Fast Water Animation Using the Wave Equation with Damping

Y. Nishidate and G. P. Nikishkov

University of Aizu, Aizu-Wakamatsu 965-8580, Japan
niki@u-aizu.ac.jp
http://www.u-aizu.ac.jp/~niki

Abstract. A simple method for animation of water waves is presented. The two-dimensional wave equation with damping is used to obtain a finite difference scheme for height distribution. A computational procedure employs explicit time integration. High frame rates are typically obtained for real-time animation of water waves.

1 Introduction

During the last two decades, researchers has devoted some attention to animation of water movement under various conditions. Detailed reviews of techniques for realistic water modeling and rendering have been presented by Iglesias [1] and Adabala and Manohar [2]. According to the reviews, approaches to dynamic modeling of fluids can be divided into visually convincing techniques and physically accurate techniques.

Visually convincing techniques usually use heuristic approaches to simulate fluid-like behavior, which creates 'nice' visual impression. These techniques can provide high frame rates for animation. However, they may require complicated tuning of empirical parameters.

Physically accurate techniques are mostly based on the numerical solution of some form of the Navier-Stokes equations [3, 4, 5, 6]. Three-dimensional problems have been solved by Foster and Metaxas [4]. Other authors [3, 5, 6] employ simplified two-dimensional Navier-Stokes differential equations. Since the solution of the Navier-Stokes equations is a computationally intensive task it usually makes real-time animation impossible. In computer graphics it is recognized that faster animation is more important than its physically-based realism. We devote our efforts to short turnaround time of water wave animation while still preserving simplified physical and mathematical foundations of an algorithm.

In this paper, we propose a method for animation of water waves, which is based on the two-dimensional wave equation with damping. It can be shown that the wave equation without damping is equivalent to the linearized shallow water equation. This justifies the use of the wave equation for modeling water waves. Using the wave equation with damping allows us to produce acceptable realistic picture of waves characterized by decreasing of wave height with time. Our dynamic model of water waves propagation utilizes explicit time integration

V.S. Sunderam et al. (Eds.): ICCS 2005, LNCS 3515, pp. 232–239, 2005.

of discrete analogue of the wave equation with damping. Initial conditions are specified in the form of height excitation at one or several places. Additional excitations can be introduced at any time during animation procedure. This helps to simulate different phenomena such as rain drops or boat movement on the water surface. A computer code for animation of water waves typically provides high frame rates.

2 Computational Model

2.1 Governing equation

The linearized shallow water equation [6] assumes that water speed varies slowly in space:

$$\frac{\partial u}{\partial t} + g\frac{\partial h}{\partial x} = 0, \quad \frac{\partial v}{\partial t} + g\frac{\partial h}{\partial y} = 0, \quad \frac{\partial h}{\partial t} + d\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right) = 0\,. \tag{1}$$

Here it is supposed that the water surface is parallel to xy coordinate plane; u, v are fluid velocities along $x-$ and $y-$ directions; h is the height of the water surface; d is the water depth and g is the gravitational acceleration.

If to differentiate first equation in respect to x, second equation in respect to y, third equation in respect to t and to combine all three equations, the following second order differential equation containing only the height h can be obtained:

$$\frac{\partial^2 h}{\partial t^2} = gd\left(\frac{\partial^2 h}{\partial x^2} + \frac{\partial^2 h}{\partial y^2}\right)\,. \tag{2}$$

It can be easily seen that for $d = const$ the above shallow water equation coincides with the two-dimensional wave equation without damping. From everyday experience we understand that the wave equation without damping is not sufficient for realistic water movement modeling. For example, wakes that spread behind a boat as it moves forward are decreased with time. This phenomenon can be modeled if to utilize the wave equation with damping, which can be written down in the following from:

$$\frac{\partial^2 h}{\partial t^2} - k\frac{\partial h}{\partial t} = c^2\left(\frac{\partial^2 h}{\partial x^2} + \frac{\partial^2 h}{\partial y^2}\right)\,, \tag{3}$$

where c is the wave speed and k is the damping constant.

2.2 Finite difference procedure

Finite Difference Equation. For water surface animation, let us introduce a rectangular grid of nodal points for controlling the water height h. Each nodal point is characterized by two subscripts as shown in Fig. 1: i, which counts grid points along x and j, which counts grid points along y. Superscripts are used to relate the quantity to time moment. The water height h at location x_i, y_j at time moment t is denoted as $h^t_{i,j}$.

