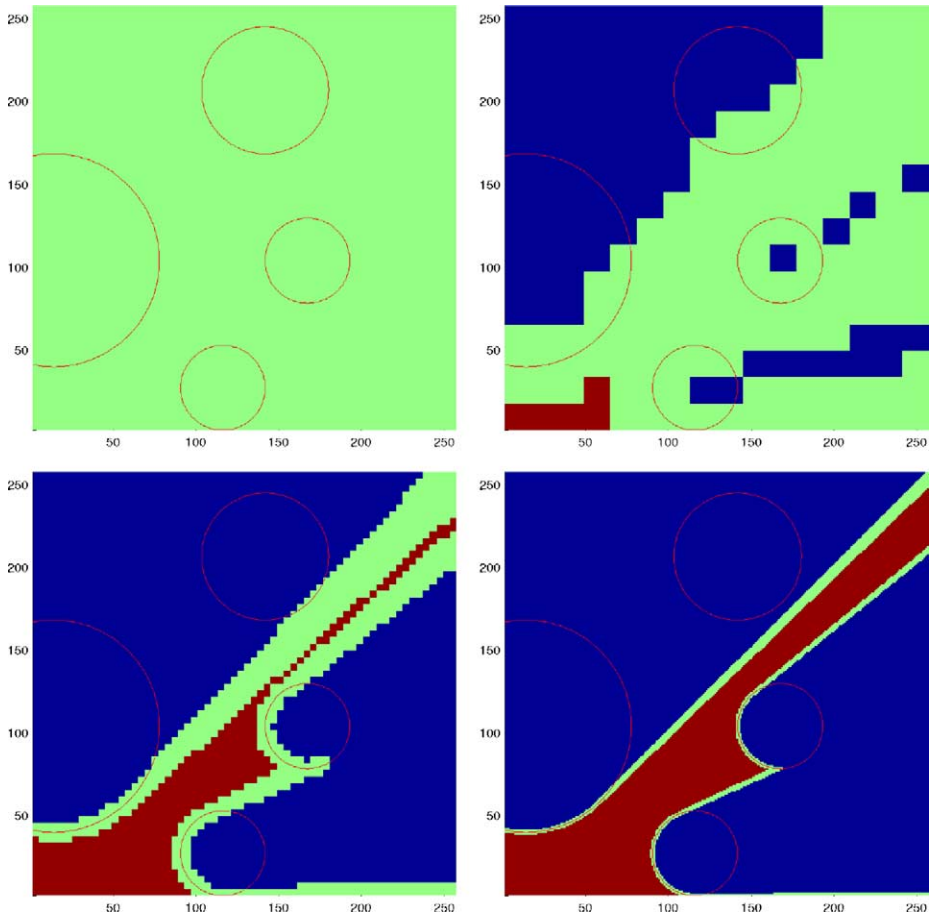

Fig. 6. This is a schematic diagram for the multi-resolution algorithm. Occluded voxels are depicted in blue and visible ones in red. The regions are target for next level refinement. The red curves represent the boundaries of the occluders, and the vantage point is positioned at $(1,1)$. The sizes of the voxels are: 64×64, 16×16, 4×4, and 1×1.

2.5. Multi-scale considerations

If we consider the visibility problem in applications related to human vision, such as 3D virtual environment rendering, it is natural to put a scale parameter into the size of the objects related to the distance of the object from the vantage point. We want to ignore certain *isolated and small* objects that are *far away* from the vantage point using this information. It is important to notice that a collection of closely positioned small objects can form a visible ensemble, seen for example in clouds and trees. This observation has been implemented in the work reported in [27].

Using the level set representation of the virtual environment in conjunction with the solution properties of certain PDEs, we are able to deal with this issue easily without explicitly considering each object separately. The idea is to *dilate* the interface first so that small objects can merge to form ensembles of larger size. We then *shrink* the interfaces (one possibility is to perform curvature driven motion) such that remaining small objects will disappear (see Fig. 7). The result of this approach follows the regularization effect of viscosity solution theory for Hamilton–Jacobi equations. It is basic mathematical morphology, and can be done easily, see e.g. [4,24].

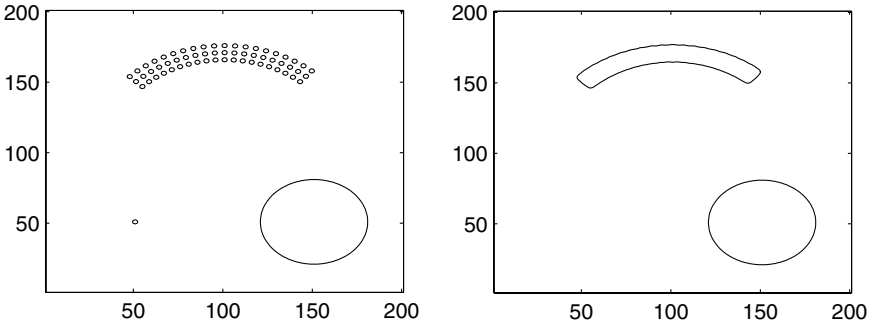

Fig. 7. An example of grouping.

We shall return to a brief discussion of scales at the end of this paper, after we discuss the dynamics of visibility.

3. Dynamic visibility

In this section, we introduce an implicit framework for tracking the change in visibility when the vantage point or the underlying occluders are changing in time. Typically, disconnected components of the invisible regions may merge and one single connected component may break up into several pieces; completely occluded objects may emerge into the scene and occlude parts of the domain that was previously visible. In many of these situations, the topology of the occlusion changes and implicit methods such as the level set methods become attractive options.

For simplicity, we consider the case in which the vantage point is moving and the ocluders are stationary. We formulate the visibility problem so that the points which are on the boundaries of the visible regions on the surfaces of the occluders can easily be identified. The dynamics of these points are derived so that one can track the visible regions according to the motion of the vantage point $\mathbf{x}_o$.

For a single convex object, the horizon determines the visibility information on the surface. Therefore, tracking the motion of the horizon for all time gives us incremental information on the change of the visible portion of the object. For non-convex objects or multiple objects, these horizon extends the rays to other parts of the surfaces, and thereby creating another type of occlusion boundaries which we coin "terminators", based on The Merriam-Webster Dictionary. We remark that "terminators" correspond to shadow boundaries if $\mathbf{x}_o$ is viewed as a point light source. Similar to the horizons, one side of a terminator is visible while the other not. In summary, the points forming the boundaries of visible regions on given surfaces can be placed into two categories:

- points that are part of the horizon;
- points that border *shadows cast by some surface* (terminators).

We shall see that the motion of the horizon is characterized by the orthogonality constraint and it, in turn, becomes a part of the constraints of the terminator motion.

In our level set formulation, we create a continuous function whose zero level set captures the curves described above. We also propose a method that relates each point on the terminator to a point on the horizon of the surface casting the shadow. This description should be global, that is, quantities should vary "continuously" with respect to points not on the surface.

For a single convex object, tracking the horizon is certainly the optimal solution. With the fast sweeping algorithms and local storage strategies, the complexity of the level set approach to track the horizon and

terminator curves is formally $\mathcal{O}(N)$ both in operation count and in storage. Here, the number N is the number of points used to resolve the curves. In more realistic applications, we need to consider the situations in which (1) the velocity fields for the horizons and terminators become singular, (2) some "hidden"

