Mode-Splitting for Highly Detailed, Interactive Liquid Simulation

H. Cords*
University of Rostock

Figure 1: *A real-time water environment simulated and rendered at 30 FPS. The water can be handled interactively by mouse dragging.*

Abstract

This work introduces a new technique for highly detailed, interactive liquid simulations. Similar to the mode-splitting method (used e.g. in oceanography), we separate the simulation of the low-frequency liquid flow and the high-frequency free surface waves. Hence, the performance for highly detailed liquid simulations can be increased immensely. Thereby, we use the 2D wave equation for surface simulation and the 3D Navier-Stokes equations for describing the liquid flow. Thus, the surface as well as the liquid flow are simulated physically-based, resulting in highly detailed and fully interactive liquid simulations: The liquid flows according to gravity, ground, obstacles and interactions, and the surface interacts with impacts, moving obstacles or simulates the propagation of highly detailed surface waves. Our method obtains realistic results at high framerates. Therefore, it is very suitable for today's video games, VR-Environments or medical simulators.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism

Keywords: water simulation, liquid simulation, real-time rendering, wave equation, smoothed particle hydrodynamics

1 Introduction

The complex and highly dynamic physical and visual behavior of liquids does not allow the real-time simulation of photo-realistic liquids with current desktop PCs yet. Real-time approaches for simulating liquids can be classified as follows:

- Empirical surface simulations,
- physically-based surface simulations (e.g. wave equation),
- physically-based volume simulations (Navier-Stokes equations).

Realistic animation results can actually only be reached with the physically-based approaches. Due to the high costs of full physically-based liquid simulations, those real-time approaches suffer from low details or simplified models.

This paper presents a method based on the idea of mode-splitting to increase the quality of the simulated liquid immensely: The intense low-frequency 3D flow simulation is separated from the high-frequency surface simulation. Thus, a fully interactive, highly detailed liquid simulation at high framerates can be achieved. Our approach refers to interactivity as detailed surface interactivity (e.g. moving obstacle, rain, surface wave generation) as well as to volume interactivity (e.g. moving the volume of liquid, shallow water wave generation). Such a high resolution liquid simulation in addition to full interactivity at high framerates cannot be reached with existing real-time methods.

Navier-Stokes based methods can simulate real liquid flows and the movement of the free surface (boundary layer) in 3D. However, they are limited to a low sampling of the liquid volume in real-time environments. In detail, Eulerian (grid based) approaches are limited to a low volume sampling and Largrangian methods (particle based methods) are limited to a few thousand particles. Thus, highly detailed phenomena e.g. small free surface waves, cannot be sampled in real-time. Additionally, due to the necessary large time-steps in real-time applications, the simulated liquid tends to exhibit high viscosity. But these methods allow a full user interaction with the whole liquid volume (e.g. a virtual glass of water can be handled interactively like a glass of water in reality). Moreover, the flow of a liquid volume (e.g. creek) can be determined automatically in accordance with the environment.

*e-mail: hilko.cords@uni-rostock.de

On the other hand, strict surface simulations are restricted to flat surfaces with surface wave propagation and interferences. A common approach is the use of the 2D wave equation. Although a 3D dynamic flow cannot be simulated, the wave propagation of a moving obstacle, e.g. a boat, can be simulated with excellent results – even for liquids with low viscosity. This approach achieves highly detailed results at reasonable framerates, due to the existence of fast solvers for the wave equation (partial differential equation of second order in 2D).

The proposed technique combines the advantages of both physically-based approaches, to reach a real-time and interactive liquid simulation with a detailed surface. Finally, we utilize a 2,5D rendering approach for visualization: Since most liquid surfaces are 2,5D, using height fields for surface representations is an intuitive approach. However, this assumption is not valid in special scenarios – e.g. splashing or breaking waves.

2 Related Work

Photo-realistic simulation and rendering of liquids and related effects is still not affordable in real-time on today's desktop computers (e.g. [Carlson et al. 2004] [Hong and Kim 2005] [Guendelman et al. 2005] [Müller et al. 2005]). In fact, the high dynamics of flowing liquids and the complexity of interaction between e.g. light and water prohibit a photo-realistic and universal simulation at interactive rates. Thus, a lot of work has been done in the field of computer graphics simplifying or approximating the physical behavior of liquids as well as the physically based visual appearance.

In the field of computer graphics, the Navier-Stokes equations are usually solved with particle-based systems (e.g Smoothed Particle Hydrodynamics - SPH), the finite difference approach or hybrid methods [Adabala and Manohar 2002]. Determining the physically exact solution within a defined error margin is computationally very intensive. Large quantities of water such as a lake or the sea are still impossible to simulate at interactive frame rates on personal computers. In [Stam and Fiume 1995] the first real-time approach using SPH is presented, simulating gaseous phenomena. This is extended to the interactive simulation of fluids in [Müller et al. 2003] allowing simulations with a few thousand particles at interactive rates. The finite difference approach for simulating liquids was introduced to the computer graphics community as the marker and cell method in [Harlow and Welch 1965] and has also become popular [Foster and Metaxas 1996] [Carlson et al. 2004] [Foster and Fedkiw 2001], whereas the implicit technique for interactive simulation of fluids was introduced in [Stam 1999]. This approach has been enhanced for execution on the GPU with reasonable frame rates [Harris 2005]. Many extensions to these techniques have been proposed, including approaches replacing the 3D Navier-Stokes equations by 2D ones and extracting a height field from density values with good performance [Chen and da Vitoria Lobo 1995]. A SPH based real-time approach including rigid body interaction has recently been presented [Amada 2006]. This approach uses a traditional marching cubes algorithm for surface extraction and achieves good interactive results. Lately, the Lattice Boltzmann method has been become popular (e.g. [Thuersey et al. 2007]): Instead of solving the Navier-Stokes equations directly, the Boltzmann equation of kinetic theory of gases is solved. For neatly chosen parameters, the resulting flows are equivalent. The 2D wave equation can also be used to simulate surface waves (e.g. [Gomez 2000]). The propagation of radial waves can be used to simulate raindrop impacts on water surfaces or e.g. a moving boat with reasonable results. However, the water is not interactive, merely the propagation of radial waves is simulated. Recently, a promising method of solving the wave equation was presented [Yuksel et al. 2007]: The radial surface wave propagation is determined by the use of particles in 2D. A heightfield is determined by the particle positions, resulting in good results at high framerates. The coupling of fluid and objects is also discussed. Many liquid-specific effects (e.g. rain or puddles) can be simulated very realistically using neat tricks, without a complex underlying physical simulation [Tatarchuk 2006].

