Simulation and Animation of Fire and Other Natural Phenomena in the Visual Effects Industry

Duc Nguyen[1], Doug Enright[2] and Ron Fedkiw[3]

[1] Computer Science Department, Stanford University, *dqnguyen@stanford.edu*
[2] Mathematics Department, University of California, Los Angeles, *enright@math.ucla.edu*
[3] Computer Science Department, Stanford University, *fedkiw@cs.stanford.edu*

1 Abstract

In this article, we give a brief overview of methods that are currently used to simulate and animate fire and other natural phenomena for the special effects industry. First, we discuss the use of the ghost fluid method to treat thin flame fronts. Then, we illustrate the use of this method to model and animate fire for computer graphics purposes. Next, we discuss the use of the one phase, two-dimensional Euler equations to model highly detailed large scale phenomena such as the nuclear explosions seen in the movie "Terminator 3: Rise of the Machines". We also briefly touch upon the use of more traditional (in a computer graphics sense) particle-based techniques for modeling flames and explosions. Finally, we close the article with the particle level set method and its application to the animation and rendering of complex water surfaces.

2 Introduction

The modeling of natural phenomena remains a challenging problem in the visual effects community. Natural looking animation of "fluid-like" behavior, e.g. smoke, water, and fire, is highly desirable and hard to achieve. Virtual fire effects are desirable due to the dangerous nature of this phenomenon. The complexity of the motion exhibited by these natural phenomena defies the ability of animators to produce realistic animations by hand. The ever increasing use of computer animation in feature films to create photorealistic effects has motivated researchers in computer graphics (CG) to examine the extensive computational fluid dynamics (CFD) literature for algorithms that can be adapted for use in an animation environment.

Researchers have used numerical techniques to solve partial differential equations describing physical phenomena for many years. Consider multi-phase incompressible flow including the effects of viscosity, surface tension and gravity. Any numerical approach to this problem needs both a method for tracking (or capturing) the interface location as well as a method for enforcing the appropriate boundary conditions at the interface. See [46, 2, 44]

for numerical methods that used front tracking, volume of fluid and level set methods respectively for modeling the location of the interface. These methods all extend the δ-function formulation of [33] to treat the jump conditions present in multiphase incompressible flow. A drawback of the δ-function formulation is that it regularizes physically discontinuous quantities at the interface, producing a numerically continuous profile for the density, viscosity and pressure. This numerical smearing can be problematic, e.g a continuous pressure profile does not adequately model surface tension forces, prompting the use of additional source terms in order to numerically model these forces. An alternate strategy for enforcing the interface boundary conditions is based upon the ghost fluid method (GFM) of Fedkiw *et al.* [12]. Discontinuities are implicitly enforced with the GFM, avoiding any numerical smoothing of discontinuous quantities across the interface. The ghost fluid method and related techniques have been used to model discontinuities in compressible and incompressible flows [12, 23, 3], flames and detonations [29, 13], solid fluid coupling [11] and Stefan problems [18, 17, 4]. A newly proposed, fully conservative ghost fluid method has been used to track material interfaces, inert shocks and detonation waves [27].

For multiphase incompressible flow, the interface moves with local fluid velocity only and individual fluid particles do not cross the interface. However, when we consider interfaces where a chemical reaction is taking place, the interface moves with the local unreacted fluid velocity plus a reaction term that accounts for the conversion of one fluid into the other. As an example, consider combustion in premixed flames. Assuming that the flame front is infinitely thin allows one to treat the flame front as a discontinuity separating two incompressible flows. The unreacted material undergoes reaction as it crosses the interface, producing a lower density (larger volume) reacted material. The material must instantaneously expand as it crosses the interface, implying that the normal velocity is discontinuous across the interface in addition to the discontinuities in the density, viscosity and pressure. This discontinuity in the normal velocity can also be seen when a material undergoes a phase transition at the interface. The methods in [22, 34, 39, 47] are all based on a δ-function formulation and thus incorrectly force the discontinuous velocity field to be continuous. This smearing can be quite problematic since it adds a compressible character to the flow field near the interface, i.e. the divergence free condition is not exactly satisfied in each separate subdomain. Partial solutions to these problems where proposed in [20] where the authors were able to remove the numerical smearing of the normal velocity, thus obtaining a sharp interface profile. However, the method in [20] is excessively complicated and cannot support merging of discontinuities or interfaces with high curvature. In [29], Nguyen *et al.* combined the ghost fluid method and the irregular boundary Poisson pressure treatment of [25] to treat two phase incompressible flow with a phase change at the interface. This new approach maintains a sharp interface representation similar to [20], but is simpler to

implement and readily treats both regions of high curvature and topological changes in the interface. The level set method [31] was used to capture the location of the interface.

Accurate modeling of the motion of a contact discontinuity itself, for incompressible flows, has been a challenge for level set methods. Recently, a new method, the "particle level set method" [7], has been used to accurately track contact discontinuities for incompressible flows [9]. The particle level set method conserves mass to an accuracy comparable to explicit front tracking and volume of fluid methods while maintaining the flexibility and ease of use of the original level set method.

Due to the robustness and ease of programming of these interface methods in three spatial dimensions combined with the ever increasing speed and memory of desktop computers, visual effects companies have used physics-based animation algorithms to model fire and water. These methods also produce realistic looking behavior on the coarse computational grids commonly used in a production animation environment.

3 Low-Speed Flames

3.1 Governing Equations

We ignore viscous effects and consider the equations for inviscid incompressible flow,

$$\mathbf{V}_t + (\mathbf{V} \cdot \nabla)\,\mathbf{V} + \frac{\nabla p}{\rho} = 0, \tag{1}$$

where equation 1 holds independently in each fluid. In addition, we assume that $\nabla \cdot \mathbf{V} = 0$ in each fluid, including near the interface. The interface velocity is $\mathbf{W} = D\mathbf{N}$, where D is the normal component of the interface velocity defined by $D = (V_N)_u + S$ with the "u" subscript indicating that the normal velocity is calculated using the velocity of the unreacted material only. This is important to note since V_N is discontinuous across the interface. A curvature dependent flame speed is assumed, i.e. $S = S_o + \sigma\kappa$ where S_o and σ are constants and κ is the local curvature of the interface.

Conservation of mass and momentum imply the standard Rankine-Hugoniot jump conditions across the interface,

$$[\rho(V_N - D)] = 0 \tag{2}$$

$$\left[\rho(V_N - D)^2 + p\right] = 0 \tag{3}$$

as well as the continuity of the tangential velocities, $[V_{T_1}] = [V_{T_2}] = 0$, assuming that $S \neq 0$. Denoting the mass flux in a reference frame moving with speed D by

$$M = \rho_r \left((V_N)_r - D \right) = \rho_u \left((V_N)_u - D \right) \tag{4}$$

allows us to rewrite equation 2 as $[M] = 0$. The "r" subscript denotes a reacted material quantity. Substitution of $D = (V_N)_u + S$ into equation 4 yields $M = -\rho_u S$.

Equation 3 can be rewritten as

$$\left[\frac{M^2}{\rho} + p \right] = 0 \tag{5}$$

or as

$$[p] = -M^2 \left[\frac{1}{\rho} \right] \tag{6}$$

using $[M] = 0$.

