Projection Methods Coupled to Level Set Interface Techniques*

JINGYI ZHU AND JAMES SETHIAN

Lawrence Berkeley Laboratory and Department of Mathematics, University of California, Berkeley, California 94720

Received January 28, 1991; revised August 28, 1991

In this work, we consider hydrodynamic problems with cold flame propagation by merging a second-order projection method for viscous Navier–Stokes equations with modern techniques for computing the motion of interfaces propagating with curvature-dependent speeds. This is part of our efforts to try and approximate the solution of a simplified model of turbulent combustion. Results are given for a simple model of a flame burning in driven cavities and shear layers. © 1992 Academic Press, Inc.

INTRODUCTION

In this paper, we merge modern techniques for computing the solution to the viscous Navier–Stokes equations with new techniques for computing the motion of interfaces propagating with curvature-dependent speeds. The resulting algorithm tracks the motion of an evolving interface in a complex flow field and easily handles complex changes in the front, including the development of spikes and cusps, topological changes, and breaking/merging. As examples, we apply the resulting algorithm to interface boundaries in a driven cavity and in a shear layer and cold flame propagation in a hydrodynamic field.

At the core of this work are two separate numerical algorithms. The first is a second-order projection method due to Bell, Colella, and Glaz [1] which extends the original projection technique introduced by Chorin [4, 5]. This algorithm is used to track the evolving hydrodynamic flow field and has produced some dramatic results of evolving flow, see Bell and Marcus [2, 3]. The second set of numerical algorithms is a new class of schemes to follow the evolution of propagating interfaces. These techniques, introduced by Osher and Sethian [12], rely on a level set partial differential equation to describe the motion of the propagating interface which may be approximated by borrowing technology from the numerical solution of hyperbolic conservation laws. The merger of these two techniques results in a robust approach to hydrodynamic problems with interfaces.

The work presented here, in some sense, is the next stage in the incorporation of the interface methodology presented in [12] into modern techniques for computational fluid mechanics. Starting from the original interface work in [12], the interface equations were coupled to the equations of compressible gas dynamics in [10] to study the Rayleigh–Taylor and Kelvin–Helmholtz instabilities. In that work, two approaches were examined. In one approach, the interface equation was directly incorporated as a fifth conservation law and approximated using standard shock technology. Alternatively, the compressible flow equations were solved at each time step for the velocity field, which was then used to advect the interface. In [10] the coupling between the interface and the underlying hydrodynamics rested on locating the interface to determine the appropriate constant in the γ-gas law.

In this paper, we focus on the application of the level set interface methodology to the case of viscous flow. These equations are solved by a projection-type method which first advances the velocity field by one time step and then enforces incompressibility as a constraint by projecting the new time step data onto the space of incompressible flows. As examples, we consider simulations in which the location of the interface may change as a result of both self-propagation (due to local geometric properties, for example) and, through underlying advection, by the hydrodynamic flow field. However, in this paper we do not consider problems in which the feedback mechanism is completed and the interface affects the hydrodynamic field. One such example of complex fluid–interface interactions with an extremely intricate feedback mechanism between the propagating interfaces and an underlying heat diffusion equation applied to crystal growth and dendrite solidification was studied using the level set methodology in [16].

The outline of this paper is as follows. First (Section 1), we begin with a description of the second-order projection method as developed by Bell, Colella, and Glaz [1]. Next (Section 2), we derive the interface equations for the level set approach and discuss numerical approximations. In

* This work was supported in part by the Applied Mathematics Subprogram of the Office of Energy Research, U.S. Department of Energy under Contract DE-AC03-76SF00098. The first author is also a Hans Lewy Post-doctoral Fellow.

Section 3, we present the numerical details of our implementation and give results of numerical simulations of propagating interfaces in driven cavities, shear layers, and the numerical simulations of cold flames in hydrodynamic fields.

1. THE SECOND-ORDER PROJECTION METHOD

1.1. *Theory*

In this section we describe a second-order projection method for the time-dependent, incompressible Navier–Stokes equations developed by Bell, Colella, and Glaz [1]. The Navier–Stokes equations for incompressible flows are

$$\mathbf{u}_t + (\mathbf{u}\cdot\nabla)\mathbf{u} = -\nabla p + \frac{1}{\mathrm{Re}}\nabla^2\mathbf{u} \tag{1.1}$$

$$\nabla\cdot\mathbf{u} = 0 \tag{1.2}$$

on a domain Ω, where $\mathbf{u}$ is the velocity field, p is the pressure, and Re is the Reynolds number. The initial-boundary value problems for the Navier–Stokes equations include specifying the initial $\mathbf{u}$ throughout Ω, and the boundary condition for $\mathbf{u}$, but not for p on $\partial\Omega$.

The original projection method for incompressible flows was introduced by Chorin [4, 5] to view the incompressibility condition as a constraint. The central idea is to update the velocity by first ignoring the incompressibility, marching forward one time step and then projecting onto the space of the incompressible flows. It was inspired by the Hodge decomposition which states that a vector field $\mathbf{v}$ defined on a domain Ω can be uniquely decomposed into a divergence-free part $\mathbf{u}$, which satisfies the boundary condition $\mathbf{u}\cdot\mathbf{n} = 0$, where $\mathbf{n}$ is the unit normal vector to the boundary, and the gradient of some scalar function ϕ. We shall follow the construction given by Bell, Colella, and Glaz [1].

We begin by reorganizing the terms in the Navier–Stokes equations in the form

$$\mathbf{u}_t + \nabla p = \frac{1}{\mathrm{Re}}\nabla^2\mathbf{u} - (\mathbf{u}\cdot\nabla)\mathbf{u}. \tag{1.3}$$

Thus, by the Hodge decomposition, $\mathbf{u}_t$ is the divergence-free component obtained by projection of the right-hand side onto the space of incompressible flows. We extend this argument to the semi-discrete form, defining $\mathbf{u}^n$ to be the velocity at $t = n\,\Delta t$, $p^{n+1/2}$ the pressure at $t = (n+\frac{1}{2})\,\Delta t$. Given $\mathbf{u}^n$, $p^{n-1/2}$, we hope to find $\mathbf{u}^{n+1}$ and $p^{n+1/2}$ satisfying

$$\frac{\mathbf{u}^{n+1}-\mathbf{u}^n}{\Delta t} + \nabla p^{n+1/2} = \frac{1}{\mathrm{Re}}\nabla^2\left(\frac{\mathbf{u}^n+\mathbf{u}^{n+1}}{2}\right) - (\mathbf{u}\cdot\nabla\mathbf{u})^{n+1/2} \tag{1.4}$$

$$\nabla\cdot\mathbf{u}^{n+1} = 0. \tag{1.5}$$

While this scheme is second order in time, it is not practical because of the poor conditioning of the linear system that arises [1]. The second-order method we describe here, like the original first-order method, computes an intermediate vector field and then projects it onto the divergence-free field. The other two parts of this scheme are the evaluation of the non-linear advection term $(\mathbf{u}\cdot\nabla\mathbf{u})^{n+1/2}$ and the projection itself.

