Technical Report No. UIUCDCS-R-2004-2438 (Engr. UILU-ENG-2004-1738)

Controllable Smoke Animation with Guiding Objects

LIN SHI and YIZHOU YU
University of Illinois at Urbana-Champaign

Abstract

This paper addresses the problem of controlling the density and dynamics of smoke (a gas phenomenon) so that the synthetic appearance of the smoke (gas) resembles a still or moving object. Both the smoke region and the target object are represented as implicit functions. As a part of the target implicit function, a shape transformation is generated between an initial smoke region and the target object. In order to match the smoke surface with the target surface, we impose carefully designed velocity constraints on the smoke boundary during a dynamic fluid simulation. The velocity constraints are derived from an iterative functional minimization procedure for shape matching. The dynamics of the smoke is formulated using a novel compressible fluid model which can effectively absorb the discontinuities in the velocity field caused by imposed velocity constraints while reproducing realistic smoke appearances. As a result, a smoke region can evolve into a regular object and follow the motion of the object while maintaining its smoke appearance.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Physically based modeling I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation I.6.8 [Simulation and Modeling]: Types of Simulation—Animation

Keywords: Constrained Animation, Fluid Simulation, Implicit Functions, Level Sets, Shape Matching, Shape Transformations, Velocity Constraints

1 Introduction

Amorphous but elegantly moving matters, such as clouds, fog and smoke, usually give people plenty of space for imagination. We would be excited when a cloud in the sunset sky assumes the approximate shape of an animal or some other real object. It is indeed an exhilarating event because of its rareness. For the same reason, ghosts and deities are usually described to manifest themselves from smoke or clouds. Even a fairy tale has the following scene: as Aladdin rubbed the lamp to try to get a better look, the lamp came to life; the lamp launched a long, blue stream upward; the blue smoke rose toward the ceiling, and finally became an enormous, blue genie!

We would like to develop techniques for digitally reproducing similar effects. Such techniques have many applications in the entertainment industry especially in advertising and film making. In some of the recent movies, there have been voxel water horses emerging from a flooding river [Kapler 2002], and the mummy manifests itself from sand. Our goal in this paper is to introduce methods that produce physically plausible motion for a gas phenomenon, which at the same time, assumes a recognizable static or dynamic shape. In the rest of the paper, we choose *smoke* as a representative of such gas phenomena. Nevertheless, the approach introduced here is not only limited to smoke.

Our goal in this paper has the following implications:

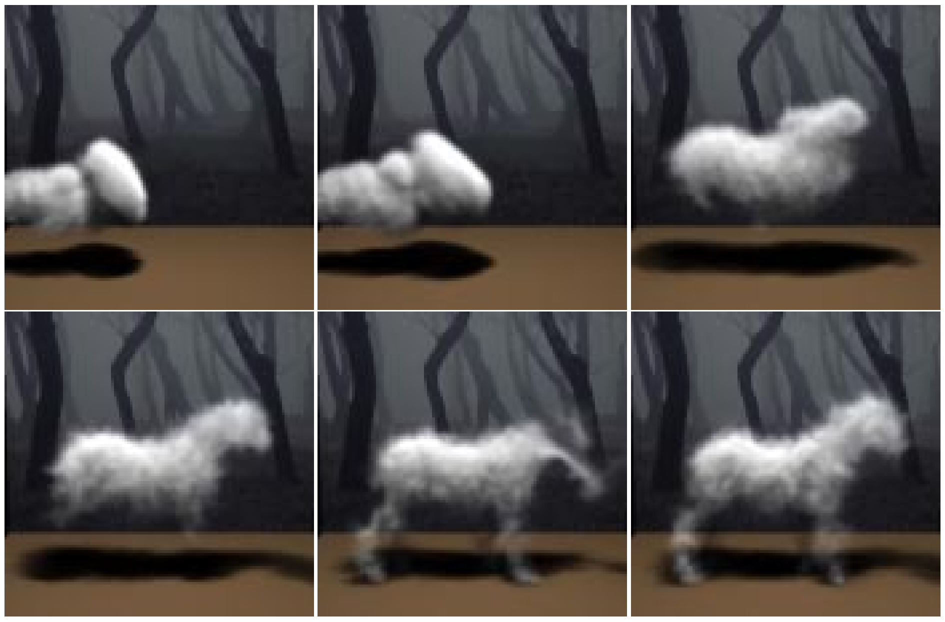
- the motion during two-way transitions between irregular smoke regions and regular object shapes should be natural and have realistic smoke appearance;
- the global shape of the smoke should be able to approximate a static or moving object for an arbitrarily long period of time while maintaining its characteristic local structure and motion;
- When smoke objects interact with each other or with the environment, the objects should exhibit the properties of smoke so that a strong wind or other regular objects can easily destroy the shape of such objects.

An example with a smoke horse is shown in Fig. 1(a) to illustrate this goal.

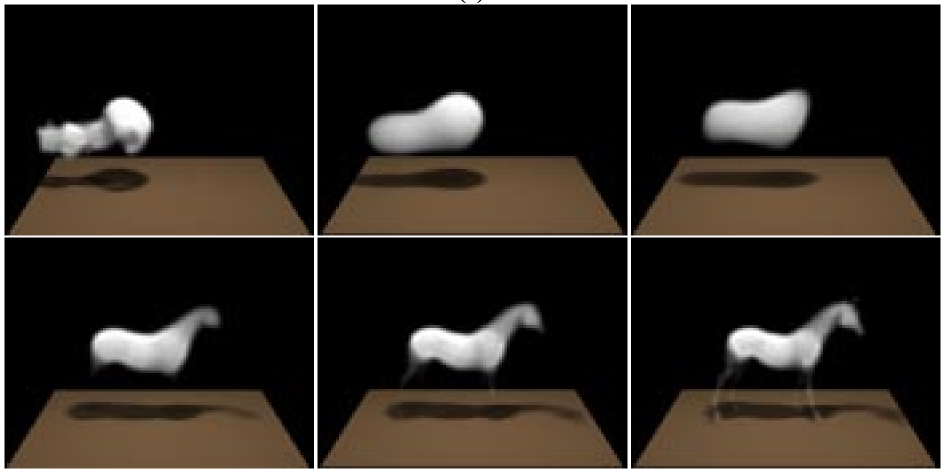
Smoke consists of a collection of light-scattering tiny particles floating in the air. Creating the above dramatic effects is challenging since the smoke density in a fluid medium always tends to drift from a nonuniform distribution to a uniform one. Solving the proposed problem requires the maximum level of control of this process while maintaining a believable appearance of smoke. When there is a conflict between controllability and physics rules, we choose to relax the physics rules since the desired effects can largely be considered as a supernatural phenomenon. Our goal in this paper is consistent with one of the general objectives of graphics research: the development of techniques that allow easy user-level control of the modeling and animation processes.

This paper presents an effective solution to the proposed problem. Our solution involves implicit functions (level sets) defined for both smoke-object transitions and object motion. Since they are functions of both space and time, these level sets represent both the shape of the target objects as well as their evolution over time. These implicit functions serve as the underlying ”storyboard” guiding the motion of the smoke. Thus, the problem becomes how to impose constraints on the motion of the smoke so that the smoke density distribution approximately matches these evolving level sets while the realistic appearance of smoke is being maintained. This is actually a control problem that can be solved by a dynamic feedback process. The basic idea of our solution lies in the use of artificial feedback forces on the smoke so that subtle changes in the movement of the smoke reduces the shape discrepancy between the smoke and the target object. Such feedback forces are actually realized by velocity adjustments and constraints. They do not exist in the real world, and need to be carefully orchestrated to achieve the desired effects.

Major contributions of this paper include an overall framework for solving the proposed problem, an automatic scheme for target object matching based on velocity constraints imposed on the motion of the smoke, and an empirical compressible fluid model for effectively integrating constraints into the velocity field. These velocity constraints are derived from a shape matching functional. Simple but effective methods for smoke objects to interact with each other or the environment are also developed.



(a)



(b)

Figure 1: (a) An initial smoke blob evolves into a smoke horse. A wind blows the head away. When the wind recedes, the head grows back. (b) The underlying shape transformation for the smoke-horse transition.

2 Related Work

The modeling of smoke and other gaseous phenomena has received much attention from the computer graphics community over the last two decades. Early models focused on a particular phenomenon and animated the smoke density directly without modeling its velocity [Gardner 1985; Perlin 1985; Ebert et al. 1998]. Additional details were added using solid textures whose parameters were animated over time. [Musgrave 1997] proposed a procedural hypertexture [Perlin and Hoffert 1989] approach, to model dynamic fractal cloudlets which can be arranged to approximate regular shapes such as animals. Each cloudlet was animated separately. A common feature shared by these models and the voxel tool presented in [Kapler 2002] is that they lack any dynamical feedback which is crucial to realistic animation.

A more natural way to model the motion of smoke is to simulate the equations of fluid dynamics. Recently, [Foster and Metaxas 1997b] used relatively coarse grids to produce nice smoke motion in three-dimensions. Their simulations are only stable if the time step is small enough. To alleviate this problem and make the simulations more efficient, Stam introduced a new model which is unconditionally stable and could be run at any speed [Stam 1999]. This was achieved using a combination of a semi-Lagrangian advection scheme [Staniforth and Cote 1991] and implicit solvers. [Fedkiw et al. 2001] introduced vorticity confinement and a higher-order interpolation technique. As a result, the simulations can keep finer details on relatively coarse grids.

The level set method [Osher and Sethian 1988; Sethian 1999; Osher and Fedkiw 2001] is a robust computational method to track the surface of a volume of fluid. [Foster and Fedkiw 2001] made significant contributions to fluid simulation and control through the introduction of a hybrid liquid volume model combining implicit surfaces and massless marker particles. The work was further improved substantially in terms of accuracy in [Enright et al. 2002] by using particles on both sides of the interface between air and water. The level set method have also inspired very interesting methods for fast surface reconstruction from unorganized points [Zhao et al.

2001] and geometric object editing [Museth et al. 2002]. What is important in these methods is the quality of the resulting static geometry instead of the dynamic surface evolution itself. In comparison, we are more concerned with the quality of the dynamics itself for the realistic appearance of smoke.

In terms of fluid control, Stam [Stam 1995] applied multiple types of vector fields to control the global trajectories of fluids. Foster and Metaxas [Foster and Metaxas 1997a] introduced embedded controllers that allow animators to specify and control a three dimensional fluid animation without knowledge of the underlying equations or the method used to solve them. [Foster and Fedkiw 2001] suggest to apply animator-designed ”fake” space curves and surfaces to control the motion of liquids. The tangents of the curves or the normals of the surfaces indicate the directions of motion. [Lamorlette and Foster 2002] also includes animator-defined time and space curves to control the structures of flames. However, the general objective of these attempts is not about making fluids look like regular still or moving objects.

