Eurographics/SIGGRAPH Symposium on Computer Animation (2003)
D. Breen, M. Lin (Editors)

Particle-Based Fluid Simulation for Interactive Applications

Matthias Müller, David Charypar and Markus Gross

Department of Computer Science, Federal Institute of Technology Zürich (ETHZ), Switzerland

Abstract

Realistically animated fluids can add substantial realism to interactive applications such as virtual surgery simulators or computer games. In this paper we propose an interactive method based on Smoothed Particle Hydrodynamics (SPH) to simulate fluids with free surfaces. The method is an extension of the SPH-based technique by Desbrun to animate highly deformable bodies. We gear the method towards fluid simulation by deriving the force density fields directly from the Navier-Stokes equation and by adding a term to model surface tension effects. In contrast to Eulerian grid-based approaches, the particle-based approach makes mass conservation equations and convection terms dispensable which reduces the complexity of the simulation. In addition, the particles can directly be used to render the surface of the fluid. We propose methods to track and visualize the free surface using point splatting and marching cubes-based surface reconstruction. Our animation method is fast enough to be used in interactive systems and to allow for user interaction with models consisting of up to 5000 particles.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism

1. Introduction

1.1. Motivation

Fluids (i.e. liquids and gases) play an important role in every day life. Examples for fluid phenomena are wind, weather, ocean waves, waves induced by ships or simply pouring of a glass of water. As simple and ordinary these phenomena may seem, as complex and difficult it is to simulate them. Even though Computational Fluid Dynamics (CFD) is a well established research area with a long history, there are still many open research problems in the field. The reason for the complexity of fluid behavior is the complex interplay of various phenomena such as convection, diffusion, turbulence and surface tension. Fluid phenomena are typically simulated off-line and then visualized in a second step e.g. in aerodynamics or optimization of turbines or pipes with the goal of being as accurate as possible.

Less accurate methods that allow the simulation of fluid effects in real-time open up a variety of new applications. In the fields mentioned above real-time methods help to test whether a certain concept is promising during the design phase. Other applications for real-time simulation techniques for fluids are medical simulators, computer games or any type of virtual environment.

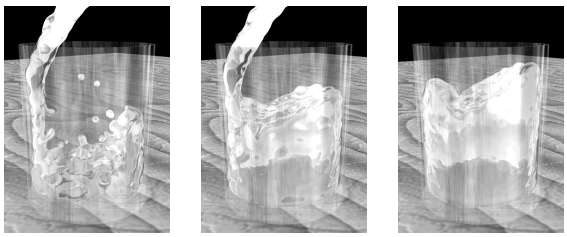

Figure 1: *Pouring water into a glass at* 5 *frames per second.*

1.2. Related Work

Computational Fluid Dynamics has a long history. In 1822 Claude Navier and in 1845 George Stokes formulated the famous Navier-Stokes Equations that describe the dynamics of fluids. Besides the Navier-Stokes equation which describes conservation of momentum, two additional equations namely a continuity equation describing mass conservation and a state equation describing energy conserva-

tion are needed to simulate fluids. Since those equations are known and computers are available to solve them numerically, a large number of methods have been proposed in the CFD literature to simulate fluids on computers.

Since about two decades, special purpose fluid simulation techniques have been developed in the field of computer graphics. In 1983 T. Reeves[17] introduced particle systems as a technique for modeling a class of fuzzy objects. Since then both, the particle-based Lagrangian approach and the grid-based Eulerian approach have been used to simulate fluids in computer graphics. Desbrun and Cani[2] and Tonnesen[22] use particles to animate soft objects. Particles have also been used to animate surfaces[7], to control implicit surfaces[23] and to animate lava flows[20]. In recent years the Eulerian approach has been more popular as for the simulation of fluids in general[18], water [4, 3, 21], soft objects[14] and melting effects[1].

So far only a few techniques optimized for the use in interactive systems are available. Stam's method[18] that is grid based is certainly an important step towards real-time simulation of fluids. For the special case of fluids that can be represented by height fields, interactive animation techniques are available as well[6].

Here we propose a particle-based approach based on Smoothed Particle Hydrodynamics to animate arbitrary fluid motion.