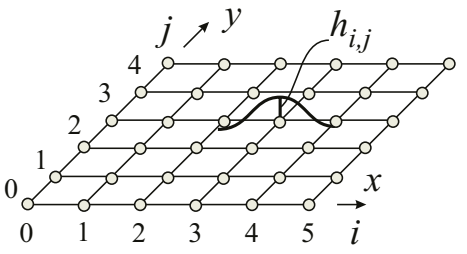


Fig. 1. Finite-difference grid for modeling wave movement

The finite difference approximation produces the following discrete analogue of Eq. (3) for equal point steps along coordinate axes $\Delta x = \Delta y$:

$$\begin{aligned} &\frac{h_{i,j}^{t+\Delta t} - 2h_{i,j}^{t} + h_{i,j}^{t-\Delta t}}{\Delta t^2} + k\frac{h_{i,j}^{t} - h_{i,j}^{t-\Delta t}}{\Delta t} \\ &= c^2 \frac{4h_{i,j}^{t} - h_{i+1,j}^{t} - h_{i-1,j}^{t} - h_{i,j+1}^{t} - h_{i,j-1}^{t}}{\Delta x^2} \, . \end{aligned} \tag{4}$$

After regrouping terms we arrive at the following expression for time integration of the water height:

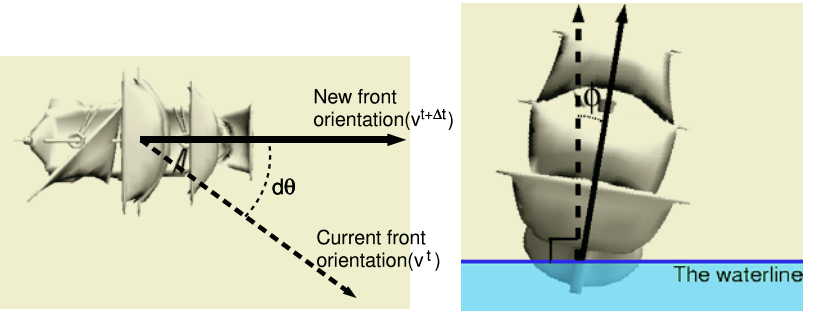
$$\begin{aligned} &h_{i,j}^{t+\Delta t} = h_{i,j}^{t} + (1 - k\Delta t)\left(h_{i,j}^{t} - h_{i,j}^{t-\Delta t}\right) \\ &+\frac{\Delta t^2 c^2}{\Delta x^2}\left(4h_{i,j}^{t} - h_{i+1,j}^{t} - h_{i-1,j}^{t} - h_{i,j+1}^{t} - h_{i,j-1}^{t}\right) \, . \end{aligned} \tag{5}$$

Influence of neighbors at time t in Eq. (5) is estimated with the use of the water height at four neighboring points: left, right, top and bottom. It is possible to take into account all eight neighboring points for the height computation at point i, j. Then the modified expression for time integration becomes:

$$\begin{aligned} &h_{i,j}^{t+\Delta t} = h_{i,j}^{t} + (1 - k\Delta t)\left(h_{i,j}^{t} - h_{i,j}^{t-\Delta t}\right) \\ &+\frac{\Delta t^2 c^2}{\Delta x^2}[4h_{i,j}^{t} - h_{i+1,j}^{t} - h_{i-1,j}^{t} - h_{i,j+1}^{t} - h_{i,j-1}^{t} \\ &+\frac{1}{2}\left(4h_{i,j}^{t} - h_{i+1,j+1}^{t} - h_{i+1,j-1}^{t} - h_{i-1,j+1}^{t} - h_{i-1,j-1}^{t}\right)] \, . \end{aligned} \tag{6}$$

Numerical experiments demonstrate that wave shapes look visually better when eight neighbors are taken into account. Thus, the Eq.(6) is preferable for water animation.

Integration, Initial Conditions and Boundary Conditions. An explicit scheme is used here for for time integration of the water profile. It has the obvious advantage of simplicity. A possible drawback of explicit integration is its instability for large integration time steps. The Courant-Friedrichs-Levy (CFL) stability condition requires that no wave travels farther than one cell in one time step. Our experience shows that for water animation problems the integration (5) or (6) is stable for real-time animation with reasonable values of the wave speed c and the damping constant k.