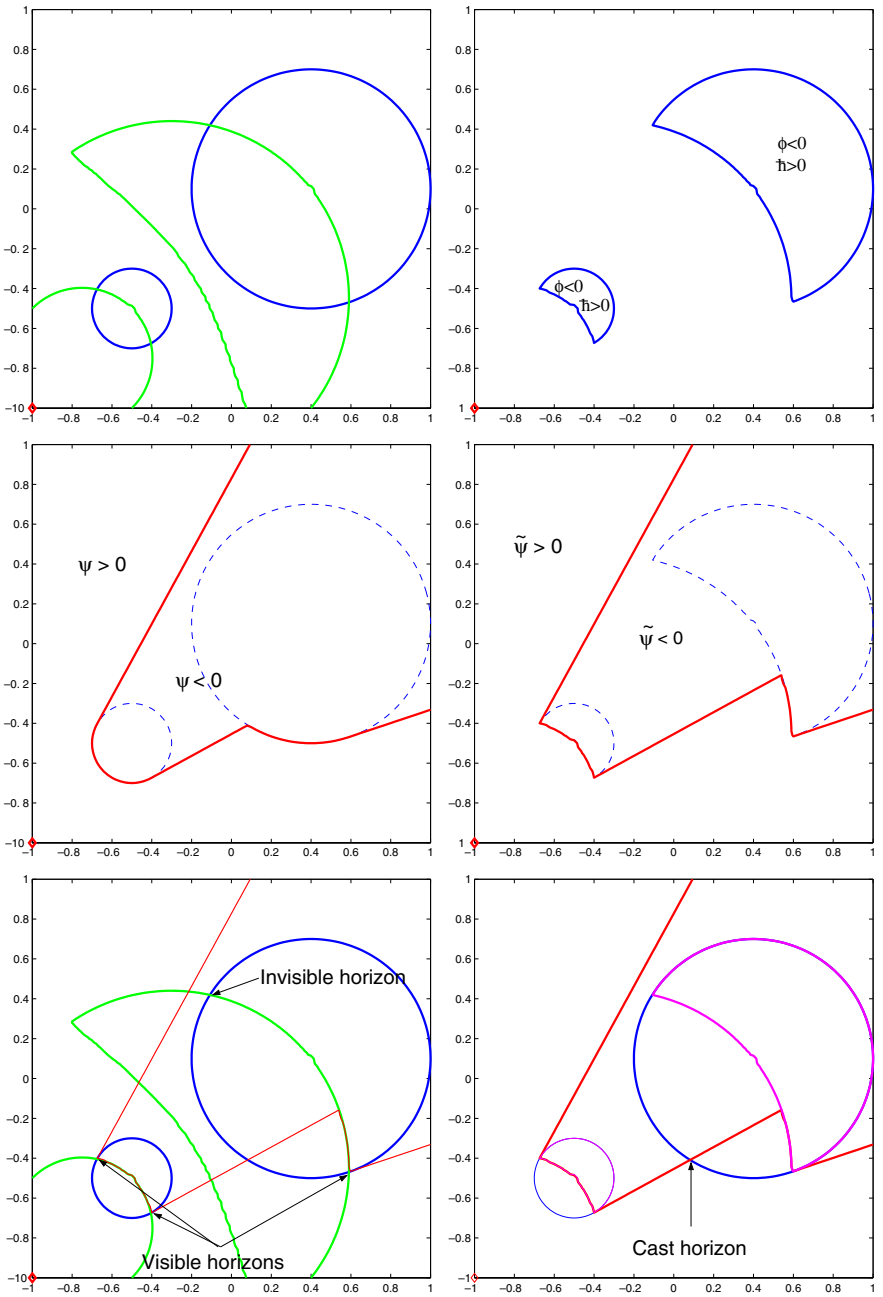

Fig. 8. Finding the visible horizons and their casts. The occluders are the two circles depicted by the blue curves, and the vantage point is located at $(-1,-1)$. The green curves are the zero level set of $\hbar$. Visible horizons and their casts are characterized by the intersections of different level set functions as described in the text.

object may appear and create new terminators. Indeed, these are among the most difficult problems in this area. We will address this difficulty in a later subsection. If the visibility information in the whole ambient space is needed, we think that our previous "static" approach might generate more elegant and easier-to-implement algorithms with comparable performance (see Fig. 13). However, the additional information about how shadow boundary moves may be used together with the static visibility algorithm for many dynamic type applications.

In the following, we will first propose an implicit characterization of the horizons and their terminators. The procedures involved consist of our "static" algorithm and some simple Boolean operations on sets.

3.1. Finding the horizon and the terminator implicitly

3.1.1. Finding the horizon

We extend the orthogonality condition that defines the horizon and arrive at

$$\hbar(\mathbf{x},t) = (\mathbf{x}-\mathbf{x}_0)\cdot\nabla\phi(\mathbf{x}). \tag{3.1}$$

We observe that its zeros correspond to the critical point of ϕ along the flow lines that pass through, i.e. $\psi(\mathbf{x}) = \phi(\tilde{\mathbf{y}})$ for some $\tilde{\mathbf{y}} \in \tilde{\mathscr{L}}(O,\mathbf{x})$ and $\nabla\phi(\tilde{\mathbf{y}})\cdot r(\tilde{\mathbf{y}}) = 0$. We also noted above that $\hbar$ completely determines the visibility of any convex object embedded in ϕ

$$\{\hbar(\mathbf{x}) \leqslant 0\} \cap \{\phi = 0\} \iff \text{visible}.$$

In general cases, where there are multiple objects (convex and non-convex), $\hbar$ does not give exact visibility information anymore. It just provides local visibility information just as local extrema may not be absolute extrema.

Due to its definition (3.1), $\hbar$ will still be non-negative on the parts of the surface facing the source, even though those parts are completely occluded. Instead, $\hbar$ gives a conservative "estimate" of the shadow

$$\{\hbar(\mathbf{x}) > 0\} \cap \{\phi = 0\} \Rightarrow \text{invisible}.$$

Thus the *visible* horizon is

$$\{\phi = 0\} \cap \{\hbar = 0\} \cap \{\psi \geqslant 0\},$$

where ψ is the visibility function coming from our static algorithm. Fig. 8 gives an example of horizons found this way.

3.1.2. Finding the terminator

How do we find the terminator? Our idea is to overshoot each ray that is tangent to the visible horizons when it hits another part of Γ; thus the intersections of these rays and Γ correspond to exactly the terminators. This "overshooting" strategy is, of course, implemented by an auxilliary level set function $\tilde{\psi}$, and $\{\tilde{\psi} = 0\}$ will cut through Γ on the terminator, therefore, providing an implicit representation of the terminator. Consider $\tilde{\phi} = \max(\phi, -\hbar)$, then $\{\tilde{\phi} \leqslant 0\}$ corresponds to the set $\{\hbar \geqslant 0\} \bigcap \{\phi \leqslant 0\}$, a set created by "carving off" a neighborhood of the visible portions the occluders. We notice that the occlusion generated by the set $\{\hbar \geqslant 0\} \bigcap \{\phi \leqslant 0\}$ is the same [4] as $\{\phi \leqslant 0\}$. Therefore, we can construct $\tilde{\psi}$ as the solution of (2.8) with $\tilde{\phi}$ on the right-hand side, instead of ϕ. Consequently, the shadow boundaries, $\{\tilde{\psi} = 0\}$, extending from the horizons cut through portions of the original occluders $\{\phi \leqslant 0\}$, and the intersections corresponds to the terminators. Fig. 8 shows the operations described above in a simple two circle setting. Additionally, we

[4] Modulo a small subset of $\{\phi \leqslant 0\}$, which we know is invisible by definition.

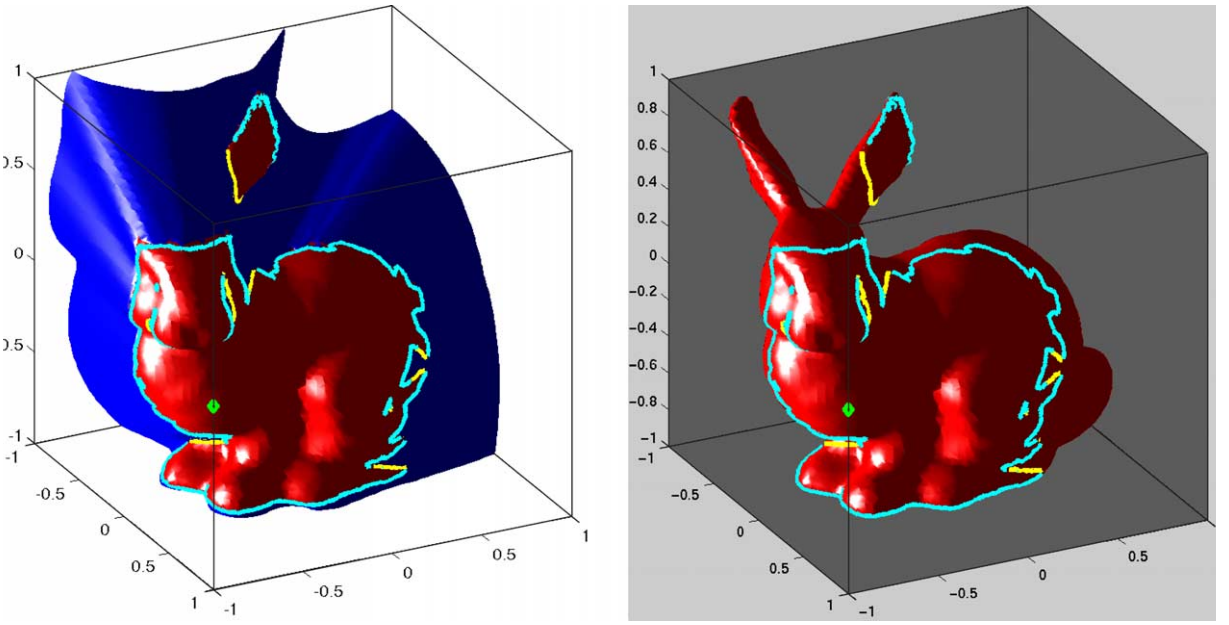

Fig. 9. Visible contour (portions of horizon and terminator that are visible).

can even make $\{\tilde{\psi} = 0\}$ perpendicular to Γ locally around the intersection by evolving the following PDE for a small amount of time (see [19]):

$$\tilde{\psi}_\tau + \mathrm{sgn}(\phi)\nabla\tilde{\psi} \cdot \frac{\nabla\phi}{|\nabla\phi|} = 0, \quad \tilde{\psi}(\mathbf{x}, \tau = 0) = \tilde{\psi}(\mathbf{x}).$$

Here τ is a pseudo-time related to the PDE. If the system is evolved to τ_1, then the zero level set of $\tilde{\psi}(\mathbf{x}, \tau_1)$ will be orthogonal to $\{\phi = 0\}$ in a tube with radius τ_1 around $\{\phi = 0\} \cap \{\tilde{\psi}(\mathbf{x}, \tau_1) = 0\}$. We then use $\tilde{\psi}(\mathbf{x}, \tau_1)$ as $\tilde{\psi}(\mathbf{x})$.

With these characterizations, we can easily identify the visible contours. See Figs. 9–12 [5] for examples. In these figures, the visible portions of the horizons and the terminators are depicted as cyan and yellow curves, respectively. A green circle is drawn to reveal the location of the vantage point in each setting. The boundaries between visible and invisible regions are represented by blue surfaces. We observe that the blue surfaces cut through the objects exactly at the visible contours.