More recently, [Treuille et al. 2003] proposes a gradient-based method for controlling smoke simulations through user-specified keyframes. The problem is cast as matching dynamically evolved smoke density with the specified static density distributions at a sparse set of keyframes. The elegant part of this approach is dynamically simulating the derivatives of the velocity field in the same framework for simulating the velocity field itself. A novel multiple shooting scheme is also designed for matching multiple keyframes. However, since the derivative of the velocity field with respect to each control parameter needs to be evaluated throughout a portion of an animation sequence, this approach is computationally expensive. Like other gradient-based approaches, there is also a possibility to be stuck in local minima. In comparison, the method introduced in this paper is not designed for sparse keyframes. It can maintain smoke appearances around the (animated) target objects for an arbitrarily long time. The configurations of our guiding objects may be specified at every frame to define continuous rigid-body motion and deformation.

Warping and morphing techniques [Wolberg 1990; Gomes et al. 1997] are very useful tools for transforming images and objects. Warping does not have a clear target image or object while morphing usually produces transition between two objects. [Sims 1992] introduced a technique for successively warping images using vector fields. However, the results from the operation are not well controlled. 3D volume morphing methods [Hughes 1992; Lerios et al. 1995; Turk and O’Brien 1999; Alexa et al. 2000] can achieve well-behaved shape interpolation by considering both boundary and interior points. Actively moving implicit surfaces [Desbrun and Cani-Gascuel 1998] can be used to generate metamorphosis between shapes without a correspondence. Most morphing techniques focus on the smoothness of the planned transition instead of its physical plausibility. In many experiments, we adopt the method in [Turk and O’Brien 1999] to generate an underlying shape transformation guiding smoke-object shape transitions.

3 Overview

In our method, we define an implicit function for the target object and try to drive an irregularly shaped smoke region so that a specific isosurface of the density distribution of the smoke closely matches the boundary (zero) level set of the target object. The implicit function for the target object is named the *guiding implicit function* and is denoted as $D(\mathbf{x}, t)$ where $\mathbf{x}$ represents a point in a 2D or 3D space, and the extra variable t means the target shape may move or deform over time. The density distribution of the simulated smoke is denoted as $\rho(\mathbf{x}, t)$. We use dynamic force feedback to actively influence the smoke isosurface so that the resulting dynamic surface represented by $\Gamma_\rho = \{x|\rho(\mathbf{x}, t) - \tau = 0\}$ approximately matches the zero level set of the target object, $\Gamma_D = \{x|D(\mathbf{x}, t) = 0\}$ where τ is a threshold used for defining a boundary isosurface for the smoke region which is assumed to have internal densities higher than τ. Note that $\rho(\mathbf{x}, t) - \tau = 0$ is actually an implicit function for the smoke region. At every time step, given the shape discrepancy between the two isosurfaces, feedback forces are applied to the smoke boundary to reduce the amount of discrepancy. Since forces and accelerations are connected through the Newton’s law, in practice, such feedback forces are actually realized by velocity adjustments.

There are two essential stages in an animation where smoke evolves into an object. The first stage involves a shape transition between the smoke region and the target object. In the second stage, the transformed smoke region needs to keep track of the object’s own nonrigid deformation or rigid-body motion. These two stages can be treated in a unified framework. The initial smoke region obtained by thresholding can be considered as an irregular object. 3D shape morphing techniques can be applied to generate a morph sequence between the initial smoke shape and the target object shape (Fig. 1(b)). As a result, we can obtain an intermediate shape at any instance during the whole transition period. Thus, the shape transition between the smoke and the target object can be viewed as a nonrigid shape deformation, and any method designed for the second stage can be applied as well to this stage where the morph sequence is used to generate the guiding implicit function which the smoke tries to match. Therefore, we only need to develop an approach for the smoke to track object motion. A static object is a special case for this problem.

Nevertheless, tracking object motion using the smoke is by no means trivial. When there is temporal coherence and the frame-to-frame motion of the target object is small, it is possible to obtain velocity constraints on the boundary isosurface of the smoke by exploiting the local gradient flow of the boundary. If a physical simulation of the smoke satisfies these boundary velocity constraints, the difference between the two boundary isosurfaces will be decreased. However, when there is fast frame-to-frame motion, the target object from two consecutive frames might have a huge gap in position or orientation. Local gradient structure of the implicit functions cannot guarantee efficient tracking results any more. When this happens, we use the target object to transport the smoke inside from its location in the previous frame to its location in the current frame as if the target object is a container with a hard boundary. The hard boundary disappears once the smoke is in position again. Note that this step may not be physically realistic, and is specially designed to achieve the desired phenomenon.

The overall solution to the proposed problem has the following components:

- The animator needs to pick a target object and chooses a smoke region from a simulation. A 3D morph sequence is created between the initial smoke shape and the target object. The target object for each frame is voxelized into a discrete implicit function.
- For each frame, the system determines whether large motion has occurred by checking the amount of overlap between the two underlying object shapes at the previous and current frames. The strategy for smoke transportation is executed if the amount of overlap falls below a predefined threshold. Small motion tracking is always performed no matter whether large motion occurred or not.
- During each iteration, the system also simulates potential interactions among multiple smoke objects and interactions between a smoke object and its environment. The shape of a smoke region may be destroyed during such interactions, and a new morph sequence can be generated between the remaining smoke and the target object.

4 Guiding Objects

The input to our system includes the target objects and their motion. When the objects are not static, the configuration of the objects needs to be specified at every frame. The type of motion may include simple rigid-body motion, more complicated articulated body motion, or even nonrigid deformations. All target objects are internally represented as implicit functions. Our representation for the guiding implicit function $D(\mathbf{x}, t)$ is the signed distance function which is zero on the object boundary and positive at the interior. The signed distance function has the advantage to conveniently provide the shortest distance between any point on the smoke boundary and the boundary of the guiding implicit function. Therefore, input objects need to be converted to this representation even if they are already given as implicit functions.

Since smoke simulation will be performed on a volume grid, the smoke implicit function $\rho(\mathbf{x}, t) - \tau$ is always directly represented on this discrete grid in the form of a density value at each voxel. The boundary isosurface of this function is obtained by labeling the voxels containing a density value close to the threshold τ. The signed distance function of a guiding object at a specific frame is also represented on the same volume grid by first discretizing the object's original representation followed by a conversion to signed distance values.

As discussed in Section 3, a part of the animation may involve a shape transition between the smoke region and the target object. An intermediate shape should be generated at each frame during this transition. This intermediate shape serves as the guiding object for that frame. And it needs to be represented as an implicit function as well. In practice, we apply 3D shape morphing and masking techniques (see Section 7.1) to generate a shape transition sequence which can produce such an intermediate shape at any intermediate frame. These intermediate shapes should be represented as or converted to signed distance functions whenever they are needed.

5 Shape Matching

5.1 Velocity Constraints for Small Motion

Suppose at a certain time t_i during simulation, the guiding implicit function is $D(\mathbf{x}, t_i)$. We would like to evolve the smoke density at the previous step $\rho(\mathbf{x}, t_{i-1})$ so that the zero isosurface of the updated function $\rho(\mathbf{x}, t_i) - \tau$ approximately matches the boundary of the guiding object.

Since we are concerned with matching two isosurfaces, let us first look at a criterion for measuring shape discrepancy. If we represent an object as a point set, two objects A and B exactly match each other if and only if both sets $A-B$ and $B-A$ are empty, which is also equivalent to that the volumes of both $A - B$ and $B - A$ are zero. The summed volumes of these two sets indicate the level of discrepancy between two shapes. Mathematically, we need to use characteristic functions of the shapes and integrals to represent these two volumes. Let us define $\chi_D(\mathbf{x}, t_i) = 1$ if $D(\mathbf{x}, t_i) \geq 0$, and $\chi_D(\mathbf{x}, t_i) = 0$, otherwise; and define $\chi_\rho(\mathbf{x}, t_{i-1}) = 1$ if $\rho(\mathbf{x}, t_{i-1}) - \tau \geq 0$, and $\chi_\rho(\mathbf{x}, t_{i-1}) = 0$, otherwise. The level of discrepancy between the two zero isosurfaces can be measured by the following integral,

$$e_v = \int \chi_\rho(\mathbf{x}, t_{i-1})(1-\chi_D(\mathbf{x}, t_i))d\mathbf{x} + \int \chi_D(\mathbf{x}, t_i)(1-\chi_\rho(\mathbf{x}, t_{i-1}))d\mathbf{x} \tag{1}$$

where the global minimum is zero and can be reached when the two characteristic functions coincide. (1) can be simplified to $\int \chi_D d\mathbf{x} + \int \chi_\rho(1 - 2\chi_D)d\mathbf{x}$ where χ_ρ is the shape variable and χ_D is the fixed guiding shape for a specific frame since we would like $\rho(\mathbf{x}, t_{i-1})$ to approximate $D(\mathbf{x}, t_i)$. Therefore, reducing the shape discrepancy between the two is equivalent to minimizing the following functional,

$$\int \chi_\rho(1 - 2\chi_D)d\mathbf{x}. \tag{2}$$

According to calculus of variations [Gelfand and Fomin 1963], one can show that the first variation of the integral in (2) with respect to the smoke boundary surface is simply based on the second part of its integrand and the normal directions of the smoke boundary. Therefore, the negative variational gradient minimizing the functional in (2) with respect to the smoke boundary is as follows.

$$\left.\frac{\delta e_v}{\delta \Gamma_\rho}\right|_{\mathbf{x}=\mathbf{x_b}} = (1 - 2\chi_D)\left.\frac{\nabla\rho}{\|\nabla\rho\|}\right|_{\mathbf{x}=\mathbf{x_b}} \tag{3}$$

where Γ_ρ represents the smoke boundary and $\frac{\nabla\rho}{\|\nabla\rho\|}$ represents the unit inward normal at the smoke boundary. Detailed discussion on the derivation of this variational gradient is presented in Appendix A.

Consider a specific point $\mathbf{x_b}$ on the boundary of the smoke, the sign of $D(\mathbf{x_b}, t_i)$ indicates its location with respect to the guiding object. Note that the gradient of $\rho(\mathbf{x}, t_{i-1})$ at $\mathbf{x_b}$ points to the interior of the smoke and is perpendicular to the boundary. Under the small motion assumption, the smoke and guiding object should have overlap. If $\mathbf{x_b}$ is inside the guiding object, we need to move it slightly along the negative gradient direction; otherwise, move it along the positive gradient direction. (3) means iteratively perturbing all the smoke boundary points simultaneously in this way would gradually decrease the costs in (2) and (1). Since a new smoke boundary is formed by the relocated points after each iteration, the gradient of the new boundary should be used for moving points in the subsequent iteration. This scheme for functional minimization bears resemblance to gradient descent for regular function minimization. As gradient descent, this scheme cannot always converge to the global minimum if the initial shape of the unknown is not sufficiently close to the target shape. For example, if the smoke region does not have overlap with the guiding object, this scheme would gradually shrink the smoke region until it disappears and at the same time decrease the cost in (1) to be the volume of the guiding object which is actually a correct local minimum.

