Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2009)
E. Grinspun and J. Hodgins (Editors)

# Guiding of Smoke Animations Through Variational Coupling of Simulations at Different Resolutions

Michael B. Nielsen[1,†], Brian B. Christensen[1,‡], Nafees Bin Zafar[2,§], Doug Roble[2,¶] and Ken Museth[1,2,3,‖]

[1]Aarhus University [2]Digital Domain [3]DreamWorks Animation

**Abstract**

*We propose a novel approach to guiding of Eulerian-based smoke animations through coupling of simulations at different grid resolutions. Specifically we present a variational formulation that allows smoke animations to adopt the low-frequency features from a lower resolution simulation (or non-physical synthesis), while simultaneously developing higher frequencies. The overall motivation for this work is to address the fact that art-direction of smoke animations is notoriously tedious. Particularly a change in grid resolution can result in dramatic changes in the behavior of smoke animations, and existing methods for guiding either significantly lack high frequency detail or may result in undesired features developing over time. Provided that the bulk movement can be represented satisfactorily at low resolution, our technique effectively allows artists to prototype simulations at low resolution (where computations are fast) and subsequently add extra details without altering the overall "look and feel". Our implementation is based on a customized multi-grid solver with memory-efficient data structures.*

Categories and Subject Descriptors (according to ACM CCS): Computer Graphics [I.3.7]: Animation—

## 1. Introduction

Grid-based Navier-Stokes solvers, like [Sta99, FSJ01], are commonly used for smoke effects in movie production, but they require significant CPU and memory resources. This is especially problematic in the context of visual effects since photorealistic smoke typically call for simulations on high-resolution grids. Simulation effects often require many iterations to create the desired look. In order to have quick turn-around times, artists often perform their art-direction iterations at a low resolution to determine the most effective simulation setup, and then run the simulations at final high resolutions. However, the change of resolution often completely changes the overall "look" of the animation, which may cause the composition to fail from the director's viewpoint [FGP07]. This property is especially evident when the initial grid resolution is very low and the solution has far from converged. It is primarily caused by the numerical viscosity of the discretization and is in turn further enhanced by the inherent non-linearity of the Navier-Stokes equations.

There is no set methodology for scaling simulation settings such that the high resolution simulation matches the look. Instead the artist has to intuit the alterations from experience and engage in an iterative process in high resolution as well.

The goal of this paper is to propose a technique for Eulerian grid-based smoke simulations [Sta99, FSJ01] that allows animators to use low-resolution input simulations to *guide* higher-resolution ones in such a way that details are added, but the overall (*i.e.* low frequency) flow is preserved as well as possible. Therein lies the assumption that the desired overall bulk movement is representable by the low resolution simulation , and our approach presupposes that this is the case. Our method is comprised of a physically based fluid simulation combined with a novel pressure projection step that minimizes the deviation of the velocity field's low frequencies from the low-resolution guiding flow, whilst constraining the velocity field to be divergence free.

The feasibility of guiding, or tracking, based on low res-

† e-mail:bang@cs.au.dk
‡ e-mail: bbc@cs.au.dk
§ e-mail:nafees@d2.com
¶ e-mail:doug@d2.com
‖ e-mail:museth@acm.org


SCA 2009, New Orleans, LA, August 1–2, 2009.
© 2009 ACM 978-1-60558-610-6/09/0008 $10.00

**Figure 1:** *A velocity, density, and temperature source injects hot high-velocity smoke subject to a buoyancy force.* ***Left:*** *Unguided simulation* $(256^3)$. ***Middle:*** *Unguided simulation* $(64^3)$. ***Right:*** $256^3$ *simulation guided by the* $64^3$ *simulation. There is poor resemblance between the unguided simulations. The guided simulation follows the general flow of the low resolution simulation, and adds dynamic high frequency detail.*

olution input simulations has previously been demonstrated by Bergou *et al.* [BMWG07] for thin shells and by Thürey *et al.* [TKPR06] for Lattice Boltzmann (LBM) and Smoothed Particle Hydrodynamics (SPH) simulations of liquids. Like Bergou *et al.* we analyze the problem in a mathematical framework and formulate guiding as a set of constrained equations. The motivation that led us to take this approach was that simpler strategies, considered to some extent by previous work, did not suffice. In particular upsampling a low-resolution velocity field followed by pressure projection (see Figure 5.c) or blending the velocity field with a guiding velocity field (see discussion in [TKPR06]) is incapable of producing higher frequencies and causes significant smoothing of high frequency detail, respectively. Another relatively simple strategy is to blend the low frequencies of the velocity field with a low-resolution guiding simulation before pressure projection. This is essentially the idea presented by Thürey *et al.* except that they consider guiding particles. However this approach may, in our experience (see section 8), introduce undesired features over time when applied to an Eulerian-based smoke simulation. We hypothesize that this is due in part to the loss of explicit control over the low frequencies during the standard pressure projection step.

Our proposed workflow consists of first applying existing fluid control methodologies to create a low resolution simulation. Next step is to invoke our framework with the low resolution simulation as input, possibly iteratively to control several frequency bands separately. Additional sub-grid motion details can then be generated using one of the recently developed turbulence synthesis methods [SB08, KTJG08, NSCL08].

Provided that an appropriate low-resolution simulation can be found, our framework allows the look development of the bulk movement to take place in low resolution. Thereby it offers a starting point in high resolution for control that pertains directly to high frequency features. In particular this paper claims the following novel contributions over previous work:

- A variational formulation of a guiding velocity field and a resulting set of linear equations.
- A practical implementation of our guiding framework, including methods for lowpass filter estimation and handling of boundaries.
- A custom multi-threaded multigrid implementation based on a fast and compact dynamic matrix storage format.

## 2. Related Work

The concept of high level animation control of full Navier-Stokes based simulations was first advocated by Foster and Metaxas [FM97]. Later Treuille *et al.* [TMPS03] proposed a gradient descent based optimization framework for keyframe control of smoke simulations. Specifically the framework optimizes for forces that result in the fluid assuming keyframed poses. McNamara *et al.* [MTPS04] improved the speed of this framework by several orders of magnitude by adopting the adjoint method. While powerful and generic, simulation time seems to limit the resolutions feasible with this approach, as an entire fluid simulation has to be run for each step in the optimization.

Thürey *et al.* [TKPR06] successfully demonstrated guiding simulations for SPH and LBM. In particular, they propose a fast technique for controlling low frequencies of liquid animations by applying a combination of feedback and attraction forces. Contrary to our work, [TKPR06] is force-based, which (as noted in their paper) makes it difficult to enforce constraints on the velocity field during pressure projection.Thürey *et al.* do not directly address this problem, but refer to the work of Shi *et al.* [SY05] for a possible solution. Shi *et al.* consider the problem of making liquid simulations follow rapidly changing target animations. They propose a force-based solution that is comprised of a velocity- and shape-feedback force as well as a potential function. The shape-feedback force of Shi *et al.* is designed to be divergence free, and is therefore unaffected by the pressure projection step. However, the shape-feedback force is specifically designed for target shapes and is not well suited for smoke simulations with no specific target shapes.

