Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational Physics 220 (2007) 612–625
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

Face offsetting: A unified approach for explicit moving interfaces

Xiangmin Jiao *

College of Computing, Georgia Institute of Technology, 801 Atlantic Drive, Atlanta, GA 30338, USA

Received 1 November 2005; received in revised form 12 May 2006; accepted 21 May 2006
Available online 7 July 2006

Abstract

Dynamic moving interfaces are central to many scientific, engineering, and graphics applications. In this paper, we introduce a novel method for moving surface meshes, called the *face offsetting method*, based on a generalized Huygens' principle. Our method operates directly on a Lagrangian surface mesh, without requiring an Eulerian volume mesh. Unlike traditional Lagrangian methods, which move each vertex directly along an approximate normal or user-specified direction, our method propagates faces and then reconstructs vertices through an eigenvalue analysis locally at each vertex to resolve normal and tangential motion of the interface simultaneously. The method also includes techniques for ensuring the integrity of the surface as it evolves. Face offsetting provides a unified framework for various dynamic interface problems and delivers accurate physical solutions even in the presence of singularities and large curvatures. We present the theoretical foundation of our method, and also demonstrate its accuracy, efficiency, and flexibility for a number of benchmark problems and a real-world application.
© 2006 Elsevier Inc. All rights reserved.

Keywords: Interface propagation; Moving meshes; Entropy condition; Huygens' principle; Face offsetting

1. Introduction

Dynamic moving surfaces arise in many scientific and engineering applications, such as crystal growth, dendritic solidification, microchip fabrication, multiphase flows, combustion, and biomedical applications. Computer simulations of these applications require numerical techniques to determine the position of a surface (or interface) as it evolves under a speed or velocity field governed by some physical laws. This problem is commonly referred to as *interface propagation*. Because singularities and topological changes may develop during the evolution of the surface, this problem poses daunting challenges in developing accurate, efficient, and robust techniques, and has attracted significant interests in the past two decades [1–11].

In general, there are two broad categories of techniques for moving interfaces: *Lagrangian methods*, which *track* the interface using an *explicit* surface representation, and *Eulerian methods*, which *capture* the interface using an *implicit* representation. In recent years, Eulerian methods (such as level set methods [1–3], volume-

* Tel.: +1 404 385 0596; fax: +1 404 385 7337.
E-mail address: jiao@cc.gatech.edu.

0021-9991/$ - see front matter © 2006 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2006.05.021

of-fluid methods [12–15], and phase-field methods [16]) have made significant advancements and become the dominant methods for moving interfaces for their simplicity and robustness, although their accuracy may be low in the presence of singularities and topological changes. Lagrangian methods, such as marker particle [17] and front tracking methods [4–6], can potentially provide higher accuracy at lower cost and frequently suit better than implicit representations for applications such as multi-component simulations. Unfortunately, existing Lagrangian methods often suffer from instabilities in the form of growing oscillations or self-intersections [3] and sometimes must resort to *ad hoc* techniques to trim off self-intersections [18,19]. Due to this mixed success of pure Lagrangian or Eulerian methods, in an attempt to seek a better compromise between the accuracy of Lagrangian methods and robustness of Eulerian methods, a number of *hybrid methods* have appeared in recent years [7–10]. These methods unfortunately are more complex to implement than traditional methods and still suffer from smearing of singularities (although milder than pure Eulerian methods).

In this paper, we introduce a novel method for moving interfaces, called the *face-offsetting method* (FOM). As a Lagrangian method, FOM propagates an explicit surface mesh without requiring a volume mesh. Unlike traditional Lagrangian methods, which moves each vertex passively using some standard numerical integration techniques, FOM propagates the faces of the mesh and then reconstructs the vertices from the propagated faces by performing an eigenvalue analysis locally at each vertex to resolve the normal motion (for obtaining surface geometry) and tangential motion (for maintaining mesh quality) simultaneously. This new approach is motivated by a new formulation of interface propagation, referred to as the *generalized Huygens' principle*, which extends the well-known *entropy-satisfying Huygens' principle* behind the level set methods [3]. This new principle explores some underlying connections between traditional Lagrangian and level set formulations of moving interfaces, unifies the view for *advective* motion (such as rigid-body motion and transport of fluids) and *wavefrontal* motion (such as burning, erosion, and deposition), and in turn provides us a new foundation for Lagrangian methods to obtain accurate physical solutions for both types of motions even at singularities, and to ensure the integrity of the surface as it evolves.

The remainder of this paper is organized as follows. Section 2 revisits some fundamental concepts behind interface propagation and introduces the generalized Huygens' principle. Section 3 introduces the face offsetting method for advective motion, and Section 4 generalizes it to wavefrontal motion. Section 5 reports some experimental results using benchmark problems and a real-world application.

2. Unified view of moving interfaces

2.1. Moving interfaces

Dynamic surfaces appear in different forms in a wide range of applications. In this paper, we will focus on *moving interfaces*, where an interface refers to an *orientable* surface Γ^0 (or Γ), separating its *inside* Γ^+ from its *outside* Γ^-. The problem that we address in this paper, referred to as *interface propagation*, is to determining the position of a moving interface at a specific time, given its position and either a *velocity field* $\dot{\boldsymbol{u}}(\boldsymbol{x},t):\Gamma\times\mathbb{R}\to\mathbb{R}^3$ or a *normal speed* $f(\boldsymbol{x},t):\Gamma\times\mathbb{R}\to\mathbb{R}$. In the traditional Lagrangian formulation, the differential equation for propagating an interface under a given velocity is

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t}=\dot{\boldsymbol{u}}(\boldsymbol{x},t), \tag{1}$$

and under a given normal speed is

$$\frac{\partial\boldsymbol{x}}{\partial t}=f\boldsymbol{n}(\boldsymbol{x},t), \tag{2}$$

where $\boldsymbol{n}$ is the unit surface normal $\partial\boldsymbol{x}/\partial u\times\partial\boldsymbol{x}/\partial v/\|\partial\boldsymbol{x}/\partial u\times\partial\boldsymbol{x}/\partial v\|$ given a 2-D unit-length orthogonal parameterization $\boldsymbol{x}(u,v):\mathbb{R}^2\to\mathbb{R}^3$. In principle, a given velocity field can be converted into a normal speed and vice versa, as a tangential motion does not change the shape of the interface. However, propagation under a normal speed is in general more difficult than under a velocity field because the normal direction must be inferred from the geometry of the moving interface.

If the velocity field or normal speed is continuous and the moving interface remains smooth during its evolution, then interface propagation involves only a simple numerical solution to the differential equations. Unfortunately, when singularities occur as the interface evolves, a moving interface may exhibit different behaviors near singularities depending on the application. As illustrated in Fig. 1, when a 2-D interface expands at a sharp corner under normal motion, an ambiguity arises in how the singularity would look like afterward: some motions (such as ridge-body motion, bending, transport of fluids, etc.) tend to preserve the corner, whereas others (such as burning, etching, and deposition) tend to round the corner. We refer to these different types of motions as *advection-like* (or simply *advective*) and *wavefront-like* (or simply *wavefrontal*), respectively. The motivations behind these terms will become clear shortly.

