

# UC Berkeley

## UC Berkeley Previously Published Works

### Title

Fluids in deforming meshes

### Permalink

<https://escholarship.org/uc/item/3md8k3d4>

### Authors

Feldman, Bryan E  
O'Brien, James F  
Klingner, Bryan M  
[et al.](#)

### Publication Date

2005-07-29

### DOI

10.1145/1073368.1073405

### Supplemental Material

<https://escholarship.org/uc/item/3md8k3d4#supplemental>

Peer reviewed

# Fluids in Deforming Meshes

Bryan E. Feldman    James F. O'Brien    Bryan M. Klingner    Tolga G. Goktekin

University of California, Berkeley

---

## Abstract

*This paper describes a simple modification to an Eulerian fluid simulation that permits the underlying mesh to deform independent of the simulated fluid's motion. The modification consists of a straightforward adaptation of the commonly used semi-Lagrangian advection method to account for the mesh's motion. Because the method does not require more interpolation steps than standard semi-Lagrangian integration, it does not suffer from additional smoothing and requires only the added cost of updating the mesh. By specifying appropriate boundary conditions, mesh boundaries can behave like moving obstacles that act on the fluid resulting in a number of interesting effects. The paper includes several examples that have been computed on moving tetrahedral meshes.*

Categories and Subject Descriptors (according to ACM CCS): 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:** Natural phenomena, physically based animation, computational fluid dynamics, moving meshes.

---

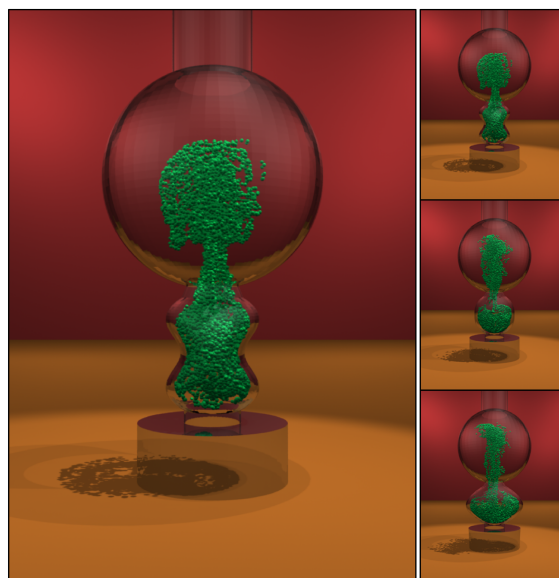
## 1. Introduction

In recent years a number of simulation-based methods for generating realistic fluid animations have been proposed. These techniques use computational nodes that are either fixed in space (Eulerian) or that move with the fluid (Lagrangian). Currently the former appear to be more popular and have found widespread use in the visual effects industry.

In this paper we describe a hybrid Lagrangian-Eulerian approach that simulates fluid in a predominately Eulerian fashion but that also allows prescribed motion and deformation of the underlying mesh independent of the fluid motion. By specifying appropriate boundary conditions, the fluid motion can be left unaffected by that of the mesh, or the fluid can be made to treat mesh boundaries as moving obstacles that effect the fluid's motion.

The method consists of a straightforward modification of the commonly used semi-Lagrangian integration scheme. This modification has a simple, intuitive explanation and can also be easily derived from an ALE (arbitrary Lagrangian-Eulerian) formulation of the advection term in the Navier-Stokes equations.

We have implemented this modification in the framework of the the fixed tetrahedral mesh method described by [FOK05]. The modification only required minor changes and imposes little additional computational cost. We include several examples, such as the one in Figure 1, that demonstrate how this method can be used to achieve a variety of interesting results. While we have only tested this modified



**Figure 1:** Rhythmic contractions of the lower chamber induce motion in the fluid within a glass container with an open top. The fluid motion is illustrated by the behavior of the massless marker particles.

semi-Lagrangian integration scheme with tetrahedral grids, it should be easily applicable to other discretizations such as standard regular grids or octrees.

