Proceedings of the 1991 Winter Simulation Conference
Barry L. Nelson, W. David Kelton, Gordon M. Clark (eds.)

HEIGHT-FIELD FLUIDS FOR COMPUTER GRAPHICS

Michael Kass

Advanced Technology Group
Apple Computer, Inc.
20705 Valley Green Drive
Cupertino, CA 95014

ABSTRACT

We present a new method for animating water based on a simple, rapid and stable solution of a set of partial differential equations resulting from an approximation to the shallow water equations. The approximation gives rise to a version of the wave equation on a height-field where the wave velocity is proportional to the square root of the depth of the water. The resulting wave equation is then solved with an alternating-direction implicit method on a uniform finite-difference grid. The computational work required for an iteration consists mainly of solving a simple tridiagonal linear system for each row and column of the height field. A single iteration per frame suffices in most cases for convincing animation.

Like previous computer-graphics models of wave motion, the new method can generate the effects of wave refraction with depth. Unlike previous models, it also handles wave reflections, net transport of water and boundary conditions with changing topology. As a consequence, the model is suitable for animating phenomena such as flowing rivers, raindrops hitting surfaces and waves in a fish tank as well as the classic phenomenon of waves lapping on a beach. The height-field representation prevents it from easily simulating phenomena such as breaking waves, except perhaps in combination with particle-based fluid models.

1 INTRODUCTION

The problem of realistically modeling scenes containing water has captured the attention of a number of computer-graphics researchers in recent years [Max 1981; Peachy 1986; [Fournier and Reeves 1986; Ts'o and Barsky 1987; Masten et al. 1987]. The omni-presence of water as well as the complexities and subtleties of its motion have made it an attractive subject of study. Yet existing computer-graphics models of water motion adequately cover only a very small range of interesting water phenomena. Among other effects, they fail to account for wave reflections, net transport of water and boundary conditions with changing topology. A computationally inexpensive method of simulating these phenomena will be presented here. Based on solving a partial-differential equation on the surface of a height-field, the method is easy to implement and very stable. The approximations involved may not be suitable for high-precision engineering applications, but they produce pleasing animation with little effort.

Many popular methods for modeling water surfaces work well for producing still images, but are unsuitable for animation because they do not include realistic models for the evolution of the surface over time. Examples of these techniques include stochastic subdivision [Lewis 1987] and Fourier synthesis [Masten et al. 1987]. Other techniques work well only in large bodies of water away from boundaries [Perlin 1985; Max 1981; Schachter 1980]. Recently, the realism of water modeling in computer graphics was substantially improved by three papers [Peachy 1986; Fournier and Reeves 1986; [Ts'o and Barsky 1987] that took into account refraction due to changing wave velocity with depth. In each case, specialized methods based on tracking individual waves or wave-trains were used to avoid the need to directly solve a differential equation. These papers deal adequately with waves hitting a beach, but they leave a wide range of water phenomena unexplored. None of the papers includes simulations of reflected waves. In addition, the underlying model in each case is that particles of water move in circular or ellipsoidal orbits around their initial positions, so there can be no net transport or flow. Finally, none of the papers considers situations in which the boundary conditions change through time altering the topology of the water -- for example a wave pushing water up over an obstacle and down the other side to create a puddle. It appears to be very difficult to deal with these phenomena efficiently by tracing waves.

Two alternatives to tracing the propagation of waves or wave-trains exist. One is to simulate the fluid by the interaction of a large number of particles [Miller and Pearce 1989; [Sims 1988], and the other is to directly solve a partial differential equation describing the fluid dynamics [Patel&Dvinsky 1987; Kallinderis&Baron 1989; Miyata&Nishimura 1985]. Both have been used by hydro-dynamicists to create iterative simulations of fluid flow. The problem is that a truly accurate simulation of fluid mechanics usually requires computing the motion throughout a volume. This means that the amount of computation per iteration grows at least as the cube of the resolution. If there are linear systems to be solved at every iteration, the computational cost can grow even faster. In addition, the number of iterations required may grow as the resolution is increased. As a consequence, accurate simulation of fluid mechanics is typically reserved for vectorized supercomputers or very highly parallel machines.

For the purposes of animation, accuracy is much less important than stability and speed. An animator using techniques of physical simulation will typically have to experiment with a number of different conditions of a simulation before achieving satisfying motion. If the experiments take too much time or if the numerical methods become unstable, the process can become excruciating.