In the current context, we can modify the above iterative minimization scheme to avoid the local minimum for two separate shapes. When $D(\mathbf{x_b}, t_i)$ is negative, indicating $\mathbf{x_b}$ is outside the guiding object, it should be moved along a direction along which the directional derivative of $D(\mathbf{x}, t_i)$ is positive to bring it closer to the boundary of the guiding object. Since this condition must be satisfied either by the positive or by the negative gradient direction of $\rho(\mathbf{x}, t_{i-1})$, we only need to choose the right one of these two instead of always following the positive gradient. We still keep the original scheme when $\mathbf{x_b}$ is inside the guiding object.

Since we would like to realistically evolve the boundary of $\rho(\mathbf{x}, t_{i-1}) - \tau$ into the target shape, the dynamic evolution should follow the above iterative procedure to minimize the integral in (1) as well as follow the physics rules in smoke simulation as closely as possible. To achieve these goals, the smoke simulation should satisfy velocity constraints derived from the minimization procedure. Since (3) shows a first-order scheme, we should adopt its direction field but adjust its magnitude to improve stability.

A velocity $\mathbf{u}$ on the smoke boundary can be decomposed into a normal component $\mathbf{u_n}$ and a tangential component $\mathbf{u_t}$. Based on (3), the normal component $\mathbf{u_n}$ at a smoke boundary point $\mathbf{x_b}$ at time t_i is defined to be

$$\mathbf{u_n}(\mathbf{x_b}, t_i) = C_n \cdot \min(d_{max}, |D(\mathbf{x_b}, t_i)|) \cdot \frac{\nabla\rho}{||\nabla\rho||} \cdot \mathrm{msgn}(\mathbf{x_b}, t_i) \tag{4}$$

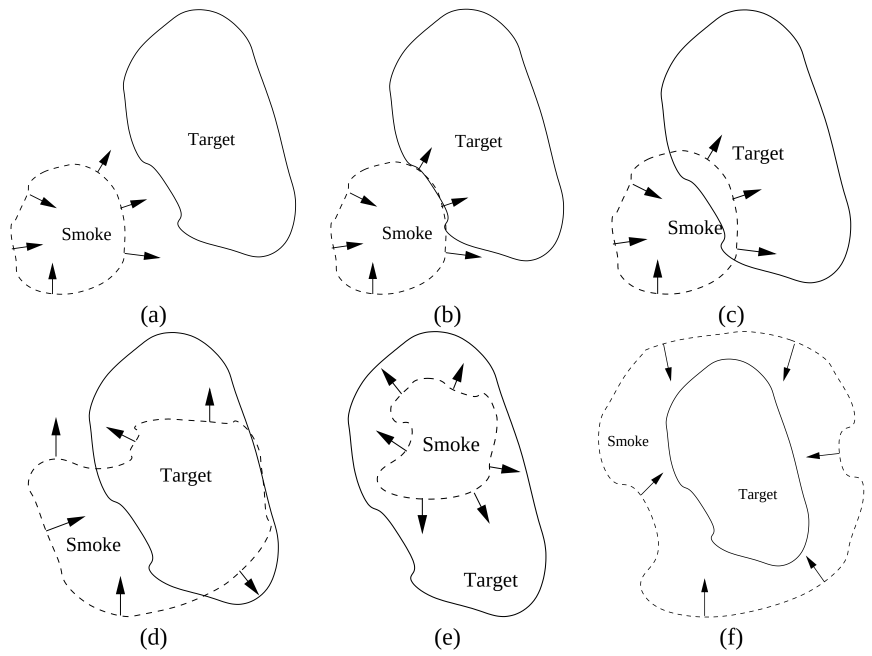


Figure 2: Normal velocity constraints on the smoke boundary for various situations. They are based on our revised minimization scheme for Eq. (2). (a) The smoke region is completely outside the guiding object; (c) the smoke region partially overlaps with the guiding object; (b)&(d) a portion of the smoke boundary touches the boundary of the guiding object. The smoke velocity should not be affected in (b) while it should be reduced to zero in (d); (e) the smoke region is completely inside the guiding object; (f) the smoke region encloses the guiding object.

where C_n is a constant scaling parameter, $|D(\mathbf{x_b}, t_i)|$ is the magnitude of the signed distance function, indicating how far away $\mathbf{x_b}$ is from the current boundary of the guiding object, d_{max} is a clamping upper bound for the distance, and msgn$(\mathbf{x_b}, t_i)$ adjusts the direction of the velocity vector according to our modified scheme. Specifically, msgn$(\mathbf{x_b}, t_i) = -1$, if $D(\mathbf{x_b}, t_i) > 0$; msgn$(\mathbf{x_b}, t_i) =$ sgn$(\nabla D \cdot \nabla \rho)$, if $D(\mathbf{x_b}, t_i) \leq 0$. The incorporation of the term $\min(d_{max}, |D(\mathbf{x_b}, t_i)|)$ can alleviate overshooting when $\mathbf{x_b}$ is already close to the boundary of the guiding object. However, this term should only be present in the normal component when the smoke region already has some overlap with the guiding object. Otherwise, it would reduce the velocity of the smoke to zero when it touches the boundary of the guiding object, and keep the smoke from entering the object. Various situations for setting up normal velocity constraints are summarized in Fig. 2. Note that when the volume of the smoke differs from the object, these velocity constraints do not preserve mass. This is a tradeoff we need to make between physics and shape matching.

Constraints on the tangential component are also crucial. Although the tangential component does not directly affect the shape of the level sets, it does affect the surrounding velocity field. While a zero tangential component would make the smoke surface less alive, the numerical stability of the shape matching procedure may be compromised if it becomes overly large. The maximum allowable tangential component is actually dependent on the local geometry of the smoke surface. If the surface is flat, the tangential displacement can be large without destroying the original shape while a highly curved surface is certainly more vulnerable. Therefore, the magnitude of the tangential component should have an upper bound related to the surface curvature. Since we would like to follow physical simulation as faithfully as possible, we can simply clamp the physically generated tangential components against the upper bound when their magnitude becomes too large. Thus, the constrained tangential component is simply defined to be

$$\mathbf{u_t}(\mathbf{x_b}, t_i) = \min(||\mathbf{u_t^*}||, \frac{C_t}{K})\frac{\mathbf{u_t^*}}{||\mathbf{u_t^*}||} \tag{5}$$

where $\mathbf{u_t^*}$ is the tangential component generated from a simulation, C_t is a constant parameter, and K is the surface curvature. In practice, we use mean curvature. A robust implementation of mean curvature on a volume grid can be found in [Museth et al. 2002].

Note that the normal and tangential velocity constraints are derived velocities for matching two shapes. They are different from those constraints defined in [Fedkiw et al. 2001; Foster and Fedkiw 2001] for fluid-object interaction and fluid control. Our guiding objects are invisible ”ghost” objects and they do not directly interact with the smoke.

5.2 Velocity Constraints for Large Motion

When the frame-to-frame motion of the guiding object becomes excessively large, the two instances of the guiding object at two consecutive frames may have little overlap or no overlap at all. Although the matching scheme developed in the previous section can eventually converge, many iterations would be needed to actually reach convergence. For efficiency considerations, we directly transport the smoke from the location of the first object instance to the second. During this direct transportation, the guiding object is assumed to have hard boundaries and the part of the smoke that is already inside becomes trapped and moved together with the guiding object. Meanwhile, velocity constraints should still be imposed at the boundary and interior of the guiding object so that these constraints can bring along the surrounding region and generate turbulent flows to create a fluid appearance and ”evidences” for the large motion. These constraints are not for transporting the smoke inside the guiding object. Suppose a point on the guiding object moved from $\mathbf{x_{i-1}}$ to $\mathbf{x_i}$ during two consecutive frames at time t_{i-1} and t_i, a velocity constraint

$$\mathbf{u_L} = \frac{\mathbf{x_i} - \mathbf{x_{i-1}}}{t_i - t_{i-1}}$$

should be imposed at $\mathbf{x_{i-1}}$ at time t_{i-1}. There is such a constraint for every voxel on and inside the guiding object at t_{i-1}. Note that the smoke transportation scheme in this section only accounts for large rigid-body motion and leaves nonrigid deformation to the velocity constraints in the previous section.

The velocity constraints we have come up need to be applied either at the boundary or interior of the smoke when the shape difference between the guiding object and the smoke region becomes sufficiently large. To make this statement more concrete, we are going to discuss in the following sections how to measure the shape difference and how to detect and evolve the smoke boundary.

5.3 Error Metrics

We use two different error metrics for measuring the shape difference. The velocity constraints are only applied when the specific metric being used exceeds some threshold. The first one is the volume discrepancy e_v defined in Eq. (1). The second metric is a generalization of the L^p norm to object boundaries,

$$e_{L^p} = (\int_{\Gamma_\rho} D^p(\mathbf{x},t)d\mathbf{x})^{1/p} \tag{6}$$

where p is positive, Γ_ρ is the boundary of the smoke and $D(\mathbf{x},t)$ is the distance function for the guiding object at time t. The latter two quantities have been defined in Section 3. A special case of the L^p norm is the L^∞ norm, which is equivalent to

$$e_{L^\infty} = \max_{\mathbf{x}\in\Gamma_\rho} D(\mathbf{x},t) \tag{7}$$

Although the velocity constraints were derived using the first metric, both metrics have the same global minimum and can effectively reduce the shape difference. However, the dynamic behavior of the smoke under their respective control can be quite different, especially when p becomes large in the second metric. For instance, e_v allows small portions of the smoke boundary to be far away from the guiding object as long as the integral remains small while e_{L^∞} keeps all the boundary points of the smoke within a certain distance from the guiding object. As a result, e_v or e_{L^p} with low p values give rise to more realistic smoke appearances on the boundary and less clear object structures while e_{L^p} with high p values becomes more appropriate when clear object structures are desirable.

5.4 Smoke Evolution and Boundary Detection

The smoke implicit function $\phi = \rho - \tau$ is evolved passively over time by the wind velocity field $\{\mathbf{u}\}$ which may be partially constrained. It can be easily shown [Osher and Sethian 1988] that the equation to update ϕ is as follows.

$$\phi_t + \mathbf{u}\cdot\nabla\phi = 0. \tag{8}$$

Except for the density threshold τ for boundary detection, this equation coincides with the advection equation for the smoke density in [Fedkiw et al. 2001]. This equation can be solved either by the semi-Lagrangian method or by the upwind scheme [Sethian 1999]. Note that both operate on the whole voxel grid instead of boundary voxels only. We have implemented both methods. Both of them can produce visually realistic results. Since a smoke boundary tends to evolve relatively slowly, the choice of a numerical method is not very critical.

