Individual Time Stepping for SPH Fluids

Liangliang He, Xiaojuan Ban†, Xu Liu and Xiaokun Wang

School of Computer & Communication Engineering, University of Science & Technology Beijing, China

Abstract

We present a novel adaptive stepping scheme for SPH fluids, in which particles have their own time-steps determined from local conditions, e.g. Courant condition. These individual time-steps are constrained for global convergence and stability. Fluid particles are then updated asynchronously. The approach naturally allocates computing resources to visually complex regions, e.g. regions with intense collisions, thereby reducing the overall computational time. The experiments show that our approach is more efficient than the standard method and the method with globally adaptive time steps, especially in highly dynamic scenes.

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

1. Introduction

Physics-based fluid simulations are widely used, for example, in movies, virtual realities, and even in computer games, despite the difficulty due to the complexity of fluid behavior. Lagrangian methods based on Smoothed Particle Hydrodynamics (SPH) [Mon92], which have been gaining increased interest in graphics community [MCG03, BT07, SP09, HLL*12, ICS*14], are able to successfully simulate complex scenes with versatile effects [IOS*14]. To produce appealing visual results with small-scale details, SPH fluids demand a high discretization resolution, i.e. large number of particles. The computational expenses of simulating millions of particles are, however, too large for practical use [IABT11]. To cope with the increasing demand for more detailed fluids, adaptive methods, that adapt either the spatial resolution or the sampling in time, have been proposed. They follow the idea to allocate computing resources to regions with complex flow behavior.

Space adaptive methods [DC99, APKG07, SG11] dynamically exchange particle sets. Large particles are divided into smaller ones when a high discretization resolution is needed, and vice versa. In contrast to homogeneous fluids, these adaptively sampled particle fluids use fewer particles to produce the similar details, thus are more efficient. However, difficulties exist in reproducing quantity when refining particles [OK12]. Furthermore, while neighborhood searching is usually the bottleneck [IABT11], non-uniform smoothing radii make it slower since hierarchical data structures such as kd-trees have to be used [IOS*14].

As an alternative to adaptive spatial discretization, time adaptive methods adapt the sampling distance in time. Globally adaptive time stepping methods [Mon92, DC96, IAGT10] use a single time step adjusted in each simulation step with respect to the CFL condition [Mon92] for all particles. Since globally adaptive time steps update all particles according to the particle that has the current smallest time step, it is not the best efficient. Locally adaptive time stepping methods [DC99, GB14, GP11] use different time steps for particles. In [DC99], each particle evaluates forces only when needed, according to its current individual time step determined from individual stability conditions.

In this paper, we adopt the idea of [DC99] for weakly compressible SPH (WCSPH) [Mon94, BT07]. To make the simulation of stiff fluids stable, small time steps are spread to neighbor particles in order to respond to strong shocks. The proposed simulation is very similar to the globally adaptive time stepping methods, except performing neighborhood searching and forces evaluating for only fraction of all particles in every simulation step. The position, velocity and density of the particle that currently does not need force evaluations are interpolated. We show that the presented simulation algorithm naturally allocates computing resources to visually complex regions with intense collisions (Fig. 1), thus reduces the computational time.

† banxj@ustb.edu.cn

DOI: 10.2312/egsh.20151010

2. Method

2.1. Basic SPH

In SPH, the fluid to be simulated are represented by a set of particles and a set of governing equations. Each particle i has several attributes, namely

m_i	Mass
ρ_i	Density
p_i	Pressure
$\mathbf{x}_i$	Position
$\mathbf{v}_i$	Velocity

The relations between these attributes are described by governing equations. For WCSPH [Mon94, BT07], the governing equations are

$$\rho_i = \sum_j m_j W_{ij} \tag{1}$$

$$p_i = \frac{\rho_0 c_S^2}{\gamma}\left(\left(\frac{\rho_i}{\rho_0}\right)^\gamma - 1\right) \tag{2}$$

$$\frac{d\mathbf{x}_i}{dt} = \mathbf{v}_i \tag{3}$$

$$\frac{d\mathbf{v}_i}{dt} = -\sum_j m_j \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2}\right)\nabla W_{ij} + \mathbf{g} \tag{4}$$

where $W_{ij} = W\left(\mathbf{x}_i - \mathbf{x}_j, h\right)$ the kernel function with h the smooth radii, j iterates all the neighbors, $\rho_0 = 1000, c_S, \gamma = 7$ the constants, ∇ the gradient operator and $\mathbf{g}$ the external force. For the force term (i.e. Eq. (4)), other forces, e.g. viscous force and surface tension force, can be included. In our implementation, we use the viscous force and kernels presented in [Mon92] and rigid-fluid coupling methods of [AIA*12].