Instead of solving Eq. (1.4) with the incompressibility constraint (1.5), we solve the intermediate velocity field $\mathbf{u}^*$,

$$\frac{\mathbf{u}^*-\mathbf{u}^n}{\Delta t} + \nabla p^{n-1/2} = \frac{1}{\mathrm{Re}}\nabla^2\left(\frac{\mathbf{u}^n+\mathbf{u}^*}{2}\right) - (\mathbf{u}\cdot\nabla\mathbf{u})^{n+1/2}. \tag{1.6}$$

Let us project $\mathbf{u}^*$ onto the divergence-free field and write

$$\mathbf{u}^* = \mathbf{u}^{n+1} + \Delta t\,\nabla p^*; \tag{1.7}$$

then Eq. (1.6) becomes

$$\frac{\mathbf{u}^{n+1}-\mathbf{u}^n}{\Delta t} + \nabla(p^{n-1/2}+p^*) = \frac{1}{\mathrm{Re}}\nabla^2\left(\frac{\mathbf{u}^n+\mathbf{u}^{n+1}}{2}\right) - (\mathbf{u}\cdot\nabla\mathbf{u})^{n+1/2} + \frac{\Delta t}{2\mathrm{Re}}\nabla^2(\nabla p^*) \tag{1.8}$$

$$\nabla\cdot\mathbf{u}^{n+1} = 0. \tag{1.9}$$

In other words, the left-hand side is decomposed into the divergence-free part and the gradient of the scalar field $p^{n-1/2}+p^*$. By comparing Eqs. (1.8)–(1.9) with Eqs. (1.4)–(1.5) we see that

$$p^{n+1/2} = p^{n-1/2} + p^* \tag{1.10}$$

if the term $(\Delta t/2\mathrm{Re})\,\nabla^2(\nabla p^*)$ vanishes. This can be achieved by an iterative process. Namely, we take $p^{n+1/2,k}$ and solve Eq. (1.6) for $\mathbf{u}^{*,k}$; once we have $\mathbf{u}^{*,k}$ we decompose the right-hand side of Eq. (1.6) into

$$\frac{\mathbf{u}^{n+1,k}-\mathbf{u}^n}{\Delta t} + \nabla p^{n+1/2,k+1} = \frac{1}{\mathrm{Re}}\nabla^2\left(\frac{\mathbf{u}^n+\mathbf{u}^{*,k}}{2}\right) - (\mathbf{u}\cdot\nabla\mathbf{u})^{n+1/2}. \tag{1.11}$$

Then we use this $p^{n+1/2,k+1}$ to repeat the process until we have the convergence $\mathbf{u}^{*,k} \to \mathbf{u}^{n+1,k} \to \mathbf{u}^{n+1}$ and $p^{n+1/2,k} \to p^{n+1/2}$. The first guess $p^{n+1/2,0}$ can be taken as $p^{n-1/2}$ for $n \geqslant 1$, and for $n = 0$ we take zero as the first guess. In practice, we have found that for Δt small enough, iteration is not needed beyond the first time step and the single step process in (1.11) is sufficient for our purpose. At the initial time step, we do not have the exact pressure field, so it is necessary to iterate this process more than once.

1.2. *Implementation*

Next, we describe the spatial discretization and the evaluation of the non-linear terms. For simplicity, we assume a rectangular domain with $n \times m$ grid points. First we point out that the terms $(\mathbf{u} \cdot \nabla \mathbf{u})^{n+1/2}$ are determined explicitly, based entirely on information at t^n. Thus, solving Eq. (1.11) in time reduces to solving the heat equation with a known source term. The discretization is done in the following way: the velocity and pressure gradients are defined in the centers of all cells and the pressure is defined on all corners. The diffusion operator is approximated by central difference with special care taken at the boundary points. This leads to a five-band linear system which, together with the known source terms, can be solved by a standard conjugate gradient method.

Next, we briefly describe the evaluation of the known source term $(\mathbf{u} \cdot \nabla \mathbf{u})^{n+1/2}$. First, the equation is set on the centers of all cells. Thus, $(\mathbf{u} \cdot \nabla \mathbf{u})^{n+1/2}$ is evaluated at the center of a cell. It is approximated by the edge values at $t^{n+1/2}$ in the following way:

$$\begin{aligned}(\mathbf{u} \cdot \nabla \mathbf{u})^{n+1/2}_{i,j} = {} & \frac{1}{2}(u_{i+1/2,j} + u_{i-1/2,j}) \\ & \times \frac{\mathbf{u}_{i+1/2,j} - \mathbf{u}_{i-1/2,j}}{\Delta x} \\ & + \frac{1}{2}(v_{i,j+1/2} + v_{i,j-1/2}) \\ & \times \frac{\mathbf{u}_{i,j+1/2} - \mathbf{u}_{i,j-1/2}}{\Delta y}. \qquad (1.12)\end{aligned}$$

To illustrate the way of calculating the edge values at $t^{n+1/2}$, we take $u_{i+1/2,j}$ as an example. We can use Taylor expansion both in time and space and extrapolate either from the left cell,

$$u^{n+1/2,L}_{i+1/2,j} = u^n_{i,j} + \frac{\Delta x}{2} u_{x,i,j} + \frac{\Delta t}{2} u_{t,i,j}, \qquad (1.13)$$

or from the right cell. We choose to approximate values by means of an upwinding scheme. In this method we use the Godunov scheme. Since the scheme is explicit we require the CFL condition

$$\max_{ij} \left(\frac{|u_{ij}|\, \Delta t}{\Delta x}, \frac{|v_{ij}|\, \Delta t}{\Delta y} \right) \leqslant 1 \qquad (1.14)$$

to be satisfied.

We construct the spatial and temporal derivatives by the following steps. For spatial derivatives we compute a linear profile within each cell first, and then limit the slopes so new maxima and minima are not introduced. For the temporal derivatives we go back to the Navier–Stokes equations to obtain an approximation for $\mathbf{u}_t$ by information at the previous time step. Again, we present the algorithm as derived by Bell, Colella, and Glaz; details may be found in [1].

The last part of each iteration is the projection itself. The direct computation of the divergence-free part of $\mathbf{v}$ is based on a discrete Galerkin formulation. That is, we find a local basis for the finite-dimensional discrete incompressible vector field and use the representation in this basis to determine the projection of any vector onto this field. To guarantee that such a finite-dimensional basis exists we need some requirements for the discrete gradient G and the discrete divergence D. The Hodge decomposition is based on the fact that if we integrate by parts we find that

$$\int_\Omega \mathbf{u} \cdot \nabla \phi \, d\mathbf{x} = -\int_\Omega \phi \nabla \cdot \mathbf{u} \, d\mathbf{x} = 0. \qquad (1.15)$$

One crucial part of the numerical formulation is the construction of the finite difference gradient operator G and the divergence operator D to satisfy the discrete analogue

$$(\mathbf{u}, \nabla \phi) = -(D \cdot \mathbf{u}, \phi) \qquad (1.16)$$

for discrete velocity field $\mathbf{u}$ and scalar field ϕ. This is a summation-by-parts procedure and one operator will define the other. In our case we define G first at the center of a cell to be

$$(G\phi)_{i,j} = \begin{pmatrix} \phi_x \\ \phi_y \end{pmatrix} = \begin{pmatrix} \dfrac{\begin{pmatrix} \phi_{i+1/2,j+1/2} + \phi_{i+1/2,j-1/2} \\ -\phi_{i-1/2,j+1/2} - \phi_{i-1/2,j-1/2} \end{pmatrix}}{2\Delta x} \\ \dfrac{\begin{pmatrix} \phi_{i+1/2,j+1/2} + \phi_{i-1/2,j+1/2} \\ -\phi_{i+1/2,j-1/2} - \phi_{i-1/2,j-1/2} \end{pmatrix}}{2\Delta x} \end{pmatrix} \qquad (1.17)$$

for both interior and boundary points. By the summation-by-parts of Eq. (1.16) we determine the discrete divergence operator implicitly.