1.3. Our Contribution

We propose a method based on Smoothed Particles Hydrodynamics (SPH)[9] to simulate fluids with free surfaces. Stam and Fiume first introduced SPH to the graphics community to depict fire and other gaseous phenomena[19]. Later, Desbrun used SPH to animate highly deformable bodies[2]. We extend his method focussing on the simulation of fluids. To this end, we derive the viscosity and pressure force fields directly from the Navier-Stokes equation and propose a way to model surface tension forces. For the purpose of interactivity, we designed new special purpose smoothing kernels. Surface tracking and surface rendering at interactive rates are difficult problems for which we describe possible solutions.

2. Smoothed Particle Hydrodynamics

Although Smoothed Particle Hydrodynamics (SPH) was developed by Lucy[9] and Gingold and by Monaghan[5] for the simulation of astrophysical problems, the method is general enough to be used in any kind of fluid simulation. For introductions to SPH we refer the reader to Monaghan[10] or Münzel[13].

SPH is an interpolation method for particle systems. With SPH, field quantities that are only defined at discrete particle locations can be evaluated anywhere in space. For this purpose, SPH distributes quantities in a local neighborhood of each particle using radial symmetrical smoothing kernels. According to SPH, a scalar quantity A is interpolated at location $\mathbf{r}$ by a weighted sum of contributions from all particles:

$$A_S(\mathbf{r}) = \sum_j m_j \frac{A_j}{\rho_j} W(\mathbf{r} - \mathbf{r}_j, h), \tag{1}$$

where j iterates over all particles, m_j is the mass of particle j, $\mathbf{r}_j$ its position, ρ_j the density and A_j the field quantity at $\mathbf{r}_j$.

The function $W(\mathbf{r}, h)$ is called the smoothing kernel with core radius h. Since we only use kernels with finite support, we use h as the radius of support in our formulation. If W is even (i.e. $W(\mathbf{r}, h) = W(-\mathbf{r}, h)$) and normalized, the interpolation is of second order accuracy. The kernel is normalized if

$$\int W(\mathbf{r}) d\mathbf{r} = 1. \tag{2}$$

The particle mass and density appear in Eqn. (1) because each particle i represents a certain volume $V_i = m_i/\rho_i$. While the mass m_i is constant throughout the simulation and, in our case, the same for all the particles, the density ρ_i varies and needs to be evaluated at every time step. Through substitution into Eqn. (1) we get for the density at location $\mathbf{r}$:

$$\rho_S(\mathbf{r}) = \sum_j m_j \frac{\rho_j}{\rho_j} W(\mathbf{r} - \mathbf{r}_j, h) = \sum_j m_j W(\mathbf{r} - \mathbf{r}_j, h). \tag{3}$$

In most fluid equations, derivatives of field quantities appear and need to be evaluated. With the SPH approach, such derivatives only affect the smoothing kernel. The gradient of A is simply

$$\nabla A_S(\mathbf{r}) = \sum_j m_j \frac{A_j}{\rho_j} \nabla W(\mathbf{r} - \mathbf{r}_j, h) \tag{4}$$

while the Laplacian of A evaluates to

$$\nabla^2 A_S(\mathbf{r}) = \sum_j m_j \frac{A_j}{\rho_j} \nabla^2 W(\mathbf{r} - \mathbf{r}_j, h). \tag{5}$$

It is important to realize that SPH holds some inherent problems. When using SPH to derive fluid equations for particles, these equations are not guaranteed to satisfy certain physical principals such as symmetry of forces and conservation of momentum. The next section describes our SPH-based model and techniques to solve these SPH-related problems.

3. Modelling Fluids with Particles

In the Eulerian (grid based) formulation, isothermal fluids are described by a velocity field $\mathbf{v}$, a density field ρ and a pressure field p. The evolution of these quantities over time is given by two equations. The first equation assures conservation of mass

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0, \tag{6}$$

while the Navier-Stokes equation[15] formulates conservation of momentum

$$\rho\left(\frac{\partial \mathbf{v}}{\partial t}+\mathbf{v}\cdot\nabla\mathbf{v}\right)=-\nabla p+\rho\mathbf{g}+\mu\nabla^2\mathbf{v}, \qquad (7)$$

where $\mathbf{g}$ is an external force density field and μ the viscosity of the fluid. Many forms of the Navier-Stokes equation appear in the literature. Eqn. (7) represents a simplified version for incompressible fluids.