Algorithm 1 Basic SPH

```
while animating do
    for all particle i do
        find neighbors j
    for all particle i do
        compute ρ_i, p_i (e.g. Eq. (1), (2))
    for all particle i do
        compute forces (e.g. Eq. (4))
    for all particle i do
        v_i(t+Δt) = v_i(t) + Δt dv_i/dt (t)
        x_i(t+Δt) = x_i(t) + Δt v_i(t+Δt)
    t = t + Δt
```

The density derivative is given by

$$\frac{d\rho_i}{dt} = \sum_j m_j \mathbf{v}_{ij} \nabla W_{ij} \tag{5}$$

where $\mathbf{v}_{ij} = \mathbf{v}_i - \mathbf{v}_j$. We use density derivative to interpolate density and then the pressure.

Alg. 1 shows the basic SPH simulation algorithm, in which the semi-implicit Euler numerical integration scheme is used.

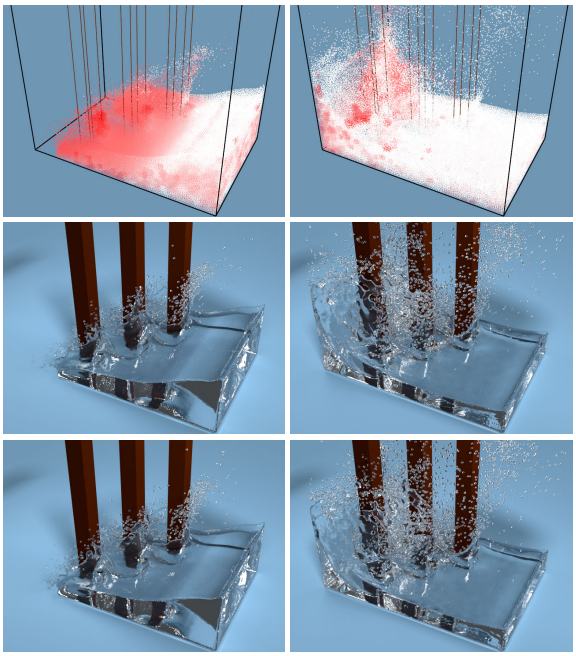

Figure 1: *Breaking dam with obstacles simulated using presented method. Top: Particles colored according to individual step (red for small dts and white for large dts). Middle: Corresponding frames. Bottom: Corresponding frames simulated using globally adaptive stepping method.*

2.2. Globally adaptive time stepping

The time step of SPH fluids must be constrained for numerical stability and convergence. The Courant-Friedrich-Levy (CFL) condition

$$\Delta t_{CFL} \leq \lambda_v \left(\frac{h}{v_{max}}\right) \tag{6}$$

ensure that numerical propagation speed is higher than physical propagation speed, where $v_{max} = \max_i\left(\|\mathbf{v}_i\|\right)$ is the maximum of the particle velocities, coefficient $\lambda_v < 1$. In addition, particle forces have to be considered

$$\Delta t_f \leq \lambda_f \left(\sqrt{\frac{h}{f_{max}}}\right) \tag{7}$$

where $f_{max} = \max_i\left(\left\|\frac{d\mathbf{v}_i}{dt}\right\|\right)$ denotes the maximum force per unit mass of particles, $\lambda_f < 1$. In [IAGT10], $\lambda_v = 0.4, \lambda_f = 0.25$ are used for PCISPH. Since WCSPH uses a stiff EOS which restricts the time step [SP09, IABT11], we use $\lambda_v = 0.1, \lambda_f = 0.05$ for globally adaptive time stepping method.

Instead of using a constant time step in Alg. 1, we can now adjust it dynamically by

$$\Delta t = \min\left(\Delta t_{CFL}, \Delta t_f\right). \tag{8}$$

2.3. Individual Time Stepping

In SPH fluids, each particle only interacts with its neighbors. The converge of the simulation of each particle is determined locally. Allowing particles to have different time steps is more efficient than using a globally restricted time step for all particles . Our individual time stepping method computes time step for each particle i according to

$$\Delta t_i = \min_j \left(\lambda_v \frac{h}{\|\mathbf{v}_j\|}, \lambda_f \sqrt{\frac{h}{\|d\mathbf{v}_j/dt\|}}\right) \tag{9}$$

where j iterates over all neighbors. In contrast to [DC99], we take the neighbors into account, which make the algorithm stable for stiff fluids. The proposed method, however, requires small coefficients due to the asynchrony. We use $\lambda_v = 0.05, \lambda_f = 0.025$, i.e. half of that of globally adaptive time stepping method. Fortunately, this restriction will be counteracted by the fact that the proposed method updates only a small fraction of all particles in every time step.