Our normal and tangential velocity constraints are positioned at the boundary of the smoke region. Enforcing these constraints requires the detection of the smoke boundary at every time step. This can be easily achieved since this smoke boundary is just the zero isosurface of the smoke implicit function.

6 Smoke Simulation for Constrained Velocities

Since we exploit velocity constraints to achieve shape matching, a smoke simulation framework that can effectively incorporate hardwired velocity constraints without producing visual discontinuities is desirable. Before we introduce our revised model for smoke simulation, let us first have a look at the already adopted mathematical formulations for fluid simulation in the graphics literature [Foster and Metaxas 1997b; Stam 1999; Yngve et al. 2000; Fedkiw et al. 2001].

6.1 Previous Formulations

The dynamics of a compressible fluid can be modeled using the following Navier-Stokes equations:

$$\begin{aligned} \frac{\partial\rho_f}{\partial t} &= -\nabla\cdot(\rho\mathbf{u}) && (9)\\ \rho_f\frac{\partial\mathbf{u}}{\partial t} &= -\rho_f(\mathbf{u}\cdot\nabla)\mathbf{u} + \nu\nabla^2\mathbf{u} + \frac{\nu}{3}\mathbf{H}(\mathbf{u}) + \rho_f\mathbf{f} - \nabla P && (10)\end{aligned}$$

where ρ_f is the density of the fluid which is different from the smoke density, $\mathbf{u}$ represents the velocity field, P is the pressure term, $\mathbf{f}$ is an external force field, and ν is the kinematic viscosity of the fluid. The x component of $\mathbf{H}(\mathbf{u})$ is $\nabla\cdot(\frac{\partial\mathbf{u}}{\partial x})$, and the y and z components are defined similarly. The first equation arises from the conservation of mass, but the volume of the fluid is compressible with increasing density for a decreasing volume. The second equation is for the conservation of momentum; the first term on the right hand side is the convective term; the second and third terms model accelerations due to viscous forces; the last two terms model accelerations due to external forces and forces arising from the pressure gradient.

These equations can effectively model high-speed velocity fields such as shock waves generated by explosions [Yngve et al. 2000]. However, a strict time step condition is necessary for stable numerical solutions. For liquids and low-speed gaseous phenomena, the compressibility effects are negligible, and conservation of mass becomes equivalent to conservation of volume. The assumption of incompressibility leads to the following equations which make more efficient numerical methods possible.

$$\begin{aligned} \nabla\cdot\mathbf{u} &= 0 && (11)\\ \frac{\partial\mathbf{u}}{\partial t} &= -(\mathbf{u}\cdot\nabla)\mathbf{u} - \frac{1}{\rho_f}\nabla P + \nu\nabla^2\mathbf{u} + \mathbf{f} && (12)\end{aligned}$$

where the first equation means the fluid is volume-preserving, and the second one is similar to the corresponding equation in the aforementioned model for compressible fluids. [Stam 1999] proposed an unconditionally stable numerical method for these equations by adopting semi-Lagrangian tracing and the Helmholtz-Hodge Decomposition.

If we focus on gases, the effects of viscosity are also negligible. Therefore, the diffusion terms in the above models can be left out. Simulating gaseous phenomena thus reduces to solving the following incompressible and inviscid Euler equations [Fedkiw et al. 2001]:

$$\nabla\cdot\mathbf{u} = 0 \tag{13}$$

$$\frac{\partial \mathbf{u}}{\partial t} \quad = \quad -(\mathbf{u}\cdot\nabla)\mathbf{u} - \nabla P + \mathbf{f} \tag{14}$$

where the factor $\frac{1}{\rho_f}$ in (12) has been integrated into the ”pressure” P which is used to guarantee zero divergence, but does not represent actual pressure any more.

6.2 An Empirical Equation for Compressible Gases

The basic framework in this paper is a dynamic feedback system where the velocity constraints defined in Section 5 are dynamically updated every time step according to the shape discrepancy between the smoke region and the underlying guiding object. The purpose of such velocity constraints is to reduce the amount of discrepancy. These velocity constraints have poor spatial and temporal coherence because constraints at spatially or temporally adjacent voxels may be quite different. In addition, the constraints are only imposed at the boundary of the smoke region which is essetially a thin layer in the simulation volume. Since these artifical velocity constraints do not exist in the real world, we argue that existing fluid simulation methods cannot incorporate them without introducing artifacts.

First, the incompressible scheme in Eq. (13) and (14) does not work well. If we impose velocity constraints which generate discontinuities in the velocity field, concentrated high pressure tends to appear rapidly in regions nearby to guarantee zero divergence. Those sudden high pressures further influence surrounding velocities and generate temporal discontinuities and visual artifacts. Such artifacts have been observed in our experiments. Second, existing compressible schemes would not fit our purpose very well, either. Simulating true compressibility is very expensive and requires a strict time step condition. On the other hand, [Chorin 1967] introduces an unphysical scheme called ”artificial compressibility”, which allows a certain degree of compressibility during a transition period, and converges to incompressible fluid simulation when a steady solution has been reached. However, this convergence is achieved over time instead of within each time step, and is only guaranteed when external forces are absent. In our simulation, incoherent external forces are exerted constantly at every time step since we need to update the velocity constraints. Thus, the scheme in [Chorin 1967] is unlikely to converge and may even become unstable. In addition, the sound speed used in this scheme needs to be unusually small to prevent temporal discontinuities. An unreasonable sound speed further compromises the realism in the simulation results.

The nature of our control scheme demands a new fluid simulation technique which does not have to be physically accurate, but needs to produce results that are visually appealing. What we need is a simulation technique that approximates the behavior of incompressible fluids while absorbing discontinuities where velocity constraints are present. We would also like the technique to be stable and efficient. To achieve our goals, we propose an empirical scheme that does not strictly enforce incompressibility. Recall that the pressure P in (14) can be estimated numerically using the Poisson equation [Stam 1999; Fedkiw et al. 2001]

$$\nabla^2 P = \frac{1}{\Delta t}\nabla\cdot\mathbf{u}, \tag{15}$$

where Δt is the size of the simulation time step. It has been shown in [Fedkiw et al. 2001] that imposing a feedback force field $-\nabla P$ to the velocity field strictly enforces zero divergence. Our technique first decomposes the pressure field into two components, $-\mu\Delta t\nabla^2 P$ and $P' = P + \mu\Delta t\nabla^2 P$, where μ is a constant factor with units of area divided by time, and the negative sign in front of the first component is due to the fact that a local maximum of P is typically a local minimum of $\nabla^2 P$. Note that the first component is the negative Laplacian of the pressure field, and the second one is a blurred version of the original pressure. This decomposition of the pressure field in turn splits the feedback force field into two components, $\mu\nabla(\Delta t\nabla^2 P)$ and $-\nabla P - \mu\nabla(\Delta t\nabla^2 P)$. To allow a certain degree of compressibility, especially in the regions with velocity constraints, our scheme only applies the first force component to the velocity field immediately while buffering the second one for later time steps. Thus, the discretized version of (14) changes to

$$\frac{\mathbf{u}(t+\Delta t) - \mathbf{u}(t)}{\Delta t} = -(\mathbf{u}\cdot\nabla)\mathbf{u} + \mu\nabla(\Delta t\nabla^2 P) + \mathbf{f}. \tag{16}$$

Because of (15), $\Delta t\nabla^2 P$ in (16) can be replaced with $\nabla\cdot\mathbf{u}$. If we further replace the left hand side of (16) with a continuous partial derivative of the velocity field, we arrive at the following single partial differential equation that is capable of simulating compressible gaseous phenomena:

$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u}\cdot\nabla)\mathbf{u} + \mu\nabla(\nabla\cdot\mathbf{u}) + \mathbf{f} \tag{17}$$

where $0 \le \mu < \infty$ with a typical value between 0 and 1. It is used to adjust the magnitude of the feedback force from the pressure field to the velocity field.

Since our scheme only uses a filtered version of the original pressure every time step, the feedback force from the pressure to the velocity field is weakened. Nevertheless, this does not mean a portion of the pressure simply disappears, but mean that the release of the energy preserved in the rest of the pressure is delayed. In Appendix B, we show that, for a bounded workspace without sources and sinks, $\nabla(\nabla\cdot\mathbf{u}) = 0$ everywhere is equivalent to $\nabla\cdot\mathbf{u} = 0$ everywhere. This result indicates that our new formulation of the feedback force from the pressure to the velocity field can eventually reduce the divergence of the velocity to zero when there are no external forces. In addition, our formulation allows large time steps and does not involve a sound speed. While other decompositions are possible, the Laplacian operator in our pressure decomposition makes it particularly convenient to robustly obtain the components through a diffusion process, which will be discussed in Section 7.2.

In practice, we have found that our formulation can effectively reproduce realistic fluid motion as well as incorporate frequently inserted velocity constraints without generating obvious visual discontinuities (see Section 8.1).

7 Implementation

7.1 Shape Transition

7.1.1 Variational Shape Morphing and Interpolation

As our first option, we apply the shape transformation method introduced in [Turk and O'Brien 1999] to generate a morph sequence between the shape of a smoke region and a target object. This method represents the whole morph sequence as a variational implicit function defined in a space of $n+1$ dimensions. This implicit function, which is based on radial basis functions, interpolates the source and target shapes defined in an n-dimensional space and handles topological shape changes automatically. The extra dimension is aligned with the temporal axis so that the source and target shapes are n-dimensional slices of the implicit function at time zero and one, respectively. In our situation, we apply this method to obtain an interpolating 4-D function $\Phi(\mathbf{r}, s)$ such that $\Phi(\mathbf{r}, 0)$ and $\Phi(\mathbf{r}, 1)$ reproduce the starting smoke shape and the target object, respectively. By fixing s to a value between 0 and 1, $\Phi(\mathbf{r}, s)$ represents the analytic from of an intermediate shape. In practice, we found that the rate of shape transformation usually was not uniform and the target shape started to loom only when the time becomes very close to one if the target shape is complicated and the parameter s is scheduled linearly with respect to the actual time scale. Therefore, we decided to warp the temporal axis and schedule s as a piecewise linear function of the actual time t used for an animation. The local slope of the piecewise linear function is used to adjust the transformation rate and make it perceptually more uniform. Better morph sequences can be obtained when the source and target objects are well aligned in terms of position and orientation. The amount of relative translation and rotation should then be uniformly distributed back across the frames of the resulting sequence so that there is simultaneous deformation, translation and rotation from frame to frame. It is desirable for the intermediate shapes to have an approximately constant volume since a gas usually does not significantly change volume even when it is compressible. Therefore, we verify the volume at each intermediate frame, and apply morphological operations on the voxel grid to shrink or expand the shape.