One of the main bottlenecks for interactive animation methods established is the surface or volume reconstruction from simulation data. In the majority of cases an expensive marching cubes algorithm [Lorensen and Cline 1987] or sometimes a surface splatting technique [Zwicker et al. 2001] is used, substantially reducing the possible amount of interactively simulated liquids. However, fast rendering of 2.5D surfaces can be achieved with LOD vertex displacement mapping, storing the height information in a texture, which is accessed in the vertex shader to translate the vertex positions [Kryachko 2005]. In principle, those techniques can be applied to the surface rendering algorithm presented here.

Real-time *rendering* of water surfaces is usually either based on environment mapping techniques or approximating raytracing techniques. The former approach includes a fast technique, where the environment is split by the water surface: The scene parts lying underwater are drawn into an environment map, which is accessed during the surface rendering pass by a fragment shader program to simulate the typical refractive behavior of water surfaces [Sousa 2005]. Even though non-physical, this approach achieves good visual results and performance [Belyaev 2003], visualizing lakes and the open water. Image-based double refraction using an environment map can be realized with a two pass rendering approach [Wym05]: The refractive object is split into back and front faces. Restricted to static environment maps (lying at infinite distance), the approach achieves reasonable results at high frame rates. Using two heightfields (ground and surface field), a simplified raytracer can be realized on the GPU [Baboud and Decoret 2006]: The intersection with refracted and reflected rays is calculated on the GPU with reasonable frame rates. Other GPU-based approaches towards real-time raytracing built upon some major simplifications have recently been proposed [Szirmay-Kalos et al. 2005] [Popescu et al. 2006].

The mode splitting method is used e.g. in oceanography to increase performance of ocean simulation models. The Princeton Ocean Model (POM) was one of the first free surface models, using the mode splitting method [Blumberg and Mellor 1987].

3 Our Approach

According to liquid simulations solving the full 3D Navier-Stokes equations for incompressible flows in real-time, we observe the following problems:

1. Due to the constraint of real-time, relatively large time-steps have to be used.
2. Due to the constraint of real-time, a relatively low sampling density of the liquid volume has to be used (Eulerian methods: small grid size, Lagrangian methods: few particles).

Hence, fast and highly detailed surface effects (e.g. a moving boat) cannot be simulated in such environments. On the other hand, the physical description of a liquid volume presumes the use of the Navier-Stokes equations. Thus, we propose the mode-splitting method (known from physics) to allow the additional simulation of highly detailed surface effects: The idea is to split the simulation of high and low frequency dynamics (sec. 3.1). Highly detailed surface waves are simulated according to the physical wave propagation described by the wave equation: The 2D wave equation (sec. 3.2) is solved with the finite difference method (FDM). The