After the derivation of operator D we define the vector space

$$V^d = \{\mathbf{u}\colon (D\mathbf{u})_{i+1/2,j+1/2} = 0; \quad i = 1, ..., n-1, j = 1, ..., m-1\}. \qquad (1.18)$$

Since the discrete fluid velocity space V is the direct sum of V^d and the space of discrete potentials, Stephens *et al.* [17] have shown that the dimension of this space

$$\begin{aligned}\dim V^d &= \dim V - \dim G \\ &= \dim V - [(n+1)(m+1) - \dim(\operatorname{Ker} G)]. \qquad (1.19)\end{aligned}$$

We note that $\dim V = 2nm$ and $\dim(\operatorname{Ker} G) = 2$, so

$$\dim V^d = (n-1)(m-1). \qquad (1.20)$$

To look for a basis of V^d we need only to look for a basis of the discrete stream function field, which is a scalar field. Let $\psi^{i+1/2,j+1/2} = 1$ on the corner $(i+1/2, j+1/2)$ and 0 everywhere else; then the velocity field induced by this stream function, $G^{\perp}\psi^{i+1/2,j+1/2}$ is, first of all, divergence free and, second, they are linearly independent. Furthermore, there are exactly $(n-1)(m-1)$ interior corner points; we conclude that they form a basis of V^d.

Once we have an explicit basis, we write the projection of any vector $\mathbf{v}$ in the form

$$P\mathbf{v} = \sum_{i,j} \alpha_{i+1/2,j+1/2} G^{\perp}\psi^{i+1/2,j+1/2} \qquad (1.21)$$

and, since $\mathbf{v} - P\mathbf{v}$ is perpendicular to V^d, we have

$$\begin{aligned}&\left(\sum_{i,j} \alpha_{i+1/2,j+1/2} G^{\perp}\psi^{i+1/2,j+1/2},\ G^{\perp}\psi^{i'+1/2,j'+1/2}\right) \\ &\quad = (\mathbf{v}, G^{\perp}\psi^{i'+1/2,j'+1/2}) \qquad \text{for all} \quad i', j'. \qquad (1.22)\end{aligned}$$

These constitute the linear system for the projection.

The linear system is in fact a discrete form of the vorticity-stream function equation $-\nabla^2\psi = \omega$, with the inclusion of boundary conditions for ψ. The stencil is derived from forms of the discrete gradient operator G and the divergence operator D thus defined through the adjoint relation (1.16), and in this case, it is the standard five-point turned by an angle of $\pi/4$, with mesh size $\sqrt{2}\,h$ for the case $\Delta x = \Delta y = h$. This decouples the system into two independent systems; which is not surprising, since $\dim(\operatorname{Ker} G) = 2$.

2. PROPAGATING INTERFACES BY LEVEL SET TECHNIQUE

In this section, we describe the details of a new class of algorithms for the following moving interfaces. These techniques were introduced in [12] and grew out of a link between interfaces propagating with curvature-dependent speed and hyperbolic conservation laws discussed in [14, 15]. They can be used to track highly complex moving interfaces in two and three space dimensions. Because these techniques do not rely on a discrete parameterization of the front itself, they naturally handle situations in which the front may develop cusps and spikes, change topology, and break/merge. The equations of motion and numerical approximations were discussed in [12]. Recently, these techniques have been applied to interface problems in the development of singularities in mean curvature flow [14, 15], compressible gas dynamics [10], and crystal growth and dendrite solidification [16]. In addition, theoretical analysis of mean curvature flow based on the level set model presented in [12] has recently been developed in [9]. Below, we present the basic ideas behind these techniques. The most straightforward derivation of the basic equations of motion was given in [10], and we follow that derivation below.

2.1. *Equations of Motion*

Suppose we wish to follow the evolution of an interface $\Gamma(t)$ propagating with speed F normal to itself, in either two or three dimensions. The essential idea is to construct a function $\phi(\mathbf{x}, t)$ defined in all of the domain, such that the level set $\{\phi = 0\}$ always corresponds to the position of the front $\Gamma(t)$. That is,

$$\Gamma(t) = \{\mathbf{x} : \phi(\mathbf{x}, t) = 0\}. \qquad (2.1)$$

Suppose we can smoothly extend F to all of the domain. (In fact, we only need to construct F in a neighborhood of the zero level set to derive the equation of motion, but the practical implementation is easier with a full extension). We can then derive a partial differential equation for the evolution of ϕ. Initialize $\phi(\mathbf{x}, 0)$ such that

$$\phi(\mathbf{x}, 0) = \text{distance from } \mathbf{x} \text{ to } \Gamma(t), \qquad (2.2)$$

where the plus (minus) sign is chosen if $\mathbf{x}$ is inside (outside) the initial front $\Gamma(t=0)$. Pick any level set $\phi = C$ and let $\mathbf{x}(t)$ be the trajectory of a particle located on this level set, so that for all time we have

$$\phi(\mathbf{x}(t), t) = C. \qquad (2.3)$$

Since the level set moves with speed F normal to itself, we must have that

$$\frac{\partial \mathbf{x}}{\partial t} \cdot \mathbf{n} = F, \qquad (2.4)$$

where $\mathbf{n}$ is the normal vector given by $\mathbf{n} = -\nabla\phi/|\nabla\phi|$.

By the chain rule, we have that

$$\phi_t + \nabla\phi \cdot \frac{\partial \mathbf{x}}{\partial t} = 0 \tag{2.5}$$

and substitution yields

$$\phi_t - F\,|\nabla\phi| = 0 \tag{2.6}$$

$$\phi(\mathbf{x}, t=0) = \text{given}. \tag{2.7}$$

This equation yields the motion of the front $\Gamma(t)$ with normal velocity F on the level set $\phi = 0$. We refer to this equation as a "Hamilton–Jacobi" level set formulation. Strictly speaking, it is only a Hamilton–Jacobi equation in the case when F is constant, but the flavor of Hamilton–Jacobi equations is present.

To summarize, we have derived an equation of motion for a higher dimensional function ϕ for which the level set $\varphi = 0$ always corresponds to the motion of the original front. Another way to say this is that we have transformed the Lagrangian equation which would have resulted from a parameterization of the moving interface into an Eulerian equation on a fixed grid of one higher dimension. We have traded an $(n-1)$-dimensional hypersurface for an n-dimensional Eulerian problem.

Fortunately, the advantages of this exchange far outweigh the additional computational energy required by the extra dimension. To begin, we observe that the function $\phi(\mathbf{x}, t)$ always remains a function, even if the level surface $\phi = 0$ corresponding to the front $\Gamma(t)$ changes topology, breaks, or merges. In such cases, parameterizations of the front often break down. As an example, consider two circles in R^2 expanding outwards with normal velocity $V = 1$. The initial function ϕ is double-humped. As ϕ evolves under the Hamilton–Jacobi equation of motion, the topology of the front $\phi = 0$ changes. When the two circles expand, they meet and merge into a single closed curve with two corners. This is reflected in the change of topology of the level set $\phi = 0$.

Thus, the level set approach avoids the complex bookkeeping that plagues discrete parameterization techniques when the interface changes topology. Another advantage is that the technique is applicable in any number of space dimensions: calculations of interfaces propagating in three space dimensions are discussed in detail in [12].

Finally, the central advantage of this approach is that, because we have posed an Eulerian problem for the motion of the propagating interfaces, fixed grid finite differences may be used to approximate the equations of motion. While care must be taken to choose difference schemes that satisfy an entropy condition for propagating fronts, the most basic versions of the schemes presented in [12] are extremely straightforward and simple to program. In the next section we give the motivation and form of the most basic technique.