To develop a numerical framework for propagating an interface under either advective or wavefrontal motion, we must first have a unified view of moving interfaces, which we refer to as the *generalized Huygens' principle*, based on an extension of the well-known *entropy-satisfying Huygens' principle* [3].

2.2. Entropy-satisfying Huygens' principle

Huygens' principle is a mathematical concept in wave optics [20] and is closely related to many moving-interface problems. This principle states that every point on a wavefront is a source of secondary waves that expand in all directions, and the new wavefront at a subsequent time is the *envelope* (the tangent surface) of the secondary waves. If the media are homogeneous and isotropic, then the envelope of a secondary wave from a point source is a sphere, and the new wavefront is the envelope of the spheres that are centered at the points on the current wavefront, where the radii are proportional to the wave speed. Huygens' principle can be used to interpret some physical phenomena intuitively. For instance, in combustion the particles on the burning front radiate signals (such as heat) in all directions, and the unburnt particles that receive signals the latest form the new front.

Under uniform speed, the connection between Huygens' principle and moving interfaces is well known for wavefront-like motion (see e.g. [3]). Note that even under *nonuniform* speed the Lagrangian and level set formulations for moving interfaces are also differential forms of the envelope of spheres, except that each point moves normal to the envelope instead of normal to the interface, where the difference in the normals is second order in time. However, since Huygens' principle was formulated originally for wave propagation rather than interface propagation, its differential forms and direct numerical approximations (such as traditional Lagrangian methods) can lead to self-intersections.

The difference between wave propagation and interface propagation lies in the so-called *entropy condition* [3]. In the context of Huygens' principle, this condition means that if a particle on the interface is under the influence of another source point, then this particle will not be a source of the new interface. This modified notion of Huygens' principle is commonly referred to as the *entropy-satisfying Huygens' principle* in the level set methods and is a key concept behind some of their numerical discretizations [3].

Geometrically, the difference between the constructions with and without the entropy condition is essentially the difference between the *envelope of balls* (E_b) and the *envelope of spheres* (E_s), where E_b is the (outer) *boundary* of the union of the (solid) balls, and E_s is the tangent surface of the (hollow) spheres, as illustrated in Fig. 2. By definition, E_b is free of self-intersections but E_s is not. For smooth surfaces $E_b \subseteq E_s$, and their dif-

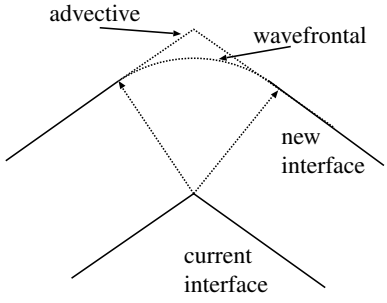


Fig. 1. Different local behaviors of advective and wavefrontal propagation.

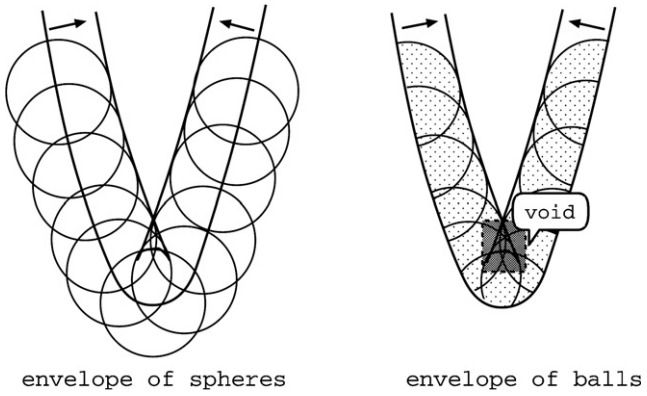


Fig. 2. Two-dimensional illustration of difference between envelopes of spheres and balls.

ference is composed of the "void" points that are on E_s but fall in the interior of a ball centered on the interface. Therefore, this envelope-of-balls notion also reduces to the traditional Lagrangian and level-set formulations if there are no void points, but it is a more complete description than the differential equations. This principle is still valid even if the interface is not differentiable, so the solution is well-defined even at singularities.

2.3. Generalized Huygens' principle

The entropy-satisfying Huygens' principle is an accurate description for wavefront-like motion, such as burning, etching, and deposition, but is only a rough approximation for advection, such as rigid-body motion, bending, and transport of fluids. If the surface is not smooth, then the solution based on this principle may lead to inaccurate and even incorrect solutions for advective motion.

To develop a general solution for advective and wavefrontal propagation, we introduce a set of generic terminology: given an interface Γ, the *field of influence* (FOI) of a point $\boldsymbol{p}$ on Γ is the set of points in its path within a given time frame, where the shape of the field may be problem dependent. The *shell of influence* (SOI) of p is the boundary of its FOI, and the SOI of Γ is composed of the boundary of the field influenced by all the points on Γ. The problem of interface propagation, therefore, is to construct the SOI of Γ given the FOI of all the points. We refer to this formulation as the *generalized Huygens'* or the *shell-of-influence* principle for moving interfaces. In this paper, we consider the cases where the FOI of a point is either a ball (the region reachable by the signals from a point in wavefrontal propagation) or a curve (the trajectory of a point in advective propagation).

The above description is fairly general but may have difficulties if the speed or velocity of the interface vary in sign with respect to the outward normal of the interface (e.g., if the interface moves inwards in some areas but outwards in other areas). In such a situation, the FOI of each point inherits the sign of its speed, and the SOI of the interface is the sum of the positive and negative SOIs. A point on both positive and negative SOIs cancels out and will not be on the SOI of the interface. This signed variation of the principle will be useful in later discussions.

2.4. Moving surface meshes

In many numerical simulations, an interface is discretized by a *surface mesh*, and interface propagation must reposition this mesh as the surface evolves. A surface mesh is a collection of topological entities of 0, 1, or 2 dimensions along with their *incidence* and *adjacency* relationships. We refer to the zero-dimensional entities as *vertices*, the one-dimensional entities *edges*, and the two-dimensional entities *faces*. We assume the mesh is *conformal*, meaning that two adjacent faces intersect at an edge or vertex, and two adjacent edges intersect at a vertex. For ease of presentation, we will focus on closed triangular surface meshes (or more precisely, 2-D simplicial manifolds without boundary), but the method that we will present easily generalizes, and

has been applied, to surfaces with boundary discretized using quadrilateral or mixed meshes. We will also limit our attention to the cases where the topology of the surface and the connectivity of the mesh do not change during the evolution.

3. Face offsetting for advection-like motion

The generalized Huygens' principle suggests a natural approach to propagate a surface mesh: determine the FOI of the interface by propagating each face and then approximate the SOI by reconstructing the vertices from the propagated faces. We refer to this idea as the *face offsetting* method (or FOM). We will first present the method for advection and then generalize it to wavefrontal motion in the next section.

