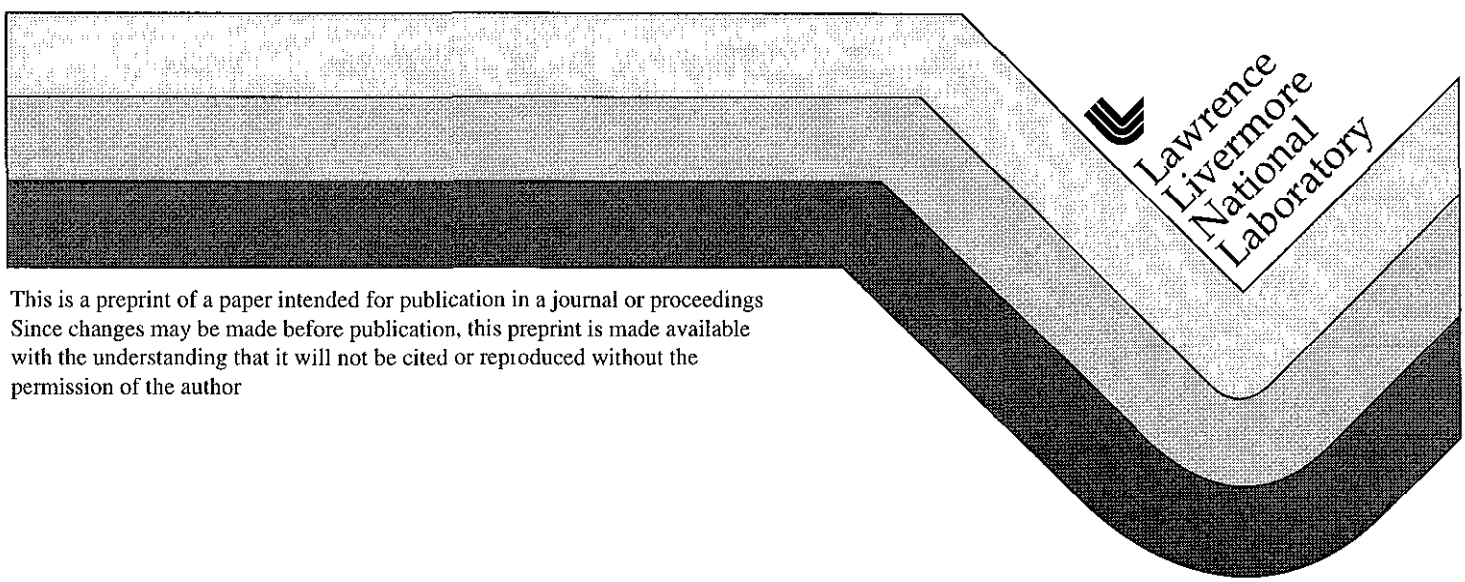
UCRL-JC-129378
PREPRINT

A Particle-Based Model for Water Simulation

C.M Stein
N L Max

This paper was prepared foı submittal to the
SIGGRAPH '98
Orlando, FL
July 19-24, 1998

January 1998



A Particle-Based Model for Water Simulation

Clifford M Stein* Nelson L Max†

Lawrence Livermore National Laboratory
University of California, Davis

Abstract

The Smooth-Particle Applied Mechanics (SPAM) model is a relatively recent physical modeling technique It can model both fluids and solids using free-moving particles An implemented SPAM model is described that solved the compressible Navier-Stokes equations to produce animations of splashing and pooling water Because the particle positions are known explicitly each timestep, the SPAM technique produces data amenable to visualization A ray-tracing renderer is also described It samples the underwater light-field distribution and stores the information into a Light Accumulation Lattice which is used for scattered light calculations and caustics

1 Introduction

Practical water modeling has been a long time goal of the graphics community In the past, researchers have had to implement simplified water models because of the lack of available CPU power to solve sophisticated physical models However, with the past growth in computing power, graphics researchers can now draw upon the past works of physicists to implement models with realistic looking behavior Coupled with advances in rendering techniques, today's images are attaining unprecedented realism

Both physically based and non-physically based water models have been developed in past graphics research Physically-based models are computer formulations of the mathematical descriptions of fluids, such as the oft-mentioned Navier-Stokes Equations found in Appendix B While the physics and mathematics are based on basic principles such as mass, momentum, and energy conservation, the resulting equations describing the fluid motion can be difficult to understand both conceptually and mathematically Numerical solvers of these equations must be crafted with care in order to avoid instabilities and to ensure accurate results It is the attempt to de-emphasize these issues, or to avoid them entirely, that leads some graphics researchers to use non-physically based models