2.2. *Link between Propagating Interfaces and Hyperbolic Conservation Laws*

It is tempting to use a central difference approximation to the gradient in Eq. (2.6) and thus produce the obvious explicit scheme (central difference in space, forward difference in time) for the update ϕ^{n+1}. Unfortunately, such an approximation is unworkable, for reasons which we now explain. For details, see [12, 14, 15].

Consider the simple case of a front propagating with speed function $F = 1 - \varepsilon\kappa$, where ε is a small parameter and κ is the local curvature. The equation of motion for the propagating function ϕ is then given by (see [12])

$$\phi_t - |\nabla\phi| = \varepsilon\nabla \cdot \left(\frac{\nabla\phi}{|\nabla\phi|}\right); \tag{2.8}$$

here we have used the coordinate-free definition of the curvature. Numerical evidence in [14, 15], followed by a proof in [12] shows that for $\varepsilon > 0$, the right-hand side diffuses sharp gradients and forces the ϕ to stay smooth for all time. Conversely, for $\varepsilon = 0$, corners develop and a singularity develops in the curvature. This situation is analogous to solutions of hyperbolic conservation laws, in which the absence of viscosity on the right-hand side allows the development of shock discontinuities in the propagating solution. Indeed, an entropy is required to force the correct solution for propagating interfaces which is equivalent to the one required for hyperbolic conservation laws. A full description of this entropy condition and the link between propagating interfaces and hyperbolic conservation laws are given in [14, 15].

Thus, an accurate numerical approximation to the equation for a propagating interface must pick out the correct entropy-satisfying solution and avoid excessive smearing at sharp discontinuities. This leads quite naturally to the use of schemes borrowed from the numerical solutions of hyperbolic conservation laws, where stable, consistent, entropy-satisfying schemes have a rich history. For an overview of shock schemes for solving conservation laws, see [11].

2.3. *Numerical Approximation to the Level Set Equations*

Complete explanation of the use of shock schemes for approximating the level set equation (2.6) may be found in [12]. Briefly, consider a one-dimensional version of the level set equation, and let

$$H(\phi_x) = (\phi_x^2)^{1/2}. \tag{2.9}$$

Then a forward time-discrete version of Eq. (2.6) may be written as

$$\phi^{n+1} = \phi^n + \Delta t\, H(\phi_x). \tag{2.10}$$

Let g be an appropriate numerical flux function approximating H. Then we may directly approximate the spatial term and write

$$\phi_i^{n+1} = \phi_i^n + \Delta\, t g(D_x^- \phi_i^n, D_x^+ \phi_i^n), \tag{2.11}$$

where D_x^+ (D_x^-) is the forward (backward) difference operator. In multiple space dimensions and the special case where $H(u) = u^2$, a particularly straightforward numerical flux function was given in [8, 12], namely,

$$\phi_i^{n+1} = \phi_i^n + \Delta t(F) \times ((\min(D_x^- \phi_i^n, 0))^2 + (\max(D_x^+ \phi_i^n, 0))^2). \tag{2.12}$$

This conservative monotone scheme is an *upwind* method, in that it differences in the direction of propagating characteristics. Equation (2.12) completely specifies the numerical approximation to Eq. (2.6). Details may be found in [12].

2.4. *Addition of Underlying Advection*

In the problems under study in this paper, the underlying advection plays the key role in the transport of the interface. This may be easily incorporated in the level set framework. Let U be the velocity field throughout the domain. Then the equation of motion for an interface propagating normal to itself with speed F and advected by an underlying fluid with velocity U is given by

$$\phi_t - F\,|\nabla\phi| + U \cdot \nabla\phi = 0. \tag{2.13}$$

3. NUMERICAL IMPLEMENTATION AND RESULTS

In this section we describe briefly the numerical implementation and present our results. The fluid calculation part is independent of the front propagation part and therefore can be tested before we merge the two parts. This part of the algorithm is identical to the one in [1]. We have developed codes to solve both the Dirichlet and the periodic boundary values problems. For each case we compute the solution on uniform grids with $\Delta x = \Delta y = \frac{1}{64}$, $\frac{1}{128}$ and $\Delta t = \min(\Delta x/2, \Delta t_{\text{cfl}})$, where Δt_{cfl} is the largest time step the CFL condition allows. The time stepping is absolutely stable, since it is a Crank–Nicholson approximation and is done by a straightforward conjugate gradient method. The evaluation of the nonlinear terms involves Taylor extrapolations and solving Riemann problems. We use the Godunov scheme for this purpose. The projection part is the most time-consuming part as far as computing time is concerned. We use a preconditioned conjugate gradient method to solve the linear system and choose the incomplete Cholesky decomposition to be the preconditioner.

For the Dirichlet problems we choose the driven cavity problem because of its distinctive vortex feature and literature on the subject. We solve the Navier–Stokes equations in the unit box $\Omega = [0, 1] \times [0, 1]$. The initial condition is that $\mathbf{u} = 0$ inside the box with no-slip and no-flow boundary condition along the boundaries except at the top, where $\mathbf{u} = (1, 0)$. With Reynolds number Re = 1000, the flow takes about 50 dimensionless time units to become fully developed. We examine the velocity field at this time and note that, besides the main eddy, which is located near the box center, a little bit above and to the right of the box center, there are secondary eddies near the lower corners. The strength of these eddies is substantially weaker than that of the main eddy.

In the case of periodic problems, the absence of boundaries eliminates the boundary layers and viscosity plays a negligible part in the motion when it is small. For solutions to the Navier–Stokes equations with large Reynolds

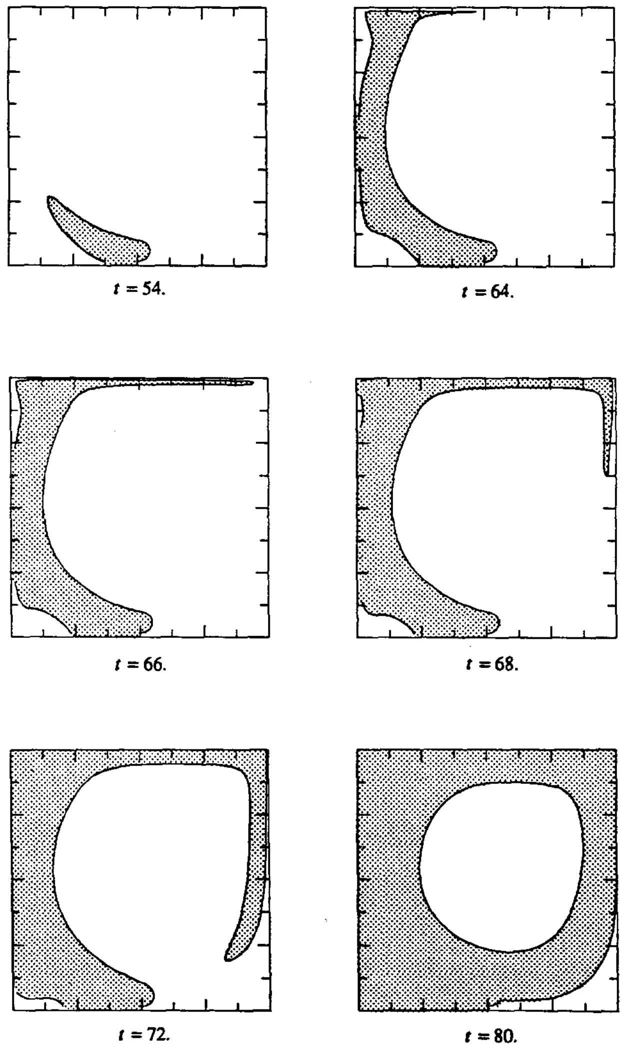

FIG. 1. Constant ignition at a bottom point: $F(\kappa) = 0.01$.

