DOI: 10.1111/j.1467-8659.2012.03196.x
Pacific Graphics 2012
C. Bregler, P. Sander, and M. Wimmer
(Guest Editors)

Volume 31 (2012), Number 7

Realtime Two-Way Coupling of Meshless Fluids and Nonlinear FEM

Lipeng Yang[1], Shuai Li[1], Aimin Hao[1], and Hong Qin[2]

[1]State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, China, lishuaiouc@126.com
[2] Stony Brook University, Stony Brook, USA, qin@cs.stonybrook.edu

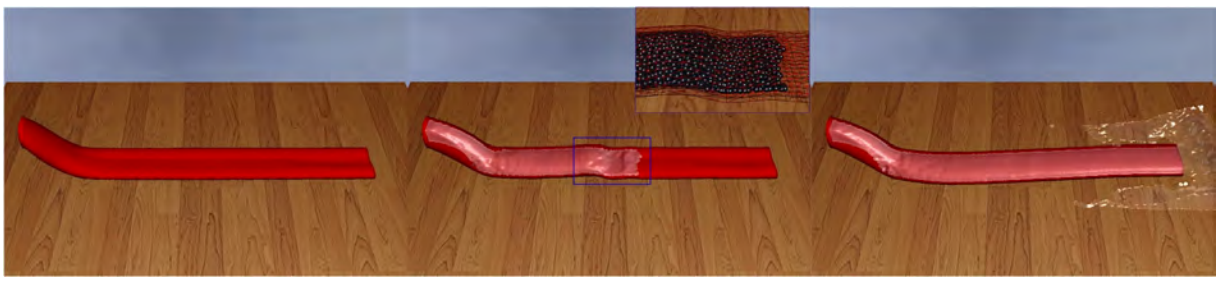

Figure 1: *The coupling of hose and fluids.*

Abstract

In this paper, we present a novel method to couple Smoothed Particle Hydrodynamics (SPH) and nonlinear FEM to animate the interaction of fluids and deformable solids in real time. To accurately model the coupling, we generate proxy particles over the boundary of deformable solids to facilitate the interaction with fluid particles, and develop an efficient method to distribute the coupling forces of proxy particles to FEM nodal points. Specifically, we employ the Total Lagrangian Explicit Dynamics (TLED) finite element algorithm for nonlinear FEM because of many of its attractive properties such as supporting massive parallelism, avoiding dynamic update of stiffness matrix computation, and efficient solver. Based on a predictor-corrector scheme for both velocity and position, different normal and tangential conditions can be realized even for shell-like thin solids. Our coupling method is entirely implemented on modern GPUs using CUDA. We demonstrate the advantage of our two-way coupling method in computer animation via various virtual scenarios.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Physically based modeling

1. Introduction

With the rapid development of computer simulation techniques for fluids and deformable solids during the last two decades, many works in recent years started to turn their foci to physics-based fluid-solid interaction and its complicated modeling to demonstrate complex natural phenomena that we take for granted in real-world settings. Especially, in certain applications such as virtual surgery that involves blood vessels and their intervention, both fluids (e.g., blood) and deformable solids (e.g., blood vessels) must be simulated accurately and efficiently while accommodating their high-fidelity interaction. In this paper, we focus on modeling the two-way coupling of fluids and deformable solids in real time.

Many methods have been proposed to compute the coupling force, for example, penalty force method [MST*04], direct forcing method [BTT09], etc. The coupling forces obtained from the penalty force method are mainly determined

by the distance between fluid particles and solids, it is relatively simple to implement, but has many drawbacks in stiffness coupling and energy conserving. Direct forcing method has been used to couple fluids with rigid body, but it is hard to be directly used for mesh-based deformable solids. A better way to handle the interface condition between fluids and solids is to adopt the ghost particle method [Fed02], of which a ghost particle with the same mass, pressure, viscosity, density, and mirrored velocity is dynamically generated to keep the fluid particle away from penetrating through the solid boundary when a fluid particle is close enough to the solid boundary. However, the ghost particle method is not suitable to couple with deformable solids, especially when the interface is complex, and it is hard to generate ghost particles properly [BTT09].

Despite the recent success of fluid-solid coupling simulation [BTT09, AIA*12], especially for the interaction among fluids and deformable objects [MST*04, ACF11], certain difficulties still prevail and need to be resolved for high-fidelity geometric representation, accurate physical modeling, and accelerated computing. Specific challenges are documented as follows.

First, for the purpose of simplified geometry, most of existing approaches prefer particle-based models, so that both solids and fluids can be processed in a unified framework while avoiding tracking the accurate interface between solids and fluids. When the deformable solids need to be more accurately represented by polyhedral meshes, existing methods cannot be trivially transplanted to simulate the high-fidelity fluid-solid interaction.

Second, from the standpoint of physics-based modeling, although SPH-like meshless methods have natural advantages in simulating fluid behavior and solid deformation while accommodating topological changes, they are far from adequate and convenient to represent heterogeneous material properties according to different physical laws. Especially for accurate nonlinear large deformation subject to various material properties, mesh-based Finite Element Method (FEM) is frequently employed. Thus, how to respectively take advantage of meshless and mesh-based methods when seeking a hybrid approach towards effective fluid-solid coupling deserves our immediate research efforts.

Third, when focusing on the computation of popular SPH methods, we have seen its rapid deployment on modern GPUs. Yet, in order to reduce the heavy computational cost of the SPH and FEM coupled simulation procedure, a unified and robust framework together with its exploitation of CUDA-based parallelly accelerated computation is urgently needed.

In this paper, we present a novel method to couple SPH-based fluids and nonlinear FEM-based deformable solids in real time. Specifically, the salient contributions of this paper include:

- We propose a method to dynamically generate proxy particles over the fluid-solid interface, which naturally serve as the local geometric/physical representation of solids with a unique goal to accurately interact with fluid particles.
- We seamlessly combine the direct forcing method with a predictor-corrector scheme to compute the coupling forces between fluid particles and proxy particles, of which different boundary conditions and non-penetration robustness can be well guaranteed.
- We design the CUDA-based parallel algorithms for the entire simulation pipeline, which can achieve realtime fluid-solid coupling even for complex scenarios.