1.1 Non-Physically Based Models

Reeves, [19], described an early non-physically based model that employed vast numbers of tiny moving particles to suggest turbulent fluid movement Only external body forces such as gravity and repulsive boundaries influenced the motion of these particles The particles themselves could not interact with each other While this model was capable of generating good looking turbulent behavior, such as splashes and waterfalls, the model was unsuitable for modeling lakes and other bodies of water due to the particles' inability to "support" each other

Miller and Pearce [12] described a simple particle system that was used to animate viscous fluids Their system consisted of particles that interacted with each other through simple repulsive and attractive forces which were functions of the distance separating pairs of particles By adjusting the parameters of the interaction forces, they could vary the types of materials they were modeling The gamut of materials they could simulate ranged from "powders" to "fluids" to "solids" Their model, while producing good-looking highly-viscous substances, was not appropriate for modeling inviscid fluids such as water

Chiba [4] presented a "quasi-physically-based" fluid model which attempted to capture the complex behavior of water Theirs was a particle method, like Reeves's But, the motion of their particles was governed by a set of heuristics which endeavored to address some of the shortcomings of Reeves's method For example, the particles in Chiba's model had volume, and were capable of interacting with each other based on a set of simple rules This allowed Chiba's particles to collect and pool

1.2 Physically Based Models

There have also been several types of physically based water models implemented in past graphics research One of the earlier models, Max's [11], was simply a summation of sinusoidal waveforms of varying frequencies and amplitudes using the Stokes coefficients to model the deep-ocean waves

More recent techniques, such as Fournier [7] and Peachey [18], used trochoids to model the ocean surface It is useful and correct to describe the motion of particles of water throughout a deep water wave as an elliptical path whose orbit is a function of the depth of the water and the amplitude of the wave Using these facts, Fournier and Peachey generated images of waves rolling up to the beach In their images, one can watch the long smooth deep-ocean waves refract as they enter shallower water and grow choppier Because the waves themselves were just height fields on a two dimension lattice, the waves could never truly fold over themselves as they crashed Fournier added a wind force to help "bend" the wave over, thus giving the appearance of a plunging wave

Kass and Miller [9] implemented a finite difference scheme to solve a set of simplified shallow-water equations based upon a 2D height-field Their model exhibited wave refraction and wave reflection off boundaries However, because theirs was a height-field approach, it could not produce splashes or crashing waves

Foster [6] implemented a 3D incompressible finite difference water simulation capable of capturing a variety of effects In order to render his images, however, Foster placed marker particles in various locations throughout the model and watched them advect according to the local velocities

Another disadvantage of full-fledged three-dimensional simulations is the heavy toil on computer resources Chen et al [3] attempted to reduce this toll by computing the fluid flow in two dimensions only, but using the pressure to specify the height field

This paper presents an alternative particle-based physical-modeling technique known as both *Smooth-Particle Applied Mechanics* (SPAM) and *Smooth-Particle Hydrodynamics* (SPH) [10], [13], [14] [8] In this method particles are free to move about, and, because of this, it can capture the effects of splashing and pooling In addition to this, the position of each particle is known explicitly each timestep, its data is amenable to visualization

*kliph@ix netcom com
†max2@llnl gov

2 Water Modeling

In discretizing the continuum, one must decide in which "framework" to work The *Lagrangian* framework describes the motion of particles of mass in the continuum The *Eulerian* framework describes the behavior of the fluid at any fixed location in the modeling space If one were to place mobile sensors that tracked the density, motion, and temperature of particle masses as they flowed through the modeling space, then one would be operating in the Lagrangian framework The Eulerian framework can be likened to placing stationary sensors throughout the fluid that are capable of measuring the changes in densities, velocities and temperatures at fixed locations

Because explicit knowledge of the fluid distribution throughout the system simplifies the rendering process, Lagrangian models are well suited for visualization purposes

2.1 Discretization Methods

A variety of numerical methods are available to solve systems of partial differential equations (PDEs) such as the Navier-Stokes In schemes like Finite Difference Methods (FDM), Finite Element Methods (FEM), and Smooth-Particle Applied Mechanics(SPAM), state values such as density, thermal energy, velocity and mass, are stored in a collection of "nodes" which are distributed throughout the system These nodes represent either spatial locations or particle masses The arrangement of nodes is guided by the need to quickly, and accurately, determine spatial derivatives

Finite difference schemes, long popular for their simplicity, use structured grids, and are suited for the Eulerian framework Continuous operations like spatial derivatives are replaced by taking weighted "differences" of values from neighboring nodes Although simple to implement, FDMs such as Foster's, generate data whose surfaces are not amenable to visualization Furthermore, FDMs are ill-suited to handle problems with irregular domains