Recent work has also successfully focused on improving run times and memory usage of full Navier-Stokes simula-

tions by developing procedural methods that synthesize the high frequency detail, given a low resolution simulation as input [SB08, KTJG08, NSCL08]. Although the goal of our work bears similarities with these synthesis methods we emphasize that it is fundamentally different in that we partly simulate the high frequency detail. Our method also allows for non-physically based inputs which enhances the ability of animators to art-direct fluid animations. We stress that our work in no way precludes the procedural synthesis methods, but is meant to work with such techniques to produce high resolution simulations quickly.

Concurrently with our work, Mullen *et al.* [MCP*09] developed energy preserving integrators for simplicial grids. These integrators can also be applied to obtain higher resemblance between low and high resolution simulations.

## 3. Algorithm Overview

The flow of an inviscid, incompressible fluid is described by the inviscid Euler equations $\frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v}\cdot\nabla)\mathbf{v} = -\nabla p + \mathbf{f}$ and $\nabla\cdot\mathbf{v} = 0$, where $\mathbf{v}$ denotes the velocity of the fluid, $p$ denotes pressure, $\mathbf{f}$ is an external force, and the fluid density is assumed to be 1 for simplicity. In computer graphics, this set of equations is often solved using the operator splitting approximation described in [Sta99]. The central idea is to decouple self-advection, addition of external forces and enforcement of incompressibility by sequentially solving for each of these terms. As illustrated in Figure 2, our method only requires modification of the pressure projection step that solves for the latter term. Specifically, our method proceeds as following for each iteration of the guided, high-resolution simulation:

1. Obtain the low-resolution, guiding velocity field $\mathbf{v}_{\text{low}}$ through simulation or art direction.
2. Upsample $\mathbf{v}_{\text{low}}$ to high resolution.
3. Advect and add forces using traditional methods to obtain the intermediate velocity field $\tilde{\mathbf{v}}$ in high resolution.
4. Obtain the guiding weights $\alpha$ through art direction or an automated process.
5. Solve a modified projection step using the upsampled $\mathbf{v}_{\text{low}}$, the guiding weights $\alpha$ and $\tilde{\mathbf{v}}$ to obtain the new high-resolution, guided velocity field $\mathbf{v}$.

The guiding weights $\alpha$ will be explained in the following section.

## 4. Variational Model of Guiding

In this section we analyze guiding velocity fields for fluid animation in a mathematical framework. As discussed in the introduction, existing simpler methods do in our experience not suffice. Motivated by this, we consider guiding of fluids as a constrained variational problem. Furthermore we derive the resulting linear equations that solve for the stationary

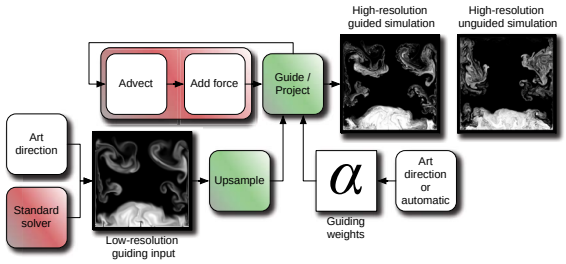


**Figure 2:** *Our method consists of two components (in green) which are easily integrated in a traditional fluid simulation pipeline (in red). The low-resolution input velocity field is upsampled for the guided projection step, which ensures a similar bulk movement of the high-resolution simulation but with added detail. An unguided simulation of high resolution is provided for reference.*

point of this variational problem. We first derive the variational problem continuously but could equally well have derived the linear equations directly from a specific discretization. As a prelude we briefly summarize how the pressure projection step can be regarded as a variational problem.

### 4.1. Preliminaries

The Poisson equation

$$\nabla\cdot\tilde{\mathbf{v}} = \Delta p \tag{1}$$

combined with $\mathbf{v} = \tilde{\mathbf{v}} - \nabla p$ enforces the continuity condition of the inviscid Euler equations, where $\mathbf{v}$ and $\tilde{\mathbf{v}}$ are velocity fields, $\mathbf{v}$ is divergence free and $p$ is pressure. It can be derived from the minimization of the difference between two velocity fields $\mathbf{v}$ and $\tilde{\mathbf{v}}$ subject to the continuity constraint that $\mathbf{v}$ be divergence free (see [FP02] pp. 202-204). Mathematically this amounts to minimizing

$$\frac{1}{2}\int_\Omega |\mathbf{v}(\mathbf{r}) - \tilde{\mathbf{v}}(\mathbf{r})|^2\, d\mathbf{r} \tag{2}$$

subject to the constraint

$$\nabla\cdot\mathbf{v}(\mathbf{r}) = 0 \tag{3}$$

where $\mathbf{r}$ is the position vector and $\Omega$ is the fluid domain. Eq. (2) and Eq. (3) can be combined into the saddle point problem

$$\int_\Omega \left\{ \frac{1}{2}|\mathbf{v}(\mathbf{r}) - \tilde{\mathbf{v}}(\mathbf{r})|^2 - \lambda(\mathbf{r})\nabla\cdot\mathbf{v}(\mathbf{r}) \right\} d\mathbf{r} \tag{4}$$

where $\lambda(\mathbf{r})$ are scalar-valued Lagrange multipliers. Taking the first variation and solving for a stationary point, one obtains Eq. (1) and Eq. (3) with $\lambda$ replacing $p$ (hence "'pressure"' is, in fact, a Lagrange multiplier).

### 4.2. Guiding Equations

To let a given low-resolution velocity field, $\mathbf{v}_{\text{low}}$, guide a high-resolution velocity field, $\mathbf{v}$, we add an additional term to

Eq. (2) and Eq. (3). The new term prescribes that a *smoothed version of the high-resolution velocity field should be as close as possible to the low-resolution input velocity field upsampled to high resolution*. Enforcing that the convolution of the high-resolution velocity field should be *identical* (by imposing a constraint) to the low-resolution input velocity field upsampled to high resolution leads to ill-posed problems for most lowpass filters (*i.e.* a (unique) solution does not exist). Mathematically, our term is formulated as

$$R = \frac{1}{2}\int_\Omega \left|\left[\mathcal{F} * \mathbf{v}\right](\mathbf{r}) - \mathbf{v}_{\text{low}}(\mathbf{r})\right|^2 d\mathbf{r} \tag{5}$$

where $\mathcal{F}$ is a, possibly spatially varying, lowpass filter, $*$ denotes a convolution and $\mathbf{v}_{\text{low}}$ is the low-resolution input velocity field upsampled to high resolution. Next we combine Eq. (4) and Eq. (5) into the saddle point problem

$$\int_\Omega \left\{ \frac{\alpha(\mathbf{r})}{2}\left|\mathbf{v}(\mathbf{r}) - \tilde{\mathbf{v}}(\mathbf{r})\right|^2 - \lambda(\mathbf{r})\nabla\cdot\mathbf{v}(\mathbf{r}) + \frac{(1-\alpha(\mathbf{r}))}{2}\left|\left[\mathcal{F}*\mathbf{v}\right](\mathbf{r}) - \mathbf{v}_{\text{low}}(\mathbf{r})\right|^2 \right\} d\mathbf{r} \tag{6}$$