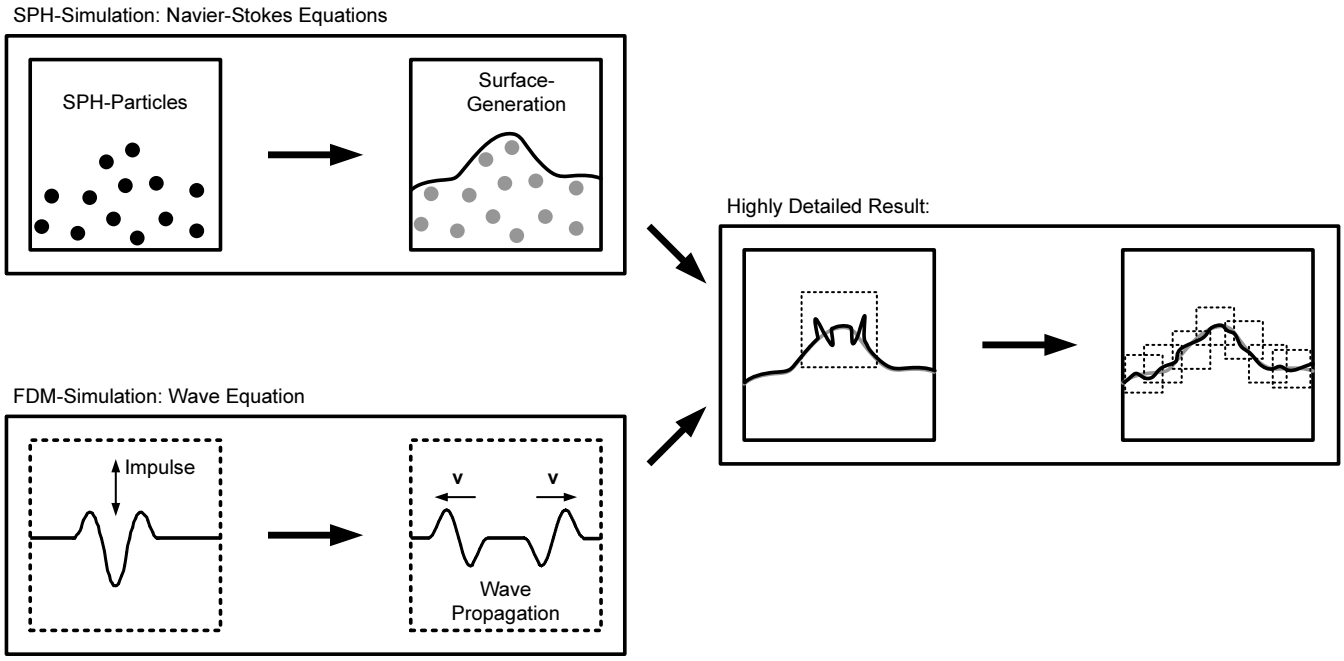


Figure 2: *Basic principle: The global behavior of the liquid, described by the Navier-Stokes equations, is simulated with the SPH method (top, 2D crosscut). Additionally, the surface is simulated in high detail by the wave equation (bottom, 1D example). The results can be combined to a highly detailed surface (right).*

liquid flow is described by the Navier-Stokes equations. These are solved with the smoothed particle hydrodynamics (SPH) method (sec. 3.3).

The key idea is recapitulated in Figure 2 and the following table:

Region	Dimension	Simulation	Principle
Liquid Surface	2D	FDM	Wave Eq.
Liquid Volume	3D	SPH	N.-S. Eq.

Coupling the two methods described (sec. 3.4, 3.5), a basin of highly detailed, physically-based liquid can be handled interactively. For visualization (sec. 3.6), we use a height field-based rendering approach. As mentioned above, most liquid surfaces can be rendered appropriately as height fields. However, complex liquid phenomena, such as breaking waves or splashes, cannot be visualized as height fields.

3.1 Mode Splitting

A *mode* describes the state of an oscillation according to the order of the harmonic and associated eigenfrequency. E.g. a laser resonator of length l can amplify frequencies $\nu(N)$ of the Nth mode (c: speed of light):

$$\nu(N) = N \cdot \frac{c}{2 \cdot l}. \tag{1}$$

The *mode splitting method* originates from physical simulations in the fields of oceanography and laser physics. In oceanography the method is used to simulate simultaneously high frequency waves (external gravity waves – induced by tide and atmospheric pressure, e.g. water waves, free surface waves) and low frequency waves (internal gravity waves – circulations induced by wind and density gradients, e.g. vertical turbulences). Thereby, different algorithms are used, adapted to the specific problems: The external and internal waves are solved separately with different time steps. The fast moving external waves need to be solved at small time-steps. The slow moving internal waves are more expensive to solve (due to complex turbulences), but a much larger time step can be used.

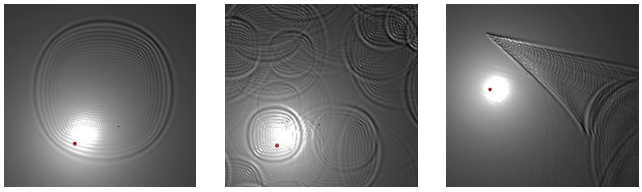
We adapt this idea to liquid simulations in the field of computer graphics: In practice, the surface is most important for the visual appearance of liquids – e.g. the complex, highly dynamic and highly detailed water surface of a small river head. However, the 3D dynamic flow e.g. of a river cannot be seen directly – it can just be seen indirectly – through surface variations or objects moving with the flow. This principle can be seen in Figure 4: The complex 3D flow (4b) cannot be observed after surface extraction (4c). With respect to the visual importance of the surface dynamics, the surface should be sampled and simulated appropriately to get highly detailed results. However, the 3D flow can be sampled with a low resolution. Thus, we use the 2D wave equation for surface simulation and a 3D SPH-based Navier-Stokes equations solver for volume flow simulation.

3.2 Surface Simulation

The general wave equation describes the propagation of waves in time t and space $\mathbf{x}$. For liquid surface waves the 2D wave equation can be used, describing the circular wave propagation:

$$\triangle f(\mathbf{x},t) - \frac{1}{c^2}\frac{\partial^2 f(\mathbf{x},t)}{\partial t^2} = 0. \tag{2}$$

Thereby, $\triangle = \nabla^2 = \sum_1^2 \frac{\partial^2}{\partial x_i^2}$ is the Laplace operator in 2D and c is the velocity at which waves propagate across the surface. Being a linear partial differential equation, the wave equation can be solved straightforwardly with an Eulerian finite difference approach. A high-resolution grid can be used for discretization – the algorithm is still fast because of the two dimensional nature of the problem. Boundary conditions (e.g. collision objects) can easily be included as wave reflections from boundaries: Adapting $f(\mathbf{x},t)$ at



(a) (b) (c)

Figure 3: *Lit solutions of the 2D wave equation – seen from above: A radial wave propagates from center (a). Several radial waves propagate (b), resulting in a raindrop impact animation. A swimming object is moving on the surface (c). Due to the 2D FDM approach, the wave propagation of free surface waves can be simulated highly detailed in real-time.*

boundary positions models the reflection process. Radially propagating waves can be created at position $\mathbf{x}_0$ and time t_0 by displacing $f(\mathbf{x}_0, t_0)$ – see Figure 3a,b. Displacing several $f(\mathbf{x}_i, t_0)$ can create e.g. wave trains or the typical wave propagation of a moving boat (Figure 3c). According to the liquid depth, a velocity map can vary c – in shallow water phase, velocity depends on water depth. A damping factor less than 1 should be included in the simulation for numerical stability purposes. In combination with a damping map the damping according to e.g. the type of ground can be described.