The use of particles instead of a stationary grid simplifies these two equations substantially. First, because the number of particles is constant and each particle has a constant mass, mass conservation is guaranteed and Eqn. (6) can be omitted completely. Second, the expression $\partial\mathbf{v}/\partial t+\mathbf{v}\cdot\nabla\mathbf{v}$ on the left hand side of Eqn. (7) can be replaced by the substantial derivative $D\mathbf{v}/Dt$. Since the particles move with the fluid, the substantial derivative of the velocity field is simply the time derivative of the velocity of the particles meaning that the convective term $\mathbf{v}\cdot\nabla\mathbf{v}$ is not needed for particle systems.

There are three force density fields left on the right hand side of Eqn. (7) modeling pressure ($-\nabla p$), external forces ($\rho\mathbf{g}$) and viscosity ($\mu\nabla^2\mathbf{v}$). The sum of these force density fields $\mathbf{f}=-\nabla p+\rho\mathbf{g}+\mu\nabla^2\mathbf{v}$ determines the change of momentum $\rho\frac{D\mathbf{v}}{Dt}$ of the particles on the left hand side. For the acceleration of particle i we, thus, get:

$$\mathbf{a}_i=\frac{d\mathbf{v}_i}{dt}=\frac{\mathbf{f}_i}{\rho_i}, \qquad (8)$$

where $\mathbf{v}_i$ is the velocity of particle i and $\mathbf{f}_i$ and ρ_i are the force density field and the density field evaluated at the location of particle i, repectively. We will now describe how we model the force density terms using SPH.

3.1. Pressure

Application of the SPH rule described in Eqn. (1) to the pressure term $-\nabla p$ yields

$$\mathbf{f}_i^{\text{pressure}}=-\nabla p(\mathbf{r}_i)=-\sum_j m_j\frac{p_j}{\rho_j}\nabla W(\mathbf{r}_i-\mathbf{r}_j,h). \qquad (9)$$

Unfortunately, this force is not symmetric as can be seen when only two particles interact. Since the gradient of the kernel is zero at its center, particle i only uses the pressure of particle j to compute its pressure force and vice versa. Because the pressures at the locations of the two particles are not equal in general, the pressure forces will not be symmetric. Different ways of symmetrization of Eqn. (9) have been proposed in the literature. We suggest a very simple solution which we found to be best suited for our purposes of speed and stability

$$\mathbf{f}_i^{\text{pressure}}=-\sum_j m_j\frac{p_i+p_j}{2\rho_j}\nabla W(\mathbf{r}_i-\mathbf{r}_j,h).. \qquad (10)$$

The so computed pressure force is symmetric because it uses the arithmetic mean of the pressures of interacting particles.

Since particles only carry the three quantities mass, position and velocity, the pressure at particle locations has to be evaluated first. This is done in two steps. Eqn. (3) yields the density at the location of the particle. Then, the pressure can be computed via the ideal gas state equation

$$p=k\rho, \qquad (11)$$

where k is a gas constant that depends on the temperature. In our simulations we use a modified version of Eqn. (11) suggested by Desbrun[2]

$$p=k(\rho-\rho_0), \qquad (12)$$

where ρ_0 is the rest density. Since pressure forces depend on the gradient of the pressure field, the offset mathematically has not effect on pressure forces. However, the offset does influence the gradient of a field smoothed by SPH and makes the simulation numerically more stable.

3.2. Viscosity

Application of the SPH rule to the viscosity term $\mu\nabla^2\mathbf{v}$ again yields asymmetric forces

$$\mathbf{f}_i^{\text{viscosity}}=\mu\nabla^2\mathbf{v}(\mathbf{r}_a)=\mu\sum_j m_j\frac{\mathbf{v}_j}{\rho_j}\nabla^2 W(\mathbf{r}_i-\mathbf{r}_j,h). \qquad (13)$$

because the velocity field varies from particle to particle. Since viscosity forces are only dependent on velocity differences and not on absolute velocities, there is a natural way to symmetrize the viscosity forces by using velocity differences:

$$\mathbf{f}_i^{\text{viscosity}}=\mu\sum_j m_j\frac{\mathbf{v}_j-\mathbf{v}_i}{\rho_j}\nabla^2 W(\mathbf{r}_i-\mathbf{r}_j,h). \qquad (14)$$

A possible interpretation of Eqn. (14) is to look at the neighbors of particle i from i's own moving frame of reference. Then particle i is accelerated in the direction of the relative speed of its environment.

