# Particle Merging-and-Splitting

Nghia Truong, Cem Yuksel, Chakrit Watcharopas, Joshua A. Levine, Robert M. Kirby

**Abstract**—Robustly handling collisions between individual particles in a large particle-based simulation has been a challenging problem. We introduce particle *merging-and-splitting*, a simple scheme for robustly handling collisions between particles that prevents inter-penetrations of separate objects without introducing numerical instabilities. This scheme merges colliding particles at the beginning of the time-step and then splits them at the end of the time-step. Thus, collisions last for the duration of a time-step, allowing neighboring particles of the colliding particles to influence each other. We show that our merging-and-splitting method is effective in robustly handling collisions and avoiding penetrations in particle-based simulations. We also show how our merging-and-splitting approach can be used for coupling different simulation systems using different and otherwise incompatible integrators. We present simulation tests involving complex solid-fluid interactions, including solid fractures generated by fluid interactions.

**Index Terms**—Particle-based Simulation, Collision Handling, Solid-Fluid Coupling

## 1 Introduction

Particle-based simulations are commonplace in computer graphics, used for simulating a wide variety of physical phenomena for different material types and phases. In particular, high-resolution simulations involving a large number of particles can deliver complex animations with rich visual detail. Yet, handling interactions between individual particles, especially collisions, has been a challenging problem. This is not only because of the sheer number of interactions that can occur but also due to the difficulty of robustly enforcing collision constraints in the presence of other interactions between particles.

Not all particle-based simulations must explicitly consider pair-wise particle collisions. For example, Eulerian fluid simulation handle interactions among particles differently, without directly considering collisions. In such cases, explicit collisions are needed for handling boundaries and coupling with other simulation systems.

However, particle-level collisions cannot always be safely ignored. For example, particle-based simulations involving (non-granular) solids must properly incorporate particle collisions. Unfortunately, common collision handling techniques based on force or impulse formulations can be unstable or fail to resolve collisions, as we show in this paper. Poorly handled particle collisions can lead to catastrophic problems, such as inter-penetrations of separate objects and instabilities that cause numerical failure or unnatural material behavior. In particular, fracture simulations are highly sensitive to such instabilities, as the velocity spike of a single particle can cause an entire object to instantly crumble.

In this paper, we introduce *merging-and-splitting*, a simple method for robustly handling collisions between individual particles in particle-based simulations. When two (or more) particles come into contact, we first *merge* them into a larger *meta-particle* based on an inelastic collision formulation. A meta-particle behaves as a single particle during numerical integration. This treatment allows us to compute the collective momentum of the colliding particles, considering the influence of the surrounding particles. After numerical integration, we *split* the colliding particles, following the principles of elastic collision with momentum conservation and energy preservation. This approach facilitates information exchange between the colliding particles for the duration of the time-step. Our tests show that our merging-and-splitting scheme completely prevents inter-penetrations without introducing instabilities.

Merging-and-splitting also allows coupling different particle-based simulations using different and otherwise incompatible numerical systems. Our tests with coupling different particle-based simulations include mass-spring systems for deformable objects, peridynamics [1] for brittle solids, and Smoothed Particle Hydrodynamics (SPH) [2], [3] and Fluid-Implicit Particle (FLIP) [4] for fluids. In our tests, we handle solids using implicit integration, SPH using explicit integration, and FLIP using a semi-implicit scheme (explicit advection and implicit pressure projection). We use our merging-and-splitting technique for handling collisions between particles of the same simulation system for solids and particles simulated using different integration schemes. In fact, the coupling between different simulation systems in our tests is handled entirely via merging-and-splitting. Our results clearly show that our merging-and-splitting scheme is effective in coupling different particle-based formulations with different types of numerical integrators. Using our scheme, we also demonstrate unprecedented simulation scenarios, like solid fracture due to fluid interaction.

## 2 Related Work

There is a large body of work on collision detection and handling in computer graphics. In this section we briefly review the most related methods on collision handling.

**Force-based Collision Response.** Arguably the most popular technique for handling collisions is by applying


- *N. Truong, C. Yuksel, and R. M. Kirby is with University of Utah.*
- *C. Watcharopas is with Kasetsart University.*
- *J. A. Levine is with University of Arizona.*


*Manuscript received November 13, 2020; revised November 13, 2020.*

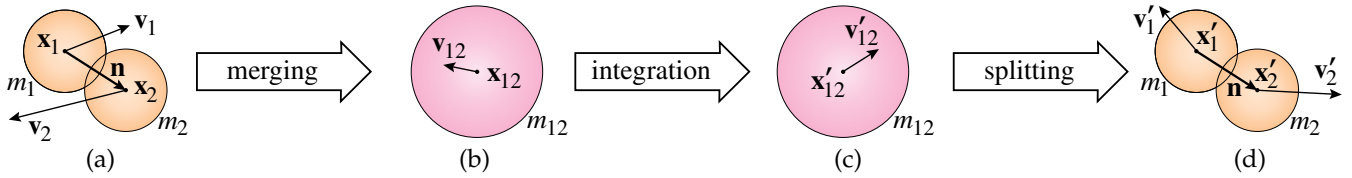



**Fig. 1: The overview of merging-and-splitting approach:** (a) colliding particles are merged into (b) a meta-particle using the principles of inelastic collision. This meta-particle is used during time integration to compute (c) its position and velocity at the end of the time-step. Finally, the meta-particle is split into (d) the two original particles with updated velocities, using the principles of elastic collision with momentum and energy conservation.

penalty forces between colliding objects [5]–[7]. The magnitude of a penalty force is determined based on the amount of intersection between the two objects, so typical penalty force formulations do not completely eliminate penetration. More importantly, the results are sensitive to the stiffness of the penalty force, which is often controlled by a user-defined parameter. Penalty forces can be distance-based [6], [8] or volume-based [9], [10]. Their stability can be improved using multiple contact points [11] or a global contact treatment [12], and a friction term can be included [13]. Smooth collision responses can be achieved using continuous penalty forces [14]. Harmon et al. [15], [16] present a method using asynchronous variational integrators and a contact barrier potential to avoid penetrations and conserve momentum and energy, but simulations involving a large number of collisions require numerous iterations, making this approach inefficient for particle-based simulations. Penalty forces are also commonplace in cloth simulations [17]–[20]. Notably, *rigid impact zones* were introduced [18], [19] to handle collisions by rigidifying the entire collision region, which have similarities with our approach.

Similar force-based formulations are also used in solid-fluid coupling [21] by directly applying forces between nearby particles [22], [23] or introducing additional pressure force [24], [25]. The simplicity of the force formulation allows coupling Eulerian fluid simulations with particle-based solid simulations [26] and particle-based fluid simulations with deformable solids [27]–[29] and, using continuous collision detection, sheet-based cloth simulations [30], [31]. Recent work on handling collisions between fluid particles and rigid boundaries include density maps [32], volume maps [33], liquid boundaries [34] and pressure boundaries [35], [36]. Density maps can also be augmented to handle frictional contact [37], and strong fluid-rigid body coupling simulation is achieved by interlinked pressure solvers [38].

**Impulse-based Collisions.** Another popular alternative for handling collisions is impulse-based formulations, which are common in rigid-body dynamics [39]–[43]. The momentum and energy transfer between two colliding objects are handled instantly by directly modifying the velocities of colliding objects. Rest-in-contact situations can be handled by careful treatments [40], [44]. Notably, iteratively solving for the impulses needed has been widely used for achieving complex scenarios, such as stacking and static friction [42], [44], [45].

**Constrained Dynamics.** Collision response can be formulated as constraints that prevent penetration [46]. It is often implemented using Linear Complementary Programming (LCP) for searching the reaction space for the feasible contact behavior [47] and it can be used for modeling interactions of rigid bodies and deformable bodies [48]–[52], granular materials [53], fluids [54], [55], and quasi-rigid bodies [56]. Formulated as an optimization problem, LCP is suitable for solving contact problems for which the optimal solutions are rest positions with minimum energy exchange [49], [57]–[59]. Robust contact handling which facilitates smooth rolling and sliding, stacking and impact handling was achieved by a formulation based on implicit complimentary constraints and Lagrange multipliers [52].

**Grid-based Solutions.** Hybrid Eulerian-Lagrangian simulation methods can avoid explicitly handling collisions by solving the aggregate collision behavior on a grid. For example, fluid simulations using FLIP have been coupled with hair [60] and cloth [61] simulations. The Material Point Method [62] is another good example, providing simulations of various material types, such as snow [62], multi-species [63], [64] with phase transition [63], sand [65], [66], elastoplastic solids, viscoelastic fluids, foams and sponges [67]–[69], anisotropic elastoplastic materials [70], [71], fluid-sediment mixture [72]. MPM can also achieve solid-fluid coupling simulation [64], [72], [73], dynamic fracture [71], ductile fracture [74] and frictional contact [75].

## 3 Merging-and-Splitting

We assume that each particle has a non-zero mass and a spherical shape with non-zero radius. This is not necessarily the case for all particle-based simulations. Some of them use massless particles and some treat them as point samples with no size. Therefore, we begin with assigning a mass and a radius to each particle for collision purposes.