numbers, the inviscid solutions will provide us with a good approximation. Thus, to simplify the calculation and concentrate on the vortex dynamics and its impact on burning flames, we instead solve the Euler equations. The linear algebra part for periodic geometry is similar to the Dirichlet geometry, except that we need to pay more attention to the null space elements. We consider the smoothed jet flow in the periodic domain [0, 1] × [0, 1] and use the same initial data as in [1], where

$$u = \begin{cases} \tanh\left(\dfrac{y-0.25}{\rho}\right) & \text{for } y \leqslant 0.5 \\ \tanh\left(\dfrac{0.75-y}{\rho}\right) & \text{for } y > 0.5 \end{cases} \tag{3.1}$$

$$v = \delta \sin(2\pi x). \tag{3.2}$$

We also choose the parameters $\rho = \frac{1}{30}$ and $\delta = 0.05$ to help us to compare our results with the results in [1]. We have obtained identical vorticity contours to those in [1]. As we refine the calculations, the vorticity layers become thinner and thiner. This agrees with the results in [1].

Next we consider a propagating interface by studying the propagation of a cold flame burning with speed $F = S - \varepsilon\kappa$, where ε is a small parameter and κ is the local curvature, see [14]. The equation to be solved now is

$$\phi_t - S\,|\nabla\phi| + U\cdot\nabla\phi = \varepsilon\,\frac{\phi_{xx}\phi_y^2 - 2\phi_{xy}\phi_x\phi_y + \phi_{yy}\phi_x^2}{\phi_x^2 + \phi_y^2}. \tag{3.3}$$

Here, we repeat the calculations performed in [13], obtained using a volume-of-fluid interface technique. Since the flame propagation does not affect the fluid calculation in this model, we use a first-order explicit scheme to update ϕ. A second-order version of the level set scheme is derived in [12]. The passive advection part $U\cdot\nabla\phi$ is well known [7]

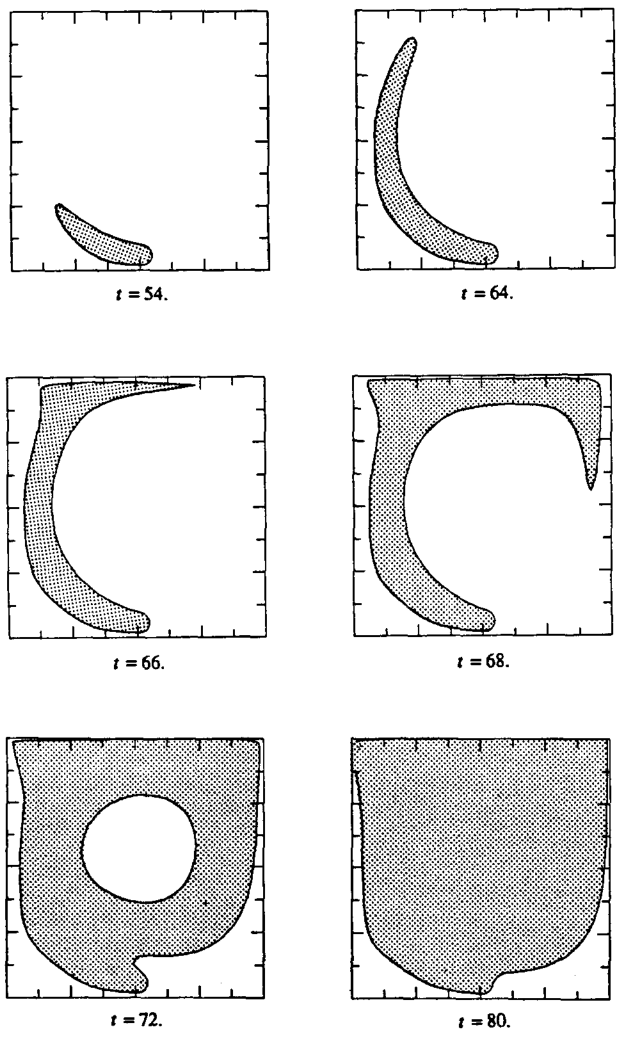

FIG. 2. Constant ignition at a bottom point: $F(\kappa) = 0.01 - 0.01\kappa$.

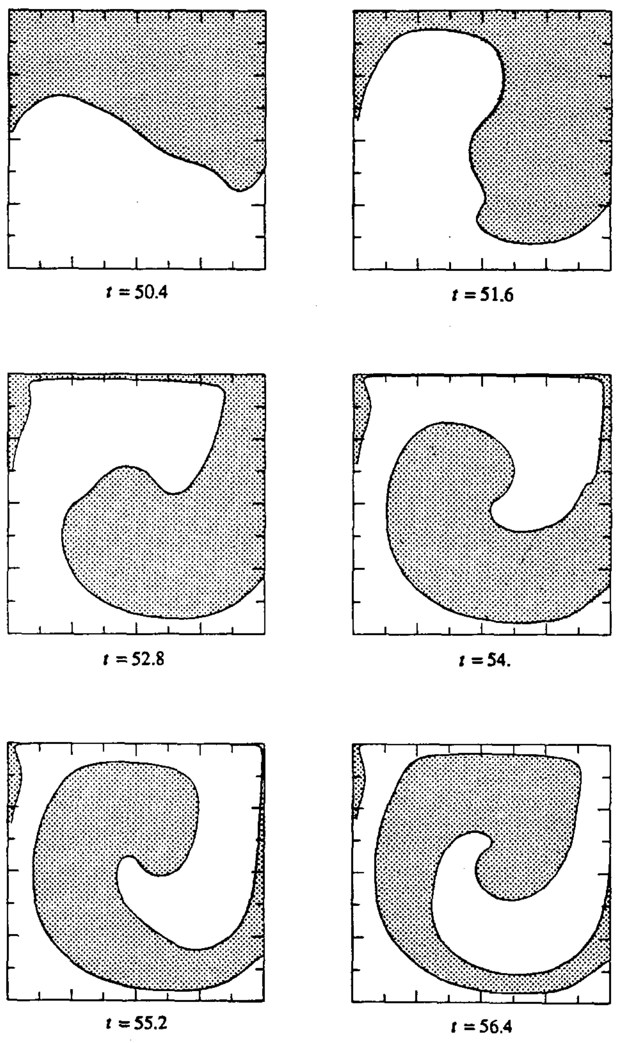

FIG. 3. Passive advection: $F(\kappa) = 0$.

and we simply use a first-order upwind difference scheme. For the hyperbolic part $S|\nabla\phi|$, as we mentioned in Section 2, a particularly simple numerical flux function is the Enquist–Osher scheme and we use it in our calculations. Given a burning speed $F=S-\varepsilon\kappa$, the parabolic part requires a much smaller time step than the one allowed by the fluid calculation. Therefore we introduce another time step $\tilde{\Delta}t$ inside the fluid calculation time step interval, used for the updating of ϕ within a single time step Δt, while the fluid velocity used remains the same for this time period calculation. We find that letting $\tilde{\Delta}t=\Delta x^2/4\varepsilon$ is adequate for this purpose. For the approximation of κ, we use central difference for both the first and second derivatives. To avoid numerical singularity when $|\nabla\phi|$ is close to zero, we smooth the expression for κ in (3.3), replacing $|\nabla\phi|^2$ by $|\nabla\phi|^2+\delta^2$ in the denominator where δ is some small positive constant.