Here, we examine the differential equation approach with the goal of constructing the fastest stable simulation which yields a wide range of convincing motion. We begin by considering a very simplified subset of water flow where the water surface can be represented as a height field and the motion is uniform through a vertical column. This subset of water flow is representative of a variety of non-turbulent shallow-water phenomena. Under these conditions, we can approximate the equations of motion of the water in terms of a grid of points on a height-field. The amount of computation

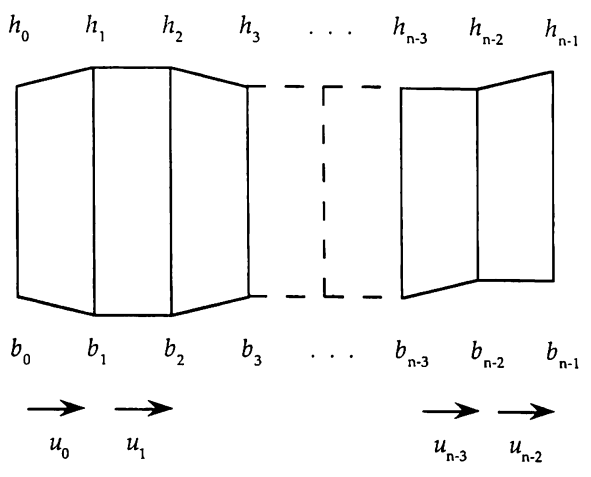

Fig. 1: Discrete two-dimensional height-field representation of the water surface h, the ground bottom b, and the horizontal water velocity u

can then be proportional to the number of samples on the surface of the water which varies as the square of the resolution instead of the cube.

Integration of the partial-differential equations is done with an alternating-direction implicit technique [Press et al. 1986]. The result is a very stable integration scheme which is also very fast. Stability derives from the use of an implicit integration scheme; speed derives from the tridiagonal structure of the required linear systems which are solvable in linear time. Because of the stability, the time-step of the solution can be made equal to the frame time of the animation in most cases.

2 SHALLOW WATER EQUATIONS

In lieu of simulating the full Navier-Stokes equations of fluid flow, we begin with a vastly simplified set of equations which has been widely used for shallow water [Le Mehaute 1976; [Crapper 1984; Stoker 1957]. The simplification arises from three approximations. The first approximation is that the water surface is a height field. This, of course, has some obvious limitations. The water cannot splash and waves cannot break. However, so long as the forces on the water are sufficiently gentle, the height-field assumption will not introduce error. The second assumption is that the vertical component of the velocity of the water particles can be ignored. Once again, the limitations of this assumption are fairly clear. If a disturbance creates very steep waves on the water surface, the model will cease to be accurate. The third assumption is that the horizontal component of the velocity of the water in a vertical column is approximately constant. If there is turbulent flow or unusually high friction on the bottom, this assumption will break down. Nonetheless, the experience of hydrodynamicists suggests that this is a very useful approximation to phenomena ranging from the effect of a single rain drop to the refraction of waves in a sea port.

For simplicity, we begin with a height-field curve in two dimensions. Later, the same techniques will be extended to a height-field surface in three dimensions. Let $z = h(x)$ be the height of the water surface and let $z = b(x)$ be the height of the ground. If $d(x)=h(x)-b(x)$ is the water depth and $u(x)$ is the horizontal velocity of a vertical column of water, the shallow water equations that follow from the above assumptions[Crapper 1984; Stoker 1957] can be written as follows:

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + g\frac{\partial h}{\partial x} = 0 \qquad (1)$$

$$\frac{\partial d}{\partial t} + \frac{\partial}{\partial x}(ud) = 0 \qquad (2)$$

where g is the gravitational acceleration. Eq. 1 expresses Newton's law $F=ma$ while eq. 2 expresses the constraint of volume conservation. Note that even with the above three simplifying assumptions, the resulting differential equations are non-linear. A further simplification which is often used is to ignore the second term in eq. 1 and linearize around a constant value of h. This will be reasonable if the fluid velocity is small and the depth is slowly varying. The resulting equations are then:

$$\frac{\partial u}{\partial t} + g\frac{\partial h}{\partial x} = 0 \qquad (3)$$

$$\frac{\partial h}{\partial t} + d\frac{\partial u}{\partial x} = 0 \qquad (4)$$

If we differentiate eq. 3 with respect to x, then differentiate eq. 4 with respect to t and finally substitute for the cross-derivatives, we end up with

$$\frac{\partial^2 h}{\partial t^2} = gd\frac{\partial^2 h}{\partial x^2} \qquad (5)$$

which is the one-dimensional wave equation with wave velocity $\sqrt{gd}$. While this degree of simplification is suspect for many engineering purposes, our experience suggests that the resulting equations are quite adequate for a wide range of animation applications.