Starting with $[D] = 0$,

$$\left[\frac{\rho V_N - \rho(V_N - D)}{\rho} \right] = 0 \tag{7}$$

$$\left[\frac{\rho V_N - M}{\rho} \right] = 0 \tag{8}$$

and

$$[V_N] = M \left[\frac{1}{\rho} \right] \tag{9}$$

where the last equation follows since $[M] = 0$. It is often more convenient to write

$$[\mathbf{V}] = M \left[\frac{1}{\rho} \right] \mathbf{N} \tag{10}$$

using the fact that $[V_{T_1}] = [V_{T_2}] = 0$.

3.2 Level Set Equation

The level set equation

$$\phi_t + \mathbf{W} \cdot \nabla \phi = 0 \tag{11}$$

is used to keep track of the interface location as the set of points where $\phi = 0$. The unreacted and reacted materials are then designated by the points where $\phi > 0$ and $\phi \leq 0$ respectively. To keep the values of ϕ close to those of a signed distance function, i.e. $|\nabla \phi| = 1$, the reinitialization equation

$$\phi_\tau + S(\phi_o)\left(|\nabla\phi| - 1\right) = 0 \tag{12}$$

is iterated for a few steps in fictitious time, τ. The level set function is used to compute the normal

$$\mathbf{N} = \frac{\nabla\phi}{|\nabla\phi|} \tag{13}$$

and the curvature

$$\kappa = -\nabla \cdot \mathbf{N} \tag{14}$$

in a standard fashion. For more details on the level set function see the recently published book by Osher and Fedkiw [30].

3.3 Projection Method

First, $\mathbf{V}^\star = \langle u^\star, v^\star, w^\star \rangle$ is defined by

$$\frac{\mathbf{V}^\star - \mathbf{V}^n}{\triangle t} + (\mathbf{V} \cdot \nabla)\,\mathbf{V} = 0 \tag{15}$$

and then the velocity field at the new time step, $\mathbf{V}^{n+1} = \left\langle u^{n+1}, v^{n+1}, w^{n+1} \right\rangle$, is defined by

$$\frac{\mathbf{V}^{n+1} - \mathbf{V}^\star}{\triangle t} + \frac{\nabla p}{\rho} = 0 \tag{16}$$

so that by combining equations 15 and 16 to eliminate $\mathbf{V}^\star$ results in a velocity field which satisfies equation 1. Taking the divergence of equation 16 gives

$$\nabla \cdot \left(\frac{\nabla p}{\rho}\right) = \frac{\nabla \cdot \mathbf{V}^\star}{\triangle t} \tag{17}$$

after setting $\nabla \cdot \mathbf{V}^{n+1}$ to zero. Equations 16 and 17 can be rewritten as

$$\mathbf{V}^{n+1} - \mathbf{V}^\star + \frac{\nabla p^\star}{\rho} = 0 \tag{18}$$

and

$$\nabla \cdot \left(\frac{\nabla p^\star}{\rho}\right) = \nabla \cdot \mathbf{V}^\star \tag{19}$$

eliminating their dependence on $\triangle t$ by using a scaled pressure, $p^\star = p\triangle t$. See [5] for more details.

3.4 Treating the Jump Conditions

Since the normal velocity is discontinuous across the interface, one has to use caution when applying numerical discretizations near the interface. For example, when discretizing the unreacted fluid velocity near the interface, one should avoid using values of the reacted fluid velocity (and vice versa). Following the ghost fluid methodology, a band of ghost cells on the reacted side of the interface is populated with unreacted ghost velocities that can be used in the discretization of the unreacted fluid velocity. Similarly, reacted ghost velocities are defined on a band of ghost cells on the unreacted side of the interface and used in the discretization of the reacted fluid velocity. This is done using equation 10 to obtain

$$u_u^G = u_r - M\left(\frac{1}{\rho_r} - \frac{1}{\rho_u}\right) n_1 \tag{20}$$

$$v_u^G = v_r - M\left(\frac{1}{\rho_r} - \frac{1}{\rho_u}\right) n_2 \tag{21}$$

and

$$w_u^G = w_r - M\left(\frac{1}{\rho_r} - \frac{1}{\rho_u}\right) n_3 \tag{22}$$

where $\mathbf{N} = (n_1, n_2, n_3)$ is the local unit normal. n_1, n_2 and n_3 are computed at the appropriate grid locations using simple averaging, e.g. $(n_1)_{i+\frac{1}{2},j,k} = \frac{(n_1)_{i,j,k}+(n_1)_{i+1,j,k}}{2}$. Similarly, reacted ghost velocities are calculated at unreacted grid locations using

$$u_r^G = u_u + M\left(\frac{1}{\rho_r} - \frac{1}{\rho_u}\right) n_1 \tag{23}$$

$$v_r^G = v_u + M\left(\frac{1}{\rho_r} - \frac{1}{\rho_u}\right) n_2 \tag{24}$$

and

$$w_r^G = w_u + M\left(\frac{1}{\rho_r} - \frac{1}{\rho_u}\right) n_3. \tag{25}$$

When solving equation 19, the jump in pressure given by equation 6 as

$$[p^*] = -\Delta t M^2 \left(\frac{1}{\rho_r} - \frac{1}{\rho_u}\right) \tag{26}$$

is accounted for using the techniques developed in [25, 23].

3.5 Examples

Figure 1 shows the computed velocity for merging flame fronts and illustrates the ability of our algorithm to treat merging in one spatial dimension. After merging, the domain contains a single phase incompressible fluid which must have a constant velocity. Figure 2 shows the time evolution of two initially circular flame fronts as they grow to merge together. Figure 3 shows a snapshot of the velocity field, illustrating its discontinuous nature across the interface. Figure 4 shows the simulation of a flame vortex interaction. As the flame moves downward and is distorted by the initial vortex, secondary vortices are generated behind the flame front. Notice that part of the vortex has passed through the flame front and part of it can still be seen just below the flame front. In figure 5, we show the grid refinement study of the flame vortex interaction. Note the first-order accurate convergence for the flame location. Figure 6 shows the time evolution of two initially spherical flame fronts as they grow and merge together.

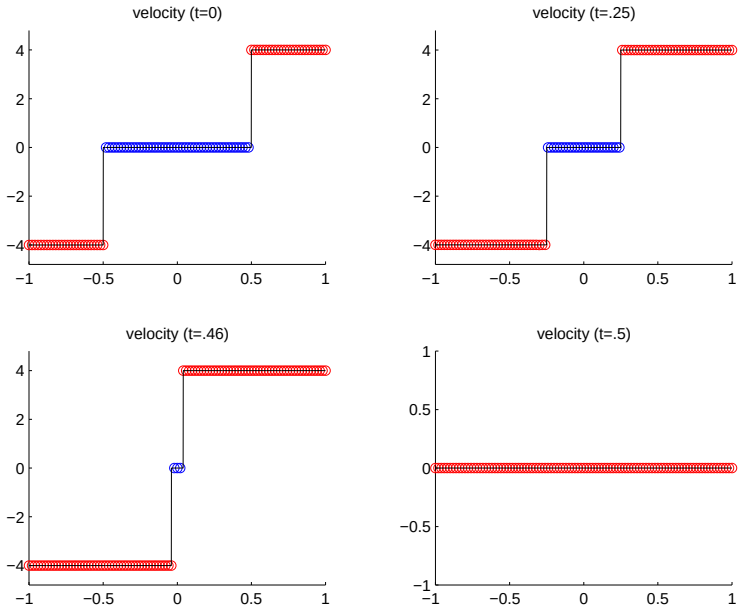


Fig. 1. Merging flames in one spatial dimension.