Our approach for robustly handling particle-level interactions is a scheme that we call *merging-and-splitting*. This scheme first *merges* colliding particles (Fig. 1a) into a larger *meta-particle* (Fig. 1b). We treat this meta-particle as a regular particle, rather than a compound object made up of multiple particles. This crucial simplification allows seamlessly including the meta-particle in arbitrary particle-based simulation systems without modification. The meta-particle is integrated along with the other particles in the system for computing its updated position and velocity at the end of the time-step (Fig. 1c). Finally, we *split* the meta-particle into the original colliding particles (Fig. 1d).

Consider two particles with masses $m_1$ and $m_2$, positions $\mathbf{x}_1$ and $\mathbf{x}_2$, and velocities $\mathbf{v}_1$ and $\mathbf{v}_2$ colliding with each other. We merge the two particles based on an inelastic

collision formulation, such that the total mass, position, and velocity of the meta-particle become

$$m_{12} = m_1 + m_2 \ , \tag{1}$$
$$\mathbf{x}_{12} = \left(m_1\mathbf{x}_1 + m_2\mathbf{x}_2\right)/m_{12} \ , \tag{2}$$
$$\mathbf{v}_{12} = \left(m_1\mathbf{v}_1 + m_2\mathbf{v}_2\right)/m_{12} \ . \tag{3}$$

While this merging operation conserves momentum, it does not conserve kinetic energy, because it is based on inelastic collision principles. In our interaction scheme we compute the change in kinetic energy, $\Delta E$, and store it as a potential energy in a *virtual bond* between the two colliding particles, where

$$\Delta E = \frac{m_1 m_2}{2m_{12}} \left(\mathbf{v}_1 - \mathbf{v}_2\right)^2 \ . \tag{4}$$

Note that $\Delta E$ cannot be negative, so we always store a non-negative potential energy in the virtual bond between the two particles.

The meta-particles are used during the time-step integration instead of the original particles. However, while computing the forces between the meta-particles and the surrounding particles, we use the relative positions of the merged particles that make up the meta-particles. Let $\mathbf{n}$ be the vector connecting the centers of the two colliding particles in the beginning of the time-step, such that

$$\mathbf{n} = \mathbf{x}_2 - \mathbf{x}_1 \ . \tag{5}$$

We assume that the meta-particle preserves the relative orientation of the merged particles throughout the time-step integration, so $\mathbf{n}$ remains constant. Therefore, we can find the positions of the merged particles using the updated position of the meta-particle $\mathbf{x}'_{12}$, such that

$$\mathbf{x}'_1 = \mathbf{x}'_{12} - \left(m_2/m_{12}\right)\mathbf{n} \ , \text{ and} \tag{6}$$
$$\mathbf{x}'_2 = \mathbf{x}'_{12} + \left(m_1/m_{12}\right)\mathbf{n} \ . \tag{7}$$

Thus, $\mathbf{x}'_1 = \mathbf{x}_1 + \Delta\mathbf{x}$ and $\mathbf{x}'_2 = \mathbf{x}_2 + \Delta\mathbf{x}$ for $\Delta\mathbf{x} = \mathbf{x}'_{12} - \mathbf{x}_{12}$. Note that these simple equations can be used to calculate the intermediate positions of the merged particles at any time within the time-step for computing forces between the meta-particles and the rest of the simulation system. Since meta-particles are treated as regular particles, forces acting on the meta-particles are applied at the center of mass of the meta-particles.

During splitting, a portion of the potential energy $\Delta E$ stored in the virtual bond converts back into kinetic energy, while the rest dissipates. The amount of energy restoration is controlled by a user-defined coefficient $\alpha \in [0, 1]$ that serves as a restitution parameter (we use $\alpha = 1$ in our tests, unless otherwise specified, so we fully conserve energy for collision handling without any dissipation). Let $\mathbf{v}'_{12}$ be the velocity of the meta-particle at the end of the time-step integration. The final velocities of the particles $\mathbf{v}'_1$ and $\mathbf{v}'_2$ satisfy momentum and energy conservation equations

$$m_{12}\mathbf{v}'_{12} = m_1\mathbf{v}'_1 + m_2\mathbf{v}'_2 \tag{8}$$
$$\alpha\Delta E + \frac{1}{2}m_{12}{\mathbf{v}'_{12}}^2 = \frac{1}{2}m_1{\mathbf{v}'_1}^2 + \frac{1}{2}m_2{\mathbf{v}'_2}^2 \ . \tag{9}$$

Using these two equations we can write

$$\left(\mathbf{v}'_{12} - \mathbf{v}'_1\right)^2 = s^2 \ , \tag{10}$$

where

$$s^2 = \frac{2\alpha\Delta E}{m_{12}\left(m_1/m_2\right)} \ . \tag{11}$$

Here the only unknown is $\mathbf{v}'_1$, and once it is solved, $\mathbf{v}'_2$ can be calculated using momentum conservation (Equation 8).

Equation 10 describes the general energy and momentum conservation laws for splitting a meta-particle into any two particles (with masses $m_1$ and $m_2$) without considering the initial configuration prior to merging. Therefore, Equation 10 has infinitely many solutions. We consider the initial conditions of the collision event to narrow down the solution space.

We can split the change in velocity $\mathbf{v}'_1 - \mathbf{v}_1$ into two components: one along the collision direction $\hat{\mathbf{n}} = \mathbf{n}/|\mathbf{n}|$; and one along an orthogonal direction, using

$$\mathbf{v}'_1 = \mathbf{v}_1 + \mu\hat{\mathbf{n}} + \boldsymbol{\epsilon} \ , \tag{12}$$

where $\mu$ is some scalar value and $\boldsymbol{\epsilon}$ the orthogonal velocity change, such that $\boldsymbol{\epsilon}\cdot\hat{\mathbf{n}} = 0$. Substituting this into Equation 10 yields the quadratic equation

$$\mu^2 - \left(2\hat{\mathbf{n}}\cdot\left(\mathbf{v}'_{12} - \mathbf{v}_1 - \boldsymbol{\epsilon}\right)\right)\mu + \left(\mathbf{v}'_{12} - \mathbf{v}_1\right)^2 - s^2 = 0 \ , \tag{13}$$

which has a closed form solution to $\mu$ for a given $\boldsymbol{\epsilon}$.

For simplicity, we favor solutions with minimal orthogonal momentum exchanges between colliding particles and take $\boldsymbol{\epsilon} = \mathbf{0}$, resulting in momentum exchange along $\hat{\mathbf{n}}$ only. If the quadratic equation has two real roots for $\mu$, we take the smaller one, since it ensures that the velocities $\mathbf{v}'_1$ and $\mathbf{v}'_2$ separate the particles, such that

$$\left(\mathbf{v}'_2 - \mathbf{v}'_1\right)\cdot\hat{\mathbf{n}} \geq 0 \ . \tag{14}$$

The other (larger) root for $\mu$ would lead to final velocities pointing towards each other, which would be an unacceptable solution for two colliding particles.

Yet, since we impose no restrictions on the time-step integration, $\mathbf{v}'_{12}$ can theoretically be any arbitrary value and the resulting Equation 13 may have no real solution for $\mu$ with $\boldsymbol{\epsilon} = \mathbf{0}$. In this case, we consider the solution that minimizes the magnitude of the orthogonal term $|\boldsymbol{\epsilon}|$.

We derive a closed form solution to this minimization geometrically, as shown in Fig. 2. Notice that Equation 10 defines a sphere of valid solutions for $\mathbf{v}'_1$ that conserve energy centered at $\mathbf{v}'_{12}$ with radius $s$. If the ray $\mathbf{v}_1 + \mu\hat{\mathbf{n}}$ intersects with this sphere, we get up to two solutions for $\mu$, corresponding to the two intersection points of the ray with the sphere of valid solutions (Fig. 2a). If there is no real root for Equation 13, the ray does not intersect with this sphere (Fig. 2b). In that case, the point along the ray that minimizes $|\boldsymbol{\epsilon}|$ is the closest point between the line $\mathbf{v}_1 + \mu\hat{\mathbf{n}}$ and the sphere in Equation 10. Using this property and solving for $\mu$ we can write

$$\mu = \hat{\mathbf{n}}\cdot\left(\mathbf{v}'_{12} - \mathbf{v}_1\right) \tag{15}$$
$$\boldsymbol{\epsilon} = \left(|\mathbf{w}| - s\right)\frac{\mathbf{w}}{|\mathbf{w}|} \tag{16}$$

where $\mathbf{w} = \mathbf{v}'_{12} - \mathbf{v}_1 - \mu\hat{\mathbf{n}}$. Notice that in this solution $\boldsymbol{\epsilon}$ is perpendicular to $\hat{\mathbf{n}}$.