2. Related Work

Relevant to the central theme of this paper, we now briefly review previous works in three categories: SPH, FEM, and coupling.

SPH. As a meshless Lagrangian method, Smoothed Particle Hydrodynamics (SPH) has been widely used in computer graphics to simulate various kinds of liquid phenomena. Its initial formulation for fluid dynamics originated from [Mon92]. Then, Müller derived interparticle forces from SPH and the Navier-Stokes equations to interactively simulate water with free surfaces in [MCG03]. However, it is difficult to model strong incompressible fluid. To solve this problem, [BT07] proposed a Weakly Compressible SPH (WCSPH) method by introducing Tait equation. Alternatively, [SP09] proposed a Predictive-Corrective Incompressible SPH (PCISPH) method, which corrects pressure of fluid particles iteratively to provide smooth density and pressure distributions. [IABT11] presented an efficient parallel framework to conduct SPH-based fluid simulation using multi-core CPUs and discussed the performance of PCISPH and WCSPH, which resulted in impressive results for low-viscous incompressible fluids with up to 12 million particles. Meanwhile, some SPH-based methods were also proposed to simulate deformable solids. For example, [BIT09] proposed a co-rotational SPH method by introducing the co-rotational idea. [LZLW11] further improved its stability by using anisotropic SPH kernels. In addition, a variety of methods were proposed to accelerate the computing, including adaptively sampling based SPH [APKG07, SG11], and GPU based SPH [HKK07, GSSP10, KE12].

FEM. FEM is a popular method to numerically solve physical models. However, for large models with nonlinear constitutive behavior, its heavy computational cost usually cannot be afforded. To accelerate computation, TLED method was proposed by Miller et al. [MJLW07], and then it was implemented on GPUs by [TCO08, JWM10] to further improve its efficiency. Besides, [TCC*09] expanded TLED to model the anisotropic viscoelasticity of soft tissue. Generally speaking, TLED has many attractive properties. First, the spatial derivatives can be precomputed in TLED, since it-

s measurement for deformation is only referred to the initial configuration of solid. Second, it avoids the implicit solving of linear system by employing explicit time integration. Third, the stress is directly evaluated from strain, thus it is easier to incorporate the underlying constitutive models while requiring no tangent matrix computation.

Coupling. Given SPH based fluid simulation, a variety of methods were proposed to model fluid-fluid and fluid-solid coupling. [MSKG05] proposed a technique to model the interaction of multiple SPH fluids that have different physical properties and dynamic phases. To simulate melting and solidification phenomena with SPH, a unified framework was proposed by [SSP07]. The framework was then extended by [LD08] to couple fluids with deformable thin shells. [LAD08] proposed a novel method to simulate the porous material and its two-way coupling with fluids, of which the material properties can be changed in accordance with fluid absorption and emission. [BTT09] proposed an interacting algorithm for boundary velocity and position based on a predictor-corrector scheme, however, it can only support the two-way coupling of fluid and rigid bodies, and usually gives rise to the problem of particle stacking around boundaries. Then, [IAGT10] solved the stacking problem by employing PCISPH to iteratively correct the pressure and the particle position. But in their method both fluids and the boundary were still represented with particles. Recently, [IUDN10] proposed a particle-based model to simulate the melting and freezing of ice objects, and achieved interactive simulation with the help of CUDA. Additionally, [QPN*10] proposed to couple blood flow (SPH) and vessel wall (mass-spring model) by employing an improved repulsive boundary condition. [AIA*12] proposed a pressure-based coupling method to simulate the interaction of arbitrarily-complicated rigid objects and fluids, which employed boundary particles to represent rigid objects and effectively avoided the problem due to high pressure ratios. Therefore, it can afford large time steps. However, the method heavily relied on pre-sampled objects and was very hard to be generalized to handle solids with large deformation.

When FEM is adopted to simulate complex deformable solids, the aforementioned coupling methods tend to lose its competitiveness. To our best knowledge, only a few works achieved limited success on this subject. For example, [MST*04] proposed to place virtual boundary particles on the solid surface according to Gaussian quadrature rules, and employed Lennard-Jones-like force to model repulsion and adhesion. However, it is hard to simultaneously enforce non-penetration and avoid implausible elastic coupling, since the penalty-based coupling forces need to introduce stiffness to coupling. [SSIF07] proposed a framework to embed arbitrary sample points dynamically into solids to handle collision, whose basic idea is somehow similar to our method in spirit. [ACF11] employed an implicit FEM solver to simulate deformable solids on GPUs, and achieved interactive coupling simulation by extracting an implicit fluid surface to define contacting constraints.

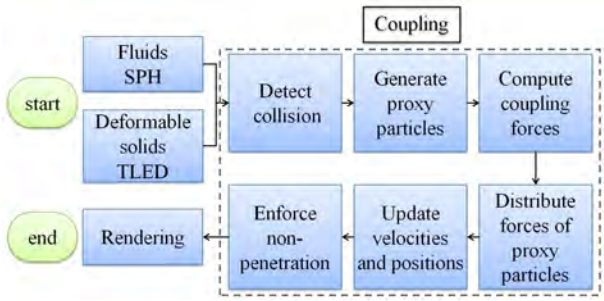

Figure 2: *The pipeline of each simulation cycle.*

3. Algorithmic Overview

3.1. Algorithm Architecture

Fig. 2 shows the algorithmic flow of our method. In each simulation cycle, it updates the new position and velocity based on the results of last simulation cycle, and then employs a screen space based method to track the fluid surface and render the current simulation results in real time. We detail the algorithm as follows:

- **Simulate the behaviors of fluids and solids.** Respectively employ SPH and TLED models to compute velocities and positions of fluid particles and solid points without considering their interaction.
- **Detect collision.** Detect collision between solid surface polygons and fluid particles.
- **Generate proxy particles.** Generate proxy particles according to the collision positions and the physical properties of solid. The velocities of proxy particles can be obtained via interpolation in the local region of a solid.
- **Compute coupling forces.** Compute the coupling forces between proxy particles and fluid particles by enforcing energy conservation into the direct forcing method.
- **Distribute forces of proxy particles.** Distribute the coupling forces of proxy particles to the mesh vertices of solid.
- **Update velocities and positions.** Update velocities and positions of fluid particles and solid vertices driven by the updated coupling forces.
- **Enforce non-penetration.** Detect fluid-solid collision again based on the updated positions, and correct the positions of collided fluid particles to enforce the non-penetration constraints.
- **Rendering.** Render the results in real time by optimizing the screen space based fluid surface tracking method proposed by [VdLGS09].