Finite Element methods, which can be regarded as a superset of FDMs, offer much more flexibility in handling irregular geometries because they are designed for unstructured meshes Weighted combinations of basis functions (which are localized) determine the state values at any given location Consequently, spatial derivatives are determined by a weighted sum of the basis function gradients Although very attractive mathematically, FEMs are difficult to implement FEMs also have particular difficulty with the Lagrangian framework because of the possibility that edges might cross each other as nodes move about; this is known as "tangling"

It is the shortcomings of FDM and FEM that SPAM attempts to resolve In this method, state values are derived from a weighted combination of neighboring interpolation points Spatial derivatives are similarly calculated using the weighting function Due to its free-form nature, the SPAM technique easily allows both Eulerian and Lagrangian calculations to be performed In a Lagrangian framework, the interpolation points represent particles of mass In an Eulerian framework, the interpolation points are just locations in the domain, as in FDMs

2.2 SPAM Equations of Motion

The SPAM modeling technique was first described by Lucy, [10], and widely popularized by Monaghan [13], and [14] among others It has been employed to solve a wide variety of problems ranging from small scale fluid motion to calculation of astrophysical problems (its first reported application) to the modeling of solids

Both[13] and [8] provide good descriptions and analyses of the SPAM technique For the sake of brevity, only a quick derivation is provided in Appendix A

A collection of particles, also called interpolation points, make up the material of interest In a Lagrangian frame, these particles are free to move about, while in an Eulerian scheme, they are fixed The particles have mass, m, density ρ, internal energy, e, and velocity, $\mathbf{u}$

The Navier-Stokes equations of motion for a compressible fluid are written in the following SPAM formulation:

The *continuity equation* describes the density change in a fluid For incompressible flows such as water, $D\rho/Dt = 0$ For compressible flows, the SPAM description for the change in density is

$$\frac{d\rho_i}{dt} = \sum_j m_j(\mathbf{u}_i - \mathbf{u}_j)\ \nabla W_{ij} \tag{1}$$

The *momentum equation* describes the accelerations that particles of mass undergo Accelerations are calculated by the following equation:

$$\frac{d\mathbf{u}_i}{dt} = \sum_j m_j \left(\frac{\sigma_i}{\rho_i^2} + \frac{\sigma_j}{\rho_j^2}\right)\ \nabla W_{ij} + \mathbf{g} \tag{2}$$

where $\mathbf{g}$ denotes the external gravity field

The *energy equation* describes the change in internal energy (temperature) for a fluid It takes into account the effects of viscosity as well as the "work" done on the system in the form of compression and expansion For the models where temperature has a negligible effect on pressure, such as water models, this equation can often be ignored in order to save CPU cycles The SPAM description of energy change is as follows

$$\frac{de_i}{dt} = -\sum_j m_j \left(\frac{\sigma_i}{\rho_i^2} + \frac{\sigma_j}{\rho_j^2}\right) : \mathbf{u}\nabla W_{ij} - \sum_j m_j \left(\frac{\mathbf{Q}_i}{\rho_i^2} + \frac{\mathbf{Q}_j}{\rho_j^2}\right)\ \nabla W_{ij}, \tag{3}$$

where $\mathbf{Q}$, in the above equation, is the heat-flux vector and is defined by

$$\mathbf{Q}_i \equiv -\kappa\nabla T_i = -\kappa \sum_j m_j \frac{(T_i - T_j)}{\rho_{ij}} \nabla W_{ij} \tag{4}$$

T_i is the temperature of particle i and ρ_{ij} is the symmetrized mean density as described in Appendix A

The double dot-product notation, $A : B$, is defined by Equation (23)

The stress tensor σ and heat flux vector $\mathbf{Q}$, which are defined by (21) and (22), respectively, are easily generated in the SPAM framework with the techniques already discussed in the appendix

Equations (1), (2), and (3) form a system of ordinary differential equations (ODEs) that are relatively easy to evaluate The SPAM method's ability to reduce the Navier-Stokes equations, which form a non-trivial set of PDEs, to an ODE system without any needs for grids or meshes makes it simple and easy to implement

2 3 Weighting Functions

It is widely recognized that the results of an SPAM calculation can vary depending on the weighting function used Hoover [8] presents a discussion of several weighting functions and results of their use For purposes of this paper, Lucy's kernel was used

$$W_L(r,h) = (1+3\frac{r}{h})(1-\frac{r}{h})^3 \qquad 0 \le r \le h \tag{5}$$

2 4 SPAM Implementation

Calculations of the time derivatives is a time consuming process involving three[1] $O(n^2)$ traversals through a collection of n particles The quadratic term is due to the worst case scenario when all