Our individual time stepping algorithm is illustrated in Alg. 2. In our algorithm, each particle i maintains a few additional variables

$\frac{d\rho_i}{dt}$	Density derivative
Δt_i	Time step
Δt_i^{raw}	Raw time step
t_i^{last}	Last updated time

The algorithm has the same skeleton as the globally adaptive time stepping methods do. The key is that for only the *active* particles neighborhood searching and forces evaluating are performed. Particle i is *active* if $t_i^{last} + \Delta t_i < t_{sim}$. In line 16, small time steps are spread to neighbors, which lets the simulation respond to shocks. In line 20 and 21, velocities and positions are interpolated. We make some observations. First, line 9 and line 16 are needed to be performed only for neighbors of active particles. Second, for inactive particles, reusing the last neighbor lists leads to small errors which is negligible.

Algorithm 2 Individual time stepping for SPH

1: **while** animating **do**
2: select *active*
3: **for all** *active particle i* **do**
4: find neighbors j
5: **for all** *particle i* **do**
6: **if** *active* **then**
7: compute ρ_i, p_i (e.g. Eq. (1), (2))
8: **else**
9: interpolate ρ_i, p_i using $\frac{d\rho_i}{dt}$
10: **for all** *active particle i* **do**
11: compute forces (e.g. Eq. (4))
12: compute $\frac{d\rho_i}{dt}$ (e.g. Eq. (5))
13: $\Delta t_i^{raw} = \min\left(\lambda_v \frac{h}{\|\mathbf{v}_i\|}, \lambda_f \sqrt{\frac{h}{\|d\mathbf{v}_i/dt\|}}\right)$
14: $t_i^{last} = t_{sim}$
15: **for all** *particle i* **do**
16: $\Delta t_i = \min_j \left(\Delta t_j^{raw}\right)$
17: $\Delta t_{sim} = \min_i \left(\Delta t_i\right)$
18: **for all** *particle i* **do**
19: $\Delta t = t_{sim} + \Delta t_{sim} - t_i^{last}$
20: $\mathbf{v}_i\left(t_i^{last} + \Delta t\right) = \mathbf{v}_i\left(t_i^{last}\right) + \Delta t \frac{d\mathbf{v}_i}{dt}\left(t_i^{last}\right)$
21: $\mathbf{x}_i\left(t_i^{last} + \Delta t\right) = \mathbf{x}_i\left(t_i^{last}\right) + \Delta t \mathbf{v}_i\left(t_i^{last} + \Delta t\right)$
22: $t_{sim} = t_{sim} + \Delta t_{sim}$

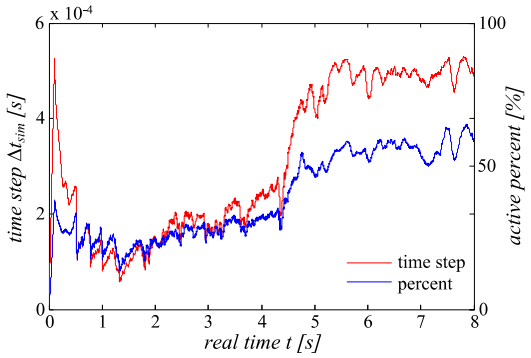

Figure 2: *Time step Δt_{sim} and active percent evolution for breaking dam scene.*

3. Results

The proposed simulation algorithm is compared with basic WCSPH method and also the global time stepping method. All timings are given for an Intel 3.50 GHz CPU with 4 cores. The simulation software is parallelized with OpenMP. For the videos, we reconstruct fluid surface using anisotropic kernels [YT13]. Images were rendered with Blender. For WCSPH, the density fluctuation is set to 0.01.

Our method naturally allocates computing resources to complex regions with intense collisions. This is shown in Fig. 1 where the individual time steps are color-coded: the particles with collisions to the obstacles have less time steps and cost more computational time. A side-by-side comparison of the tested methods is also given in Fig. 1,which shows that the visual result of the proposed method is in good agreement with the standard method. The evolution of the time step and of the percent of active particles are shown in Fig. 2. In Fig. 3, a high dynamic scene with 1.2M particles is simulated using proposed method. The performance measurements and simulation data are summarized in Tab. 1.

scene	method	#p	real time	total comp. time	avg. Δt_{sim} (avg. active pct.)	speed up
breaking dam	constant step	153K	8s	175min	0.11ms	-
	globally adaptive	153K	8s	41min	0.46ms	-
	individual stepping	153K	8s	27min	0.23ms (31%)	1.5 (6.4)
stirring pool	globally adaptive	1.2M	10s	32.1h	0.12ms	-
	individual stepping	1.2M	10s	14.8h	0.064ms (29%)	2.1

Table 1: *Comparison of individual stepping method and standard SPH methods.*

Figure 3: *Stirring pool scene. A hand interacts with a water pool simulated by 1.2M particles.*

4. Conclusions

We proposed an efficient individual time stepping method for SPH fluids. Our method updates neighbors and forces of particles only when needed, which naturally allocates computing resources to complex regions. Future work would be extend the proposed method to IISPH [ICS*14].