An important property of our formulation is that, in any case, the resulting velocities are guaranteed to separate

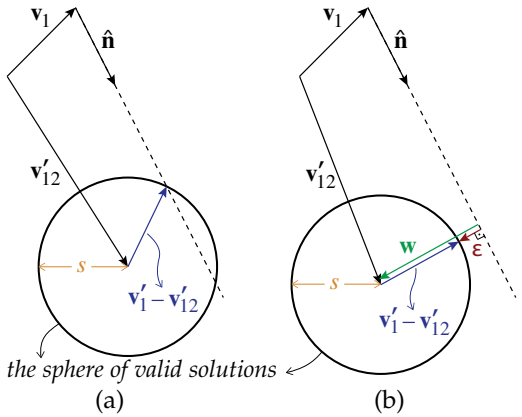



**Fig. 2:** Geometric solutions to the quadratic equation in Equation 13 showing the sphere of valid solutions centered around $\mathbf{v}'_{12}$ that conserve momentum and energy: (a) quadratic equation has roots, and (b) quadratic equation has no real roots.

the particles, as in Equation 14. The solution to $\mathbf{v}'_2$ can be computed from momentum conservation, such that

$$\mathbf{v}'_2 = \frac{m_{12}}{m_2}\mathbf{v}'_{12} - \frac{m_1}{m_2}\mathbf{v}'_1 \ .$$

Therefore, we can write

$$(\mathbf{v}'_2 - \mathbf{v}'_1) \cdot \hat{\mathbf{n}} = -\frac{m_{12}}{m_2}(\mathbf{v}'_1 - \mathbf{v}'_{12}) \cdot \hat{\mathbf{n}} \ .$$

If Equation 13 has no root, $(\mathbf{v}'_1 - \mathbf{v}'_{12}) \cdot \hat{\mathbf{n}} = 0$, since $(\mathbf{v}'_1 - \mathbf{v}'_{12})$ is perpendicular to $\hat{\mathbf{n}}$ (Fig. 2b). Otherwise, we pick the smaller root, which makes $(\mathbf{v}'_1 - \mathbf{v}'_{12}) \cdot \hat{\mathbf{n}} \leq 0$ (Fig. 2a). Therefore, Equation 14 is always satisfied.

Note that any solution that satisfies Equation 10 conserves both momentum and energy. The procedure described above picks a valid solution that conforms to the initial collision configuration of the merged particles. Therefore, whether Equation 13 has a real root and whether some minimal orthogonal momentum exchange between particles must be assumed has no practical consequence, since both momentum and energy are always conserved. Furthermore, we impose no restrictions on the time integration, so we have no control over the resulting $\mathbf{v}'_{12}$ value, which may require a non-zero orthogonal term to preserve the initial particle configuration. Thus, it would be unreasonable to always expect a solution with $|\boldsymbol{\epsilon}| = 0$. Nonetheless, in any case we pick a consistent solution that minimizes the orthogonal term.

In our tests, we observed that when colliding particles are free (i.e. not connected to other particles) or have weak connections to other particles, we often find a solution for $\mu$ with $|\boldsymbol{\epsilon}| = 0$. On the other hand, if the motion of one particle is restricted (such as when the object is pushed against the simulation boundary or particles are attached to immobile points), which in turn restricts the motion of the meta-particle, a high percentage of the solutions contain a non-zero orthogonal term (i.e. $|\boldsymbol{\epsilon}| > 0$).

If more than two particles collide within the same time-step, we recursively merge these particles into larger meta-particles. After computing the updated momentum for the final meta-particle, we recursively split them in the inverse order of merging. We have empirically verified that the order of merging does not affect the outcome of the final particle velocities after splitting. Thus, recursive merging can begin with any pair of particles.

Note that by using merging-and-splitting, we assume that collisions between the particles are not instantaneous, but instead they can take as long as one time-step. This allows the neighboring particles to interact with the colliding particles and influence the outcome of the collision event. Collisions that require even longer interactions than one time-step, such as rest-in-contact situations, are handled in the subsequent time-steps.

## 4 Coupling Different Integrators

Our merging-and-splitting approach can also be used for coupling different particle-based simulation systems using different integrators. We achieve this by introducing collision-based interactions between particles of different simulation systems.

When the particles of the two systems come into contact, we merge them into meta-particles, as explained in the previous section. We include these meta-particles in *both* simulation systems. However, since we impose no restrictions on the integrators used for the two systems, the two integrators are likely to produce two different results for each meta-particle. Since we cannot allow two different solutions for one meta-particle, we must combine the two results into one solution. We do so by producing a *synchronized velocity* for each meta-particle using the two solutions. The meta-particle positions are updated using the synchronized velocities to ensure that the two systems produce consistent results. Combining the solutions of two different integrators is not a new concept [76]–[78]. What is different about our approach is that we formulate our synchronization similar to the merging operations we use for generating meta-particles.

Let $\mathbf{v}^A_{12}$ and $\mathbf{v}^B_{12}$ be the velocities of a meta-particle generated by the two systems at the end of the time-step integration. We calculate the synchronized velocity $\mathbf{v}'_{12}$ using a weighted average of the two solutions. Consider that before merging the particles with masses $m_1$ and $m_2$ originally belong to the first and the second simulation systems, respectively. We compute the synchronized velocity $\mathbf{v}'_{12}$ using

$$\mathbf{v}'_{12} = \left(m_1\mathbf{v}^A_{12} + m_2\mathbf{v}^B_{12}\right) \Big/ m_{12} \ . \tag{17}$$

This provides a weighted average of the meta-particle momenta generated by the two integrators, using the mass percentages of the two particle types in the meta-particle as weights. Yet, similar to merging, this operation leads to energy dissipation. To avoid this, we also consider the weighted average of the kinetic energy

$$\bar{E}_k = \frac{1}{2}m_1\left(\mathbf{v}^A_{12}\right)^2 + \frac{1}{2}m_2\left(\mathbf{v}^B_{12}\right)^2 \ . \tag{18}$$

Since Equation 17 cannot preserve all of this kinetic energy, we add $\Delta E_k$, the energy lost in Equation 17, to the virtual bond of the meta-particle, using

$$\Delta\bar{E}_k = \frac{m_1 m_2}{2m_{12}}\left(\mathbf{v}^A_{12} - \mathbf{v}^B_{12}\right)^2 \ . \tag{19}$$

During splitting, we multiply this energy with another user-defined parameter, $\beta \in [0, 1]$, which determines the percentage of this energy that should be preserved, and add it to the left side of Equation 9. This replaces Equation 11 with

$$s^2 = \frac{2\left(\alpha \Delta E + \beta \Delta \bar{E}_k\right)}{m_{12}\left(m_1/m_2\right)} \,. \tag{20}$$

The synchronized velocities produced by these operations ensure that we have a single solution for each meta-particle. Coupling different simulation systems using merging-and-splitting has the obvious advantage that there is no need to track the interface between the two systems. Furthermore, it allows coupling particle-based simulation systems that are designed for different material types and material behavior for handling interesting simulation scenarios that are beyond the capabilities of current unified simulation systems.

## 5 Implementation Details

Our merging-and-splitting scheme can be used in various ways for handling collisions within a simulation system or coupling different simulation systems. Yet, there are a number of details specific to merging-and-splitting that need to be considered. In this section we provide the details of our implementation and the reasons behind our implementation decisions. While most of these details are specific to our implementation, they represent a list of potential issues that one needs to consider for any implementation of merging-and-splitting. A significant portion of our implementation decisions are related to the collision detection scheme we use.

### 5.1 Collision Detection

Our implementation uses a simple collision detection scheme that merely checks whether particles intersect at the beginning of each time-step. If two particles intersect, they are merged and integrated as a meta-particle. If the meta-particles are split before the positions update, our method guarantees that the split particles will not move towards each other, so they may no longer be in contact in the beginning of the next time-step. While this may appear like a positive outcome at first glance, it can fail to prevent penetrations for contact situations that require multiple time-steps to resolve (such as rest-in-contact). Therefore, in our implementation we perform splitting after position update. This means that the same two particles will remain in contact in the beginning of the next time-step as well. Thus, if we only rely on the particle positions for collision detection, two particles that come into contact would perpetually remain in contact. We avoid this by introducing a secondary rule for collision detection, such that intersecting particles are merged only if their velocities are towards each other. Let $r_1$ and $r_2$ be the collision radii of the colliding particles. We use the following two simple rules for collision detection:

$$\text{Collision Rule 1:} \quad |\mathbf{x}_2 - \mathbf{x}_1| < r_1 + r_2 \tag{21}$$

$$\text{Collision Rule 2:} \quad \mathbf{n} \cdot (\mathbf{v}_2 - \mathbf{v}_1) < 0 \tag{22}$$

For handling the first rule, limiting the time-step size is important to ensure that we do not miss collisions. To compute a safe time-step size, we employ a typical CFL condition that limits the motion of particles within a time-step to no more than particle radius. This ensures that our simple collision detection mechanism does not miss head-on collisions between particles, but collisions at grazing angles can be missed. Note that alternative approaches such as using a continuous collision detection scheme [79] instead would entirely avoid this limit on the time-step size.

The second rule ensures that the colliding particles do not perpetually remain in contact. Indeed, since our merging-and-splitting scheme produces final velocities for the merged particles that point away from each other, it is guaranteed that the two intersecting particles will not be merged in the beginning of the next time-step. However, if the collision event must last longer than one time-step (such as rest-in-contact situations), the intersecting particles must be merged the next time-step as well. To facilitate this, our implementation uses a two-stage integration scheme. In the first stage, we only merge intersecting particles with velocities pointing towards each other (following the second rule). We mark intersecting particles with velocities pointing away from each other, but we do not merge them. If the marked particles at the end of the time-step integration have velocities pointing towards each other, we recompute the time-step integration in a second stage by merging those marked particles in the beginning of the time-step. Obviously, this two-stage integration scheme nearly doubles the computation time.

In theory, at the end of the second stage, we can detect that other marked particles that were not merged prior to the second stage may require merging. Therefore, handling all collisions with this approach may need more than two stages. Yet, in our implementation we only use two stages and we have not observed any practical consequences of not adding extra stages as needed.

