EUROGRAPHICS '0x / N.N. and N.N.
(Editors)

Volume 0 (200x), Number 0

Enhanced Lattice Boltzmann Shallow Waters for real-time fluid simulations

J. Ojeda[1] and A. Susín[2]

[1]LSI, Universitat Politècnica de Catalunya, Spain
[2]MA1, Universitat Politècnica de Catalunya, Spain

Abstract

We present a novel approach at simulating fluids in real-time by coupling the Lattice Boltzmann Method for Shallow Waters (LBMSW) with particle systems. The LBM can handle arbitrary underlying terrain and arbitrary fluid depth, which, in turn, allows us to extend it to track dry regions. The LBM is also two-way coupled with rigid bodies. The particle system adds more detail to the LBM; breaking waves are detected from the surface simulation and particles are generated to provide the effect, taking effectively certain amounts of fluid and reintegrating it back once they fall over again. Both the LBM and the particle simulation are implemented in CUDA, although rigid bodies are simulated in CPU. Finally, we show the effectiveness of the method on commodity hardware.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation I.6.8 [Simulation and Modeling]: Types of Simulation—Animation

1. Introduction

Nowadays there is an increasing interest in achieving physically realistic simulations in interactive applications, as videogames are. One of the most complex effects is the simulation of fluids which has been traditionally relegated to procedural techniques like [Tes01], which work great for large amounts of water as in a sea, but aren't easily coupled with solid objects and can't simulate the flow of the fluid. A novel approach in this field was presented in [YHK07] using wave trains on 2D particles.

For full 3D simulation of water there are many works. In general, they are quite costly and may still be difficult to bring to real-time animations. We refer the reader to [Bri08] for an overview on grid based simulations and [Sol09] and references therein for information about particle based fluids.

Alternatively, if only the fluid surface is required, to physically simulate the flow of a fluid a shallow water framework is preferred. It is based on the Shallow Water Equations (SWE) and [LvdP02] were the first to introduce them to the graphics field. Among other works, [TMFSG07] used them to simulate breaking waves and were more recently ported to CUDA and coupled with a particle system in [CM10]. Other formulations include using Smoothed Particle Hydrodynamics (SPH) like [Cor07, SBC*11] or even the Lattice Boltzmann Method (LBM); [Thü07] coupled it with a full 3D LBM simulation and [TSB07] simulated the currents of the strait of Gibraltar.

In this fashion, we present an alternative method to simulate fluids at interactive framerates, by using the LBMSW formulation with the following key features:

- Tracking of dry regions and increased stability.
- Two-way coupling with rigid body motion.
- Full particle coupling with breaking wave detection conditions.

Although we have used a ballistic particle system, the particle coupling is loose enough to allow any kind of particle system, like SPH.

2. Methodology

The main steps this hybrid fluid simulation carries out for each time step can be summarized as follows:

1. LBMSW fluid simulation and dry region tracking (Sections 2.1 and 2.2).
2. Rigid body simulation and two-way coupling with the LBMSW (Section 2.3).

3. Particle coupling and simulation (Section 2.4).
4. Render the scene.

The first step is to advance the LBMSW fluid simulation, which takes into account external forces and tracking of dry regions. Next, the simulation of rigid bodies is done, in our case using the Bullet Physics library, and the rigid bodies are then coupled with the fluid. Breaking wave conditions are executed over the fluid surface, detecting when a wave is steep enough to break, generating particles which subtract volume from the LBMSW. These particles evolve independently in successive frames until they hit the surface of the fluid again and are reintegrated. Finally, the scene is visualized.

2.1. Lattice Boltzmann Method for Shallow Waters simulation

The Lattice Boltzmann Method provides a discrete model based on arithmetic operations, which can be easily adapted to parallel architectures like CUDA. The fluid is simulated by the movement of particles (represented by distribution functions df) in a limited set of directions $\mathbf{e}_i$ defined by the discretization used.

As the Shallow Waters framework simulates the surface of a fluid as a heightfield, the discretization of the LBM is restricted to 2D, so we use the common D2Q9 model for the LBM, assuming an adimensional parametrization as in [Thü07].