There are two kinds of ignition: one is to ignite the fluid located in the ignition cell only once which we call "sparked ignition"; the other is to constantly ignite the cell which we call "constant ignition." The difference is that for "constant ignition," the ignition point stays burned for ever; while for "sparked ignition," this point is ignited only once. In terms of the algorithm, the "sparked ignition" is described by Eq. (2.12), and the "constant ignition" has to satisfy the extra condition that $\phi(\mathbf{x},t)\geqslant 0$ for $t\geqslant 0$, where $\mathbf{x}$ is the ignition point. If we denote the region where $\phi\geqslant 0$ is burned and $\phi<0$ is unburned, a first-order algorithm for "constant ignition" is the following: first we solve Eq. (2.12) in one time step by the first-order method as in the "sparked" algorithm and define $\tilde{\phi}^{n+1}$ to be the updated value; then

$$\phi^{n+1}=\max(\tilde{\phi}^{n+1},\phi^0). \tag{3.4}$$

In the problem of driven cavity, we take a 64×64 grid for both the fluid calculation and flame burning. In Figs. (1–5) we show the results with the driven cavity flow. In Figs. 1–2 we constantly ignite the bottom midpoint (0.5, 0.1). We run the fluid flow until $t=50$ to let the flow to be fully

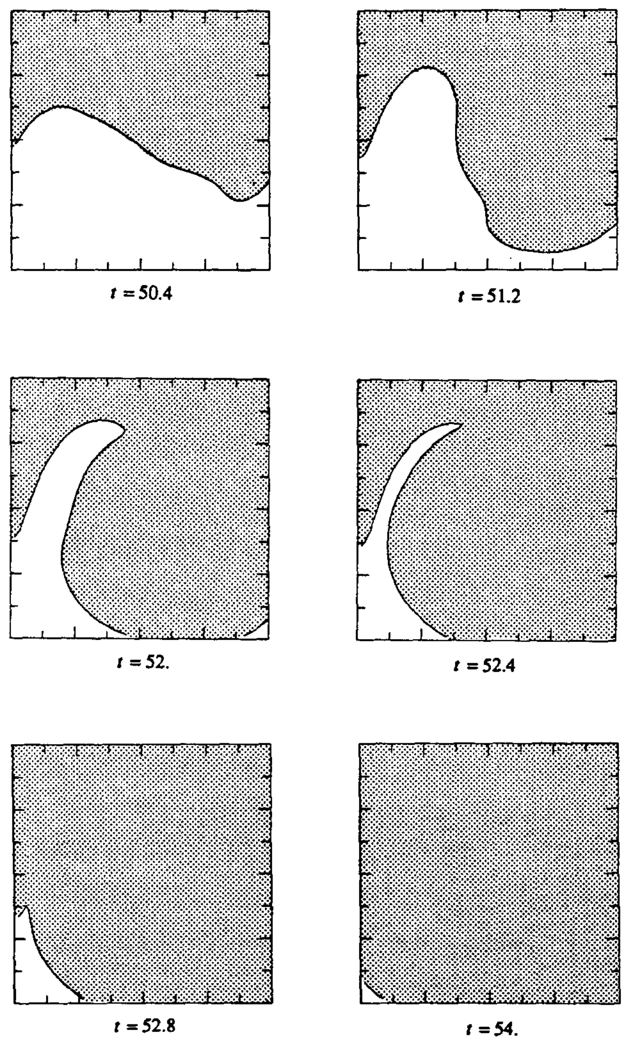

FIG. 4. Sparked ignition along the center line: $F(\kappa)=0.1$.

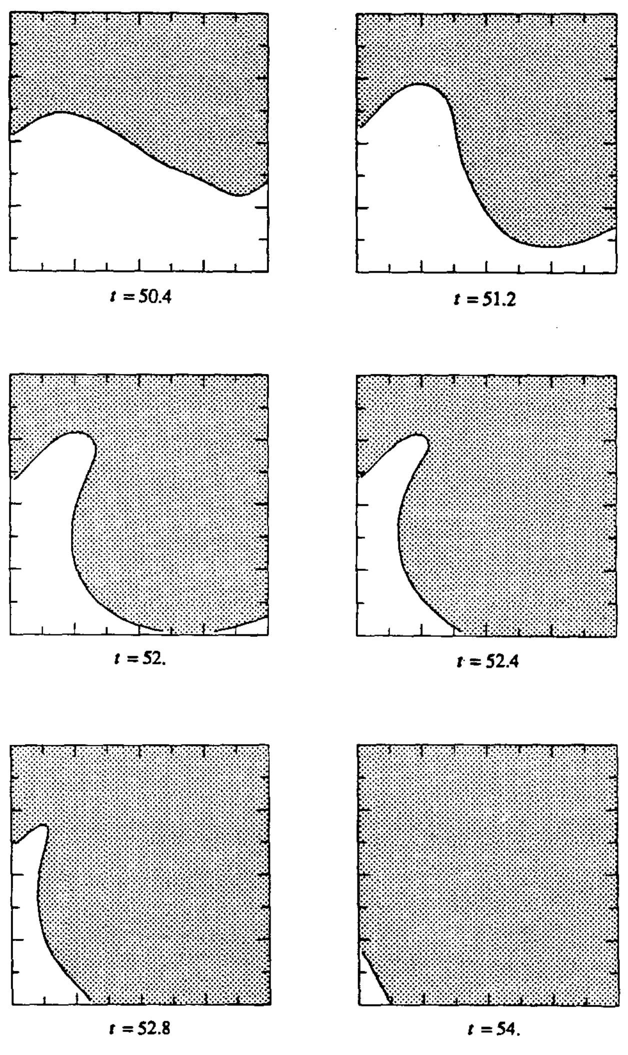

FIG. 5. Sparked ignition along the center line: $F(\kappa)=0.1-0.01\kappa$.

developed. To capture the fine detail of the flow we choose the burning speed $S = 0.01$. These calculations are similar to those in [13]. Figure 1 shows the burning flame at this constant burning speed. At $t = 64$, there is a sharp spike appearing, indicating that the flame is being dragged by the maximal velocity along the top. We also see the effect of the eddy around the lower left corner at this time and later at $t = 66$ and 68. At $t = 72$ the tip of the flame continues to be dragged toward the ignition area and, at $t = 74$, it has merged with the rest of the flame, pushing the flame toward the center. In Fig. 2 we include the curvature-dependence and let $\varepsilon = 0.01$. At $t \leqslant 64$ the spike has not formed as in the previous case. At $t = 66$, there is the spike, but not pushed as far as the case when $\varepsilon = 0$; at $t = 68$, the situation is reversed. This shows the role played by the curvature as the positive curvature slows down the burning speed and negative curvature speeds up the burning speed. For $t \leqslant 66$ the positive curvature around the tip leads the flame and, at the same time, creates a concave side of the flame front near the center of the box. The negative curvature starts to gain more weight after $t = 66$ and helps the flame moving towards the center. At $t = 68$, the tip moves further than that of the case $\varepsilon = 0$, mostly due to the spread of the flame from the side near the box center. The merging appears earlier in this case than in the previous case, since the flame is pushed more toward the center and the negative curvature speeds up the burning. At $t = 80$ we see that the unburned region around the center has disappeared.