An alternative solution would be to modify the intersecting particle positions at the end of each time-step to ensure that the previously merged particles no longer intersect. However, while this is a commonly-used technique in computer graphics [23], [29], position corrections often inject extra energy into the system that impacts the stability of the simulations or introduces artificial vibrations. Moreover, this seemingly minor energy injection of position correction can lead to catastrophic events with fracture simulations, instantly shattering all bonds and causing entire objects to explode into individual particles. Also, position corrections can cause other particles to intersect. That is why we use the more expensive two-stage integration approach in our implementation.

We use our merging-and-splitting scheme for handling collisions between solid particles integrated using the same system. However, we ignore intersections of neighboring particles that are directly connected via springs. Thus, we can have high-resolution solid simulations with neighboring particles almost intersecting with each other, without introducing undesirable internal collisions between neighboring particles.

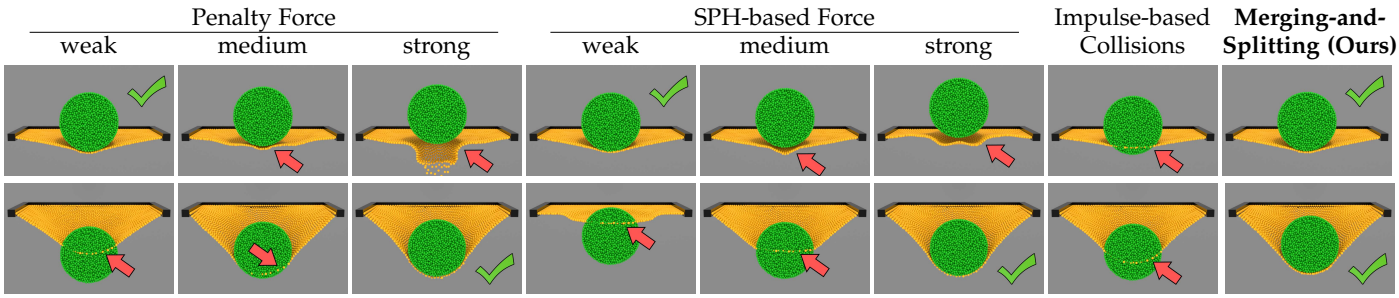



**Fig. 3: Comparison of different methods for handling solid-solid collisions:** cross-section views of simulations including a soft elastic cloth hit by a solid ball in free fall. The bottom row shows the same simulation with the solid ball falling from a higher initial position (i.e. faster impact). All particles have identical mass and weak/medium/strong penalty and SPH-based forces differ by an order of magnitude. Only our merging-and-splitting approach prevents penetrations without introducing instability for both cases.

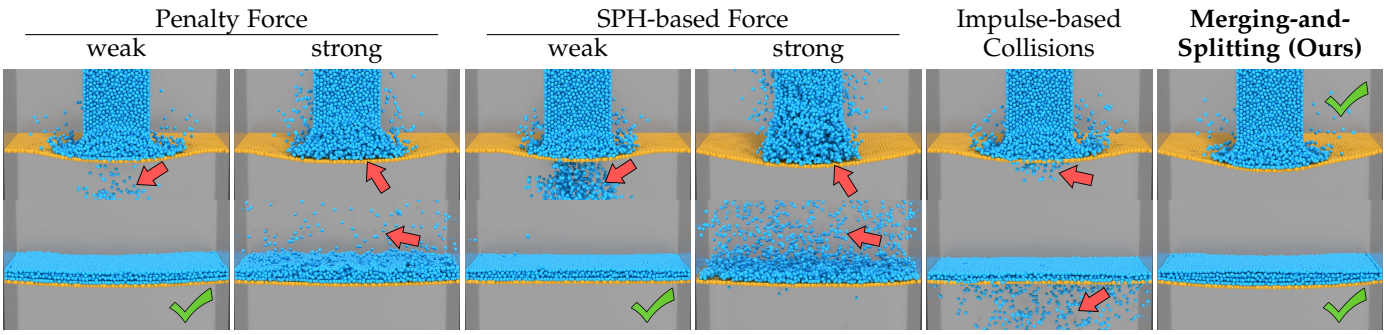



**Fig. 4: Comparison of different methods for handling solid-fluid collisions:** cross-section views of simulations including a soft elastic cloth hit by a fluid beam. Each column shows the same simulation at two different time steps. All particles have identical mass and weak/strong penalty and SPH-based force stiffness parameters differ by an order of magnitude. Notice that only our merging-and-splitting approach prevents penetrations without introducing instability.

### 5.2 Solid-Fluid Coupling

Solid-fluid coupling is an important application in computer graphics [21]. In our system, after collision detection, we merge colliding particle pairs recursively. When coupling integrators for solid and fluid simulations, large chains of particle intersections can occur at the solid-fluid boundary. Merging all intersecting particles of a chain into a single meta-particle can rigidify the entire solid-fluid interface and lead to unnatural results. We use a simple fix that merely limits the number of particles in a meta-particle using a threshold $n$. This breaks large chains of intersecting particles into multiple groups. Our tests show that the simulations are not sensitive to the value of this threshold, unless a very small or a very large number is used. When $n$ is too small, ignored collisions due to this limit can lead to penetrations. On the other hand, when $n$ is too large, the entire solid-fluid interface can get rigidified. We use a number between 8 and 64 in our tests, producing similar results. Our implementation performs merging in the order of particle indices. To introduce randomness in grouping, before starting to merge each group, we can randomly pick a limit between two user-specified limits $n_{\min}$ and $n_{\max}$.

Limiting the number of particles in a meta-particle has the obvious theoretical drawback that some relatively small fraction of collision events would be ignored at each time step. Yet, as our experiments confirm, this is of little practical concern. This is because, with randomized grouping, we do not persistently ignore specific pairs of colliding particles, i.e. a collision event that is ignored in one time step is likely to be considered in the next time step.

## 6 Results

We evaluate our merging-and-splitting method by comparing it to typical alternative techniques for handling collisions in particle-based simulations and presenting large simulations examples, including solid-fluid coupling tests using different simulation systems. We use $\alpha = 1$, $\beta = 0$, and $n_{\min} = 8, n_{\max} = 64$ in all simulations, unless otherwise specified. The rendered surfaces are generated using level-sets [80] for fluid particles and tetrahedralization of the solid mesh in the beginning of the simulation [81] for solid particles.

### 6.1 Comparisons

We compare our merging-and-splitting method with force-based approaches, using penalty [6], [8] or SPH-based [22], [23] force formulations, and impulse-based collision handling methods [39]–[43].

Fig. 3 shows comparisons of different methods for handling collisions within a particle-based solid simulation. The top and bottom rows show a ball falling onto an elastic

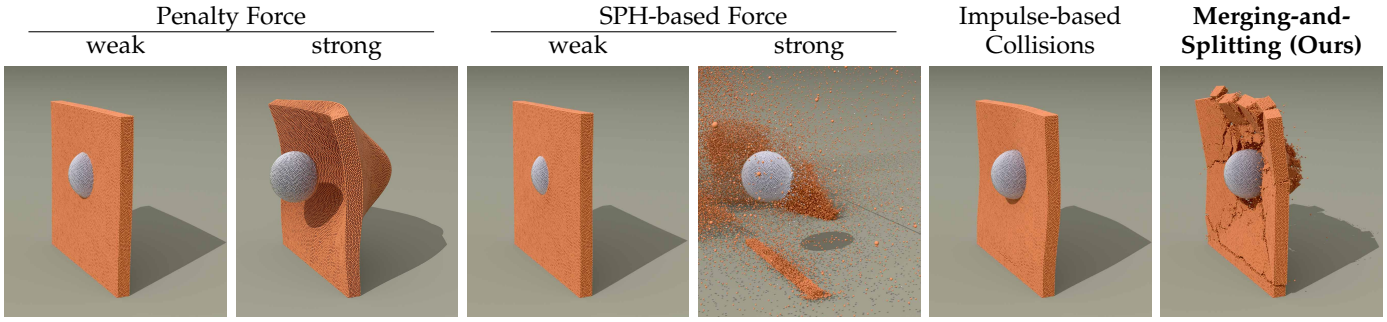



**Fig. 5: Comparison of different methods for collision handling in fracture simulation:** a solid brittle wall is hit by a dense ball with $20\times$ heavier particles. The bottom row shows the same simulation from a different view at a later time step. Notice that only our merging-and-splitting approach can produce a stable fracture simulation with pleasing result.

cloth from two different heights. The ball is simulated using peridynamics [1] with implicit integration and a high threshold that prevents fracturing. The cloth is simulated using a mass-spring system within the same implicit integration system used for peridynamics. As can be seen in the figure, using a weak penalty or SPH-based force fails to resolve penetration. Increasing the stiffness of the force formulation helps, but when the force is too strong with either force-based formulation, it can make the simulation system highly unstable. In practice, such instabilities can be avoided by carefully tuning the stiffness parameter, but the right stiffness values depend on the collision scenario. Notice that the stiffness that works for the top row does not work for the bottom row showing faster impact and vice-versa. Impulse-based collisions fail to prevent penetration, since they handle colliding particles in isolation using instant velocity updates. Our merging-and-splitting approach completely prevents penetration and properly resolves the collisions without introducing instabilities for both cases. Notice that the deformations of the cloth with merging-and-splitting are similar to the deformations achieved with force-based formulations using the right stiffness parameters.