The function $f(\mathbf{x}, t)$ can be used directly as a height field. The surface normals can be calculated straightforwardly. Sometimes it is convenient to additionally smooth the height field or the associated normals.

3.3 Liquid Simulation

The dynamic flow of liquids is described by the conservation of momentum (Navier-Stokes equations)

$$\rho\left(\frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v}\cdot\nabla)\mathbf{v}\right) = -\nabla p + \mu\triangle\mathbf{v} + \rho\mathbf{f} \tag{3}$$

and the conservation of mass (continuity equation):

$$\frac{\partial \rho}{\partial t} + \nabla\cdot(\rho\mathbf{v}) = 0, \tag{4}$$

where $\mathbf{v}$ is the velocity field, ρ the density field, p the pressure field, $\nabla = (\frac{\partial}{\partial x_1}, \frac{\partial}{\partial x_2}, \frac{\partial}{\partial x_3})^T$, μ is the viscosity and $\mathbf{f}$ is the acting external force (e.g. gravity). Since we assume incompressible liquids, the density is constant ($\frac{\partial \rho}{\partial t} = 0$), resulting in the mass conservation

$$\nabla\mathbf{v} = 0. \tag{5}$$

In general, the Navier-Stokes equations are solved with the Eulerian formulation (finite difference method, grid based) or the Lagrangian formulation (particle-based, e.g SPH). Numerically, the benefits from SPH for real-time fluid simulation are:

- Simple and fast handling of boundary conditions as collisions,
- mass conservation is guaranteed (number of particles = const.; mass of each particle = const.),
- the nonlinear convective acceleration term $(\mathbf{v}\cdot\nabla)\mathbf{v}$ can be neglected in particle systems, because the particles are moving with the fluid and thus, the convection is directly included,
- less memory requirements.

Hence, we use SPH for solving the Navier-Stokes-Equations in 3D - a good introduction can be found in [Monagham 1992]. The liquid volume is discretized by particles. These particles represent the physical properties of the surrounding liquid. Each particle gets a position $\mathbf{x} = (x, y, z)$, a velocity $\mathbf{v} = (v_x, v_y, v_z)$ and a mass m (Figure 4a,b). Basically, these particles are used in combination with a smoothing kernel $W_h(\mathbf{x})$ for solving the Navier-Stokes Equations: The continuous field quantities are distributed in the local neighborhood according to the discrete particle positions and the smoothing kernels $W_h(\mathbf{x})$. According to SPH, $W_h(\mathbf{x})$ is an interpolation kernel, which must satisfy volume conservation

$$\int W_h(\mathbf{x})d\mathbf{x} = 1 \tag{6}$$

and the Dirac delta function in the limit $h \to \infty$:

$$\lim_{h\to\infty} W_h(\mathbf{x}) = \delta(\mathbf{x}). \tag{7}$$

The smoothing length h defines the width of the smoothing kernel:

$$W_h(\|\mathbf{x}\| > h) = 0. \tag{8}$$

Hence, h also defines the distance at which particles interact with each other. According to SPH, scalar quantities $A(\mathbf{x})$ can be estimated for n particles as follows:

$$\langle A(\mathbf{r})\rangle = \sum_{i=1}^{n} \frac{m_i}{\rho(\mathbf{x}_i)} A(\mathbf{x}_i) W_h(\mathbf{x} - \mathbf{x_i}), \tag{9}$$

$$\langle \nabla A(\mathbf{r})\rangle = \sum_{i=1}^{n} \frac{m_i}{\rho(\mathbf{x}_i)} A(\mathbf{x}_i) \nabla W_h(\mathbf{x} - \mathbf{x_i}). \tag{10}$$

Thus, local velocity and pressure changes can be determined and are used for integration of particle positions and velocities. We use the smoothing kernels for pressure and viscosity, pressure force $\mathbf{f}_i^{\text{pressure}}$ and viscosity force $\mathbf{f}_i^{\text{visc}}$ as described in [Müller et al. 2003]:

$$W_h^{\text{press}}(x) = \frac{15}{\pi h^6} \begin{cases} (h-x)^3 & : \ 0 \le x \le h \\ 0 & : \ \text{otherwise,} \end{cases} \tag{11}$$

$$W_h^{\text{visc}}(x) = \frac{15}{2\pi h^3} \begin{cases} -\frac{r^3}{2h^3} + \frac{r^2}{h^2} + \frac{h-2r}{2r} & : \ 0 \le x \le h \\ 0 & : \ \text{otherwise,} \end{cases} \tag{12}$$

$$\mathbf{f}_i^{\text{press}} = -\sum_j m_j \frac{p_i + p_j}{2\rho_j} \nabla W_h^{\text{press}}(\mathbf{x}_i - \mathbf{x}_j), \tag{13}$$