## 2. Background

The simulation of fluids has proven to be an important tool in the visual effects industry and several researchers have developed methods for realistically modeling both gases and liquids. Some examples that simulate smoke include [FSJ01], [FM97], [Sta99], [FSJ01], and [FOK05]. Simulating liquids with free surfaces additionally requires a surface tracking method. [FF01] introduced the use of level sets for surface tracking, which was improved to the particle level set method in [EMF02]. More recently, [BGOS05] demonstrated a robust technique based on semi-Lagrangian contouring.

Interaction with obstacles has always been an important part of these simulations [GAD03], [Pes02], [YOH00]. [CMT04] recently proposed a method for simulating the two-way interaction of rigid bodies and fluids while [GSLF05] presents a method for coupling fluids with both deformable and rigid shells.

The special case of translating meshes has been addressed by other researchers. Recently, [SCP\*04] used the principle of Galilean Invariance to allow the grid to follow the visually important portion of the fluid. [REN\*04] achieved this same goal by moving their mesh by whole grid increments and translating values in their data array.

Meshless methods that use Lagrangian particles for simulation of fluids have shown significant promise. Examples of these methods include [TPF89], [DC96], [CD97], [SAC\*99], [MCG03], [PTB\*03], and [MKN\*04]. These meshless methods are particularly well suited to deformation as points may move freely without concern for maintaining a good mesh. However, meshless methods have yet to gain the widespread acceptance in the graphics community seen by mesh-based Eulerian methods.

Arbitrary Lagrangian-Eulerian (ALE) methods simulate on meshes that move independent of a fixed coordinate system or the fluid motion. These methods have proven to be very useful in the simulation of highly deformable elastic materials. ALE was first presented by [HAC74] who used finite differences to solve compressible fluid problems. [DFSG77] later applied ALE to the finite element setting. An excellent survey of subsequently developed ALE methods appears in [DHPRF04].

## 3. Methods

The techniques we describe here generalize Eulerian fluid simulation to allow for deforming meshes. The basic idea is that at all times, velocities are computed with respect to a fixed world coordinate system. The moving mesh simply provides a time-varying discretization of the fluid properties. The method we present here essentially amounts to solving the standard ALE advection formulation using the semi-Lagrangian method.

Conceptually, our method is independent of the type of mesh used. Because meshes that use cubical cells (*e.g.*

regular rectilinear and octree) need to maintain their regular structure, deformations of these grids are limited to rigid body transformations and uniform scaling. Tetrahedral meshes, on the other hand, allow for a substantially richer set of deformations. As such, all results shown in this paper are generated using a modification of the tetrahedral mesh method described in [FOK05].

The governing equations for an incompressible, inviscid fluid using an ALE advection term (see *e.g.* [DHPRF04]) take the form:

$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u} - \bar{\mathbf{u}}) \cdot \nabla \mathbf{u} - \frac{\nabla p}{\rho} + \frac{\mathbf{f}}{\rho} \quad (1)$$

subject to the mass conservation constraint for incompressible fluids:

$$\nabla \cdot \mathbf{u} = 0 \quad (2)$$

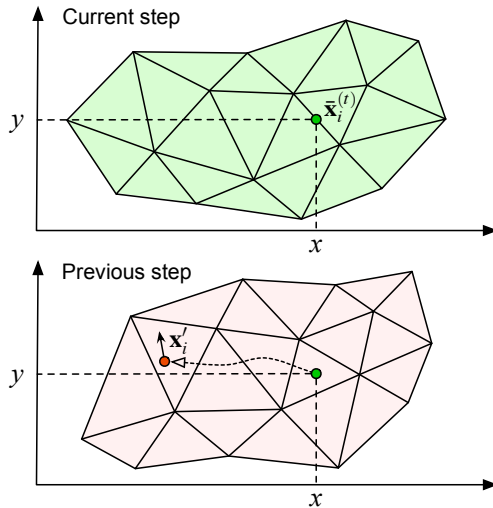
In these equations,  $\mathbf{u}$  is the fluid velocity,  $\bar{\mathbf{u}}$  mesh velocity,  $t$  time,  $p$  pressure,  $\rho$  density, and  $\mathbf{f}$  any external forces. The symbol  $\nabla$  denotes the vector of differential operators  $\nabla = [\partial/\partial x, \partial/\partial y, \partial/\partial z]^T$ . Note that if the mesh velocity,  $\bar{\mathbf{u}}$ , is zero, these equations reduce to the standard inviscid Euler equations. ALE methods typically integrate the advection term of Equation (1) by directly applying some integration scheme to the mesh velocities. In our case, we account for the mesh displacement over a time step directly during semi-Lagrangian advection.

We solve this system using the fractional step method as described by [Sta99] with three distinct steps: *add forces* which computes accelerations due to buoyancy or other forces acting on the fluid, *velocity advection* which uses a semi-Lagrangian method to account for transport of the velocity field, and *pressure correction* which solves for and applies pressure forces that ensure incompressibility. Additionally, for rendering purposes we advect a large number of massless tracker particles.

### 3.1. Generalized semi-Lagrangian step

Due to its simplicity and stability, the semi-Lagrangian method has become the standard tool for solving the advection of the velocity field [Sta99]. Let  $\bar{\mathbf{x}}$  be the locations where velocities are stored (in our case the face centers). For a fixed mesh, the semi-Lagrangian method updates the velocity value at the  $i^{th}$  face center by first tracing backwards from  $\bar{\mathbf{x}}_i$ , to a location  $\mathbf{x}'_i$ . The current velocity stored at  $\bar{\mathbf{x}}_i$ ,  $\mathbf{u}(\bar{\mathbf{x}}_i, t)$ , is then set to be the interpolated value of the previous velocity field at the location  $\mathbf{x}'_i$ ,  $\mathbf{u}(\mathbf{x}'_i, t - \Delta t)$ .

We generalize the above procedure to account for the change in  $\bar{\mathbf{x}}$  as the mesh deforms. Let  $\bar{\mathbf{x}}^{(t)}$  be the locations where velocities are stored at time  $t$ . As before, we first trace back from  $\bar{\mathbf{x}}_i^{(t)}$  to a point  $\mathbf{x}'_i$  using the old velocities. Then, we set the new velocity at  $\bar{\mathbf{x}}_i^{(t)}$  to the value interpolated from the old velocity field at  $\mathbf{x}'_i$ . Since the velocities from the previous step are stored in different locations,  $\bar{\mathbf{x}}^{(t-\Delta t)}$ , we have to trace back and interpolate using this previous mesh (see Figure 2).



**Figure 2:** A 2-D representation of our generalized semi-Lagrangian method. We trace back from the position where a velocity is stored in the new mesh  $\bar{\mathbf{x}}_i = (x, y)$ , interpolate the velocity using the old mesh and update the velocity in the new mesh.

Our process is conceptually similar to performing a standard semi-Lagrangian step using the old mesh and then resampling velocities at positions in the new mesh. This, however, involves an additional resampling step to the one already present in the semi-Lagrangian step. Since resampling smooths the velocity field, we prefer tracing back from the new mesh locations and therefore only resample once per advection step just as in a standard semi-Lagrangian step. Because we only resample once during the semi-Lagrangian step, our method incurs only the smoothing of standard semi-Lagrangian integration.

### 3.2. Boundary Conditions

The procedure for mass conservation is unchanged by the mesh movement. Open boundary conditions are enforced by setting a fixed ambient pressure (usually zero). Where we wish to enforce closed boundary conditions we must ensure that the fluid's normal velocity matches that of the appropriate boundary face in the mesh. The method for imposing these types of boundary conditions during pressure correction is well known. While variations of the approach we use almost certainly have been used by others within the computer graphics community, we include a description for completeness.