Similar tests involving solid-fluid coupling are shown in Fig. 4. In these tests, a cylinder-shaped fluid column with high velocity falls onto an elastic cloth. The fluid particles are simulated using SPH [3] with explicit integration and the elastic cloth is simulated using a mass-spring system with implicit integration. The solid-fluid coupling is handled entirely via particle-level collisions. As can be seen in the figure, penetrations cannot be avoided with weak penalty or SPH-based forces. Stronger forces lead to fluid particles bouncing back with high velocity. Again, these problems can be avoided by tuning the stiffness parameter accordingly. Impulse-based collisions cannot prevent penetration either and consistent low-velocity impacts without position correction lead to fluid particles slowly passing through the cloth layer. Our merging-and-splitting method produces perfect separation between the fluid and the solid systems and completely prevents penetration without introducing instabilities.

Particle-based fracture simulations using peridynamics are typically handled using penalty forces [1], which require extremely small time step size ($\Delta t \approx 10^{-7}$) and parameter tuning until a desirable animation is produced.

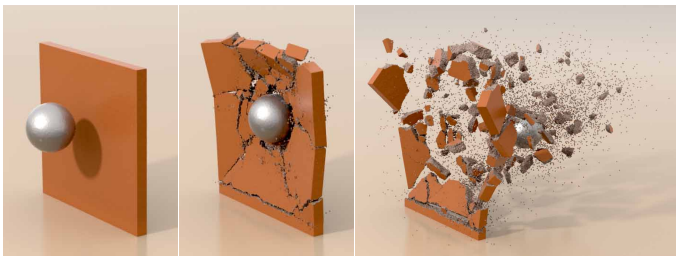


**Fig. 6:** Frames from our simulation with high impact collision between a ball and a brittle wall using peridynamics with merging-and-splitting.

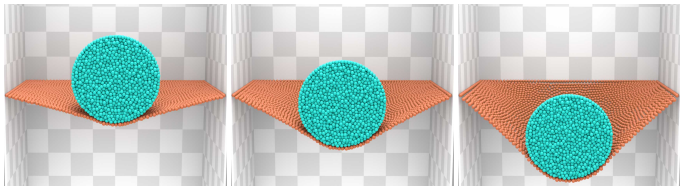



**Fig. 7: Different mass ratios:** cross-section views of simulations including a soft elastic cloth hit by a solid ball made up of the same or heavier particles using our merging-and-splitting method.

This is presented in Fig. 5 with a brittle wall hit by a high-velocity ball with $20\times$ heavier particles, both simulated using peridynamics. When the collision forces are too strong, the wall crumbles into tiny pieces. Weak forces or impulse-based collisions lead to penetrations before fracture. Eventually, all collision handling methods lead to fracture, but they differ by the amount of inter-penetration occurring prior to fracture and the visual quality of the fracture. Our merging-and-splitting scheme, in comparison, produces stable and expected results without any parameter tuning (Fig. 6).

Our merging-and-splitting method can stably handle variations in mass, since meta particles allow significant energy transfer between objects during each time step without introducing instabilities, as demonstrated in Fig. 7.

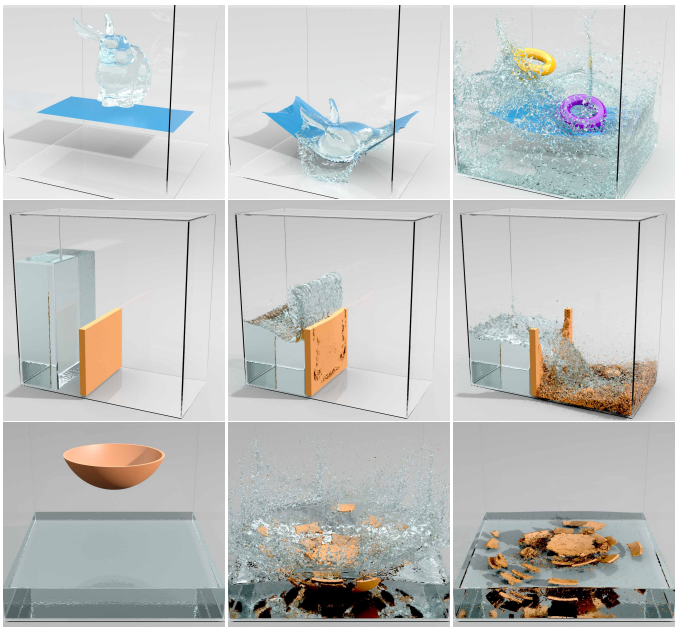


**Fig. 8:** Example frames from our simulation tests for coupling peridynamics and SPH using merging-and-splitting.

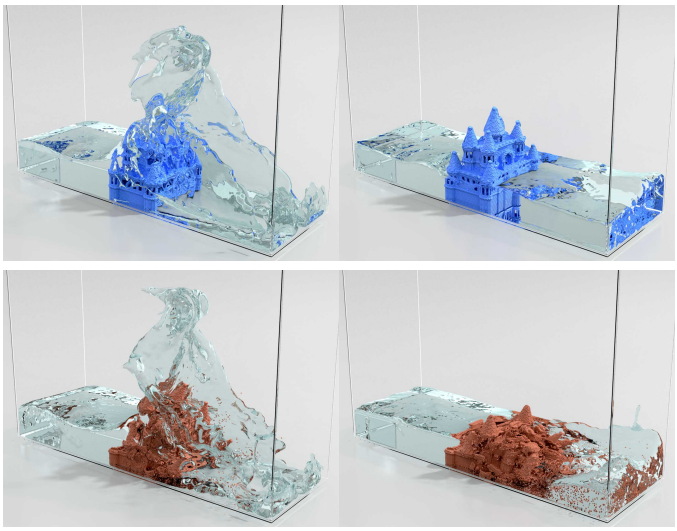


**Fig. 9:** Frames from our simulation system coupling FLIP with (top) mass-spring system and (bottom) peridynamics.

**TABLE 1:** Average computation times per time step.

| Scene | Fig. 6 | Fig. 8 top | Fig. 8 middle | Fig. 8 bottom | Fig. 9 top | Fig. 9 bottom |
|---|---|---|---|---|---|---|
| Solid Particles | 247.9K | 102K | 112.5K | 121.9K | 191K | 191K |
| Fluid Particles | – | 845K | 655K | 6.5M | 1.4M | 1.4M |
| Collision Detection | 53 ms | 73 ms | 270 ms | 392 ms | 42 ms | 43 ms |
| **Particle Merging** | 3.1 ms | 34.0 ms | 66.5 ms | 222.6 ms | 40.5 ms | 37.0 ms |
| Integration Stage-1 | 0.8 sec | 0.5 sec | 2.6 sec | 12.9 sec | 4.5 sec | 2.2 sec |
| Integration Stage-2 | 0.7 sec | 0.4 sec | 1.9 sec | 12.1 sec | 4.2 sec | 2.1 sec |
| **Particle Splitting** | 0.4 ms | 0.4 ms | 8.9 ms | 18.2 ms | 10.4 ms | 9.7 ms |
| Total Time | 3.5 sec | 1.6 sec | 5.3 sec | 27.4 sec | 10.6 sec | 6.4 sec |

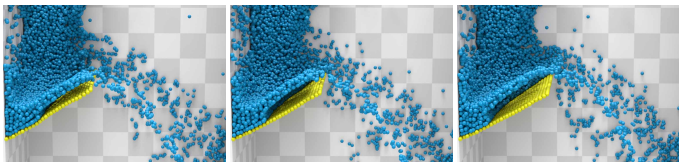
### 6.2 Solid-Fluid Coupling

Different examples of solid-fluid coupling using merging-and-splitting are shown in Fig. 8. In these examples the fluid particles are simulated using SPH, the cloth is simulated using a mass-spring system, and the brittle solid objects are simulated using peridynamics. The cloth model contains only a single layer of particles. Nonetheless, no fluid particle penetrates through the cloth surface. The interactions using merging-and-splitting provide two-way coupling between the fluid and the cloth model as well as the cloth and the tori. The other examples, showing SPH and peridynamics coupling, present solid fracturing due to fluid interaction, enabling new forms of simulation scenarios that can be robustly handled using our merging-and-splitting approach. In the dam break example, the first impact of fluid particles lead to small fractures around the wall. These fractures form weak points that ultimately break the wall and the broken pieces are carried away by the fluid. The next example shows a bowl breaking as it hits the water surface. The reflected waves push the bottom part of the bowl up, together with fluid particles that were previously gathered on it, forming a secondary splash. All of these complex examples provide clear evidence of two-way coupling between the two simulation systems, including high-velocity impact situations that are traditionally challenging to handle robustly.

Fig. 9 shows similar coupling examples using FLIP and a mass-spring system or peridynamics, simulated using implicit integration. Although FLIP typically uses massless marker particles, we assigned mass to fluid particles for simulating interactions with solid particles using our merging and splitting method. As can be seen in the figure, similar levels of complex interactions between solid and fluid simulations can be achieved using FLIP as well. Since FLIP uses an Eulerian pressure solver, in our implementation, coupling FLIP with another particle-based simulation system also involves marking the grid cells occupied by *all* particles in both systems. This is necessary to ensure that the pressure projection step of FLIP correctly identifies which cells are occupied with particles and which cells are empty. Other than this minor modification to FLIP, the two simulation systems handling solids and fluids run separately, exchanging information via merging and splitting alone.