(a) Boat view from above (b) Boat view from front

Fig. 2. Boat orientation: the direction angle θ, increment of the direction angle $d\theta$ and the tilt angle ϕ

In order to compute the height values at time $t+\Delta t$ we need the height values for two previous time moments t and $t-\Delta t$. The initial conditions consist of: 1) setting $h^0_{i,j} = 0$ at time $t_0 = 0$; 2) setting of initial excitations $h^1_{i,j} = e_{i,j}$ at time $t_1 = \Delta t$. Nonzero height values at time t_1 can be specified at any number of grid points. Minimum initial excitation at time t_1 can include just one nonzero height value. After specification of two initial fields $h_{i,j}$, integration of Eq. (6) produces the water height fields for any time in the future. If the damping constant k is nonzero then the wave amplitude is decreasing with time.

In addition to initial excitation, any height values can be changed at any time moment. Using random number generators in space and in time, it is possible to simulate rain making indentations at the water surface, which imitate rain drops. Surface profile correction allows one to simulate movement of a boat on the water surface. Reflections of water waves at the boundary of the water reservoir are modeled with the symmetry boundary conditions. If the grid point i, j is located on the boundary then symmetrical interior points are employed instead of lacking height values outside the grid in the time integration procedure.

2.3 Boat on the Water Surface

Presented here the water wave animation procedure may be suitable for game development. Let us consider an application example where the boat follows the mouse position in a manner similar to magnetic force interaction. Boat movement is controlled by an acceleration vector $\boldsymbol{a}$ and by tilt angle ϕ as shown in Fig. 2. Components of the acceleration vector are calculated using location of the boat (x_b, y_b) and location of the mouse (x_m, y_m):

$$a_x = \frac{x_m - x_b}{\sqrt{(x_m - x_b)^2 + (y_m - y_b)^2}}, \quad a_y = \frac{y_m - y_b}{\sqrt{(x_m - x_b)^2 + (y_m - y_b)^2}}. \tag{7}$$

The above relations imitate magnetic force interaction. The acceleration vector is used to calculate the velocity vector v^t of the boat and new boat position.

The angle between current boat orientation and the acceleration vector $d\theta$ can be calculated as follows:

$$d\theta = \arccos\left(\left(\frac{v^t}{|v^t|}\right) \bullet \left(\frac{v^{t+\Delta t}}{|v^{t+\Delta t}|}\right)\right) . \tag{8}$$

The initial boat orientation is given by the programmer or by the user. The direction angle θ defining boat orientation and the tilt angle ϕ are calculated as:

$$\theta^{t+\Delta t} = \theta^t + d\theta, \quad \phi = \kappa\, d\theta , \tag{9}$$

where κ is an appropriate constant in order to make boat movement natural. After moving the boat to new position, negative increment is applied to the water height value at the previous boat position. This imitates wakes that spread behind the boat.

3 Implementation

The computational procedure for animation of water waves can be implemented in the following steps:

1. Define a grid for water height computation.
2. Apply initial deflections to some grid points.
3. Compute new water height values using wave profiles at two previous time moments using Eq. (6).
4. Render the water surface.
5. If necessary apply height excitations at grid points (rain drops, boat movement *etc.*)
6. Go to step 3.

To produce realistic water surface images with reflection, surface texture coordinates are generated following the sphere mapping algorithm [7]. The algorithm provides a reflection vector and texture coordinates that produce suitable reflections on the water surface.

The wave animation algorithm has been programmed in C++ language using OpenGL library for rendering. Desktop computer with Intel Pentium 4 2.4GHz processor, 512MB RAM and NVIDIA GeForce4 graphics card was used to perform real time animations. The algorithm was tested on square grids of different sizes, with different number of initial deflections and with different values of the model parameters.

4 Examples

In this section, several examples of application of the proposed water wave animation algorithm to various water movements are presented. In the first example, the surface deflection is applied to water surface at three points. The surface