3.2. SPH Model

Fluid behavior is controlled by the Navier-Stokes equations, which comprise two equations. The first equation is momentum equation, which can be formulated by the fundamental Newton's second law:

$$\rho \frac{d\mathbf{v}}{dt} = -\nabla P + \mu \nabla^2 \mathbf{v} + \mathbf{f}, \tag{1}$$

where ρ is the fluid's density, $\mathbf{v}$ is the velocity, P is the fluid's pressure, μ is its viscosity coefficient, and $\mathbf{f}$ represents the external force.

The second equation is continuity equation, which can impose mass conservation for fluid.

$$\frac{d\rho}{dt} + \rho(\nabla \cdot \mathbf{v}) = 0. \tag{2}$$

In principle, SPH is a meshless Lagrangian method, it uses particles to discretize the continuum. The particles carry individual properties, which should be smoothed in the volume surrounding each particle node. According to SPH, a scalar quantity A at location $\mathbf{r}$ is estimated by a weighted sum of all particles in its local supporting domain:

$$A_s(\mathbf{r}) = \sum_j m_j \frac{A_j}{\rho_j} W(\mathbf{r} - \mathbf{r}_j, h), \tag{3}$$

where m_j is the mass of particle j, $\mathbf{r}_j$ is its position, ρ_j is the density.

The function $W(\mathbf{r},h)$ is the smoothing kernel with supporting radius h. In the interest of robustness, the kernel is preferred to be smooth and normalized. Here we adopt the kernels proposed by [MCG03].

The acceleration of all particles can be computed by solving Eq.1 and Eq.2 at each time step for every particle using the SPH formulations. Then we employ Leap-Frog scheme to update their velocities and positions.

To avoid strong compressibility, we take the weakly compressible pressure formulation into account, which can guarantee a small density ratio between the current density ρ and the initial density ρ_0. The formulation originated from the Tait equation [BT07]:

$$P = \frac{\rho_0 c_s^2}{\gamma} \left(\left(\frac{\rho}{\rho_0} \right)^{\gamma} - 1 \right), \tag{4}$$

with $\gamma = 7$, and c_s is the speed of sound in the fluid.

3.3. TLED Model

TLED adopts discretized equations of equilibrium for large deformation as

$$\mathbf{M}\ddot{\mathbf{U}} + \mathbf{D}\dot{\mathbf{U}} + \mathbf{K}(\mathbf{U})\mathbf{U} = \mathbf{R}, \tag{5}$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{U}$ represents an array of global nodal displacements, $\mathbf{D}$ is the damping matrix, $\mathbf{K}(\mathbf{U})$ is the stiffness matrix nonlinearly dependent on $\mathbf{U}$, $\mathbf{R}$ represents an array of external forces, and $\mathbf{M}$ and $\mathbf{D}$ are two constants.

Now we set a stage to briefly describe the GPU-based TLED model, a more detailed description is beyond the technical scope of this paper, please refer to [MJLW07, TCO08] for more technical details. In each time step:

1. Invoke a CUDA thread for each nodal point, assign displacements, forces and constraints to nodal points.
2. Invoke a CUDA thread for each element, compute the following variables in turn: deformation gradient $\mathbf{F}$, strain-displacement matrix $\mathbf{B}_L$, second Piola-Kirchhoff stress $\mathbf{S}$, element nodal forces, and then sum these forces to form the total nodal forces $\mathbf{f}$.
3. Invoke a CUDA thread for each nodal point, and update displacement $\mathbf{u}$ using the central difference method.

4. Coupling Simulation

4.1. Proxy Particle Generation

After collision detection, we can get the contact polygons and particles. In fact, it is difficult to directly compute the coupling forces, especially when a fluid particle collides with more than one polygon. To facilitate the coupling force computing, we generate proxy particles dynamically over the solid surface to represent the local region of the solid. Then the coupling interaction is simplified to sphere-sphere collision, this way the forces can be computed efficiently.

We shall first explain how to compute the position, mass, velocity of the proxy particle. To afford different boundary conditions, we also need to compute the normal of the proxy particle, which can also be taken as the normal of the boundary at the contact position.

As illustrated in Fig.3(b), when the FEM mesh collides with a fluid particle p_i, a proxy particle p_p is created over the boundary with predefined radius r_0 of the fluid particle. And the mass of the proxy particle m_p can be computed according to its volume and the density of the solid.

To define the position of the proxy particle, we need to estimate the normal of the boundary at the contact position first. We compute the distance d_{ij} from fluid particle p_i to the *contacted triangle* Tri_j. Then we use d_{ij} to construct a weighting function (as shown in Eq.7). By taking the normal $\mathbf{n}_j$ of the collided triangle Tri_j into account, we can get the normal $\mathbf{n}_c$ as

$$\mathbf{n}_c = \frac{\sum_j w_{ij}\mathbf{n}_j}{||\sum_j w_{ij}\mathbf{n}_j||}, \tag{6}$$

$$w_{ij} = max\left(0, \frac{r_0 - d_{ij}}{r_0}\right). \tag{7}$$

As shown in Fig. 3(b), we place the proxy particle to be in front of the fluid particle along the direction $-\mathbf{n}_c$. Then we