To make our fluid conserve mass, the velocity field must obey a volume-scaled divergence free condition,  $V_j(\nabla \cdot \mathbf{u}^{(t)}) = 0$ , where  $V_j$  is the volume of each cell  $j$ . The intermediate velocity,  $\mathbf{u}^*$  from the advection step does not, in general, satisfy this condition. We solve for a pressure field that will correct velocities such that the above condition is met for each cell. Velocities at faces on the boundary of the fluid with obstacles are constrained to have the same normal component as the velocity of the obstacle and

therefore should not be changed by the pressure. To find the appropriate pressure field, we first generate matrices that encode our derivative operators.  $\mathbf{G}$  is a  $(N_o \times N_c)$  matrix that when multiplied by a vector,  $\mathbf{p}$ , of pressures at all of the  $N_c$  cells, produces the pressure gradient at the  $N_o$  open boundary and interior faces.  $\mathbf{D}$  is a  $(N_c \times N_o)$  matrix that produces the open boundary and interior faces' contribution to the volume-scaled divergence for each cell. Given these matrices we solve for  $\mathbf{p}$  in the symmetric, positive-definite linear system:

$$\mathbf{D}\mathbf{G}\mathbf{p} = \frac{\rho}{\Delta t} \mathbf{b} \quad (3)$$

where  $\mathbf{b}$  is a  $(N_c \times 1)$  vector whose entries are the volume-scaled divergence of  $\mathbf{u}^*$  computed using *all* faces of that cell. Once  $\mathbf{p}$  is found it is used to accelerate the open and interior face velocities by  $\mathbf{u}_o^{(t)} = \mathbf{u}_o^* - (\Delta t/\rho)\mathbf{G}\mathbf{p}$ .

### 3.3. Mesh deformation

Use of the semi-Lagrangian method to account for advection allows for a simple and natural way to alter the mesh. As explained in prior sections, we account for the mesh's deformation during the semi-Lagrangian and pressure correction steps. The only additional cost is re-calculating mesh properties such as node positions, face normals, and cell volumes each time the mesh is altered. In our implementation this takes  $< 5\%$  of the total computation time. As the semi-Lagrangian step uses both previous and updated velocity positions, these must be stored while all other mesh properties can simply be updated after the semi-Lagrangian step.

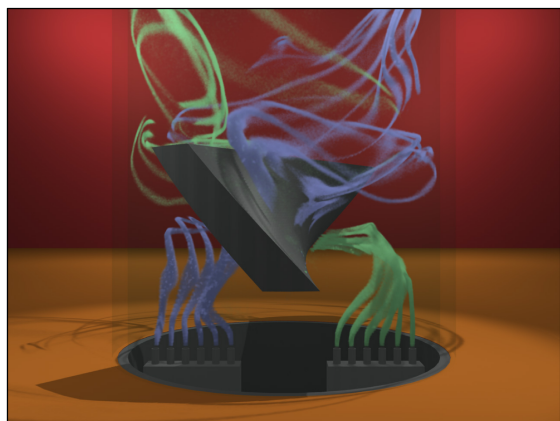
The method is limited by acceptable deformations of the mesh. Cubical cells are limited to uniform scaling and rigid deformations. With the tetrahedral method of [FOK05], and as is typical for Finite Element or Finite Volume methods, some care must be taken to avoid bad elements when deforming a mesh. Elements with poor aspect ratios will yield poor numerical results, while inverted elements will produce garbage. Therefore, the deformation technique should preserve mesh quality. A number of researchers using ALE methods have investigated the problem of "rezoning" — finding favorable interior node positions of a mesh given a prescribed deformation of the boundary nodes (see *e.g.* [DHPRF04],[KH98]). In our work, we have used relatively benign deformations such as squeezing (Figure 1, Figure 5) or rigid motion (Figure 3). For Lagrangian deformation of solids, [ITF04] presented techniques that behave well even in the presence of poorly shaped or inverted elements and such techniques might be extendable to fluid simulation.