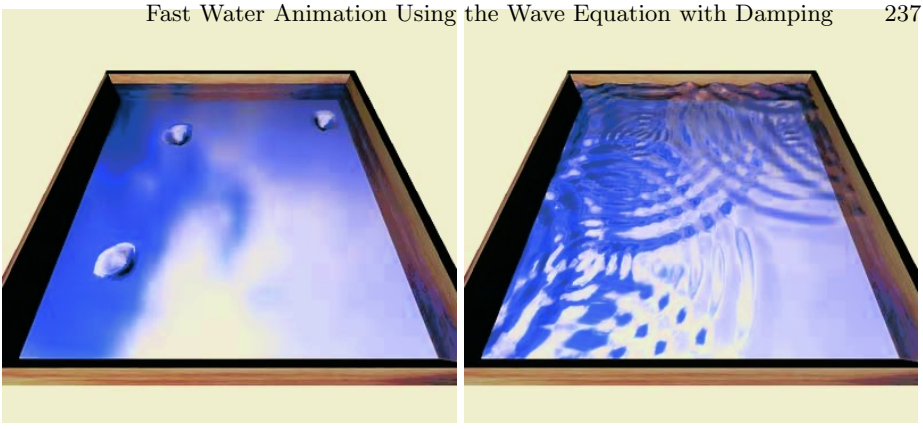


Fig. 3. Wave simulations with three initial deflections

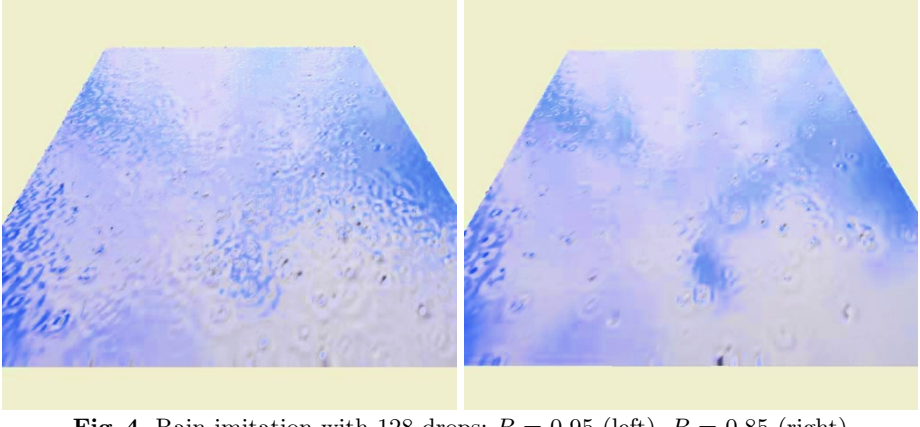


Fig. 4. Rain imitation with 128 drops: $R = 0.95$ (left), $R = 0.85$ (right)

profile after reflections of waves from the walls is shown in Fig. 3. The finite difference grid 64×64 is used. The frame rate is about 230 fps (frames per second). The value of the damping coefficient is set to zero in order to demonstrate effects of reflections and interference.

The example of Fig. 4 illustrates simulation of rain drops. The surface deflections are applied randomly at 128 grid points imitating rain drops. The surface is divided into the 128×128 grid. A coefficient $R = (1 - k\Delta t)$ is set to 0.95 and to 0.85 for the left and right pictures of Fig. 4 respectively. The code runs with the frame rate about 100 fps.

Fig. 5 shows simulation of wave propagation inside a concave-shaped reservoir. The Π-shaped water surface is produced by cutting off the part of the rectangular mesh. The surface is represented by the 64×64 grid. Four pictures demonstrate the process of spreading waves with reflections from the walls and interference. The frame rate is about 250.

Fig. 6 presents two screen shots of boat movement on the water surface. The coefficient $R = 0.95$ is employed. Fractal mountains are used for a coast line.