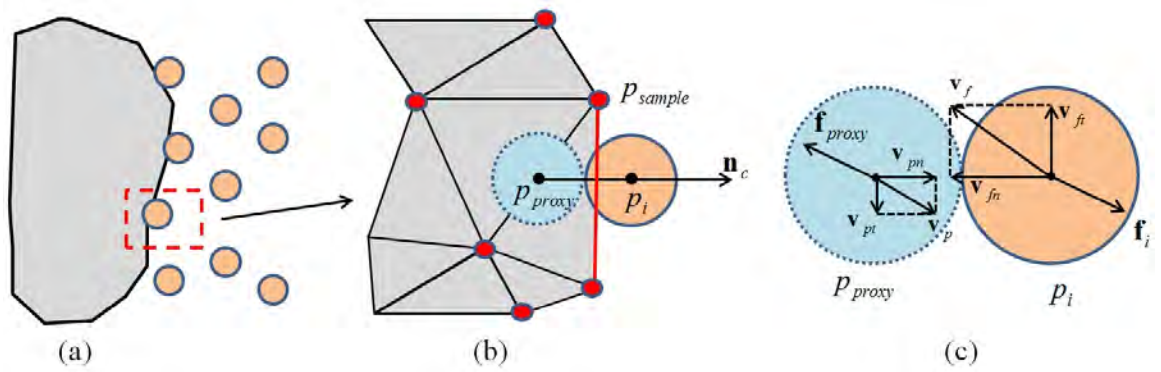

Figure 3: *Proxy particle generation and coupling force computing. (a) shows the collision procedure between fluids and deformable solids. (b) shows a fluid particle p_i that collides with solid mesh. The edge in red represents the collided triangle. The red points are sampled points. (c) shows the velocities before coupling, and the obtained coupling forces.*

search the nodal points in the neighboring region of the contacted triangles, and denote them as *sample points* p_j. Thus the velocity of the proxy particle can be obtained by interpolating the corresponding properties of the sampled points. When the coupling is done, the force will be distributed to the sampled points along the opposite direction. For the purpose of simplicity, we adopt $w_j = m_j/d_j^2$ as a weighting function to interpolate the velocity $\mathbf{v}_p$, where d_j is the distance between the proxy particle and the sampled point j, m_j is the mass of the sampled point. Thus the velocity of the proxy particle can be computed by using

$$\mathbf{v}_p = \frac{1}{\sum_j w_j} \sum_j w_j \mathbf{v}_j. \tag{8}$$

With the position, mass, velocity, normal of the proxy particle already computed in the aforementioned procedures, we can proceed to compute the coupling force efficiently.

4.2. Coupling Force Computation

To model the normal and tangential boundary conditions, as shown in Fig.3, we first respectively project the velocities of proxy particles and fluid particles to the normal direction and the tangential direction, then compute $\mathbf{v}_{pn}$, $\mathbf{v}_{pt}$, $\mathbf{v}_{fn}$, and $\mathbf{v}_{ft}$. Here $\mathbf{v}_p$, $\mathbf{v}_f$ respectively denotes the velocities of proxy particle and fluid particle, and we label a single star $(*)$ over the related variables to denote the unknown velocities after collision. To get the four unknown variables, by simultaneously considering momentum conservation along both normal and tangential directions, we can formulate the following equations

$$m_p \mathbf{v}^*_{pn} + m_f \mathbf{v}^*_{fn} = m_p \mathbf{v}_{pn} + m_f \mathbf{v}_{fn}, \tag{9}$$

$$m_p \mathbf{v}^*_{pt} + m_f \mathbf{v}^*_{ft} = m_p \mathbf{v}_{pt} + m_f \mathbf{v}_{ft}, \tag{10}$$

where m_p represents the mass of proxy particle, and m_f represents the mass of fluid particle.

To model the restitution in normal direction, we introduce Newton's coefficient e as

$$e = -\frac{\mathbf{v}^*_{pn} - \mathbf{v}^*_{fn}}{\mathbf{v}_{pn} - \mathbf{v}_{fn}}, \tag{11}$$

where $e = 0$ means that the collision is perfectly inelastic, while $e = 1$ means the collision is perfectly elastic.

Let us assume accurate physical analysis, the restitution in tangential direction is rather complex. For the simplification purpose, we define a variable δ to control the different slip condition.

$$\delta = \frac{\mathbf{v}^*_{pt} - \mathbf{v}^*_{ft}}{\mathbf{v}_{pt} - \mathbf{v}_{ft}}, \tag{12}$$

where $\delta = 0$ means no-slip in the collision, while $\delta = 1$ states the collision is free to slip. By solving Eq.9 to Eq.12, we can obtain the velocities of proxy particle and fluid particle. The collision between the two particles is directly controlled by parameters e and δ, and both of their value ranges are [0,1].

After computing velocities, we need to update the velocities and positions for fluid particles and FEM nodal points. For fluid particles, their velocities and positions can be directly updated. As for proxy particles, we need to compute the coupling forces of proxy particles and then distribute them to the sampled points. Inspired by the idea of direct forcing, we compute the force using

$$\mathbf{f}_p = \frac{(\mathbf{v}^*_p - \mathbf{v}_p) m_p}{\Delta t}. \tag{13}$$

Furthermore, the force distribution of proxy particle can be formulated as

$$\mathbf{f}_j = \frac{w_j}{\sum_j w_j} \mathbf{f}_p, \tag{14}$$

where $\mathbf{f}_j$ is the distributed force of sampled point j, and w_j has the same meaning as that in Eq.8.

4.3. Penetration Handling

To prohibit penetration of the fluid particles into the solid boundary, we detect collision based on the updated positions of FEM nodal points and fluid particles. Just like the similar situation in [BTT09], when a fluid particle collides with the FEM mesh, we correct the position of the fluid particle in an additional step. Moreover, to improve the robustness of the coupling, we need to synchronously handle the situation that one fluid particle collides with more than one polygon.