The same variational implicit functions for shape transformation can be used for surface interpolation [Carr et al. 2001; Turk and O'Brien 2002]. Therefore, they can be used for generating an analytic implicit function to approximate a polygonal mesh model by interpolating the vertices of the mesh. The interpolating implicit function and the polygonal mesh have the same dimensionality. Guiding objects originally represented as polygonal meshes are converted into implicit functions in this way.

7.1.2 Shape Masking

Shape masking is an easier-to-implement alternative for generating shape transitions. In this scheme, the target object can be simply a fixed shape undergoing rigid-body motion. We also require an additional masking shape, $M(\mathbf{x}, t)$, whose scale, position and orientation can all vary with time. The guiding shape at each time step is defined to be the intersection between the target and masking objects. For example, the masking object can be a moving sphere with changing radius. At the beginning of a transition, the sphere has a tiny radius and does not have any overlap with the target object. The sphere then moves closer to the target object while increasing its radius so that the amount of overlap becomes larger. At the end, the sphere encompasses the target object and the transition is complete. One caveat with this scheme is that the volume of the guiding shape changes with time, which requires an increasing amount of smoke during the transition.

7.2 Numerical Smoke Simulation

We have implemented numerical solutions for two formulations, our new compressible fluid formulation and the incompressible and inviscid formulation in [Fedkiw et al. 2001]. We also implemented enhancements that allow the integration of our velocity constraints. The details of these enhancements will be introduced in the next section.

The implementation of the incompressible scheme consists of three basic steps [Fedkiw et al. 2001]: compute an intermediate fluid velocity field, $\{\mathbf{u}^*\}$, from (14) ignoring the pressure term by first adding external force times the time step, and then solving the advection part $(\mathbf{u} \cdot \nabla)\mathbf{u}$ by using semi-Lagrangian tracing [Staniforth and Cote 1991]; obtain the pressure P by solving the Poisson equation,

$$\nabla^2 P = \frac{1}{\Delta t} \nabla \cdot \mathbf{u}^*; \tag{18}$$

where Δt is the size of the time step; finally, subtract the gradient of P from the intermediate velocity,

$$\mathbf{u} = \mathbf{u}^* - \Delta t \nabla P. \tag{19}$$

To simulate our compressible fluid formulation, we still keep the first step unchanged, but revised the other two. Although our formulation does not involve pressure, directly solving Eq. (17) may not be stable. Introducing pressure during the numerical process improves stability. Therefore, we actually solve Eqs. (15) and (16) instead. In the above steps, after obtaining the pressure by solving the Poisson equation, we apply a diffusion process,

$$\frac{\partial P}{\partial t} = \mu \nabla^2 P, \tag{20}$$

and solve for the new pressure, P', using a stable implicit method,

$$\frac{P' - P}{\Delta t} = \mu \nabla^2 P' \tag{21}$$

where the Laplacian of the new pressure $\nabla^2 P'$ instead of $\nabla^2 P$ appears on the right hand side. The new pressure can be obtained by solving the sparse linear system arising from discretizing this equation. Finally, the intermediate velocity is updated as follows,

$$\mathbf{u} = \mathbf{u}^* + \mu \Delta t \nabla (\nabla^2 P'). \tag{22}$$

Note that solving the diffusion equation to obtain the new pressure is an extra step we performed during smoke simulation.

We discretize the workspace into a finite volume grid. Keep all vector components on the faces of the voxels, and retain all the scalar fields at the center of each voxel. The velocity at any point inside a voxel can be obtained by linearly interpolating each component of the velocity vector separately. Additional implementation details about discretization and estimation of finite differences can be found in [Fedkiw et al. 2001; Stam 1999].

Our overall numerical solution involves multiple (around three) time steps between two consecutive frames. Small and large object motion tracking are carried out in separate time steps. There is at most one optional time step dealing with large motion between two frames. And there may be one or more time steps for small motion tracking since multiple iterations may be necessary to achieve good boundary matching. In the following, we introduce the details of these time steps in the context of the numerical solution for the incompressible formulation in [Fedkiw et al. 2001]. It is straightforward to modify these details for our compressible fluid formulation.

A time step for small motion tracking has the following substeps:

- Compute the intermediate fluid velocity field $\{u^*\}$, detect the boundary of the smoke region using a density threshold and set up velocity constraints at the boundary by modifying $\{u^*\}$.

- Solve the final fluid velocity field using (18) and (19) while satisfying the constraints.

- Use the obtained velocity field to evolve the smoke density distribution applying the semi-Lagrangian method.

The optional time step for large motion between two frames is executed before the other time steps. It has the following substeps:

- Obtain the intermediate fluid velocity field and set up velocity constraints for large motion.

- Solve the final fluid velocity field while satisfying the constraints.

- Use the obtained velocity field to evolve the smoke outside the guiding object; use the guiding object to transport the smoke inside and the transported smoke density overwrites existing density at each destination voxel.

If we choose to apply our compressible fluid formulation, the second substeps in both types of time steps should solve equations (18), (21) and (22) instead. It is straightforward to apply additional vorticity confinement [Fedkiw et al. 2001] in the second substeps as well since these substeps are basically the major part of a single forward simulation step. In our experiments, we always apply vorticity confinement.

7.3 Enforcing Velocity Constraints

Most of the time, we only set up and enforce velocity constraints at a selected subset of the voxels on the smoke boundary for small motion tracking. This process is initiated once the adopted error metric (see Section 5.3) exceeds a prescribed threshold. A larger threshold produces more lively smoke motion while a smaller threshold shows object boundary more clearly. The voxels on the smoke boundary are sorted according to their distance to the guiding object. Voxels with larger distance are assigned with higher priority. Velocity constraints are assigned to the boundary voxels with highest priority first. If the shape discrepancy does not drop significantly, additional voxels with next level of priority become constrained. This process continues until the error drops below the threshold or most of the boundary voxels are constrained. Once the error becomes sufficiently small, constrained voxels get released gradually in the reversed order.

The desired velocity constraints are imposed after the computation of the intermediate velocity field $\{\mathbf{u}^*\}$. Solving the Poisson equation may alter the velocities in these constraints and make it difficult for the smoke to converge to the target shape if these constraints are not explicitly enforced. Enforcing these constraints means the constrained velocities should not be affected during this step. Eq. (19) indicates that ∇P should be zero at the constrained voxels. Therefore, the pressure p needs to satisfy such gradient constraints in addition to the Poisson equation.

Using finite differences to discretize both left and right hand sides of (18), we obtain a sparse linear system of equations, $\mathbf{AP} = \mathbf{b}$ where $\mathbf{P}$ is the vector of unknown pressures and $\mathbf{A}$ is the coefficient matrix with a sparse structure. A discrete gradient estimation at a voxel involves three or six of its direct neighbors depending on whether central differences are being used. For symmetry consideration, when $\nabla P = 0$ at voxel (i, j, k), we impose that $P(i, j, k)$ should equal the pressure at its six direct neighbors. For every equation involving $P(i, j, k)$ in the linear system, we need to replace $P(i, j, k)$ with the pressure at one of its direct neighbors. Eventually, we can eliminate $P(i, j, k)$ from all the equations. Changes like this result in a system with a reduced number of unknown variables. The conjugate gradient method [Press et al. 1988] is the natural choice to solve the resulting linear system.

Since the smoke boundary surface is typically a closed surface, we avoid strictly enforcing boundary conditions everywhere on the surface because that would result in a singular matrix $\mathbf{A}$ if the volume inside the smoke region is not preserved. When it is necessary to cover the whole smoke boundary with constraints, we reduce the density of the constraints by setting a small distance threshold d_t. Any voxels within that distance from a constrained voxel are not allowed to be constrained. We typically choose d_t to be between 1 and 3 voxels. Alternatively, Dirichlet boundary conditions can be adopted by directly specifying pressure values on the smoke boundary. For example, one can force voxel (i, j, k) and all its direct neighbors to have zero pressure.

7.4 Smoke Transportation for Large Motion

We assume that the frame-to-frame motion of the guiding object can be decomposed into a rigid-body transformation and a nonrigid deformation. Smoke transportation only handles large rigid transformation and leaves small rigid transformation as well as additional nonrigid deformation to small motion tracking. To move the smoke to a new position and orientation specified by the rigid transformation, we apply backward mapping with a modified trilinear interpolation scheme. A voxel inside the transformed object takes the smoke density at a corresponding position in the original object. The density at the corresponding position should be interpolated only from the densities at those surrounding voxels that are also inside the original object to preserve the object boundary.

7.5 Thin Parts

It is hard for the smoke surface to reach some of the thin parts of the guiding object. That is because the gradient and curvature estimations are inaccurate at these locations due to insufficient sampling. We explicitly label those voxels that should have smoke but actually does not after a long time. If there is a smoke-filled voxel which is adjacent to one of those tagged voxels, we set a velocity constraint at the first voxel and the velocity points to the tagged voxel. Thus, smoke can be propagated gradually into the thin parts.

7.6 Time Complexity

Since we enforce shape matching by embedding velocity constraints into a conventional smoke simulation, the time complexity of our approach is slightly higher than the latter. The extra work at each time step includes obtaining the signed distance function of the guiding object and solving the diffusion equation for the pressure among others. The complexity of the distance transform is $O(n \log m)$ where n is the number of voxels in the grid and m is the maximum number of voxels on the propagating front [Sethian 1999]. In the worst case, $m = O(n)$ which usually does not occur unless the object boundary becomes close to a fractal surface. In all our experiments, the actual running time of the distance transform is less than an original smoke simulation step without constraints. The complexity of solving the sparse linear system arising from the diffusion equation is on the same order as that of solving the Poisson equation which is part of the original simulation. Solving these two equations is the most time-consuming step which occupies approximately 60% of the total running time. The complexity of all the other extra work is linearly proportional to the size of the voxel grid. Therefore, in practice, our complete algorithm maintains the same order of magnitude of the original complexity of smoke simulation.

8 Experimental Results

We have successfully tested our complete algorithm in a variety of examples. Some of the images from these examples are shown in Fig. 1 and Fig. 4-8. The target objects in these examples include both free-form objects and letters. Most of the target objects are converted to implicit functions from triangular meshes. If there are multiple objects in the same example, smoke simulation was performed on all of them simultaneously. Most of the simulations have been finished on an AMD 2100+ processor. A voxel grid of size between 64x64x64 and 200x200x200 has been adopted for the simulations. The examples took 15 seconds/frame on the lower end of the grid resolution and 60 seconds/frame on the higher end. The final images were rendered by ray-tracing the smoke volume density distribution [Kajiya and von Herzen 1984].

