Pacific Graphics 2016
E. Grinspun, B. Bickel, and Y. Dobashi
(Guest Editors)

Volume 35 (2016), Number 7

A Multilevel SPH Solver with Unified Solid Boundary Handling

Tetsuya Takahashi and Ming C. Lin

The University of North Carolina at Chapel Hill
http://gamma.cs.unc.edu/MultiLevelSPH/

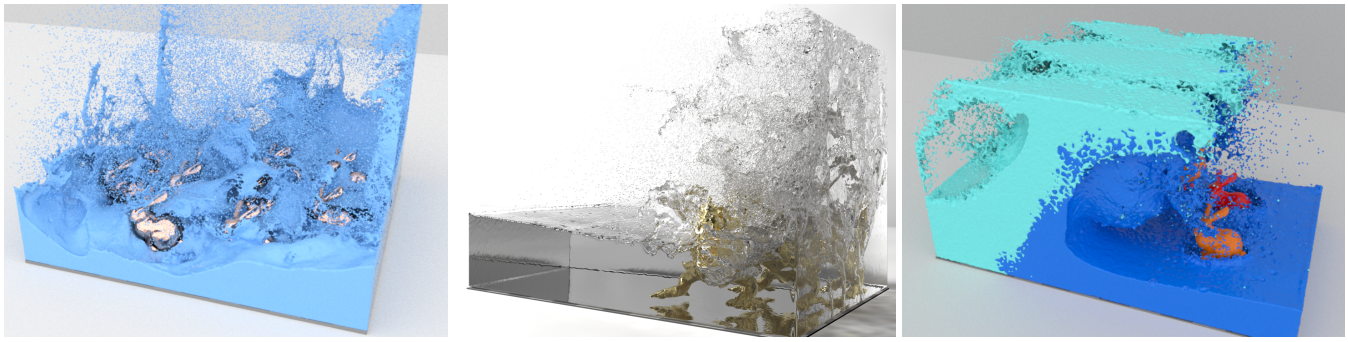

Figure 1: *High-resolution incompressible fluids simulated with our multilevel solver. Our method outperforms other particle-based solvers in the pressure solve, and its computational cost scales nearly linearly with respect to the number of particles.*

Abstract

We propose a geometric multilevel solver for efficiently solving linear systems arising from particle-based methods. To apply this method to particle systems, we construct the hierarchy, establish the correspondence between solutions at the particle and grid levels, and coarsen simulation elements taking boundary conditions into account. In addition, we propose a new solid boundary handling method to solve a pressure Poisson equation in a unified manner. We demonstrate that our method can handle general fluid simulation scenarios including two-way fluid-solid coupling, and the computational cost of this new solver scales nearly linearly with respect to the number of unknowns, unlike previous solvers for particle-based methods.

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

1. Introduction

Lagrangian particle-based methods, such as Smoothed Particle Hydrodynamics (SPH), have become popular because of the attractive advantages of particles, e.g., conceptually simpler discretization and collision handling at the particle level, and thus particle-based methods have been extensively adopted in fluid simulation [IOS*14]. To generate realistic liquid behaviors, enforcing liquid incompressibility is essential, and various particle-based methods have been proposed [KTO96, CR99, BT07, SBH09, SP09, RWT11, HLL*12, BLS12, MM13, ICS*14, BK15]. In particular, solving a pressure Poisson equation (PPE) has been proven to be an effective approach, and the PPE is commonly solved with a stationary iterative solver (e.g., Jacobi method) or Krylov method (e.g., Conjugate Gradient (CG)). In addition to enforcing the incompressibility, discretizing fluids at a high resolution is also important and contributes to the quality of liquid behaviors avoiding numerical dissipations and producing fine details near fluid surfaces. However, enforcing the incompressibility under high-resolution discretization is likely to be computationally challenging because previously used solvers (e.g., Jacobi method and CG) do not scale well and require more iterations to solve the PPE as the number of particles increases.

To address this problem, in the Eulerian grid-based approach, scalable multigrid (MG) methods have been used [MST10, CM11, CM12, FWD14, WMRSF15] with the regular Cartesian grid, which can also be used to construct the hierarchy. This multilevel approach is also adopted for deformable body simulation by constructing the hierarchy from embedded grid structures [ZSTB10, MZS*11] or computing nearly optimal coarser-level structures from finer-levels [Mül08, WOR10]. Unlike deformable objects, however, it is difficult to apply the multilevel approach to particle-based methods because the connectivity of particles changes at every simulation step, requiring expensive remeshing to keep the quality of coarser-level meshes and to avoid mesh tangling. Although Cummins and Rudman [CR99] adopted a fixed Cartesian grid for particle-based fluid simulation, solutions at the particle and grid levels are generally inconsistent because of the different discretization approaches, and thus their method can diverge, stagnate, or fail to achieve optimal efficiency of multilevel solvers.

We propose a new multilevel method for efficiently solving the PPE arising from the particle-based fluid simulation. Our method offers the following technical contributions:

- **A multilevel method for particle systems** that constructs hierarchies, establishes the correspondence between solutions at the particle and grid levels, and coarsens simulation elements taking boundary conditions into account.
- **A new solid-handling method for the PPE** to ensure the solvability of the system regardless of the particle configurations while approximating the original solution to make it solvable with our multilevel approach.

Our solver is able to achieve up to one order of magnitude performance gain in the pressure solve as compared to one of the state-of-the-art particle-based solver, implicit incompressible SPH (IISPH) [ICS*14], and the cost of our method scales nearly linearly with respect to the number of unknowns in the system, considerably outperforming existing particle-based methods.

2. Related Work

Particle-based methods have been extensively studied, and we refer to [IOS*14] for their basis and applications. We focus our discussions on particle-based methods closely related to ours.

2.1. Incompressibility

Early works used an equation of state to compute pressures from densities [DG96, MCG03]. In particular, SPH methods using the Tait equation [Mon94, BT07], known as weakly compressible SPH (WCSPH), can generate higher pressures mitigating the volume compression of fluids, and thus have been widely adopted even in recent SPH works because of the simplicity and effectiveness [HWZ*14, RLY*14]. However, since WCSPH usually produces excessively high pressures and thus strong pressure forces, smaller time steps are necessary for stable fluid simulations. Additionally, it is also undesirable that WCSPH cannot explicitly specify the tolerance of the fluid compression.

To address these issues, several approaches that globally solve a system with iterative methods have been developed. Adopting the pressure projection commonly used in the Eulerian approach [Bri15], incompressible SPH (ISPH) [CR99, SL03] and Moving Particle Semi-implicit (MPS) [KTO96, PTB*03] were proposed. Similar projection-based approaches were also proposed using a Voronoi diagram [SBH09] and a power diagram [dGWH*15].

Another recent trend is to locally solve a system with iterative methods. Predictive-corrective incompressible SPH (PCISPH), which iteratively predicts and corrects particle density in a Jacobi manner, was proposed by Solenthaler and Pajarola [SP09]. This predictive-corrective scheme was also adopted in local Poisson SPH [HLL*12]. Macklin and Müller [MM13] proposed position-based fluids (PBF) adopting position-based dynamics (PBD) [MHHR07] for density constraints, and improved the robustness. Another constraint-based solver was also proposed by Bodin et al. [BLS12] to improve the accuracy. To further improve the efficiency and robustness compared to PCISPH, Ihmsen et al. [ICS*14] proposed IISPH, which decomposes the Laplacian operator in the PPE into the divergence and gradient operators to accelerate the propagation of updated pressures with the Jacobi method. However, since IISPH generates not diagonally dominant systems, it does not produce smooth pressure fields, and CG (which is generally faster than the Jacobi method) cannot be applied to IISPH [TDNL16]. In [ICS*14], it is demonstrated that IISPH outperforms ISPH when both approaches use the Jacobi method. Cornelis et al. [CIPT14] proposed combining IISPH with Fluid Implicit Particle (FLIP) [ZB05] to further improve the performance. Bender and Koschier [BK15] proposed divergence-free SPH, which enforces not only the density-invariance condition but also divergence-free condition, similar to [KS14], to reduce the density deviations at the next step.