[1]Density and velocity gradient calculations are performed during the first pass through the particles The second pass is used to calculate the stress

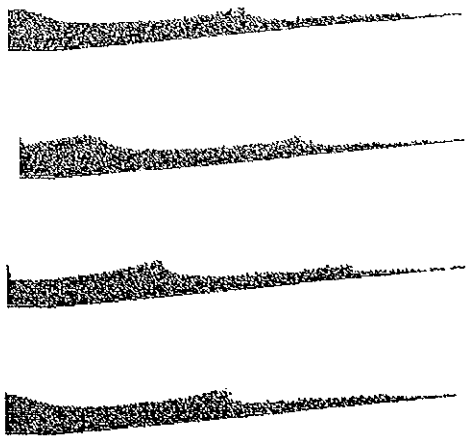


Figure 1 A "piston" comprised of repelling particles oscillates back and forth on a track to produce waves This simulation contained 2345 particles

particle-pair interactions must be evaluated However, because we used a weighting kernel with a finite width, we were able to divide the modeling regime into boxes where particles only interacted with particles in neighboring boxes This improves the running time dramatically

3 Simulation Results

The following section discusses two simulations of water using the SPAM method

3 1 Water Waves

Figure 1 illustrates the evolution of a water wave The run was a 2D simulation containing 2345 particles A repelling plane boundary served as the ocean floor to contain the particles; particles that strayed too close to the ocean floor were pushed up and away from the boundary The strength of the push was inversely proportional to the distance the particle was to the boundary The leftmost boundary served as the wave-maker It consisted of a vertical "column" of repelling particles that oscillated on a track This motion induced the rolling waves which crested and plunged as they approached the shore

The particles were one kilogram in size and were initially arranged on a 0 1 by 0 1 meter lattice (which produced an average initial density, ρ_o, of 100 kg/meter^2—this simulation was a 2D calculation) The energy independent equation of state, $P = B((\rho/\rho_o)^2 - 1)$ was used This produces a fluid with a speed of sound, c, equal to $c^2 = \partial P/\partial\rho \approx 2B/\rho_o$ The described simulations used a speed of sound approximately equal to 20 meters per second Thus, an incompressible fluid, such as water, was approximated by a slightly compressible fluid

After the particles were placed in their initial configuration, the simulation was run with stationary boundaries This gave the particles an opportunity to "relax" Once the initial motion had damped out, the oscillating wave-maker was put into motion

and strain-rate tensors, and the heat flux vector Finally, the inter-particle forces are calculated during the last pass

3 2 Falling Column of Water

The second simulation presented in this paper is that of a collapsing column of water A water column is placed inside a box and "released" It then proceeds to collapse and fill the box This was a purely three dimensional "water" simulation This simulation contained 2755 particles Due to the turbulence of the simulation, the speed of sound for this simulation was approximately 30 meters per second The walls of the container were repelling plane boundaries, as in the wave-maker example Three frames of animation are illustrated in Figure 4

4 Rendering

In addition to producing water with realistic looking behavior, we wanted render realistic looking images of the water A fully-recursive ray-tracer was implemented to capture the interaction between light and water It was capable of modeling scattered light and caustics

Because SPAM particles are really mass distributions which made the Metaball [16] and "Blobby" [2] rendering methods seemed to be the obvious visualization technique to use However, metaball rendering proved slow Instead, a Marching Cubes [5] algorithm was implemented Holes in the tessellation of the surface were eliminated by using Montani's [15] alternative "complementary" triangulation scheme

The SPAM definition of density is a convenient thresholding function to use; the thresholding value can be something fairly close to that of water (1000 kg/m^3) Surface normals were found through a variation of Blinn's bump-mapping algorithm [1] in that they were calculated by evaluating $-\nabla\rho$ on the facet surfaces This produces smoothly varying surface normals across coarsely faceted surfaces is popular because it results in excellent image quality with relatively low numbers of surface facets

4.1 Illumination

To produce realistic images of water such as caustic effects and light scattering, we needed to know the structure of the light field throughout the water Our algorithm is derived from the works of Watt [20] and Nishita [17]

To calculate the amount of light scattered towards the viewer's eye by the water, we must effectively solve the following equation for those viewing rays that pierce the water

$$L_{AB} = \int_A^B e^{-\tau|x-A|} \int_{4\pi} \beta(\mathbf{x}, \omega, \omega') E_i(\mathbf{x}, \omega') d\omega' dx \qquad (6)$$

