2013 13th International Conference on Computer-Aided Design and Computer Graphics

Synthesizing Solid-induced Turbulence for Particle-based Fluids

Xuqiang Shao, Zhong Zhou, Jinsong Zhang, Wei Wu
State Key Laboratory of Virtual Reality Technology and Systems
Beihang University
Beijing, China
shaoxuqiang@gmail.com

***Abstract*—Simulating the accompanying turbulent details of fluid-solid coupling is still challenging, as numerical dissipation always plagues current fluid solvers. In this paper, we propose a novel particle-based method to simulate turbulent details generated behind objects in SPH fluids. A turbulence production model, approximating the boundary layer theory on the fly in SPH fluids, is proposed to identify which fluid particles shed from object surfaces and are seeded as vortex particles. Then the fluctuating velocities stemming from the generated vorticity field are calculated using an SPH-like summation interpolant formulation of the Biot-Savart law. And the stable evolution of vorticity field is solved by introducing an artificial dissipation term into the vorticity N-S equation. The advantages of our turbulence synthesis method in computer animation are demonstrated via various virtual scenarios.**

***Keywords*-turbulence synthesis; SPH; fluid-solid coupling; particle-based fluid; fluid simulation;**

I. Introduction

In computer animation, particle-based SPH method is becoming increasingly popular to create realistic animation of many complex phenomena, such as water, smoke and fire. Actually, more visually interesting natural phenomena emerge when complex objects are coupled to fluids (Figure 1). For SPH fluid solvers, however, the realistic simulation of turbulent details in fluid-solid coupling is still challenging.

The damping of turbulent details is inevitable due to both numerical dissipations and limited resolution discretization of fluid solvers. A popular approach to reduce the numerical dissipations in Eulerian method is the use of higher order advection schemes [1], [2]. But these methods are not applicable to SPH fluid, because the advection term is not needed for particle systems. Adaptive particle sampling [3] and two-scale scheme [4] address the problem by refinement of visually important regions. But the generated turbulence is difficult to be preserved in the global flow with the coarse computational nodes, when they move out of the high-resolution region. In addition, the subdivision level of particles is hard to be determined for the turbulence recovering (Figure 1(b)).

A more effective method of modeling turbulent details is to augment the basic fluid solver with synthetic turbulence model [5]–[9]. These methods can simulate visually realistic turbulence in fluids, however, the turbulence induced by solid bodies are not their focus.

To simulate turbulence behind objects, Pfaff et al. [10] present a physically plausible turbulence model to seed vortex particles where the separated boundary layers will transit into actual turbulence. This method successfully synthesizes turbulence around rigid bodies for Eulerian grid fluid. However, it is time-consuming to create a pre-computed artificial boundary layer that captures the characteristics of turbulence generation around objects. Several researchers have paid attention to simulating turbulence around solid bodies in SPH fluids. Bo et al. [11] present a new method to create and preserve the turbulent details generated around moving objects, in which a high-resolution overlapping grid need to be bounded to each object and translates with the object. Yuan et al. [12] propose a swirling incentive particle (SIP) method, which introduces random swirling-probability values to efficiently mimic the stochastic nature of the turbulence. But sometimes unrealistic turbulence may be created from an inappropriate choice of parameters, especially when handling complex deformable boundaries.

From the comparison of three coupling results in Figure 1, we conclude that the simulation of turbulence around objects (Figure 1(c)) makes the solid-fluid coupling more visually realistic. In order to simulate turbulent details behind the moving solid bodies, we propose a novel turbulence synthesis model which can be incorporated into particle-based solid-fluid coupling. By approximating the boundary layer theory on the fly in SPH fluids, we can identify which fluid particles shed from object surfaces and are seeded as vortex particles. Then we propose an SPH-like summation approximation of Biot-Savart law to calculate the fluctuating velocities stemming from the vorticity field. The stable evolution of vorticity field is solved by introducing an artificial dissipation term into the vorticity N-S equation.

The results, as shown in the accompanying video, demonstrate that the proposed turbulence synthesis method achieves the realistic simulation of turbulent details behind solid bodies, which can be regarded as a complementary for a particle-based fluid simulator.

II. Related Work

Since Müller et al. [20] used SPH to produce compelling fluid simulations at interactive rates, SPH has been used to model such phenomena as incompressible fluid [26]–[28],