3.1. Propagate faces

The first step of face offsetting is to propagate each face to determine its FOI. Depending on where the given speed or velocity is available, we can compute the new position of any point on the face using a standard time-integration scheme, such as one of the following:

- if the velocity is available on and near the interface, then use the fourth-order Runge–Kutta scheme (see e.g. [21]);
- if the velocity is available only on the interface, then use the first-order Euler scheme;
- if the normal speed is available on each face, then use the first-order Euler scheme to integrate along the face normal.

We refer to the new position of all the points of a face as its *face offset*, which is parallel to the face when the speed or velocity is uniform but may not be parallel when nonuniform. Given a small time step, the FOI of a face is bounded between the face and its offset. For a piecewise-linear approximation of the interface, the offset of a face is flat, and we need to obtain the plane containing the offset. For uniform speed or velocity, it suffices to integrate the motion only at the centroid of the face. For nonuniform speed or velocity, we integrate the motion at the vertices or quadrature points within each face to determine the plane.

3.2. Reconstruct vertices

After obtaining the face offsets, we then determine the new position of each vertex from these offsets. This step is critical and nontrivial especially for propagation under normal speed, because a direct numerical integration of the motion along an approximate normal direction at vertices can easily lead to wrong results at singularities or severely distort the mesh [3]. We present a technique to obtain the new surface and maintain the mesh quality simultaneously by computing both normal and tangential displacements.

3.2.1. Formulation

To reconstruct a vertex v in an advective motion, observe that its FOI (the trajectory of v under advection) is contained in the FOIs of its incident faces, so its new position must be contained in the SOI of its incident faces given that there is no topological change. This motivates us to move v to the intersection of the planes that contain the offsets of the faces incident on v. For clarity, we perform a coordinate transformation locally so that v becomes the origin. Each plane passing through a point $\boldsymbol{p}$ with unit normal $\boldsymbol{n}$ can be expressed by a linear equation, $\boldsymbol{n}^{\mathrm{T}}\boldsymbol{x} = \delta$, where $\delta = \boldsymbol{n}^{\mathrm{T}}\boldsymbol{p}$. Suppose v has m incident faces. The intersection of the face offsets is then the solution of an $m \times 3$ linear system

$$\boldsymbol{N}\boldsymbol{x} = \boldsymbol{a}, \tag{3}$$

of which each row corresponds to an incident face of v.

Eq. (3) may be under- or overdetermined depending on m. To address this difficulty, we generalize the notion of the intersection of a set of planes in a least squares sense and define it to be the point in $\mathbb{R}^3$ that

minimizes a weighted sum of squared distances to the planes. This least-squares intersection of face offsets is then the solution to a 3×3 linear system

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \tag{4}$$

where $\boldsymbol{A} = \boldsymbol{N}^{\mathrm{T}}\boldsymbol{W}\boldsymbol{N}$, $\boldsymbol{b} = \boldsymbol{N}^{\mathrm{T}}\boldsymbol{W}\boldsymbol{a}$, and $\boldsymbol{W}$ is an $m \times m$ diagonal matrix with W_{ii} equal to the weight (the face area, in specific) of the ith incident face on v. We refer to the solution to (4) as the *offset intersection*. This formulation shares some similarity with the quadric error metric in [22]. The matrix $\boldsymbol{A}$ may be nearly rank deficient (i.e., its columns may be nearly linearly dependent), and an eigenvalue analysis will turn out to be useful.

3.2.2. Eigenvalue analysis

The matrix $\boldsymbol{A}$ in (4) is symmetric and positive semi-definite, so it has an eigenvalue decomposition $\boldsymbol{A} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{\mathrm{T}}$, where its eigenvalues $\lambda_i = \Lambda_{ii}$ are all real and nonnegative, and their corresponding eigenvectors $\boldsymbol{e}_i$ are the column vectors of $\boldsymbol{V}$. Assume $\lambda_1 \geqslant \lambda_2 \geqslant \lambda_3$. We refer to the vector space spanned by the eigenvectors corresponding to relatively large eigenvalues of $\boldsymbol{A}$ as its *primary space* and the complementary space as its *null space*. From the definition $\boldsymbol{A} = \boldsymbol{N}^{\mathrm{T}}\boldsymbol{W}\boldsymbol{N}$, we have the singular value decomposition

$$\sqrt{\boldsymbol{W}}\boldsymbol{N} = \boldsymbol{U}\sqrt{\boldsymbol{\Lambda}}\boldsymbol{V}^{\mathrm{T}}, \tag{5}$$

where $\boldsymbol{U}$ is an $m \times 3$ matrix, and $\sqrt{\boldsymbol{W}}$ denotes the diagonal matrix whose ith diagonal entry is the square root of W_{ii} (and similarly for $\sqrt{\boldsymbol{\Lambda}}$).

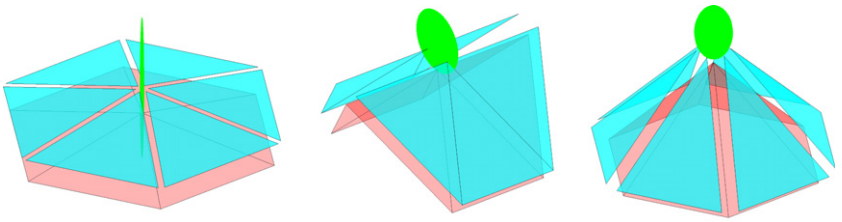
The above eigenvalues and eigenvectors have geometric meanings, as illustrated in Fig. 3. In the figure, the upper triangles depict the face offsets of the lower triangles. The ellipsoids are centered at the offset intersections, and their axes are aligned along the eigenvectors, with semi-axes proportional to their corresponding eigenvalues. From the figure, we can observe the following properties:

- If the offsets are (nearly) parallel to a plane γ (cf. Fig. 3(a)), then $\boldsymbol{A}$ has one large eigenvalue, and its primary space is orthogonal to γ, whereas the null space is parallel to γ.
- If the offset surface is (nearly) parallel to two distinct planes with a dihedral angle $\gg 0$ and $\ll \pi$ (cf. Fig. 3(b)), then $\boldsymbol{A}$ has two large eigenvalues, whose corresponding eigenvectors bisect the two planes (except at nearly right dihedral angles), and its null space is parallel to the ridge.
- If the offset surface forms a sharp corner (cf. Fig. 3(c)), the null space is empty.

Therefore, we restrict the displacement $\boldsymbol{d}$ within the primary space, i.e.,

$$\boldsymbol{d} = \sum_{i=1}^{k} \boldsymbol{e}_i^{\mathrm{T}}\boldsymbol{b}\boldsymbol{e}_i/\lambda_i, \tag{6}$$

where k is the dimension of the primary space. The vector $\boldsymbol{d}$ is a regularized offset intersection, and we refer to its unit vector as the *offset direction*. Suppose we have a reliable way to classify ridges and corners, such as the method in [23]. Generally speaking, we choose k to be one, two, and three at a smooth, ridge, and corner vertex, respectively, but filter out the eigenvectors whose corresponding eigenvalues are smaller than $\varepsilon\lambda_1$ (e.g., $\varepsilon \approx 0.003$) to avoid instability due to division by too small numbers.