The term L_{AB} describes the radiance accumulated along a ray segment AB towards the eye through a transmitting medium where τ is the attenuation coefficient per unit length and $\beta(\mathbf{x}, \omega, \omega')$ is the directional volume scattering coefficient at point $\mathbf{x}$ from the direction ω' to the direction ω $E_i(\mathbf{x}, \omega')$ is the incident irradiance from the direction ω' at point $\mathbf{x}$ Thus, according the (6), we need to know the underwater light-field distribution in order to calculate the amount of light scattered toward the eye

To solve (6), we, like Watt and Nishita, fill our fluid volume with light volumes Our surface was tessellated using the Marching Cubes algorithm We then created light volumes by shooting rays from the light sources in the scene through every triangular facet on the surface We then created prism-like light volumes by refracting the light rays as they crossed the MC surface and tracing them through as they transmit through the particle collection Ray tracing the light volumes explicitly to calculate the scattered light and caustics proved time consuming and difficult, the sides are non-planar and light volumes are potentially self-intersecting

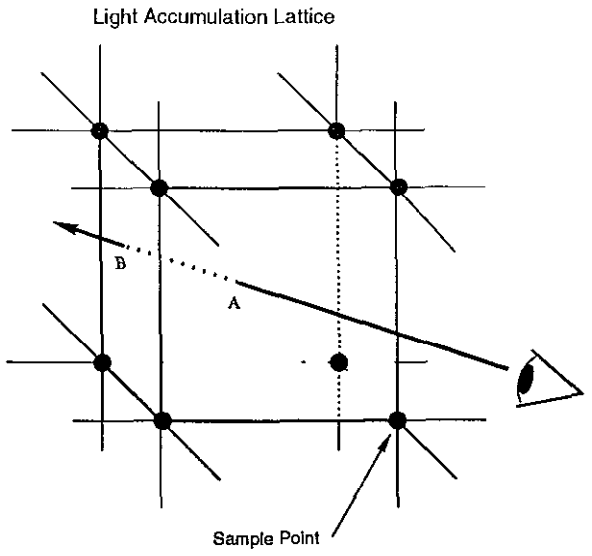


Figure 2 A viewing ray entering a Light Accumulation Lattice cell at point A and exiting and point B

Instead, the underwater light-field distribution was sampled on a uniform lattice into a datastructure called a *Light Accumulation Lattice* (LAL) Irradiance vectors detailing the direction and the energy per unit area flowing through the sample points were stored on the lattice's nodes Because a sample point could lie in multiple light volumes, irradiance vectors were stored using linked lists Thus, the light field anywhere in the volume can be trilinearly interpolated from the sample points in the LAL Ambient light from the sky was stored as a downward pointing irradiance vector in all of the sample points that lay in the water All irradiance values were corrected for transmission attenuation and for expansion and compression of the light volume, the light grows "brighter" as the light is "focused" into a smaller area due to the increased energy density (irradiance)

Equation (6) was solved approximately by making several simplifying assumptions First, the radiance of the light scattered in the direction of the eye, L_A and L_B, was only evaluated when a viewing ray entered and exited the box formed by eight neighboring sample points as illustrated in Figure 2 Second, it was assumed that the radiance varied linearly between points A and B This gives us an approximation to (6) as follows:

$$L_{scattered} = \int_0^D e^{-\tau x}\left(\frac{D-x}{D}L_A + \frac{x}{D}L_B\right)dx \tag{7}$$

where D is the distance between points A and B

Integrated analytically, (7) becomes:

$$L_{scattered} = \frac{L_A}{\tau}\left(1-e^{-\tau D}\right) + \frac{L_B - L_A}{D\tau^2}\left(1-(1+\tau D)e^{-\tau D}\right) \tag{8}$$

Caustics on submerged surfaces were calculated from the sampled irradiance vectors Figure 3 illustrates the combination of light scattering and caustic effects on the bottom of a swimming pool The water dataset contained 15912 particles arranged in a simple sinusoid pattern The final image contained 34798 Marching Cube facets 10164 illumination volumes were processed into a 63 by 63 by 63 LAL Preprocessing (extracting the isosurface and filling the LAL) took 757 seconds on a 266MHz PentiumII-based computer Rendering took only 941 seconds

Figure 4 shows three frames on animations of a collapsing column of water The animation contained 2755 particles of water

5 Conclusion and Future Work

The Smooth-Particle Applied Mechanics technique is a practical method for calculating fluid motion with free surface flows The SPAM method easily allows one to turn a system of partial differential equations into a set of ordinary differential equations The reduction does not require any grids which easily allows one to follow the motion of all of the particles in the system without worrying about mesh entanglement