3.2. The dynamics of the horizon

Let $\mathbf{x}_o(t)$ be the position of the vantage point and $\mathbf{x}(t)$ be a corresponding point on the horizon at time t. We first consider a single convex occluder Ω embedded by the signed distance function ϕ. This translates into the following constraints on $\mathbf{x}(t)$:

$$\begin{cases} \phi(\mathbf{x}(t)) = 0, \\ (\mathbf{x} - \mathbf{x}_o) \cdot \nabla\phi(\mathbf{x}) = 0. \end{cases} \tag{3.2}$$

[5] The terrain data were obtained from ftp://ftp.research.microsoft.com/users/hhoppe/data/gcanyon/.

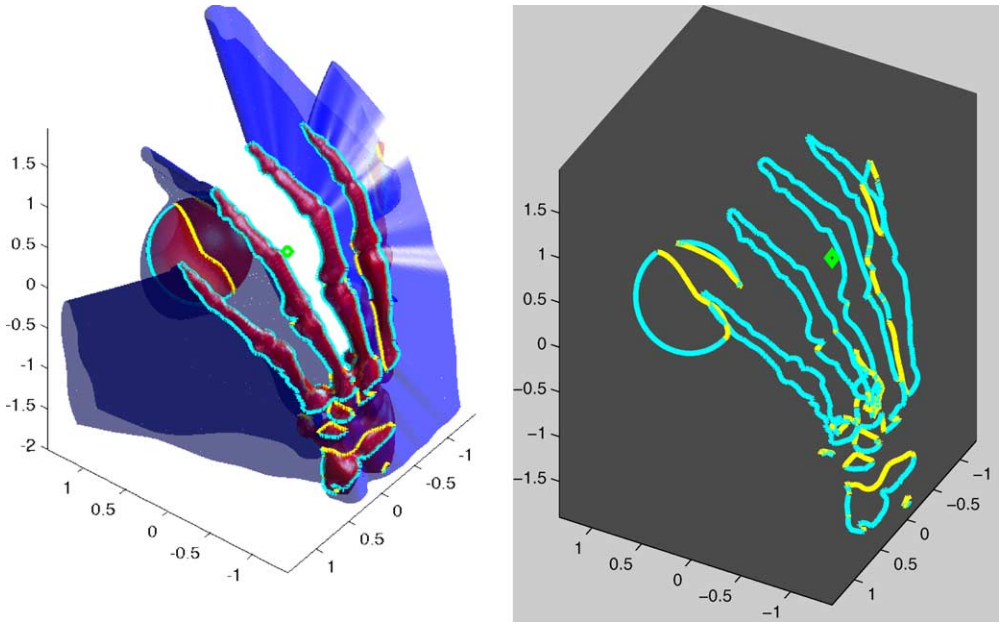

Fig. 10. Visible contour (portions of horizon and terminator that are visible).

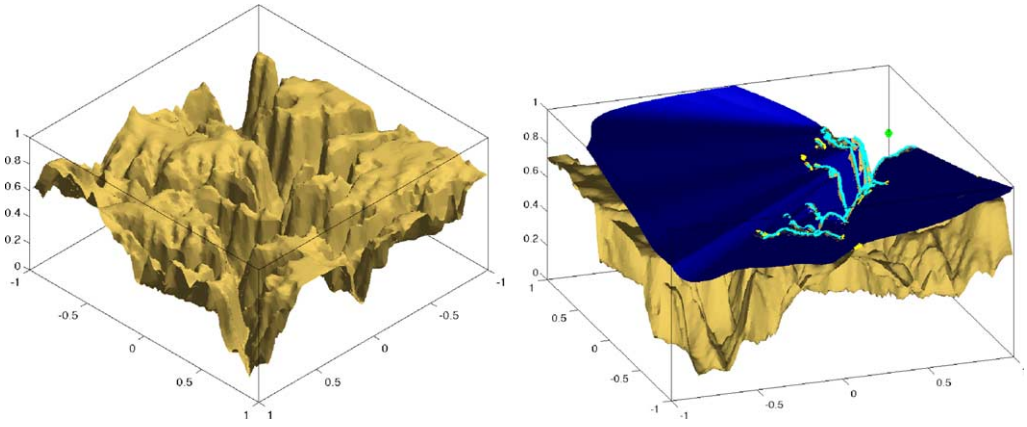

Fig. 11. Silhouettes and terminators obtained from the elevation data of Grand Canyon.

Let $n(\mathbf{x})$ denote the outer normal of $\partial\Omega$ at $\mathbf{x}$. In our level set framework, $n(\mathbf{x})$ is the restriction of $\nabla\phi/|\nabla\phi|$ to the zero level set of ϕ. In two dimensions, we can invert the above constraints and derive that

$$\dot{\mathbf{x}} = \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \frac{1}{\kappa} \frac{\dot{\mathbf{x}}_o \cdot n(\mathbf{x})}{|\mathbf{x} - \mathbf{x}_o|} r(\mathbf{x}). \tag{3.3}$$

Here, κ is the curvature of the occluding surface at $\mathbf{x}$. This formula is derived in detail in Appendix A. In three space dimensions, the horizon becomes a closed curve $\Gamma(s) = \mathbf{x}(s,t)$, where s is the arc length of $\Gamma(s)$. Let P be the plane tangent to $\dot{x}_o$, passing through $\Gamma(s)$ and x_o. Let $\beta(\sigma)$ be the curve on the intersection of P and $\partial\Omega$. Then, locally at t and x, we have a two-dimensional visibility problem on the plane P, in which $\beta(\sigma)$

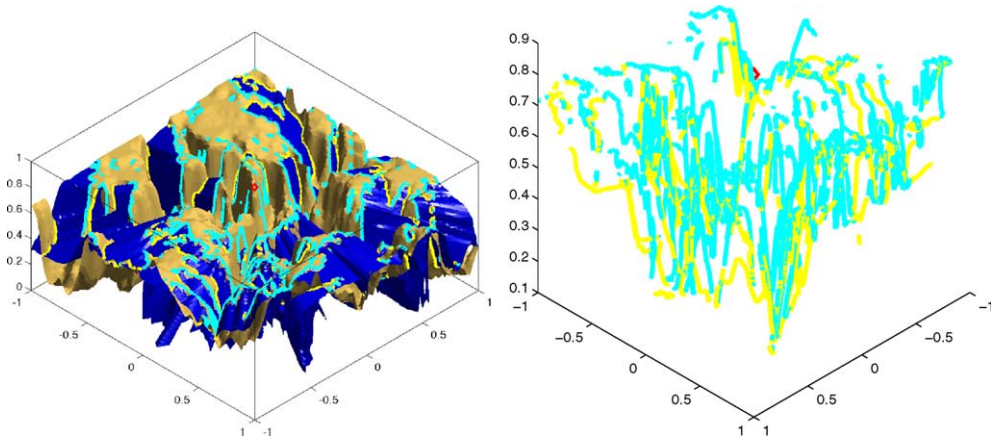

Fig. 12. Terminators and terminators obtained from the elevation data of Grand Canyon.

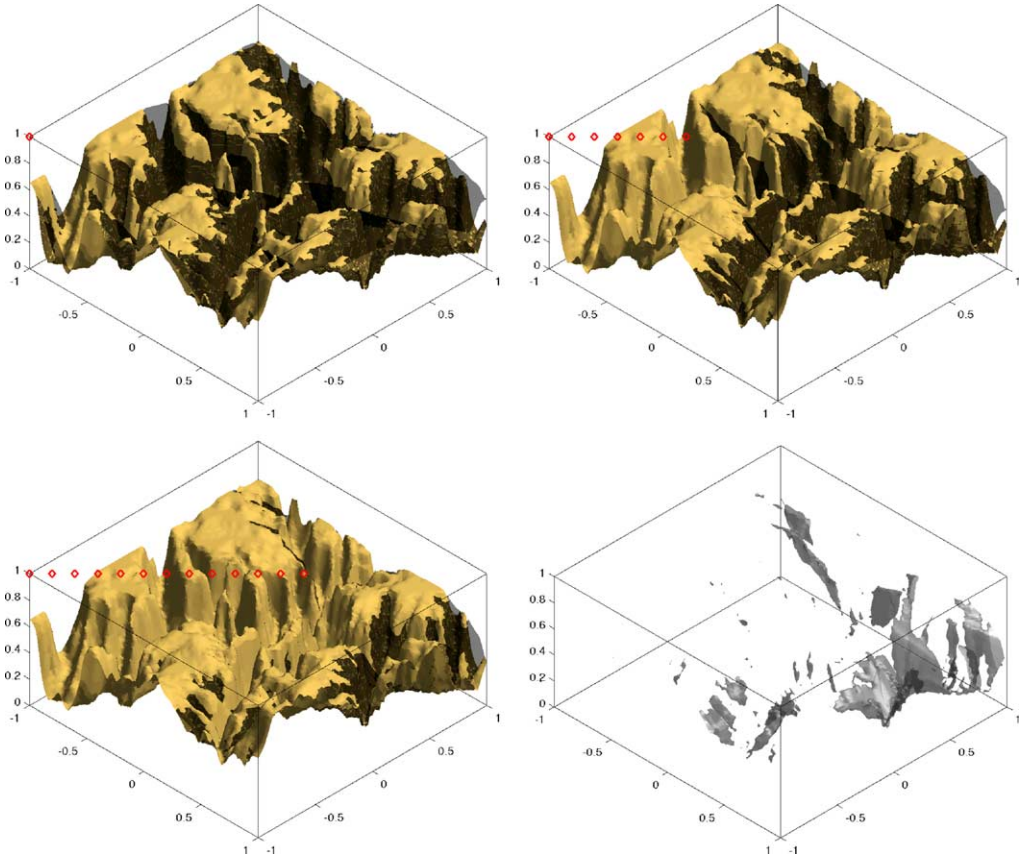

Fig. 13. By taking the intersection of the occlusion during a trajectory of the observer, we can find the cumulative occlusion easily and efficiently. The following pictures show a progression of the cumulative occlusion subject to an observer (“spy plane”) moving across a region of Grand Canyon.