(a) Smooth. (b) Ridge. (c) Corner.

Fig. 3. Correlation of relative sizes of eigenvalues and local flatness of surface.

3.3. Redistribute vertices

Vertex redistribution is an efficient and often effective approach to maintain or improve mesh quality. Redistributing vertices on a surface mesh is subtle because small perturbations to the geometry may accumulate and lead to very large errors over many iterations. For example, the popular Laplacian smoothing, which moves every vertex toward a weighted average of its neighboring vertices [24], may severely shrink the domain and smear the sharp features of a surface. The geometric errors associated with vertex redistribution may be alleviated by projecting the vertices onto a high-order approximation of the surface after moving them [25]. Such an approach may be cumbersome for moving interfaces, so a common practice in interface propagation has been to avoid vertex redistribution but only adapt the mesh by adding or removing vertices or changing the connectivity [6,9], although vertex redistribution can be potentially much more efficient.

In FOM, during vertex reconstruction we compute a tangential motion at each vertex, which we denote by $\boldsymbol{t}$ and require to be orthogonal to $\boldsymbol{d}$. Each vertex is then moved by $\boldsymbol{d}+\boldsymbol{t}$. Leveraging the eigenvalue analysis in the preceding subsection, we compute $\boldsymbol{t}$ as a weighted average (e.g., weighted by angle, edge length, or unity) of the neighborhood of v projected onto the null space, i.e.,

$$\boldsymbol{t} = \boldsymbol{T}\boldsymbol{T}^{\mathrm{T}}\left(\sum_{i=1}^{m} w_i \boldsymbol{c}_i\right) \Bigg/ \left(\sum_{i=1}^{m} w_i\right), \tag{7}$$

where $\boldsymbol{T}$ is a $3\times(3-k)$ matrix whose column vectors are the bases of the null space of $\boldsymbol{A}$, $\boldsymbol{c}_i$ is the vector from v to the offset centroid of the ith face incident on v, and w_i is its associated weight. We refer to this procedure as *null-space smoothing*.

This simple smoothing scheme tends to preserve sharp features or large-curvature areas, as it moves ridge vertices along their tangent lines and does not alter corners. In addition, it introduces a very small volume error. To obtain an estimate of the volume error, consider moving one vertex by $\boldsymbol{t}$ for a static surfaces (i.e., $\boldsymbol{d}=0$). The volume change due to moving the vertex by $\boldsymbol{t}$ is $\delta V = \sum_{i=1}^{m} a_i \boldsymbol{n}_i^{\mathrm{T}}\boldsymbol{t}/3$, where a_i and $\boldsymbol{n}_i$ are the area and unit normal of the ith incident face of v, respectively. Let $\boldsymbol{s}^{\mathrm{T}} = \sum_{i=1}^{m}\sqrt{a_i}\boldsymbol{U}_i^{\mathrm{T}}$, where $\boldsymbol{U}_i^{\mathrm{T}}$ is the ith row vector of $\boldsymbol{U}$ in (5). Then

$$\begin{aligned} 3\delta V &= \|\boldsymbol{W}\boldsymbol{N}\boldsymbol{t}\|_1 && (8)\\ &= \|\sqrt{\boldsymbol{W}}\boldsymbol{U}\sqrt{\boldsymbol{\Lambda}}\boldsymbol{V}^{\mathrm{T}}\boldsymbol{t}\|_1 && (9)\\ &= \sum_{i=1}^{3}\sum_{j=1}^{m}\sqrt{a_j}U_{ji}\sqrt{\lambda_i}(\boldsymbol{t}^{\mathrm{T}}\boldsymbol{e}_i) && (10)\\ &= \sum_{i=1}^{3} s_i\sqrt{\lambda_i}(\boldsymbol{t}^{\mathrm{T}}\boldsymbol{e}_i). && (11) \end{aligned}$$

If the mesh is coarse, then s_i (and similarly $\boldsymbol{t}^{\mathrm{T}}\boldsymbol{e}_i$) are likely to have comparable sizes for different i, and null-space smoothing tends to introduce a relatively small error by filtering out the components corresponding to large λ_i. Furthermore, let h be a measure of average edge length. Following the analysis in [22], the smaller eigenvalues corresponding to the null space of $\boldsymbol{A}$ are proportional to h^4 with a coefficient grows monotonically with curvatures, whereas the larger eigenvalues are proportional to h^2. Since null-space smoothing filters out large components at ridges and corners, δV is $\mathrm{O}(h^4)$ even near singularities, compared to $\mathrm{O}(h^3)$ for Laplacian smoothing near singularities. When running mesh smoothing for many iterations, null-space smoothing would be even more advantageous as it stops changing the shape after the mesh reaches a steady state tangentially, but Laplacian smoothing may continue to shrink the volume until all points are coplanar.

3.4. Limit time steps

FOM simultaneously moves vertices within the primary space to update the geometry and within the null space to control mesh quality. If the motion in either space is too large, then the mesh may become invalid due

to mesh folding or self-intersection. We now present a technique to prevent self-intersections, assuming that the surface does not change topology during its evolution.

After solving the displacements for all vertices, we can construct another motion where every vertex v moves along a straight line from $\boldsymbol{p}_v$ with the direction of $\boldsymbol{u}_v = \boldsymbol{d}(v) + \boldsymbol{t}(v)$ within a time step Δt, such that v is at $\boldsymbol{p}_v + \beta\boldsymbol{u}_v$ at time $\beta\Delta t$, and every face remains flat during the time interval. Under this motion, if no point on the interface falls under the influence of another point of the same sign, then the interface is free of self-intersections. A violation of this condition is accompanied by a local reversal of the orientation at a face or along an edge. Therefore, it suffices to determine the maximum β that prevents such reversals.

Let $\boldsymbol{q}_v^{(\beta)}$ (or simply $\boldsymbol{q}_v$) denote the position of vertex v at time $\beta\Delta t$, and let $\boldsymbol{q}_{i,j}$ denote $\boldsymbol{q}_i - \boldsymbol{q}_j$ (and similarly for $\boldsymbol{p}_{i,j}$ and $\boldsymbol{u}_{i,j}$). The normal to a triangle $\boldsymbol{p}_1\boldsymbol{p}_2\boldsymbol{p}_3$ at time $\beta\Delta t$ is then

$$\boldsymbol{q}_{2,1} \times \boldsymbol{q}_{3,1} = \beta^2\boldsymbol{c}_2 + \beta\boldsymbol{c}_1 + \boldsymbol{c}_0, \tag{12}$$

where $\boldsymbol{c}_2 = \boldsymbol{u}_{2,1} \times \boldsymbol{u}_{3,1}$, $\boldsymbol{c}_1 = \boldsymbol{p}_{2,1} \times \boldsymbol{u}_{3,1} - \boldsymbol{p}_{3,1} \times \boldsymbol{u}_{2,1}$, and $\boldsymbol{c}_0 = \boldsymbol{p}_{2,1} \times \boldsymbol{p}_{3,1}$. The face normal is $\boldsymbol{c}_0$ when $\beta = 0$, and it is reversed at time $\beta\Delta t$, where β is the positive solution of the quadratic equation

$$\boldsymbol{c}_0^{\mathrm{T}}(\beta^2\boldsymbol{c}_2 + \beta\boldsymbol{c}_1 + \boldsymbol{c}_0) = 0, \tag{13}$$

which can be solved using the quadratic formula or its alternative form depending on the sign of $\boldsymbol{c}_0^{\mathrm{T}}\boldsymbol{c}_1$ to avoid cancellation errors (see e.g. [21]).