where $\alpha \in (0;1]$ is a scaling parameter that determines the relative weight of each of the terms. In section 8 we will show how $\alpha$ can be used to introduce artistic control. For now we make the following observations; 1) $\alpha$ can vary both spatially and temporally, 2) for $\alpha = 1$ we obtain a normal unguided fluid simulation and 3) the solution to the saddle point problem, $\mathbf{v}$, is divergence free due to the constraint enforced by the Lagrange multipliers. To solve this saddle point problem for $\mathbf{v}$ we employ the calculus of variations. This amounts to deriving the stationary points of Eq. (6) that have zero first variation. Here we focus on the first variation, $\delta R$, of our term $R$ given by Eq. (5), and next combine with the first variation of Eq. (4) considered in [FP02]. Assume in the following that the velocity field $\mathbf{v}$ gives rise to the desired stationary point, $R_{\text{stationary}}$. That is, $R_{\text{stationary}} = R(\mathbf{v})$. Next we consider $\delta\mathbf{v}$ to be the variation of $\mathbf{v}$, and ignore second order terms (*i.e.* terms of the kind $\delta\mathbf{v}^2$) that do not contribute to the first variation. The first variation, $\delta R$, then reads as

$$\begin{aligned}
\delta R &= R - R_{\text{stationary}} = R(\mathbf{v}+\delta\mathbf{v}) - R(\mathbf{v}) \\
&= \int_\Omega \left[\mathcal{F}*\delta\mathbf{v}\right](\mathbf{r}) \cdot \left[\left[\mathcal{F}*\mathbf{v}\right](\mathbf{r}) - \mathbf{v}_{\text{low}}(\mathbf{r})\right] d\mathbf{r} \\
&= \int_\Omega \delta\mathbf{v}(\mathbf{r}) \cdot \int_\Omega \mathcal{F}(\mathbf{q}-\mathbf{r})\left[\left[\mathcal{F}*\mathbf{v}\right](\mathbf{q}) - \mathbf{v}_{\text{low}}(\mathbf{q})\right] d\mathbf{q} d\mathbf{r}
\end{aligned}$$

which holds from the definition of convolution, the linearity of the integral, and the definition and distributivity of the dot product. The integration variable $\mathbf{q}$ arises from the expansion of $\mathcal{F}*\delta\mathbf{v}$ into its integral definition. Since $\delta\mathbf{v}(\mathbf{r})$ is arbitrary in all points of the domain, and $\delta R = 0$ for a stationary point, we obtain the following necessary and sufficient condition for a stationary point of Eq. (5):

$$\int_\Omega \mathcal{F}(\mathbf{q}-\mathbf{r})\left[\left[\mathcal{F}*\mathbf{v}\right](\mathbf{q}) - \mathbf{v}_{\text{low}}(\mathbf{q})\right] d\mathbf{q} = 0$$

Combining this integral with the integral obtained for the first variation of Eq. (4), derived in [FP02], and including the scaling parameter $\alpha > 0$, a necessary and sufficient condition for a stationary point of Eq. (6) is:

$$\begin{aligned}
&\mathbf{v}(\mathbf{r}) + \tfrac{1}{\alpha(\mathbf{r})}\nabla\lambda(\mathbf{r}) + \tfrac{1}{\alpha(\mathbf{r})}\int_\Omega (1-\alpha(\mathbf{q}))\mathcal{F}(\mathbf{q}-\mathbf{r})\left[\mathcal{F}*\mathbf{v}\right](\mathbf{q}) d\mathbf{q} \\
&\quad = \tilde{\mathbf{v}}(\mathbf{r}) + \tfrac{1}{\alpha(\mathbf{r})}\int_\Omega (1-\alpha(\mathbf{q}))\mathcal{F}(\mathbf{q}-\mathbf{r})\mathbf{v}_{\text{low}}(\mathbf{q}) d\mathbf{q}
\end{aligned} \tag{7}$$

where the right hand side is known (the reason for dividing by $\alpha$ is that the implementation of Eq. (8) simplifies). The linear system obtained by discretizing and combining Eq. (7) with the constraints in Eq. (3) is however not well suited for relaxation methods such as Gauss Seidel which forms part of a multigrid implementation. This is due to the fact that Gauss Seidel employs a division by the diagonal entry, which is zero in the lower part of the matrix corresponding to Eq. (3) since no terms including $\lambda$ are present. In order to obtain a linear system with non-zero diagonal entries in all rows, we apply the constraint Eq. (3) to Eq. (7) instead of discretizing Eq. (3) explicitly:

$$\begin{aligned}
&\nabla\cdot\left(\tfrac{1}{\alpha(\mathbf{r})}\nabla\lambda(\mathbf{r}) + \tfrac{1}{\alpha(\mathbf{r})}\int_\Omega (1-\alpha(\mathbf{q}))\mathcal{F}(\mathbf{q}-\mathbf{r})\left[\mathcal{F}*\mathbf{v}\right](\mathbf{q}) d\mathbf{q}\right) \\
&\quad = \nabla\cdot\left(\tilde{\mathbf{v}}(\mathbf{r}) + \tfrac{1}{\alpha(\mathbf{r})}\int_\Omega (1-\alpha(\mathbf{q}))\mathcal{F}(\mathbf{q}-\mathbf{r})\mathbf{v}_{\text{low}}(\mathbf{q}) d\mathbf{q}\right)
\end{aligned} \tag{8}$$

where we have used that $\nabla\cdot\mathbf{v}(\mathbf{r}) = 0$. If $\alpha$ is not spatially varying, Eq. (7) becomes

$$\begin{aligned}
&\mathbf{v}(\mathbf{r}) + \tfrac{1}{\alpha}\nabla\lambda(\mathbf{r}) + \tfrac{(1-\alpha)}{\alpha}\int_\Omega \mathcal{F}(\mathbf{q}-\mathbf{r})\left[\mathcal{F}*\mathbf{v}\right](\mathbf{q}) d\mathbf{q} \\
&\quad = \tilde{\mathbf{v}}(\mathbf{r}) + \tfrac{(1-\alpha)}{\alpha}\int_\Omega \mathcal{F}(\mathbf{q}-\mathbf{r})\mathbf{v}_{\text{low}}(\mathbf{q}) d\mathbf{q}
\end{aligned} \tag{9}$$

and Eq. (8) simplifies to

$$\begin{aligned}
&\tfrac{1}{\alpha}\Delta\lambda(\mathbf{r}) + \tfrac{(1-\alpha)}{\alpha}\nabla\cdot\int_\Omega \mathcal{F}(\mathbf{q}-\mathbf{r})\left[\mathcal{F}*\mathbf{v}\right](\mathbf{q}) d\mathbf{q} \\
&\quad = \nabla\cdot\tilde{\mathbf{v}}(\mathbf{r}) + \tfrac{(1-\alpha)}{\alpha}\nabla\cdot\int_\Omega \mathcal{F}(\mathbf{q}-\mathbf{r})\mathbf{v}_{\text{low}}(\mathbf{q}) d\mathbf{q}
\end{aligned} \tag{10}$$

By combining and discretizing Eq. (7) and Eq. (8) (or Eq. (9) and Eq. (10) if $\alpha$ is not spatially varying), we get a linear system of $(D+1)N$ equations in $(D+1)N$ unknowns ($\mathbf{v}$ and $\lambda$), where $D$ is the dimension and $N$ is the number of grid points.