3 DISCRETIZATION

In order to solve eq. 5, we need to construct a discrete representation of the continuous partial-differential equation. There are two established techniques for doing so. The first is the finite-difference technique where the continuous functions are represented by a collection of samples. The second is the finite-element technique where the continuous functions are represented as the sum of a collection of continuous basis functions. Here, the finite-difference technique works particularly well because of the simple height-field representation. The resulting algorithm is very easy to implement and the linear systems involved are tridiagonal.

Figure 1 shows the discrete representation of the height-field in two dimensions. Note that the samples for u lie halfway in between the samples of h. After experimenting with a number of finite-difference approximations to equations 3 and 4, the most stable version we have found is

$$\frac{\partial h_i}{\partial t} = \left(\frac{d_{i-1}+d_i}{2\Delta x}\right)u_{i-1} - \left(\frac{d_i+d_{i+1}}{2\Delta x}\right)u_i \qquad (6)$$

$$\frac{\partial u_i}{\partial t} = \frac{-g(h_{i+1}-h_i)}{\Delta x} \qquad (7)$$

where Δx is the separation of the samples along the x direction. Putting the above two equations together, we get

$$\frac{\partial^2 h_i}{\partial t^2} = -g\left(\frac{d_{i-1}+d_i}{2(\Delta x)^2}\right)(h_i - h_{i-1}) + g\left(\frac{d_i+d_{i+1}}{2(\Delta x)^2}\right)(h_{i+1} - h_i) \quad (8)$$

which is a discrete approximation to eq. 5.

4 INTEGRATION

The finite differences convert the partial-differential equation into an ordinary differential equation involving h and its time derivatives. The remaining problem is to solve the ordinary differential equation. While there are a number of possible choices of solution method, the wave equation is a notoriously bad example for explicit differential equation methods such as Euler's method or Runge-Kutta integration. As the wave velocity approaches one sample per iteration, explicit methods tend to diverge very rapidly. Since the wave speed is proportional to the square-root of the depth, an ordinary explicit method would have to use a time-step appropriate for the deepest water in the model. Implicit methods, on the other hand, do not suffer from these difficulties.

For simplicity, we use a first-order implicit method which appears to be perfectly adequate. Let $h(n)$ to denote h at the nth iteration and let dots denote differentiation with time. Then the first-order implicit equations can be written

$$\frac{h(n) - h(n-1)}{\Delta t} = \dot{h}(n) \quad (9)$$

$$\frac{\dot{h}(n) - \dot{h}(n-1)}{\Delta t} = \ddot{h}(n) \quad (10)$$

Note that the right-hand sides of these equations are evaluated at time n which corresponds to the end of the iteration rather than time n-1 which corresponds to the beginning of the iteration. This is what makes the iteration implicit and stable. Rearranging the above, we get

$$h(n) = h(n-1) + \Delta t\,\dot{h}(n-1) + (\Delta t)^2\ddot{h}(n) \quad (11)$$

$$h(n) = 2h(n-1) - h(n-2) + (\Delta t)^2\ddot{h}(n) \quad (12)$$

$$h_i(n) = 2h_i(n-1) - h_i(n-2) - g(\Delta t)^2\left(\frac{d_{i-1}+d_i}{2(\Delta x)^2}\right)(h_i(n) - h_{i-1}(n)) + g(\Delta t)^2\left(\frac{d_i+d_{i+1}}{2(\Delta x)^2}\right)(h_{i+1}(n) - h_i(n)) \quad (13)$$

We are still left with non-linear equations because d depends on h. In order to solve these equations rapidly, we need a final linearization. Once again there are several possible choices, but a particularly well-behaved linearization is to regard d as a constant during the iteration. This means that the wave velocity is fixed as a function of x. It limits the non-linearities to changing the wave velocities in-between iterations and virtually ensures that the iteration will not diverge. With this linearization the next value of h can be calculated from previous values with the symmetric tridiagonal linear system

$$\mathbf{A}h_i(n) = 2h_i(n-1) - h_i(n-2) \quad (14)$$

where the matrix A is given by

$$\mathbf{A} = \begin{pmatrix} e_0 & f_0 & & & & & \\ f_0 & e_1 & f_1 & & & & \\ & f_1 & e_2 & \ddots & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & \ddots & e_{n-3} & f_{n-3} & \\ & & & & f_{n-3} & e_{n-2} & f_{n-2} \\ & & & & & f_{n-2} & e_{n-1} \end{pmatrix} \quad (15)$$

and the elements of A are as follows:

$$e_0 = 1 + g(\Delta t)^2\left(\frac{d_0+d_1}{2(\Delta x)^2}\right)$$

$$e_i = 1 + g(\Delta t)^2\left(\frac{d_{i-1}+2d_i+d_{i+1}}{2(\Delta x)^2}\right) \quad (0 < i < n-1)$$

$$e_{n-1} = 1 + g(\Delta t)^2\left(\frac{d_{n-2}+d_{n-1}}{2(\Delta x)^2}\right)$$

$$f_i = -g(\Delta t)^2\left(\frac{d_i+d_{i+1}}{2(\Delta x)^2}\right) \quad (16)$$