The time-step constraint along an edge is trickier as it involves the normals to two faces and their offsets. We can simplify the computation by requiring the normals of the offsets point toward the positive side of the bisector of the face normals. This is solved by simply replacing the first $\boldsymbol{c}_0$ in (13) with this bisector to obtain β within each of its incident faces and then taking the smaller β of the incident faces of the edge.

After obtaining β for all the faces and edges, then the maximum allowed time step is $\alpha\Delta t$, where α is the smaller value between 1 and the smallest β (or a fraction of the smallest β to tolerate roundoff errors). If $\alpha < 1$, then we scale both $\boldsymbol{d}$ and $\boldsymbol{t}$ by α at the vertices to obtain a self-intersection-free surface. The scaled displacement is identical to propagating the interface for a time step $\alpha\Delta t$ under the given speed with reduced mesh smoothing if the face offsets were integrated using the first-order Euler scheme, but a slightly perturbed solution when using the fourth-order Runge–Kutta scheme.

4. Generalization to wavefrontal propagation

Compared to advective motion, wavefrontal motion is more difficult because the FOI of each point under this motion is a ball, and the SOI of each face is a nonlinear patch. A normal speed is typically used in wavefrontal motion, further complicating the problem. In this section, we extend FOM to address this motion by replacing each face offset in (3) with the tangent plane of the SOI of the face.

Suppose we already know the offset direction $\boldsymbol{d}$ at each vertex v under a local coordinate frame with v as the origin. For each face σ incident on v, let $\widetilde{\sigma}$ denote its offset under advective motion, $\boldsymbol{n}$ the normal to $\widetilde{\sigma}$, and $\boldsymbol{s}$ the direction from $\boldsymbol{d}$ orthogonal to the opposite edge of v in $\widetilde{\sigma}$. Let γ be the tangent plane of the SOI of σ at its intersection with the line from v along the direction $\boldsymbol{d}$. If $\boldsymbol{d}^{\mathrm{T}}\boldsymbol{s} \geqslant 0$, then γ must contain $\widetilde{\sigma}$; if $\boldsymbol{d}^{\mathrm{T}}\boldsymbol{s} < 0$, then γ has the same distance to v as $\widetilde{\sigma}$ but its normal is $\boldsymbol{d}$ instead of $\boldsymbol{n}$. We say σ is *contracting* locally at v in the former (i.e., $\boldsymbol{d}^{\mathrm{T}}\boldsymbol{s} \geqslant 0$) and is *expanding* in the latter. Fig. 4 illustrates the two cases in 2-D.

A key question is how to compute the offset direction $\boldsymbol{d}$. If the interface is locally contracting at v along all directions, then the solution is identical to that of advection. However, if the interface is expanding along some directions, an iterative procedure may be necessary, as $\boldsymbol{d}$ depends on the tangent planes, which depend on $\boldsymbol{d}$ itself. We use the advective solution to obtain an initial guess for $\boldsymbol{d}$, and then run additional steps to correct $\boldsymbol{d}$ if necessary by solving (6) with new tangent planes.

The solution of (6) provides both the direction and the length. In practice, the initial guess of $\boldsymbol{d}$ is typically a good estimate in terms of the direction but may overshoot in length at expansion. Except for some pathological cases (such as at singular saddle points), it suffices to use this direction and correct only the length. The problem then reduces to a simple equation with only one unknown. In particular, the optimal length l in the least squares sense along direction $\boldsymbol{d}$ is the solution to

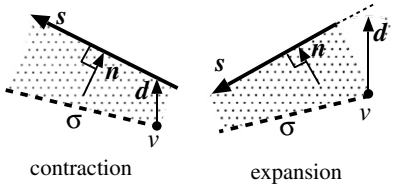


Fig. 4. Two-dimensional illustration of contraction and expansion.

1: propagate faces by integrating in time at vertices or quadrature points;

2: determine displacements $\boldsymbol{d}$ at vertices in primary space for advection;

3: correct $\boldsymbol{d}$ for wavefrontal motion;

4: compute tangential motion $\boldsymbol{t}$ within null space for vertex redistribution;

5: obtain maximum time step $\alpha\Delta t$;

6: return $\alpha\Delta t$ and $\alpha(\boldsymbol{d}+\boldsymbol{t})$;

Fig. 5. Time stepping of face-offsetting method for advective and wavefrontal motion.

$$(\boldsymbol{d}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{d})l = \boldsymbol{d}^{\mathrm{T}}\boldsymbol{b}\|\boldsymbol{d}\|_2, \tag{14}$$

where $\boldsymbol{A}$ and $\boldsymbol{b}$ are defined in (4) with the tangent planes of the SOIs of the faces. If $\boldsymbol{d} = 0$, then $l = 0$. Otherwise, substituting in the definitions of $\boldsymbol{A}$ and $\boldsymbol{b}$ and defining the "facial contribution" l_i to be the signed distance from v to the SOI of the ith face along $\boldsymbol{d}$, we can then write l as a weighted sum of

$$l = \left(\sum_{i=1}^{m} \mu_i l_i\right) \Bigg/ \left(\sum_{i=1}^{m} \mu_i\right), \tag{15}$$

where

$$\mu_i = \begin{cases} a_i & \text{if expanding,} \\ a_i \cos^2(\theta_i) & \text{if contracting,} \end{cases} \qquad l_i = \begin{cases} c_i & \text{if expanding,} \\ c_i/|\cos(\theta_i)| & \text{if contracting,} \end{cases} \tag{16}$$

where a_i is the area of the ith face, c_i is the distance from v to the ith face offset incident on v, and θ_i is the angle between $\boldsymbol{d}$ and $\boldsymbol{n}_i$. This equation gives higher priority to expansion, which tends to smooth out corners and in turn have a stabilizing effect on the interface at saddle points.

Fig. 5 summarizes the complete face-offsetting method for advective and wavefrontal motion. It takes an interface mesh, the speed or velocity, a time step Δt, and other control parameters (including the threshold ε for the null-space, and the advective/wavefrontal switch) as input, and returns a potentially reduced time step $\alpha\Delta t$ and the scaled displacements. The driver routine that calls this procedure may then advance the time by $\alpha\Delta t$ and then repeat the process until the desired time step is reached. Note that if the speed or Δt is zero, then this algorithm reduces to feature-preserving surface mesh smoothing with integrated checking for mesh folding. Therefore, if an additional smoothing step is needed, one can simply invoke this procedure with $\Delta t = 0$.