We use here the popular BGK operator to define how particles distributions interact at each cell, so the Lattice Boltzmann Equation can be written as

$$f_i(\mathbf{x}+\mathbf{e}_i\Delta t, t+\Delta t) = (1-\omega) f_i(\mathbf{x},t) + \omega f_i^{eq} + \mathcal{F}_i, \quad (1)$$

where f_i is the df in the $\mathbf{e}_i$ direction, ω is the relaxation parameter and f_i^{eq} is the local equilibrium distribution which we use like in, e.g., [Sal99]

$$f_i^{eq}(h,\mathbf{u}) = \begin{cases} h\left(1-\frac{5}{6}gh-\frac{2}{3}\mathbf{u}^2\right), & i=0, \\ \lambda_i h\left(\frac{gh}{6}+\frac{\mathbf{e}_i\cdot\mathbf{u}}{3}+\frac{(\mathbf{e}_i\cdot\mathbf{u})^2}{2}-\frac{\mathbf{u}^2}{6}\right), & i\neq 0, \end{cases} \quad (2)$$

where $\lambda_i = 1$ for $i = 1..4$ and $\lambda_i = 1/4$ for $i = 5..8$, g is the gravity and h and $\mathbf{u}$ are the macroscopic fluid properties, surface height level and velocity, which are computed as

$$h(\mathbf{x},t) = \sum_i f_i, \quad \mathbf{u}(\mathbf{x},t) = \frac{1}{h}\sum_i \mathbf{e}_i f_i. \quad (3)$$

From Equation 1, $\mathcal{F}_i$ are the external forces applied to the LBMSW, which take into account the underlying terrain among others, and we define them as in [Zho11].

Here we will also define another value η, which is used in the following sections, as

$$\eta(\mathbf{x},t) = h(\mathbf{x},t) + z_b(\mathbf{x}), \quad (4)$$

where z_b is the underlying terrain, defined as a heightfield.

2.2. Dry regions

We modify the original algorithm to allow the tracking of dry regions, i.e., cells that do not contain any fluid. We can make the distinction between Fluid and Empty cells just by imposing a threshold ε, which defines the minimal height a cell must satisfy to be considered Fluid. After an iteration of the classic algorithm is completed, we look for cells who may have felt below the threshold (Fluid_to_Empty) or cells who may have their height above the threshold and have Empty neighbours (Empty_to_Fluid).

In the Fluid_to_Empty case, the cell is tagged Empty and we redistribute the remainder of the fluid among the neighbouring Fluid cells favoring the slope of the underlying terrain as

$$f_i(\mathbf{x}+\mathbf{e}_i\Delta t) = f_i(\mathbf{x}+\mathbf{e}_i\Delta t) + h(\mathbf{x},t)\cdot(\xi_i/\xi_{total}), \quad (5)$$

where ξ_{total} is the sum of all ξ_i, which are computed as

$$\xi_i = \begin{cases} -(\nabla z_b\cdot\mathbf{e}_i), & \text{if } -(\nabla z_b\cdot\mathbf{e}_i)>0 \text{ and cell at} \\ & (\mathbf{x}+\mathbf{e}_i) \text{ is Fluid}, \\ 0, & \text{otherwise.} \end{cases} \quad (6)$$

In the Empty_to_Fluid case, the neighbouring Empty cells are just tagged as Fluid cells in order to make them available to take part in the simulation in the next iteration.

This modification to the original algorithm introduces a problem. From Equation 3, if the fluid level comes down, the velocities can grow large and lead to instabilities. In contrast to [GRGT10], where they worked around this by using a minmod flux limiter, we have used a more direct approach. As instabilities are introduced to the LBM when the fluid velocities surpass the wave propagation speed for the actual lattice, this means that the LBM is restricted to subcritical flows, defined by their Froude number being $Fr < 1$:

$$Fr = \sqrt{\frac{\mathbf{u}\cdot\mathbf{u}}{gh}}. \quad (7)$$

We define an upper limit parameter $\phi < 1$ for this ratio. When, due to low fluid height, the ratio does not hold, we compute a new suitable velocity $\mathbf{u}$ for the fluid and replace the dfs of the cell with new ones computed from Equation 2. We can further use this condition to dampen high velocities through the whole fluid domain and ensure a stable simulation.