3.3. Surface Tension

We model surface tension forces (not present in Eqn. (7)) explicitly based on ideas of Morris[12]. Molecules in a fluid are subject to attractive forces from neighboring molecules. Inside the fluid these intermolecular forces are equal in all directions and balance each other. In contrast, the forces acting on molecules at the free surface are unbalanced. The net forces (i.e. surface tension forces) act in the direction of the surface normal towards the fluid. They also tend to minimize the curvature of the surface. The larger the curvature, the higher the force. Surface tension also depends on a tension coefficient σ which depends on the two fluids that form the surface.

The surface of the fluid can be found by using an additional field quantity which is 1 at particle locations and 0

everywhere else. This field is called *color field* in the literature. For the smoothed color field we get:

$$c_S(\mathbf{r}) = \sum_j m_j \frac{1}{\rho_j} W(\mathbf{r} - \mathbf{r}_j, h). \tag{15}$$

The gradient field of the smoothed color field

$$\mathbf{n} = \nabla c_s \tag{16}$$

yields the surface normal field pointing into the fluid and the divergence of $\mathbf{n}$ measures the curvature of the surface

$$\kappa = \frac{-\nabla^2 c_s}{|\mathbf{n}|}. \tag{17}$$

The minus is necessary to get positive curvature for convex fluid volumes. Putting it all together, we get for the surface traction:

$$\mathbf{t}^{\text{surface}} = \sigma\kappa\frac{\mathbf{n}}{|\mathbf{n}|} \tag{18}$$

To distribute the surface traction among particles near the surface and to get a force density we multiply by a normalized scalar field $\delta_s = |\mathbf{n}|$ which is non-zero only near the surface. For the force density acting near the surface we get

$$\mathbf{f}^{\text{surface}} = \sigma\kappa\mathbf{n} = -\sigma\nabla^2 c_S \frac{\mathbf{n}}{|\mathbf{n}|} \tag{19}$$

Evaluating $\mathbf{n}/|\mathbf{n}|$ at locations where $|\mathbf{n}|$ is small causes numerical problems. We only evaluate the force if $|\mathbf{n}|$ exceeds a certain threshold.

3.4. External Forces

Our simulator supports external forces such as gravity, collision forces and forces caused by user interaction. These forces are applied directly to the particles without the use of SPH. When particles collide with solid objects such as the glass in our examples, we simply push them out of the object and reflect the velocity component that is perpendicular to the object's surface.

3.5. Smoothing Kernels

Stability, accuracy and speed of the SPH method highly depend on the choice of the smoothing kernels. The kernels we use have second order interpolation errors because they are all even and normalized (see Fig. 2). In addition, kernels that are zero with vanishing derivatives at the boundary are conducive to stability. Apart from those constraints, one is free to design kernels for special purposes. We designed the following kernel

$$W_{\text{poly6}}(\mathbf{r}, h) = \frac{315}{64\pi h^9} \begin{cases} (h^2 - r^2)^3 & 0 \le r \le h \\ 0 & \text{otherwise} \end{cases} \tag{20}$$

and use it in all but two cases. An important feature of this simple kernel is that r only appears squared which means

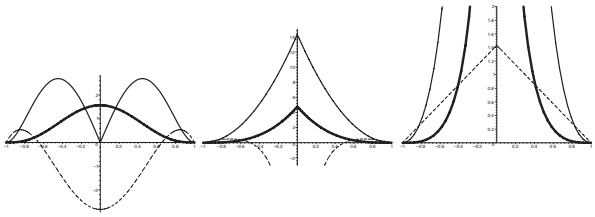

Figure 2: *The three smoothing kernels W_{poly6}, W_{spiky} and $W_{viscosity}$ (from left to right) we use in our simulations. The thick lines show the kernels, the thin lines their gradients in the direction towards the center and the dashed lines the Laplacian. Note that the diagrams are differently scaled. The curves show 3-d kernels along one axis through the center for smoothing length $h = 1$.*