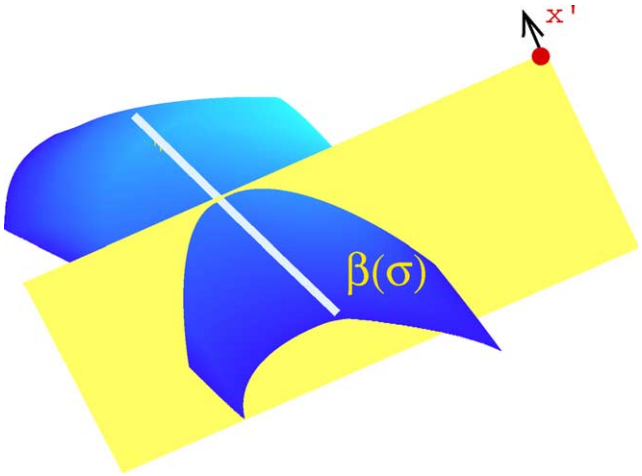

Fig. 14.

defines the boundary of the objects. Following this reasoning, κ should naturally be taken from $\beta(\sigma)$. See Fig. 14.

Alternatively and more naturally under our level set formulation, we rederive the above motion law as

$$\dot{\mathbf{x}} = \frac{II(\mathbf{x}-\mathbf{x}_0)}{|II(\mathbf{x}-\mathbf{x}_0)|^2}\left(\dot{\mathbf{x}}_0 \cdot \frac{\nabla\phi}{|\nabla\phi|}\right), \tag{3.4}$$

where II is the *second fundamental form*, which can conveniently be extended to the other level sets and takes the form

$$II = \frac{1}{|\nabla\phi|} P_{\nabla\phi} \nabla^2\phi P_{\nabla\phi}.$$

Here, $P_{\nabla\phi}$ is the orthogonal projection matrix projecting vectors to the plane with normal vector parallel to $\nabla\phi$. Thus the κ in (3.3) denotes the normal curvature of the surface in the viewing direction.

For a detailed derivation and implementation, please see Appendixes A.4, A.3, and A.6. Fig. 15 shows a result of horizon motion on a non-convex body.

3.3. The dynamics of the terminator

Assume that $\mathbf{x}$ is a terminator point and $\mathbf{x}^*(\mathbf{x})$ is its generator. In two dimensions, the motion of $\mathbf{x}$ is determined by the following constraints:

$$\begin{cases} \phi(\mathbf{x}) = 0, \\ \frac{\mathbf{x}-\mathbf{x}_o}{|\mathbf{x}-\mathbf{x}_o|} = \frac{\mathbf{x}^*-\mathbf{x}_o}{|\mathbf{x}^*-\mathbf{x}_o|}. \end{cases} \tag{3.5}$$

Inverting, we find that the motion of the terminator can be written as follows:

$$\dot{\mathbf{x}} = \frac{1}{r \cdot \mathbf{n}(\mathbf{x})}\left(\frac{|\mathbf{s}|}{|\mathbf{s}^*|}\dot{\mathbf{s}}^* \cdot (r^*)^{\perp} + \dot{\mathbf{x}}_o \cdot r^{\perp}\right)\mathbf{n}^{\perp}(\mathbf{x}), \tag{3.6}$$

where r^* is the velocity of the generator $\mathbf{x}^*$,

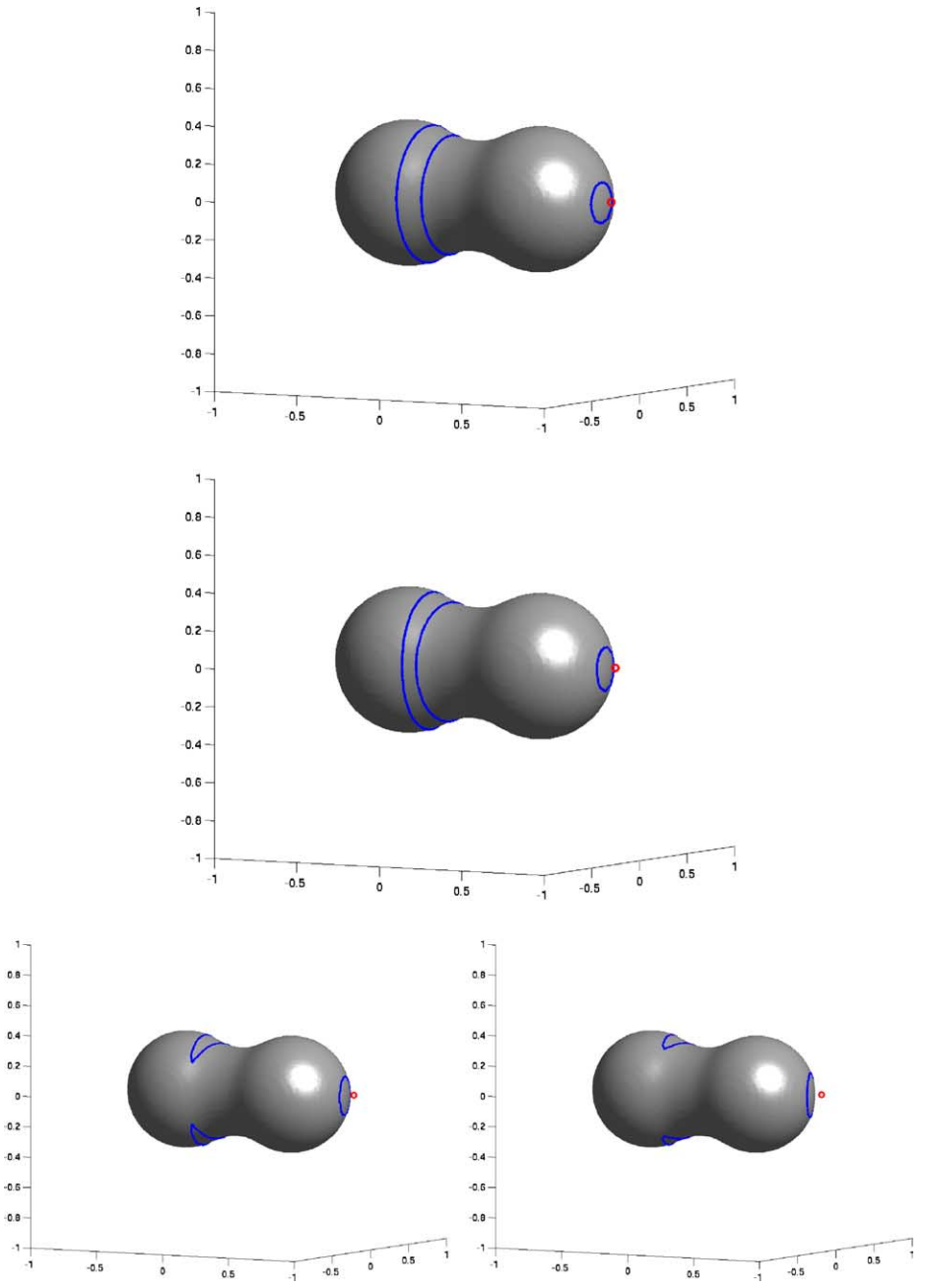

Fig. 15. A example of moving silhouette around a non-convex occluder. Observe that the silhouette curves break and change topology.

$$\mathbf{n}^{\perp}(\mathbf{x}) := \begin{pmatrix} \phi_{x_2}(\mathbf{x}) \\ -\phi_{x_1}(\mathbf{x}) \end{pmatrix} \Big/ |\nabla \phi(\mathbf{x})|,$$

and similarly for $r^{\perp}$. See Appendix A.5 for a detailed derivation. See Fig. 16 for a computational result using this formula. We notice that these constraints also tell us how the shadow boundaries should move.