Note that right-hand side of eq. 14 can be regarded as an extrapolation of the previous motion of the fluid surface. Some interesting effects are possible by slightly changing the extrapolation. In particular, if the equation is changed to be

$$\mathbf{A}h_i(n) = h_i(n-1) + (1-\tau)(h_i(n-1) - h_i(n-2)) \quad (17)$$

then τ introduces some damping in the extrapolation. If $\tau = 0$, then it reduces to eq. 14, but if τ is between zero and one, it will make the waves damp out over time. The visual effect is that of a viscous fluid.

There is one further subtlety of importance in the two-dimensional case. Even though eq. 14 was derived from eq. 6 which specifies conservation of volume, there is no guarantee that the results of the iteration will precisely conserve volume. The primary cause of departures from volume-conserving behavior is that the iteration may leave $h_i < b_i$ for some index i. To compensate for this negative volume, the iteration will create excess positive volume elsewhere. While the effect is small, it can accumulate over time and create substantial drift. If the entire surface acquires a small net upwards velocity it will very quickly create noticeable amounts of water. To combat this effect, the following simple projection appears to be adequate. After each iteration, find the connected pieces of the fluid. This can be done by scanning the h and b vectors in order and testing whether $h_i < b_i$. For each connected piece

of the fluid, calculate the old volume and the new volume. If the new volume is different, distribute the difference uniformly over the samples in the connected region.

We can now state the entire algorithm for the two-dimensional case in some detail:

Begin by specifying $h(0)$, $h(1)$ and b.
Loop for j starting at 2 incrementing by one.
  If there are net sources or sinks of water, add or subtract the amounts from the current and last values of h.
  Calculate d from $h(j-1)$ and b. If $h_i < b_i$ then $d_i = 0$.
  Calculate the new value of h from $h(j-1)$ and $h(j-2)$ using eq. 14.
  Adjust the new value of h to conserve volume as above.
  If $h_i < b_i$ for some index i, set $h_i(j)$ and $h_i(j-1)$ to $b_i - \varepsilon$.
  The resulting value of h is $h(j)$.

While there are a number of possible refinements, this is the basic version of the two-dimensional case. It can be implemented in one to two pages of very efficient, straightforward C code.

5 THREE DIMENSIONS

Height fields in two dimensions are interesting, but moving to three dimensions opens up a much wider range of possibilities. Fortunately, the three-dimensional equations can be approximated by a series of two-dimensional equations, so the complexity does not increase radically. The basic wave equation for water in three dimensions is the same as the two-dimensional case except that the second derivative of h with respect to x is replaced with the Laplacian.

$$\frac{\partial^2 h}{\partial t^2} = gd\left(\frac{\partial^2 h}{\partial x^2} + \frac{\partial^2 h}{\partial y^2}\right) = gd\nabla^2 h \qquad (18)$$

In order to solve the equations in three dimensions, we rely on the alternating-direction method[Press et al. 1986]. The basic idea of the method is to take eq. 18 and split the right-hand side of it into the sum of two terms, one of which is independent of y and the other of which is independent of x.

We then divide the iteration into two sub-iterations. In the first sub-iteration, we replace the right-hand side of eq. 18 with the first term, and in the second sub-iteration, we replace the right-hand side of eq. 18 with the second term. More specifically, in the first sub-iteration, we solve the equation

$$\frac{\partial^2 h}{\partial t^2} = gd\frac{\partial^2 h}{\partial x^2} \qquad (19)$$

and in the second sub-iteration, we solve the equation

$$\frac{\partial^2 h}{\partial t^2} = gd\frac{\partial^2 h}{\partial y^2}. \qquad (20)$$

The advantage of this technique is that the required linear systems remain tridiagonal so the computational cost per iterations is proportional to the number of samples on the surface. The resulting implementation remains very simple. For the first sub-iteration, we compute the update as before on each row of the height-field. For the second sub-iteration, we do the same for each column in the height-field. While artifacts can potentially arise from the favored directions, our experience with the alternating-direction formulation of these equations is very favorable. More so than in two-dimensions, it is important to be careful with the details of the volume conservation. Errors manifest themselves as line artifacts along the x or y axes.