4 Fire for Computer Graphics

Recently, this work on two-phase incompressible flame discontinuities [29] was extended to simulate and animate fire for computer graphics [28]. It is

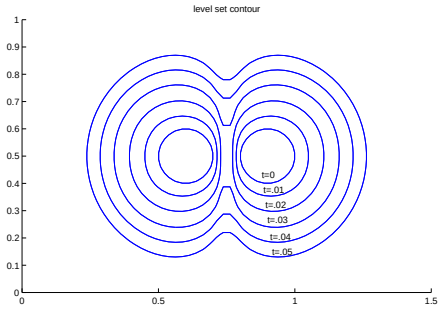


Fig. 2. Time evolution of two initially circular flame fronts as they grow to merge together.

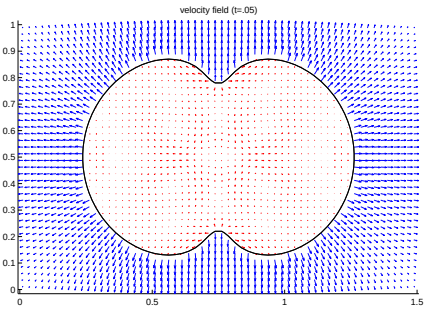


Fig. 3. Discontinuous velocity field depicted shortly after the two flame fronts merge.

suitable for both smooth (laminar) and turbulent flames and it can be used to animate the burning of both solid or gaseous fuels. The incompressible Euler equations are used to independently model both the vaporized fuel and the hot gaseous products. A physically based model is used for the expansion that takes place when a vaporized fuel reacts to form hot gaseous products, and a related model is used for the similar expansion that takes place when a solid fuel is vaporized into a gaseous state. In addition to the hot gaseous products, smoke and soot rise under the influence of buoyancy and are rendered using a blackbody radiation model. The method also models and renders the blue core that results from the free radicals in the chemical reaction zone where fuel is converted into products. The method also allows for the flame front and smoke to interact with objects, with flammable objects able to catch on fire.

Buoyancy forces were added as body forcing terms to equation 1 to model rising flames, i.e.

$$\mathbf{V}_t + (\mathbf{V} \cdot \nabla)\,\mathbf{V} + \frac{\nabla p}{\rho} = \mathbf{f}. \tag{27}$$

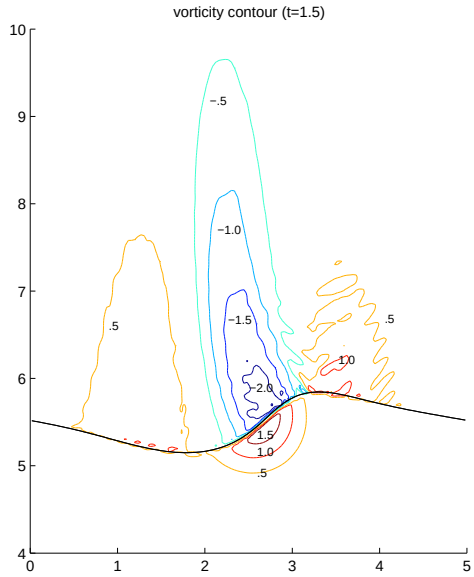


Fig. 4. Flame vortex interaction - secondary vorticity generation.

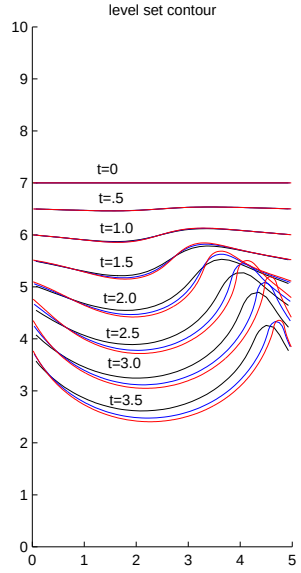


Fig. 5. Flame vortex interaction - grid refinement.

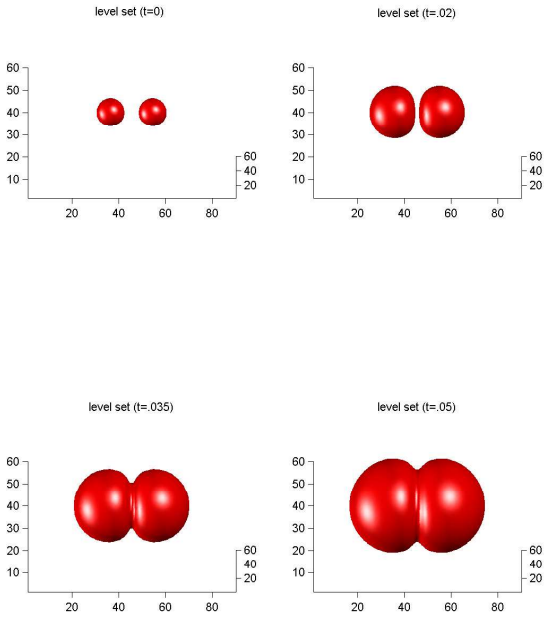


Fig. 6. Time evolution of two initially spherical flame fronts as they grow to merge together.

Additional transport equations for the temperature and the density of soot resulting from the chemical reaction at the flame front need to be modeled in order to obtain the necessary information for the visualization of fire. The smoke density ρ of flame is passively advected with fluid velocity, i.e.

$$\rho_t + (\mathbf{V} \cdot \nabla)\, \rho = 0, \tag{28}$$

and the behavior of the temperature field around flame front can be described as a combination of advection and radiation transport as

$$T_t + (\mathbf{V} \cdot \nabla)\, T = -c_T \left(\frac{T - T_{air}}{T_{max} - T_{air}} \right)^4. \tag{29}$$

T is the temperature, and T_{max} and T_{air} are the maximum fire temperature and the surrounding air temperature respectively.

The projection method described in section 3.3 is used to update equation 27. Unconditional stability of a numerical scheme is important for its use in an animation environment, leading the CG community to adopt a semi-Lagrangian approach [6, 42, 40] to discretize the convective term in equations 27, 28, and 29. The use of a semi-Lagrangian method may introduce large amounts of numerical dissipation, especially on the coarse computational grids commonly used in the graphics community. The method

of choice to reduce this dissipation is the "vorticity confinement" method of Steinhoff [43]. This method was first introduced to the computer graphics community in [10]. The vorticity confinement body force adds additional small scale rolling features characteristic of fire and smoke in a numerically consistent manner. This is accomplished by adding a swirling body force in regions of the flow which posses large amounts of vorticity and are thus sensitive to excessive amounts of artificial dissipation.