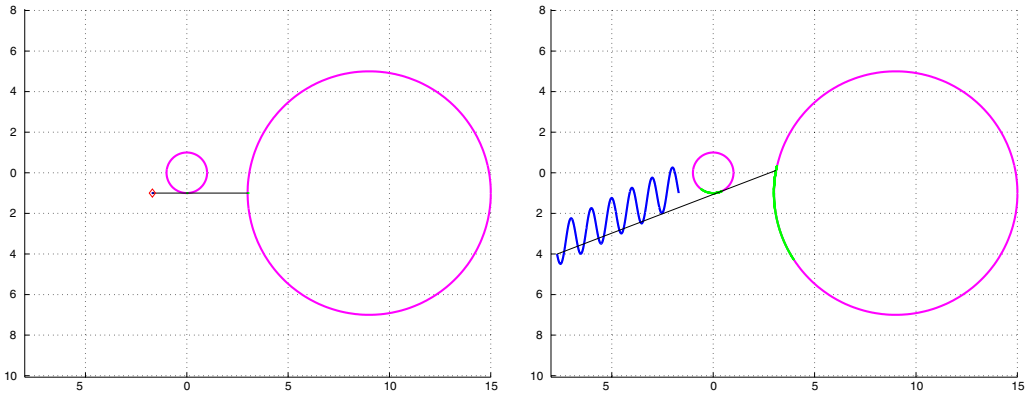

Fig. 16. A result of tracking the terminator and terminator motion using the formulas derived in this paper. The blue curve represents the trajectory of the vantage point and the green curves represent the paths of the terminator and terminator. The black line links the current position of the vantage point and the terminator; it shows that the colinearity of the vantage point, the horizon and its terminator is preserved. (For interpretation of the reference to colour in this figure legend, the reader is referred to the web version of this article.)

In three dimensions, we can reduce the instantaneous motion to a two-dimensional problem on the correct section of the surface following the reasoning given in the previous subsection.

3.3.1. Motions of the shadow boundaries

How does the shadow move in space? We can constrain a point on the shadow boundary to move only normal to the viewing direction (ergo, the shadow boundary):

$$\begin{cases} \mathbf{x}' \cdot r = 0, \\ \frac{\mathbf{x}-\mathbf{x}_o}{|\mathbf{x}-\mathbf{x}_o|} = \frac{\mathbf{x}^*-\mathbf{x}_o}{|\mathbf{x}^*-\mathbf{x}_o|}. \end{cases} \tag{3.7}$$

3.3.2. Motions of horizons and terminators of dynamic surfaces

We remark that we are able to derive the motion laws of the visible contours even when the occluders are changing shapes. In this case, the embedding level set function ϕ is a function of space and time, $\phi(\mathbf{x}, t)$ and differentiating formulas (3.2) and (3.5) with respect to t will bring $\phi(\mathbf{x}, t)$ into the equations.

3.4. Analysis of the motion

The formulae derived in the previous subsection can be regarded as a Lagrangian description of the horizon/terminator motion. We extend the velocity to the domain near the surfaces and obtain the corresponding velocity field $\mathbf{v}(\mathbf{x})$. We then evolve the level set function(s) u by

$$u_t + \mathbf{v} \cdot \nabla u = 0.$$

In the context of our visibility framework, u denotes the level set functions involved, such as $\phi, \psi, \ldots$, etc. The velocity fields for a horizon and its terminator do not depend on the function u. In horizon motion, we evolve $\hbar$, the velocity is a function of position, time, $\dot{\mathbf{x}}_o$, and the derivatives of ϕ, i.e. $\mathbf{v} = \mathbf{v}(t, \mathbf{x}, \mathbf{x_o}, \nabla\phi, D^2\phi)$. Furthermore, the level set function to be evolved is $\hbar$. In terminator motion, we evolve $\tilde{\psi}$, we have $\tilde{\mathbf{v}} = \tilde{\mathbf{v}}(t, \mathbf{x}, \mathbf{x_o}, \nabla\phi, D^2\phi, \hbar)$. Therefore, we are evolving the following two level set equations:

$$\hbar_t + \mathbf{v}(\mathbf{t}, \mathbf{x}, \mathbf{x}_o, \nabla\phi, D^2\phi) \cdot \nabla\hbar = 0,$$

$$\tilde{\psi}_t + \tilde{\mathbf{v}}(\mathbf{t}, \mathbf{x}, \mathbf{x}_o, \nabla\phi, D^2\phi, \hbar) \cdot \nabla\tilde{\psi} = 0.$$

These are simple convection equations whose viscosity solutions are well studied, provided that the velocity fields are bounded. We only have to be careful near singularities.

Formula (3.3) reveals a few interesting facts. First, we notice that the speed of the horizon motion is inversely proportional to the normal curvature in the viewing direction and to the distance between the horizon and the vantage point. If the vantage point is moving in the tangent direction v, the horizon will not move (since $\dot{\mathbf{x}}_o \cdot \mathbf{n} = 0$). The speed of the horizon motion becomes singular if the curvature of the surface at the horizon location becomes zero. On strictly convex objects, this will never happen. If we restrict our analysis to a single connected smooth non-convex object, we see easily that at the instance in which a horizon point moves into the location where $\kappa = 0$, a neighborhood of this location becomes completely visible. This signifies the disappearance of the horizon point. If the course of the vantage point is reversed, we get the genesis of a new horizon point.

Formula (3.6) tells us that the motion of a terminator point becomes singular when it is a horizon point ($v \cdot \mathbf{n} = 0$). On a single non-convex smooth surface, this happens precisely when a horizon point and its cast across the concavity collide into each other at the location where $\kappa = 0$. In the setting where there are multiple strictly convex objects, this also describes the changing of the terminator into a visible horizon point which is previously invisible. Therefore, the singularities of the horizons and terminators describe a part of their genesis. A complete genesis of the visible contours includes another part, in which a hidden object suddenly becomes visible. We shall discuss this point in a later subsection.

3.5. Relating horizon and its terminator

To move the terminator, following the notation used in the previous section, we need to find $\mathbf{x}^*(\mathbf{x})$ for each point $\mathbf{x}$ on the terminator. Of course, on the continuum level, $\mathbf{x}$ and $\mathbf{x}^*(\mathbf{x})$ are related by

$$\mathbf{x}^*(\mathbf{x}) := \mathbf{x} - \omega(\mathbf{x}) r(\mathbf{x}),$$

$\omega(\mathbf{x})$ can be computed by

$$\omega(\mathbf{x}) = |\mathbf{x} - \mathbf{x}_o| - \rho(r(\mathbf{x})),$$

where r and ρ are defined. In fact, finding ρ or ω is equivalent to solving the visibility problem. Here, we propose an implicit method to find the connection between a horizon point and its terminator in a fashion consistent with our PDE approach; i.e. propagating information along the characteristics of a first order PDE.

In general, let $r(\mathbf{x})$ be the ray vector field. We can propagate any "seeding" horizon point along the ray. We will create a vector field P by the following procedures:

1. Set $P = P_0 \notin \Omega$.
2. Let $\hbar$ be defined as in (3.1), T_a be a thin tube of radius a around the horizon, and $\partial H_a = T_a \cap \{\hbar \geqslant 0\}$. Solve by upwinding

$$\vec{\nabla} P \cdot r(\mathbf{x}) = 0, \qquad P(\mathbf{z}) = \mathbf{z} \quad \forall \mathbf{z} \in \partial H_a \cup \{\mathbf{x}_o\}.$$

Again, from the method of characteristics, P is extended constant to the ray direction from the horizon. See Fig. 17. The solution P is a piecewise continuous function that is continuous around the terminator. When we move the points $\mathbf{x}$ near terminator, we also move move the points $P(\mathbf{x})$, which are points near the horizon. By continuity, we will have the right motion with the terminator.

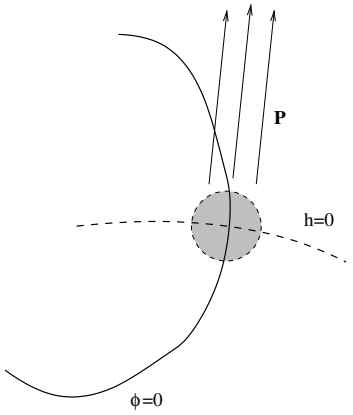

Fig. 17. Propagating P.

3.6. Reinitialization and emergence-time estimate

It can be easily seen from figure 18 that a completely hidden object may suddenly become visible at a later time during the journey of the vantage point. At the time of emergence, we need to reinitialize our algorithm, i.e., we need to find the visible contours on the newly emerged surfaces to get the correct visibility information. This was discussed in [12] in the usual setting of computation geometry, in which objects of consideration consist of polygons.

Assuming that we are merely tracking the visibility boundaries on the objects. How do we know when to initialize? We can formalize the reasoning as follows. We define the map $\mathscr{G}^{-1} : S^{d-1} \mapsto \{\mathbf{x}_i\}_{i=0}^{N(\theta)}$ such that

$$\mathscr{G}^{-1}(\theta) := \{\mathbf{x}_i \in \mathbb{R}^d : \nabla\phi(\mathbf{x}_i)/|\nabla\phi(\mathbf{x}_i)| = \theta \quad \text{and} \quad \phi(\mathbf{x}_i) = 0\}.$$

This map is the inverse of the Gauss map in the case that $\{\phi = 0\}$ is a strictly convex hypersurface. Let $\mathscr{S}$ be the set containing x and x^*. We reinitialize whenever there exists an $x \in S$ such that $\exists y \in \mathscr{G}(r(\mathbf{y}))$ with $\mathbf{x}^*(\mathbf{x}) \prec y \prec \mathbf{x}$. This provides an explicit criterion for reinitialization, but it is certainly not a trivial task. In our implicit formulation, we have enough information about the spatial structure of the occluders that we are able to derive an estimate on the time of emergence of an hidden object by using the knowledge of: (1) how the shadow moves; (2) how far a hidden surface is from the shadow boundaries.