2.2. Multilevel Particles

To improve the efficiency by reducing the number of particles, approaches using spatially adaptive particles have been proposed. Adams et al. [APKG07] used different sizes of particles to allocate more computational resources to regions, which are important in terms of visual quality and fluid dynamics, e.g., surfaces and near solid objects. This method was extended to improve the robustness by blending particle properties over time [OK12]. To avoid direct interactions between particles at different levels, Solenthaler and Gross [SG11] proposed separating domains for fine- and coarse-scale particles. Although these approaches can reduce the number of particles, energy dissipation (i.e., damping) can be introduced into the simulation because of the coarser level particles and may negatively affect liquid behaviors.

Cummins and Rudman [CR99] used MG to solve the PPE with ISPH adopting the Cartesian grid for the hierarchy construction. However, they did not include a Dirichlet boundary condition at all, which is necessary to handle general liquid simulation scenarios. In addition, they used a mirroring approach for the Neumann boundary condition that copies physical quantities to the opposite side of solid boundaries, limiting its applicability to rectangular domains only. Thus, how to address complex Dirichlet and Neumann boundary conditions in the MG setting is unclear. Moreover, they did not address discrepancies of solutions at the particle and grid levels, which can occur due to various factors, such as different discretization methods, particle irregularities, and kernel definitions. Consequently, this approach can diverge, stagnate, or fail to achieve the optimal efficiency of MG. Their experiments were limited to 2D simulations and were never tested on fine-scale scenarios. Thus, how this method works in such scenarios was not demonstrated. The method of Raveendran et al. [RWT11] can be considered as a two-level approach consisting of the finest particle level and the coarsest grid level, and uses the result of the pressure projection on the grid to accelerate the convergence of an iterative solver on the particles. However, since the coarse pressure projection is not sufficiently accurate and does not consider the solution discrepancies, this approach cannot significantly reduce the number of solver iterations on the particles.

3. SPH Fluid Simulation

We first briefly describe the simulation algorithm of our fluid solver, which is similar to that of the Eulerian approach (Section 3.1), and then explain the PPE for particle-based fluids clarifying differences from the Eulerian approach (Section 3.2).

As our underlying fluid solver, we employ ISPH [CR99, SL03] since ISPH constructs a linear system whose matrix is symmetric positive (semi-) definite (SPD) and produces smooth pressures allowing us to solve the system with either CG or MG. Other methods, which have properties similar to ISPH, can be used as well.

3.1. Simulation Algorithm

In the Lagrangian setting, incompressible flow for particles can be described by the continuity equation $\frac{d\rho_i}{dt} + \rho_i \nabla \cdot \mathbf{u}_i = 0$, and the Navier-Stokes equations $\frac{d\mathbf{u}_i}{dt} = -\frac{1}{\rho_i}\nabla p_i + \frac{\mathbf{F}_i^{\mathrm{v}}}{m_i} + \frac{\mathbf{F}_i^{\mathrm{ext}}}{m_i}$, where ρ_i denotes density of particle i, t time, $\mathbf{u}_i$ velocity, p_i pressure, $\mathbf{F}_i^{\mathrm{v}}$ viscosity force, m_i mass, and $\mathbf{F}_i^{\mathrm{ext}}$ external force. Similar to the Eulerian approach, we take the operator splitting to enforce the fluid incompressibility. First, we find neighbor fluid and solid particles to compute the density. Then, we predict the intermediate velocity $\mathbf{u}_i^*$ and the intermediate density ρ_i^*. Next, we solve the PPE to obtain pressure. Finally, we compute pressure forces and integrate velocities and positions. For more algorithm details, we refer to Section 3.4 of the survey paper [IOS*14].

3.2. PPE for Particle-based Methods

Unlike the traditional Eulerian approach that enforces the divergence-free condition [Bri15], we use the density-invariance condition to avoid the volume drift, as commonly employed in the particle-based methods [KTO96, SL03, ICS*14, BK15] . By combining the velocity change caused by pressure forces $\frac{d\mathbf{u}_i}{dt} = -\frac{1}{\rho_i^{t+1}}\nabla p_i$ and the density prediction based on the continuity equation $\frac{\rho_i^{t+1}-\rho_i^*}{\Delta t} = -\rho_i^{t+1}\nabla \cdot \mathbf{u}_i^*$, ($\Delta t$: time step), we obtain the PPE as $-\nabla^2 p_i = \frac{\rho_i^* - \rho_0}{\Delta t^2}$ (ρ_0: rest density) with Dirichlet boundary condition $p_i = 0$ on free surfaces and Neumann boundary condition $\frac{dp_i}{d\mathbf{n}_i} = 0$ ($\mathbf{n}_i$: normal) on solid boundaries [SL03]. This PPE can be rewritten in the matrix form as $\mathbf{Ap} = \mathbf{b}$, where $\mathbf{A}$ denotes a coefficient matrix, and $\mathbf{p}$ and $\mathbf{b}$ concatenation of pressures and source terms (right hand side of the PPE), respectively. Since the PPE uses the density-invariance condition, which generally does not satisfy the compatibility condition (i.e., the summation of right hand side is not equal to 0), and $\mathbf{A}$ has null-space (i.e., $\mathbf{A}$ is rank deficient) when a standard Laplacian discretization is used (as done in [CR99, SL03, KTO96, PTB*03]), a Dirichlet boundary condition is necessary for each of the groups consisting of neighboring particles to ensure the solvability of the PPE [Bri15, TDNL16].

Because of the negative source term, which is likely to occur at particles with a smaller number of neighbor particles (e.g., near free surfaces), solving the PPE without special care generates negative pressures leading to the tensile instability in particle-based methods [Mon00, SB12, HWZ*14]. Therefore, $\mathbf{p} \geq 0$ must be simultaneously enforced to avoid the tensile instability turning the linear system $\mathbf{Ap} = \mathbf{b}$ into a Linear Complementarity Problem (LCP) $\mathbf{Ap} = \mathbf{b} \perp \mathbf{p} \geq 0$. Although stationary iterative methods can relatively easily solve the LCP with the clamping approach (e.g., projected Jacobi), solving the LCP is numerically more difficult than linear systems, especially with Krylov methods, which prohibit clamping negative pressures in each iteration [ICS*14], and thus require specialized techniques [DS05]. Even with MG solvers, convergence can be delayed [CM12].

Since solving the LCP is more costly and complex, we approximate the solutions of the LCP by solving a linear system with appropriately set a Dirichlet boundary condition, similar to [KTO96, SL03]. To ensure $\mathbf{p} \geq 0$, we treat fluid particles whose source term is negative as Dirichlet boundary condition (see [TDNL16]). Note that this treatment enforces $\mathbf{b} \geq 0$, and thus $\mathbf{p} \geq 0$ is guaranteed. In practice, we set particles as a Dirichlet boundary condition if $\rho_i^* < 0.99\rho_0$ to avoid erroneous classifications because of the particle irregularity and clamp negative pressures to 0 after the pressure solve to avoid the tensile instability and particle adhesion.

4. Solid Boundary Handling

We first describe the PPE discretization based on ISPH, its issues, and previous approaches (Section 4.1), and then present our solid boundary handling method (Section 4.2). Note that our solid boundary handling can be used without MG while it allows us to solve the PPE with MG by approximating the original solution.

To solve the PPE with solid boundaries on moving particles, it is necessary to adaptively assign roles to the particles, and we classify particles as follows. We call solid particles for the Neumann boundary condition *Neumann particles* (rendered as beige), fluid particles without any neighbors *isolated particles* (magenta), fluid particles used for the Dirichlet boundary condition $p_i = 0$ (i.e., if $\rho_i^* < 0.99\rho_0$) *Dirichlet particles* (blue), fluid particles without fluid neighbors and with solid neighbors *separated particles* (green), and otherwise *Poisson particles* (cyan). Since isolated particles always satisfy the condition of Dirichlet particles, we set pressures of isolated particles as 0, similar to Dirichlet particles. While we analytically set pressures of separated particles excluding them from the linear system (see Section 4.2), we treat pressures of Poisson particles as unknown variables.