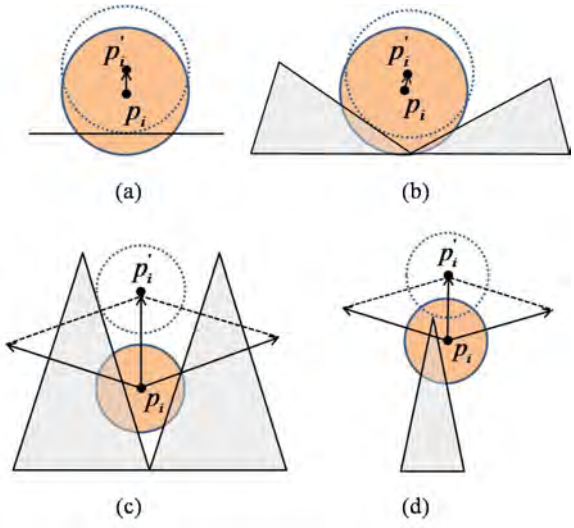

Figure 4: *Different boundary conditions. (a) a particle collides with one triangle. (b) a particle collides with two triangles. (c-d) a particle collides with two triangles with conflicting normals.*

As shown in Fig.4(a), when a fluid particle only collides with a single polygon, the penetration problem can be solved by moving it along the normal direction of the polygon. However, when the fluid particle interacts with more than one polygon, the particle direction to be corrected cannot be directly obtained. In such a case, we need to compute the correct motion direction $\mathbf{dir}_i$ of the fluid particle, and we initiate $\mathbf{dir}_i$ with the normal $\mathbf{n}_c$ obtained from Eq.6.

To enforce the non-penetration constraint, the fluid particle i should keep a certain distance away from all the triangles, which is at least equal to its radius r_0. When penetration occurs, the fluid should move $r_0 - d_{ij}$ along the normal direction $\mathbf{n}_j$ of the triangle. We project $\|r_0 - d_{ij}\|\mathbf{n}_j$ to $\mathbf{dir}_i$ and set its maximal value to be the distance that the fluid particle actually moves.

$$dis = \max_j \left\{ \|r_0 - d_{ij}\| \frac{\mathbf{n}_j \cdot \mathbf{dir}_i}{\|\mathbf{n}_j\|} \right\}. \qquad (15)$$

The above correction method works well when polygon normals do not have conflict with each other, which means that the projection of $\mathbf{n}_j$ to $\mathbf{dir}_i$ is always positive. However, for the boundary conditions shown in Fig.4(c-d), the aforementioned method cannot enforce the non-penetration constraint properly. We should compute $\mathbf{dir}_i$ by taking all the normals of the collided triangles into account. If the direction is readily available, we move the fluid particle along such direction until no penetration occurs. When neither of them can resolve the penetration, we skip the correction step. It will be naturally solved in subsequent time steps, since it is usually caused by deformable solids.

Although this correction step may lead to a higher density of the fluid around the boundary, the density ratios can be rapidly re-balanced in subsequent time steps with the help of the weakly compressible SPH model being employed.

5. CUDA-based Implementation

The proposed coupling method is entirely implemented on GPUs with CUDA. To accelerate memory access, all of the physical properties such as positions, velocities, and forces are stored in the global video memory, while the constant values are stored in registers. We detail the implementation as follows.

SPH. To compute the density, pressure, and force of the fluid particle, the searching for neighboring particles is required, which is also the most time-consuming task in SPH simulation. We transplant the efficient framework from the NVIDIA CUDA "Particles" demo to conduct this task. We invoke one CUDA thread for each fluid particle to handle other related computation, including the calculation of SPH kernel integration, density, pressure forces and viscosity forces, and the dynamic update of the fluid particle velocity and position.

Coupling. To compute the coupling force, we first invoke one CUDA thread for each fluid particle, of which we detect collision, generate proxy particles, and solve Eq.9 to Eq.12, update the velocities and positions of fluid particles, and compute the forces of proxy particles. This scheme is also used in particle position correction. To accelerate collision detection, we compute an Axis-Aligned Bounding Box (AABB) for each polygon of solid mesh, and use a uniform grid to subdivide the working space into a grid with uniform cells. The cell size is determined by the maximum length of AABB edges and the radius of the fluid particle. Then we search the neighboring polygons of the fluid particles and conduct intersection test among fluid particles and polygons, which can greatly improve the efficiency of collision detection.

Distributing coupling forces. After the coupling forces have been computed, the forces of proxy particles have to be redistributed to the sampled points according to Eq.14. Since the coupling forces of proxy particles are parallelly computed on GPUs and each sampled point may contribute to more

than one proxy particle, it is difficult to distribute forces to the sampled points efficiently. As shown in Fig.5, we devise a specific algorithm to solve this problem. First, we estimate the maximum number of sample points that a proxy particle may involve, and allocate memory for each proxy particle to store the indices of the sampled points and their corresponding forces (Fig.5(a)). Since the generated force array must be quite large, naively traversing it and adding the forces for each sampled point will severely increase the computing time. To avoid it, as shown in Fig.5(b), we sort the array based on the indices of the sampled points using a radix sort algorithm. Meanwhile, we locate the first and last index of the array for each sample point, and invoke an additional thread for each sampled point to sum up the forces. Comparing with the naive traversing method, our method can reduce the computing time from 70ms to 6ms for a scene containing 40k fluid particles and 10k FEM nodal points.

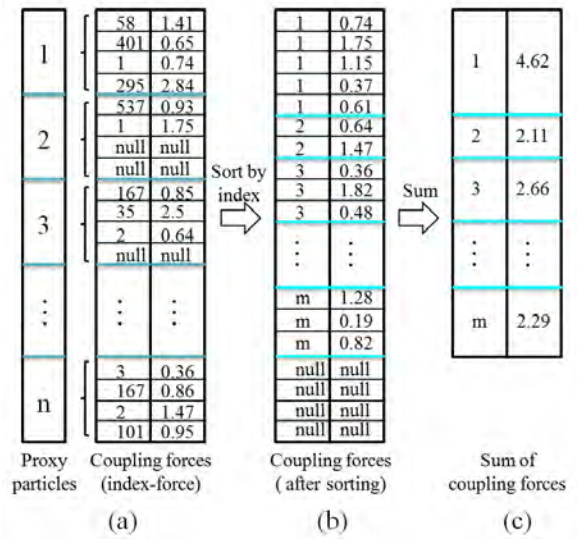

Figure 5: *Flow chart of distributing the coupling forces. (a) shows the data structure of coupling forces. Both the indices and corresponding forces of the sampled points need to be stored. (b) shows the sorted array of coupling forces based on the indices of the sampled points. (c) shows the result of force distribution.*

6. Experimental Results and Discussion

We implemented our method on a PC with a Geforce GTX 580 GPU, Intel Xeon E5630 CPU using C++, CUDA, and GLSL APIs. We demonstrate the advantages of our method via various different scenarios, of which the solid deformations are all simulated using the nonlinear TLED method in our experimental scenarios, and fluids are simulated using weakly compressible SPH.