### 6.3 Performance

The performance of our simulations largely depends on the performances of the underlying particle-based systems used. The performance results of our tests are included in the Table 1. Notice that particle merging and particle splitting operations take only a negligible fraction of the computation time. Most of the computation time is spent in the integration steps. In particular, peridynamics integration, involving an implicit solver for a large number of particles, each of which is connected to hundreds of other particles via stiff springs, can be considerably slow. Yet, because our implementation uses a two-stage integration scheme, our merging-and-splitting implementation effectively doubles the computation time (by introducing an additional integration step). This additional overhead can be reduced by using the result of the first integration step as the initial guess for the second integration step, an optimization that is not included in our tests.



(a) $\alpha = 0, \beta = 0$ (b) $\alpha = 1, \beta = 0$ (c) $\alpha = 1, \beta = 1$

**Fig. 10:** Cross-section view of a simple solid-fluid coupling test, including a column of fluid falling onto an elastic cloth model, demonstrating the impact of different energy conservation parameters: (a) no energy preservation with $\alpha = 0$ and $\beta = 0$, (b) full energy preservation for merging using $\alpha = 1$ and no energy preservation with velocity synchronization using $\beta = 0$, and (c) full energy preservation with $\alpha = 1$ and $\beta = 1$.

### 6.4 Parameters

Our merging-and-splitting approach has only $\alpha$ and $\beta$ parameters that control the energy conservation behavior for collisions and coupling different integrators (Equation 20). With $\alpha = \beta = 1$ all merging-and-splitting operations fully conserve energy and all energy loss is due to damping or time-step integration. Using $\alpha = \beta = 0$, a portion of the energy is still conserved, but only as needed for momentum conservation, and merging-and-splitting operations dissipate the rest. The impact of different $\alpha$ and $\beta$ parameters with merging-and-splitting is shown with a simple example in Fig. 10. Notice that energy conservation in this example has a relatively minor impact on the final result, causing the fluid to jump back slightly higher after impact. This is typical for all simulations we have tested, but it may be possible to design counter examples where energy conservation can play a more prominent role.

Our implementation also includes $n_{\min}$ and $n_{\max}$ parameters that limit the number of particles within a meta-particle. Fig. 11 shows impact of limiting the number of particles in a meta-particle. In this example, a block of fluid fall onto an elastic cloth, such that the initial impact covers the entire cloth surface (Fig. 11a). Therefore, allowing an unlimited number of particles to merge into a single meta-particle effectively rigidifies the entire cloth, significantly impacting the simulation outcome (Fig. 11b). In this case, the rigidification remains permanent, as the solid-fluid contact persists, preventing any deformation. Limiting the number of particles in a meta-particle using limit parameters $n_{\min}$ and $n_{\max}$ leads to similar results (Fig. 11c-e). However, when the limit is too small, it prevents properly resolving some collision events, failing to stop some fluid particles to pass through the cloth model (Fig. 11f). While this example has been carefully chosen to present the problems of using a very small limit (i.e. $n_{\min} = 4$) or an unlimited number of particles in a meta-particle, with all our experiments (including this one) we observed similar results when using parameter values for $n_{\min}$ and $n_{\max}$ set to 8 to 64, respectively. Therefore, we conclude that our implementation is not sensitive to the values of the limit parameters for the scenes we tested.

## 7 Discussion

Conceptually, our merging-and-splitting scheme can be considered similar to impulse-based collisions. However, unlike explicit impulse-based collisions that instantaneously resolve the collisions, we keep the particles in contact for the entire duration of a time step. This is a crucial component of our method, which allows colliding objects to exchange a substantial amount of momentum, beyond what can be stably accomplished using force-based or impulse-based formulations within a time step. When using explicit integration, this guarantees that the particles do not penetrate further during the time step. With implicit integration, merging allows information exchange between different colliding bodies through the meta-particles while solving the implicit system. This makes merging-and-splitting particularly favorable for implicit integration.

Since meta-particles are split at the end of each time step with relative velocities of colliding particles guaranteed to be pointing away from each other, colliding particles should not be considered "glued," though they maintain contact throughout a time step. Meta-particles merely facilitate neighboring particles on either side to exchange information.

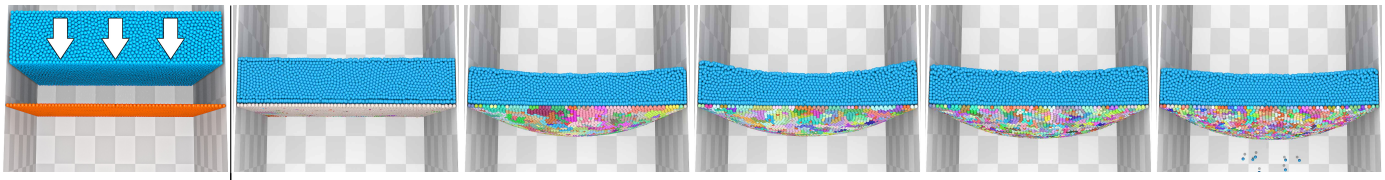
Note that most of the problems we present regarding force-based collision formulations can be resolved by increasing the stiffness and reducing the time-step size or by solving for the magnitude of the force/impulse within an implicit system.

Position correction is a common approach used in computer graphics for preventing penetrations. In fact, position correction (along with parameter tuning) plays a crucial role in various simulations in prior work using force-based collision models to demonstrate seemingly stable collision responses. However, position correction artificially injects or removes energy. This can have catastrophic results especially with fracture simulations of brittle materials using peridynamics. Furthermore, position correction may move particles into collisions with other particles. Therefore, we have entirely avoided position corrections in all our tests in this paper. Consequently, we do not compare our approach to position-based dynamics [45], [82], [83], which completely relies on position updates.

Merging-and-splitting using the implementation we describe in this paper can properly handle rest-in-contact situations. When a particle rests on another particle, both with no velocity, they are not merged before our first integration step. After the first integration step, however, the resulting velocity changes indicate that the particles must be merged. As a result, they preserve their relative positions during the recomputation in the second integration step.

While our merging and splitting operations are based on the principles of momentum and energy conservation, we impose no restrictions on the integrators, which are treated as black-boxes. Therefore, the velocity update provided by the integrators do not necessarily conserve energy or momentum.

While we have used similar particle sizes in most of our tests, merging-and-splitting does not inherently require the particles to have a uniform size. Particles with significantly different sizes can be handled, as long as collisions can be



(a) Initial conditions | (b) $n_{\min} = \infty$, $n_{\max} = \infty$ | (c) $n_{\min} = 32$, $n_{\max} = 64$ | (d) $n_{\min} = 16$, $n_{\max} = 32$ | (e) $n_{\min} = 8$, $n_{\max} = 16$ | (f) $n_{\min} = 4$, $n_{\max} = 8$

**Fig. 11:** A block of fluid falling with gravity onto a soft elastic cloth: (a) the initial conditions before the collision and (b-f) the simulation state a few seconds after the collision, comparing different parameters for $n_{\min}$, and $n_{\max}$ that limit the number of particles in a meta-particle. Cloth particles having the same color are merged into the same meta-particle. Notice that (b) using an unlimited number of particles in a meta-particle with $n_{\min} = n_{\max} = \infty$ leads to rigidification and (f) using too small limits causes leaks.

safely detected. Similarly, if the particle distribution is not dense enough and that the material includes large-enough holes, penetrations may occur.

The orthogonal momentum exchange in Equation 12 can be considered a form of frictional contact. Yet, this is not a physically-based friction formulation. Instead, our formulation aims to minimize orthogonal momentum transfer between colliding particles. Thus, the lack of a physically-based friction model is a limitation of our meta-particle splitting formulation. It is important to note that properly modeling friction may require modifying time-step integration accordingly, which we deliberately avoided to provide a general coupling solution for otherwise incompatible simulation systems.

Another limitation of our formulation is that we assume that meta-particles maintain the relative positions of colliding particles throughout the time step. An interesting future direction would be exploring angular momentum conservation for meta particles and introducing rotations during the time-step integration or prior to splitting. This could be handled by treating meta-particles as rigid bodies, similar to rigid impact zones [19]. However, this would also require modifying the particle-based simulation system accordingly, so that it can handle rigid bodies along with particles.

## 8 Conclusion

We have introduced merging-and-splitting, a new model for robustly handling collisions with particle-based simulations. This approach also allows coupling different particle-based simulation systems using different integrators that are designed for representing different material types and phases. We have shown that our method is effective in handling collision within a simulation system and coupling separate simulations of different materials. We have also shown novel simulation examples involving solid fracture due to fluid interaction.

An interesting direction for future research would be testing the effectiveness of the merging-and-splitting approach for reproducing macro-scale behavior in solid-fluid coupling, such as computing drag and lift forces, capillary effect, as well as buoyancy. Also, using merging-and-splitting for coupling different fluids, such as gasses and liquids, may reveal interesting challenges and new applications for future research. In addition, since merging-and-splitting only considers kinetic energies and linear momenta of the colliding particles, taking angular momentum into account with merging-and-splitting would be an interesting future research direction for physics-based animation.

## References


[1] J. A. Levine, A. W. Bargteil, C. Corsi, J. Tessendorf, and R. Geist, "A peridynamic perspective on spring-mass fracture," in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA '14, 2014, pp. 47–55.

[2] M. Desbrun and M.-P. Gascuel, "Smoothed particles: A new paradigm for animating highly deformable bodies," in *Proceedings of the Eurographics Workshop on Computer Animation and Simulation '96*, 1996, pp. 61–76.