While the simulations described in this paper modeled water by approximating a compressible liquid, perhaps future work could be directed to modeling purely incompressible flows Also, because computation time is proportional to the number of particles in the system, it would be beneficial if one could develop a simulation where groups of slow moving particles could be "collapsed" into single large particles and vice versa

The technique of storing the underwater light-field in the Light Accumulation Lattice leads to relatively quick renderings, but uses a great deal of memory and leaves less than satisfactory underwater caustics due to the interpolation If viewed from above the water, the distortion due to the refraction effects lessens the effect

6 Acknowledgments

This work was performed under the auspices of the U S Department of Energy by Lawrence Livermore National Laboratory under contract No W-7405-ENG-48

A SPAM Derivation

Suppose the variable A represents a material's generic property, such as density or temperature At an interpolation point j, this property has a value denoted by A_j The hydrodynamic value of $A(\mathbf{r})$ at any point $\mathbf{r}$ in space is calculated from a smoothed average of the values A_j corresponding to interpolation points j in the local area The smoothed-averaging process is accomplished by a weighting function, $W(\mathbf{r},h)$ The parameter h characterizes the "width" of influence of W This weighting function has two key properties [13]:

$$1 = \int_V W(\mathbf{r},h)d\mathbf{r} \tag{9}$$

and

$$\lim_{h\to 0} W(\mathbf{r},h) = \delta(\mathbf{r}), \tag{10}$$

where $\delta(\mathbf{r})$ is the delta function

The property $A(\mathbf{r})$ at any point $\mathbf{r}$, is defined by

$$A(\mathbf{r}) = \int_V A(\mathbf{r}')\delta(\mathbf{r}-\mathbf{r}')d\mathbf{r}' \tag{11}$$

If we use the weighting function $W(\mathbf{r},h)$ as an approximation to $\delta(\mathbf{r})$ and we then discretize the continuum into small "chunks" of mass, we obtain the following SPAM definition:

$$A(\mathbf{r}) \equiv \sum_{j=1}^{n} A_j \frac{m_j}{\rho_j} W(\mathbf{r}-\mathbf{r}_j,h) \tag{12}$$

A.1 Spatial Derivatives

The SPAM technique offers a simple and elegant method for calculating spatial derivatives To find the derivative along the x direction, we differentiate (12) and obtain

$$\frac{\partial A(\mathbf{r})}{\partial x} = \sum_j \frac{\partial}{\partial x}\left(A_j \frac{m_j}{\rho_j} W(\mathbf{r}-\mathbf{r}_j,h)\right) \tag{13}$$

Because $W(\mathbf{r},h)$ is the only term that depends on $\mathbf{r}$ we get the following SPAM gradient definition:

$$\nabla A(\mathbf{r}) \equiv \sum_j A_j \frac{m_j}{\rho_j} \nabla W(\mathbf{r}-\mathbf{r}_j,h) \tag{14}$$

Often it is desirable to write formulations in symmetric forms This is easily done in the SPAM framework First, we expand the gradient $\nabla(\rho A)$ using the chain rule, and rearrange to obtain

$$\rho \nabla A = \nabla(\rho A) - A\nabla\rho \tag{15}$$

The expansion of (15) in SPAM notation then produces Hoover's [8] alternate form of the gradient evaluated at a particle location i,

$$\nabla A_i = \sum_j m_j \frac{(A_j - A_i)}{\rho_{ij}} \nabla W_{ij} \tag{16}$$

where W_{ij} is shorthand notation for $W(\mathbf{r}_i - \mathbf{r}_j, h)$ and the term ρ_{ij} is either the arithmetic or geometric mean of the densities of particles i and j

The density calculation in SPAM can be performed in primarily two ways The first is to use the SPAM definition of density with ρ substituted for A in (12) This definition leads to the following formulation for density for particle i:

$$\rho_i = \sum_j m_j W_{ij} \tag{17}$$

Unfortunately, (17) can lead to problems in the calculation of free surface flows because it leads to lower-than-standard densities at the free surface This creates large pressure gradients at the surface causing particles to "boil" away due to a lack of a restoring atmospheric pressure at the free surface This can be remedied with the addition of a surface-tension force to keep particles from escaping Alternatively, one can assign each particle an initial density ρ_o and then integrate the continuity equation (18) like any other field variable with formulation 1 This was the approach used in this work