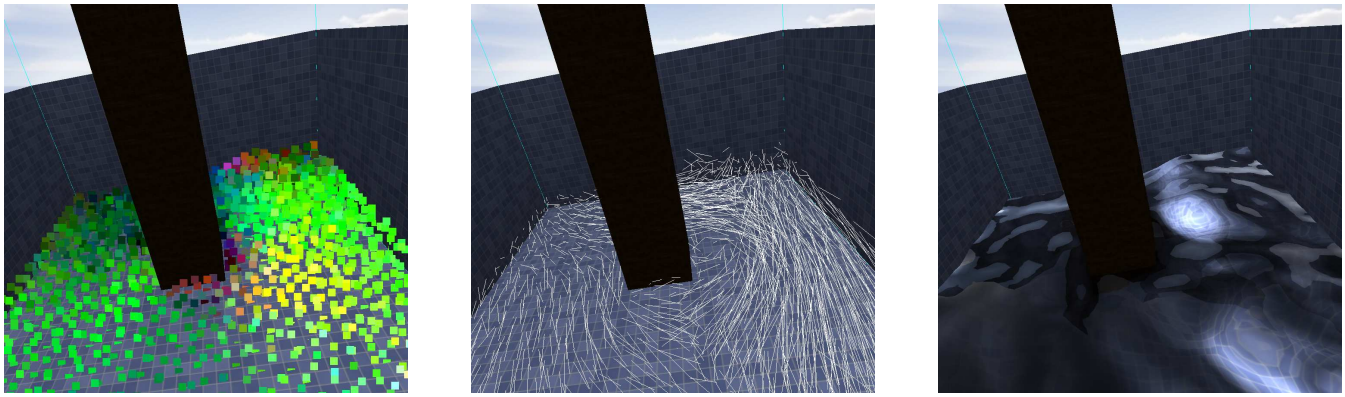
$$\mathbf{f}_i^{\text{visc}} = -\mu \sum_j m_j \frac{\mathbf{v}_j - \mathbf{v}_i}{2\rho_j} \nabla^2 W_h^{\text{visc}}(\mathbf{x}_i - \mathbf{x}_j). \tag{14}$$

According to equation 8, just neighbors closer than distance h influence the physical properties at a given location. Computationally, these can be found with good performance, if an additional space dividing grid with grid length h is used. Hence, just the associated and neighboring grid elements have to be checked for potential influencing particles.

3.3.1 Collisions

Collisions of liquid particles with objects are handled by using a force vector field surrounding collision objects:

$$\mathbf{f}_{\text{col}} \sim \frac{1}{d+1} \cdot \mathbf{n}_{\text{object}}, \tag{15}$$



(a) (b) (c)

Figure 4: *Principle of SPH simulation: The volume is discretized using particles (in this example: 2000 particles). These are used to solve the Navier-Stokes equations. In image (a) the velocities along the x,y and z-axis are color encoded ($\mathbf{v} * scale + offset = (r, g, b)$). In image (b) the velocities are shown according to their direction and their norm (length of vector). The extracted surface is shown in (c): Due to the low volume sampling with just 2000 particles, the SPH-generated surface is low-detailed.*

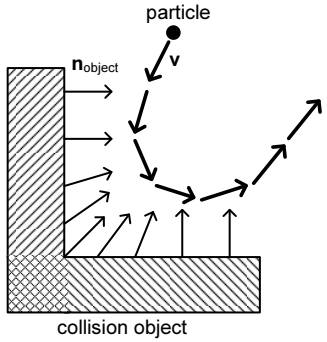


Figure 5: *Principle of the collision handling.*

where d is the closest distance between object and particle and $\mathbf{n}_{\text{object}}$ the normal vector of the object at the point of closest approach. Thus, the force $\mathbf{f}_{\text{col}}$ is acting on each particle being close to collision objects. The object normals $\mathbf{n}_{\text{object}}$ are smoothed around hard object edges (Figure 5). Hence, the construction of collision objects using basic objects is possible and leads to correct collision handling. This explicit approach even handles particles with high velocities: A particle inside an object will experience a force with object normal direction. Thus, even moving obstacles can be handled, whereas colliding particles should observe conservation of momentum.

Using the described force field, the collision handling process is restricted to a few time-steps, resulting in a substantially more stable simulation than the intuitive approach of reflecting velocities with friction η ($\mathbf{v}_{\text{new}} = \eta\text{reflect}(\mathbf{v}, \mathbf{N})$). The instantaneous velocity changes of this basic approach in combination with large time-steps tend to destabilize the particles surrounding the collision position.

3.3.2 Free Surface Extraction

The height field surface is created by means of an 2,5D iso-surface of an implicit function. The potential ϕ for n particles with positions $\mathbf{x}_i$ ($i = 1 \dots n$) is determined by the following spherical potentials (h: iso-radius):

$$\phi(\mathbf{x}) = \sum_{i=1}^{n} \sqrt{1 - \frac{r_i^2}{h^2}} \tag{16}$$

$$r_i = \|\mathbf{x} - \mathbf{x}_i\| \tag{17}$$

The square root (eq. 16) can be approximated to increase performance. The exact height field is determined by a threshold. In fact, just surface particles and their neighbors contribute to the resulting height field. These particles can be detected according to their actual number of neighbors. Performance can be increased if only these particles are used for surface generation. An example of a generated surface is shown in Figure 4c.