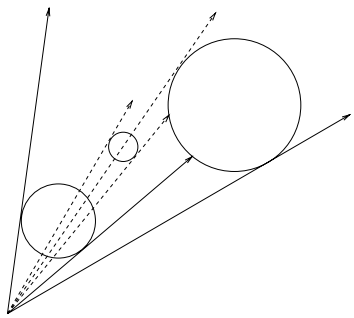

Fig. 18. Model scenario I.

Given the current vantage point position and its motion, we want to estimate the emergence time for an object that is occluded. We begin by assuming the the curvatures of the surfaces locally around the regions of interest are constant. The diagram in Fig. 19 shows a model configuration: The small circle is initially occluded by the larger circle on the left. We want to estimate the time interval δt between this instance t_0 and the time $t_1 = t_0 + \delta t$ when the small circle first emerges into the scene.

Following the discussion above, consider a point $\mathbf{y}$ on the shadow boundaries away from the horizon such that $\nabla\phi(\mathbf{y}) \perp r(\mathbf{y})$ and $\nabla\phi(\mathbf{y}) \cdot \nabla\phi(\mathbf{y}(\mathbf{x})) > 0$. This is the point closest to some hidden part of the objects.

For consistency of notation, we will use D in place of $\mathbf{y}$. Let d_E be the distance from E to the shadow boundary containing C and D and the circle centered at O'. Let ρ and ρ' be the radius of the circle centered at O and O', respectively. Let d denote the distance between D and C'. We have the following identities:

$$CC' = \rho \tan\frac{\delta\theta}{2}, \quad CD = d - CC' = d - \rho \tan\frac{\delta\theta}{2} \Rightarrow DE = CD \sin\delta\theta,$$

$$O'A' = \frac{1}{\cos\delta\theta}\rho', \quad DE = \frac{d_E}{\cos\delta\theta}.$$

Therefore, we can find $\delta\theta$ from the last two equalities. Since we know how fast the horizon is moving, we can then determine δt. Let $\dot{\mathbf{x}} = |\dot{\mathbf{x}}|_\infty$,

$$\frac{DE}{\delta t} = |\dot{\mathbf{x}}|_\infty, \quad \delta t = \frac{DE}{|\dot{\mathbf{x}}|_\infty}.$$

3.6.1. Further considerations — spatial and temporal scales

We have mentioned in the beginning of this paper that the approach of moving the visible contour may not be more efficient than simply performing implicit ray casting in general "large world" configurations. Here we construct such a case to validate our arguments.

Consider a non-convex part of an object as shown by the dashed red curve in Fig. 20. Suppose the dashed curve is broken down into dense small disconnected components. In the case where the red curve is the non-convex part of a connected component and with the viewing direction being depicted in Fig. 20, the terminator will move continuously on the non-convex part of the object without need for reinitialization.

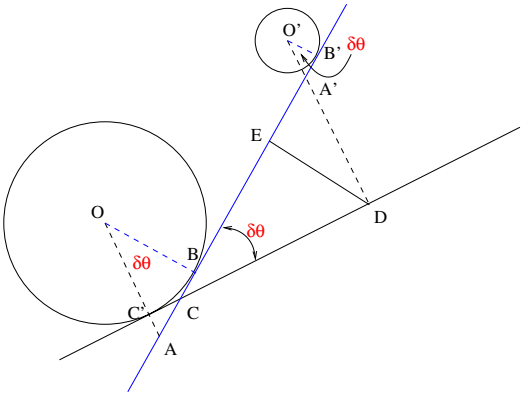

Fig. 19. A diagram for emergence time estimate.

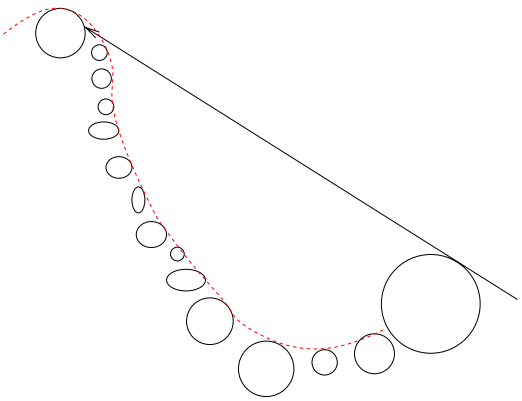

Fig. 20. Bad case for moving curves.

In essense, this is really an issue of spatial and temporal scales. If the time scale of interest is significantly larger than the distance between these objects, i.e. the vantage point and the occluding surfaces are moving relatively fast when compared to the size of the occluders, then frequent reinitialization is inevitable, and the dynamic approach may be inpractical. In this case, we can reconsider the strategy mentioned earlier in Section 2.5 of merging these small pieces together, and consider the dynamics in the new "homogenized" setting.

4. Conclusion and future directions

In this article, we introduced an implicit ray casting algorithm that is parallelizable. This was then extended to a multi-resolution algorithm for near optimal efficiency. Our one-pass, one level algorithm can be implemented on other grid geometry by either considering the discrete Galerkin Method, or by implementing the interpolation procedure defined in Appendix A. We showed that the implicit framework captures accurately the shadow boundaries, which include the horizon and terminator curves. We studied how these objects move when the source point is moving. Explicit formulae which reveal the relations between the motions and the local/global geometry of the given configuration are derived and are tightly coupled with our level set framework for implementation. Questions such as "how soon will this hidden object appear" in many situations can be answered as a result of our algorithm.

There is a rich pool of applications related to the visibility problem described in this paper. Currently, we are working on problems related to navigation, visibility with occluders changing shapes in time, in non-uniform media. Our solutions will combine approaches both from the PDE formulation and the algorithms in computational geometry.

Appendix A

A.1. Interpolation schemes

Since the majority of visibility applications benefit from the simplicity of Cartesian grids, we need to adapt the algorithm in order to take advantage of this. As described in the algorithm, at each grid point $x = (i, j, k)$, we need to determine an upwind neighbor x' and find the value of $\psi(x')$. In most cases, x' does

not lie on the grid. Therefore, we need to interpolate the values of ψ from the grid points closest to x'. For simplicity and speed considerations, we choose to perform linear interpolation in 2D and bilinear interpolation in 3D. In Fig. 2, we use $\psi(P_1)$ and $\psi(P_2)$ for linear interpolation in the 2D case and use $\psi(P_i)$, $i = 1, 2, 3, 4$, for bilinear interpolation of ψ.

We note that a fast marching or fast sweeping strategy for determining distance from the source point and passing values can be used in place of this interpolation.

Let ψ_{int} be the interpolant near x', we know that $\psi_{\text{int}}(x') = \psi(x') + \mathcal{O}(h^2)$. Thus, the discrete visibility equation (2.3) is in effect

$$\psi(x) = \min(\psi_{\text{int}}(x'), \phi(x)).$$

A.2. Examples of star-shaped updating sequence (sweeping)

There are many different ways of implementing a star-shaped updating sequence. One approach is to use the algorithm based on the heap sort strategy [31] to find grid nodes for update based on their distance to the vantage point. However, due the complexity of performing heap sort, this algorithm is not optimal.

Alternatively, we use a sweeping approach in our simulation. For example, let us consider a Cartesian grid in 2D and assume that the vantage point lies on a grid node; we can then consider separately the visibility problem in each of the four quadrants centered at the vantage point. For simplicity, let us assume that the vantage point is at the origin and the grid is represented by the lattice $[-n_x, n_x] \times [-n_y, n_y] \subset \mathbb{Z}^2$. A compact way of writing this sweeping sequence in C/C++ is

```
for(sl=-1; sl<=1; sl+=2)
for(s2=-1; s2<=1; s2+=2)
for(i=0; (sl<0?i>=-nx: i<=nx); i+=sl)
for(j=0; (s2<0?j>=-ny: j<=ny); j+=s2)
  update ψ_{i,j}.
```

In the case where $\mathbf{x}_o$ does not lie on a grid node, we describe an easy modification to the updating sequence above. Let $\mathbf{x}_o \in I_o := [x_{i_0}, x_{i_0+1}) \times [y_{j_0}, y_{j_0+1})$. Update the values of ψ on the vertices of I_o. Then update the grid nodes in the strips $\{(x_i, y_j) : i = i_0, i_0 + 1 \text{ and } j = -n_y \text{ to } n_y\}$ and $\{(x_i, y_j) : i = -n_x, n_x \text{ and } j = j_0, j_{0+1}\}$. Finally, update the remaining four quadrants independently. See Fig. 21 for a depiction of this approach.

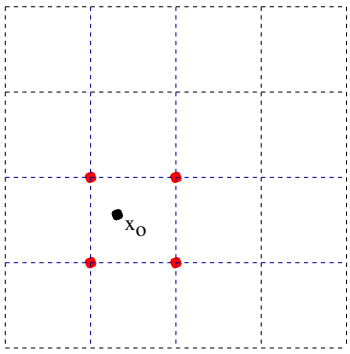

Fig. 21. The red points denote the cell vertices. (For interpretation of the reference to colour in this figure legend, the reader is referred to the web version of this article.)

Finally, we remark that the for loops presented above are meant to demonstrate one possible upwind update sequence for the construction of the solution. In real implementations, one should break it up for better efficiency.

A.3. Finding the curvature of a specified direction

As we argued in Section 3.2, the three-dimensional problem of determining the motion of the horizon can be reduced to an instantaneous two-dimensional problem. In order to move the horizon in this manner, we need to evaluate the curvature of the surface in the specified direction. Here we present a way to do that.