5. Numerical results

In this section, we present some experimental results to demonstrate the effectiveness of FOM for both advective and wavefrontal interface propagation. We focus on assessing the accuracy of FOM especially in the presence of singularities, so our tests use meshes with fixed connectivity and only redistribute vertices using the build-in null-space smoothing. We first evaluate FOM for advective motion using two flow fields, which

have been previously used to test other moving-interface methods [7,8]. These flows are relatively easy if the surface remains smooth but nontrivial for mesh smoothing in the presence of sharp features or large curvature. We then consider propagation under wavefrontal motion, and demonstrate an application of FOM to the simulation of solid rocket motors.

5.1. Advective motion

5.1.1. Rotation of slotted sphere

Rotation has been widely used in the literature to test moving interface methods [7–9,14]. We adopt the problem of rotating the Zalesak's sphere [7], which is a unit sphere with a slot of 1/3 unit wide and 5/6 unit high aligned along the YZ plane. This sphere is centered at $(5/3, 0, 0)$ and then rotated counterclockwise around the origin in the XY plane with velocity

$$u(x, y, z) = -y, \tag{17}$$

$$v(x, y, z) = x, \tag{18}$$

$$w(x, y, z) = 0. \tag{19}$$

A time step of $\pi/100$ was used, so one revolution takes 200 time steps.

Fig. 6 shows the solutions using the coarsest mesh (with 157 vertices and 310 triangles, with average edge length roughly equal to the width of the slot) at intervals of one-fifth revolution. The meshes of the initial ($t = 0$) and final ($t = 2\pi$) configurations are drawn on top of each other. All the sharp features were preserved accurately, and the sphere was returned back to its original position at $t = T$, although the vertices were redistributed slightly. The relative volume error was 0.0025% after the complete rotation for the coarsest mesh and converged superlinearly as the mesh was refined (reduced by a factor of 16 after refining by a factor of 8), because null-space smoothing introduces very small perturbation even for coarse meshes and at singularities. In contrast, the "conservative" particle level set method in [7] lost 2.3% of volume when using one million grid points for this problem.

5.1.2. Reversal vortex flow

We now consider a more difficult problem of a reversal vortex flow. This flow stretches a body to create large curvatures without actually forming sharp singularities. This problem poses some similar challenges

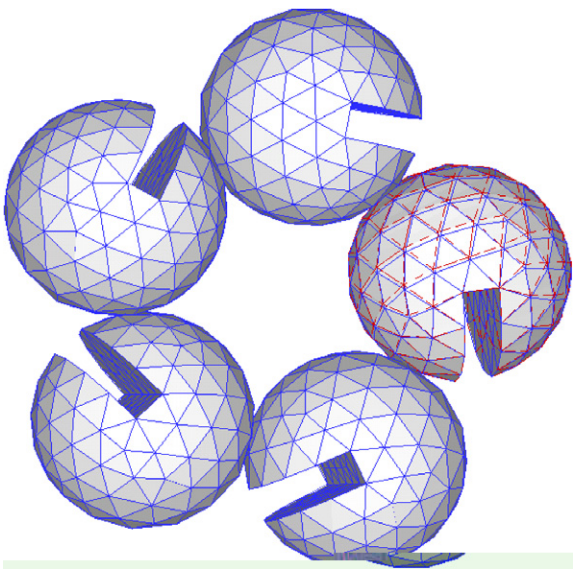


Fig. 6. Six snapshots of rotating slotted sphere at intervals of one-fifth revolution. Right-most sphere shows initial (dashed) and final (solid) meshes overlaid on top of each other.

as complex advective flows in real-world applications (such as turbulence), as the velocity field is very strong and nonuniform. The spatial velocity field of the vortex is given by

$$u(x,y,z) = \sin^2(\pi x)(\sin(2\pi z) - \sin(2\pi y)), \tag{20}$$

$$y(x,y,z) = \sin^2(\pi y)(\sin(2\pi x) - \sin(2\pi z)), \tag{21}$$

$$z(x,y,z) = \sin^2(\pi z)(\sin(2\pi y) - \sin(2\pi x)), \tag{22}$$

and we modularize time by multiplying the cosinusoidal function $\cos(\pi t/T)$ to make the flow periodic. The initial sphere has radius 0.15 centered at $(0.5, 0.75, 0.5)$. In [8], the period was chosen to be 2 or 4 to demonstrate the adaptive capability of their method for handling strong distortions. As we used fixed-connectivity meshes, we chose the test with the shorter period ($T = 2$).

Two triangular meshes were used in our test: a coarse mesh with 5832 vertices and 11,660 triangles and a fine mesh with 23,238 vertices and 46,472 triangles. The time step was chosen to be 0.01, so the whole computation took 200 iterations. At each time step, we obtained the face offsets by integrating the motion at vertices. On a Linux machine with a 3.2 GHz Pentium D processor, it took about 1.2 and 5 seconds per time step for the coarse and fine meshes, respectively. Fig. 7 shows the snapshots for the coarse mesh at times $t = 0$, 1, and 2. The relative volume error was 7.2×10^{-4} for the coarse mesh and 8.9×10^{-5} for the fine mesh. These errors were comparable with the conservative surface-marker volume-of-fluid method [8], which had a relative volume error ranging between 5.2×10^{-5} and 4×10^{-3} using a 32^3 mesh with 7748 additional markers at maximum deformation. Note that if Laplacian smoothing were used, the volume loss would have been more than 65%. If we restrict smoothing within the tangent planes (instead of the null space), the volume error would have been about 1% because the curvature is very large at maximum deformation.

5.2. Wavefrontal motion

Wavefrontal motion is more difficult than advective motion because of the nonlinearity. Some methods designed for advective processes, including the particle level set method [7] and surface-marker volume-of-fluid method [8] mentioned earlier do not support wavefrontal motion with a given normal speed.

5.2.1. Expanding cubes

In this test, we consider the problem of a cube expanding under unit speed. We study the convergence of FOM under wavefrontal motion using a series of meshes with fixed mesh connectivity. Fig. 8(a) shows the result using the coarsest mesh, where the colors indicate the magnitudes of total displacements of the vertices. As a reference, Fig. 8(b) shows the result using the advective variation. Under wavefrontal motion, the correct surface areas for wavefrontal motion and advective motion differ by 44.3% after unit time. Fig. 8(c) shows the evolution of the surface area for different meshes. The numerical solutions converged superlinearly to the exact solution as the mesh was refined. Note that if the cube were contracting, then the wavefrontal and advective variations of FOM would both deliver accurate results and be subjected to only roundoff error for this geometry.