A few important properties of the derived equations are:

- The guiding velocity field, $\mathbf{v}_{\text{low}}$, does not have to be divergence free.
- The derived guiding equations are self-consistent in the sense that if a fluid velocity field, $\mathbf{v}$, is used to guide itself, meaning that if $\mathbf{v}_{\text{low}} = \mathcal{F}*\mathbf{v}$, then the result will be $\mathbf{v}$ itself. We have verified this experimentally.
- A solution to Eq. (7) and Eq. (8) exists simply because a divergence free velocity field exists. We leave a rigorous proof of the conditions required for uniqueness of the solution as future work. In practice it is our experience that the solution to the linear system converges to the desired precision in few multigrid cycles unless $\alpha \ll 1$.
- By *construction* of the variational problem, the low frequencies of the solution to the linear system will be a

blend of the low frequencies in $\tilde{\mathbf{v}}$ and $\mathbf{v}_{\text{low}}$, and the contribution from each of these in the final velocity field can be controlled using $\alpha$.

### 4.3. The Discretization of the Guiding Equations

We discretize the system of equations, Eq. (7-8) or Eq. (9-10), on a staggered MAC grid using finite difference approximations. That is, velocities are stored on cell faces, and the pressure/Lagrange multipliers are stored in cell centers. The operator matrix for the linear system, Eq. (9) combined with Eq. (10), is depicted in Figure 3. Note that we solve for all unknowns simultaneously. The gradient,$\nabla\lambda$, divergence, $\nabla\cdot\tilde{\mathbf{v}}$, and Laplacian, $\Delta\lambda$, are discretized using the usual second order approximations [FSJ01]. The term $\frac{(1-\alpha)}{\alpha}\int_\Omega \mathcal{F}(\mathbf{q}-\mathbf{r})\,[\mathcal{F}*\mathbf{v}]\,(\mathbf{q})d\mathbf{q}$ is discretized by a sum of point-wise multiplications at cell faces ranging over the support of the lowpass filter. Let $F$ be the discretized version of the lowpass filter, $\mathcal{F}$, then the discretization is $\frac{(1-\alpha)}{\alpha}\sum_{\mathbf{q}\in\text{supp}F} F(\mathbf{q}-\mathbf{r})\,[F*\mathbf{v}]\,(\mathbf{q})$. Likewise the convolution and the right-hand-side correlation are implemented as a sum. The guiding terms involving the divergence are discretized by the usual second order approximation to the divergence. The upper part of the matrix, the first $DN$ equations, are discretized on staggered cell faces, whereas the last $N$ equations are discretized in cell centers. The resulting equation system is linear and sparse. However, the system of equations is asymmetric (see Figure 3) and due to the support of the lowpass filters, the matrix requires an impractical amount of memory when stored uncompressed. However, by employing the multigrid method [BHM00], the system can be solved to sufficient precision in few multigrid cycles, and the matrix stored in a compact format.

## 5. Boundaries

A common complication in fluid simulations is boundary conditions. The situation is further complicated when dealing with simulations at multiple resolutions, since sampling issues might cause a voxelization of the boundary to differ between the simulations.We represent boundaries as the zero level set of a signed distance function sampled at the same resolutions as the simulations. This means that boundaries are limited to having features representable within the frequency space of the simulation resolutions to avoid sampling artifacts. Particularly, boundary features that are too thin to be represented properly in the low-resolution simulation might appear in the high-resolution simulation and potentially result in visual artifacts. We have left this issue as a direction for future work.

If the lowpass filter, $\mathcal{F}$, overlaps the boundaries, one might worry that visual artifacts could occur. We have explored relaxing the guiding weights near boundaries (in order to make the guiding more loose or disable it entirely). We have, however, not found this necessary, as we did not observe a qualitative improvement of the result. In fact, we observed that turning guiding off inside narrow pipes of fluid, caused the high-resolution fluid to move much faster than the guiding input due to less numerical dissipation, which is not desirable. We have also seen no major impact on convergence of the multigrid solver. However, if a large fraction of the domain is part of the boundary, disabling guiding in these parts of the domain speeds up the multigrid solver considerably.

### 5.1. The Penalization Method

Internal boundaries are traditionally hard to address in the multigrid paradigm. This is primarily due to the fact that the domain embedded by the internal boundaries is excluded by the multigrid solve, which in turn complicates the otherwise simple rectangular fluid domain. However, it is possible to treat internal boundaries implicitly, and basically solve for pressure and velocity everywhere in the computational domain, *i.e.* the multigrid solver itself is not explicitly aware of boundaries. One method that does exactly this is the *penalization method* of Angot *et al.* [ABF99, KCR08]. This method also has the advantage of being non-iterative, which makes it a good candidate for our application. An alternative iterative method for handling boundaries in multigrid context was recently proposed in [MCPN08]. The penalization method solves for pressure and velocity everywhere. However, inside boundaries, the velocity, $\mathbf{v}$, is penalized towards the prescribed velocity of the boundary, $\bar{\mathbf{v}}$, and at the surface of the boundary, the velocity satisfies the no-slip condition (the Euler equations, being inviscid, are subject to free-slip, so we rely here on numerical viscosity to make the problem well-posed). Mathematically the penalization is achieved by adding a penalization term, $\frac{1}{\eta}\chi_{\text{boundary}}(\mathbf{v}-\bar{\mathbf{v}})$, to the left-hand side of the momentum equation, where $\chi_{\text{boundary}}$ is the characteristic function of the boundary, and $\eta \ll 1$:

$$\frac{\partial\mathbf{v}}{\partial t} + (\mathbf{v}\cdot\nabla)\,\mathbf{v} + \nabla p + \frac{1}{\eta}\chi_{\text{boundary}}(\mathbf{v}-\bar{\mathbf{v}}) = \mathbf{f} \qquad (11)$$

In practice we use $\eta = 10^{-20}$, consult [ABF99] for a study of the effect of $\eta$. As discretization is not addressed in the original references [ABF99, KCR08], we derive here how to discretize the penalization term to allow for arbitrarily large time steps. In particular an implicit discretization is required, as an explicit discretization restricts the time step to be in the order of $\Delta t \leq 2\eta$, which is highly unpractical. This can be seen by performing a Von Neumann analysis of the simplified equation $\frac{\partial\mathbf{v}}{\partial t} + \frac{1}{\eta}\mathbf{v} = 0$. We describe first how to discretize the penalized momentum equation and how it affects the Poisson equation. Finally we consider how to discretize the penalized guiding equations.

Outside boundaries the discretization is equal to the non-penalized discretization since the characteristic function $\chi_{\text{boundary}} = 0$. Inside the boundary we discretize the velocity

$$\left(\begin{array}{c|c} \vdots & \vdots \\ \mathbf{v}(\mathbf{r}) + \frac{(1-\alpha)}{\alpha}\int_\Omega \mathcal{F}(\mathbf{q}-\mathbf{r})\left[\mathcal{F}*\mathbf{v}\right](\mathbf{q})d\mathbf{q} & \frac{1}{\alpha}\nabla\lambda(\mathbf{r}) \\ \vdots & \vdots \\ \hline \vdots & \vdots \\ \frac{(1-\alpha)}{\alpha}\nabla\cdot\int_\Omega \mathcal{F}(\mathbf{q}-\mathbf{r})\left[\mathcal{F}*\mathbf{v}\right](\mathbf{q})d\mathbf{q} & \frac{1}{\alpha}\Delta\lambda(\mathbf{r}) \\ \vdots & \vdots \end{array}\right)\left(\begin{array}{c} \vdots \\ \mathbf{v}(\mathbf{r}) \\ \vdots \\ \hline \vdots \\ \lambda(\mathbf{r}) \\ \vdots \end{array}\right) = \left(\begin{array}{c} \vdots \\ \tilde{\mathbf{v}}(\mathbf{r}) + \frac{(1-\alpha)}{\alpha}\int_\Omega \mathcal{F}(\mathbf{q}-\mathbf{r})\mathbf{v}_{\text{low}}(\mathbf{q})d\mathbf{q} \\ \vdots \\ \hline \vdots \\ \nabla\cdot\tilde{\mathbf{v}}(\mathbf{r}) + \frac{(1-\alpha)}{\alpha}\nabla\cdot\int_\Omega \mathcal{F}(\mathbf{q}-\mathbf{r})\mathbf{v}_{\text{low}}(\mathbf{q})d\mathbf{q} \\ \vdots \end{array}\right)$$