Table 1 documents the parameter values used in our

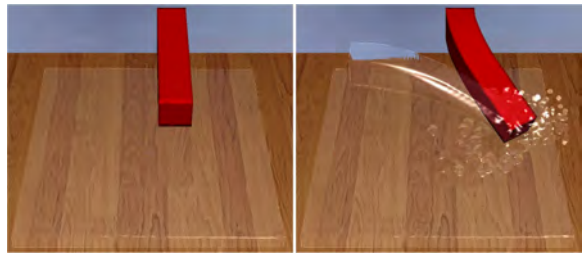

Figure 6: *Fluids poured over a deformable beam.*

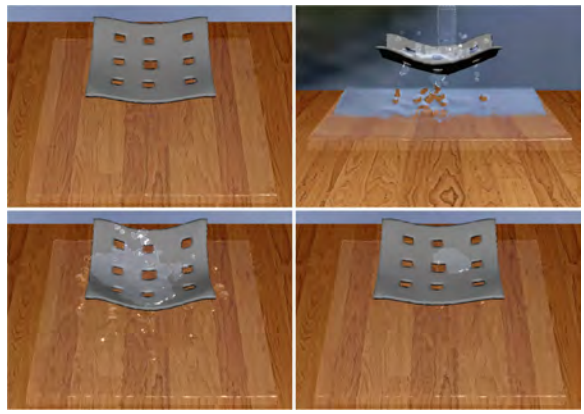

Figure 7: *The coupling of cloth and fluids.*

experiments. It is extremely challenging to simulate low-viscous incompressible fluids in real time, since the time step must be limited in consideration of numerical stability. Therefore, we set the viscosity 200 times its physically meaningful value so that relatively larger time step can be afforded towards a goal of guaranteeing the numerical stability [Kel06]. The parameter c_s in WCSPH is also

Figure 8: *Fluids poured into a deformable bowl.*

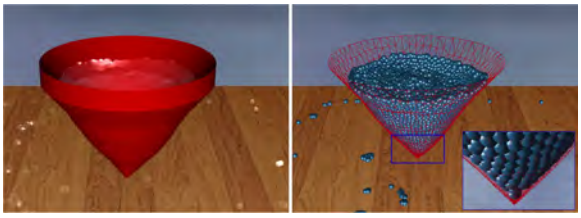

Figure 9: *Fluids poured into a deformable funnel with sharp corner. The right figure shows the interaction between the inner surface of the funnel and fluid particles.*

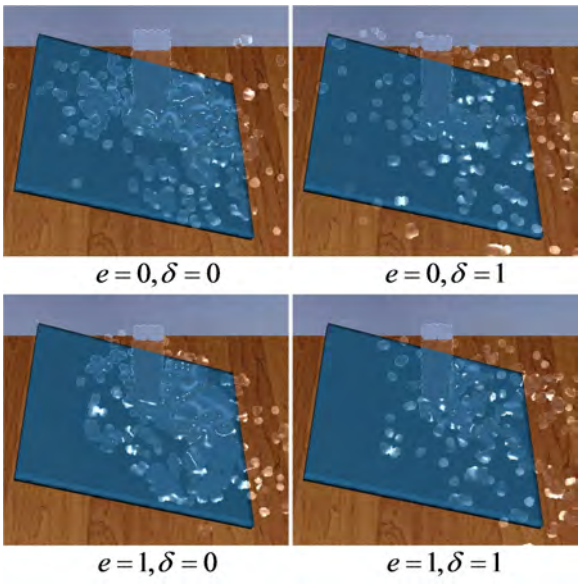

Figure 10: *Illustration of predictor-corrector results under different boundary conditions.*

factitiously set small for more stable results, nonetheless, this might cause slightly higher density ratios. For example, the measured average density ratios in our experiments are around 1.1∼1.5, which is not perfect enough but still better than directly adopting ideal gas equation, compared with [MCG03, BT07, SP09, IAGT10]. By these means, we can achieve numerically-stable coupling simulation using a constant time step 0.0015s, which is intrinsically constrained by the Courant-Friedrich-Levy (CFL) condition for SPH.

Table 1: *Parameter values used in the experiments.*

Properties	Parameters	Values	Unit
Density	ρ_0	1000	kg/m^3
Mass	m_f	0.0002	kg
Support	h	0.01	m
Viscosity	μ	0.2	$Pa \cdot s$
Time step	Δt	0.0015	s
Speed of sound	c_s	20	m/s

Table 2 shows statistics for the average time cost (in milliseconds) for each simulation step (each animation frame only includes one time step), which states that the time cost increases with the increase of fluid particles, tetrahedral elements and surface polygons. Meanwhile, we also document the number of tetrahedrons (#Tet) and the number of surface triangles (#ST) for each scenario.

Fig.1 is a scenario that water flows through a segment of hose. It demonstrates the coupling effects between fluids and shell-like thin solids. Here the hose constrains the fluid flow, while the fluids exert pressure on the hose. Note that non-penetration can be robustly achieved.

Fig.6 is an animation scenario where fluid is splashed over a deformable beam, and the beam bends to right due to fluid pressure. To intuitively demonstrate the result of the two-way coupling, here we ignore the gravity of the beam and add fixed constraints to one end of the beam.

Fig.7 shows the animation effects when fluid is poured onto the cloth with holes, which demonstrates that our coupling method can handle complex interface between fluids and deformable solids.

Fig.8 shows the interaction between fluids and a deformable bowl when water is poured into the bowl. The high-fidelity large deformation of the bowl can be handled robustly with our nonlinear FEM model.

Fig.9 shows the penetration handling for complex objects with sharp corners. In this scenario, the top of the deformable funnel is fixed, and water is poured into it. As shows in the figure, there are many tiny triangles towards the bottom, which have even more conflicting normals than that being shown in Fig.4(c). However, the funnel can naturally hold the water while no penetration occurs in the region of sharp corner.