In Figs. 3–5 we study the motions of an interface initially dividing the box. First (Fig. 3) we show the passive advection of this interface; thus $F = 0$. Then we ignite the upper part of the fluid only once and set the burning speed $S = 0.1$ (Fig. 4). The flame is approaching from the upper part to the lower part. For the part of the fluid on the right, the fluid velocity is in the same direction as the burning and the region is quickly burned. In the left part the situation is

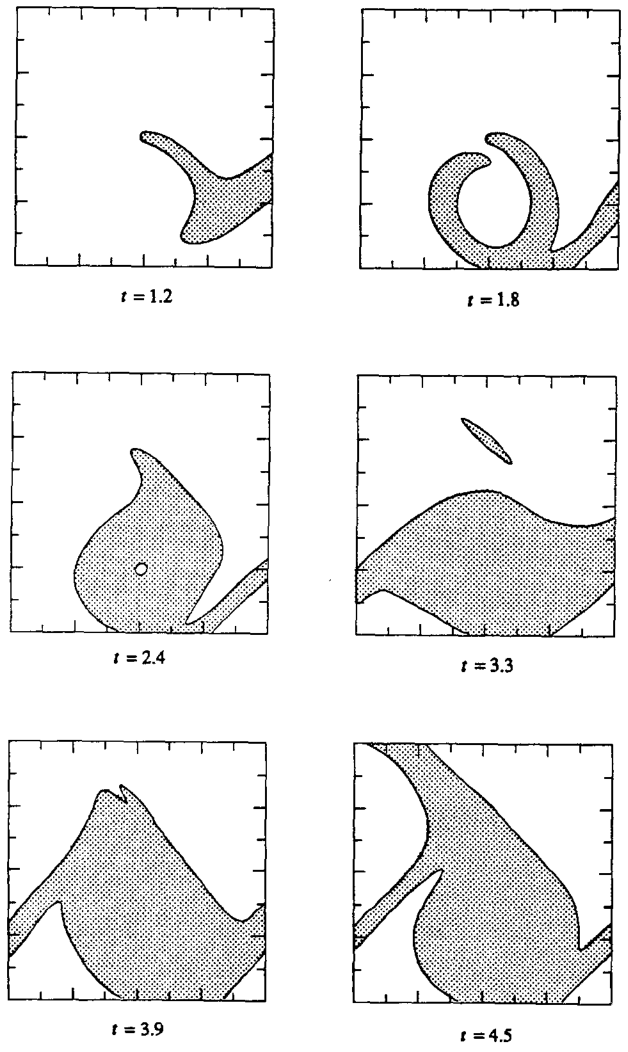

FIG. 6. Constant ignition at the center: $F(\kappa) = 0.2$.

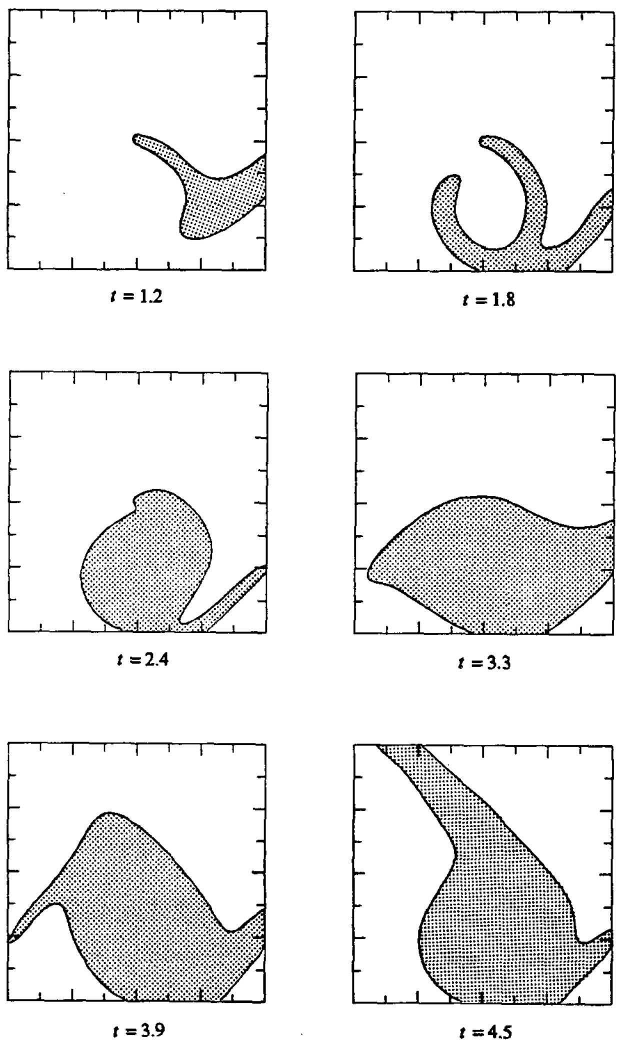

FIG. 7. Constant ignition at the center: $F(\kappa) = 0.2 - 0.01\kappa$.

reversed and there is a competing balance between the fluid motion and the burning. The sign of $S - \varepsilon\kappa - U \cdot \nabla\phi/|\nabla\phi|$ can help us determine the direction of burning, so that we can expect the relative flame front positions for different cases through the analysis of this term. The fluid velocity is at its maxima halfway between the center and the left boundary, so there is an unburned narrow region moving into the burned region by the fluid flow, but it is burned eventually. We compare the case $\varepsilon = 0$ to the case $\varepsilon = 0.01$ and see the shortening and smoothing of this narrow region in the latter case. This is caused by the curvature-dependence as we expected.

The final example is the doubly-periodic jet which results in two shear layers with opposite signs. Here we take a 128×128 grid for calculations. The burning speed $S = 0.2$. We ignite at the center constantly and, again, compare the case $\varepsilon = 0$ (Fig. 6) with the case $\varepsilon = 0.01$ (Fig. 7). For $t \leqslant 1.8$ there is not much difference. The flame moves to the right with the jet first and is pushed down by the formation of the fluid vortex later. At $t = 2.4$ we see that there is a small region surrounded by flame and a tip sticking up, in the case of no curvature-dependence (Fig. 6), while in the curvature-dependence case (Fig. 7) the flame has burned all the interior and the tip is much smoother. At $t = 3.3$ we find that the flame is split into two parts due to the strong shear layer in Fig. 6, but this does not happen in Fig. 7, since the flame does not reach that shear layer region. At $t = 3.9$ the flame branch to the left-lower becomes thinner and continues to reach out at $t = 4.5$ in Fig. 6. In Fig. 7, however, it is shrinking ($t = 3.9$) and finally disappears at $t = 4.5$.