The first step in generating the small scale detail is to identify where it comes from. In incompressible flow, the vorticity, $\boldsymbol{\omega} = \nabla \times \mathbf{V}$, provides the small scale structure. Each small piece of vorticity can be thought of as a paddle wheel trying to spin the flow field in a particular direction. Artificial numerical dissipation damps out the effect of these paddle wheels, and the key idea is to simply add it back. First normalized vorticity location vectors $\mathbf{N} = \boldsymbol{\eta}/|\boldsymbol{\eta}|$, where $\boldsymbol{\eta} = \nabla|\boldsymbol{\omega}|$, that point from lower vorticity concentrations to higher vorticity concentrations are computed. Then the magnitude and direction of the paddle wheel force is computed as $\mathbf{f}_{conf} = \epsilon h(\mathbf{N} \times \boldsymbol{\omega})$ where $\epsilon > 0$ is used to control the amount of small scale detail added back into the flow field, and the dependence on the spatial discretization h guarantees that as the mesh is refined the physically correct solution is still obtained. Figure 7 is a simple demonstration of smoke rising. Notice how the vortices are preserved in the smoke when using a vorticity confinement technique.

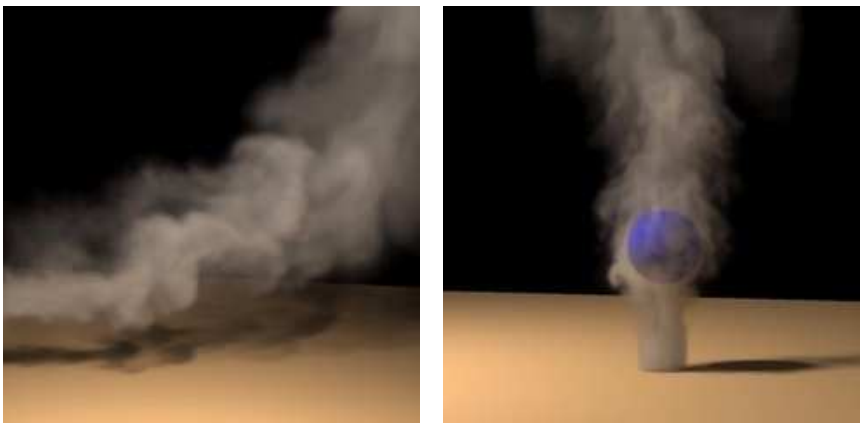


Fig. 7. Rising smoke computed using the vorticity confinement method.

An important part of the visual appearance of fire is the expansion of the gas as it undergoes a transformation from unburnt fuel into hot gaseous products. The Rankine-Hugoniot jump conditions at the interface naturally capture this outward expansion of the gas that is next to impossible to achieve

using the low level hacks and random numbers usually resorted to by the computer graphics animation community.

Figure 8 (left) shows two different blue cores with a differing flame speed S. The zero level set is used to represent this blue core with the unreacted gas inside the blue core. Figure 8 (right) shows three different flames with different density ratios of unreacted and reacted gases. This density ratio increases from left to right, making the flame appear fuller and more turbulent. Figure 9 shows the ability of our model to interact with objects. In this example, a flammable ball flies through the fire and catches on fire. Figure 10 illustrates the use of a turbulent flame model to simulate a flamethrower. Finally, figure 11 shows two burning logs that are placed on the ground and used to emit fuel. The crossways log on the top is not lit so the flame is forced to flow around it.

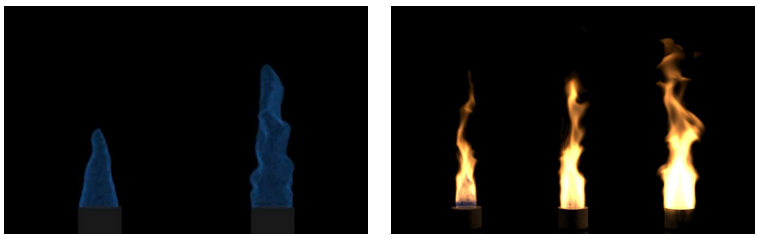


Fig. 8. Blue reaction zone cores (left) and CG fire (right).

5 Smoke Simulation for Large Scale Phenomena

Large scale phenomena such as nuclear explosions are challenging to animate realistically. For example, it is difficult to simulate the nuclear destruction of an entire city with the level of detail necessary for a feature film. Moreover, reference video footage tends to be of low quality and resolution, so computer simulations of these phenomena are preferable. The previously discussed methods can produce near real time results on small grids and interactive results on moderate sized grids. However, on very large grids of the scale $2000 \times 2000 \times 2000$, these methods are impractical. In fact, a grid of this size would require about 120GB of memory just to store the density and velocity field as floats (and twice that much for doubles), which is well beyond the capability of a high end workstation. Besides these storage limitations, a fully three dimensional compressible flow simulation of this phenomenon is not suitable for use in a production environment due to the small time step requirement resulting from the large sound speed in the compressible equations.

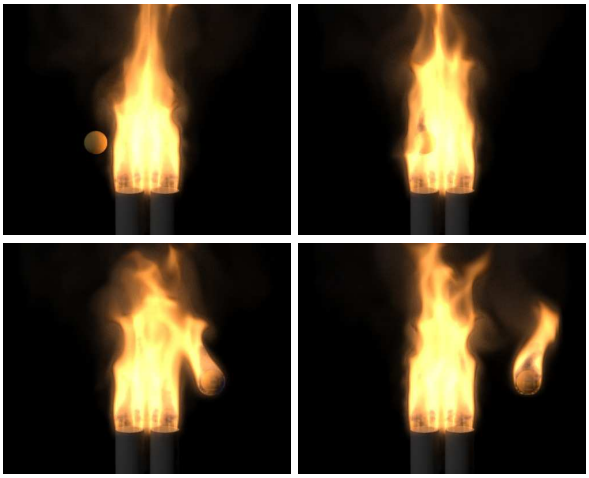


Fig. 9. A flammable ball passes through a gas flame and catches on fire.

Fig. 10. A turbulent gas flame model of a flamethrower.

As an alternative approach which avoids these difficulties, in [35], an efficient method for simulating highly detailed large scale participating media such as the nuclear explosions shown in figure 12 is presented. This effect was accomplished by simulating the motion of particles in a fluid dynamics generated velocity field. The key idea is the combination of a few high resolution two-dimensional physically based flow fields with a moderate sized three-dimensional Kolmogorov velocity field tiled periodically throughout space. Thus to obtain results on the scale of $2000 \times 2000 \times 2000$, we only need to simulate a few 2000×2000 grids saving about a factor of 2000 in both simulation time and memory. This computational savings is important in a production environment.

The velocity of the smoke was modeled with the one phase two-dimensional incompressible Euler equations as in equation 27. In contrast with the fire simulation, we don't need to worry about the jump conditions when we solve these equations. As in section 4, a semi-Lagrangian method is used to discretize the convection term in equation 27. We also passively advect temperature and density through the flow field. After generating a few two-

Fig. 11. A CG campfire.

dimensional velocity fields, we define a three-dimensional velocity field via interpolation. Parallel and cylindrical interpolations give nice results, however the method is not limited to these two interpolation schemes.