978-1-4799-2576-6/13 $31.00 © 2013 IEEE
DOI 10.1109/CADGraphics.2013.40

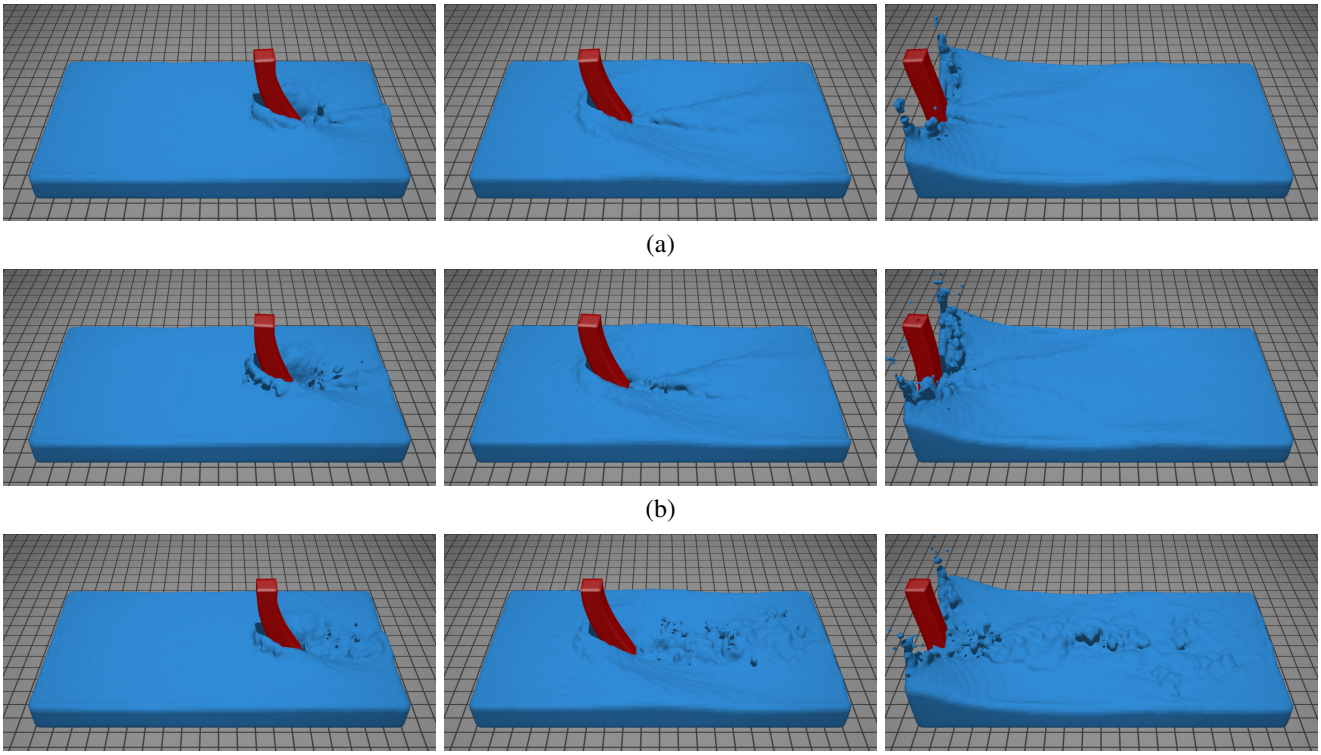

Figure 1. Comparison. (a) Pure solid-fluid coupling. (b) Adaptive sampling [3] in the coupling. (c) Turbulence synthesis.

hairs [29], melting [30], viscoelastic material [31], sand [32] and fluid-solid coupling [21]–[24], [34], [35].

The loss of turbulence due to inherent numerical dissipation is one of the prominent difficulties in fluid simulation. Fedkiw et al. [13] propose vorticity confinement to amplify existing grid vorticity in the fluid. However, if the uniform simulation grid is not fine enough to capture the desired details, vorticity confinement cannot recover them. The numerical dissipation problem can also be alleviated by using higher order advection schemes [1], [2], but turbulence represented by these methods are still limited by the underlying grid resolution. By solving the N-S equations with higher-resolution grids in a local concerned space, several adaptive methods [3], [4], [14] can model turbulence formation around objects. When moving out of the high-resolution region, the vortices are difficult to be preserved.

Procedure synthesis methods, which add further details in a sub-grid by using noise functions, have been introduced. Kim et al. [6] use a wavelet decomposition to detect where small-scale detail is being lost, and apply an incompressible turbulence function to reintroduce these details. Schechter et al. [7] track bands of turbulent energy using a simple linear model, and create the turbulent velocity using flow noise. Narain et al. [8] present a fast and effective technique for simulating turbulent fluids through a sub-grid turbulence model that can be integrated into existing simulations and tracks the production and evolution of turbulent flow automatically. The vortex particle method is more suitable to synthesize turbulent details in fluids. Selle et al. [15] make use of seeded vortex particles to introduce additional vorticity into the grid fluid for highly turbulent flows. But their method requires the artist to specify where these particles are injected into the flow. Park and Kim [16] model gaseous phenomena by entirely distributing vortex particles on the whole grid and utilizing a pure Lagrangian simulation with a vorticity transport equation. Some other grid-particle methods [9], [17], [18] incorporating vortex particles into Eulerian grid fluid can model visually realistic turbulence effects. However, these methods mainly deal with the preservation of vortices already represented in the overall flow rather than the turbulence formation around objects.

Several works focus on the simulation of turbulence generated around objects. Pfaff et al. [10] employ vortex particles in an Eulerian grid fluid to simulate obstacle-induced turbulence. Based on boundary layer theory, their method identifies areas where the separated layers will transit into actual turbulence and seeds vortex particles. However, it is time-consuming to create a precomputed artificial boundary layer that captures the characteristics of turbulence generation around objects. Bo et al. [11] provide a physically

plausible method to model the turbulent details around both rigid and deformable objects in SPH-based gaseous fluids, in which a high-resolution overlapping grid is bounded to each object and translates with the object. In order to reduce numerical dissipations consuming turbulent details in SPH fluids, Jang et al. [19] incorporate the Hermite-interpolation scheme into a particle-advection process. And for capturing multi-scale vortical motions efficiently, they propose large-scale kernels and a small-scale vorticity (SSV) model, respectively. By seeding swirling incentive particles (SIPs) around objects immersed in SPH fluids, Yuan et al. [12] introduce random swirling-probability values to efficiently mimic the stochastic nature of the turbulence.