B The Equations of Motion

The general field equations is a set of partial differential equations that describe the conservation laws for mass, momentum, and energy From that set, the Navier-Stokes equations for fluid mechanics are derived by making the following assumptions about the fluid: it satisfies the Fourier Law of Heat Conduction; it is Newtonian; and it satisfies the Stokes relation

In computer graphics, and certainly in modeling unsteady flows, it is convenient to follow the motion of particles in a continuum Therefore it is only natural that we take a *Lagrangian* view of our system In the Lagrangian framework, the compressible Navier-Stokes equations are

$$\frac{1}{\rho}\frac{D\rho}{Dt} = -\nabla\ \mathbf{u} \tag{18}$$

$$\rho\frac{D\mathbf{u}}{Dt} = \nabla\ \sigma + \rho\mathbf{g} \tag{19}$$

$$\rho\frac{De}{Dt} = \sigma : \nabla\mathbf{u} - \nabla\ \mathbf{Q} \tag{20}$$

where ρ is density, $\mathbf{u}$ is velocity, $\mathbf{g}$ is the external gravity field, and e is internal energy

The stress tensor σ is defined by

$$\sigma_{ij} = -(P + \frac{2}{n}\mu\nabla\ \mathbf{u})\delta_{ij} + 2\mu\varepsilon_{ij} \tag{21}$$

where P is pressure, μ is the viscosity coefficient, n is spatial dimension, and ε_{ij} is the *strain rate* tensor which has the form $\varepsilon_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)$

The heat flux vector $\mathbf{Q}$ is given by

$$\mathbf{Q} \equiv -\kappa\nabla T, \tag{22}$$

where κ is the coefficient of heat conduction in Fourier's Law Temperature is a function of internal energy, $T = e/c_v$ for an ideal gas where c_v is the specific heat at constant volume

The $A : B$ (double dot-product) notation stands for the double contraction of tensors to produce a scalar term:

$$A : B = \sum_i \sum_j A_{ij} B_{ji} \tag{23}$$

References

[1] James F Blinn Simulation of wrinkled surfaces In *Computer Graphics (SIGGRAPH '78 Proceedings)*, volume 12, pages 286–292, August 1978

[2] James F Blinn A generalization of algebraic surface *ACM Transactions on Graphics*, 1(3):235–256, July 1982

[3] Jim X Chen and Niels da Vitoria Lobo Toward interactive-rate simulation of fluids with moving obstacles using navier-stokes equations *Graphical Models and Image Processing*, 57(2):107–116, March 1995

[4] Norishige Chiba, Shinji Sanakanishi, Kenichiro Yokoyama, and Isao Ootawara Visual simulation of water currents using a particle-based behavioural model *The Journal of Visualization and Computer Animation*, 6 155–171, 1995

[5] H E Cline, W E Lorenson, S Ludke, and C R Crawford B C Teeter Two algorithms for the three-dimensional reconstruction of tomograms *Medical Physics*, 15(3) 320–327, May/June 1988

[6] Nick Foster and Dimitri Metaxas Realistic animation of liquids *Graphics Models and Image Processing*, 58(5) 471–483, September 1996

[7] Alain Fournier A simple model of ocean waves In *Computer Graphics*, volume 20, No 4, pages 75–84 ACM SIGGRAPH, August 1986

[8] W G Hoover, T G Pierce, C G Hoover, J O Shugart, C M Stein, and A L Edwards Molecular dynamics, smoothed-particle applied mechanics, and irreversibility *Computers Math Applic*, 28(10-12) 155–174, 1994

[9] Michael Kass and Gavin Miller Rapid, stable fluid dynamics for computer graphics *Computer Graphics*, 24(4):49–55, August 1990

[10] L B Lucy A numerical approach to the testing of the fission hypothesis *The Astronomical Journal*, 82(12) 1013–1024, December 1977

[11] Nelson L Max Vectorized procedural models for natural terrain: Waves and islands in the sunset *Computer Graphics*, ?(?) 317–324, August 1981

[12] Gavin Miller and Andrew Pearce Globular dynamics A connected particle system for animating viscous fluids *Comput and Graphics*, 13(3) 305–309, 1989

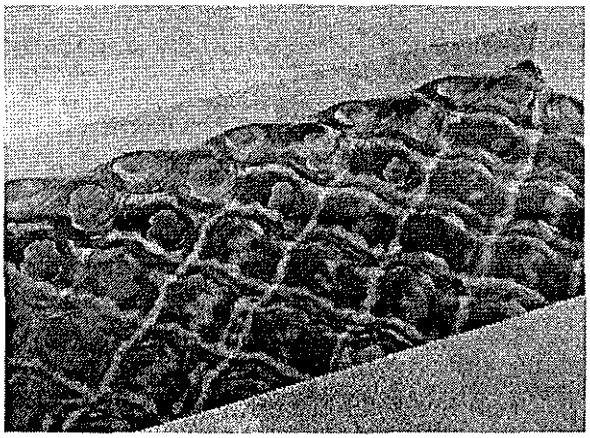


Figure 3: The combination of underwater light scattering and caustic effects in a swimming approximately six feet deep and filled with pure seawater