Since we build a three-dimensional velocity field from multiple two-dimensional solutions of Euler equations, it is desirable to add a fully three-dimensional component to the velocity field. This is accomplished using a Kolmogorov spectrum which is described in detail in [41] and used in many applications, e.g. to model fire in [24]. The main idea is to use random numbers to construct an energy spectrum in Fourier space that subsequently determines the structure of the velocity field. After constructing an energy spectrum in Fourier space, one enforces the divergence free condition and uses an inverse FFT to obtain a velocity field full of small scale eddies. Since the velocity field is periodic, a single grid can be used as a tiling of all of space. Moreover, one can use two grids of different sizes to increase the period of repetition to the least common multiple of their lengths alleviating visually troublesome spatial repetition (although this is a minor point for us since we blend the Kolmogorov velocity field with the non-periodic CFD/interpolation generated velocity field). We also fill the time domain by constructing a few Kolmogorov velocity fields, assigning each one to a different point in time, and interpolating between them at intermediate times. In practice, two spectrums are usually enough and we alternate between them every 24 frames (24 frames equals one second for film). Finally, at any point in space and time, we define the total velocity field as a linear combination of the Kolmogorov

Fig. 12. A CG Nuclear Explosion.

velocity field and the CFD/interpolation generated velocity field. Again, we stress that we do not use this to construct a three-dimensional velocity field, but instead compute the velocity at a point in space and time on the fly using the Kolmogorov velocity field, the two-dimensional CFD generated velocity fields, and chosen interpolation rules.

Once we have implicitly defined our flow field at every point of interest in space and time, we can passively advect particles through the flow using $\mathbf{x}_t = \mathbf{u}$ where $\mathbf{x}$ is the particle position for the rendering purpose. The particle attributes include temperature, density of smoke, velocity and position. If desired, copies of two-dimensional flow fields and the three-dimensional Kolmogorov velocity field can be distributed to multiple processors where particles can be passively evolved with no intercommunication requirements. This allows one to generate an incredibly large numbers of particles, although we have found that even one processor can readily generate enough particles to move the bottleneck to the rendering stage.

6 Alternative Techniques

Particle-based techniques have been quite popular for the animation of fire. Starting with the early use of a particle system for a visual effects shot in the Genesis sequence in "Star Trek II: The Wrath of Khan" [36], particle systems continue to provide a simple, yet quite effective way to model "fuzzy"

natural phenomena, including smoke and fire. We briefly review two recently proposed particle-based methods to model fire for computer graphics purposes. Feldman *et al.* [14] coupled a particle system with an appropriately modified one-phase Navier-Stokes solver for the purpose of animating suspended particle explosions. Lamorlette and Foster [24], focused on developing a method which provides a great degree of user control over the flame behavior, including imposing non-physically based behavior, while still preserving the essentially "chaotic" nature of fire.

In [14], the authors used a one phase incompressible fluid model. However, instead of using a divergence free condition for incompressible flow, the typical divergence free condition was modified to model the expansion of gas after a chemical reaction happens. Particles are used to keep track of the motion of particulate fuel and soot, and interact with the underlying fluid system through the transfer of momentum and heat.

Specifically, a single phase three-dimensional incompressible Euler equation (as in equation 27) is used to model the hot gas and products combined. The divergence free condition is modified as

$$\nabla \cdot \mathbf{V} = \phi \tag{30}$$

where ϕ is zero except where the detonation process and subsequent expansion is taking place. In order to enforce equation 30, a modified version of equation 19 is solved, i.e.

$$\triangle p^{\star} = \rho \left(\nabla \cdot \mathbf{V}^{\star} - \phi \right), \tag{31}$$

to determine the pressure field. Note that the density is spatially constant and has been moved to the right hand side. The fluid temperature is described as

$$T_t + (\mathbf{V} \cdot \nabla) T = -c_r \left(\frac{T - T_a}{T_{max} - T_a} \right)^4 + c_k \nabla^2 T + \frac{1}{\rho c_v} H_t \tag{32}$$

where T_a is the ambient temperature, and H is the heat energy transferred to the fluid from the reacting particles.

A particle model is used to represent the fuel and solid combustion products (i.e. soot). A particle possesses attributes including its position, velocity, mass, temperature, thermal mass, volume, and type. The particle is governed by

$$\mathbf{x}_{tt} = \frac{\mathbf{f}}{m} \quad \text{and} \quad Y_t = \frac{H_t}{c_m}, \tag{33}$$

where $\mathbf{x}$ denotes the location of the particle, H is the heat energy transferred to the particle or generated by combustion, Y is the particle temperature, and c_m is the particle's thermal mass.

The underlying fluid and the particles interact through the transfer of momentum and heat energy. When the particle moves through the fluid, a constant coefficient drag force law is imposed. This force is given by

$\mathbf{f} = \alpha_d r^2(\bar{\mathbf{u}} - \mathbf{x_t}) \parallel \bar{\mathbf{u}} - \mathbf{x_t} \parallel$, where α_d is the drag coefficient, r is the particle radius, and $\bar{\mathbf{u}}$ is the fluid velocity interpolated to the location of the particle. Similarly, $H_t = \alpha_h r^2(\bar{T} - Y)$ governs the thermal transfer between the particles and the fluid. A fuel particle will ignite when its temperature rises above an ignition point. The burning particles generate heat at the rate $H_t = b_h z$, where b_h denotes the amount of heat released per unit combusted mass of the fuel and z is the burning rate. While a particle is burning, the gaseous products produced from the combustion process are accounted for in the fluid by modifying the ϕ value of the cell in which the particle resides, i.e. $\Delta\phi = \frac{1}{V} b_g z$, where V is the volume of the cell and b_g denotes the volume of gas released per unit combusted mass less the volume of gas consumed.

While this technique is effective in providing a physically plausible model to capture the visual after-effects of an explosive process while avoiding the numerical difficulties and small time steps associated with a fully featured grid-based compressible flow model [49], the flame (explosion) is discretely defined by particles resulting in a spatially incoherent front. In the case of exploding fireballs, the lack of a spatially coherent interface is visually acceptable, but as indicated by the authors in [14], their method has difficulty in dealing with flame fronts with a higher degree of spatial structure such as seen in burning liquid fuels. The use of a spatially coherent interface technique, e.g. a level set method, avoids this visually disturbing feature as illustrated by the simulation of a flamethrower using the method of Nguyen *et al.* in figure 10.

Lamorlette and Foster [24] noted that fire can be used in feature films as a dramatic element, thus requiring the maximum level of control while at the same time maintaining a believable appearance. An example of the use of fire in this manner is the stylized fire-breathing dragon in the movie "Shrek". To provide the degree of control required to model fire-breathing dragons, Lamorlette and Foster proposed a particle-based fire animation tool utilizing many of the classical computer graphics techniques for simulating natural phenomena, including stochastic methods, procedural noise, Kolmogorov turbulent noise, artificial wind fields, and flame control curves.