III. Algorithmic Overview

A. Algorithm Architecture

Figure 2 shows the algorithmic flow of our turbulence synthesis method. In each simulation loop, we update the new position and velocity based on the results of last simulation loop, and then employ a uniformly sampled distance field to track the fluid surface and render the current simulation results. We detail the algorithm as follows:

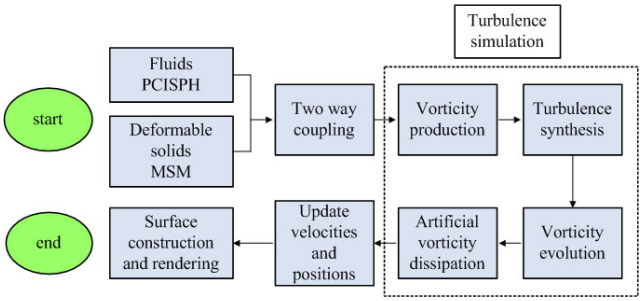

Figure 2. The pipeline of each simulation cycle.

Simulate the motions of fluids and solids. Respectively employ predictive-corrective incompressible SPH (PCISPH) and meshless shape matching (MSM) models to compute velocities and positions of fluid and solid particles without considering their interaction.

Two way coupling. Compute the two way coupling forces between fluid particles and solid particles.

Vorticity production. Identify which fluid particles shed from object surfaces and are seeded as vortex particles by approximating the boundary layer theory in SPH fluids.

Turbulence synthesis. Calculate the fluctuating velocities stemming from the vorticity field using an SPH-like summation interpolant formulation of the Biot-Savart law.

Vorticity evolution. Evolve the generated vorticity field by calculating the vorticity stretching and diffusion terms.

Artificial vorticity dissipation. Dissipate the generated vorticity field using an artificial dissipation term.

Update velocities and positions. Update velocities and positions of fluid and solid particles according to the updated coupling forces.

Surface construction and rendering. Construct the implicit surfaces based on the method of [34], and render the extracted triangle surfaces using Pov-ray.

B. SPH Fluids

Generally, the governing Navier-Stokes equations of Lagrangian fluid solvers are given as

$$\frac{d\mathbf{u}}{dt} = -\frac{1}{\rho}\nabla p + \mu\nabla^2\mathbf{u} + \mathbf{f} \tag{1}$$

$$\frac{d\rho}{dt} = -\rho\nabla\cdot\mathbf{u} \tag{2}$$

where $\mathbf{u}$ denotes the velocity, ρ the density, p the pressure, $\mathbf{f}$ the external force, and μ viscosity coefficient.

SPH method discretizes the continuum into a collection of particles, and every particle carries individual properties. A field variable A at location $\mathbf{x}_i$ is estimated by a weighted sum of neighboring particles j located within a distance h:

$$A(\mathbf{x}_i) = \sum_j m_j \frac{A_j}{\rho_j} W(\mathbf{x}_{ij}, h) \tag{3}$$

where m_j is the mass, and $\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$. The function $W(\mathbf{x}_{ij}, h)$, also written as W_{ij}, is a smoothed and normalized kernel with the support radius h [20].

We adopt the density model in [25] which allows large density ratio to compute the particle density

$$\rho_i = m_j \sum_j W(\mathbf{x}_{ij}, h) \tag{4}$$

The symmetric pressure force F^p and viscosity force F^v are formulated as:

$$F_i^p = -\frac{m_i}{\rho_i}\sum_j \frac{m_j}{\rho_j}\frac{p_i + p_j}{2}\nabla W(\mathbf{x}_{ij}, h) \tag{5}$$

$$F_i^v = \frac{\mu m_i}{\rho_i}\sum_j \frac{m_j}{\rho_j}(\mathbf{u}_j - \mathbf{u}_i)\nabla^2 W(\mathbf{x}_{ij}, h) \tag{6}$$

We adopt the PCISPH method [27], which enforces incompressibility and allows larger time steps, to iteratively compute the pressure p_i. In each iteration, the predicted position $\mathbf{x}_i^*(t+\Delta t)$ and velocity $\mathbf{u}_i^*(t+\Delta t)$ of particle i are computed based on $\mathbf{x}_i(t)$, $\mathbf{u}_i(t)$ and the predicted pressure forces. And the predicted density $\rho_i^*(t+\Delta t)$ is calculated using the updated interparticle distance $\mathbf{x}_{ij}^* = \mathbf{x}_i^*(t+\Delta t) - \mathbf{x}_j^*(t+\Delta t)$. Then, the particle pressure that corrects the predicted density error $\rho_{err_i}^*(t+\Delta t) = \rho_i^*(t+\Delta t) - \rho_0$ is updated as

$$p_i(t) += \delta\rho_{err_i}^*(t+\Delta t) \tag{7}$$

where δ is a precomputed value. Finally, the predicted pressure force is used to recompute the positions and velocities for all particles. This procedure is repeated until all predicted density fluctuation $\rho^*_{err_i}(t+\Delta t)$ is lower than a predefined maximum value η.

C. Solid Deformation

The previous deformation methods [21], [23], [34] have the instability problems for large deformations, and require too restrictive time steps to handle stiff materials. We adopt the meshless shape matching (MSM) method [33] to model deformable objects. The idea is to replace energies by geometric constraints and forces by distances of current positions to goal positions. These goal positions are determined via a generalized shape matching of an undeformed rest state with the current deformed state of the point cloud. Specifically, given two sets of particles $\mathbf{x}_i^0$ and $\mathbf{x}_i$, we find the rotation matrix R which minimizes

$$\sum_i m_i(R(\mathbf{x}_i^0 - \mathbf{x}_{cm}^0) - (\mathbf{x}_i - \mathbf{x}_{cm}))^2 \tag{8}$$

where $\mathbf{x}_{cm}^0$ and $\mathbf{x}_{cm}$ are mass centers of the initial shape and the current shape respectively. Then the goal position g_i of a particle i is $\mathbf{g}_i = R(\mathbf{x}_i^0 - \mathbf{x}_{cm}^0) + \mathbf{x}_{cm}$. Finally, elastic deformation is modeled by pulling a deformed geometry towards the well-defined goal positions using

$$\mathbf{x}_i(t+\Delta t) = \mathbf{x}_i(t) + \mathbf{u}_i\Delta t + \alpha(\mathbf{g}_i(t) - \mathbf{x}_i(t)) + \frac{(\Delta t)^2\mathbf{f}}{m_i} \tag{9}$$

where $\alpha \in [0...1]$. The method of [33] proves that the way of treating the internal elastic forces keeps MSM unconditionally stable, and allows relative larger time steps.