We repeat the above calculations for a 256×256 grid and show the results in Figs. 8 and 9. It is obvious that the

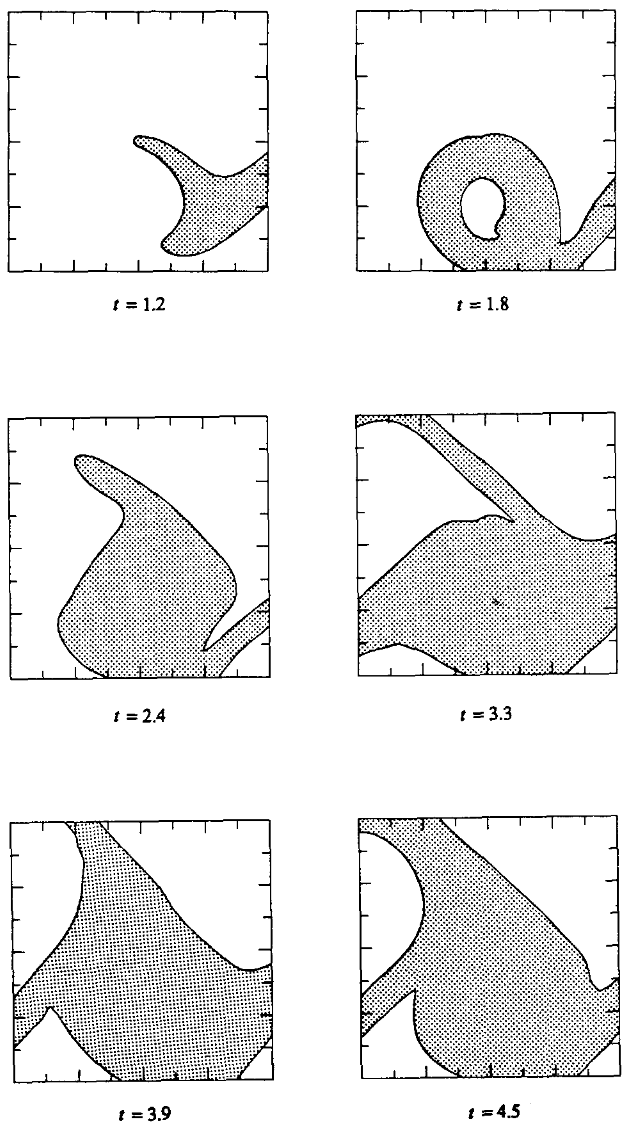

FIG. 8. Constant ignition at the center: $F(\kappa) = 0.2$.

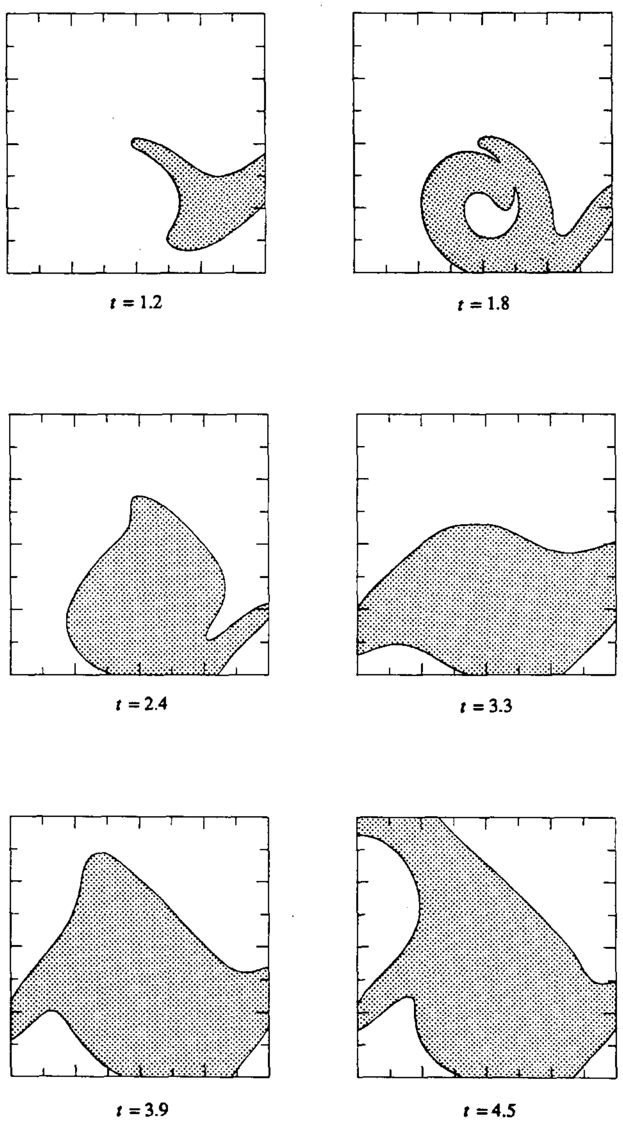

FIG. 9. Constant ignition at the center: $F(\kappa) = 0.2 - 0.01\kappa$.

burning rate is higher in the refined calculations than the coarse ones. For example, the tip of the flame is already merged into the ignition region at $t=1.8$ in the 256×256 case. The reason is the thinning of the shear layers in more refined calculations as shown in [1] and our fluid calculations. Across the shear layer, the velocity changes its direction, with the approximated magnitude 1 in both directions. As the shear layer becomes thinner, the transition becomes sharper and the magnitude of velocity reaches its maxima on both sides of the shear layer in shorter distances. In [1], Bell *et al.* showed that for a fixed time, the kinetic energy increases when calculations are refined. Therefore, with the increase of magnitude of the velocity, the flame is spread faster. This is what we see in the difference between these two calculations with different grid sizes.

To summarize, we have studied the cold flame burning in several fluid flows with strong vorticity dominance. Our interests are the interplays between the fluid velocity and flame burning and the effect of the curvature-dependence. We have shown the easy handling of corner formation of flames by the level set technique and, consistently, the smoothing effect by the curvature-dependence. The next stage of the research is to study the complete feedback system, with inclusion of volume expansion and baroclinic vorticity production along the flame fronts. We can also use second-order schemes [12] to improve the accuracy of the front position. Finally the methods described here can be easily extended to three-dimensional problems and therefore will generate many more interesting results.

ACKNOWLEDGMENTS

We thank A. Chorin, J. Bell, P. Colella, and P. Concus for helpful discussions. Calculations were performed on a CRAY-XMP at the Lawrence Livermore National Laboratory.

REFERENCES

1. J. Bell, P. Colella, and H. Glaz, *J. Comput. Phys.* **85**, 257 (1989).
2. J. Bell and D. Marcus, "A Second-Order Projection Method for Variable-Density Flows," UCRL-JC-104132, Lawrence Livermore National Laboratory Report, May 1990 (unpublished).
3. J. Bell and D. Marcus, "Vorticity Intensification and Transition to Turbulence in a Three-Dimensional Shear Flow," AIAA-91-1649, 22nd AIAA Fluid Dynamics, Plasma Dynamics and Lasers Conference, Honolulu, HI, June 24–27, 1991.
4. A. J. Chorin, *Math. Comput.* **22**, 745 (1968).
5. A. J. Chorin, *Math. Comput.* **23**, 341 (1969).
6. A. J. Chorin, *J. Comput. Phys.* **35**, 1 (1980).
7. P. Colella, *J. Comput. Phys.* **87**, 171 (1990).
8. B. Engquist and S. Osher, *Math. Comput.* **34**, 45 (1980).
9. L. C. Evans and J. Spruck, *J. Diff. Geom.* **33**, 635 (1991).
10. W. Mulder, S. J. Osher, and J. A. Sethian, *J. Comput. Phys.* **100**, 209 (1992).
11. S. J. Osher, *SIAM J. Num. Anal.* **21**, 217 (1984).
12. S. J. Osher and J. A. Sethian, *J. Comput. Phys.* **79**, 12 (1988).
13. J. A. Sethian, *J. Comput. Phys.* **54**, 425 (1984).
14. J. A. Sethian, *Commun. Math. Phys.* **101**, 487 (1985).
15. J. A. Sethian, *J. Diff. Geom.* **31**, 131 (1990).
16. J. A. Sethian and J. D. Strain, *J. Comput. Phys.* **98**, 231 (1992).
17. A. B. Stephens, J. B. Bell, J. M. Solomon, and L. B. Hackerman, *J. Comput. Phys.* **53**, 152 (1984).