In practice, the height field and its normal field should be additionally smoothed, to reduce unwanted surface ripples caused by the discrete sampling of the liquid volume with few particles. Conserving the detailed information at intensive liquid motions, we propose an adaptive smoothing scheme: Depending on the kinetic energy

$$E_{\text{kin}} = \sum_{i=1}^{n} \frac{1}{2} m_i \, \mathbf{v}_i^2 \tag{18}$$

of the particle system with n particles, velocities $\mathbf{v}_i$ and masses m_i ($i = 1 \dots n$), the number of smoothing steps is determined. If E_{kin} exceeds a defined threshold, no smoothing occurs. Below this threshold, the number of smoothing steps is increased as E_{kin} increases until the maximum number of smoothing steps is reached. Thus, still liquids occurs to be flat and turbulent liquids remain detailed.

3.4 Simulation Time-Steps

The surface simulation (wave equation) has to be synchronized with the volume simulation (SPH). In general, the discretization of the surface is more detailed: Due to strict time step constraints depending on the degree of discretization, the surface simulation time-steps $\triangle t_{\text{surface}}$ are smaller than the volume simulation time-steps $\triangle t_{\text{volume}}$. We assume $\triangle t_{\text{surface}}$ to be an integer fraction N_{time} of $\triangle t_{\text{volume}}$:

$$N_{\text{time}} = \frac{\triangle t_{\text{volume}}}{\triangle t_{\text{surface}}}. \tag{19}$$

Thus, N_{time} surface simulation steps are completed per volume simulation step (Figure 6):

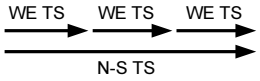


Figure 6: *Example of time step (TS) synchronization: $N_{time} = 3$. In this example, the dynamic wave equation (WE) is solved three times, while the Navier-Stokes equations (N-S) is solved once.*

3.5 Combining Surface and Volume Simulation

The determination of the final surface just depends on the different field resolutions, because we are using a height field based rendering approach (even for the SPH method). As for the simulation time-steps, we assume the SPH-generated surface (size: $X_{\text{SPH}} \times Y_{\text{SPH}}$) to be an integer fraction N_{surf} of the more detailed wave equation surface (size: $X_{\text{WE}} \times Y_{\text{WE}}$):

$$N_{\text{surf}} = \frac{X_{\text{WE}}}{X_{\text{SPH}}} = \frac{Y_{\text{WE}}}{Y_{\text{SPH}}}. \tag{20}$$

The SPH-generated surface is up-sampled bilinearly and the resulting height fields summed up (Figure 7). This procedure is suitable to be performed texture-based on the GPU.

A simulation coupling can be realized by increasing the value of $f(\mathbf{x}, t)$ (section 3.2) depending on the scaled derivation of the extracted SPH-field. And the positions of SPH-particles can be adapted according to high surface waves. Thus, a coupling of both simulations can be realized.

3.6 Rendering

Since we aim at real-time liquid simulation at high framerates, fast rendering becomes significant. Thus, fast approximating techniques are favored. At least, plausible *water* simulation should contain the approximation of optical effects e.g. reflections, refractions and caustics. Therefore, the height field-based liquid region is rendered using a cubemap (containing the environment) for approximating the effects of reflection and refraction. Thus, surface variations (positions, normals) are used for calculating reflection and refraction vectors from eye coordinates. Finally, the cubemap is accessed using these vectors. The composition of reflection and refraction is described by the Fresnel Equations. Caustics arise from light that is focused by reflective or refractive objects, producing remarkable light effects. A water rendering system should include some caustics to enhance realism. We use a fast texture based approach. A planar light map is generated via light raytracing using Snell's law. The light map is projected onto the scene from light view. However, to be fast, this planar approach fails to light the objects inside water or complex groundfields physically correctly. Therefore a real off-line raytracing approach is recommended. But we think the lack of physical accuracy is hardly noticeable and, due to the demand of interactivity, acceptable.

In particular, the use of effects on a per pixel basis (e.g. bumpmapping, chaotic reflexes, motion blurring) can enhance realism even more. In general, the animator can use the whole scope of shader effects to reach desired effects. In principle, any other liquid can be rendered similarly. In particular, non-transparent liquids, e.g. milk, cola or crude oil, can be simplified with respect to reflections, refractions and caustics. The visual behavior of those substances can be approximated within specific shader programs.

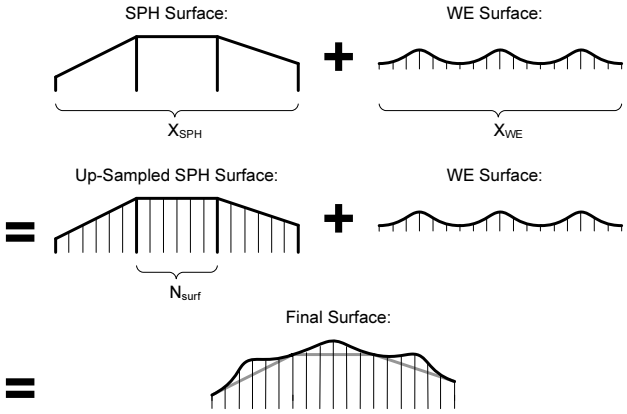


Figure 7: *Combining the resulting height fields (with different resolutions) from SPH-simulation and wave equation (WE) simulation.*

4 Implementation and Results

The proposed algorithms were implemented using OpenGL 2.0 and the related shading language GLSL in C++, as well as the simulating 3D SPH method and the wave equation solver. The presented examples were performed on a dual-core desktop PC with a $2,6$ GHz AMD Athlon 64 CPU, 2GBs of RAM and a graphics card based on an ATI Radeon x1900 GPU. We use a parallel implementation with one core simulating the physics with SPH and handling the user interactions and one core solving the wave equation, extracting the surface from particle system, creating the caustics-textures and finally, rendering:

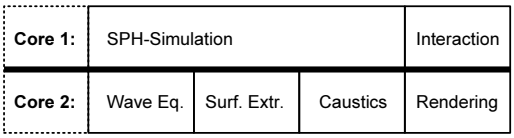


Figure 8: *Parallelization.*

Being unoptimized yet, we believe that the rendering algorithm could achieve much higher frame rates using a straightforward fully shader-based implementation (e.g. Render2VertexBuffer would speed up the height field and normal generation immensely). However, the measured frame rates achieved with the actual implementation are quite reasonable, as shown in table 1 (Resolution: 950×950 pixels).