D. Two Way Coupling

We simulate the two-way coupling of PCISPH fluids and MSM-based solids by combining the coupling method [34] and [35]. The solid objects are uniformly sample with a band of boundary particles, and the thickness of the boundary particle band equals the support radius h of fluid particles. Based on the method of [35], we consider the relative contributions of the boundary particles to fluid particles when computing densities and coupling forces for PCISPH fluids.

IV. Turbulence Simulation

A. Vorticity Production

Objects immersed in the fluid are obvious turbulence generators. According to boundary layer theory [36], the friction of objects enforces a tangential flow velocity of zero at the solid boundaries. This leads to the formation of a thin layer with reduced flow speed, called the boundary layer. The curl of tangential flow velocity $\mathbf{u}_{\tan}$ in the boundary layer leads to the creation of a thin sheet of vorticity $\omega = \nabla \times \mathbf{u}_{\tan}$. At regions of high flow instability, the boundary layer is separated from the solid surface, and vorticity is ejected from the boundary layer and enters the flow as turbulence.

Because the boundary layer is a very thin sheet, we make an approximate assumption that the boundary layer of an SPH-based fluid consists of fluid particles whose distances to solid surface are less than a small threshold d_T. In this paper, we set $d_T = h$, so the neighboring fluid particles of all solid surface particles form the thin boundary layer. A fluid particle of the boundary layer is called as **BLFP**.

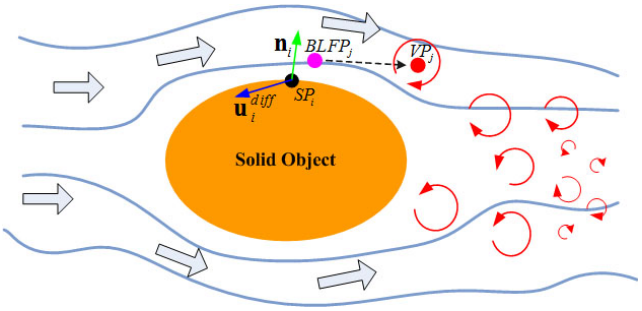

Figure 3. Our vorticity production method. A **BLFP** j shedding from the **SP** i obtains vorticity, and becomes a **VP** when having no neighboring solid particles.

Then we detect the separation points (**SP**) where **BLFP**s separate from the solid surfaces. Our approach is divided into three steps. First, we use the gradient ∇C_i of the smoothed color field C_i, which estimates the fluid-fluid interface normal in [37], to calculate the normal $\mathbf{n}_i$ of solid particles on the solid-fluid interface. Second, by using the formulation $\mathbf{u}_i^{diff} = \frac{\sum_j (\mathbf{u}_i - \mathbf{u}_j)W(\mathbf{x}_{ij},h)}{\sum_j W(\mathbf{x}_{ij},h)}$, we calculate the smoothed relative velocity $\mathbf{u}_i^{diff}$ of each solid surface particle i relative to all its neighboring fluid particles. Third, for each solid surface particle i, we compute the dot product $\mathbf{u}_i^{diff} \cdot \mathbf{n}_i$ between $\mathbf{u}_i^{diff}$ and $\mathbf{n}_i$. If $\mathbf{u}_i^{diff} \cdot \mathbf{n}_i < 0$, the solid particle i is determined as a **SP**.

In order to create turbulence induced by solid bodies, we determine which **BLFP**s actually shed from its neighboring **SP**s. For a pair of neighboring particles **BLFP** j and **SP** i, we calculate the dot product $\mathbf{x}_{ji} \cdot \mathbf{u}_{ji}$ between their relative position $\mathbf{x}_{ji}$ and their relative velocity $\mathbf{u}_{ji}$. If $\mathbf{x}_{ji} \cdot \mathbf{u}_{ji} > 0$, the **BLFP** j is separating from solid boundary, and obtains the vorticity ω_j formulated as:

$$\omega_j = \gamma(\mathbf{u}_{ji}^{\tan} \times \mathbf{n}_i) \tag{10}$$

where $\mathbf{u}_{ji}^{\tan}$ is the relative velocity along the tangent direction of the solid surface. γ evaluates the effects of fluid viscosity and solid material.

When a BLFP carrying the vorticity does not have neighboring solid particles, it is regarded as a vortex particle (**VP**)

seeded into fluids. All vortex particles form the vorticity field of the fluid flow. As shown in Figure 3, based on our vorticity production method, the turbulent details gradually form behind solid objects in a physically plausible way instead of being randomly seeded [12], [15].

B. Fluctuating Velocity Computation

To incorporate turbulent fluctuations into PCISPH fluids, we model the instantaneous velocity field $\mathbf{u}$, which describes the dynamics of fluid particles, by superposing the mean velocity field $\mathbf{U}$ with a rapidly fluctuating component $\mathbf{u}^v$. $\mathbf{U}$ is from the calculation of the momentum equation using PCISPH method, which is described before; $\mathbf{u}^v$ stems from the vorticity field created by our vorticity production model.