2.3. Rigid objects

For the coupling of dynamic objects we follow the same strategy as [YHK07, CM10], but instead of tessellate the mesh of the object, we introduce a proxy model to decouple the complexity of the interaction from the object mesh.

This proxy model is composed of a set of spheres with some properties, and can be understood as a rough discretization of the object mesh. The properties defined for the

spheres are the radius S^r, the position $\mathbf{S^p} = (S^{p_x}, S^{p_y}, S^{p_z})^T$ and a normal vector $\mathbf{S^n} = (S^{n_x}, S^{n_y}, S^{n_z})^T$. During the simulation the spheres will also hold a velocity $\mathbf{S^v} = (S^{v_x}, S^{v_y}, S^{v_z})^T$.

The main forces exerted to a dynamic object due to fluid behaviour are buoyancy, drag and lift. They are computed at the sphere positions. Required values from the fluid are bilinearly interpolated and we assume the simulation plane is xz, so $\hat{y} = (0,1,0)^T$.

The buoyancy force is proportional to the displaced fluid and points upward. For the ith sphere, it is defined as

$$\mathbf{f}_i^{buoy} = \begin{cases} 0, & \text{if } S_i^{p_y} - S_i^r > \eta(\mathbf{S^p}), \\ g\rho V_{sub}\hat{y}, & \text{otherwise,} \end{cases} \tag{8}$$

where ρ is the density of the fluid and V_{sub} is the volume of the submerged part of the sphere computed as

$$V_{sub} = \int_{-S_i^r}^{top} \pi(S_i^{r2} - x^2)dx, \tag{9}$$

with $top = (\eta(\mathbf{S^p}) - (S_i^{p_y} - S_i^r))$.

Drag force is a resistive force dependent on the actual velocity of the object with regard to the fluid. Lift is perpendicular to the oncoming flow direction, in contrast with drag, which is parallel. They are defined as

$$\mathbf{f}_i^{drag} = -\frac{1}{2} C_D A_{2D} \|\mathbf{u}_{rel}\| \mathbf{u}_{rel}, \tag{10}$$

$$\mathbf{f}_i^{lift} = -\frac{1}{2} C_L A_{2D} \|\mathbf{u}_{rel}\| \left(\mathbf{u}_{rel} \times \frac{\mathbf{S}_i^{\mathbf{n}} \times \mathbf{u}_{rel}}{\|\mathbf{S}_i^{\mathbf{n}} \times \mathbf{u}_{rel}\|} \right), \tag{11}$$

where C_D and C_L are the drag and lift coefficients, $\mathbf{u}_{rel}$ is the relative velocity of the sphere with respect to the fluid and A_{2D} is the area of the circle that cuts the sphere at fluid level $\eta(\mathbf{S^p})$.

The inverse coupling is also done, the objects modify the fluid behaviour. The computations are done per sphere as well. We compute the displaced fluid h_o and the introduced velocity from the sphere $\mathbf{u}_o$, as

$$h_o = \exp(-depth) \cdot C_{dis} \cdot depth, \tag{12}$$

$$\mathbf{u}_o = \exp(-depth) \cdot C_{adp} \cdot \mathbf{S}_i^{\mathbf{v}}, \tag{13}$$

where $depth$ is the difference between the submerged height of the sphere and the fluid level and C_{dis} and C_{adp} are parameters in the range $[0,1]$ that allow to dampen the effect of the coupling as desired.

With these new h_o and $\mathbf{u}_o$ we can update the LBM dfs at the sphere positions, interpolating between the nearest cells, as

$$f_0 = f_0 - h_o \tag{14}$$

$$f_i = f_i + f_i^{eq}(h_o, \mathbf{u}_o) + \frac{f_0^{eq}(h_o, \mathbf{u}_0)}{\mathsf{w}_o}, \tag{15}$$

where $\mathsf{w}_o = 5$ for $i = 1..4$ and $\mathsf{w}_o = 20$ for $i = 5..8$, calculated from the contributions each $\mathbf{e}_i$ provides on the D2Q9 model [HL97]. With this computation we effectively push the fluid that the obstacle is displacing to the neighbour cells, as shown in Figure 1.