4.1. ISPH Discretization

According to [IOS*14], $-\nabla^2 p_i$ is discretized as $\sum_j a_{ij}(p_i - p_j)$, where $a_{ij} = -(V_i + V_j)\frac{\mathbf{x}_{ij}\cdot\nabla W_{ij}}{\|\mathbf{x}_{ij}\|^2 + 0.01h^2} > 0$ with the kernel definition in [MCG03] (j: index for fluid neighbors, V: volume, $\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$, and h: kernel radius). When we consider fluid and solid particles assuming that solid particle pressures are definable, $-\nabla^2 p_i = -\nabla^2 p_i^{\mathrm{fluid}} - \nabla^2 p_i^{\mathrm{solid}} = \sum_j a_{ij}(p_i - p_j) + \sum_s a_{is}(p_i - p_s)$ (s: index for solid neighbors). When Neumann boundary condition $\frac{dp_i}{d\mathbf{n}_i} = 0$, i.e., $p_i = p_s$, is applied, we obtain $-\nabla^2 p_i = \sum_j a_{ij}(p_i - p_j)$.

With this formulation, unfortunately, particle configurations that cannot determine p_i occur because the PPE does not satisfy the compatibility condition. Specifically, p_i cannot be determined, when particle i is a separated particle, or a Poisson particle with no channel (via other neighboring Poisson particles) to at least one Dirichlet particle [TDNL16]. One possible approach is to separately address these particles after we check particle connectivities. However, it is hard to efficiently and exactly check particle connectivities (e.g., using the flood fill) because we need to propagate geometric information one by one over all particles. If particle pressures cannot be determined, for example, a widely used collision handling method proposed in [AIA*12], which depends on fluid particle pressures, cannot be used.

Some previous works [KTO96, PTB*03, SL03] converted Neumann particles to Poisson particles to make the PPE solvable by

connecting Poisson and separated particles via converted Poisson particles. However, this approach has some limitations [TDNL16]. First, since Neumann particles are incorporated into the linear system, the size of the system becomes larger leading to increased memory and computation cost. Second, due to the solid particles newly incorporated in the system, pressures can be underestimated resulting in particle penetrations with [AIA*12]. Third, this approach cannot solve a linear system with solid objects floating in the air, which can frequently occur in two-way interactions of fluids and solids, because converted Poisson particles in the objects may not have a channel to Dirichlet particles.

To address these issues, Takahashi et al. [TDNL16] proposed introducing a new pressure term, which generally increases the diagonal components, to make the system solvable. However, their approach does not ensure the diagonal dominance of the resulting linear system, and thus MG and CG may fail to converge. Additionally, the increased diagonal components may erroneously underestimate pressures.

4.2. Unified Handling for Solid Boundary

To handle various simulation scenarios, we take an approach that can always ensure the solvability of the linear system in a unified manner regardless of particle configurations. Our method introduces a new term, which increases the diagonal component, into the left hand side of the PPE to make it solvable while amplifying the right hand side to counteract underestimated pressures compared to the solutions, which can be obtained from the original linear system.

For our solid boundary handling scheme, we employ the method of [AIA*12]. This method puts one layer of particles on solid boundaries defining $\delta_i = \frac{1}{\sum_s W_{is}}$, which adjusts the contribution of solid particles based on their sampling density to alleviate over sampling (see [AIA*12] for details). According to [ICS*14], when the incompressibility is enforced by the pressure force from solid particles $\mathbf{F}_i^{\mathrm{p,solid}}$, the continuity equation can be discretized as

$$\frac{\rho_0 - \rho_i^*}{\Delta t} = \sum_s \frac{\rho_0}{\delta_s} \Delta \mathbf{u}_i \cdot \nabla W_{is} = \rho_0 \Delta \mathbf{u}_i \cdot \sum_s \frac{1}{\delta_s} \nabla W_{is}, \tag{1}$$

where $\Delta \mathbf{u}_i$ denotes the velocity change due to $\mathbf{F}_i^{\mathrm{p,solid}}$ (i.e., $\Delta \mathbf{u}_i = \frac{\mathbf{F}_i^{\mathrm{p,solid}}}{m_i} \Delta t$). In [AIA*12], the pressure force from solid particles is defined as $\mathbf{F}_i^{\mathrm{p,solid}} = -m_i \sum_s \frac{\rho_0}{\delta_s} \frac{p_i}{\rho_i^2} \nabla W_{is} = -m_i \frac{\rho_0}{\rho_i^2} p_i \sum_s \frac{1}{\delta_s} \nabla W_{is}$, and thus we obtain the following relation from Eq. (1), $\Delta \mathbf{u}_i$ and $\mathbf{F}_i^{\mathrm{p,solid}}$:

$$\frac{\rho_0^2}{\rho_i^2} \| \sum_s \frac{1}{\delta_s} \nabla W_{is} \|^2 p_i = \frac{\rho_i^* - \rho_0}{\Delta t^2}.$$

Since $-\nabla^2 p_i^{\mathrm{solid}}$ can be reformulated as $-\frac{\rho_i}{m_i} \nabla \cdot \left(\frac{m_i}{\rho_i} \nabla p_i^{\mathrm{solid}} \right)$, which corresponds to the combination of the continuity equation and pressure force, we obtain $-\nabla^2 p_i^{\mathrm{solid}} = \frac{\rho_0^2}{\rho_i^2} \| \sum_s \frac{1}{\delta_s} \nabla W_{is} \|^2 p_i$ and

$$\sum_j a_{ij}(p_i - p_j) + \alpha_i p_i = b_i, \tag{2}$$

where $\alpha_i = \frac{\rho_0^2}{\rho_i^2} \| \sum_s \frac{1}{\delta_s} \nabla W_{is} \|^2 \geq 0$ and $b_i = \frac{\rho_i^* - \rho_0}{\Delta t^2}$. Note that while non-negative α_i ensures the diagonal dominance of the system unlike [TDNL16], p_i can be underestimated leading to particle penetrations (see Figure 2 (left)).

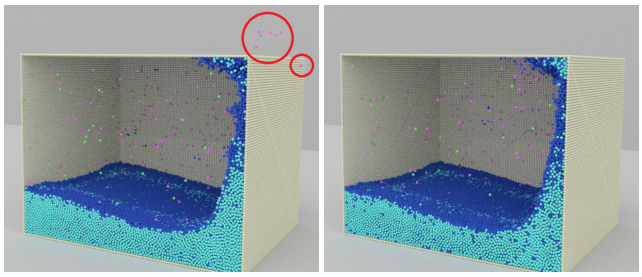

Figure 2: *Dam break. Without the source term amplification, pressures can be underestimated, failing to prevent particle penetrations, as noted by red circles, (left), whereas with the source term amplification, particle penetrations are prevented (right).*

To compensate the underestimation, we approximate the original solution by amplifying the source term (right hand side) as $\beta_i b_i$, with an amplification factor $\beta_i \geq 1$. Since pressures of the original and modified PPE can be computed by $\frac{b_i + \sum_j a_{ij} p_j}{\sum_j a_{ij}}$ and $\frac{\beta_i b_i + \sum_j a_{ij} p_j}{\sum_j a_{ij} + \alpha_i}$, respectively, pressure differences Δp_i are written as

$$\Delta p_i = \frac{b_i + \sum_j a_{ij} p_j}{\sum_j a_{ij}} - \frac{\beta_i b_i + \sum_j a_{ij} p_j}{\sum_j a_{ij} + \alpha_i}.$$

We can exactly solve $\Delta p_i = 0$ and obtain β_i as

$$\beta_i = \frac{b_i \sum_j a_{ij} + \alpha_i b_i + \alpha_i \sum_j a_{ij} p_j}{b_i \sum_j a_{ij}}.$$

Since physical values change only slightly between consecutive steps, and pressures of all particles are always definable with our method, we can assume $p_i \approx \tilde{p}_i$ ($\tilde{p}_i$: pressure at the previous step). Further assuming $p_i \approx p_j$ because of the nature of the PPE's solution, the amplified source term can be computed by

$$\beta_i b_i = \frac{\sum_j a_{ij} + \alpha_i}{\sum_j a_{ij}} b_i + \alpha_i \tilde{p}_i.$$

When $\sum_j a_{ij}$ is very small, $\beta_i b_i$ can be too large leading to infinitely large pressures. To avoid this, we clamp $\frac{\sum_j a_{ij} + \alpha_i}{\sum_j a_{ij}}$ with a limiting factor $\gamma \geq 1$ (we empirically found that $\gamma = 5.0$ works well) and finally obtain the PPE as

$$\sum_j a_{ij}(p_i - p_j) + \alpha_i p_i = \min\left(\gamma, \frac{\sum_j a_{ij} + \alpha_i}{\sum_j a_{ij}} \right) b_i + \alpha_i \tilde{p}_i. \tag{3}$$

For separated particles, β_i cannot be defined because they have no fluid neighbors (i.e., $\sum_j a_{ij} = 0$). In this case, we directly set their pressures without including these particles in the PPE (as Dirichlet boundary condition) based on Eq. (2) clamping negative values as $p_i = \max\left(0, \frac{b_i}{\alpha_i}\right)$.