In [24], the basic structural element of the flame is given by particles connected together into an interpolating B-spline curve. The particles are affected by arbitrary wind fields, Brownian diffusion, the motion of the fire source (with velocity $\mathbf{V}_p$), and thermal buoyancy. After defining these various flame behaviors, the motion of a given interpolating particle with temperature T_p is given by

$$\frac{d\mathbf{x}_p}{dt} = w(\mathbf{x}_p, t) + d(T_p) + \mathbf{V}_p + c(T_p, t; t_p), \tag{34}$$

where $w(\mathbf{x}_p, t)$ is the controlling wind field, $d(T_p)$ is the amount of random Brownian motion modeling diffusive processes, and $c(T_p, t; t_p)$ represents the thermal buoyancy with t_p indicating a functional dependence on the overall lifetime of the particle. To avoid modeling a temperature field over the entire

domain, the thermal buoyancy is made to depend upon the lifetime of the particle, i.e. $c(T_p, t; t_p) = -\beta g(T_o - T_p)t_p^2$, with T_o the ambient temperature and β the coefficient of thermal expansion. An explicit Runge-Kutta method is used to advance equation 34 in time. The spline is resampled after each time step with the particles redistributed to ensure no aliasing features are present. Individual flame strands are allowed to pinch off after reaching a set length at a rate given by a statistical model. The size of the separated flame is determined according to a normal distribution. An ad hoc estimate of the amount of fuel left in a separated flame is made in order to control how long it remains visible.

To provide visual fullness, a model of the actual visible flame region is necessary. A volumetric model is used, with the volume defined by a shape offset volume-of-revolution technique. A simple, two-dimensional profile is chosen from a library of shapes and rotated about the B-spline based particle backbone of the flame. A volumetric $1/r^2$ light density function motivated by the combustion process actually occurring at the flame front is then defined according to the position of the swept out parametric surface. This density function is statistically point sampled to easily allow for additional deformation and noise fields to be added. To provide for additional detail which is missing from the simple volumetric flame model discussed above, two levels of structural noise are added to the model. The first models fluctuations in the combustion process at the base of the flame. These fluctuations propagate up the structural backbone of the flame, causing the volumetrically sampled points to be displaced. Perlin's flow noise model [32] is used to determine the size of the fluctuations. For a finer, more turbulence driven perturbation of the sample points, Kolmogorov noise is added. The volumetric description of the flame readily allows for merging of neighboring flames, as in the level set method. After performing all of these operations, the particles are rendered. The color of each volumetric flame particle is not physically based, rather it is taken from a reference photograph texture mapped to the swept out two-dimensional library profile. The intensity of the light is based upon the $1/r^2$ falloff light density function previously discussed. This entire procedure is repeated each time step, with no frame-to-frame coherency information retained. For further details about this technique, the reader is referred to the original paper [24]. Overall, this method for flame animation allows for a much finer degree of control than a purely physics-based model at the cost of sacrificing the ability to procedurally capture many of the subtle effects which visually define fire.

7 A Few Comments on Water

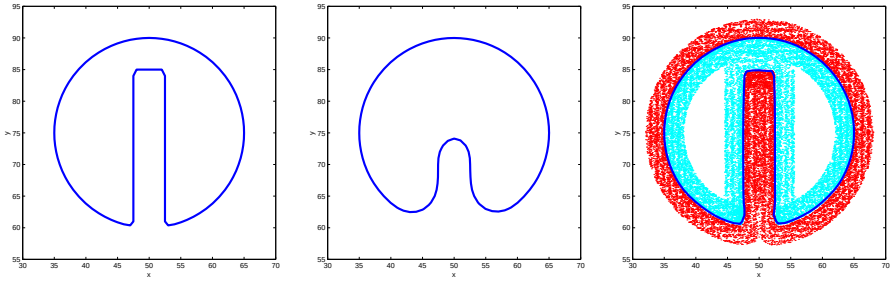
In contrast to the modeling and simulation of the physically complex phenomena which describe low speed flame propagation, the tracking of a passively advected interface between two liquid phases, e.g. air and water, is almost

simple by comparison. However, unlike fire which is visually defined by the volume of soot and the hot gaseous products produced during the combustion process, the defining characteristic of water is the thin, smooth contact discontinuity between the water and air. This fact places a host of stringent requirements on any method used to represent this interface, i.e. the method needs to: accurately track the underlying flow characteristics, readily allow for changes in the topology of the interface (pinching and merging), be computationally tractable, and not suffer from excessive amounts of numerical diffusion while maintaining a spatially and temporally smooth interface.

Lagrangian based interface techniques [19, 45] partially satisfy some of these requirements, exhibiting an excellent ability to track underlying flow characteristics without significant amounts of numerical diffusion. However, the use of a discrete, marker particle based method to represent the fluid does not readily allow for a visually smooth interface to be reconstructed from the fluid particles. Moreover, a connected marker particle approach requires the use of complex remeshing algorithms to deal with topological changes.

Eulerian based methods, e.g. Volume of Fluid (VOF) [21, 37] and level set methods [31], readily allow for topological changes due to the underlying grid representation used. However, these schemes are all subject to large amounts of mass loss and can be computationally expensive due to the embedding of the lower-dimensional interface in a higher-dimensional grid. VOF schemes attempt to cure the mass loss problem by imposing an artifical mass conservation constraint. In VOF schemes, regions of high curvature and thin, filamentary regions of the interface have difficulty in accurately sampling the underlying flow field, causing small pieces of "flotsam" and "jetsam" to form and move in visually unnatural ways. Alternatively, level set methods can maintain a smooth interface due to the signed distance property of the level set function. High order accurate HJ-WENO schemes [38] along with periodic reinitialization [44] can alleviate the diffusion of large scale interface features, but numerical dissipation makes it difficult to maintain small scale features. An example of the adverse effects of numerical dissipation can be seen in figure 13, where the notched disk in figure 13(a) undergoes a rigid body rotation. After one rotation of the disk the level set function has experienced an excessive amount of nonphysical diffusion in the corners and throughout the thin notch as seen in figure 13(b). A recently proposed interface capturing technique, the particle level set method [7], combines the strengths of Lagrangian and Eulerian methods.

The interface in the particle level set method is represented as the zero isocontour of a signed distance function ϕ. Massless marker particles are placed in a band about the interface to act as a diffusion control mechanism. The results of the particle level set method can be seen in figure 13(c) where the thin notch along with the sharp corners have maintained their original shape with little to no diffusion. The particles move according to $d\mathbf{x}_p/dt = \mathbf{V}(\mathbf{x}_p)$, and each particle possesses a radius r_p and a sign s_p. Since the level set is tracking

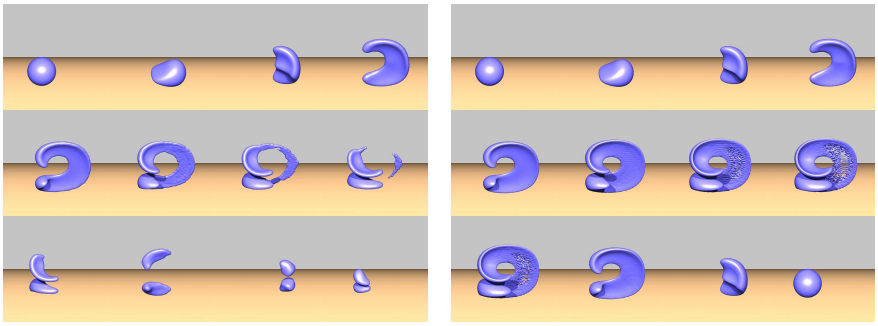


(a) Initial (b) Level Set Only (c) Particle Level Set

Fig. 13. Rigid body rotation of a notched disk.