[3] M. Müller, D. Charypar, and M. Gross, "Particle-based fluid simulation for interactive applications," in *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA '03, 2003, pp. 154–159.

[4] Y. Zhu and R. Bridson, "Animating sand as a fluid," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 965–972, Jul. 2005.

[5] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer, "Elastically deformable models," in *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '87, 1987, pp. 205–214.

[6] D. Terzopoulos, J. Platt, and K. Fleischer, "Heating and melting deformable models (From Goop to Glop)," in *Graphics Interface '89*, Jun. 1989, pp. 219–226.

[7] M. Moore and J. Wilhelms, "Collision detection and response for computer animation," *SIGGRAPH Comput. Graph.*, vol. 22, no. 4, pp. 289–298, Jun. 1988.

[8] D. Tonnesen, "Modeling Liquids and Solids using Thermal Particles," *Proceedings of Graphics Interface*, pp. 255–262, 1991.

[9] S. Hasegawa, N. Fujii, K. Akahane, Y. Koike, and M. Sato, "Real-time rigid body simulation for haptic interactions based on contact volume of polygonal objects," *Transactions of the Society of Instrument and Control Engineers*, vol. 40, no. 2, pp. 122–131, 2004.

[10] F. Faure, S. Barbier, J. Allard, and F. Falipou, "Image-based collision detection and response between arbitrary volume objects," in *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA '08, 2008, pp. 155–162.

[11] E. Drumwright, "A fast and stable penalty method for rigid body simulation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 1, pp. 231–240, Jan. 2008.

[12] B. Heidelberger, M. Teschner, R. Keiser, M. Müller, and M. Gross, "Consistent penetration depth estimation for deformable collision response," *Proceedings of Vision, Modeling, Visualization*, pp. 339–346, 2004.

[13] B. Mishra, "A review of computer simulation of tumbling mills by the discrete element method: Part i—contact mechanics," *International Journal of Mineral Processing*, vol. 71, no. 1, pp. 73 – 93, 2003.

[14] M. Tang, D. Manocha, M. A. Otaduy, and R. Tong, "Continuous penalty forces," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 107:1–107:9, Jul. 2012.

[15] D. Harmon, E. Vouga, B. Smith, R. Tamstorf, and E. Grinspun, "Asynchronous contact mechanics," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 87:1–87:12, Jul. 2009.

[16] D. Harmon, Q. Zhou, and D. Zorin, "Asynchronous integration with phantom meshes," in *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation - SCA '11*, Aug. 2011, p. 247.

[17] D. Baraff and A. Witkin, "Large steps in cloth simulation," in *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '98, 1998, pp. 43–54.

[18] X. Provot, "Collision and self-collision handling in cloth model dedicated to design garments," in *Computer Animation and Simulation '97*, 1997, pp. 177–189.

[19] R. Bridson, R. Fedkiw, and J. Anderson, "Robust treatment of collisions, contact and friction for cloth animation," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 594–603, Jul. 2002.

[20] A. Selle, J. Su, G. Irving, and R. Fedkiw, "Robust high-resolution cloth using parallelism, history-based collisions, and accurate friction," *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 2, pp. 339–350, Mar. 2009.

[21] D. Koschier, J. Bender, B. Solenthaler, and M. Teschner, "Smoothed particle hydrodynamics for physically-based simulation of fluids and solids," in *EUROGRAPHICS 2019 Tutorials*. The Eurographics Association, 2019.

[22] M. Becker, H. Tessendorf, and M. Teschner, "Direct Forcing for Lagrangian Rigid-Fluid Coupling," *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, pp. 493–503, 2009.

[23] X. Shao, Z. Zhou, J. Zhang, and W. Wu, "Lagrangian particle-based simulation of fluid-solid coupling on graphics processing units," *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields*, 2014.

[24] N. Akinci, M. Ihmsen, G. Akinci, B. Solenthaler, and M. Teschner, "Versatile rigid-fluid coupling for incompressible SPH," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 62:1–62:8, Jul. 2012.

[25] N. Akinci, J. Cornelis, G. Akinci, and M. Teschner, "Coupling elastic solids with smoothed particle hydrodynamics fluids," *Computer Animation and Virtual Worlds*, vol. 24, no. 3-4, pp. 195–203, 2013.

[26] O. Genevaux, A. Habibi, and J. michel Dischler, "Simulating fluid-solid interaction," in *in Graphics Interface*, 2003, pp. 31–38.

[27] M. Muller, S. Schirm, M. Teschner, B. Heidelberger, and M. Gross, "Interaction of fluids with deformable solids," *Computer Animation and Virtual Worlds*, vol. 15, no. 3-4, pp. 159–171, 2004.

[28] L. Yang, S. Li, A. Hao, and H. Qin, "Realtime two-way coupling of meshless fluids and nonlinear fem," *Comp. Graph. Forum*, vol. 31, no. 7pt1, pp. 2037–2046, Sep. 2012.

[29] X. Shao, Z. Zhou, N. Magnenat-Thalmann, and W. Wu, "Stable and fast fluid–solid coupling for incompressible SPH," *Computer Graphics Forum*, 2015.

[30] P. Du, M. Tang, C. Meng, R. Tong, and L. Lin, "A fluid/cloth coupling method for high velocity collision simulation," in *Proceedings of the 11th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry*, ser. VRCAI '12, 2012, pp. 309–314.

[31] M. Huber, B. Eberhardt, and D. Weiskopf, "Boundary handling at cloth-fluid contact," *Computer Graphics Forum*, vol. 34, no. 1, pp. 14–25, 2015.

[32] D. Koschier and J. Bender, "Density maps for improved SPH boundary handling," in *Proceedings of the 2017 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA '17. ACM, 2017, pp. 1–10.

[33] J. Bender, T. Kugelstadt, M. Weiler, and D. Koschier, "Volume maps: An implicit boundary representation for SPH," in *Motion, Interaction and Games*, ser. MIG '19, 2019.

[34] J. Cornelis, M. Ihmsen, A. Peer, and M. Teschner, "Liquid boundaries for implicit incompressible SPH," *Computers & Graphics*, vol. 52, pp. 72 – 78, 2015.

[35] S. Band, C. Gissler, A. Peer, and M. Teschner, "Mls pressure boundaries for divergence-free and viscous SPH fluids," *Computers & Graphics*, vol. 76, pp. 37 – 46, 2018.

[36] S. Band, C. Gissler, M. Ihmsen, J. Cornelis, A. Peer, and M. Teschner, "Pressure boundaries for implicit incompressible SPH," *ACM Trans. Graph.*, vol. 37, no. 2, Feb. 2018.

[37] J. Bender, T. Kugelstadt, M. Weiler, and D. Koschier, "Implicit frictional boundary handling for SPH," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 10, pp. 2982–2993, 2020.

[38] C. Gissler, A. Peer, S. Band, J. Bender, and M. Teschner, "Interlinked SPH pressure solvers for strong fluid-rigid coupling," *ACM Trans. Graph.*, vol. 38, no. 1, Jan. 2019.

[39] J. K. Hahn, "Realistic animation of rigid bodies," in *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '88, 1988, pp. 299–308.

[40] B. Mirtich and J. Canny, "Impulse-based simulation of rigid bodies," in *Proceedings of the 1995 Symposium on Interactive 3D Graphics*, ser. I3D '95, 1995, pp. 181–ff.

[41] E. Guendelman, R. Bridson, and R. Fedkiw, "Nonconvex rigid bodies with stacking," *ACM Trans. Graph.*, vol. 22, no. 3, pp. 871–878, Jul. 2003.

[42] R. Weinstein, J. Teran, and R. Fedkiw, "Dynamic simulation of articulated rigid bodies with contact and collision," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 3, pp. 365–374, May 2006.

[43] E. Vouga, B. Smith, D. M. Kaufman, R. Tamstorf, and E. Grinspun, "All's well that ends well: Guaranteed resolution of simultaneous rigid body impact," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 151:1–151:19, Jul. 2017.

[44] J. Bender and A. Schmitt, "Constraint-based collision and contact handling using impulses," in *Proceedings of the 19th international conference on computer animation and social agents*, Geneva (Switzerland), Jul. 2006, pp. 3–11.

[45] M. Macklin, M. Müller, N. Chentanez, and T.-Y. Kim, "Unified particle physics for real-time applications," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 153:1–153:12, Jul. 2014.

[46] A. Witkin and D. Baraff, "Physically based modeling: Principles and practice," in *Online Siggraph '97 Course Notes*, 1997.

[47] K. Erleben, "Numerical methods for linear complementarity problems in physics-based animation," in *ACM SIGGRAPH 2013 Courses*, ser. SIGGRAPH '13, 2013, pp. 8:1–8:42.

[48] P. Lötstedt, "Numerical simulation of time-dependent contact and friction problems in rigid body mechanics," *SIAM Journal on Scientific and Statistical Computing*, vol. 5, no. 2, pp. 370–393, 1984.

[49] D. E. Stewart and J. C. Trinkle, "An Implicit Time-Stepping Scheme for Rigid Body Dynamics with Inelastic Collisions and Coulomb Friction," *International Journal for Numerical Methods in Engineering*, vol. 39, pp. 2673–2691, Aug. 1996.

[50] M. Anitescu and F. A. Potra, "Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems," *Nonlinear Dynamics*, vol. 14, no. 3, pp. 231–247, 1997.