Our solid boundary handling method introduces $\alpha_i p_i$ into the left hand side of the PPE turning a Poisson equation to a Helmholtz equation when fluid particles are contacting with solid particles. This is numerically similar to adding a Dirichlet boundary condition to the system, and thus makes the system solvable without sat-

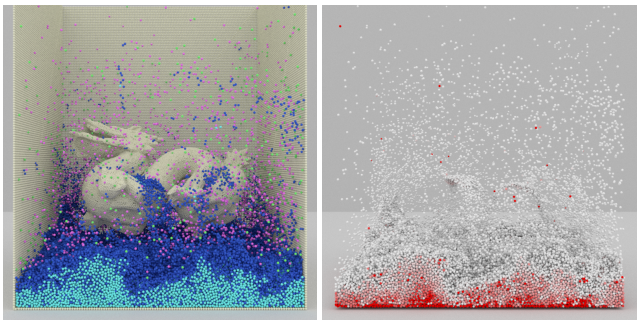

Figure 3: *A cubed fluid dropped onto a solid dragon floating in the air. Particles are color coded based on the classification on the left, and pressure on the right, where white and red represent low and high pressures, respectively. With our method, particle pressures are definable regardless of the particle configurations.*

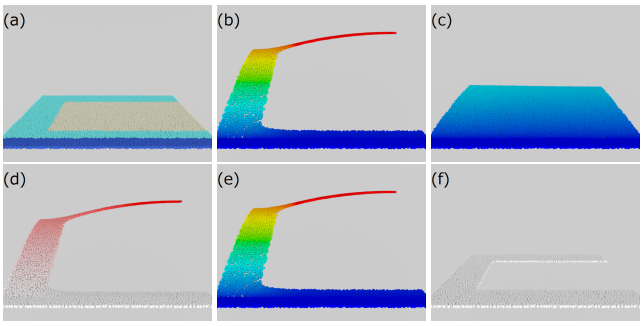

Figure 4: *Comparisons on pressure accuracy with a square domain, where positive source terms are set on Poisson particles. In (b), (c), and (e), blue, green, and red represent low, middle, and high values, respectively, while in (d) and (f), white and red represent low and high values, respectively. In (b), (d), (e), and (f), Neumann particles are not visualized. (a) Scene setup. (b) Exact solution. (c) Solution obtained with the previous approach [SL03]. (d) Solution difference (b) − (c). (e) Solution obtained with our solid boundary handling. (f) Solution difference (b) − (e).*

isfying the compatibility condition regardless of the particle configurations. When the exact solution cannot be globally defined, our method alters the solution based on the pressure force formulation minimizing the errors in regions, where the exact solution can be defined. As compared to [KTO96, PTB*03, SL03], our method constructs smaller systems and produces more accurate pressures. In addition, unlike these previous methods, our method can handle solid objects floating in the air (see Figure 3).

Figure 4 compares our solid handling method (we used an exactly computed p_i for $\tilde{p}_i$ in Eq. (3)) with a previous method [KTO96, PTB*03, SL03] and an exact solution. Since it is not possible to obtain the exact solution in general fluid simulation scenarios as explained in Section 4.1, we experiment with a static 2D square domain, where an exact solution can be defined. The previous method that treats solid Neumann particles as Poisson particles connect Poisson and Dirichlet particles, newly generating shorter paths. Consequently, pressures are restricted to low values, and this approach underestimates pressures (see (c) and (d)). On the other hand, our method can more accurately compute pressures with imperceptible errors only, as shown in (e) and (f).

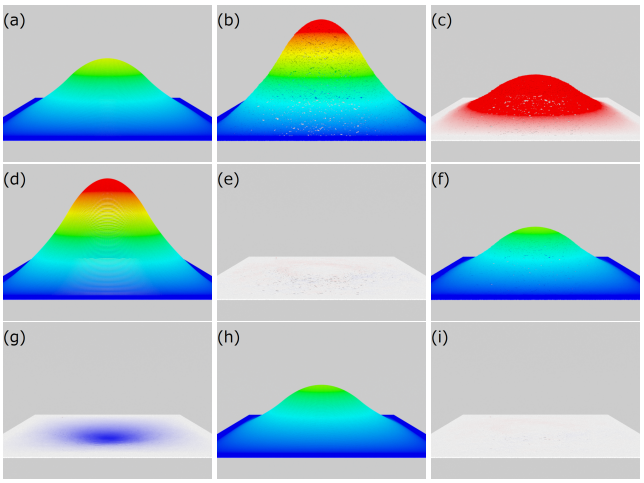

Figure 5: *Correspondence check with a square domain, whose central regions have positive source terms, and whose edges are set as Dirichlet boundary condition. In (a), (b), (d), (f), and (h), blue, green, and red represent low, middle, and high values, respectively, while in (c), (e), (g), and (i), blue, white, and red represent low, middle, and high values, respectively. (a) Solution on the grid. (b) Solution on the particles, which is larger than (a). (c) Solution difference (b) − (a). (d) Solution on the grid corrected with λ^{opt} approximating (b). (e) Solution difference (b) − (d). (f) Solution on the particles, which is smaller than (a). (g) Solution difference (f) − (a). (h) Solution on the grid corrected with λ^{opt} approximating (f). (i) Solution difference (f) − (h).*

5. Multilevel Particle-based Solver

In this section, we describe our multilevel solver, which is specifically designed for particle-based methods. For fundamentals of MG solvers, we refer to [BHM00, TOS00].

5.1. Hierarchical Structure Construction

Constructing the hierarchical structures for particle-based methods is difficult mainly because of irregular particle positions and changing particle neighbors (i.e., connectivities). For mesh coarsening, Müller [Mül08] proposed to successively coarsen finer-levels preserving only representative particles, and Sacht et al. [SVJ15] presented a method for constructing the hierarchy of surface meshes such that coarser levels are contained by their finer levels. However, these approaches are essentially designed for unstructured meshes with no frequent connectivity changes, and thus they are computationally expensive for particle-based methods. Additionally, unstructured fine-level meshes produce unstructured coarser levels, where the number of edges is likely to be large, thereby leading to higher smoothing and residual computation cost as compared to structured meshes. Moreover, unstructured meshes are not suitable for parallelization since we need to use a Jacobi smoother, which is less effective than Gauss-Seidel (GS) and Red-Black GS (RBGS) smoothers [BHM00, TOS00]. Taking these factors into account, we employ a static Cartesian grid. To simplify boundary handling, we store pressures at the center of each cell.