The velocity field $\mathbf{u}^v$ stemming from the vorticity field can be recovered via the Biot-Savart law [38], which computes $\mathbf{u}^v$ at the position $\mathbf{x}$ that is a distance $\mathbf{r} = \mathbf{x} - \mathbf{x}'$ from a vortex element $d\mathbf{x}'$ with vorticity ω by integrating over all the space:

$$\mathbf{u}^v(\mathbf{x}) = \frac{1}{4\pi} \int \frac{\omega(\mathbf{x}') \times \mathbf{r}}{|\mathbf{r}|^3} d\mathbf{x}' \tag{11}$$

For efficiently computing $\mathbf{u}_i^v$ of each fluid particle, we provide an SPH-like summation interpolant to approximate the Biot-Savart law. Specifically, by discretizing the above equation, we turn the integral into a summation formulation:

$$\begin{aligned} \mathbf{u}_i^v &= \frac{1}{4\pi} \sum_j \int_{\Omega_j} \frac{\omega(\mathbf{x}') \times \mathbf{r}}{|\mathbf{r}|^3} d\mathbf{x}' \\ &\approx \frac{1}{4\pi} \sum_j V_j \frac{\omega(\mathbf{x}_j) \times \mathbf{r}}{|\mathbf{r}|^3} \\ &= \sum_j \frac{m_j}{\rho_j} \left[\omega(\mathbf{x}_j) \times \mathbf{r}\right] \frac{1}{4\pi |\mathbf{r}|^3} \end{aligned} \tag{12}$$

If the simulation has n vortex particles, a straightforward summation would take $O(n^n)$ operations, which is too slow. So instead, based on the principle of SPH method [20], we only consider the influence of vortex particles within a certain distance h. For an approximation, we replace the term $\frac{1}{4\pi|\mathbf{r}|^3}$ with the radial symmetrical smoothing kernel $W(\mathbf{r}, h)$ of SPH method, and get the SPH-like summation interpolant formulation of the Biot-Savart law:

$$\mathbf{u}_i^v = \sum_j \frac{m_j}{\rho_j} \left[\omega(\mathbf{x}_j) \times \mathbf{r}\right] W(\mathbf{r}, h) \tag{13}$$

where W we used is the isotropic gaussian kernel $W(\mathbf{r}, h) = \frac{1}{(2\pi h^2)^{\frac{3}{2}}} \exp(-\frac{|\mathbf{r}|^2}{2h^2})$.

C. Vorticity Evolution

The evolution of the generated vorticity field is described by updating the vorticities of fluid particles according to the Lagrangian vorticity form of the N-S equations:

$$\frac{d\omega}{dt} = (\omega \cdot \nabla)\mathbf{u} + \mu \nabla^2 \omega \tag{14}$$

where $(\omega \cdot \nabla)\,\mathbf{u}$ denotes vorticity stretching, $\mu\nabla^2\omega$ denotes vorticity diffusion, and $\mathbf{u}$ is the instantaneous velocity.

We solve the vorticity stretching term by modifying its direction with $\delta t(\omega \cdot \nabla)\mathbf{u}$ [12]. The gradient of instantaneous velocity field can be easily computed using SPH summation interpolation method (Equation (3)).

A viscous flow rapidly dissipates vorticity according to the vorticity diffusion term $\mu\nabla^2\omega$. In our particle-based turbulence method, the vorticity diffusion term is calculated by the method of Particle Strength Exchange (PSE) [11], [12], [16], which approximates the Laplacian operator ∇^2 by spreading vorticity from a particle to its neighbors. For each time step Δt, the vorticity increment of a particle i according to the vorticity diffusion process is:

$$\delta\omega_i = \mu\Delta t \nabla^2 \omega_i = \mu\Delta t \sum_j \frac{m_j \omega_{ji}}{\rho_j} \nabla^2 W(\mathbf{x}_{ij}, h) \tag{15}$$

where $\omega_{ji} = \omega_j - \omega_i$.

D. Artificial Vorticity Dissipation

The vorticity field evolved by the vorticity equation (Equation (15)) has very little numerical dissipation in the simulation. And under the action of the vorticity diffusion term, all particles finally have the same non-zero vorticity values. This leads to the non-zero vortical velocity field $\mathbf{u}_v$. Thus, the simulated fluid keeps moving too long and cannot calm down, which make solid-fluid coupling unrealistic and unstable. The same problem exists in the previous methods [11], [12]. To remedy this problem, we provide an artificial vorticity dissipation term to gradually reduce the vorticity strengths of fluid particles over time. Specifically, an SPH particle i is assigned a lifetime value t_i to record how long the vorticity lasts, when it obtains the vorticity ω_i or becomes a vortex particle. Then we use a time-dependent dissipation function $\tau(t) = \exp(-\beta \int_{t_1}^{t_2} \phi(t) dt)$, which is an adaptation of the heat radiance function in [39], to calculate the rest of the vorticity of particle i at time $t + \Delta t$:

$$\begin{aligned} \omega_i(t + \Delta t) &= \omega_i(0) \exp(-\beta \int_0^{t+\Delta t} \phi(t') dt') \\ &= \omega_i(0) \exp(-\beta \int_0^{t} \phi(t') dt') \exp(-\beta \int_t^{t+\Delta t} \phi(t') dt') \\ &= \omega_i(t) \exp(-\beta \int_t^{t+\Delta t} \phi(t') dt') \end{aligned} \tag{16}$$

where β is a user-defined constant, $\omega_i(0)$ denotes the generated initial vorticity. We set $\beta = 0.04$, and use $\phi(x) = x$ for computational efficiency. Figure 4 shows the value of our time-dependent dissipation function $\tau(t)$ changes over time.