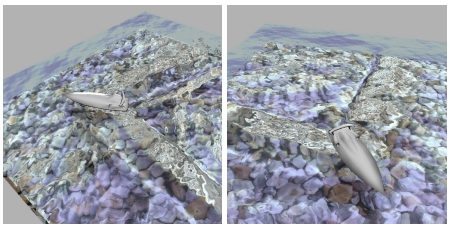

Figure 1: *The boat introduces some new waves at its tail as a result of the rigid body-fluid coupling.*

2.4. Particle coupling

Basically, from the fluid height (and its gradient and laplacian) we are able to detect which cells contain part of a breaking wave. We can compute the amount of fluid V_{tot} that should break, i.e., separate from the main fluid body, and initialize a number of particles enough to carry that amount of fluid with an appropriate velocity $\mathbf{v}$. This V_{tot}, as well as its momentum $\mathbf{v}_{xz}$, has to be subtracted from the LBMSW. We do this with the equilibrium distribution function of Equation 2 as

$$f_i = f_i - f_i^{eq}\left(V_{tot}/\Delta x^2, \mathbf{v}_{xz}\right), \tag{16}$$

where Δx is the size of a cell.

For the reintegration step, as the LBMSW has no explicit method to input vertical velocities, once the particle hits the surface again, we interpolate the fluid the particle carries among the cell's dfs. This interpolation is based on the terminal speed the particle could achieve, defined for spheres as

$$v_T = \sqrt{\frac{8rg}{3C_D}}, \tag{17}$$

where C_D is the drag coefficient. We normalize the particle's vertical speed with v_T and clamp the result to the range $[0,1]$, as $\chi = clamp(v_y/v_T, 0, 1)$.

Taking into account the previous consideration, and being V_p the fluid volume the particle carries, we update the cell's dfs as

$$f_0 = f_0 + (1-\chi) \cdot f_0^{eq\chi} \tag{18}$$

$$f_i = f_i + f_i^{eq}\left(\frac{V_p}{\Delta x^2} + \chi \cdot f_0^{eq\chi}, \mathbf{v}_{xz}\right), \tag{19}$$

$$\text{where } f_0^{eq\chi} = f_0^{eq}\left(V_p/\Delta x^2, \mathbf{v}_{xz}\right). \tag{20}$$

This way, the added volume is pushed from the cell's center to its neighbours with more energy, the faster the particle was falling. In addition, if particles are generated spontaneously, like in a heavy rain, the surface height would be

	Total	LBMSW	Solids	Particles
boat	0.82	0.35	0.47	0.00
wave 32k	15.28	0.36	0.00	14.92
wave 128k	25.71	0.36	0.00	25.35
wavegr 64k	18.79	0.39	0.00	18.40

Table 1: *Timings per frame in milliseconds. The number in the name of the example represents the number of particles, where $k = 2^{10}$. LBMSW includes the LBM simulation and the dry region tracking, Solids accounts for the two-way coupling of rigid bodies and Particles takes into account the detection, generation and reintegration phases.*

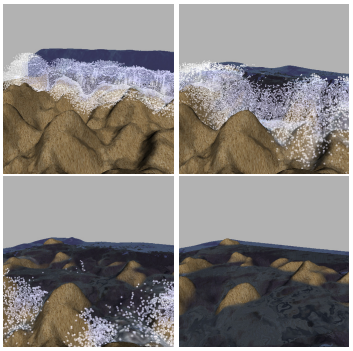

Figure 2: *A column of water breaks out over a noisy dry ground, generating a tsunami breaking wave.*

effectively raised when their volume becomes integrated as part of the fluid.

3. Results and Discussion

We have tested the method for various examples, with timings shown in Table 1. Our test system was an Intel Core2Duo E8400 with 4GB of RAM and a Nvidia GTX280 running Ubuntu 11.10. The size of the grid for all examples is set to 128x128 and the timestep is fixed to 16ms. To get a better timing, the particles were deactivated in the boat example, as were the obstacle coupling in the wave examples. The wavegr example is shown in Figure 2, the wave example being similar but without dry regions.