Let τ be the tangent vector being specified. We want to find the curvature on $\partial\Omega$ in this direction. First let $p(\mathbf{x};\tau)$ be the plane passing through $\mathbf{x}$, spanned by $\mathbf{n}(x)$ and τ, and let P be the level set function that embeds this plane. Then

$$\tilde{\tau} = \frac{\nabla\phi \times \nabla P}{|\nabla\phi \times \nabla P|},$$

where $\tilde{\tau}(\mathbf{x}) = \tau$, and the curvature is

$$k_\tau \mathbf{n} = \vec{\nabla}\tilde{\tau} \cdot \tilde{\tau}.$$

A.4. Derivation of the dynamics of the horizon

We follow the constraints (3.2):

$$\begin{cases} \phi(\mathbf{x}) = 0, \\ (\mathbf{x} - \mathbf{x}_0) \cdot \nabla\phi(\mathbf{x}) = 0 \end{cases}$$

and differentiate with respect to t, we have

$$\nabla\phi(\mathbf{x}) \cdot \dot{\mathbf{x}} = 0, \tag{A.1}$$

$$(\mathbf{x} - \mathbf{x}_o) \cdot D^2\phi(\mathbf{x})\dot{\mathbf{x}} = \dot{\mathbf{x}}_o \cdot \nabla\phi(\mathbf{x}). \tag{A.2}$$

In 2 space dimensions, these two relations uniquely determine the motion of x with given initial conditions. Writing $(\mathbf{x} - \mathbf{x}_o) = |\mathbf{x} - \mathbf{x}_o| n^\perp(\mathbf{x}) = |\mathbf{x} - \mathbf{x}_o| (-\phi_y, \phi_x)/|\nabla\phi|$, we have

$$\frac{|\mathbf{x} - \mathbf{x}_o|}{|\nabla\phi|} \begin{pmatrix} -\phi_y \\ \phi_x \end{pmatrix} \cdot \begin{pmatrix} \phi_{xx} & \phi_{xy} \\ \phi_{yx} & \phi_{yy} \end{pmatrix} \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \frac{|\mathbf{x} - \mathbf{x}_0|}{|\nabla\phi|} \begin{pmatrix} -\phi_y\phi_{xx} + \phi_x\phi_{yx} \\ -\phi_y\phi_{xy} + \phi_x\phi_{yy} \end{pmatrix} \cdot \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix},$$

and

$$\frac{|\mathbf{x} - \mathbf{x}_o|}{|\nabla\phi|} \begin{pmatrix} \phi_x & \phi_y \\ -\phi_y\phi_{xx} + \phi_x\phi_{yx} & -\phi_y\phi_{xy} + \phi_x\phi_{yy} \end{pmatrix} \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} 0 \\ \dot{\mathbf{x}}_o \cdot \nabla\phi(\mathbf{x}) \end{pmatrix}$$

$$\begin{aligned} \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} &= \frac{|\nabla\phi|}{|\mathbf{x} - \mathbf{x}_o|} \frac{1}{D} \begin{pmatrix} -\phi_y\phi_{xy} + \phi_x\phi_{yy} & -\phi_y \\ \phi_y\phi_{xx} - \phi_x\phi_{yx} & \phi_x \end{pmatrix} \begin{pmatrix} 0 \\ \dot{\mathbf{x}}_o \cdot \nabla\phi(\mathbf{x}) \end{pmatrix} \\ &= \frac{|\nabla\phi|}{|\mathbf{x} - \mathbf{x}_o|} \frac{\dot{\mathbf{x}}_o \cdot \nabla\phi(\mathbf{x})}{D} \begin{pmatrix} -\phi_y \\ \phi_x \end{pmatrix}, \end{aligned}$$

where

$$D = \det\begin{pmatrix} \phi_x & \phi_y \\ -\phi_y\phi_{xx} + \phi_x\phi_{yx} & -\phi_y\phi_{xy} + \phi_x\phi_{yy} \end{pmatrix} = -\phi_x\phi_y\phi_{xy} + \phi_x^2\phi_{yy} + \phi_y^2\phi_{xx} - \phi_x\phi_y\phi_{xy}$$
$$= \phi_x^2\phi_{yy} + \phi_y^2\phi_{xx} - 2\phi_x\phi_y\phi_{xy}.$$

Since the curvature of $\partial\Omega$ at x is

$$\kappa = \nabla \cdot \frac{\nabla\phi}{|\nabla\phi|} = \frac{1}{|\nabla\phi|^3}(\phi_x^2\phi_{yy} + \phi_y^2\phi_{xx} - 2\phi_x\phi_y\phi_{xy}),$$

the motion of x is

$$\dot{\mathbf{x}} = \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \frac{1}{\kappa}\frac{\dot{x}_o \cdot n(\mathbf{x})}{|\mathbf{x} - \mathbf{x}_o|} n^{\perp}(\mathbf{x}). \tag{A.3}$$

We define $n^{\perp}(\mathbf{x}) = (\mathbf{x} - \mathbf{x}_o)/|\mathbf{x} - \mathbf{x}_o|$.

Alternatively, we can write $\dot{x}$ in a slightly different form

$$\dot{x} = \frac{P_{\nabla\phi}\nabla^2\phi(x - x_0)}{|P_{\nabla\phi}\nabla^2\phi(x - x_0)|^2}(\dot{x}_0 \cdot \nabla\phi),$$

where P_v is the orthogonal projection matrix projecting vectors to the plane with normal vector v. Let us check this expression for the velocity of the curve. Note $\nabla\phi \cdot \dot{x} = 0$ since $v \cdot P_v w = 0$ for all vectors v and w. Also, $\nabla^2\phi(x - x_0) \cdot \dot{x} = \nabla\phi \cdot \dot{x}_0$ is satisfied since $P_v w \cdot w = |P_v w|^2$ for all vectors v and w. Thus this velocity is valid and is the first form in our alternate derivation.

Geometric interpretation. If x indeed represents the position of the curve, then $x - x_0$ is tangent to the object surface at x and so $x - x_0 = P_{\nabla\phi}(x - x_0)$. Making this replacement above gives our second form for the velocity

$$\dot{x} = \frac{P_{\nabla\phi}\nabla^2\phi P_{\nabla\phi}(x - x_0)}{|P_{\nabla\phi}\nabla^2\phi P_{\nabla\phi}(x - x_0)|^2}(\dot{x}_0 \cdot \nabla\phi).$$

This form is particularly nice because we know the second fundamental form in a level set framework is transformed to

$$II = \frac{1}{|\nabla\phi|} P_{\nabla\phi}\nabla^2\phi P_{\nabla\phi}.$$

Thus we can rewrite the velocity in its final form

$$\dot{x} = \frac{II(x - x_0)}{|II(x - x_0)|^2}\left(\dot{x}_0 \cdot \frac{\nabla\phi}{|\nabla\phi|}\right).$$

IIv evaluated at a point represents the change in the normals of the object surface in the direction of v at that point.

A.5. Derivation of the dynamics of the terminator

We assume that the level sets of ϕ near $\mathbf{x}$ are smooth curves and are not tangent to r. In two space dimension, we have two equations that determine the dynamics of $\mathbf{x}$:

$$\begin{cases} \phi(\mathbf{x}) = \text{constant}, \\ r = \tilde{r}. \end{cases} \tag{A.4}$$

Let $\mathbf{s}$ and $\tilde{\mathbf{s}}$ denote $(\mathbf{x} - \mathbf{x}_o)$ and $(\tilde{\mathbf{x}} - \mathbf{x}_o)$, respectively. Differentiating these equations, we arrive at

$$\nabla\phi(\mathbf{x}) \cdot \mathbf{x}' = 0,$$

$$\frac{\mathbf{s}'}{|\mathbf{s}|} - \frac{\mathbf{s}}{|\mathbf{s}|^2} r \cdot \mathbf{s}' = \frac{\tilde{\mathbf{s}}'}{|\tilde{\mathbf{s}}|} - \frac{\tilde{\mathbf{s}}}{|\tilde{\mathbf{s}}|^2} \tilde{r} \cdot \tilde{\mathbf{s}}'.$$

Notice that the term

$$r \cdot \mathbf{s}' \frac{\mathbf{s}}{|\mathbf{s}|} = (r \cdot \mathbf{s}')v$$

is $\mathbf{s}'$ projected onto the unit vector r. Therefore, the left-hand side denotes the projection of $\mathbf{s}'$ onto the unit vector $v^\perp$:

$$\frac{1}{|\mathbf{s}|}(\mathbf{s}' - r \cdot \mathbf{s}' v) = \frac{\mathbf{s}' \cdot r^\perp}{|\mathbf{s}|} = \frac{1}{|\mathbf{s}|} P_r \mathbf{s}'.$$

Similarly, with the right-hand side, we have the equation

$$\frac{1}{|\mathbf{s}|} P_r \mathbf{s}' = \frac{1}{|\tilde{\mathbf{s}}|} P_{\tilde{r}} \tilde{\mathbf{s}}'.$$