**Figure 3:** *Shows the linear system $Ax = b$ of $(D+1)N$ equations in $(D+1)N$ unknowns that results from combining Eq. (9) and Eq. (10) where D is the number of dimensions and N is the number of grid cells. Note that the matrix A acts as an operator on x. However to emphasize the relationship with Eq. (9) and Eq. (10) we have included the complete expressions in the matrix (this is why* e.g. $\mathbf{v}(\mathbf{r})$ *appears both as an unknown and in the matrix).*

$\mathbf{v}$ of the penalization term at time $n+1$, thus obtaining the discretization of the penalized momentum equation Eq. (11):

$$\begin{aligned} \frac{\mathbf{v}^{n+1}-\mathbf{v}^n}{\Delta t} - \mathbf{f}^n &= -(\mathbf{v}^n\cdot\nabla)\mathbf{v}^n - \nabla p^n - \frac{1}{\eta}(\mathbf{v}^{n+1}-\tilde{\mathbf{v}}^{n+1}) \\ &\Downarrow \\ \mathbf{v}^{n+1} &= \left(\tilde{\mathbf{v}}^n + \Delta t\left(-\nabla p^n + \frac{1}{\eta}\tilde{\mathbf{v}}^{n+1}\right)\right)\frac{1}{1+\frac{\Delta t}{\eta}} \end{aligned} \tag{12}$$

where $\tilde{\mathbf{v}}^n = \mathbf{v}^n + (\mathbf{f}^n - (\mathbf{v}^n\cdot\nabla)\mathbf{v}^n)\Delta t$. The usual Poisson equation arises by taking the divergence of the momentum equation subject to the constraint $\nabla\cdot\mathbf{v}^{n+1} = 0$. However, taking the divergence of Eq. (12), subject to the same constraint, leads to a slightly different system of equations. Combining with the normal Poisson equation outside boundaries we obtain the following system of equations that should be used whenever boundaries are present:

$$\nabla\cdot\hat{\mathbf{v}}^n = \nabla\cdot\left(\chi_{\text{boundary}}\psi\nabla p + (1-\chi_{\text{boundary}})\nabla p\right) \tag{13}$$

and the velocity update

$$\mathbf{v}^{n+1} = \hat{\mathbf{v}}^n - \chi_{\text{boundary}}\psi\nabla p - (1-\chi_{\text{boundary}})\nabla p \tag{14}$$

respectively, where $\psi = \frac{1}{1+\frac{\Delta t}{\eta}}$ and $\hat{\mathbf{v}}^n = (1-\chi_{\text{boundary}})\tilde{\mathbf{v}}^n + \chi_{\text{boundary}}\left(\tilde{\mathbf{v}}^n + \frac{\Delta t}{\eta}\tilde{\mathbf{v}}^{n+1}\right)\psi$.Note that we have left out the factor of $\Delta t$ in front of $p$ in both equations. This is possible since $\Delta t$ is a constant and we are only interested in $p$ up to a scale.

With these derivations in mind it is easy to devise a penalization strategy for the guiding equations. By setting $\mathcal{F} \equiv 0$ when its center point, $\mathbf{r}$, is inside a boundary, it is straightforward to verify that the following equations are equivalent to the guiding equations Eq. (7) and Eq. (8) outside the boundary, and equal to Eq. (13) and Eq. (14) inside the boundary. Due to the latter equality, the velocity is driven towards the actual boundary velocity inside the boundary. Consider $G_1$ to be the discretization of $\mathcal{G}_1$ and $G_2$ to be the discretization of $\mathcal{G}_2$, where $\mathcal{G}_1 = \frac{1}{\alpha(\mathbf{r})}\int_\Omega(1-\alpha(\mathbf{q}))\mathcal{F}(\mathbf{q}-\mathbf{r})\left[\mathcal{F}*\mathbf{v}\right](\mathbf{q})d\mathbf{q}$ and $\mathcal{G}_2 = \frac{1}{\alpha(\mathbf{r})}\int_\Omega(1-\alpha(\mathbf{q}))\mathcal{F}(\mathbf{q}-\mathbf{r})\mathbf{v}_{\text{low}}(\mathbf{q})$ are the guiding terms on the left- and right-hand side of Eq. (7). The penalized guiding equations become:

$$\nabla\cdot\hat{\mathbf{v}}^n = \nabla\cdot\left(\chi_{\text{boundary}}\psi\nabla p + (1-\chi_{\text{boundary}})\frac{1}{\alpha(\mathbf{r})}\nabla p + G_1 - G_2\right)$$

and

$$\mathbf{v}^{n+1} = \hat{\mathbf{v}}^n - \chi_{\text{boundary}}\psi\nabla p - (1-\chi_{\text{boundary}})\frac{1}{\alpha(\mathbf{r})}\nabla p - G_1 + G_2$$

These equations replace Eq. (7) and Eq. (8) whenever boundaries are present.

## 6. Filter Estimation, Upsampling and Downsampling

For some types of simulations we vary the filter, $\mathcal{F}$, throughout the domain to achieve a certain artistic goal (see section 8). For physically based guiding velocity fields, it is in our experience very hard to apply a common lowpass filter, such as a Gaussian, by heuristically determining a suitable standard deviation. In particular it often results in guided simulations that are too smooth. Instead we construct the lowpass filter based on the following intuition: To use a high resolution velocity field as input to our guiding algorithm, the velocity field would have to be smoothed and downsampled. The guiding algorithm would then subsequently reconstruct the velocity field in high resolution and use this as $\mathbf{v}_{\text{low}}$ in the minimization. Assuming that the combination of these operations can be expressed as a convolution, the estimation of the lowpass filter, $\mathcal{F}$, encompasses the effects of smoothing (with a Gaussian kernel), $G$, downsampling, $\mathcal{D}$, and upsampling, $\mathcal{U}$. Essentially the desired filter is just $\mathcal{U}\circ\mathcal{D}\circ G$. However simply constructing the discrete filter in this way is very sensitive to how $G$ is centered relative to $\mathcal{D}$. Another issue is what exactly the standard deviation of $G$ should be set to. For this reason we estimate an approximation to $\mathcal{U}\circ\mathcal{D}\circ G$ offline via an optimization process. The estimated lowpass filter is not simulation dependent but depends only on the upsampling factor, the width, $w$, of the filter as well as the downsampling and upsampling methods.