Although a direct comparison with [CM10] would not be totally fair because of the hardware difference, we think that our LBM-based method is a great alternative up to the challenge.

Performance wise, the number of particles used is more restrictive, in contrast to the size of the grid of the LBM, and may turn the simulation not interactive at all. In order to improve the situation, level-of-detail techniques could be applied to both simulations, and is part of the future work. Also, porting the rigid body simulation to GPU and applying better breaking conditions for particle generation are in our future plans.

Nevertheless, we have shown that higher-detail fluid simulations are possible with a coupling of LBMSW with particle systems. The particle system is not restricted to a particular type; the particle's simulation is independent, while the coupling in the generation and reintegration phases is maintained. Following this direction, in the future we also want to change the ballistic particle system for an SPH one.

Acknowledgements

With the support of the Research Project TIN2010-20590-C02-01 of the Spanish Government.

References

[Bri08] BRIDSON R.: *Fluid Simulation for Computer Graphics*. AK Peters, 2008. 1

[CM10] CHENTANEZ N., MÜLLER M.: Real-time simulation of large bodies of water with small scale details. In *Proc. Symposium on Computer Animation (SCA)* (2010), pp. 197–206. 1, 2, 4

[Cor07] CORDS H.: Mode-splitting for highly detailed, interactive liquid simulation. In *Proc. GRAPHITE '07* (2007), pp. 265–272. 1

[GRGT10] GEVELER M., RIBBROCK D., GÖDDEKE D., TUREK S.: Lattice-boltzmann simulation of the shallow-water equations with fluid-structure interaction on multi- and manycore processors. In *Facing the multicore-challenge*. Springer-Verlag, 2010, pp. 92–104. 2

[HL97] HE X., LUO L. S.: Theory of the lattice Boltzmann method: From the Boltzmann equation to the lattice Boltzmann equation. *Phys. Rev. E 56* (Dec 1997), 6811–6817. 3

[LvdP02] LAYTON A. T., VAN DE PANNE M.: A numerically efficient and stable algorithm for animating water waves. *The Visual Computer 18* (2002), 41–53. 1

[Sal99] SALMON R.: The lattice boltzmann method as a basis for ocean circulation modeling. *Journal of Marine Research 57*, 3 (1999), 503–535. 2

[SBC*11] SOLENTHALER B., BUCHER P., CHENTANEZ N., MÜLLER M., GROSS M.: SPH Based Shallow Water Simulation. In *Proc. VRIPHYS 11* (2011), pp. 39–46. 1

[Sol09] SOLENTHALER B.: *Incompressible Fluis Simulation and Advanced Surface Handling with SPH*. PhD thesis, Institut für Informatik, Universität Zürich, 2009. 1

[Tes01] TESSENDORF J.: Simulating ocean water. In *Simulating Nature: Realistic and Interactive Techniques*. SIGGRAPH Course Notes, 2001. 1

[Thü07] THÜREY N.: *Physically based Animation of Free Surface Flows with the Lattice Boltzmann Method*. PhD thesis, Universität Erlangen-Nürnberg, Mar 2007. 1, 2

[TMFSG07] THÜREY N., MÜLLER-FISCHER M., SCHIRM S., GROSS M.: Real-time breaking waves for shallow water simulations. In *Proc. Pacific Graphics'07* (2007), pp. 39 –46. 1

[TSB07] THÖMMES G., SEAÏD M., BANDA M. K.: Lattice boltzmann methods for shallow water flow applications. *International Journal for Numerical Methods in Fluids 55*, 7 (2007), 673–692. 1

[YHK07] YUKSEL C., HOUSE D. H., KEYSER J.: Wave particles. *ACM Trans. Graph. 26*, 3 (jul 2007). 1, 2

[Zho11] ZHOU J.: Enhancement of the LABSWE for shallow water flows. *Journal of Computational Physics 230*, 2 (2011), 394 – 401. 2