Keeping in mind that we want to solve for $\mathbf{x}'$, we move every other term to the right-hand side and arrive at

$$P_r \mathbf{x}' = \frac{|\mathbf{s}|}{|\tilde{\mathbf{s}}|} P_{\tilde{r}} \tilde{\mathbf{s}}' + P_r \mathbf{x}'_{\mathbf{o}},$$

$$\nabla\phi(\mathbf{x}) \cdot \mathbf{x}' = 0.$$

In two dimensions, $P_r w = (w \cdot r^\perp) r^\perp$, and $P_r w \cdot r^\perp = w \cdot r^\perp$, therefore, we have

$$\begin{pmatrix} r_2 & -r_1 \\ \phi_{x_1} & \phi_{x_2} \end{pmatrix} \begin{pmatrix} x'_1 \\ x'_2 \end{pmatrix} = \begin{pmatrix} \frac{|\mathbf{s}|}{|\tilde{\mathbf{s}}|} \tilde{\mathbf{s}}' \cdot r^\perp + \mathbf{x}'_o \cdot r^\perp \\ 0 \end{pmatrix},$$

and consequently,

$$\begin{pmatrix} x'_1 \\ x'_2 \end{pmatrix} = \frac{1}{r \cdot \nabla\phi(\mathbf{x})} \cdot \begin{pmatrix} \phi_{x_2} & r_1 \\ -\phi_{x_1} & r_2 \end{pmatrix} \begin{pmatrix} \frac{|\mathbf{s}|}{|\tilde{\mathbf{s}}|} \tilde{\mathbf{s}}' \cdot r^\perp + \mathbf{x}'_o \cdot r^\perp \\ 0 \end{pmatrix} = \frac{1}{r \cdot \nabla\phi(\mathbf{x})} \left(\frac{|\mathbf{s}|}{|\tilde{\mathbf{s}}|} \tilde{\mathbf{s}}' \cdot r^\perp + \mathbf{x}'_o \cdot r^\perp\right) \begin{pmatrix} \phi_{x_2} \\ -\phi_{x_1} \end{pmatrix}$$

$$= \frac{\mathbf{n}^\perp(\mathbf{x})}{v \cdot \mathbf{n}(\mathbf{x})} \left(\frac{|\mathbf{s}|}{|\tilde{\mathbf{s}}|} \tilde{\mathbf{s}}' + \mathbf{x}'_o\right) \cdot r^\perp = \frac{\left(\frac{|\mathbf{s}|}{|\tilde{\mathbf{s}}|} \tilde{\mathbf{s}}' + \mathbf{x}'_o\right) \cdot r^\perp}{r \cdot \mathbf{n}(\mathbf{x})} \mathbf{n}^\perp(\mathbf{x}),$$

where

$$\nabla^\perp\phi(\mathbf{x}) := \begin{pmatrix} \phi_{x_2} \\ -\phi_{x_1} \end{pmatrix}, \quad \text{and} \quad n^\perp := \frac{\nabla^\perp\phi(\mathbf{x})}{|\nabla^\perp\phi(\mathbf{x})|}.$$

A.6. Numerics

We computed the quantities describe in this paper using standard level set technology. Please refer to [20–22] for details.

A.7. A list of level set functions used in this paper

We provide a comprehensive list of the level set functions we construct in this paper:

ϕ: embeds the objects,

$\hbar(\mathbf{x}) := (\mathbf{x} - \mathbf{x}_o) \cdot \nabla\phi(\mathbf{x})$: characterizes the horizon,

$\tilde{\phi} := \max(\phi, -\hbar)$: $\{\tilde{\phi} \leqslant 0\} = \{\phi \leqslant 0\} \setminus \{\hbar < 0\}$, defines the same visibility as ϕ,

ψ: the visibility map resulting from the implicit ray casting on ϕ,

$\tilde{\psi}$: the visibility map resulting from the implicit ray casting on $\tilde{\phi}$, characterizes the terminator,

$P : \mathbb{R}^d \mapsto \mathbb{R}^d$: links horizon to its cast implicitly.

References

[1] D. Adalsteinsson, J.A. Sethian, An overview of level set methods for etching, deposition, and lithography development, IEEE Trans. Semiconductor Dev. 10 (1) (1997).

[2] P.K. Agarwal, M. Sharir, Ray shooting amidst convex polygons in 2D, J. Algorithms 21 (3) (1996) 508–519.

[3] P.K. Agarwal, M. Sharir, Ray shooting amidst convex polyhedra and polyhedral terrains in three dimensions, SIAM J. Comput. 25 (1) (1996) 100–116.

[4] L. Alvarez, F. Guichard, P.-L. Lions, J.-M. Morel, Axioms and fundamental equations of image processing, Arch. Rational Mech. Anal. 123 (3) (1993) 199–257.

[5] M.D. Betterton, Theory of structure formation in snowfields motivated by penitentes, suncups, and dirt cones, Phys. Rev. E 63 (2001).

[6] P. Burchard, L.-T. Cheng, B. Merriman, S. Osher, Motion of curves in three spatial dimensions using a level set approach, J. Comput. Phys. 170 (2001) 720–741.

[7] E. Catmull, A hidden-surface algorithm with anti-aliasing, in: Computer Graphics (SIGGRAPH'78 Proceedings), vol. 12(3), August, 1978, pp. 6–11.

[8] L.-T. Cheng, P. Burchard, B. Merriman, S. Osher, Motion of curves constrained on surfaces using a level set approach, J. Comput. Phys. 175 (2002) 604–644.

[9] L.T. Cheng, R. Tsai, A level set framework for visibility related variational problems, UCLA CAM Report 04-03.

[10] R. Cipolla, P. Giblin, Visual Motion of Curves and Surfaces, Cambridge University Press, Cambridge, 2000.

[11] S. Coorg, S. Teller, Real-time occlusion culling for models with large occluders, in: ACM Symposium on Interactive 3D Graphics, 1997.

[12] S. Coorg, S. Teller, Temporally coherent conservative visibility, Comput. Geom. 12 (1–2) (1999)105–124; 12th ACM Symposium on Computational Geometry, Philadelphia, PA, 1996.

[13] O. Devillers, V. Dujmovic, H. Everett, X. Goaoc, S. Lazard, H.-S. Na, S. Petitjean, The expected number of 3d visibility events is linear, SIAM J. Comput. (2003).

[14] F. Durand, 3d visibility: analysis study and applications, Ph.D. Thesis, Univeristy J. Fourier, Grenoble, France, 1999.

[15] F. Durand, G. Drettakis, J. Thollot, C. Puech, Conservative visibility preprocessing using extended projections, in: SIGGRAPH, 2000.

[16] A. Hertzmann, D. Zorin, Illustrating smooth surfaces, in: SIGGRAPH, 2001.

[17] H. Jin, A. Yezzi, Y.-H. Tsai, L.T. Cheng, S. Soatto, Estimation of 3d surface shape and smooth radiance from 2d images: a level set approachz, J. Scientific Comput. 19 (1–3) (1998) 267–292.

[18] J.J. Koenderink, Solid Shape, MIT Press Series in Artificial Intelligence, MIT Press, Cambridge, MA, 1990.

[19] S. Osher, L.-T. Cheng, M. Kang, H. Shim, Y.-H. Tsai, Geometric optics in a phase-space-based level set and Eulerian framework, J. Comput. Phys. 179 (2) (2002) 622–648.

[20] S. Osher, R. Fedkiw, Level Set Methods and Dynamic Implicit Surfaces, Springer, New York, 2002.
[21] S. Osher, J.A. Sethian, Fronts propagating with curvature-dependent speed: algorithms based on Hamilton–Jacobi formulations, J. Comput. Phys. 79 (1) (1988) 12–49.
[22] S. Osher, C.-W. Shu, High-order essentially nonoscillatory schemes for Hamilton–Jacobi equations, SIAM J. Numer. Anal. 28 (4) (1991) 907–922.
[23] S. Petitjean, A computational geometric approach to visual hulls, Int. J. Comput. Geom. Appl. 8 (4) (1998) 407–436.
[24] G. Sapiro, R. Kimmel, D. Shaked, B.B. Kimia, A.M. Bruckstein, Implementing continuous scale morphology via curve evolution, Pattern Recognition 26 (9) (1993) 1363–1372.
[25] G. Schaufler, J. Dorsey, X. Decoret, F.X. Sillion, Conservative volumetric visibility with occluder fusion, in: SIGGRAPH, 2000.
[26] J.A. Sethian, Level Set Methods and Fast Marching Methods, second ed., Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science, Cambridge University Press, Cambridge, 1999.
[27] F. Sillion, G. Drettakis, Feature-based control of visibility error: a multi-resolution clustering algorithm for global illumination, in: R. Cook (Ed.), SIGGRAPH 95 Conference Proceedings, Annual Conference Series, ACM SIGGRAPH, Los Angeles, CA, vol. 29, Addison-Wesley, Reading, MA, 1995, pp. 145–152.
[28] Y.-H.R. Tsai, Rapid and accurate computation of the distance function using grids, J. Comput. Phys. 178 (1) (2002) 175–195.
[29] Y.-H.R. Tsai, L.-T. Cheng, P. Burchard, S. Osher, G. Dynamic visibility in an implicit framework, UCLA CAM Report, 02(06), 2002.
[30] Y.-H.R. Tsai, L.-T. Cheng, S. Osher, H.-K. Zhao, Fast sweeping methods for a class of Hamilton–Jacobi equations, SIAM J. Numer. Anal. 41 (2) (2003) 673–694.
[31] J. Tsitsiklis, Efficient algorithms for globally optimal trajectories, IEEE Trans. Autom. Contr. 40 (9) (1995) 1528–1538.