The density threshold τ for smoke boundary detection is interactively determined at the beginning of the smoke-object shape transition, and fixed throughout a whole simulation. We have found that typically this threshold falls between 0.1 and 0.5. In the following, we specify other parameters used during experiments for a normalized grid with one unit length for each of its dimensions. In (4), we

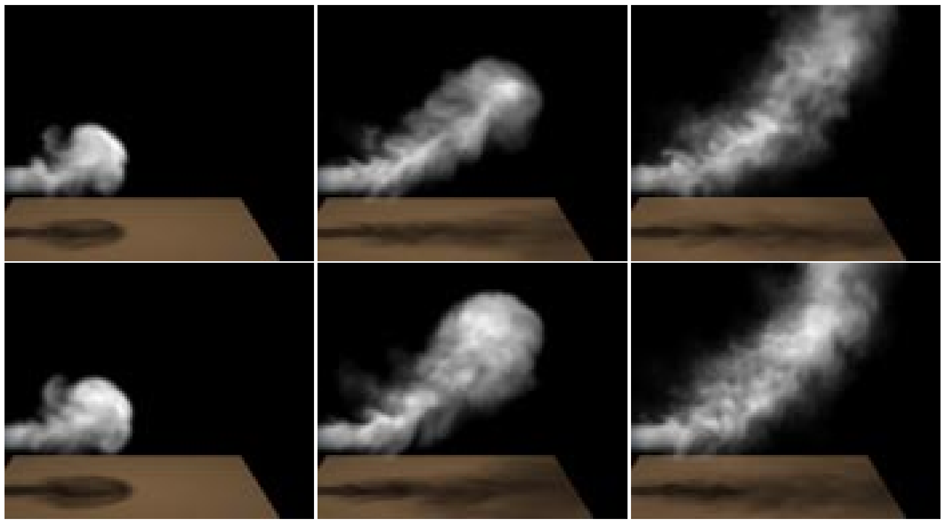


Figure 3: A comparison between two smoke formulations. The first row is generated by a model enforcing incompressibility. The second row is generated by our model with compressibility. They are visually similar, which indicates our model can also produce realistic smoke appearances. Grid size 200x200x200.

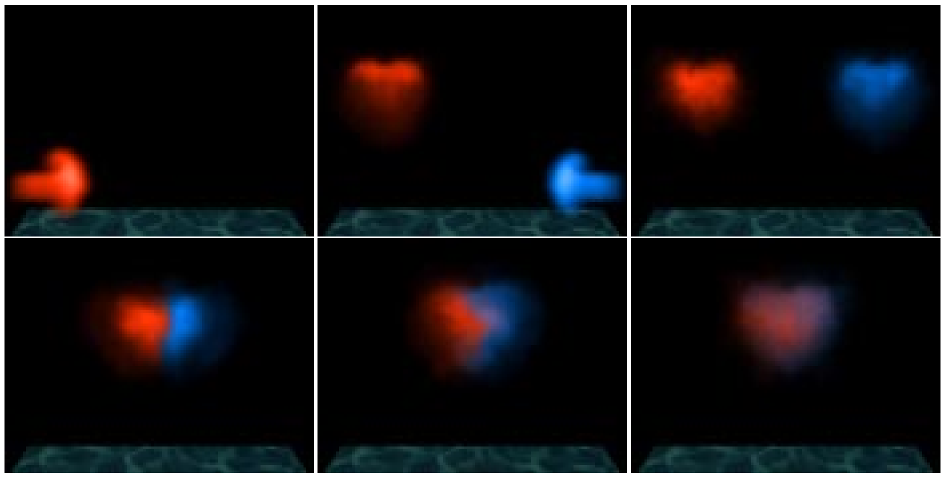


Figure 4: Two heart-shaped smoke objects collide and merge into one. Grid size 128x128x128.

typically set d_{max} between 0.05 and 0.1, and set C_n around 25. In (5), we set C_t around 2. These parameters were chosen with the assumption that the maximum velocity of the smoke is around 1 unit length per second. Obviously, by adjusting the maximum velocity, we can change the overall pace of the smoke.

8.1 Validation of the Compressible Fluid Model

We have validated our empirical fluid model by visually comparing the results from our model with results from the model in [Fedkiw et al. 2001]. We have performed two different types of comparisons. The first set of comparisons are between smoke simulations without velocity constraints. They are meant to verify whether our model can be used for a general smoke simulation as opposed to the special application in this paper. As a result, the visual quality of the smoke sequences generated by our model is comparable to those in [Fedkiw et al. 2001]. It can produce realistic smoke appearances with rolling effects. Fig. 3 shows one comparison in this category.

The second type of comparisons are between simulations with our derived velocity constraints. We have confirmed from these comparisons that our new model can integrate these velocity constraints better with much less visual artifacts. Introducing our velocity constraints into a model enforcing incompressibility would generate obvious artifacts since the nonzero divergence caused by the constraints influence surrounding velocities immediately. The reason our model can perform better is that it can absorb velocity discontinuities into pressure and release the energy in the pressure gradually. (Please find this comparison in the accompanying videos since these artifacts are easily visible in animations, but not in still images.)

8.2 Interaction with the Environment

The smoke objects should interact with the environment like smoke. When there is a sufficiently strong wind, the animator can choose to release the velocity constraints in the influence region of the wind and let the smoke move freely. When there is an intervening real object, the animator can release the original velocity constraints in the contact region and let the smoke be controlled by the boundary conditions on the real object surface as in [Fedkiw et al. 2001]. In the example shown in Fig. 1(a), a strong wind blows away the head of a smoke horse. However, the head grows back when the wind recedes. This is because we keep the underlying target object unchanged, and the target object induces velocity constraints pointing towards the head region. In such examples, the animator needs to specify the spatial region where constraints should be released as well as the starting time and duration. In Fig. 1(a), the user simply places a partitioning plane at the bottom of the horse's neck. Constraints on the same side of the plane as the head are released for a short period of time. To preserve temporal coherence, the number of constraints can be decreased or increased gradually to produce a smooth transition.

8.3 Interaction between Smoke Objects

When two smoke objects collide, their density distributions overlap and they should naturally merge into a larger smoke region since they do not have hard boundaries. Other scalar fields, such as temperature and color, can also be advected by semi-Lagrangian tracing and blended together in the same way as the density. The underlying guiding objects should either be merged or be replaced with a new one which may require a new shape transition. Velocity constraints should only be imposed on the new boundary afterwards. In the example shown in Fig. 4, two heart-shaped smoke objects merge into a larger one and their original colors get advected and blended together. The heart-shaped target object is defined by a closed-form implicit function: $(2x^2 + y^2 - z^2 - 1)^3 - 0.1x^2z^3 - y^2z^3 = 0$.

8.4 Large Motion

To demonstrate large motion tracking, we use a synthetic character equipped with motion-captured data as the underlying guiding object. The limbs of this articulated character have large frame-to-frame motion so that a small amount of smoke escapes. We also put a few smoke sources on the floor. The results from this example are shown in Fig. 5. In general, smoke does not move very fast. Its appearance would look less natural if we force the smoke to follow fast motion. Nevertheless, it can be clearly seen that the velocity constraints induced by the fast motion of the limbs creates an interesting velocity field for the rest of the environment and the smoke around the character follows this velocity field.

8.5 Tradeoff between Control and Smoke Appearance

The number of velocity constraints on the smoke boundary is the most important factor affecting the realism of the smoke simulation because our velocity constraints are kinematic constraints that do not involve dynamics. The shape of the smoke region is free to evolve without any constraints, but is guaranteed to match the guiding object with dense constraints. Thus, we can achieve various levels of tradeoff between control and smoke appearance by changing the number of constraints. However, the user does not have direct control over the number of constraints which is automatically adjusted at each time step according to the error metrics, the error thresholds and other parameters, as discussed in Section 5.3 and 7.3. There are no velocity constraints at all if the shape discrepancy is below the error threshold(s); otherwise, the number of constraints is increasing and realism is partially given up for control. Therefore, in general, larger error thresholds allow more realistic smoke appearance, but less object structures. The parameter μ in Eq. (17) can also affect the liveliness of the smoke. Larger μ's provide stronger force feedback from the pressure and typically lead to more lively motion. Fig. 9 gives a comparison of the visual quality generated by different combinations of error metrics, thresholds and μ. In this experiment, we chose to use both e_v and e_{L^∞} simultaneously. For e_v, the threshold is set between 0% and 5% of the volume of the target object. For e_{L^∞}, the error threshold is set between 0 and 4 voxels in a 128x128x128 grid. μ is set between 0.1 and 0.3.

8.6 Comparisons with other Control Schemes

A simple solution to our problem is to perform smoke simulation in the volume enclosed by the boundary surface of the guiding implicit function. Obviously, the smoke is forced to follow the object when it moves. This solution is similar to our scheme for large motion, but the hard boundary used for large motion disappears once the smoke has been moved to the target location. There are a few limitations with this solution using a hard boundary. Even though the boundary surface is not visualized during final rendering, it will still be obvious since the smoke stops or reflects at the boundary. A smooth hard boundary for the smoke is not a realistic phenomenon. The second problem is that the smoke tends to be stationary without interesting motion inside a closed volume with a limited size. A side-by-side comparison is made between this simple scheme and our method in Fig. 6. In the results generated by our method, the smoke close to the boundary of the target object still has natural motion and advection.

Our control scheme also has a few advantages and differences if compared to the keyframe control scheme in [Treuille et al. 2003]. First, our scheme is more efficient by introducing velocity constraints instead of evaluating derivatives of the velocity field with respect to all control parameters. The time complexity of our method is on the same order of magnitude as a single smoke simulation without shape matching. Therefore, our method can work with a larger problem size. For example, Fig. 7 shows an example where the smoke switches back and forth twice between the shapes of a "check" and a "X" on a 128x128x128 grid. It took only 4 hours 15 minutes to generate the 1250-frame sequence. Second, our scheme allows the smoke to stay around a shape for an arbitrarily long time instead of only at sparse keyframes. Nevertheless, if we release the velocity constraints, the smoke becomes free to evolve (Fig. 8(d)). As shown in Fig. 1(a), we can also release some of the constraints and then regain control as we wish. Third, since we adopt the distance transform, our scheme is less likely to be stuck in local minima. A notable difference is that our algorithm is targeted at object shapes while the method in [Treuille et al. 2003] is designed for arbitrary density distributions at keyframes.

9 Conclusions and Future Work

We have presented a novel technique to control the density and dynamics of smoke (a gas phenomenon) so that the synthetic appearance of the smoke (gas) resembles a still or moving object. The main focus has been on controllability and appearance. In order to match the smoke surface with the target surface, we represent both smoke and objects as implicit functions and impose carefully designed velocity constraints derived from a shape matching functional. An empirical compressible fluid model has been introduced for effectively integrating constraints into the velocity field while maintaining realistic fluid appearances. The overall framework represents a significant advance over previous methods for controlling fluids. The implementation of our system is built upon recent advances in smoke simulation and shape transformation.