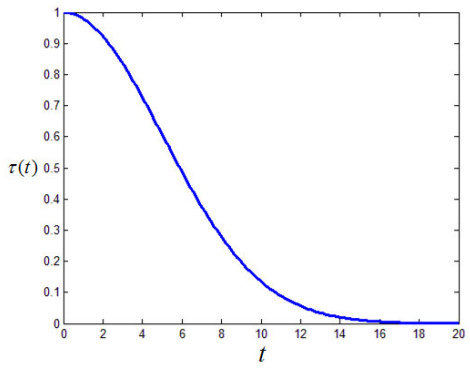

Figure 4. The time-dependent dissipation function $\tau(t)$.

Due to the introduced time-dependent dissipation function $\tau(t)$, the strength of generated vorticity field will dissipate quickly as time goes, which keeps the simulation stable, and makes the fluid look more visually viscous and realistic.

V. Results and Discussion

To assess the advantages of our method, testing cases including both 2D and 3D have been implemented on a PC with an Intel Dual-Core 2.8 CPU and 8GB RAM. In 3D animations, the implicit surface of the fluid is defined using the approach presented in [34], and then the extracted triangle meshes are rendered by Pov-ray. In 2D animations, the effect of gravity is ignored. The parameter values of the simulation are documented in Table I.

Table I
Parameter values in the experiments

Properties	2D scene	3D scene	Unit
Time step(Δt)	0.002	0.002	s
Initial spacing(r_0)	0.008	0.02	m
Support radius(h)	0.016	0.04	m
Density(ρ_0)	10^2-10^4	10^2-10^4	kg/m^3
Viscosity(μ)	0.6	0.3	$Pa \cdot s$

Figure 5 is an animation scenario that the fluid flows past a fixed black cube, which demonstrates our turbulence synthesis model. The velocity of fluid is $20m/s$. Red to yellow to blue colors illustrate that the magnitude of $\omega_i(t)$ changes from large to small. Our vorticity production method approximating boundary layer theory in PCISPH seeds vortex particles behind the cube (top left). The fluctuating velocity $\mathbf{u}^v$, which is calculated by using SPH-like summation interpolant formulation of the Biot-Savart law, is enforced on the mean velocity field $\mathbf{U}$ (top right). The spread of vorticities between fluid particles is solved by the vorticity diffusion term(bottom left). Our artificial vorticity dissipation term gradually reduces the vorticity strengths over time (bottom right). In the simulation we used up to 6 k fluid particles and 210 solid particles, and the average computational time per frame is 30 $msec$.

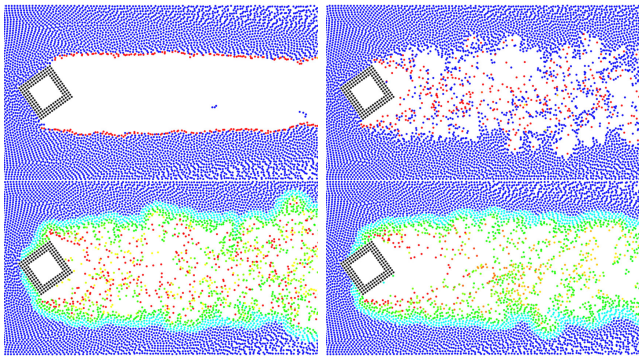

Figure 5. Our turbulence model. Vortex particle seeding (top left), synthesis turbulence (top right), vorticity diffusion (bottom left), artificial dissipation (bottom right).

In the 2D scenario shown by Figure 6, we employ the turbulence synthesis method to simulate the turbulent details induced by a fixed deformable propeller. It can be observed that the generated turbulent details can significantly improve the visual realism of the fluid-solid coupling.

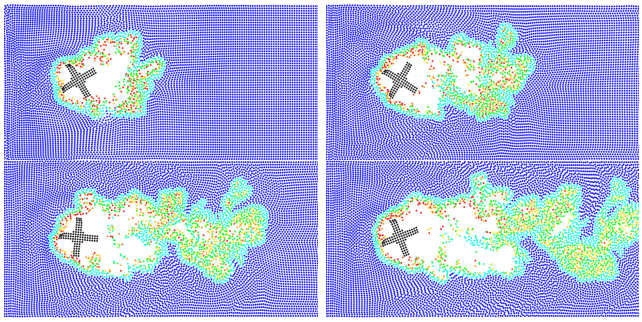

Figure 6. Turbulent details induced by a fixed deformable propeller.

Figure 7 shows that a dolphin swims across the water pool. Compared to the simulation only using the vorticity diffusion term (top), the artificial vorticity dissipation term dissipates the strength of generated vorticity field over time, which makes the turbulence behind the dolphin more stable and plausible (bottom). For the scenario with 100 k fluid particles and 5 k solid particles, the average computational time per frame is 1.5 s.

In Figure 8, we give the comparison of Yuan's turbulence method [12] (top) with our model (bottom). Due to the stochastically seeded vortex particles, Yuan's method creates some turbulence ahead of the moving bar, which makes the simulation unrealistic. Our method produces visually plausible turbulence behind the elastic bar. In addition, the artificial vorticity dissipation scheme make the turbulence generated by our method more stable and realistic.