## 4. Results and Discussion

We have implemented the method described above and used it to create the examples shown throughout the paper. We generated our tetrahedral meshes with NETGEN<sup>†</sup> and all images were rendered using PIXIE.<sup>‡</sup>

<sup>†</sup> <http://www.hp fem.jku.at/netgen/>

<sup>‡</sup> <http://pixie.sourceforge.net>



**Figure 3:** This image shows smoke spinning and rising in a tube due to the motion of a rotating blade.

The images in Figure 1 show the motion of a fluid created by squeezing and expanding the lower chamber of the glass object. As indicated by the motion of the green massless particles, the fluid first shoots out of the chamber and then is sucked back in. The tube at the top allows for inflow and outflow created by this action.

Figure 3 shows the results of simulating a rotating blade inside a tube with an open top and bottom. The smoke rises and spins due to the blade motion, not due to added buoyancy effects. This simulation was accomplished by rotating a mesh with an embedded blade about its axis while adding smoke at fixed locations in world space.

In Figure 4 we compare results of simulating on a fixed versus deforming mesh using the example of a buoyant jet. A slice of the actual simulation mesh is shown to demonstrate that the top mesh is fixed while the bottom mesh undergoes squashing and stretching. As is shown, moving the mesh creates no visible artifacts in the fluids motion.

In Figure 5 the Stanford Buddha's belly is contracted and expanded to induce a breathing motion. The motion causes smoke to be puffed out through the figure's mouth and into the surrounding volume.

We have presented a simple modification to Eulerian fluid simulation that allows the underlying mesh to deform. The implementation is a straightforward extension of the semi-Lagrangian method that causes no undesirable artifacts, such as smoothing, and requires only the additional cost of updating the mesh.