The performance of the technique mainly depends on the following parameters:

- Number of SPH-particles,
- $X_{\text{SPH}} \times Y_{\text{SPH}}$,
- $X_{\text{WE}} \times Y_{\text{WE}}$.

The results of experiments show that the SPH simulation accounts for 40-70% of the run time – with less than 4000 particles. Consequently, no detailed surfaces or detailed surface effects can be simulated at interactive rates with the SPH-method (e.g. Figure 4(c)). The proposed technique solves this problem. However, a problem of the technique shared with many other height field based liquid simulation techniques is the impossibility to visualize 3D liquid effects – e.g. splashes or breaking waves. Due to the use of the 3D simulation technique SPH, those effects can be simulated, but because of the 2,5D rendering approach they cannot be visualized.

Example [Figure]	1. N-S (SPH) [No. of Particles]	2. N-S (SPH) $[X_{\text{SPH}} \times Y_{\text{SPH}}]$	3. WE (FDM) $[X_{\text{WE}} \times Y_{\text{WE}}]$	4. FPS $[\frac{1}{s}]$	5. No. of used Cores
1	2000	50×50	400×400	30	2
3 (a)-(c)	-	-	300×300	46	1
4 (c)	2000	50×50	-	102	2
9 (a)	2000	50×50	200×200	85	2
9 (b)	2000	50×50	200×200	93	2
10 (b)-(c)	2000	50×50	200×200	75	2

Table 1: *Performance measurements of the presented algorithm and different scenarios.*

(a) (b)

Figure 9: *A pool scene with caustics (a). Milk shader used for rendering (b).*

However, every water scene representable as a height field can be simulated and rendered with high surface details. Due to the use of SPH, even a 2D flow (e.g. a river) can be simulated by using few particles (increasing performance immensely), whereas the surface remains highly detailed. Recapitulating, the benefits of the presented combination of volume and surface simulations are:

- Volume interactions (e.g. moving a glass of water, obstacles),
- surface interactions (e.g. rain, moving objects),
- automatic, natural and global flow (e.g. of a river or creek),
- objects moving with the flow (e.g. leafs),
- interactions e.g. with an avatar,
- simulations of e.g. a pool, a sea, ...

The low viscosity of real water cannot be reached in real-time with Navier-Stokes simulation methods (due to large time-steps and the belonging necessary damping), resulting in a liquid behavior more analogue to e.g. oil than real water. But these unwanted physical effects are reduced with the presented technique, due to the fast calculation of the wave equation. Hence, the viscosity of the simulated fluid appears to be low. On the other hand, while decreasing the time-steps or reducing speed the convincing simulation of liquids with a viscosity greater than the viscosity of water, such as oil, honey or molten wax can be achieved easily.

5 Conclusion and Future Work

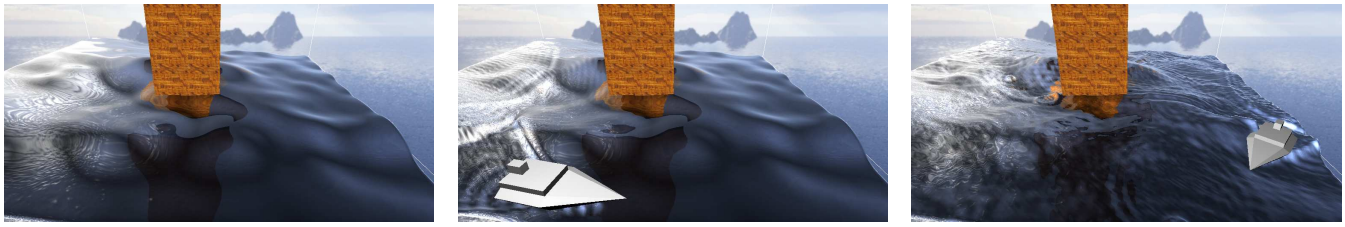
We have presented a method for highly detailed, interactive liquid simulation at high framerates. The physical model is based according to the principle of mode-splitting: The simulation of the low frequency liquid flow and the high frequency free surface waves are separated. Thereby, the surface is simulated according to the 2D wave propagation described by the 2D wave equation (FDM-method) – and the 3D liquid flow is simulated according to the full Navier-Stokes equations (SPH-method). Thus, the presented method reaches very realistic and highly detailed results. Nevertheless, the full volume flow is simulated (according to fluid dynamics) and can be handled interactively like e.g. a glass of water, as well as the propagation of surface waves – produced e.g. by impacts or a moving boat. The resulting liquid is rendered using a height field-based technique-

We demonstrated the potential of combining existing methods and a new coupling scheme according to the simulation of plausible, complex liquid effects in real-time. Of course, those effects have been used in computer animated films with perfect quality (based on slow off-line techniques), but not in real-time environments at high framerates. Hence, our approach is practical for interactive environments, e.g. VRs or video games. In view of multi-core architectures, our approach can enhance realism of such environments immensly. Although aiming for real-time environments, our fast method is interesting for high quality off-line rendering applications as well.