that is can be evaluated without computing square roots in distance computations. However, if this kernel is used for the computation of the pressure forces, particles tend to build clusters under high pressure. As particles get very close to each other, the repulsion force vanishes because the gradient of the kernel approaches zero at the center. Desbrun[2] solves this problem by using a spiky kernel with a non vanishing gradient near the center. For pressure computations we use Debrun's spiky kernel

$$W_{\text{spiky}}(\mathbf{r}, h) = \frac{15}{\pi h^6} \begin{cases} (h - r)^3 & 0 \le r \le h \\ 0 & \text{otherwise,} \end{cases} \tag{21}$$

that generates the necessary repulsion forces. At the boundary where it vanishes it also has zero first and second derivatives.

Viscosity is a phenomenon that is caused by friction and, thus, decreases the fluid's kinetic energy by converting it into heat. Therefore, viscosity should only have a smoothing effect on the velocity field. However, if a standard kernel is used for viscosity, the resulting viscosity forces do not always have this property. For two particles that get close to each other, the Laplacian of the smoothed velocity field (on which viscosity forces depend) can get negative resulting in forces that increase their relative velocity. The artifact appears in coarsely sampled velocity fields. In real-time applications where the number of particles is relatively low, this effect can cause stability problems. For the computation of viscosity forces we, thus, designed a third kernel:

$$W_{\text{viscosity}}(\mathbf{r}, h) = \frac{15}{2\pi h^3} \begin{cases} -\frac{r^3}{2h^3} + \frac{r^2}{h^2} + \frac{h}{2r} - 1 & 0 \le r \le h \\ 0 & \text{otherwise.} \end{cases} \tag{22}$$

whose Laplacian is positive everywhere with the following

additional properties:

$$\begin{aligned} \nabla^2 W(\mathbf{r},h) &= \frac{45}{\pi h^6}(h-r) \\ W(|\mathbf{r}|=h,h) &= 0 \\ \nabla W(|\mathbf{r}|=h,h) &= \mathbf{0} \end{aligned}$$

The use of this kernel for viscosity computations increased the stability of the simulation significantly allowing to omit any kind of additional damping.

3.6. Simulation

For the integration of the Eqn. (8) we use the Leap-Frog scheme[16]. As a second order scheme for which the forces need to be evaluation only once, it best fits our purposes and in our examples allows time steps up to 10 milliseconds. For the examples we used constant time steps. We expect even better performance if adaptive time steps are used based on the Courant-Friedrichs-Lewy condition[2].

4. Surface Tracking and Visualization

The color field c_S and its gradient field $\mathbf{n} = \nabla c_S$ defined in section 3.3 can be used to identify surface particles and to compute surface normals. We identify a particle i as a surface particle if

$$|\mathbf{n}(\mathbf{r}_i)| > l, \tag{23}$$

where l is a threshold parameter. The direction of the surface normal at the location of particle i is given by

$$-\mathbf{n}(\mathbf{r}_i). \tag{24}$$

4.1. Point Splatting

We now have a set of points with normals but without connectivity information. This is exactly the type of information needed for point splatting techniques[24]. However, these methods are designed to work with point clouds obtained from scanners that typically contain at least $10,000$ to $100,000$ points. We only use a few thousand particles a fraction of which are identified as being on the surface. Still surface splatting yields plausible results as shown in the results section.

We are currently working on ways to upsample the surface of the fluid. Hereby, the color field information of surface particles is interpolated to find locations for additional particles on the surface only used for rendering.

4.2. Marching Cubes

Another way to visualize the free surface is by rendering an iso surface of the color field c_S. We use the marching cubes algorithm[8] to triangulate the iso surface. In a grid fixed in space the cells that contain the surface are first identified. We start searches from all the cells that contain surface particles and from there recursively traverse the grid along the surface. With the use of a hash table we make sure that the cells are not visited more than once. For each cell identified to contain the surface, the triangles are generated via a fast table lookup.