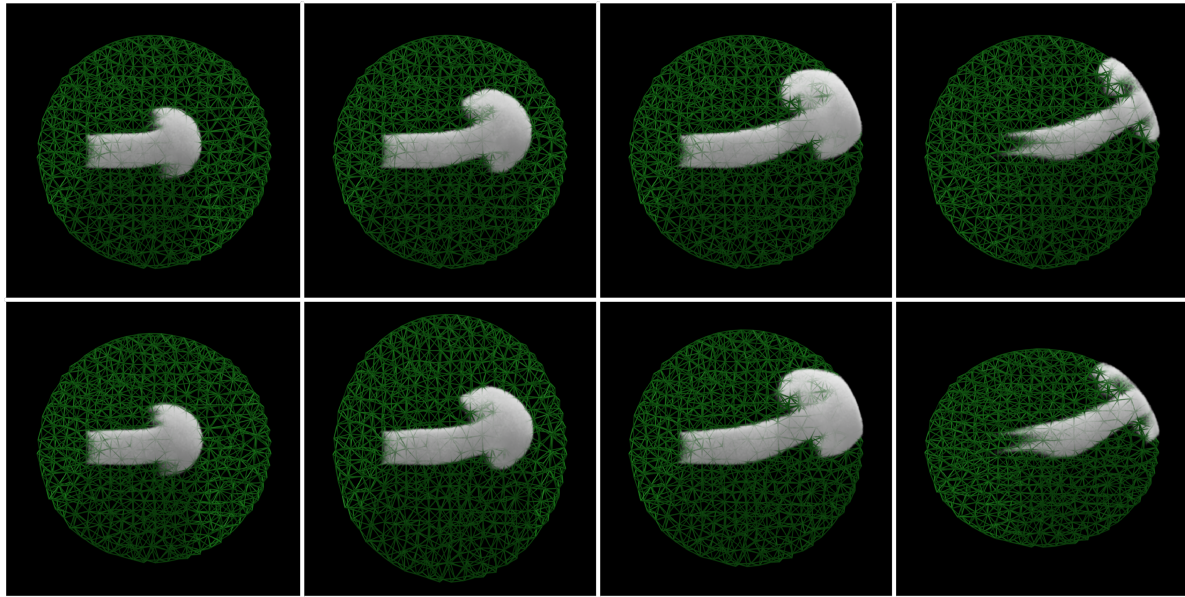
#### Acknowledgments

We thank the other members of the Berkeley Graphics Group for their helpful criticism and comments. This work was supported in part by California MICRO 03-067 and 04-066, and by generous support from Apple Computer, Alias, Pixar Animation Studios, Intel Corporation, Sony Computer Entertainment America, the Hellman Family Fund, and the Alfred P. Sloan Foundation.

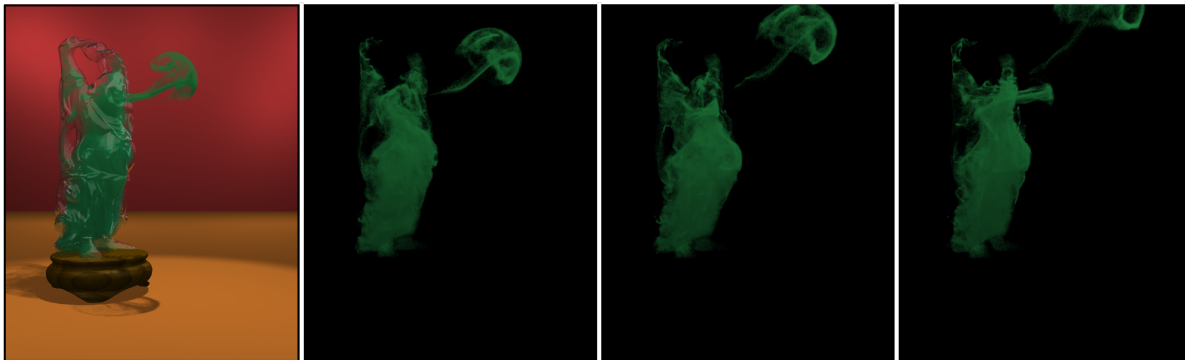
#### References

[BGOS05] BARGTEIL A. W., GOKTEKIN T. G., O'BRIEN J. F., STRAIN J. A.: A semi-lagrangian contouring method for fluid simulation. *ACM Transactions on Graphics* (2005). In press.