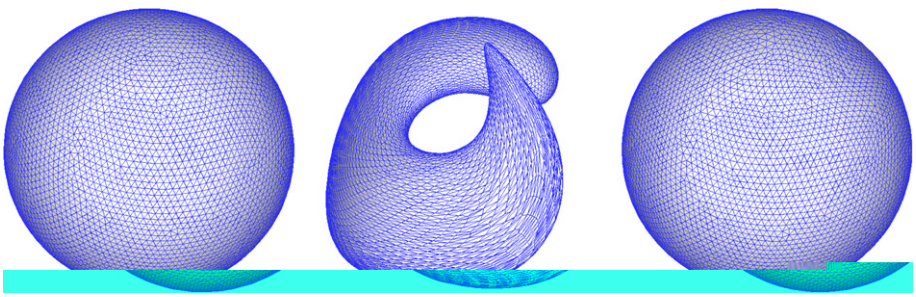


Fig. 7. Snapshots of 3-D reversal vortex test at times $t = 0$, $T/2$, and T.

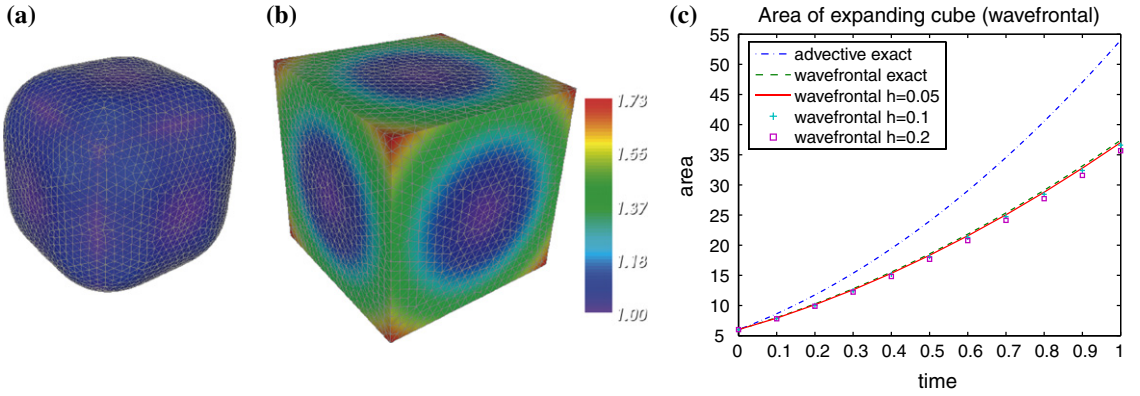


Fig. 8. Expanding cube under wavefrontal (a) and advective (b) expansion after unit time starting from unit cube. Image (c) shows evolution of area in time.

5.2.2. *Application to solid rocket motors*

Wavefrontal motion is important in many physical processes, such as burning, deposition, etching, etc. In this subsection, we present an application of FOM to the simulation of solid rocket motors. Our test performs a nearly complete burn (95% burn) of an attitude control motor (ACM) rocket, courtesy of Aerojet Inc. Our test assumed uniform speed, similar to commercial analysis tools for solid rocket motors, such as SPP'02 [26]. In reality, the regression rate of a solid rocket is fairly uniform except during initial ignition transients, so this setting is fairly representative. Unlike the existing commercial tools, however, FOM supports nonuniform speeds and has been applied to coupled simulations where the burning rate is determined by erosive burning models coupled with Navier–Stokes solvers. These coupled simulations are beyond the scope of this paper.

Fig. 9(a) shows a schematic of the geometry of the ACM rocket. Figs. 9(b)–(e) show the geometry after 0%, 25%, 50%, and 95% burns, respectively. This geometry looks deceptively simple, but it poses some major challenges because the front and rear ends of the propellant burns along the case of the rocket at a rate of two to three times the normal speed, and the singularities between the inner cylinder and the conical patches are composed of saddle points that expand and contract simultaneously along different directions. We use three meshes of different resolutions. The coarsest mesh has 5903 vertices and 11,806 triangles, and the average edge length is approximately one-twentieth of the circumference of the inner cylinder. Because the conical patches are compressed significantly during the simulation, we allowed the vertices on these conical patches to be redistributed to the cylindrical patch by marking the vertices between the patches as smooth.

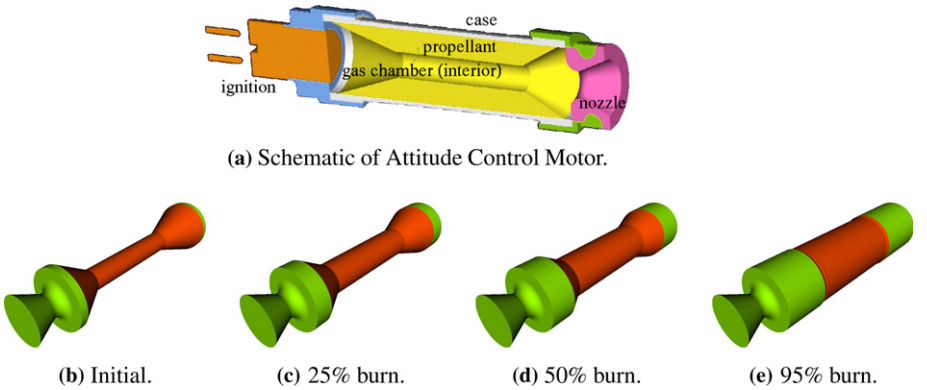


Fig. 9. Schematic of attitude control motor from Aerojet Inc., and geometry of gas chamber after 0%, 25%, 50%, and 95% burns. Darker (red) patches indicate burning surface. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

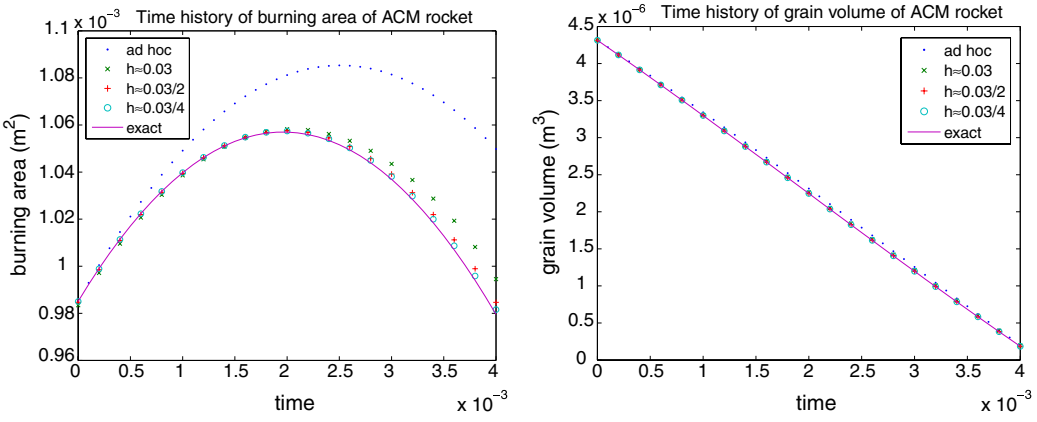


Fig. 10. Time history of burning area and propellant volume of attitude control motor under uniform speed.