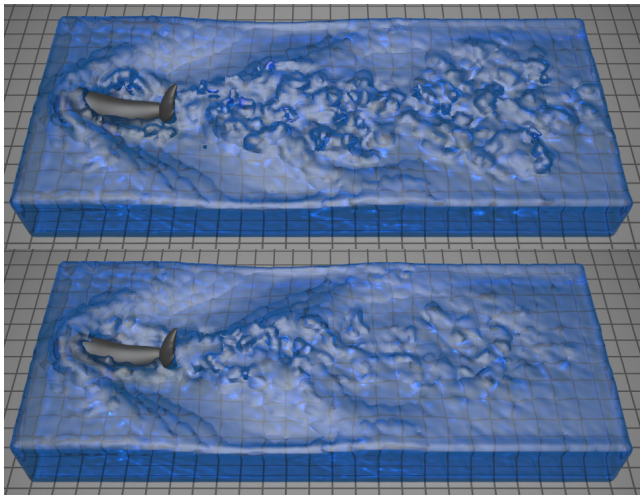

Figure 7. Dolphin swimming. Without (top) and with (bottom) artificial vorticity dissipation.

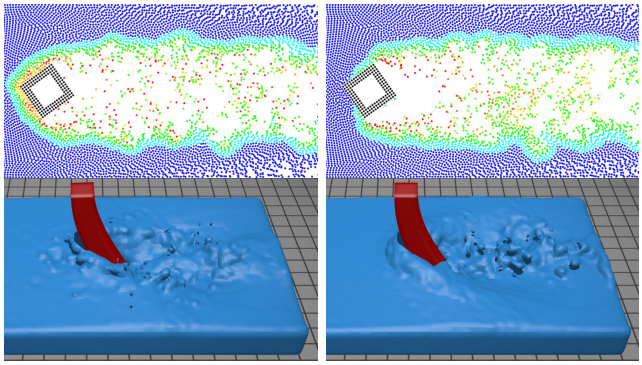

Figure 8. 2D (top) and 3D (bottom) comparisons between Yuan's method [12] (left) and our method (right).

Figure 1 shows that a deformable bar interacts with water. In Figure 1(a), the MSM-based bar is simply coupled to PCISPH water without any turbulence model, which produces too smooth water surface and lacks small-scale details; Figure 1(b) shows the results of adopting adaptive sampling method [3] in our solid-fluid coupling. The fluid particles around the bar are split into one tenth of the original size. The coupling creates bigger waves and more water splashes, however, the turbulent details still cannot be simulated; Figure 1(c) shows the turbulent details simulated by our method. Our turbulence synthesis model produces visually plausible turbulence behind the elastic bar by approximating the boundary layer theory. The artificial vorticity dissipation makes the water calm down finally, which can significantly improve the visual realism of the simulation. For a better comparison, please see all these animations in the supplemental video.

Limitations and Future Work We provide a visually plausible way to stably model the turbulent details behind solid bodies. However, the turbulence synthesis model does not make energy be conserved in the generation and dissipation of the vorticity field. So our method cannot be applied to virtual surgery systems which require physically accurate coupling results.

Bubbles and foams in solid-fluid coupling are also crucial for the realistic fluid simulation. In the future, we aim to incorporate these details into our framework. In addition, the computation will be implemented entirely on GPUs for interactive rates.

VI. Conclusions

We have proposed a novel visually plausible method to synthesize the turbulent details behind the objects in PCISPH fluids. A vorticity production model, which approximating the boundary layer theory in SPH fluid, is employed to create visually realistic vorticities behind the moving objects. And the fluctuating velocities from the vorticity field are enforced on the global flow by an SPH-like summation interpolant formulation of the Biot-Savart law. Under the influences of an artificial vorticity dissipation term introduced into the vorticity N-S equation, the vorticity field is stably evolved. Our turbulence synthesis method successfully makes particle-based fluid animations more visually realistic.

Acknowledgment

This work is supported by the National 863 Program of China (grant no. 2012AA011803), and the National Natural Science Foundation of China (grant no. 61300066).

References

[1] B. Kim and Y. Liu and I. Llamas and J. R. Rossignac, *Flowfixer: Using BFECC for fluid simulation*. In Proceedings of ACM SIGGRAPH, 2005, 51-56.

[2] J. Molemaker and J. M. Cohen and S. Patel and J. Noh, *Low viscosity flow simulations for animation*. In Proceedings of ACM SIGGRAPH/Eurographics symposium on Computer animation, 2008, 9-18.

[3] B. Adams and M. Pauly and R. Keiser and L. J. Guibas, *Adaptively sampled particle fluids*. ACM Transactions on Graphics, 2007, 48-54.

[4] B. Solenthaler and M. Gross, *Two-Scale Particle Simulation*. In Proceedings of ACM SIGGRAPH, 2011, 811-818.

[5] R. Bridson and J. Houriham and M. Nordenstam, *Curl-noise for Procedural Fluid Flow*. In Proceedings of ACM SIGGRAPH, 2007.

[6] T. Kim and N. Thürey and D. James and M. Gross, *Wavelet turbulence for fluid simulation*. ACM Transactions on Graphics, 2008, 50-57.

[7] H. Schechter and R. Bridson, *Evolving sub-grid turbulence for smoke animation*. In Proceedings of ACM SIGGRAPH/Eurographics symposium on Computer animation, 2008, 1-7.

[8] R. Narain and J. Sewall and M. Carlson and M. C. Lin, *Fast animation of turbulence using energy transport and procedural synthesis*. ACM Transactions on Graphics, 2008, 1661-1668.

[9] J. C. Yoon and H. R. Kam and J. M. Hong and S. J. Kang and C. H. Kim, *Procedural synthesis using vortex particle method for fluid simulation*. Computer Graphics Forum, 2009, 1853-1859.