- [CD97] CANI M.-P., DESBRUN M.: Animation of deformable models using implicit surfaces. *IEEE Transactions on Visualization and Computer Graphics* 3, 1 (Jan. 1997), 39–50.
- [CMT04] CARLSON M., MUCHA P. J., TURK G.: Rigid fluid: Animating the interplay between rigid bodies and fluid. In *the Proceedings of ACM SIGGRAPH 2004* (Aug. 2004), pp. 377–384.
- [DC96] DESBRUN M., CANI M.-P.: Smoothed particles: A new paradigm for animating highly deformable bodies. In *Computer Animation and Simulation 1996* (Aug. 1996), pp. 61–76.
- [DFSG77] DONEA J., FASOLI-STELLA P., GIULIANI S.: Lagrangian and eulerian finite element techniques for transient fluid-structure interaction problems. In *Trans. 4th SMIRT Conf.* (1977).
- [DHPRF04] DONEA J., HUERTA A., PONTHOT J.-P., RODRIGUEZ-FERRAN A.: *The Encyclopedia of Computational Mechanics*. John Wiley & Sons Inc., New York, 2004.
- [EMF02] ENRIGHT D., MARSCHNER S., FEDKIW R.: Animation and rendering of complex water surfaces. In *the Proceedings of ACM SIGGRAPH 2002* (Aug. 2002), pp. 736–744.
- [FF01] FOSTER N., FEDKIW R.: Practical animation of liquids. In *the Proceedings of ACM SIGGRAPH 2001* (Aug. 2001), pp. 23–30.
- [FM97] FOSTER N., METAXAS D.: Modeling the motion of a hot, turbulent gas. In *the Proceedings of ACM SIGGRAPH 97* (Aug. 1997), pp. 181–188.
- [FOK05] FELDMAN B. E., O'BRIEN J. F., KLINGNER B. M.: Animating gases with hybrid meshes. In *Proceedings of ACM SIGGRAPH 2005* (Aug. 2005).
- [FSJ01] FEDKIW R., STAM J., JENSEN H. W.: Visual simulation of smoke. In *the Proceedings of ACM SIGGRAPH 2001* (Aug. 2001), pp. 15–22.
- [GAD03] GENEVAUX O., A H., DISCHLER J. M.: Simulating fluid-solid interaction. In *Graphics Interface* (2003), pp. 31–38.
- [GSLF05] GUENDELMAN E., SELLE A., LOSASSO F., FEDKIW R.: Coupling water and smoke to thin deformable and rigid shells. In *Proceedings of ACM SIGGRAPH 2005* (Aug. 2005), pp. 457–462.
- [HAC74] HIRT C., AMSDEN A., COOK J.: An arbitrary lagrangian-eulerian computing method for all flow speeds. *Journal of Computational Physics* 14 (1974), 227–253.
- [ITF04] IRVING G., TERAN J., FEDKIW R.: Invertible finite elements for robust simulation of large deformation. In *2004 ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (July 2004), pp. 131–140.
- [KH98] KJELLGREN P., HYVARINEN J.: An arbitrary lagrangian-eulerian finite element method. *Computational Mechanics* 21 (1998), 81–90.
- [MCG03] MÜLLER M., CHARYPAR D., GROSS M.: Particle-based fluid simulation for interactive applications. In *the ACM SIGGRAPH 2003 Symposium on Computer Animation* (Aug. 2003), pp. 154–159.
- [MKN\*04] MÜLLER M., KEISER R., NEALEN A., PAULY M., GROSS M., ALEXA M.: Point based animation of elastic, plastic



**Figure 4:** This sequence compares a fixed and deforming mesh. A slice of the actual simulation mesh is shown in green. The top row features the fixed mesh while the mesh on the bottom is squashed and stretched.



**Figure 5:** The leftmost image shows smoke inside the Stanford Buddha. The remaining images show the progression of green smoke as it is exhaled out.

- and melting objects. In *2004 ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (July 2004), pp. 141–151.
- [Pes02] PESKIN C.: The immersed boundary method. *Acta Numerica* 11 (2002), 479–517.
- [PTB\*03] PREMOŽE S., TASDIZEN T., BIGLER J., LEFOHN A., WHITAKER R.: Particle-based simulation of fluids. *Computer Graphics Forum* 22, 3 (Sept. 2003), 401–410.
- [REN\*04] RASMUSSEN N., ENRIGHT D., NGUYEN D., MARINO S., SUMNER N., GEIGER W., HOON S., FEDKIW R.: Directable photorealistic liquids. In *2004 ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (July 2004), pp. 193–202.
- [SAC\*99] STORA D., AGLIATI P.-O., CANI M.-P., NEYRET F., GASCUEL J.-D.: Animating lava flows. In *Graphics Interface 99* (June 1999), pp. 203–210.
- [SCP\*04] SHAH M., COHEN J., PATEL S., LEE P., PIGHIN F.: Extended galilean invariance for adaptive fluid simulation. In *2004 ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (July 2004), pp. 13 – 221.
- [Sta99] STAM J.: Stable fluids. In *the Proceedings of ACM SIGGRAPH 99* (Aug. 1999), pp. 121–128.
- [TPF89] TERZOPOULOS D., PLATT J., FLEISCHER K.: Heating and melting deformable models (from goop to glop). In *Graphics Interface 1989* (June 1989), pp. 219–226.
- [YOH00] YNGVE G. D., O'BRIEN J. F., HODGINS J. K.: Animating explosions. In *the Proceedings of ACM SIGGRAPH 2000* (July 2000), pp. 29–36.