a contact discontinuity, particles which correspond to the $\phi > 0$ region should always remain in the $\phi > 0$ region and vice versa, however excessive amounts of numerical diffusion can cause positive particles, i.e. particles with $s_p = +1$, to end up in a $\phi < 0$ region according to the level set function. These particles are said to have "escaped" from their respective side of the interface and indicate that a first order error in the location of the interface has occurred. This first order accurate error in ϕ can be corrected for by the particles since the radius of each particle defines a local level set function, ϕ_p, which we can compare against ϕ at the corners of the grid cell containing the particle. After iterating through all the escaped particles and determining corrected ϕ values, the particles then resample their distance to the interface and adjust their radii accordingly. This error reduction technique can also be used to correct errors made when ϕ is reinitialized to be a signed distance function. In this case the particle velocity is assumed to be zero since the interface should not move. A complete description of this error reduction technique can be found in [7].

The robustness of the level set method is maintained by the particle level set method since the marker particles do not *explicitly* delineate the location of the interface. Rather, they locally capture the location of the interface through the ϕ_p function, and the level set function itself is used to automatically treat connectivity (merging and pinching of fronts). The ease-of-implementation of the level set method is maintained since the particles are disconnected and communicate with the level set function only during the error reduction stage described above. Since particles are placed within a band about the $\phi = 0$ isocontour, the interface is resolved on multiple scales by the particles. This multi-resolution approach is quite successful in preserving the volume of the level set when the interface undergoes large amounts of stretching induced by an incompressible flow field as seen in figure 14. A level set only approach as seen in figure 14(a) can not maintain regions of



(a) Level Set Only (b) Particle Level Set

Fig. 14. 3D deformation test.

high curvature and the thin (approximately one grid cell thick) pancake region formed during the deformation process. On the other hand, the particle level set method can resolve these regions on the 100^3 grid used. Also, while tearing of the interface is seen in the thin pancake region with the particle level set method, particles which remain escaped are not deleted and can contribute to the rebuilding of the interface as seen in the last row of frames in figure 14(b). The sphere, which loses over 80% of its volume by the end of the deformation process when represented with a level set only method, loses only 2% of its volume with the particle level set method. The additional cost of placing particles near the interface is offset by the ability to use much coarser volumetric grids when calculating the pressure during flow calculations without sacrificing a faithful representation of the interface.

The particle level set method along with numerical methods appropriate for fluid animation as proposed by Foster and Fedkiw [15], has made it possible to animate photorealistic looking complex, three-dimensional water surfaces [9, 8]. The result of these methods can be seen in the pouring of a glass of water as shown in figure 15. This simulation was performed using a $55 \times 55 \times 120$ computational grid.

8 Conclusion

In this paper, we presented current state-of-the-art methods of simulating fire for the computer graphics community. However, many outstanding issues remain. In particular, animators desire to have greater control over the behavior of an animation. Research is ongoing to discover new techniques which allow for the degree of control desired by animators while still preserving the subtle natural character of fire and flames.

Fig. 15. Pouring of a glass of water.

9 Acknowledgments

Research supported in part by an ONR YIP and PECASE award (ONR N00014-01-1-0620), a Packard Foundation Fellowship, a Sloan Research Fellowship, ONR N00014-03-1-0071, NSF DMS-0106694 and NSF ITR-0121288. In addition, D.E. was supported in part by an NSF postdoctoral fellowship (NSF DMS-0202459).

References

1. Adalsteinsson, D. and Sethian, J., *A Fast Level Set Method for Propagating Interfaces*, J. Comp. Phys. 118, 269–277 (1995).
2. Brackbill, J., Kothe, D. and Zemach, C., *A Continuum Method for Modeling Surface Tension*, J. Comp. Phys. 100, 335–354 (1992).
3. Caiden, R., Fedkiw, R. and Anderson, C., *A Numerical Method For Two-Phase Flow Consisting of Separate Compressible and Incompressible Regions*, J. Comp. Phys. 166, 1–27 (2001).
4. Chen, S., Merriman, B., Kang, M., Caflisch, R., Ratsch, C., Cheng, L.-T., Gyure, M., Fedkiw, R., Anderson, C. and Osher, S., *Level Set Method for Thin Film Epitaxial Growth*, J. Comp. Phys. 167, 475–500 (2001).
5. Chorin, A. J., *Numerical Solution of the Navier-Stokes Equations*, Math. Comp. 22, 745–762 (1968).
6. Courant, R., Issacson, E. and Rees, M., *On the Solution of Nonlinear Hyperbolic Differential Equations by Finite Differences*, Comm. Pure and Applied Math 5, 243–255 (1952).
7. Enright, D., Fedkiw, R., Ferziger, J. and Mitchell, I., *A Hybrid Particle Level Set Method for Improved Interface Capturing*, J. Comp. Phys. 183, 83–116 (2002).
8. Enright, D., Marschner, S. and Fedkiw, R., *Animation and Rendering of Complex Water Surfaces*, ACM Trans. on Graphics (SIGGRAPH 2002 Proceedings) 21, 736–744 (2002).