This work was supported by National Natural Science Foundation of China (No. 61272357, 61300074, 61174103).

References

[AIA*12] AKINCI N., IHMSEN M., AKINCI G., SOLENTHALER B., TESCHNER M.: Versatile rigid-fluid coupling for incompressible SPH. *ACM Transactions on Graphics (Proceedings SIGGRAPH) 31*, 4 (2012), 62:1–62:8. 2

[APKG07] ADAMS B., PAULY M., KEISER R., GUIBAS L. J.: Adaptively sampled particle fluids. *ACM Transactions on Graphics (Proceedings SIGGRAPH) 26*, 3 (2007), 48. 1

[BT07] BECKER M., TESCHNER M.: Weakly compressible SPH for free surface flows. In *ACM SIGGRAPH/Eurographics symposium on Computer animation* (2007), pp. 209–217. 1, 2

[DC96] DESBRUN M., CANI M.-P.: Smoothed particles: A new paradigm for animating highly deformable bodies. In *Eurographics Workshop on Computer Animation and Simulation (EGCAS)* (1996), Springer-Verlag, pp. 61–67. 1

[DC99] DESBRUN M., CANI M.-P.: *Space-time adaptive simulation of highly deformable substances*. Tech. Rep. 3829, INRIA, 1999. 1, 3

[GB14] GOSWAMI P., BATTY C.: Regional time stepping for SPH. In *Eurographics* (2014), The Eurographics Association, pp. 45–48. 1

[GP11] GOSWAMI P., PAJAROLA R.: Time adaptive approximate SPH. In *Workshop on Virtual Reality Interaction and Physical Simulation* (2011), The Eurographics Association, pp. 19–28. 1

[HLL*12] HE X., LIU N., LI S., WANG H., WANG G.: Local Poisson SPH for viscous incompressible fluids. *Computer Graphics Forum 31*, 6 (2012), 1948–1958. 1

[IABT11] IHMSEN M., AKINCI N., BECKER M., TESCHNER M.: A parallel SPH implementation on multi-core CPUs. *Computer Graphics Forum 30*, 1 (2011), 99–112. 1, 2

[IAGT10] IHMSEN M., AKINCI N., GISSLER M., TESCHNER M.: Boundary handling and adaptive time-stepping for PCISPH. In *Workshop on Virtual Reality Interaction and Physical Simulation* (2010), The Eurographics Association, pp. 79–88. 1, 2

[ICS*14] IHMSEN M., CORNELIS J., SOLENTHALER B., HORVATH C., TESCHNER M.: Implicit incompressible SPH. *IEEE Transactions on Visualization and Computer Graphics 20*, 3 (2014), 426–435. 1, 4

[IOS*14] IHMSEN M., ORTHMANN J., SOLENTHALER B., KOLB A., TESCHNER M.: SPH fluids in computer graphics. In *State-of-the-Art Report Eurographics* (2014), pp. 21–42. 1

[MCG03] MÜLLER M., CHARYPAR D., GROSS M.: Particle-based fluid simulation for interactive applications. In *ACM SIGGRAPH/Eurographics symposium on Computer animation* (2003), pp. 154–159. 1

[Mon92] MONAGHAN J. J.: Smoothed particle hydrodynamics. *Annual review of astronomy and astrophysics 30* (1992), 543–574. 1, 2

[Mon94] MONAGHAN J. J.: Simulating free surface flows with SPH. *Journal of computational physics 110*, 2 (1994), 399–406. 1, 2

[OK12] ORTHMANN J., KOLB A.: Temporal blending for adaptive SPH. *Computer Graphics Forum 31*, 8 (2012), 2436–2449. 1

[SG11] SOLENTHALER B., GROSS M.: Two-scale particle simulation. *ACM Transactions on Graphics (Proceedings SIGGRAPH) 30*, 4 (2011), 81:1–81:8. 1

[SP09] SOLENTHALER B., PAJAROLA R.: Predictive-corrective incompressible SPH. *ACM transactions on graphics (Proceedings SIGGRAPH) 28*, 3 (2009), 40:1–40:6. 1, 2

[YT13] YU J., TURK G.: Reconstructing surfaces of particle-based fluids using anisotropic kernels. *ACM Transactions on Graphics (Proceedings SIGGRAPH) 32*, 1 (2013), 5:1–5:12. 3