The input to the filter estimation consists of a filter width,

$w$, a 2D image, $A$, of $N$ uniformly distributed random numbers (to avoid bias), and an upsampling factor. An optimization is then performed to find the standard deviation, $\sigma$, of $G$ and the lowpass filter, $\hat{F}$, that minimizes the difference between $\mathcal{U}(\mathcal{D}(G(\sigma) * A))$ and $\hat{F} * A$, where $*$ denotes a convolution. In each iteration of the optimization, we construct a $N \times (w^2)$ system of linear equations, $M\mathbf{x} = \mathbf{b}$, that is solved in a least squares sense. The vector, $\mathbf{x}$, represents the filter, $\hat{F}$, we are estimating, while the right-hand side, $\mathbf{b}$, contains $\mathcal{U}(\mathcal{D}(G(\sigma) * A))$. The matrix, $M$, contains a row for each grid point in $A$, and each row contains the values of $A$ that fall in the $w \times w$ support of the filter we are estimating.To facilitate speedups during the multigrid solve of the guiding equations, we make the final guiding filter, $F$, separable. We have observed that $\hat{F}$ is usually very close to separable, so we construct the $w$ diagonal values of $F$ by setting them equal to the diagonal values of $\hat{F}$. The diagonal values completely define a separable $F$, as the values of $F$ are given by a tensor product of the square root of the diagonal values with themselves. This tensor product definition directly extends to three dimensions, and we construct our filters for 3D guided simulations by a triple tensor product.

We both up- and down-sample by means of a cubic convolution [Key81]. Practical experience has shown that using linear interpolation is not sufficient. Boundaries are handled by explicitly setting the velocity of the low resolution boundaries in the low resolution velocity field before upsampling. In this way the upsampling process itself is unaware of boundaries.

## 7. Multigrid Solver

We adopt the multigrid method (see [BHM00] for an introduction) to solve the linear system in Figure 3. Some of the advantages of the multigrid method are that for a grid with $N$ grid points, it requires linear $O(N)$ storage, handles both symmetric and asymmetric linear systems and the low frequencies of the solution itself act as a preconditioner.
Briefly explained, the multigrid method solves a linear system $A\mathbf{x} = \mathbf{b}$ on progressively coarser grids. It transfers solutions from coarser to finer grids and residuals from finer to coarser grids by means of interpolation and restriction operators, respectively. On each grid, or level, of the multigrid solve, a matrix operator is constructed and a relaxation method is employed for a small number of iterations. The strength of the multigrid method comes from the fact that the low frequencies of the solution converge much faster on coarser than finer grids.

For our linear system we found that extending the solver with restriction and interpolation operators that operate on cell faces as well as cell centers (in order to maintain a staggered grid on each level of the multigrid solve) improved the ratio of errors in each iteration roughly by a factor of two and required only a few multigrid cycles to converge.

The matrix operators on each multigrid level are stored in a compact storage format motivated by the fact that the stencils of the discretized equations (see Figure 3) are identical in many grid points. For example, in the case where $\mathcal{F}$ and $\alpha$ do not vary spatially, all grid points, that contain fluid and for which the support of the stencil is completely inside the spatial domain, will have identical stencils at the top multigrid level. By means of interpolation and restriction operators this property transfers to lower levels as well. Our matrix storage format is comprised of an indexed representation in which each grid point stores an index into an array of unique stencils, such that no duplicate stencils are stored. When the indexed representation is constructed, a stencil in a given grid point is only computed whenever it is detected that the stencil in the current grid point may not be identical to any of the stencils previously computed, thus speeding up the computation. Whenever $\mathcal{F}$ and $\alpha$ do not vary spatially it is possible to further reduce storage requirements, because $\mathcal{F}$ is separable, meaning that $\int_\Omega \mathcal{F}(\mathbf{q}-\mathbf{r})\,[\mathcal{F} * \mathbf{v}]\,(\mathbf{q}) d\mathbf{q}$ is also separable. Hence, only a $1D$ representation of the stencil needs to be stored. Currently we only exploit this on the top level.We do not even store the matrix operator at the topmost multigrid level when separability can be exploited because we found that both matrix multiply and relaxation can be computed efficiently, in parallel, on the fly. For the $256^3$ guided smoke spray example (see section 8 and Figure 1) using a lowpass filter of width 9, the combined effect of our indexed representation and separability reduced total storage requirements of the matrix operators to 208MB from the 4.04TB required if storing all non-zero entries.In the case of moving boundaries, the stencils at the grid points that change status from boundary to fluid or vice versa are modified by the penalization method. Although in the worst case the entire matrix has to be re-computed when this happens, in practice only a few grid points, compared to the total number of grid points in the domain, change status. Hence the matrix can be re-computed locally by only updating the stencil values that are actually affected by the change. The locality of the change is maintained throughout lower multigrid levels, hence facilitating a fast update to matrix operators at all levels.

We have taken the approach of parallelizing the individual components of our multigrid solver such as the relaxation and matrix-vector multiplication using Intel Thread Building Blocks [Rei07]. This gives a speedup of between four and five on an eight-core machine. Whenever possible, the relaxation and matrix-vector multiplication operate in parallel using separable filters only. We use Successive Over-Relaxation (SOR) with an over-relaxation parameter of 1.35 (found experimentally). Direct parallelization of SOR cannot guarantee a data flow consistent with a serial SOR algorithm, and hence the convergence analysis of the serial algorithm does not apply. However, we found that parallelizing SOR gives convergence rates equivalent to those obtained in serial. Particularly these convergence rates are significantly better than those obtained using weighted Jacobi relaxation.

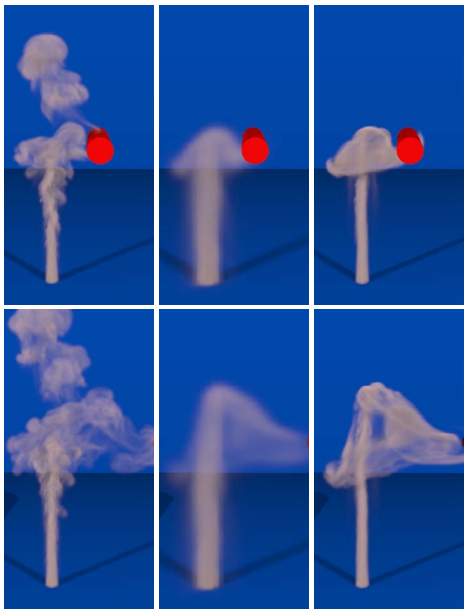

**Figure 4:** ***Upper triple:*** *High resolution unguided, low resolution unguided and high resolution guided simulations at frame 210 of a smoke column interacting with a moving boundary. In low resolution and guided high resolution the boundary touches the top of the smoke column whereas in high resolution the boundary passes through the smoke. Due to diffusion the column appears thicker in low resolution.* ***Lower triple:*** *Frame 256 from same simulation.*

## 8. Results and Discussion

We illustrate our variational guiding approach with the following examples. All simulations were run on a MAC Pro with 4GB of memory and two Intel Xeon quad-core 2.80GHz processors. We employ BFECC [KLLR07] and monotonic cubic interpolation [FSJ01] for advection. Identical time steps were used in low and high resolution.