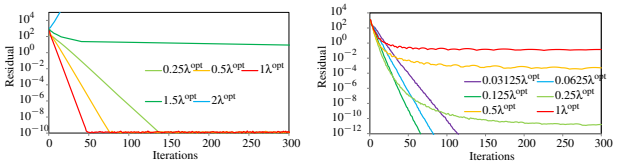

Figure 6: *Convergence profile with different λ for V-cycle (left) and MGCG (right).*

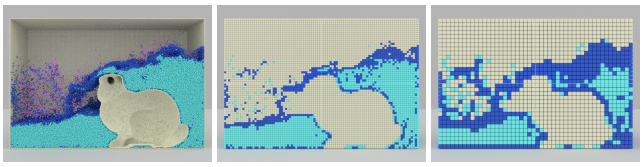

Figure 7: *Cutaway views of fluid simulation with our coarsening scheme. Particle level, the finest grid level, and the second finest grid level from left to right.*

5.2. Particle-Grid Correspondence

While the Cartesian grid has desirable properties for our method, unlike a carefully designed hierarchy, this simple choice of the regular grid introduces a new problem: solutions at the particle and grid levels are not consistent (see Figure 5), mainly because of the inconsistency of discretization at particle and grid levels, particle irregularities, kernel definitions, accuracy, and parameters. For example, when we use SPH to estimate physical values at both grid and particle levels, the estimated values cannot be consistent since grid points are not uniformly surrounded by neighbors at the grid level. If we use Finite Difference (FD) and SPH to estimate physical values at grid and particle levels, respectively, the estimated values can also be different because FD and SPH are different discretization methods. This problem is crucial, and if the solutions at the particle and grid levels are inconsistent, MG solvers converge slowly, stagnate, or diverge. To avoid these problems and hopefully achieve the optimal efficiency of MG solvers, solutions at the particle and grid levels need to be consistent.

We aim to establish the correspondence between solutions at particle and grid levels by modifying the solutions at the grid levels with a source term adjustment. Since solutions at the coarser grid levels always agree to the finest level up to the discretization accuracy, it is sufficient to establish the correspondence between the solutions at the particle and the finest grid levels. We use FD to estimate physical values at grid levels since the SPH-based estimation is more costly.

When the linear system at the particle and the finest grid levels are written as $\mathbf{A}^{\mathrm{P}}\mathbf{p}^{\mathrm{P}} = \mathbf{b}^{\mathrm{P}}$ and $\mathbf{A}^{\mathrm{G}}\mathbf{p}^{\mathrm{G}} = \mathbf{b}^{\mathrm{G}}$, respectively, the solutions are $\mathbf{p}^{\mathrm{P}} = (\mathbf{A}^{\mathrm{P}})^{-1}\mathbf{b}^{\mathrm{P}}$ and $\mathbf{p}^{\mathrm{G}} = (\mathbf{A}^{\mathrm{G}})^{-1}\mathbf{b}^{\mathrm{G}}$. Although it is optimal that each particle pressure p_i^P agrees to the interpolated grid pressure at the particle position p_i^G with a scaling parameter λ_i (i.e., $\mathbf{p}_i^P = \lambda_i \mathbf{p}_i^G$), changing connectivities makes achieving this impractical and complicated. Instead, taking into account that the overall pressure profiles are similar at the particle and grid levels, we use a global scaling factor λ obtaining a modified linear system as $\mathbf{A}^{\mathrm{G}}\tilde{\mathbf{p}}^{\mathrm{G}} = \lambda\mathbf{b}^{\mathrm{G}}$ with modified finest grid level pressure $\tilde{\mathbf{p}}^{\mathrm{G}}$ (i.e., $\tilde{\mathbf{p}}^{\mathrm{G}} = \lambda\mathbf{p}^{\mathrm{G}}$). Estimating the grid pressure at particle positions by trilinear interpolation (denoted by $\hat{I}_{\mathrm{G}}^{\mathrm{P}}$) and scaling λ, we can evaluate pressure error E by

$$E = \|\mathbf{p}^{\mathrm{P}} - \lambda\hat{I}_{\mathrm{G}}^{\mathrm{P}}\mathbf{p}^{\mathrm{G}}\|_2. \tag{4}$$

Note that Eq. (4) includes $\mathbf{p}^{\mathrm{P}}$ that we aim to get and changes over time, and thus we cannot directly minimize E in the simulation. Fortunately, an optimal λ^{opt} that minimizes E basically depends on the distributions of particles only and is not sensitive to the simulation scenarios. Thus, we precompute λ^{opt} with $\mathbf{p}^{\mathrm{P}}$ and $\mathbf{p}^{\mathrm{G}}$ obtained using CG, and determine λ by $\lambda = \gamma\min_t \lambda^t$ (γ: a tunable parameter) with which the MG solvers converge achieving a nearly optimal performance (see Figure 6). To achieve the correspondent solutions at particle and grid levels, we multiply λ when values at the particle level are transferred to the finest grid level (see Section 5.4).

Figure 5 illustrates the effect of established correspondence. While we obtain the same pressure profile with a fixed grid (see (a)), with different particle configurations, we would obtain pressures higher and lower than (a), as shown in (b) and (f), and their differences with respect to (a) are given in (c) and (g), respectively. Our method modifies the pressure on the grid approximating (b) and (f), as shown in (d) and (h), and the differences can be largely corrected to 0, as shown in (e) and (i), respectively.

Figure 6 gives convergence profiles of V-cycle and MGCG for the scene shown in Figure 5. While $\lambda = \lambda^{\mathrm{opt}}$ achieved the best performance with V-cycle, interestingly the convergence of the MGCG with λ larger than $0.125\lambda^{\mathrm{opt}}$ became stagnant and achieved the best performance with $\lambda = 0.125\lambda^{\mathrm{opt}}$, presumably because preconditioning on irregular particle distributions causes overshoots and negatively affects the convergence of CG.

5.3. Linear System Construction

To construct linear systems at the finest grid level, we use the particle classification information. One possible approach is to generate signed distance functions for solid objects and fluid domains following the grid-based approaches [Bri15]. However, this approach would erroneously compute fluid domains penetrating thin solids including volumetric objects whose only surfaces are sampled with particles as commonly done in particle-based methods [AIA*12], and thus cannot approximate the particle level solution. To avoid this problem, we classify grid cells as follows. First, if there is at least one Dirichlet particle in a cell, we classify the cell as a Dirichlet cell (rendered as blue). Second, if there is at least one Poisson particle, we classify the cell as a Poisson cell (cyan). Otherwise, we classify the cell as a Neumann cell (beige). Since isolated and separated particles do not affect the solution of the system, we ignore these particles in our coarsening scheme. Although this approach introduces some discrepancies at the particle and the finest grid levels, Dirichlet boundary condition always remains at the finest grid level, and thus we can ensure the solvability of the system.

To obtain linear systems at coarser grid levels, we use the voxel-based approach proposed by [MST10]. Figure 7 illustrates cutaway views of fluid simulation with our coarsening scheme, at the particle, the finest grid, and the second finest grid levels. It is worth noting that we tested the cut-cell approach [WMRSF15]. However, the convergence was not improved, unlike the case of the grid-based simulations. This is presumably because the geometric consistency was already lost at the approximation of particles with the finest grid. We leave this issue as future work.

5.4. Restriction and Interpolation

While there are several ways to do restriction and interpolation, as previously proposed [MST10, WMRSF15], we need to satisfy $R_l^{l+1} = k(I_{l+1}^l)^T$ (R_l^{l+1}: restriction operator from level $l+1$ to l, I_{l+1}^l: interpolation operator from level l to $l+1$, and k: a constant), known as the Galerkin property [BHM00, TOS00], to use V-cycle as a preconditioner of CG. Among commonly used restriction and interpolation operations satisfying this condition are piecewise constant interpolation and linear interpolation. In [DRW16], it is reported that the convergence rate is faster with linear interpolation than with piecewise interpolation, when the grid-based approaches are used. However, with particle-based methods, we did not observe the accelerated convergence due to the solution inconsistency between particles and grids, and the computational cost of the linear interpolation was much more expensive than the piecewise interpolation. Thus, we use the piecewise constant interpolation.

For the restriction from the particle level to the finest grid level, we compute the average, taking λ into account for correspondence, as $\phi_c = \lambda \frac{\sum_i \phi_i}{\sum_i}$, where ϕ denotes an arbitrary value, and i the index of particles in cell c. For the interpolation from the finest grid level to the particle level, we directly use the value in the cell for the particle as $\phi_i = \phi_c$. Note that we do not use the expensive trilinear interpolation I_G^P defined in Section 5.2.

5.5. Smoother

At the particle level, we use the weighted Jacobi method as a smoother to fully parallelize the pre- and post-smoothing operations, whereas at the grid levels, we use RBGS by taking advantage of the regular grid structures.