9. Enright, D., Nguyen, D., Gibou, F. and Fedkiw, R., *Using the Particle Level Set Method and a Second Order Accurate Pressure Boundary Condition for Free Surface Flows*, in Kawahashi, M., Ogut, A. and Tsuji, Y., eds., *Proceedings of the 4th ASME-JSME Joint Fluids Engineering Conference*, FEDSM2003–45144, ASME, 2003.
10. Fedkiw, R., Stam, J. and Jensen, H. W., *Visual Simulation of Smoke*, in Fiume, E., ed., *Proceedings of SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, pp. 15–22, ACM, ACM Press / ACM SIGGRAPH, 2001.
11. Fedkiw, R. P., *Coupling an Eulerian Fluid Calculation to a Lagrangian Solid Calculation with the Ghost Fluid Method*, J. Comp. Phys. 175, 200–224 (2002).
12. Fedkiw, R. P., Aslam, T., Merriman, B. and Osher, S., *A Non-oscillatory Eulerian Approach to Interfaces in Multimaterial Flows (The Ghost Fluid Method)*, J. Comp. Phys. 152, 457–492 (1999).
13. Fedkiw, R. P., Aslam, T. and Xu, S., *The Ghost Fluid Method for Deflagration and Detonation Discontinuities*, J. Comp. Phys. 154, 393–427 (1999).
14. Feldman, B. E., O'Brien, J. F. and Arikan, O., *Animating Suspended Particle Explosions*, ACM Transactions on Graphics 22, 708–715 (2003).
15. Foster, N. and Fedkiw, R., *Practical Animation of Liquids*, in Fiume, E., ed., *Proceedings of SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, pp. 23–30, ACM, ACM Press / ACM SIGGRAPH, 2001.
16. Foster, N. and Metaxas, D., *Modeling the Motion of a Hot, Turbulent Gas*, in *Proceedings of SIGGRAPH 1997*, Computer Graphics Proceedings, Annual Conference Series, pp. 181–188, ACM, ACM Press / ACM SIGGRAPH, 1997.
17. Gibou, F., Fedkiw, R., Caflisch, R. and Osher, S., *A Level Set Approach for the Numerical Simulation of Dendritic Growth*, J. Sci. Comput. 19, 183–199 (2003).
18. Gibou, F., Fedkiw, R. P., Cheng, L.-T. and Kang, M., *A Second–Order–Accurate Symmetric Discretization of the Poisson Equation on Irregular Domains*, J. Comp. Phys. 176, 205–227 (2002).
19. Harlow, F. and Welch, J., *Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface*, Phys. Fluids 8, 2182–2189 (1965).
20. Helenbrook, B. and Law, C., *A numerical Method for Solving Inocmpressilbe Flow Problems with a Surface of Discontinuity*, J. Comp. Phys. 148, 366–396 (1999).
21. Hirt, C. and Nichols, B., *Volume of Fluid (VOF) Method for the Dynamics of Free Boundaries*, J. Comp. Phys. 39, 201–225 (1981).
22. Juric, D. and Tryggvason, G., *Computations of Boiling Flows*, J. Comp. Phys. 24, 387–410 (1998).
23. Kang, M., Fedkiw, R. and Liu, X.-D., *A Boundary Condition Capturing Method for Multiphase Incompressible Flow*, J. Sci. Comput. 15, 323–360 (2000).
24. Lamorlette, A. and Foster, N., *Structural Modeling of Natural Flames*, ACM Transactions on Graphics 21, 729–735 (2002).
25. Liu, X.-D., Fedkiw, R. and Kang, M., *A Boundary Condition Capturing Method for Poisson's Equation on Irregular Domains*, J. Comp. Phys. 160, 151–178 (2000).
26. Neff, M. and Fiume, E., *A Visual Model for Blast Waves and Fracture*, in *Proceedings of SIGGRAPH 1999*, Computer Graphics Proceedings, Annual Conference Series, pp. 193–202, ACM, ACM Press / ACM SIGGRAPH, 1999.

27. Nguyen, D., Gibou, F. and Fedkiw, R., *A Fully Conservative Ghost Fluid Method and Stiff Detonation Waves*, in *12th International Detonation Symposium*, ONR, 2002.
28. Nguyen, D. Q., Fedkiw, R. and Jensen, H. W., *Physically Based Modeling and Animation of Fire*, ACM Trans. on Graphics (SIGGRAPH 2002 Proceedings) 21, 721–728 (2002).
29. Nguyen, D. Q., Fedkiw, R. P. and Kang, M., *A Boundary Condition Capturing Method for Incompressible Flame Discontinuities*, J. Comp. Phys. 172, 71–98 (2001).
30. Osher, S. and Fedkiw, R., *Level Set Methods and Dynamic Implicit Surfaces*, Springer-Verlag, New York, 2002.
31. Osher, S. and Sethian, J., *Fronts Propagating with Curvature Dependent Speed: Algorithms Based On Hamiliton-Jacobi Formulations*, J. Comp. Phys. 79, 12–49 (1988).
32. Perlin, K. and Neyret, F., *Flow Noise*, in *SIGGRAPH Technical Sketches and Applications*, ACM, 2001.
33. Peskin, C., *Numerical Analysis of Blood flow in the Heart*, J. Comp. Phys. 25, 220–252 (1977).
34. Quian, J., Tryggvason, G. and Law, C., *A Front Method for the Motion of Premixed Flames*, J. Comp. Phys. 144, 52–69 (1998).
35. Rasmussen, N., Nguyen, D., Geiger, W. and Fedkiw, R., *Smoke Simulation For Large Scale Phenomena*, in *Proceedings of SIGGRAPH 2003*, Computer Graphics Proceedings, Annual Conference Series, pp. 703–707, ACM, ACM Press / ACM SIGGRAPH, 2003.
36. Reeves, W., *Particle Systems - A Technique for Modeling a Class of Fuzzy Objects*, in *Computer Graphics (Proceedings of SIGGRAPH 83)*, volume 17, pp. 359–376, ACM, 1983.
37. Rider, W. and Kothe, D., *Reconstructing Volume Tracking*, J. Comp. Phys. 141, 112–152 (1998).
38. Shu, C. and Osher, S., *Efficient Implementation of Essentially Non-Oscillatory Shock Capturing Schemes*, J. Comp. Phys. 77, 439–471 (1988).
39. Son, G. and Dir, V., *Numerical Simulation of Film Boiling near Critical Pressures with a Level Set Method*, J. Heat Transfer 120, 183–192 (1998).
40. Stam, J., *Stable Fluids*, in *Proceedings of SIGGRAPH 99*, Computer Graphics Proceedings, Annual Conference Series, pp. 121–128, ACM, ACM SIGGRAPH / Addison Wesley Longman, 1999.
41. Stam, J. and Fiume, E., *Turbulent Wind Fields for Gaseous Phenomena*, in *Proceedings of SIGGRAPH 1993*, Computer Graphics Proceedings, Annual Conference Series, pp. 369–376, ACM, ACM Press / ACM SIGGRAPH, 1993.
42. Staniforth, A. and Côté, J., *Semi-Lagrangian Integration Schemes for Atmospheric Models - A Review*, Monthly Weather Review 119, 2206–2223 (1991).
43. Steinhoff, J. and Underhill, D., *Modification of the Euler Equations for "vorticity confinement": Application to the computation of interacting vortex rings*, Phys. Fluids 6, 2738–2744 (1994).
44. Sussman, M., Smereka, P. and Osher, S., *A Level Set Approach for Computing Solutions to Incompressible Two-Phase Flow*, J. Comp. Phys. 114, 146–159 (1994).
45. Tryggvason, G., Bunner, B., Esmaeeli, A., Juric, D., Al-Rawahi, N., Tauber, W., Han, J., Nas, S. and Jan, Y.-J., *A Front-Tracking Method for the Computations of Multiphase Flow*, J. Comp. Phys. 169, 708–759 (2001).

46. Unverdi, S. and Tryggvason, G., *A Front-Tracking Method for Vicous, Incompressible, Multi-Fluid Flows*, J. Comp. Phys. 100, 25–37 (1992).
47. Welch, S. and Wilson, J., *A Volume of Fluid Based Method for Fluid Flows with Phase Change*, J. Comp. Phys. 160, 662–682 (2000).
48. Yngve, G. and O'Brien, J., *Animating Explosions*, in *Proceedings of SIGGRAPH 2000*, Computer Graphics Proceedings, Annual Conference Series, pp. 29–36, ACM, ACM Press / ACM SIGGRAPH, 2000.
49. Yngve, G. D., O'Brien, J. F. and Hodgins, J. K., *Animating Explosions*, in *Proceedings of ACM SIGGRAPH 2000*, Computer Graphics Proceedings, Annual Conference Series, pp. 29–36, 2000.
50. Zhang, Y., Yeo, K., Khoo, B. and Wang, C., *3D Jet Impact and Toroidal Bubbles*, J. Comp. Phys. 166, 336–360 (2001).