We would like to extend this work to 2D images and video objects. The difference between a 3D object and a 2D image is that an object is a binary function while an image has multiple levels of intensities or colors. Matching the smoke boundary with the object boundary should be extended to matching the level sets of smoke

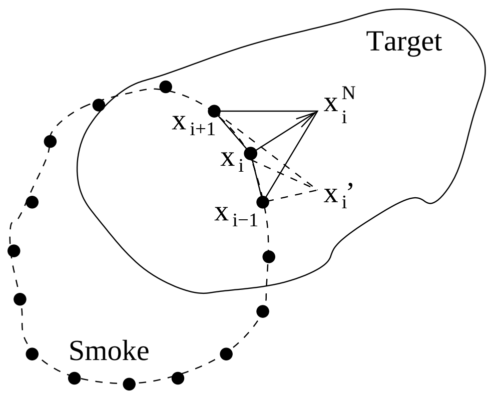


Figure 10: Moving points on the smoke boundary along their respective normal directions can reduce the shape discrepancy between the smoke region and the target object most quickly.

density with the level sets of image intensities or colors. Nevertheless, we expect the approach introduced here can be generalized to handle that circumstance by matching the boundaries of these level sets. Unlike 3D animated objects addressed in this paper, the motion of video objects is unknown, and needs to be solved using computer vision techniques such as optical flow [Brown 1992; Beauchemin and Barron 1995; Black and Anandan 1996]. Once the motion of these objects has been estimated, the approach in this paper can also be adapted to produce fluid appearances for them.

A Variational Gradient

In this section, we provide an intuitive derivation of (3) from (2) in a 2D space. The derivation for a 3D space can be obtained similarly. The smoke boundary is first discretized into a finite set of points. Consider one of the points $\mathbf{x}_i$ shown in Fig. 10. We need to perturb the position of $\mathbf{x}_i$ to reduce the integral in (2). Obviously, $\mathbf{x}_i$ should move further into the interior of the target object. If $\mathbf{x}_i$ moves along the local normal at $\mathbf{x}_i$ by an infinitesimal distance, (2) is reduced by an amount equal to the area enclosed by the polygon $\mathbf{x}_{i-1}\mathbf{x}_i\mathbf{x}_{i+1}\mathbf{x}_i^N$ where $\mathbf{x}_i^N$ represents the new location of $\mathbf{x}_i$. If $\mathbf{x}_i$ moves along some other arbitrary direction to $\mathbf{x}_i'$ by the same infinitesimal distance, (2) is reduced by an amount equal to the area enclosed by the polygon $\mathbf{x}_{i-1}\mathbf{x}_i\mathbf{x}_{i+1}\mathbf{x}_i'$. Since $\mathbf{x}_i\mathbf{x}_i^N$ is perpendicular to the local boundary, when the lengths of $\mathbf{x}_{i-1}\mathbf{x}_i$ and $\mathbf{x}_i\mathbf{x}_{i+1}$ become sufficiently small, the area of $\mathbf{x}_{i-1}\mathbf{x}_i\mathbf{x}_{i+1}\mathbf{x}_i^N$ is guaranteed to be larger than the area of $\mathbf{x}_{i-1}\mathbf{x}_i\mathbf{x}_{i+1}\mathbf{x}_i'$. That is, (2) can be decreased most quickly by moving $\mathbf{x}_i$ along the local normal of the smoke boundary. This also holds for all other points on the same boundary. By increasing the number of points in the boundary discretization, the distance between two adjacent points can be arbitrarily close to zero. By definition [Gelfand and Fomin 1963], the variational derivative of the functional in (2) with respect to the geometry of the smoke boundary exists. The variational derivative with respect to a specific boundary point is the unit normal vector at that point multiplied by an appropriate sign given in (3).

B An Equivalent Condition for $\nabla \cdot \mathbf{u} = 0$

We show that, for a bounded workspace Ω without sources and sinks, $\nabla(\nabla \cdot \mathbf{u}) = 0$ everywhere is equivalent to $\nabla \cdot \mathbf{u} = 0$ everywhere. Without loss of generality, suppose there exists a point $\mathbf{x_0} \in \Omega$ such that $\nabla \cdot \mathbf{u}(\mathbf{x_0}) > 0$. Since Ω is bounded,

$$\int_\Omega (\nabla \cdot \mathbf{u}) d\mathbf{x} = 0.$$

Therefore, there must be another point $\mathbf{x_1} \in \Omega$ such that $\nabla \cdot \mathbf{u}(\mathbf{x_1}) < 0$. Consider an arbitrary path between $\mathbf{x_0}$ and $\mathbf{x_1}$. There must exist a point $\mathbf{x_m}$ on that path such that $\nabla(\nabla \cdot \mathbf{u}(\mathbf{x_m})) \neq 0$. Otherwise, $\nabla \cdot \mathbf{u}(\mathbf{x_0}) = \nabla \cdot \mathbf{u}(\mathbf{x_1})$. Thus, $\nabla \cdot \mathbf{u}$ must be zero everywhere.

References

ALEXA, M., COHEN-OR, D., AND LEVIN, D. 2000. As-rigid-as-possible shape interpolation. In *SIGGRAPH 2000 Conference Proceedings*, 157–164.

BEAUCHEMIN, S., AND BARRON, J. 1995. The computation of optical flow. *ACM Computing Surveys 27*, 3, 433–467.

BLACK, M., AND ANANDAN, P. 1996. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding 63*, 1, 75–104.

BROWN, L. 1992. A survey of image registration techniques. *ACM Computing Surveys 24*, 325–376.

CARR, J., MITCHELL, T., BEATSON, R., CHERRIE, J., FRIGHT, W., MCCALLUM, B., AND EVANS, T. 2001. Reconstruction and representation of 3d objects with radial basis functions. In *SIGGRAPH 2001 Conference Proceedings*, 67–76.

CHORIN, A. 1967. A numerical method for solving incompressible viscous flow problems. *Computational Physics 2*, 12–26.

DESBRUN, M., AND CANI-GASCUEL, M.-P. 1998. Active implicit surface for animation. In *Proceedings of Graphics Interface*.

EBERT, D., MUSGRAVE, K., PEACHY, D., PERLIN, K., AND WORLEY, S. 1998. *Texturing and Modeling: A Procedural Approach (second edition)*. AP Professional.

ENRIGHT, D., MARSCHNER, S., AND FEDKIW, R. 2002. Animation and rendering of complex water surfaces. *ACM Transactions on Graphics 21*, 3, 736–744.

FEDKIW, R., STAM, J., AND JENSEN, H. 2001. Visual simulation of smoke. In *SIGGRAPH 01 Conference Proceedings*, 15–22.

FOSTER, N., AND FEDKIW, R. 2001. Practical animation of liquids. In *SIGGRAPH 2001 Conference Proceedings*, 23–30.

FOSTER, N., AND METAXAS, D. 1997. Controlling fluid animation. In *Proceedings of Computer Graphics International*, 178–188.

FOSTER, N., AND METAXAS, D. 1997. Modeling the motion of a hot, turbulent gas. In *SIGGRAPH 97 Conference Proceedings*, 181–188.

GARDNER, G. 1985. Visual simulation of clouds. In *Proc. of SIGGRAPH'85*, 297–304.

GELFAND, I., AND FOMIN, S. 1963. *Calculus of Variations*. Prentice-Hall.

GOMES, J., BEIER, T., COSTA, B., DARSA, L., AND VELHO, L. 1997. Warping and morphing of graphical objects. SIGGRAPH 97 course notes.

HUGHES, J. F. 1992. Scheduled fourier volume morphing. In *SIGGRAPH 92 Conference Proceedings*, 43–46.

KAJIYA, J., AND VON HERZEN, B. 1984. Ray tracing volume densities. *Computer Graphics (SIGGRAPH 84 Proceedings) 18*, 3.

KAPLER, A. 2002. Evolution of a vfx voxel tool. In *SIGGRAPH 2002 Sketches & Applications, Conference Abstracts and Applications*, 179.

LAMORLETTE, A., AND FOSTER, N. 2002. Structural modeling of flames for a production environment. In *SIGGRAPH 02 Conference Proceedings*, 729–735.

LERIOS, A., GARFINKLE, C., AND LEVOY, M. 1995. Feature-based volume metamorphosis. In *SIGGRAPH 95 Conference Proceedings*, 449–456.

MUSETH, K., BREEN, D., WHITAKER, R., AND BARR, A. 2002. Level set surface editing operators. *ACM Transactions on Graphics 21*, 3, 330–338.

MUSGRAVE, F., 1997. Great balls of fire. www.wizardnet.com/musgrave/balls_o_fire.ps.

OSHER, S., AND FEDKIW, R. 2001. Level set methods: an overview and some recent results. *Computational Physics 169*, 2.

OSHER, S., AND SETHIAN, J. 1988. Fronts propagating with curvature dependent speed: Algorithms based on hamilton-jacobi formulations. *Computational Physics 79*, 12–49.

PERLIN, K., AND HOFFERT, E. 1989. Hypertexture. In *Proc. of SIGGRAPH'89*, 253–262.

PERLIN, K. 1985. An image synthesizer. In *Proc. of SIGGRAPH'85*, 287–296.

PRESS, W., FLANNERY, B., TEUKOLSKY, S., AND VETTERLING, W. 1988. *Numerical Recipes in C*. Cambridge Univ. Press, New York.

SETHIAN, J. 1999. *Level Set Methods and Fast Marching Methods*. Cambridge University Press.

SIMS, K. 1992. Choreographed image flow. *Journal of Visualization and Computer Animation 3*, 31–43.

STAM, J. 1995. *Multi-Scale Stochastic Modeling of Complex Natural Phenomena*. PhD thesis, University of Toronto.

STAM, J. 1999. Stable fluids. In *SIGGRAPH 99 Conference Proceedings*, 121–128.

STANIFORTH, A., AND COTE, J. 1991. Semi-lagrangian integration schemes for atmospheric models: A review. *Monthly Weather Review 119*, 2206–2223.

TREUILLE, A., MCNAMARA, A., POPOVIC, Z., AND STAM, J. 2003. Keyframe control of smoke simulations. *ACM Trans. Graphics 22*, 3, 716–723.

TURK, G., AND O'BRIEN, J. 1999. Shape transformation using variational implicit functions. In *SIGGRAPH 99 Conference Proceedings*, 335–342.

TURK, G., AND O'BRIEN, J. 2002. Modelling with implicit surfaces that interpolate. *ACM Transactions on Graphics 21*, 4, 855–873.

WOLBERG, G. 1990. *Digital Image Warping*. IEEE Computer Society Press.