5. Implementation

Since the smoothing kernels used in SPH have finite support h, a common way to reduce the computational complexity is to use a grid of cells of size h. Then potentially interacting partners of a particle i only need to be searched in i's own cell and all the neighboring cells. This technique reduces the time complexity of the force computation step from $O(n^2)$ to $O(nm)$, m being the average number of particles per grid cell.

With a simple additional trick we were able to speed up the simulation by an additional factor of 10. Instead of storing references to particles in the grid, we store copies of the particle objects in the grid cells (doubling memory consumption). The reason for the speed up is the proximity in memory of the information needed for interpolation which dramatically increases the cash hit rate. Further speedup might be possible through even better clustering using Hilbert space filling curves[11]. The data structure for fast neighbor searches is also used for surface tracking and rendering.

6. Results

The water in the glass shown in Fig. 3 is sampled with 2200 particles. An external rotational force field causes the fluid to swirl. The first image (a) shows the individual particles. For the second image (b), point splatting was used to render the free surface only. In both modes, the animation runs at 20 frames per second on a 1.8 GHz Pentium IV PC with a GForce 4 graphics card. The most convincing results are produced when the iso surface of the color field is visualized using the marching cubes algorithm as in image (c). However, in this mode the frame rate drops to 5 frames per second. Still this frame rate is significantly higher than the one of most off-line fluid simulation techniques and with the next generation of graphics hardware, real-time performance will be possible.

The image sequence shown in Fig. 4 demonstrates interaction with the fluid. Through mouse motion, the user generates an external force field that cause the water to splash. The free surface is rendered using point splatting while isolated particles are drawn as single droplets. The simulation with 1300 particles runs at 25 frames per second.

For the animation shown in Fig. 5 we used 3000 particles and rendered the surface with the marching cubes technique at 5 frames per second.

7. Conclusions and Future Work

We have presented a particle-based method for interactive fluid simulation and rendering. The physical model is based on Smoothed Particle Hydrodynamics and uses special purpose kernels to increase stability and speed. We have presented techniques to track and render the free surface of fluids. The results are not as photorealistic yet as animations computed off-line. However, given that the simulation runs at interactive rates instead of taking minutes or hours per frame as in today's off-line methods, the results are quite promising.

While we are quite content with the physical model, tracking and rendering of the fluid surface in real time certainly remains an open research problem. In the future we will investigate upsampling techniques as well as ways to increase the performance of the marching cubes-based algorithm.

Acknowledgements

The authors would like to thank Simone Hieber for her helpful comments and Rolf Bruderer, Simon Schirm and Thomas Rusterholz for their contributions to the real-time system.

References

1. Mark Carlson, Peter J. Mucha, III R. Brooks Van Horn, and Greg Turk. Melting and flowing. In *Proceedings of the ACM SIGGRAPH symposium on Computer animation*, pages 167–174. ACM Press, 2002.

2. M. Desbrun and M. P. Cani. Smoothed particles: A new paradigm for animating highly deformable bodies. In *Computer Animation and Simulation '96 (Proceedings of EG Workshop on Animation and Simulation)*, pages 61–76. Springer-Verlag, Aug 1996.

3. D. Enright, S. Marschner, and R. Fedkiw. Animation and rendering of complex water surfaces. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 736–744. ACM Press, 2002.

4. N. Foster and R. Fedkiw. Practical animation of liquids. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 23–30. ACM Press, 2001.

5. R. A. Gingold and J. J. Monaghan. Smoothed particle hydrodynamics: theory and application to non-spherical stars. *Monthly Notices of the Royal Astronomical Society*, 181:375–398, 1977.

6. Damien Hinsinger, Fabrice Neyret, and Marie-Paule Cani. Interactive animation of ocean waves. In *Proceedings of the ACM SIGGRAPH symposium on Computer animation*, pages 161–166. ACM Press, 2002.

7. Jean-Christophe Lombardo and Claude Puech. Oriented particles: A tool for shape memory objects modelling. In *Graphics Interface'95*, pages 255–262, mai 1995. Quebec city, Canada.

8. William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 163–169. ACM Press, 1987.

9. L. B. Lucy. A numerical approach to the testing of the fission hypothesis. *The Astronomical Journal*, 82:1013–1024, 1977.