Our future investigation includes a better rendering approach (e.g. GPU-based approximated raytracing), a better coupling of the FDM- and SPH-method (e.g. moving, damping or disrupting of surface waves according to the flow or collisions), and the measurement of performances using the GPU or a PPU (Physics Processing Unit) for physical calculations. Finally, the use of adaptive surface rendering techniques would relieve the CPU and much larger liquid volumes could be simulated.

References

ADABALA, N., AND MANOHAR, S. 2002. Techniques for realistic visualization of fluids: A survey. *Computer Graphics Forum 21*,



(a) (b) (c)

Figure 10: *A low resolution height field (a) is extracted from the dynamic SPH-simulation. Using our method, highly detailed surface waves and surface effects can be added dynamically to the low detailed surface, resulting in a more realistic surface representation. In (b) a small boat is added, creating detailed bow waves. Additional random rain drop impacts increase the chaotic character of the surface (c).*

1, 65–81.

AMADA, T. 2006. Real-time particle based fluid simulation with rigid body interaction. In *Game Programming Gems 6*, Charles River Media, 189–205.

BABOUD, L., AND DECORET, X. 2006. Realistic water volumes in real-time. In *Eurographics Workshop on Natural Phenomena*, Eurographics.

BELYAEV, V. 2003. Real-time simulation of water surface. In *GraphiCon-2003*, OOO "MAX Press", 131–138.

BLUMBERG, A., AND MELLOR, G. 1987. A description of a coastal ocean circulation model. In *Three dimensional ocean models*, American Geophysical Union, Washington D.C., 1–16.

CARLSON, M., MUCHA, P., AND TURK, G. 2004. Rigid fluid: Animating the interplay between rigid bodies and fluid. In *ACM Transactions on Graphics*, vol. 23, 377–384.

CHEN, J. X., AND DA VITORIA LOBO, N. 1995. Toward interactive-rate simulation of fluids with moving obstacles using navier-stokes equations. *Graph. Models Image Process. 57*, 2, 107–116.

FOSTER, N., AND FEDKIW, R. 2001. Practical animation of liquids. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM Press, 23–30.

FOSTER, N., AND METAXAS, D. 1996. Realistic animation of liquids. *Graph. Models Image Process. 58*, 5, 471–483.

GOMEZ, M. 2000. Interactive simulation of water surfaces. In *Game Programming Gems*, Charles River Media, 187–195.

GUENDELMAN, E., SELLE, A., LOSASSO, F., AND FEDKIW, R. 2005. Coupling water and smoke to thin deformable and rigid shells. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*, ACM Press, 973–981.

HARLOW, F. H., AND WELCH, J. E. 1965. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *Phys. Fluids 8*, 12, 2182–2189.

HARRIS, M. 2005. Fast fluid dynamics simulation on the gpu. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Courses*, ACM Press.

HONG, J.-M., AND KIM, C.-H. 2005. Discontinuous fluids. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*, ACM Press, 915–920.

KRYACHKO, Y. 2005. Using vertex texture displacement for realistic water rendering. In *GPU Gems 2*, Addison-Wesley, 283–294.

LORENSEN, W. E., AND CLINE, H. E. 1987. Marching cubes: A high resolution 3d surface construction algorithm. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, ACM Press, 163–169.

MONAGHAM, J. 1992. Smoothed particle hydrodynamics. 543–574.

MÜLLER, M., CHARYPAR, D., AND GROSS, M. 2003. Particle-based fluid simulation for interactive applications. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, Eurographics Association, 154–159.

MÜLLER, M., SOLENTHALER, B., KEISER, R., AND GROSS, M. 2005. Particle-based fluid-fluid interaction. In *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, ACM Press, 237–244.

POPESCU, V., MEI, C., DAUBLE, J., AND SACKS, E. 2006. Reflected-scene impostors for realistic reflections at interactive rates. *Computer Graphics Forum 25*, 3 (Sept.), 313–322.

SOUSA, T. 2005. Generic refraction simulation. In *GPU Gems 2*, Addison-Wesley, 295–305.

STAM, J., AND FIUME, E. 1995. Depicting fire and other gaseous phenomena using diffusion processes. *Computer Graphics 29*, Annual Conference Series, 129–136.

STAM, J. 1999. Stable fluids. In *Siggraph 1999, Computer Graphics Proceedings*, Addison Wesley Longman, 121–128.

SZIRMAY-KALOS, L., ASZÓDI, B., LAZÁNYI, I., AND PREMECZ, M. 2005. Approximate ray-tracing on the gpu with distance impostors. In *Computer Graphics Forum (Proc. of Eurographics 2005)*, vol. 24.

TATARCHUK, N. 2006. Artistic-directable real-time rain rendering in city environments. In *SIGGRAPH'06 - Advanced Real-Time Rendering in 3D Graphics and Games - Chapter 3*, 23–63.

THUERSEY, N., SADLO, F., SCHIRM, S., MUELLER, M., AND GROSS, M. 2007. Real-time simulations of bubbles and foam within a shallow-water framework.

YUKSEL, C., HOUSE, D. H., AND KEYSER, J. 2007. Wave particles. *ACM Siggraph 2007 Conference Proceedings*.

ZWICKER, M., PFISTER, H., VAN BAAR, J., AND GROSS, M. 2001. Surface splatting. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM Press, 371–378.