Fig.10 shows scenarios that fluid is poured onto a reclining board, where different boundary conditions are imposed to verify the effectiveness of our predictor-corrector scheme, including no-slip, free-slip, elastic, and inelastic conditions. The experimental results show that our method can easily handle these conditions only by adjusting two parameters. In the interest of space, please refer to our supplementary video for more experimental demonstrations.

Compared with the boundary particle method proposed by [MST*04], our proxy particles are generated dynamically with little additional time and memory cost, whose velocity can be obtained by interpolating that of the surrounding vertices according to the mesh topological relationship. Hence, it avoids dynamically subdividing the polygons when the mesh elements are relatively sparse w.r.t. fluid particles. Compared with the penalty-force-based method [MST*04], we do not introduce stiffness into the coupling process, thus

Table 2: *Time performance (in milliseconds) of our experiments.*

Scene	#Tet	#ST	#Particles	TLED	SPH	Coupling	Display
Hose (Fig.1)	14.7k	9.9k	40k	13.8	11.0	13.0	3.2
Beam (Fig.6)	4.6k	2.8k	40k	10.9	8.3	9.5	2.8
Cloth (Fig.7)	4.2k	5.3k	40k	11.5	8.6	8.8	2.9
Bowl (Fig.8)	7.3k	5k	30k	10.7	8.2	8.4	3.2
Funnel (Fig.9)	6.0k	4.0k	30k	12.9	8.1	12.2	3.1

non-penetration and inelastic collision can be guaranteed at the same time.

Compared with [BTT09], which adopts the similar predictor-corrector scheme to our method, ours can afford deformable solids rather than simple rigid objects. Besides, although the interaction model between polygon and particle is more complex and accurate than the unified particle-based model, we can still achieve real-time efficiency that is directly benefited from our CUDA-based implementation. Moreover, our boundary handling method can deal with the case that one fluid particle interacts with more than one solid body, which was only mentioned but never actually solved in [BTT09].

Our new method dynamically generates proxy particles at the interface of solids and fluids to facilitate the interaction, and non-penetration can be achieved by a position correction step. Since collision detection and position correction are of relevance to the solid surface, the surface complexity will inevitably affect the efficiency and accuracy of the coupling method. First, the time cost for collision detection will increase when the surface is more complex and more triangles are involved. Second, the generation of proxy particle will involve more sampled points, which increases the computing time and subsequently affects the computation of the distribution of the coupling forces. Third, the penetration handling method discussed in Section 4.3 can also be affected by the surface complexity.

The main limitation of our method is that our position correction method may sometimes lead to the stacking of fluid particles around the boundary interfaces, which tends to result in high pressure ratios. We have replaced the ideal gas equation with Tait equation [BT07] to alleviate this implausible phenomenon. Nonetheless, high pressure ratios and Tait equation both impose limitation on allowable time step, and subsequently affect the numerical stability.

To afford larger time step while guaranteeing the numerical stability, we factitiously adopt high viscosity and low speed of sound in the fluid (c_s in Eq.4). High viscosity will unavoidably damp the velocities of fluid particles quickly, but can alleviate the constraint over time step to a certain extent. We set a small value for c_s in fluids according to the relationship of the time step and c_s discussed in [BT07], even though it gives rise to larger compressibility. Directly benefited from these, fluids can be simulated stably in real time as shown in our experiments.

To overcome the overly damped and compressible problems, it is feasible to straightforwardly integrate our method with PCISPH framework, just like the simulation of low-viscous incompressible fluids presented in [IAGT10, IABT11]. Nonetheless, it is difficult to achieve real-time simulation due to the intrinsic efficiency problem inherited from PCISPH.

7. Conclusion and Future Work

We have presented a novel two-way coupling method to simulate the physical interaction with high fidelity between SPH based fluids and nonlinear FEM based deformable solids. We proposed a proxy particle generating scheme to model the physical interaction of fluids and deformable solids, and designed the CUDA-based algorithm to efficiently distribute the coupling forces to FEM nodal points. Based on the predictor-corrector scheme, our coupling method can afford different boundary conditions and guarantee the constraint of non-penetration simultaneously. Moreover, our method can handle shell-like thin solids. For a scenario with 40,000 fluid particles and 14,000 tetrahedrons, our CUDA-based implementation can achieve the real-time performance in standard PC platform, and our experimental results have shown great promise towards system efficiency.

Our immediate efforts are geared towards simulating low-viscous incompressible fluids and drastically eliminating the stacking side-effect of fluid particles and high pressure ratios around the interface by improving the PCISPH method. Moreover, applying this method to real-world applications such as virtual surgery simulation deserves further investigation.

Acknowledgement: This research is supported by National Natural Science Foundation of China (No.61190120, No.61190125 and No.61190124) and NSF (IIS-0949467, IIS-1047715, and IIS-1049448). We thank Qing Xia for writing the rendering code and helping the video production. We also thank the anonymous reviewers for their constructive critiques.

References

[ACF11] ALLARD J., COURTECUISSE H., FAURE F.: Implicit fem and fluid coupling on gpu for interactive multiphysics simulation. In *ACM SIGGRAPH 2011 Talks* (2011), SIGGRAPH '11, ACM, pp. 52:1–52:1. 2, 3

[AIA*12] AKINCI N., IHMSEN M., AKINCI G., SOLENTHALER B., TESCHNER M.: Versatile rigid-fluid coupling for incompressible sph. In *ACM SIGGRAPH 2012* (2012), To appear. 2, 3

[APKG07] ADAMS B., PAULY M., KEISER R., GUIBAS L. J.: Adaptively sampled particle fluids. In *ACM SIGGRAPH 2007 papers* (2007), SIGGRAPH '07, ACM, p. 48. 2

[BIT09] BECKER M., IHMSEN M., TESCHNER M.: Corotated sph for deformable solids. In *NPH* (2009), Galin E., Schneider J., (Eds.), Eurographics Association, pp. 27–34. 2

[BT07] BECKER M., TESCHNER M.: Weakly compressible sph for free surface flows. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2007), SCA '07, Eurographics Association, pp. 209–217. 2, 4, 8, 9