In some grid-based MG solvers, e.g., [FWD14, WMRSF15], CG is used to exactly solve the system at the coarsest level. In particle-based methods, however, since we terminate solver iterations based on density error criteria [SP09, ICS*14], which are much more moderate than residual criteria used in the Eulerian method [Bri15], we do not need to obtain the exact solution at the coarsest level, and thus use the multiple RBGS smoothing for efficiency.

5.6. Implementation

We define the grid width Δx at the finest grid level as $\Delta x = h$ since it is preferable to choose a grid width two times larger than particle distances (which are generally $0.5h$ when we set $h = 4r$, where r is the particle radius) [BHM00, TOS00]. Additionally, this choice allows for the reuse of the uniform grid constructed for the neighbor search [IABT11]. We did not achieve faster convergence with a smaller grid width due to the solution gaps between particles and grids, whereas we observed slower convergence with a larger grid width (this corresponds to the method of [RWT11]) due to the less accurate approximation. While it is possible to use the multilevel method as a stand-alone solver, we prefer using it as a preconditioner for CG (i.e., as MGCG) to improve the robustness and efficiency [MST10, FWD14].

6. Results

We measured the performance on a machine with 24-core CPU and 256 GB RAM. For the simulation, we used a constant time-step based on the CFL condition, and the density deviation threshold was set to 0.01%. In the following, we used MGCG with 1 V-cycle preconditioning using 1 pre- and post- smoothing per iteration. Without using an appropriate scaling factor λ, our multilevel solver suffered from divergence or stagnation.

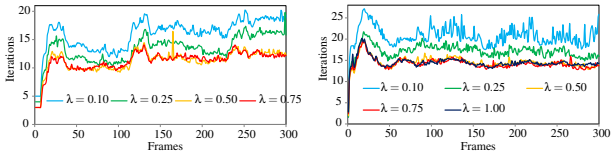

Figure 8: *Iteration profiles for different values of λ in the two scenes (left for Figure 1 (left) and right for Figure 1 (middle)).*

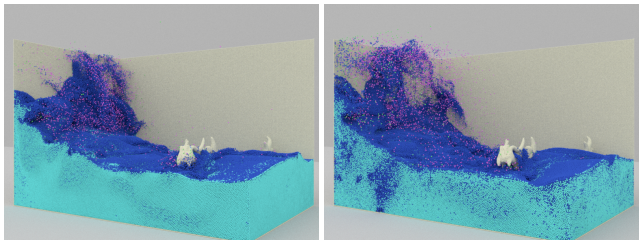

Figure 10: *Visual comparison for our method, i.e., ISPH with MGCG (left) and IISPH (right). Both methods generate comparable fluid behaviors.*

Influence of λ. To demonstrate the influence of λ, we tested several values of λ (0.10, 0.25, 0.50, 0.75, and 1.0) in the two scenes shown in Figure 1 (left) on a grid resolution of 96x64x96 with up to 822.4k particles and Figure 1 (middle) on a grid resolution of 96x64x64 with 1.0M particles. Figure 8 illustrates profiles of average iterations, where the profile of $\lambda = 1.0$ for Figure 1 (left) is not given since our MG solver did not converge. In our experiments, larger λ would cause failure of our solver, whereas smaller λ weakens the effect of the MG preconditioning slowing the convergence.

Convergence speed. We compared our MGCG solver with other solvers in the convergence speed, using the scene shown in Figure 1 (middle) with 3.4M particles on a grid of resolution 144x96x96. For this comparison, we used one of the state-of-the-art particle-based solver IISPH [ICS*14], and CG solvers commonly used in the particle-based methods: CG, Incomplete Cholesky CG (ICCG), and successive over-relaxation CG (SORCG). We used SOR in a Jacobi way (not as in Gauss-Seidel) for parallelization while we applied the IC preconditioner in a serial manner as the application of the IC is inherently serial. Figure 9 (left) illustrates a profile of the density error with respect to the computation time. Although IISPH converges faster at the early stage, the convergence becomes slower at the latter stage due to the use of Jacobi method. Consequently CG solvers become advantageous; in particular, our MGCG solver is 12x faster compared to IISPH. Though the IC and SOR preconditioners reduce the number of CG iterations, the preconditioning is costly and not effective enough to achieve a performance gain. As a result, CG is faster than ICCG and SORCG. On the other hand, our MG preconditioner can significantly reduce the CG iterations and can be efficiently applied. Consequently, our method can achieve a performance gain over CG by a factor of 4.0, which cannot be achieved with relatively simple IC and SOR preconditioners.

Overall performance. We compared our method (ISPH with

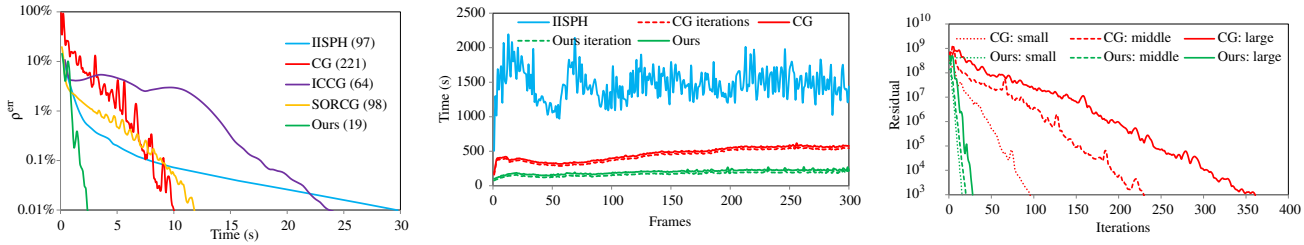

Figure 9: *(Left) Density error (ρ^{err}) profile for our MGCG method, CG, ICCG, SORCG, and IISPH with respect to time. The number in the parentheses represents the number of iterations required to converge to a density error of less than 0.01%. (Middle) Performance profile for IISPH, ISPH with CG, and our method for the middle image of Figure 1. "CG iteration" and "Ours iteration" represent the computation time for the CG iterations only while "CG" and "Ours" includes the computation time for the system construction. Our method outperforms IISPH and CG by a factor of 7.5 and 2.3, respectively. (Right) Residual profile of CG and ours for small, middle, and large scale scenarios with respect to iterations. The number of iterations for CG increases depending on the simulation scale, whereas our method requires almost the same number of iterations regardless of the simulation scale.*

Table 1: *Performance comparisons with different time steps for IISPH and our method for 434.7k particles on a grid of resolution 72x48x48. l denotes Jacobi iteration for IISPH and CG iteration for our method. t^p denotes the average pressure solve time for one frame (including the system construction for our method), and t^t denotes the average total time for one frame. The best t^p and t^t are highlighted in red. Our method outperforms IISPH by a factor of 6.3 in the pressure solve and 5.2 in the total time for their best t^p and t^t, respectively, even in a relatively small scenario.*

	IISPH			Our method		
$\Delta t(ms)$	l	$t^p(s)$	$t^t(s)$	l	$t^p(s)$	$t^t(s)$
2.08	13.15	45.97	62.56	15.71	34.97	51.33
4.16	30.66	53.80	62.23	12.63	15.33	23.70
8.32	80.46	73.43	78.01	12.22	7.31	11.88

MGCG) with previous methods (IISPH [ICS*14] and ISPH with CG) on the dam break scenario, where we used 3.4M fluid particles on a grid of resolution 144x96x96, as shown in Figure 10 (since ISPH with CG and MGCG generates essentially the same visual result, the result for CG is omitted). Our method and IISPH generate comparable visual results. Figure 9 (middle) shows the performance on the pressure computation, where profiles for CG and MGCG iterations only (excluding the system construction) and the pressure solve with CG, MGCG, and IISPH are given (IISPH does not construct the system, and thus there is no system construction cost). Our method outperforms IISPH and CG by a factor of 7.5 and 2.3 in the pressure solve, respectively, and achieves 2.6x better performance than CG in the iteration part.

Time step effect. We compared our method with IISPH using different time steps on the dam break scenario, where we used 434.7k particles on a grid of resolution 72x48x48, and summarized their performance in Table 1. With IISPH, using larger time steps can significantly increase the Jacobi iterations for convergence leading to more computational cost. On the other hand, with our method, the number of CG iterations does not increase even though larger time steps are used because of the fact that our MGCG solver can effectively handle ill-conditioned systems. This feature of our solver makes it easier to choose appropriate time steps since we can optimize the performance by just choosing the largest time step possible. In general, it is preferable to use larger time steps to accelerate the entire simulation as long as the simulation is stable. With this general rule (i.e., with $\Delta t = 8.32$ (ms)), the gain of our method over IISPH is 10.0x in the pressure solve and 6.6x in the total time.