CONCLUSION

There is a long history of people using differential equations to analyze and simulate fluid flow for engineering purposes. Here we have attempted to make use of that work to derive a simplified model that is well suited to the demands of animation. The model is stable, rapid and easy to program. The computation time is linear in the number of samples of the height field, making high-resolution simulations possible. In the three-dimensional case, the computation for each row and each column is independent, so it can be easily parallelized. Unlike models which rely on tracking individual waves or wave-trains, reflected waves, changing boundaries and net flow can be handled in a simple manner. As a consequence, this model extends the range of water effects which can be animated in a reasonable time.

By using a number of approximations it is possible to render convincing caustic shading effects at little computational cost. A wetness map adds to the realism of water flowing over sand. When combined with the fluid dynamics model, the results are encouragingly realistic.

ACKNOWLEDGEMENTS

Thanks to Gavin Miller for stimulating discussions and help in rendering images of the fluids. Thanks to the ATG graphics group for discussions and helpful advice.

REFERENCES

Crapper 1984 Crapper, G. 1984. *Introduction to Water Waves*, John Wiley & Sons, New York.

Fournier and Reeves 1986 Fournier, A. and W. Reeves. 1986. "A Simple Model of Ocean Waves," Proceedings of SIGGRAPH 86: 75-84.

Kallinderis&Baron 1989 Kallinderis, Y. and J. Baron. 1989. "Adaptation methods for a new Navier-Stokes algorithm," AIAA Journal, 27, 1: 37-43.

Le Mehaute 1976 Le Mehaute, B. 1976. *An Introduction to Hydrodynamics and Water Waves*, Springer-Verlag, New York.

Lewis 1987 Lewis, J. 1987. "Generalized Stochastic Subdivision," *ACM Transactions on Graphics*, 6, 3: 167-190.

Masten et al. 1987 Masten, G., P. Watterberg, and I. Mareda. 1987. "Fourier Synthesis of Ocean Scenes," *IEEE Computer Graphics and Application*, 7, 3: 16-23.

Masten et al. 1987 Masten, G., P. Watterberg, and I. Mareda. 1987. "Fourier Synthesis of Ocean Scenes," *IEEE Computer Graphics and Application*, 7, 3: 16-23.

Max 1981 Max, N. 1981. "Vectorized procedural models for natural terrain: Waves and islands in the sunset," Proceedings of SIGGRAPH 81: 317-324.

Miller and Pearce 1989 Miller, G. and A. Pearce. 1989. "Globular Dynamics: A connected particle system for

animating viscous fluids," *Computer Graphics* 13,3: 305-309.

Miyata&Nishimura 1985 Miyata, H. and S. Nishimura. 1985. "Finite difference simulation of nonlinear waves generated by ships of arbitrary three-dimensional configuration," Journal of Computational Physics 60: 391-436.

Patel&Dvinsky 1987 Patel, B. and A. Dvinsky. 1987 "The solution of the reynolds averaged Navier-Stokes equations in general curvilinear coordinates and its application to vehicular aerodynamics," in *Computers in Design, Manufacture and Operation of Automobile*s, Murthy and Brebbia, Eds., Springer Verlag, Berlin.

Peachy 1986 Peachy, D. 1986. "Modeling Waves and Surf," Proceedings of SIGGRAPH 86: 65-74.

Perlin 1985 Perlin, K. 1985. "An Image Synthesizer," Proceedings of SIGGRAPH 85: 287-296.

Press et al. 1986 Press, W., B. Flannery, S. Teukolsky, and W. Vetterling. 1986. Numerical Recipes: The Art of Scientific Computing, Cambridge University Press, Cambridge.

Schachter 1980 Schachter, B. 1980. "Long crested wave models," *Computer Graphics and Image Processing* 12: 187-201.

Sims 1988 Sims, C. 1988. "Particle Dreams,"[Video] *SIGGRAPH Video Review* 38/39, ACM SIGGRAPH, New York, segment 42.

Stoker 1957 Stoker, J. 1957. *Water Waves*, Interscience, New York.

Ts'o and Barsky 1987 Ts'o, P. and B. Barsky. 1987. "Modeling and Rendering Waves," *ACM Transactions on Graphics*, 6, 3: 191-214.

AUTHOR BIOGRAPHY

MICHAEL KASS is a staff research scientist with the Advanced Technology Group of Apple Computer. He received a B.A. in Artificial Intelligence from Princeton University, an M.S. in Computer Science from M.I.T., and a P.h.D. in Electrical Engineering from Stanford University. His research focus is on the use of physical simulation for computer graphics.