10. J. J. Monaghan. Smoothed particle hydrodynamics. *Annual Review of Astronomy and Astrophysics*, 30:543–574, 1992.

11. B. Moon, H. V. Jagadish, C. Faloutsos, and J. H. Saltz. Analysis of the clustering properties of hilbert space-filling curve. *IEEE Transactions on Knowledge and Data Engineering*, 13(1):124–141, 2001.

12. J. P. Morris. Simulating surface tension with smoothed particle hydrodynamics. *International Journal for Numerical Methods in Fluids*, 33(3):333–353, 2000.

13. S. A. Münzel. *Smoothed Particle Hydrodynamics und ihre Anwendung auf Akkretionsscheiben*. PhD thesis, Eberhard-Karls-Universität Thübingen, 1996.

14. D. Nixon and R. Lobb. A fluid-based soft-object model. *IEEE Computer Graphics and Applications*, pages 68–75, July/August 2002.

15. D. Pnueli and C. Gutfinger. *Fluid Mechanics*. Cambridge Univ. Press, NY, 1992.

16. C. Pozrikidis. *Numerical Computation in Science and Engineering*. Oxford Univ. Press, NY, 1998.

17. W. T. Reeves. Particle systems — a technique for modeling a class of fuzzy objects. *ACM Transactions on Graphics 2(2)*, pages 91–108, 1983.

18. Jos Stam. Stable fluids. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 121–128. ACM Press/Addison-Wesley Publishing Co., 1999.

19. Jos Stam and Eugene Fiume. Depicting fire and other gaseous phenomena using diffusion processes. *Computer Graphics*, 29(Annual Conference Series):129–136, 1995.

20. Dan Stora, Pierre-Olivier Agliati, Marie-Paule Cani, Fabrice Neyret, and Jean-Dominique Gascuel. Animating lava flows. In *Graphics Interface*, pages 203–210, 1999.

21. T. Takahashi, U. Heihachi, A. Kunimatsu, and H. Fujii. The simulation of fluid-rigid body interaction. *ACM Siggraph Sketches & Applications*, July 2002.

22. D. Tonnesen. *Dynamically Coupled Particle Systems for Geometric Modeling, Reconstruction, and Animation*. PhD thesis, University of Toronto, November 1998.

23. Andrew Witkin and Paul Heckbert. Using particles to sample and control implicit surfaces. In *Computer Graphics (Proc. SIGGRAPH '94)*, volume 28, 1994.

24. Matthias Zwicker, Hanspeter Pfister, Jeroen van Baar, and Markus Gross. Surface splatting. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 371–378. ACM Press, 2001.

(a) (b) (c)

Figure 3: *A swirl in a glass induced by a rotational force field. Image (a) shows the particles, (b) the surface using point splatting and (c) the iso-surface triangulated via marching cubes.*

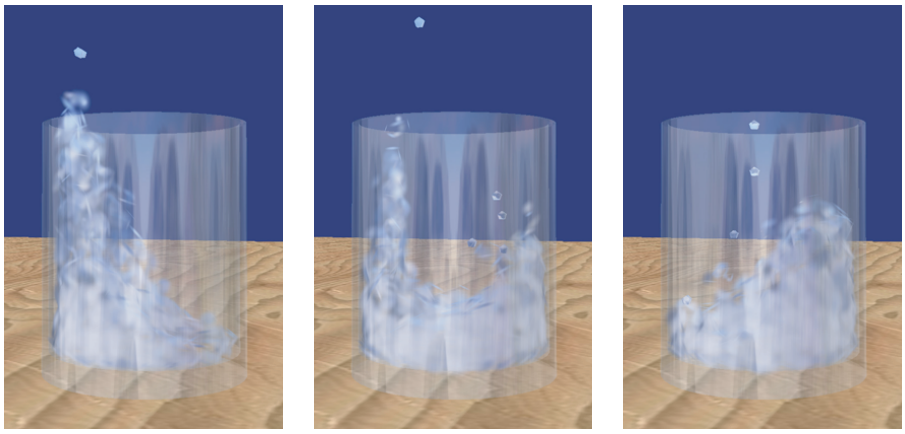

Figure 4: *The user interacts with the fluid causing it to splash.*

Figure 5: *Pouring water into a glass at* 5 *frames per second.*