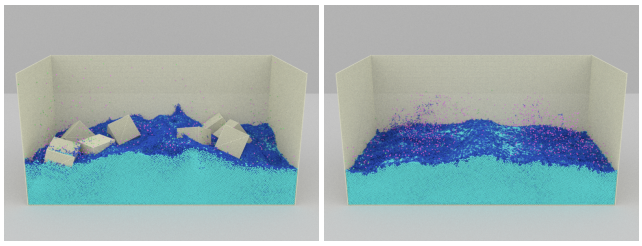

Figure 11: *Double dam break with (left) and without (right) two-way coupled solid objects. Both scenarios require similar iteration counts for convergence (see Figure 12).*

Scalability. To demonstrate the effectiveness of our method on the scalability, we performed the dam break scenario with three different scales: 123.5k particles with the grid resolution 48x32x32, 1.0M particles with the grid resolution 96x64x64, and 3.4M particles with the grid resolution 144x96x96. For this comparison, we used the residual of the PPE since the positive density error ρ^{err} (commonly used in the particle-based methods) depends on time steps [TDNL16] and changes according to the simulation scales, and thus we cannot compare different scale scenarios under a fair condition. Figure 9 (right) demonstrates the profile of residual over iterations to show the scalability of the solvers. With CG, the number of iterations increases significantly as the simulation scale becomes larger. On the other hand, our method requires almost the same number of iterations regardless of the simulation scale, as expected from the theory of the MG solvers. Since our method scales well with respect to the number of particles unlike CG, we believe that our method will be more advantageous with larger-scale scenarios.

Solid interaction. Figure 11 (left) demonstrates that our method is capable of handling two-way coupled solid objects. Additionally, we compare the number of required iterations in Figure 12 for the double dam break scenarios with and without solid cubes (see Figure 11), and show that the number of iterations is comparable regardless of the additional complexity introduced by the two-way coupled solid cubes. Our method is general and can simulate fur-

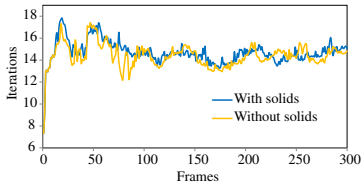

Figure 12: *Iteration profile for Figure 11. Regardless of the two-way coupled solids, the number of iterations is comparable.*

ther complex scenarios, such as one-way coupling of fluids with fast moving solid bunnies (Figure 1 (left)) and two-way coupling of solids in multiphase flows (Figure 1 (right)).

7. Discussions and Limitations

Our particle-based method uses auxiliary grid structures, and thus our method can be considered as a hybrid approach, such as FLIP. However, since the grid structures are used to merely accelerate the pressure solve on particles, our method essentially differs from FLIP. It is worth noting that our multilevel solver is an error correction approach same as other MG, and the communication between particles and grids does not introduce any errors to the final converged solution at the particle level. Because of the auxiliary grid structures, it may seem that our method weakens advantages of particle-based methods. However, we can handle collision detection and resolution at the particle level allowing for the natural coupling of fluids with solid objects (which are commonly described in the Lagrangian manner), as demonstrated in Figures 1 (left and right) and 11. In addition, the grid structures are merely auxiliary structures. Thus, we do not need to adapt the grid structures, e.g., to free surfaces and solid boundaries, unlike the grid-based approaches, allowing for quite simple and fast hierarchy construction.

Although the optimal complexity of the MG solvers is $O(N)$ with the number of unknowns N, the number of iterations slightly increases with our method as N increases. One factor for this non-optimal complexity is due to the solution inconsistency between the particle and grid levels caused by the essentially different discretization methods and heuristically determined scaling factor λ. These would be addressed by using consistent discretization methods and more accurately estimating the optimal λ.

Our MGCG method can outperform IISPH in certain scenarios. However, IISPH can be advantageous when simulations are performed under low-resolution with a soft density constraint since IISPH with Jacobi method converges faster at the early stage, and we may not be able to benefit from the fast convergence of CG at the latter stage.

To derive our solid boundary handling formulation, we assume that pressure changes over time and space are negligible between consecutive simulation steps, and these assumptions are used in the source term computation only. Since pressures are globally computed by solving the PPE, slight value changes in the source term do not significantly affect the resulting pressures. The resulting pressures are still smooth and thus do not introduce stability issues into the simulation. This fact can also be applied to the clamping of the source term. In the grid-based simulation literature, the source term modification is effectively used to compensate fluid volumes [KLL*07] and to simulate compressible fluids [FOA03] without stability issues.

8. Conclusion and Future Work

We proposed a new multilevel solver for particle-based fluids. Our method constructs the hierarchy based on the Cartesian grid, establishes the correspondence between solutions at particle and grid levels, and coarsens simulation elements taking boundary conditions into account. In addition, we proposed a solid boundary handling method that ensures the solvability of the PPE without increasing the size of the system and computational cost. We demonstrated that our method can be significantly faster than IISPH, and its cost scales nearly linearly unlike previous particle-based solvers.

There are several promising future research directions. Since our method is massively parallelizable, implementing the algorithm on a GPU [CM11, CM12] is a natural extension of our method. In the same way as [FWD14, DRW16], introducing the cell duplication technique would be effective to improve the convergence rate. Since our multilevel solver can better handle ill-conditioned systems, aggressively using larger time steps would be beneficial. Considering that MG solvers use coarser levels, applying coarse grid approaches [LZF10, EB14] to particle-based methods would be interesting. Although it is known that geometric MG is generally faster than algebraic MG [BHM00], it would be worth comparing their performance. Particularly, the smoothed aggregation technique [TJM15] would be a promising choice.

Acknowledgements

This work is supported in part by JASSO for Study Abroad, U.S. National Science Foundation, and UNC Arts and Sciences Foundation. We thank anonymous reviewers for their valuable suggestions and comments which help improving the exposition of the paper.

References

[AIA*12] AKINCI N., IHMSEN M., AKINCI G., SOLENTHALER B., TESCHNER M.: Versatile rigid-fluid coupling for incompressible SPH. *ACM Trans. Graph. 31*, 4 (2012), 62:1–62:8. 3, 4, 6

[APKG07] ADAMS B., PAULY M., KEISER R., GUIBAS L. J.: Adaptively sampled particle fluids. *ACM Trans. Graph. 26*, 3 (2007). 2

[BHM00] BRIGGS W., HENSON V., MCCORMICK S.: *A Multigrid Tutorial, Second Edition*. Society for Industrial and Applied Mathematics, 2000. 5, 7, 9

[BK15] BENDER J., KOSCHIER D.: Divergence-free smoothed particle hydrodynamics. In *Proceedings of the 2015 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2015). 1, 2, 3

[BLS12] BODIN K., LACOURSIERE C., SERVIN M.: Constraint fluids. *IEEE Trans. Vis. Comput. Graph. 18*, 3 (2012), 516–526. 1, 2

[Bri15] BRIDSON R.: *Fluid Simulation for Computer Graphics*. A K Peters/CRC Press, 2015. 2, 3, 6, 7

[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), pp. 209–217. 1, 2

[CIPT14] CORNELIS J., IHMSEN M., PEER A., TESCHNER M.: Iisph-flip for incompressible fluids. *Comput. Graph. Forum 33*, 2 (2014), 255–262. 2

[CM11] CHENTANEZ N., MÜLLER M.: Real-time eulerian water simulation using a restricted tall cell grid. *ACM Trans. Graph. 30*, 4 (2011), 82:1–82:10. 1, 9

[CM12] CHENTANEZ N., MÜLLER M.: A multigrid fluid pressure solver handling separating solid boundary conditions. *IEEE Trans. Vis. Comput. Graph. 18*, 8 (aug. 2012), 1191 –1201. 1, 3, 9