**Rising Smoke Column**

A spherical density- and hot temperature-source is placed at the bottom of an elongated domain in which the smoke velocities are subjected to buoyancy. Both the density and temperature fields are advected in the flow. The low-resolution domain is $16 \times 64 \times 16$, and the high-resolution domain is upsampled by a factor of four to $64 \times 256 \times 64$. Figure 5 illustrates that the low-resolution simulation rises slower and with less turbulence than the high-resolution simulation. Due to diffusion of densities the smoke column in low resolution appears to be thicker than in high resolution. The rightmost image shows that just using the simple approach of upsampling the low-resolution velocity field does not result in high frequency details. Guided simulations with varying $\alpha$ are shown in Figure 7 and enable a smooth interpolation between a strictly guided and an unguided simulation. In practice we run guided simulations with $\alpha = 0.65$ which appears to obtain a plausible mix of high frequency detail and guided low frequency flow. Increasing the width of the guiding filter allows for higher frequency detail to develop and we use a $9^3$ filter stencil. The guiding matrix operators required 72MB and took 119 seconds to compute. 4 multigrid cycles were required to converge to a divergence of roughly $10^{-5}$ for the guided (31 seconds per frame) and unguided (16 seconds per frame) simulations at resolution $64 \times 256 \times 64$. Figure 6 illustrates that for Eulerian smoke simulations, undesirable features may develop over time when explicitly blending the low frequencies of the current velocity field with the upsampled guiding velocity field as proposed in [TKPR06]. Granted, their method is based on guiding particles whereas we set the low frequencies everywhere in the domain. Additionally, the choice of blend-factor used in [TKPR06] may, in our experience, give plausible results for one simulation, but undesired features for another. This implies that manual experimentation in high resolution may be required. However, if a successful blend-factor can be found, this simple method provides a faster alternative to our method.

**Rising Smoke Column with Moving Boundary**

This setup is similar to the Rising Smoke Column simulation except that a moving boundary is timed to touch the top of the smoke column in low resolution as shown in Figure 4. In high resolution the boundary passes through the faster moving smoke. The guided high-resolution simulation retains the qualitative features of the low-resolution simulation and adds higher frequency detail. The initial guiding matrix operators required 280MB and took 254 seconds to compute. The guided and unguided high resolution simulations took respectively 84 and 45 seconds per frame.

**Smoke Spray**

A velocity-, density- and temperature-source injects hot high-velocity smoke subject to a buoyancy force as shown in Figure 1. Again, the guided simulation retains the low frequency qualitative features of its guiding simulation. The total storage required for the matrix operators of the guided simulation is 208MB and the computation took 696 seconds. The guided simulation took 580 seconds per frame and the unguided simulation 315 seconds at resolution $256^3$.

**Guiding Curves**

A low-resolution, non-physically based velocity field in the shape of a torus is used to guide a smoke torus (see Figure 8). Both the lowpass filter $\mathcal{F}$ and the guiding parameter $\alpha$ vary spatially; the identity filter is used in a narrow band inside the initial smoke band to prevent drift of the low-frequency velocities, and guiding is gradually diminished outside the initial smoke band by increasing $\alpha$. Vorticity confinement [FSJ01] is used to induce high-frequency instabilities. The rightmost images of Figure 8 show the same setup with a trefoil knot curve. For both simulations it was necessary to uniformly quantize the values of $\alpha$ to limit the number of unique stencils in the matrix operators and thereby the storage requirements. We have verified experimentally that the quantization did not cause visual artifacts. For the torus simulation, the matrix operators required 1496 MB and took 1061 seconds to compute. The simulation ran for 150

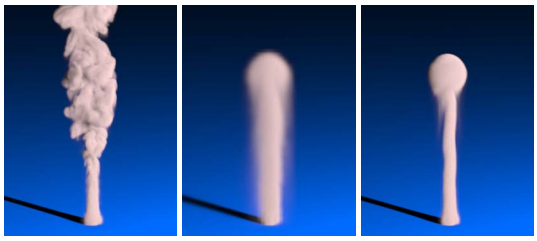

**Figure 5:** ***Left:*** *High resolution simulation.* ***Middle:*** *Low resolution simulation.* ***Right:*** *Low resolution velocity field upsampled to high resolution followed by pressure projection and density advection (note that this is not a guided simulation). High frequencies cannot develop during the standard pressure projection [Bri08].*

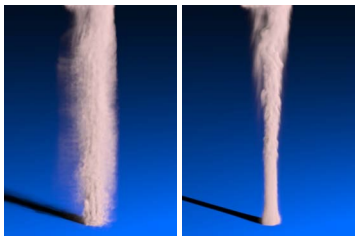

**Figure 6:** *Frame 399 from the smoke column simulation.* ***Left:*** *Explicitly blending with a low frequency guiding velocity field before pressure projection may result in undesirable features developing over time.* ***Right:*** *Guided simulation with identical blending parameter, $\alpha$, for comparison.*

seconds per frame at resolution $128^3$. The trefoil knot simulation needed 2064 MB, and the computation of the matrix operators took 2549 seconds. The simulation took 155 seconds per frame at resolution $128^3$.

The framework proposed in this paper has several limitations:

- It is an assumption of our method that an artist can obtain a satisfactory bulk movement in low resolution. If this is not the case, our method does not apply.
- Provided that an appropriate low-resolution simulation can be found, our framework allows animators to prototype in low resolution and then add details with a guided high-resolution simulation. However, for the examples presented in this paper we found the guided simulations to incur an overhead of $84-94\%$ compared to the high-resolution unguided simulation. For this reason we wish to pursue further strategies for optimization in the future.
- The amount of density-diffusion and -dissipation is significantly higher in low resolution. For this reason, features in the high-resolution guided simulation may deviate in intensity from the low-resolution simulation. Specifically, this is the case when sheets of diffused densities develop and when thin structures dissipate away in low resolution.
- Our approach fails if boundaries have very thin or fine features that cannot be represented simultaneously on the high- and low-resolution grids. At least this will be the case close to the boundary where boundary conditions must be fulfilled. We plan to address this in future work, possibly by adapting the methods for handling thin shells by Guendelman *et al.* [GSLF05] and the accurate, variational boundary coupling of Batty *et al.* [BBB07].

## 9. Conclusion

Smoke simulations are notoriously hard to art-direct due in part to numerical viscosity and the non-linearity of the governing equations. In this paper we proposed a self-consistent variational method for guiding a high resolution Eulerian smoke simulation by a lower resolution flow. The low frequencies of the guided simulation agree as much as possible with the low resolution flow, and higher frequencies can develop. It is a fundamental assumption of our method that a satisfactory low-resolution simulation exists, otherwise our method does not apply. The low-resolution flow can originate from a physically based simulation, be purely art directed or arise from a combination of the two. Our method allows for a smooth interpolation between a strictly guided and an unguided physical simulation, and appears to be free of the visual artifacts that may result from using previous methods. An interesting alternative to the multigrid approach taken in this paper might be the multilevel bases described by Oswald [Osw01]. Finally, we believe that the variational approach taken in this paper is applicable to a wide range of fluid control paradigms in computer graphics.

## Acknowledgements

This work was partially funded by the Danish Agency for Science, Technology and Innovation. We thank Ole Østerby for advice on numerics, Michael Clive and Ryo Sakaguchi for helping define the problem, our colleagues at Digital Domain for their support, and the reviewers, committee and program chairs for constructive feedback.

## References

[ABF99] ANGOT P., BRUNEAU C.-H., FABRIE P.: A penalization method to take into account obstacles in incompressible viscous flows. *Numerische Mathematik 81*, 4 (February 1999), 497–520.

[BBB07] BATTY C., BERTAILS F., BRIDSON R.: A fast variational framework for accurate solid-fluid coupling. *ACM Trans. Graph. 26*, 3 (2007), 100.