[10] T. Pfaff and N. Thurey and A. Selle and M. Gross, *Synthetic turbulence using artificial boundary layers*. In Proceedings of ACM SIGGRAPH Asia, 2009.

[11] B. Zhu and X. Yang and Y. Fan, *Creating and preserving vortical details in sph fluid*. Computer Graphics Forum, 2010, 2207-2214.

[12] Z. Yuan and Y. Zhao and F. Chen, *Incorporating stochastic turbulence in particle-based fluid simulation*. The Visual Computer, 2012, 435-444.

[13] R. Fedkiw and J. Stam and H. W. Jensen, *Visual simulation of smoke*. In Proceedings of ACM SIGGRAPH/Eurographics symposium on Computer animation, 2001, 154-156.

[14] F. Losasso and F. Gibou and R. Fedkiw, *Simulating water and smoke with an oc-tree data structure*. In Proceedings of ACM SIGGRAPH, 2004, 457-462.

[15] A. Selle and N. Rasmussen and R. Fedkiw, *A vortex particle method for smoke, water and explosions*. ACM Transactions on Graphics, 2005, 910-914.

[16] Q. Yu and F. Neyret and E. Bruneton and N. Holzschuch, *Scalable real-time animation of rivers*. In Proceedings of EUROGRAPHICS, 2009, 125-131.

[17] T. Pfaff and N. Thuerey and J. Cohen and S. Tariq and M. Gross, *Scalable fluid simulation using anisotropic turbulence particles*. ACM Transactions on Graphics, 2010.

[18] I. Pars and M. J. Kim, *Vortex fluid for gaseous phenomena*. In Proceedings of ACM SIGGRAPH/Eurographics symposium on Computer animation, 2005, 261-270.

[19] T. Jang and H. Hwang and S. Cha and M. You, *Simulating Water Turbulence in SPH Fluids*. In Proceedings of Computer Graphics International, 2012.

[20] M. Müller and D. Charypar and M. Gross, *Particle-based fluid simulation for interactive applications*. In Proceedings of ACM SIGGRAPH/Eurographics symposium on Computer animation, 2003, 154-159.

[21] M. Müller and S. Schirm and M. Teschner and B. Heidelberger and M. Gross, *Interaction of fluids with deformable solids*. Computer Animation and Virtual Worlds, 2004, 159-171.

[22] T. Harada and S. Koshizuka and Y. Kawaguchi, *Smoothed particle hydrodynamics on GPUs*. In Proceedings of Computer Graphics International, 2007, 63-70.

[23] L. P. Yang and S. Li and A. M. Hao and H. Qin, *Realtime Two-way Coupling of Meshless Fluids and Nonlinear FEM*. In Proceedings of Pacific Conference on Computer Graphics and Applications, 2012, 2037-2046.

[24] H. Schechter and R. Bridson, *Ghost SPH for Animating Water*. In Proceedings of ACM SIGGRAPH, 2012, 41-50.

[25] B. Solenthaler and R. Pajarola, *Density Contrast SPH Interfaces*. In Proceedings of ACM SIGGRAPH/Eurographics symposium on Computer animation, 2008, 211-218.

[26] M. Becker and M. Teschner, *Weakly compressible SPH for free surface flows*. In Proceedings of ACM SIGGRAPH/Eurographics symposium on Computer animation, 2007, 209-217.

[27] B. Solenthaler and R. Pajarola, *Predictive-Corrective Incompressible SPH*. In Proceedings of ACM SIGGRAPH, 2009.

[28] X. He and N. Liu and S. Li and H. Wang and G. Wang, *Local Poisson SPH For Viscous Incompressible Fluids*. Computer Graphics Forum, 2012, 1948-1958.

[29] S. Hadap and N. Magnenat-Thalmann, *Modeling dynamic hair as a continuum*. In Proceedings of EUROGRAPHICS, 2001, 329-338.

[30] K. Iwasaki and H. Uchida and Y. Dobashi, *Fast particle-based visual simulation of ice*. In Proceedings of Pacific Conference on Computer Graphics and Applications, 2010, 2215-2223.

[31] S. Clavet and P. Beaudoin and P. Poulin, *Particle-based viscoelastic fluid simulation*. In Proceedings of ACM SIGGRAPH/Eurographics symposium on Computer animation, 2005, 219-228.

[32] M. Ihmsen and A. Wahl and M. Teschner, *A Lagrangian framework for simulating granular material with high detail*. Computer and Graphics, 2013, 800-808.

[33] M. Müller and B. Heidelberger and M. Teschner and M. Gross, *Meshless Deformations Based on Shape Matching*. In Proceedings of ACM SIGGRAPH, 2005.

[34] B. Solenthaler and J. Schlafli and R. Pajarola, *A Unified Particle Method for Fluid-solid Interactions*. Computer Animation and Virtual Worlds, 2007, 69-82.

[35] N. Akinci and M. Ihmsen and G. Akinci and B. Solenthaler and M. Teschner, *Versatile rigid-fluid coupling for incompressible SPH*. ACM Transactions on Graphics, 2012, 59-68.

[36] S. B. Pope, *Turbulent flows*. Cambridge university press, 2000.

[37] M. Müller and B. Solenthaler and R. Keiser and M. Gross, *Particle-based fluid-fluid interaction*. In Proceedings of ACM SIGGRAPH/Eurographics symposium on Computer animation, 2005, 237-244.

[38] G. H. Cottet and P. D. Koumoutsakos, *Vortex methods: theory and practice*. Cambridge university press, 2000.

[39] S. W. Hasinoff and K. N. Kutulakos, *Photo-consistent reconstruction of semitransparent scenes by density-sheet decomposition*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2007, 870-885.