[CR99] CUMMINS S. J., RUDMAN M.: An SPH projection method. *Journal of Computational Physics 152*, 2 (1999), 584–607. 1, 2, 3

[DG96] DESBRUN M., GASCUEL M.-P.: Smoothed particles: A new paradigm for animating highly deformable bodies. In *Proceedings of the Eurographics Workshop on Computer Animation and Simulation* (1996), pp. 61–76. 2

[dGWH*15] DE GOES F., WALLEZ C., HUANG J., PAVLOV D., DESBRUN M.: Power particles: An incompressible fluid solver based on power diagrams. *ACM Trans. Graph. 34*, 4 (2015), 50:1–50:11. 2

[DRW16] DICK C., ROGOWSKY M., WESTERMANN R.: Solving the fluid pressure poisson equation using multigrid–evaluation and improvements. *IEEE Trans. Vis. Comput. Graph.* (2016). 7, 9

[DS05] DOSTAL Z., SCHOBERL J.: Minimizing quadratic functions subject to bound constraints with the rate of convergence and finite termination. *Computational Optimization and Applications 30*, 1 (2005), 23–43. 3

[EB14] EDWARDS E., BRIDSON R.: Detailed water with coarse grids: Combining surface meshes and adaptive discontinuous galerkin. *ACM Trans. Graph. 33*, 4 (2014), 136:1–136:9. 9

[FOA03] FELDMAN B. E., O'BRIEN J. F., ARIKAN O.: Animating suspended particle explosions. *ACM Trans. Graph. 22*, 3 (2003), 708–715. 9

[FWD14] FERSTL F., WESTERMANN R., DICK C.: Large-scale liquid simulation on adaptive hexahedral grids. *IEEE Trans. Vis. Comput. Graph. 20*, 10 (2014), 1405–1417. 1, 7, 9

[HLL*12] HE X., LIU N., LI S., WANG H., WANG G.: Local poisson SPH for viscous incompressible fluids. *Comput. Graph. Forum 31*, 6 (2012), 1948–1958. 1, 2

[HWZ*14] HE X., WANG H., ZHANG F., WANG H., WANG G., ZHOU K.: Robust simulation of sparsely sampled thin features in SPH-based free surface flows. *ACM Trans. Graph. 34*, 1 (2014), 7:1–7:9. 2, 3

[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. 7

[ICS*14] IHMSEN M., CORNELIS J., SOLENTHALER B., HORVATH C., TESCHNER M.: Implicit incompressible SPH. *IEEE Trans. Vis. Comput. Graph. 20*, 3 (2014), 426–435. 1, 2, 3, 4, 7, 8

[IOS*14] IHMSEN M., ORTHMANN J., SOLENTHALER B., KOLB A., TESCHNER M.: SPH fluids in computer graphics. In *EUROGRAPHICS 2014 State of the Art Reports* (2014), pp. 21–42. 1, 2, 3

[KLL*07] KIM B., LIU Y., LLAMAS I., JIAO X., ROSSIGNAC J.: Simulation of bubbles in foam with the volume control method. *ACM Trans. Graph. 26*, 3 (2007). 9

[KS14] KANG N., SAGONG D.: Incompressible sph using the divergence-free condition. *Comput. Graph. Forum 33*, 7 (2014), 219–228. 2

[KTO96] KOSHIZUKA S., TAMAKO H., OKA Y.: A particle method for incompressible viscous flow with fluid fragmentations. *Computational Fluid Dynamics Journal 4*, 1 (1996), 29–46. 1, 2, 3, 5

[LZF10] LENTINE M., ZHENG W., FEDKIW R.: A novel algorithm for incompressible flow using only a coarse grid projection. *ACM Trans. Graph. 29*, 4 (2010), 114:1–114:9. 9

[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), pp. 154–159. 2, 3

[MHHR07] MÜLLER M., HEIDELBERGER B., HENNIX M., RATCLIFF J.: Position based dynamics. *Journal of Visual Communication Image Representation 18*, 2 (2007), 109–118. 2

[MM13] MACKLIN M., MÜLLER M.: Position based fluids. *ACM Trans. Graph. 32*, 4 (2013), 104:1–104:5. 1, 2

[Mon94] MONAGHAN J.: Simulating free surface flows with SPH. *J. Comput. Phys. 110*, 2 (1994), 399–406. 2

[Mon00] MONAGHAN J.: SPH without a tensile instability. *Journal of Computational Physics 159*, 2 (2000), 290 – 311. 3

[MST10] MCADAMS A., SIFAKIS E., TERAN J.: A parallel multigrid poisson solver for fluids simulation on large grids. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2010), pp. 65–74. 1, 6, 7

[Mül08] MÜLLER M.: Hierarchical position based dynamics. In *VRIPHYS* (2008), pp. 1–10. 1, 5

[MZS*11] MCADAMS A., ZHU Y., SELLE A., EMPEY M., TAMSTORF R., TERAN J., SIFAKIS E.: Efficient elasticity for character skinning with contact and collisions. *ACM Trans. Graph. 30*, 4 (2011), 37:1–37:12. 1

[OK12] ORTHMANN J., KOLB A.: Temporal blending for adaptive SPH. *Comput. Graph. Forum 31*, 8 (2012), 2436–2449. 2

[PTB*03] PREMOZE S., TASDIZEN T., BIGLER J., LEFOHN A., WHITAKER R. T.: Particle-based simulation of fluids. 401–410. 2, 3, 5

[RLY*14] REN B., LI C., YAN X., LIN M. C., BONET J., HU S.-M.: Multiple-fluid SPH simulation using a mixture model. *ACM Trans. Graph. 33*, 5 (2014), 171:1–171:11. 2

[RWT11] RAVEENDRAN K., WOJTAN C., TURK G.: Hybrid smoothed particle hydrodynamics. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2011), pp. 33–42. 1, 2, 7

[SB12] SCHECHTER H., BRIDSON R.: Ghost SPH for animating water. *ACM Trans. Graph. 31*, 4 (2012), 61:1–61:8. 3

[SBH09] SIN F., BARGTEIL A. W., HODGINS J. K.: A point-based method for animating incompressible flow. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2009), pp. 247–255. 1, 2

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

[SL03] SHAO S., LO E. Y.: Incompressible SPH method for simulating newtonian and non-newtonian flows with a free surface. *Advances in Water Resources 26*, 7 (2003), 787 – 800. 2, 3, 5

[SP09] SOLENTHALER B., PAJAROLA R.: Predictive-corrective incompressible SPH. *ACM Trans. Graph. 28*, 3 (2009), 40:1–40:6. 1, 2, 7

[SVJ15] SACHT L., VOUGA E., JACOBSON A.: Nested cages. *ACM Trans. Graph. 34*, 6 (2015), 170:1–170:14. 5

[TDNL16] TAKAHASHI T., DOBASHI Y., NISHITA T., LIN M. C.: An efficient hybrid incompressible SPH solver with interface handling for boundary conditions. *Comput. Graph. Forum* (2016). 2, 3, 4, 8

[TJM15] TAMSTORF R., JONES T., MCCORMICK S. F.: Smoothed aggregation multigrid for cloth simulation. *ACM Trans. Graph. 34*, 6 (2015), 245:1–245:13. 9

[TOS00] TROTTENBERG U., OOSTERLEE C. W., SCHULLER A.: *Multigrid*. Academic press, 2000. 5, 7

[WMRSF15] WEBER D., MUELLER-ROEMER J., STORK A., FELLNER D.: A cut-cell geometric multigrid poisson solver for fluid simulation. *Comput. Graph. Forum 34*, 2 (2015), 481–491. 1, 6, 7

[WOR10] WANG H., O'BRIEN J., RAMAMOORTHI R.: Multi-resolution isotropic strain limiting. *ACM Trans. Graph. 29*, 6 (2010), 156:1–156:10. 1

[ZB05] ZHU Y., BRIDSON R.: Animating sand as a fluid. *ACM Trans. Graph. 24*, 3 (2005), 965–972. 2

[ZSTB10] ZHU Y., SIFAKIS E., TERAN J., BRANDT A.: An efficient multigrid method for the simulation of high-resolution elastic solids. *ACM Trans. Graph. 29*, 2 (2010), 16:1–16:18. 1