[BHM00] BRIGGS W. L., HENSON V. E., MCCORMICK S. F.: *A multigrid tutorial (2nd ed.)*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.

[BMWG07] BERGOU M., MATHUR S., WARDETZKY M., GRINSPUN E.: Tracks: toward directable thin shells. In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers* (New York, NY, USA, 2007), ACM, p. 50.

[Bri08] BRIDSON R.: *Fluid Simulation for Computer Graphics*. AK Peters, 2008.

[FGP07] FROEMLING E., GOKTEKIN T., PEACHEY D.: Simulating whitewater rapids in ratatouille. In *SIGGRAPH '07: ACM SIGGRAPH 2007 sketches* (New York, NY, USA, 2007), ACM, p. 68.

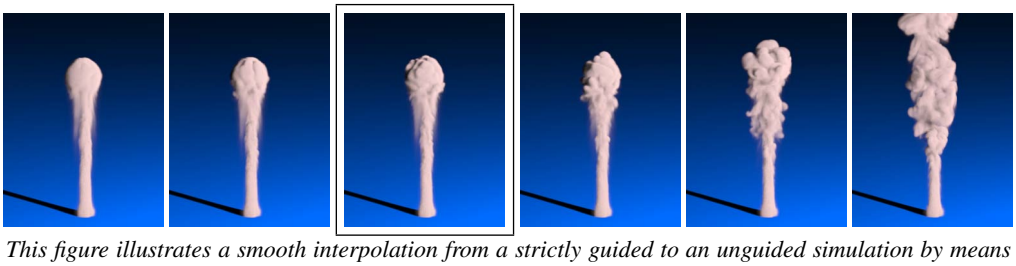

**Figure 7:** *This figure illustrates a smooth interpolation from a strictly guided to an unguided simulation by means of varying the* $\alpha$ *parameter. Left to right:* $\alpha = 0.15, 0.45, \mathbf{0.65}, 0.85, 0.97, 1.0$. *We use* $\alpha = 0.65$ *for guided simulations.*

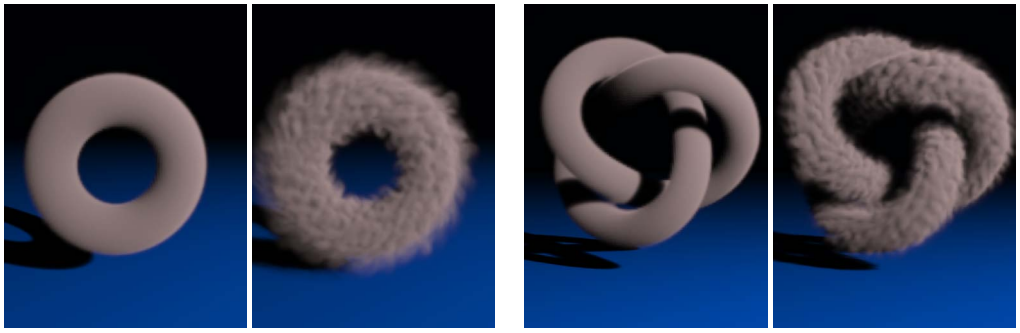

**Figure 8:** ***Left:*** *A low-resolution, non-physically based circular velocity field is used to evolve a smoke torus. Both* $\mathcal{F}$ *and* $\alpha$ *vary spatially to facilitate the desired motion while preserving the shape of the torus.* ***Right:*** *A similar setup with a trefoil knot.*


[FM97] FOSTER N., METAXAS D.: Controlling fluid animation. In *CGI '97: Proceedings of the 1997 Conference on Computer Graphics International* (Washington, DC, USA, 1997), IEEE Computer Society, p. 178.

[FP02] FERZIGER J. H., PERIC M.: *Computational Methods for Fluid Dynamics*. Springer, Berlin, 2002.

[FSJ01] FEDKIW R., STAM J., JENSEN H. W.: Visual simulation of smoke. In *Proceedings of ACM SIGGRAPH 2001* (Aug. 2001), Computer Graphics Proceedings, Annual Conference Series, pp. 15–22.

[GSLF05] GUENDELMAN E., SELLE A., LOSASSO F., FEDKIW R.: Coupling water and smoke to thin deformable and rigid shells. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers* (New York, NY, USA, 2005), ACM, pp. 973–981.

[KCR08] KOUMOUTSAKOS P., COTTET G.-H., ROSSINELLI D.: Flow simulations using particles: bridging computer graphics and cfd. In *SIGGRAPH '08: ACM SIGGRAPH 2008 classes* (New York, NY, USA, 2008), ACM, pp. 1–73.

[Key81] KEYS R.: Cubic convolution interpolation for digital image processing. *Acoustics, Speech, and Signal Processing [see also IEEE Transactions on Signal Processing], IEEE Transactions on 29*, 6 (1981), 1153–1160.

[KLLR07] KIM B., LIU Y., LLAMAS I., ROSSIGNAC J.: Advections with significantly reduced dissipation and diffusion. *IEEE Transactions on Visualization and Computer Graphics 13*, 1 (2007), 135–144.

[KTJG08] KIM T., THÜREY N., JAMES D., GROSS M.: Wavelet turbulence for fluid simulation. *ACM Trans. Graph. 27*, 3 (2008), 1–6.

[MCP*09] MULLEN P., CRANE K., PAVLOV D., TONG Y., DESBRUN M.: Energy-preserving integrators for fluid animation. *ACM Trans. Graph. to appear* (2009).

[MCPN08] MOLEMAKER J., COHEN J. M., PATEL S., NOH J.: Low viscosity flow simulations for animation. In *ACM SIGGRAPH Symposium on Computer Animation* (2008).

[MTPS04] MCNAMARA A., TREUILLE A., POPOVIĆ Z., STAM J.: Fluid control using the adjoint method. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers* (New York, NY, USA, 2004), ACM, pp. 449–456.

[NSCL08] NARAIN R., SEWALL J., CARLSON M., LIN M. C.: Fast animation of turbulence using energy transport and procedural synthesis. In *SIGGRAPH Asia '08: ACM SIGGRAPH Asia 2008 papers* (New York, NY, USA, 2008), ACM, pp. 1–8.

[Osw01] OSWALD P.: Remarks on multilevel bases for divergence-free finite elements. *Numerical Algorithms* (2001).

[Rei07] REINDERS J.: *Intel Threading Building Blocks*, first edition ed. O'Reilly, 2007.

[SB08] SCHECHTER H., BRIDSON R.: Evolving sub-grid turbulence for smoke animation. In *Proceedings of the 2008 ACM/Eurographics Symposium on Computer Animation* (2008).

[Sta99] STAM J.: Stable fluids. In *Proceedings of SIGGRAPH 99* (Aug. 1999), Computer Graphics Proceedings, Annual Conference Series, pp. 121–128.

[SY05] SHI L., YU Y.: Taming liquids for rapidly changing targets. In *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation* (New York, NY, USA, 2005), ACM, pp. 229–236.

[TKPR06] THÜREY N., KEISER R., PAULY M., RÜDE U.: Detail-preserving fluid control. In *SCA '06: Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, Switzerland, 2006), Eurographics Association, pp. 7–12.

[TMPS03] TREUILLE A., MCNAMARA A., POPOVIĆ Z., STAM J.: Keyframe control of smoke simulations. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers* (New York, NY, USA, 2003), ACM, pp. 716–723.