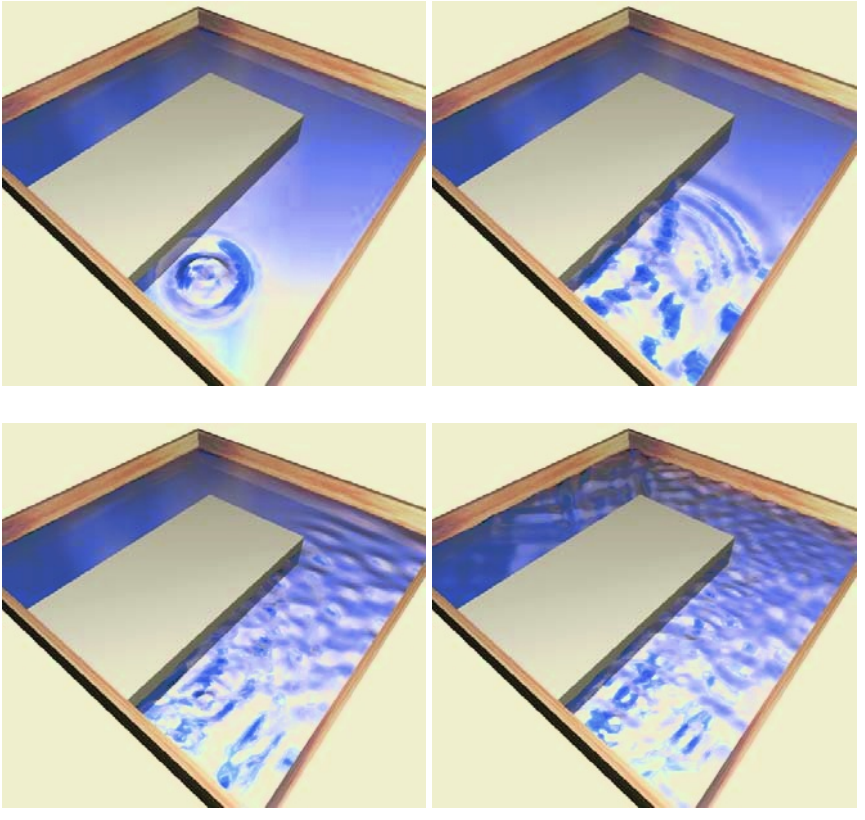


Fig. 5. Wave simulation applied to concave boundary surface

Boat trajectories and interference with some additional excitations can be seen in the pictures.

The reported frame rates are given for the code which performs both finite difference equation integration and water surface rendering for each frame. The algorithm efficiency can be increased if to perform rendering just with necessary for human eye frequency.

5 Conclusion

A simple and fast method for animation of water waves has been developed. Using the two-dimensional wave equation with damping, a finite difference relation for a water surface profile was obtained. An explicit scheme was used for time integration. One empirical coefficient was employed in order to simulate decrease of wave amplitudes with time.

Our experiments with the proposed water wave model show that it provides visually realistic animations of water waves with high frame rates. For example,

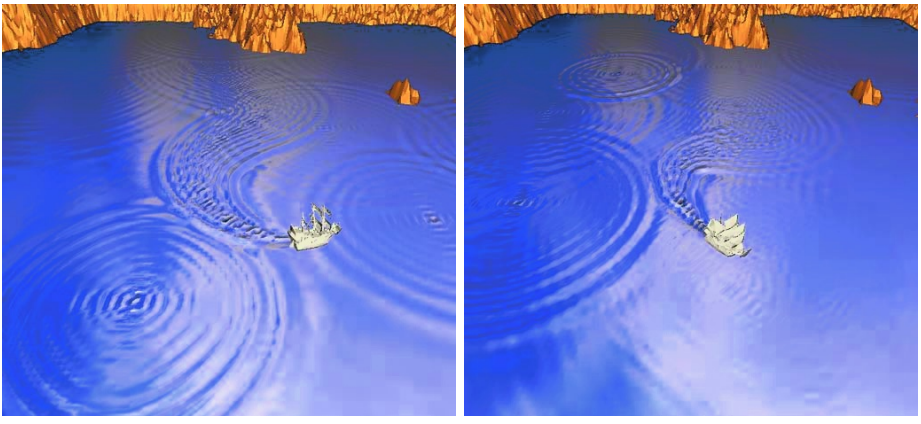


Fig. 6. Boat on the water surface

frame rate 100 fps was obtained for a mesh of 128 × 128 points using Intel Pentium 4 2.4GHz computer.

References

1. Iglesias, A.: Computer graphics for water modeling and rendering: a survey. Future Generation Computer Systems, **20** (2004) 1355-1374.
2. Adabala, L., Manohar, S.: Techniques for realistic visualization of fluids: a survey. Computer Graphics Forum **21** (2002) 65-81.
3. Kass, M., Miller, G.: Rapid, stable fluid dynamics for computer graphics. Computers and Graphics **24** (1990) 4955.
4. Foster, N., Metaxas, D.: Realistic animation of liquids. Graphical Models and Image Processing **58** (1996) 471483.
5. Chen, J.X., Lobo, N.V., Hughes, C.E., Moshell, J.M.: Real-time fluid simulation in a dynamic virtual environment. IEEE Computer Graphics and Applications. May-June 1997, 52-61.
6. Layton, A.T., van de Panne, M.: A numerically efficient and stable algorithm for animating water waves. The Visual Computer **18** (2002) 41-53.
7. Haeberli, P., Segal, M.: Texture mapping as a fundamental drawing primitive. Fourth Eurographics Workshop on Rendering, June 1993, 259-266.