YNGVE, G., O'BRIEN, J., AND HODGINS, J. 2000. Animating explosions. In *SIGGRAPH 2000 Conference Proceedings*, 29–36.

ZHAO, H.-K., OSHER, S., AND FEDKIW, R. 2001. Fast surface reconstruction using the level set method. In *1st IEEE Workshop on Variational and Level Set Methods*.

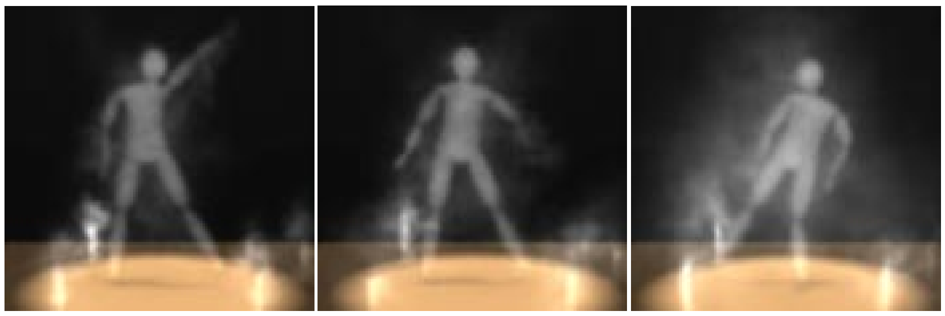


Figure 5: Large motion tracking for a synthetic character. Constraints for large motion induce a velocity field for the environment. There are a few sources on the floor to produce smoke for the environment. Grid size 128x128x128.

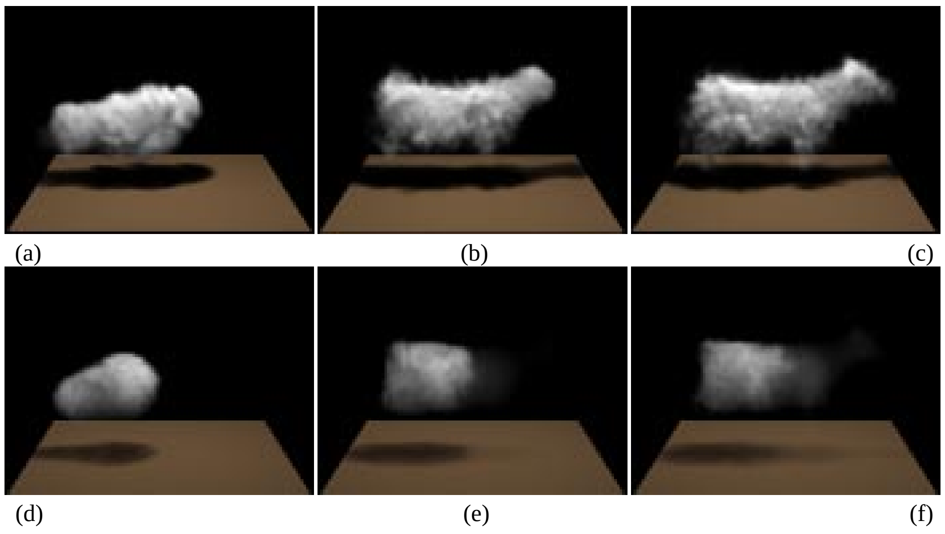


Figure 6: (a)-(c) A smoke cow generated by our method; (d)-(f) a smoke cow generated by the hard boundary scheme. Grid size 128x128x128.

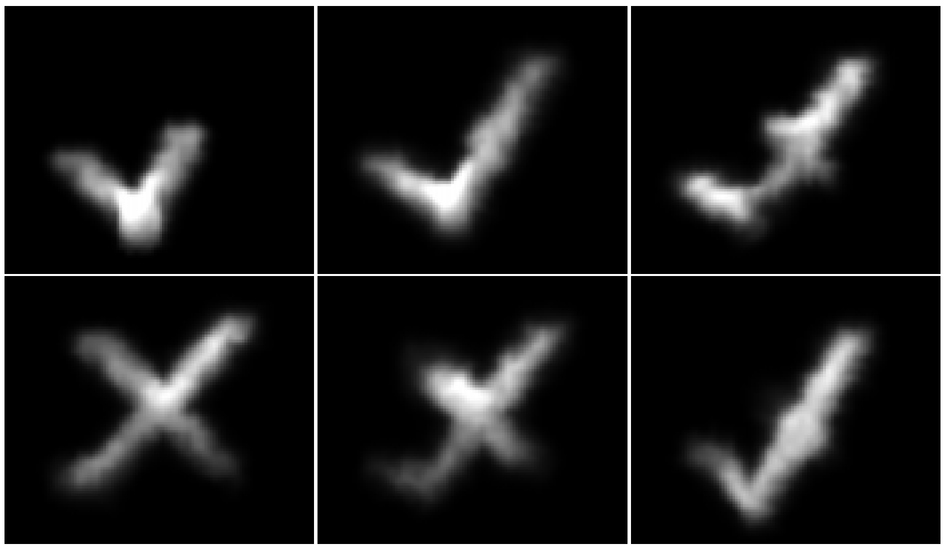


Figure 7: The smoke switches back and forth between the shapes of a ”check” and a ”X”. Only one cycle is shown here. Grid size 128x128x128.

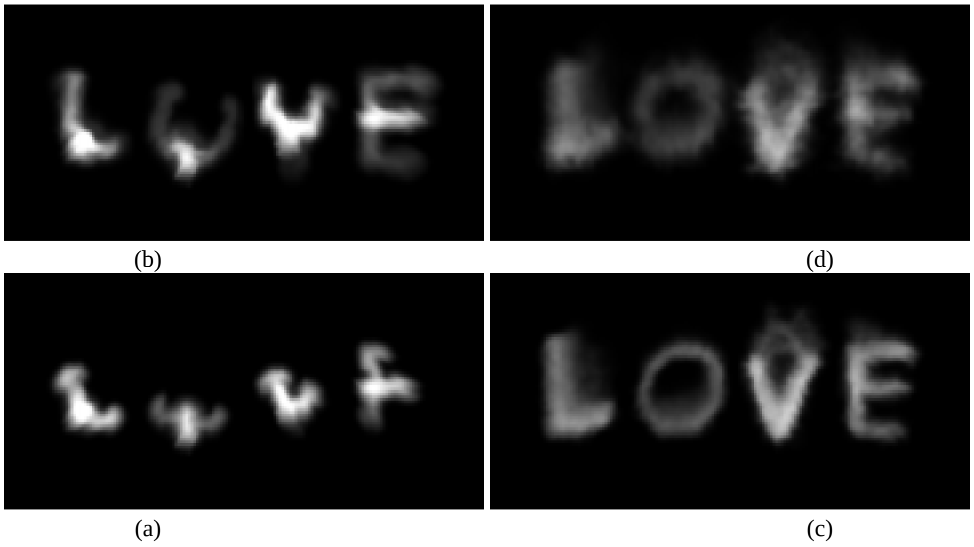


Figure 8: (a)-(d) Smoke rises to form the shape of four letters. The shapes start to disappear once the velocity constraints are released in (d). Grid size 128x128x128.

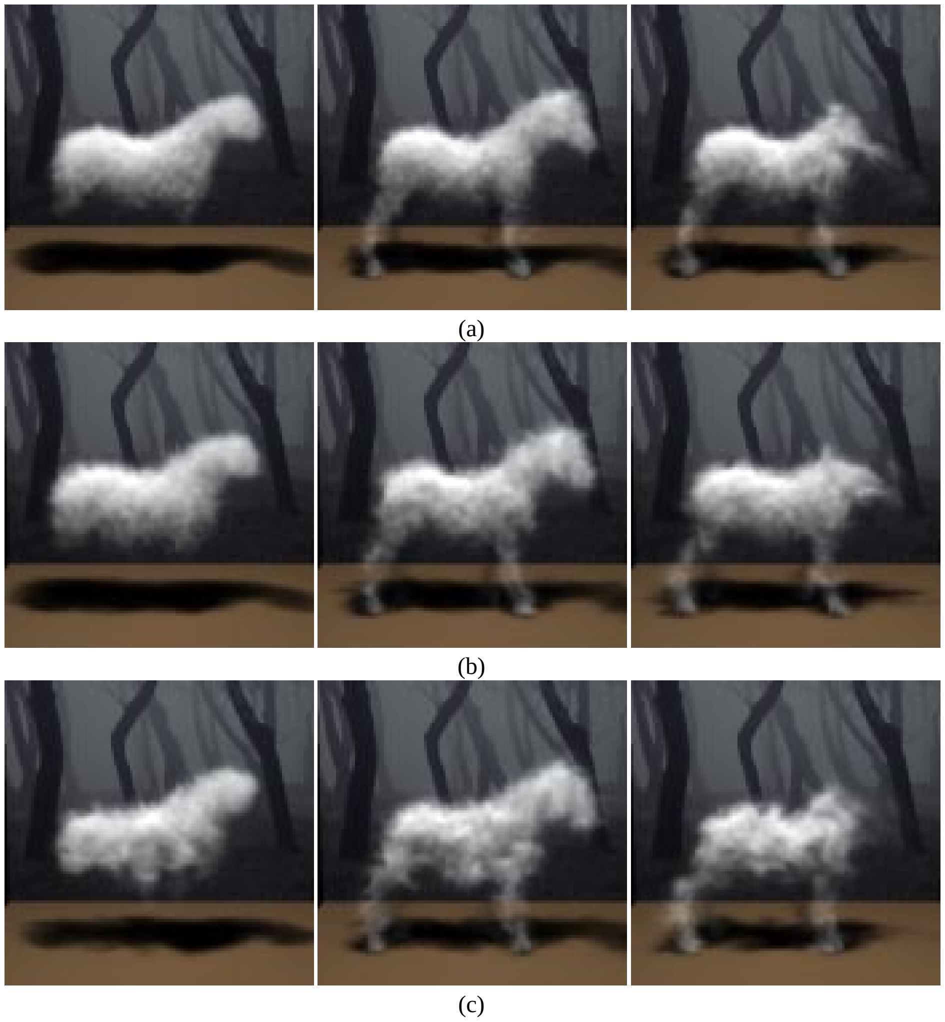


Figure 9: A comparison of the visual quality of the smoke objects by different combinations of error metrics, thresholds and the pressure diffusion parameter. (a) The error thresholds $e_v = 0\%$, $e_{L^\infty} = 0$ voxels, and the diffusion parameter $\mu = 0.1$; (b) $e_v = 3\%$, $e_{L^\infty} = 2.5$ voxels, and $\mu = 0.1$; (c) $e_v = 5\%$, $e_{L^\infty} = 4$ voxels, and $\mu = 0.3$. Grid size 128x128x128.