[51] C. Duriez, F. Dubois, A. Kheddar, and C. Andriot, "Realistic haptic rendering of interacting deformable objects in virtual environments," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 1, pp. 36–47, Jan. 2006.

[52] M. A. Otaduy, R. Tamstorf, D. Steinemann, and M. Gross, "Implicit Contact Handling for Deformable Objects," *Computer Graphics Forum*, 2009.

[53] I. Alduán and M. A. Otaduy, "SPH granular flow with friction and cohesion," in *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA '11, 2011, pp. 25–32.

[54] C. Batty, F. Bertails, and R. Bridson, "A fast variational framework for accurate solid-fluid coupling," *ACM Trans. Graph.*, vol. 26, no. 3, Jul. 2007.

[55] N. Chentanez and M. Müller, "A multigrid fluid pressure solver handling separating solid boundary conditions," in *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA '11, 2011, pp. 83–90.

[56] M. Pauly, D. K. Pai, and L. J. Guibas, "Quasi-rigid objects in contact," in *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA '04, 2004, pp. 109–119.

[57] D. Baraff, "Fast contact force computation for nonpenetrating rigid bodies," in *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '94, 1994, pp. 23–34.

[58] J. Trinkle, J. Pang, S. Sudarsky, and G. Lo, "On dynamic multi-rigid-body contact problems with coulomb friction," Texas A&M University, Tech. Rep., 1995.

[59] P. G. Kry and D. K. Pai, "Continuous contact simulation for smooth surfaces," *ACM Trans. Graph.*, vol. 22, no. 1, pp. 106–129, Jan. 2003.

[60] Y. R. Fei, H. T. Maia, C. Batty, C. Zheng, and E. Grinspun, "A multi-scale model for simulating liquid-hair interactions," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 56:1–56:17, Jul. 2017.

[61] Y. R. Fei, C. Batty, E. Grinspun, and C. Zheng, "A multi-scale model for simulating liquid-fabric interactions," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 51:1–51:16, Jul. 2018.

[62] A. Stomakhin, C. Schroeder, L. Chai, J. Teran, and A. Selle, "A material point method for snow simulation," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 102:1–102:10, Jul. 2013.

[63] A. Stomakhin, C. Schroeder, C. Jiang, L. Chai, J. Teran, and A. Selle, "Augmented mpm for phase-change and varied materials," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 138:1–138:11, Jul. 2014.

[64] A. P. Tampubolon, T. Gast, G. Klár, C. Fu, J. Teran, C. Jiang, and K. Museth, "Multi-species simulation of porous sand and water mixtures," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 105:1–105:11, Jul. 2017.

[65] G. Daviet and F. Bertails-Descoubes, "A semi-implicit material point method for the continuum simulation of granular materials," *ACM Trans. Graph.*, vol. 35, no. 4, pp. 102:1–102:13, Jul. 2016.

[66] G. Klár, T. Gast, A. Pradhana, C. Fu, C. Schroeder, C. Jiang, and J. Teran, "Drucker-prager elastoplasticity for sand animation," *ACM Trans. Graph.*, vol. 35, no. 4, pp. 103:1–103:12, Jul. 2016.

[67] D. Ram, T. Gast, C. Jiang, C. Schroeder, A. Stomakhin, J. Teran, and P. Kavehpour, "A material point method for viscoelastic fluids, foams and sponges," in *Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, ser. SCA '15, 2015, pp. 157–163.

[68] Y. Yue, B. Smith, C. Batty, C. Zheng, and E. Grinspun, "Continuum foam: A material point method for shear-dependent flows," *ACM Trans. Graph.*, vol. 34, no. 5, pp. 160:1–160:20, Nov. 2015.

[69] Y. Fang, M. Li, M. Gao, and C. Jiang, "Silly rubber: An implicit material point method for simulating non-equilibrated viscoelastic and elastoplastic solids," *ACM Trans. Graph.*, vol. 38, no. 4, Jul. 2019.

[70] C. Jiang, T. Gast, and J. Teran, "Anisotropic elastoplasticity for cloth, knit and hair frictional contact," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 152:1–152:14, Jul. 2017.

[71] J. Wolper, Y. Chen, M. Li, Y. Fang, Z. Qu, J. Lu, M. Cheng, and C. Jiang, "Anisompm: Animating anisotropic damage mechanics," *ACM Trans. Graph.*, vol. 39, no. 4, Jul. 2020.

[72] M. Gao, A. Pradhana, X. Han, Q. Guo, G. Kot, E. Sifakis, and C. Jiang, "Animating fluid sediment mixture in particle-laden flows," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 149:1–149:11, Jul. 2018.

[73] Y. Fang, Z. Qu, M. Li, X. Zhang, Y. Zhu, M. Aanjaneya, and C. Jiang, "Iq-mpm: An interface quadrature material point method for non-sticky strongly two-way coupled nonlinear solids and fluids," *ACM Trans. Graph.*, vol. 39, no. 4, Jul. 2020.

[74] S. Wang, M. Ding, T. F. Gast, L. Zhu, S. Gagniere, C. Jiang, and J. M. Teran, "Simulation and visualization of ductile fracture with the material point method," *Proc. ACM Comput. Graph. Interact. Tech.*, vol. 2, no. 2, Jul. 2019.

[75] X. Han, T. F. Gast, Q. Guo, S. Wang, C. Jiang, and J. Teran, "A hybrid material point method for frictional contact with diverse materials," *Proc. ACM Comput. Graph. Interact. Tech.*, vol. 2, no. 2, Jul. 2019.

[76] C. Farhat and N. Sobh, "A consistency analysis of a class of concurrent transient implicit/explicit algorithms," *Computer Methods in Applied Mechanics and Engineering*, vol. 84, no. 2, pp. 147 – 162, 1990.

[77] D. Olson, P. Bochev, M. Luskin, and A. V. Shapeev, "Development of an optimization-based atomistic-to-continuum coupling method," in *Large-Scale Scientific Computing*, 2014, pp. 33–44.

[78] N. M. Newmark, "A method of computation for structural dynamics," *Journal of the Eng. Mech. Div.*, vol. 85, no. 3, 1959.

[79] S. Redon, A. Kheddar, and S. Coquillart, "Fast Continuous Collision Detection between Rigid Bodies," *Computer Graphics Forum*, 2002.

[80] H. Bhattacharya, Y. Gao, and A. W. Bargteil, "A level-set method for skinning animated particle data," in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, Aug 2011.

[81] C. Watcharopas, Y. Sapra, R. Geist, and J. A. Levine, "Extracting surface geometry from particle-based fracture simulations," in *Advances in Visual Computing*, ser. Lecture Notes in Computer Science, 2015, vol. 9474, pp. 82–91.

[82] M. Müller, B. Heidelberger, M. Hennix, and J. Ratcliff, "Position Based Dynamics," in *Workshop on Virtual Reality Interaction and Physical Simulation*, 2006.

[83] J. Bender, M. Müller, M. A. Otaduy, M. Teschner, and M. Macklin, "A survey on position-based simulation methods in computer graphics," *Comput. Graph. Forum*, vol. 33, no. 6, pp. 228–251, Sep. 2014.


**Nghia Truong** received a PhD degree in Scientific Computing from School of Computing, University of Utah in 2020, under the supervision of Prof. Cem Yuksel. Prior to starting his PhD program, he completed a BS and MS degrees in Applied Mathematics and Computer Science from Tula State University, Russian Federation. His research interests are in the related fields of computer graphics, machine learning/deep learning, and data science.

**Cem Yuksel** is an associate professor in the School of Computing at the University of Utah. Previously, he was a postdoctoral fellow at Cornell University, after receiving his PhD in Computer Science from Texas A&M University in 2010. His research interests are in computer graphics and related fields, including physically-based simulations, rendering techniques, global illumination, sampling, GPU algorithms, graphics hardware, knitted structures, and hair modeling, animation, and rendering.

**Chakrit Watcharopas** is an assistant professor in the Department of Computer Science at Kasetsart University, Thailand. Previously, he received his PhD in Computer Science from Clemson University in 2004 and the MS degree in Computer Science from University of Southern California in 1997. He is currently the Deputy of the Department of Computer Science at Kasetsart and a member of the Artificial Intelligence Innovation for Healthtech (Kasetsart) where he is also the director of the Deep AI-Assisted Healthcare Group. His research interests include fracture simulation, surface extraction, phase transition, computer graphics, and deep learning for medical data and imaging.

**Joshua A. Levine** is an associate professor in the Department of Computer Science at University of Arizona. Previously, he was an assistant professor at Clemson University from 2012 to 2016, and a postdoctoral research associate at the University of Utah's SCI Institute from 2009 to 2012. He received his PhD in Computer Science from Ohio State University (2009) after completing BS degrees in Computer Engineering and Mathematics (2003) and an MS in Computer Science (2004) from Case Western Reserve University. His research interests include visualization, geometric modeling, topological analysis, mesh generation, and computer graphics.

**Robert M. (Mike) Kirby** received the MS and PhD degrees in applied mathematics, and the MS degree in computer science from Brown University, Providence, in 1999, 2002, and 2001, respectively. He is currently the Executive Director of the Utah Informatics Initiative (UI2), the Interim Director of the Scientific Computing and Imaging (SCI) Institute, and a Professor of Computing and the Associate Director of the School of Computing, University of Utah. His current research interests include scientific and data computing and visualization. He is a Senior Member of the IEEE.