[BTT09] BECKER M., TESSENDORF H., TESCHNER M.: Direct forcing for lagrangian rigid-fluid coupling. *IEEE Transactions on Visualization and Computer Graphics 15*, 3 (May. 2009), 493–503. 1, 2, 3, 6, 9

[Fed02] FEDKIW R. P.: Coupling an eulerian fluid calculation to a lagrangian solid calculation with the ghost fluid method. *J. Comput. Phys. 175*, 1 (Jan. 2002), 200–224. 2

[GSSP10] GOSWAMI P., SCHLEGEL P., SOLENTHALER B., PAJAROLA R.: Interactive sph simulation and rendering on the gpu. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2010), SCA '10, Eurographics Association, pp. 55–64. 2

[HKK07] HARADA T., KOSHIZUKA S., KAWAGUCHI Y.: Smoothed particle hydrodynamics on gpus. *Proc. of Computer Graphics International* (2007), 63–70. 2

[IABT11] IHMSEN M., AKINCI N., BECKER M., TESCHNER M.: A parallel sph implementation on multi-core cpus. *Comput. Graph. Forum 30*, 1 (2011), 99–112. 2, 9

[IAGT10] IHMSEN M., AKINCI N., GISSLER M., TESCHNER M.: Boundary handling and adaptive time-stepping for pcisph. In *VRIPHYS* (2010), Erleben K., Bender J., Teschner M., (Eds.), Eurographics Association, pp. 79–88. 3, 8, 9

[IUDN10] IWASAKI K., UCHIDA H., DOBASHI Y., NISHITA T.: Fast particle-based visual simulation of ice melting. *Computer Graphics Forum 29*, 7 (2010), 2215–2223. 3

[JWM10] JOLDES G. R., WITTEK A., MILLER K.: Real-time nonlinear finite element computations on gpu - application to neurosurgical simulation. *Computer Methods in Applied Mechanics and Engineering 199*, 49ÍC52 (2010), 3305 – 3314. 2

[KE12] KROG O. E., ELSTER A. C.: Fast gpu-based fluid simulations using sph. In *Proceedings of the 10th international conference on Applied Parallel and Scientific Computing - Volume 2* (2012), PARA'10, Springer-Verlag, pp. 98–109. 2

[Kel06] KELAGER M.: Lagrangian fluid dynamics using smoothed particle hydrodynamics, 2006. 7

[LAD08] LENAERTS T., ADAMS B., DUTRÉ P.: Porous flow in particle-based fluid simulations. *ACM Trans. Graph. 27*, 3 (Aug. 2008), 49:1–49:8. 3

[LD08] LENAERTS T., DUTRÉ P.: Unified sph model for fluid-shell simulations. In *ACM SIGGRAPH 2008 posters* (2008), SIGGRAPH '08, ACM, pp. 12:1–12:1. 3

[LZLW11] LIU N., ZHU F., LI S., WANG G.: Anisotropic kernels for meshless elastic solids. In *2011 12th International Conference on ComputerAided Design and Computer Graphics(CAD/Graphics)* (Sept. 2011), pp. 349–356. 2

[MCG03] MÜLLER M., CHARYPAR D., GROSS M.: Particle-based fluid simulation for interactive applications. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2003), SCA '03, Eurographics Association, pp. 154–159. 2, 4, 8

[MJLW07] MILLER K., JOLDES G., LANCE D., WITTEK A.: Total lagrangian explicit dynamics finite element algorithm for computing soft tissue deformation. *Communications in Numerical Methods in Engineering 23*, 2 (2007), 121–134. 2, 4

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

[MSKG05] MÜLLER M., SOLENTHALER B., KEISER R., GROSS M.: Particle-based fluid-fluid interaction. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2005), SCA '05, ACM, pp. 237–244. 3

[MST*04] MÜLLER M., SCHIRM S., TESCHNER M., HEIDELBERGER B., GROSS M.: Interaction of fluids with deformable solids: Research articles. *Comput. Animat. Virtual Worlds 15*, 3-4 (Jul. 2004), 159–171. 1, 2, 3, 8

[QPN*10] QIN J., PANG W.-M., NGUYEN B. P., NI D., CHUI C.-K.: Particle-based simulation of blood flow and vessel wall interactions in virtual surgery. In *Proceedings of the 2010 Symposium on Information and Communication Technology* (2010), SoICT '10, ACM, pp. 128–133. 3

[SG11] SOLENTHALER B., GROSS M.: Two-scale particle simulation. *ACM Trans. Graph. 30*, 4 (Aug. 2011), 81:1–81:8. 2

[SP09] SOLENTHALER B., PAJAROLA R.: Predictive-corrective incompressible sph. In *ACM SIGGRAPH 2009 papers* (2009), SIGGRAPH '09, ACM, pp. 40:1–40:6. 2, 8

[SSIF07] SIFAKIS E., SHINAR T., IRVING G., FEDKIW R.: Hybrid simulation of deformable solids. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, Switzerland, 2007), SCA '07, Eurographics Association, pp. 81–90. 3

[SSP07] SOLENTHALER B., SCHLÄFLI J., PAJAROLA R.: A unified particle model for fluid-solid interactions: Research articles. *Comput. Animat. Virtual Worlds 18*, 1 (Feb. 2007), 69–82. 3

[TCC*09] TAYLOR Z. A., COMAS O., CHENG M., PASSENGER J., HAWKES D. J., ATKINSON D., OURSELIN S.: On modelling of anisotropic viscoelasticity for soft tissue simulation: Numerical solution and gpu execution. *Medical Image Analysis 13*, 2 (2009), 234 – 244. 2

[TCO08] TAYLOR Z. A., CHENG M., OURSELIN S.: High-speed nonlinear finite element analysis for surgical simulation using graphics processing units. *Medical Imaging, IEEE Transactions on 27*, 5 (May. 2008), 650–663. 2, 4

[VdLGS09] VAN DER LAAN W. J., GREEN S., SAINZ M.: Screen space fluid rendering with curvature flow. In *Proceedings of the 2009 symposium on Interactive 3D graphics and games* (2009), I3D '09, ACM, pp. 91–98. 3