In rocket simulations, the accuracy of the burning area and propellant volume plays a critical role in determining the lift of a rocket [27]. Fig. 10 shows a quantitative comparison of the total burning area and grain volume over time compared to the exact solution based on the envelope-of-balls construction. The error for the coarsest mesh was less than 1%, and exhibited superlinear convergence as the mesh was refined. In comparison, a simple *ad hoc* approximation to this problem in [28], which restricted every point to move radially, led to a relative error up to 8% in area and did not converge to the correct solution.

6. Conclusion

In this paper, we proposed a novel framework for interface propagation, called the face-offsetting method (FOM). This method is based on a generalized Huygens' principle, which unifies the views of advective and wavefrontal interfaces. FOM introduces the entropy condition into Lagrangian surface propagation, and delivers accurate solutions to both advective and wavefrontal motions, even in the presence of singularities or using coarse meshes. In addition, the method contains a build-in smoothing technique to redistribute vertices and provides techniques to ensure the integrity of the surface mesh. Our experimental tests demonstrated that the FOM is competitive with some of the most advanced methods while being simpler to implement. FOM is not meant to replace these existing methods. However, it provides a useful solution to some problems that are difficult to address using traditional techniques, such as moving boundaries in complex simulations using body-fitted meshes.

Our method is extensible in a number of ways. Some directions include the integration of robust mesh adaptation for more complex flows, resolution of topological changes, and extension to flows involving a combination of advective and wavefrontal motions. The implementation presented in this paper uses piecewise linear faces, but it may be adapted to support propagation using high-order elements (such as 6-node triangles) to achieve higher-order accuracy, using the idea of tangent planes in the solution of wavefrontal propagation. In addition, FOM is promising to be extended to support non-manifold surfaces, which occur in many applications such as multiphase flows and the simulation of bubbles.

Acknowledgments

This work was supported by the US Department of Energy through the University of California under subcontract B523819 with the University of Illinois at Urbana-Champaign, and by NSF and DARPA under CARGO Grant #0310446. This work began as part of the author's Ph.D. thesis research and continued at the Center for Simulation of Advanced Rockets, both under the direction of Prof. Michael Heath. The author is grateful for Prof. Heath's tremendous advice and continuing support. The author also thanks Prof. John Hart of the University of Illinois, Prof. John Sullivan of the Technische Universität Berlin, and Prof. Orion Lawlor of the University of Alaska for their help and discussions. The author thanks Xinlai Ni and Andrew

Colombi of University of Illinois for their help in testing earlier versions of the method. Finally, the author thanks anonymous reviewers for their suggestions in improving the presentation of the paper.

References

[1] S. Osher, J.A. Sethian, Fronts propagating with curvature-dependent speed: algorithms based on Hamilton–Jacobi formulations, J. Comput. Phys. 79 (1988) 12–49.
[2] S. Osher, R. Fedkiw, Level Set Methods and Dynamic Implicit Surfaces, Springer, Berlin, 2003.
[3] J.A. Sethian, Level Set Methods and Fast Marching Methods, Cambridge University Press, Cambridge, 1999.
[4] D. Juric, G. Tryggvason, A front-tracking method for dendritic solidification, J. Comput. Phys. 123 (1996) 127–148.
[5] J. Glimm, J. Grove, X.L. Li, D.C. Tan, Robust computational algorithms for dynamic interface tracking in three dimensions, SIAM J. Sci. Comput. 21 (2000) 2240–2256.
[6] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, Y.-J. Jan, A front-tracking method for the computations of multiphase flow, J. Comput. Phys. 169 (2001) 708–759.
[7] D. Enright, R. Fedkiw, J. Ferziger, I. Mitchell, A hybrid particle level set method for improved interface capturing, J. Comput. Phys. 183 (2002) 83–116.
[8] E. Aulisa, S. Manservisi, R. Scardovelli, A surface marker algorithm coupled to an area-preserving marker redistribution method for three-dimensional interface tracking, J. Comput. Phys. 197 (2004) 555–584.
[9] J. Du, B. Fix, J. Glimm, X. Jia, X. Li, Y. Li, L. Wu, A simple package for front tracking, J. Comput. Phys. 213 (2006) 613–628.
[10] D. Enright, F. Losasso, R. Fedkiw, A fast and accurate semi-Lagrangian particle level set method, Comput. Struct. 83 (2005) 479–490.
[11] T.F. Dupont, Y. Liu, Back and forth error compensation and correction methods for semi-Lagrangian schemes with application to level set interface computations, Math. Comp. (2006) in press.
[12] E.G. Puckett, A.S. Almgren, J.B. Bell, D.L. Marcus, W.J. Jider, A high-order projection method for tracking fluid interfaces in variable density incompressible flows, J. Comput. Phys. 130 (1997) 269–283.
[13] M. Rudman, Volume-tracking methods for interfacial flow calculations, Int. J. Numer. Meth. Fluid 24 (1997) 671–691.
[14] W.J. Rider, D.B. Kothe, Reconstructing volume tracking, J. Comput. Phys. 141 (1998) 112–152.
[15] D.J.E. Harvie, D.F. Fletcher, A new volume of fluid advection algorithm: the stream scheme, J. Comput. Phys. 162 (2000) 1–32.
[16] L.Q. Chen, Phase-field models for microstructure evolution, Annu. Rev. Mater. Res. 32 (2002) 113–140.
[17] W.J. Rider, D.B. Kothe, A marker particle method for interface tracking, in: Proceedings of 6th International Symposium on Computational Fluid Dynamics, 1995, p. 976.
[18] T. Maekawa, An overview of offset curves and surfaces, Comput. Aided Des. 31 (1999) 165–173.
[19] B. Pham, Offset curves and surfaces: a brief survey, Comput. Aided Des. 24 (1992) 223–229.
[20] B.B. Baker, E.T. Copson, The Mathematical Theory of Huygens' Principle, Clarendon Press, Oxford, 1950.
[21] M.T. Heath, Scientific Computing: An Introductory Survey, second ed., McGraw-Hill, New York, 2002.
[22] P.S. Heckbert, M. Garland, Optimal triangulation and quadric-based surface simplification, Comput. Geom. (1999) 49–65.
[23] X. Jiao, Volume and feature preservation in surface mesh optimization, in: Proceedings of 15th International Meshing Roundtable (2006) in press.
[24] P.J. Frey, P.-L. George, Mesh Generation: Application to Finite Elements, Hermes, 2002.
[25] P. Frey, About surface remeshing, in: Proceedings of 9th International Meshing Roundtable, 2000, pp. 123–136.
[26] D. Coats, J. French, S. Dunn, D. Berker, Improvements to the solid performance program (SPP), in: 39th AIAA/ASME/SAE/ASEE Joint Propulsion Conference, Huntsville, AL, 2003, AIAA-2003-4504.
[27] G.P. Sutton, O. Biblarz, Rocket Propulsion Elements, seventh ed., Wiley-Interscience, New York, 2000.
[28] J. Blazek, Flow simulation in solid rocket motors using advanced CFD, in: 30th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit, 2003, AIAA 2003-5111.