[13] J J Monaghan Smoothed particle hydrodynamics *Annu Rev. Astron Astrophys* , 30(2):543–574, 1992

[14] J J Monaghan Simulating free surface flows with sph *Journal of Computational Physics*, 110:399–406, 1994

[15] Claudio Montani, Riccardo Scateni, and Roberto Scopigno A modified lookup-up table for implicit disambiguation of Marching Cubes *The Visual Computer*, 10(6):353–355, 1994

[16] H Nishimura, M Hirai, T Kawai, T Kawata, I Shirakawa, and K Omura Object modeling by distribution function and a method of image generation *Trans IECE Japan, Part D*, J68-D(4):718–725, 1985

[17] Tomoyuki Nishita and Eihachiro Nakamae Method of displaying optical effects within water using accumulation buffer In *Proceedings of SIGGRAPH 94*, pages 373–379 ACM SIGGRAPH, July 1994

[18] Darwyn R Peachey Modelling waves and surf In *Computer Graphics*, volume 20, No 4, pages 65–74 ACM SIGGRAPH, August 1986

[19] W T Reeves Particle systems – a technique for modeling a class of fuzzy objects *ACM Trans Graphics*, 2 91–108, April 1983

[20] Mark Watt Light-water interaction using backward beam tracing In *Proceedings of SIGGRAPH 90*, pages 377–385 ACM SIGGRAPH, August 1990

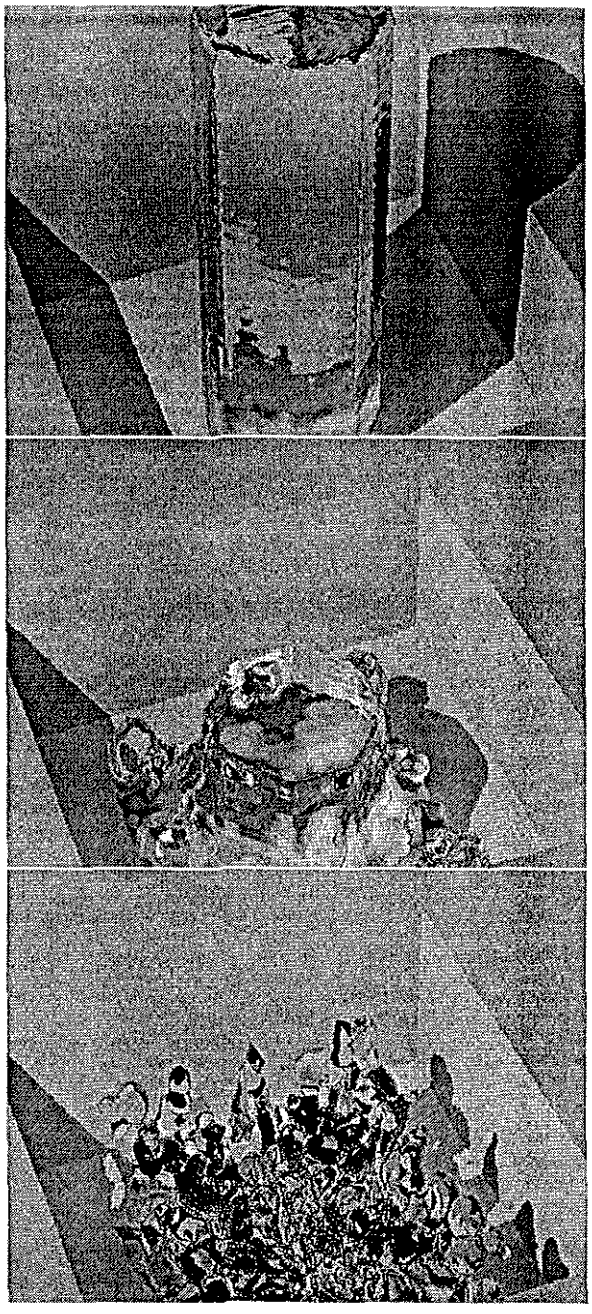


Figure 4 Three frames taken from an animation of a column of water falling into a box

Technical Information Department • **Lawrence Livermore National Laboratory**
University of California • Livermore, California 94551