Projective Peridynamics for Modeling Versatile Elastoplastic Materials

Xiaowei He, Huamin Wang, *Member, IEEE* and Enhua Wu, *Member, IEEE*

Abstract—Unified simulation of versatile elastoplastic materials and different dimensions offers many advantages in animation production, contact handling, and hardware acceleration. The unstructured particle representation is particularly suitable for this task, thanks to its simplicity. However, previous meshless techniques either need too much computational cost for addressing stability issues, or lack physical meanings and fail to generate interesting deformation behaviors, such as the Poisson effect. In this paper, we study the development of an elastoplastic model under the state-based peridynamics framework, which uses integrals rather than partial derivatives in its formulation. To model elasticity, we propose a unique constitutive model and an efficient iterative simulator solved in a projective dynamics way. To handle plastic behaviors, we incorporate our simulator with the Drucker-Prager yield criterion and a reference position update scheme, both of which are implemented under peridynamics. Finally, we show how to strengthen the simulator by position-based constraints and spatially varying stiffness models, to achieve incompressibility, particle redistribution, cohesion, and friction effects in viscoelastic and granular flows. Our experiments demonstrate that our unified, meshless simulator is flexible, efficient, robust, and friendly with parallel computing.

Index Terms—peridynamics, projective dynamics, position-based dynamics, elasticity, plasticity, viscoelasticity, granular flows.

1 Introduction

Elastoplastic materials exist everywhere and can exhibit a variety of complex behaviors. The dynamic modeling of this process in a one-dimensional space is simple and intuitive. However, if being extended to a higher-dimensional space, the material deformation and plastic yielding will become rather complicated due to the diversity of material stiffness and geometric topology. How to find a simulation method that can naturally handle various materials as well as the topological changes has always been the focus of computer graphics. While researchers have extensively studied the simulation of different deformable bodies by various models and tools, the ability to simulate them in a unified way is still rather limited.

Unstructured particles are powerful in dealing with topological changes, but its application on simulating elastoplastic bodies is still limited. Previous research on meshless simulation of deformable bodies has explored the use of the generalized moving least squares method (GMLS) [1], [2], the SPH method [3], and the local Petrov-Galerkin method [4], [5]. Since these meshless techniques are based on partial derivatives and the classical elasticity theory, the results could be very sensitive to the underlying particle distribution. Besides, they also have the difficulty in defining the partial derivative along the discontinuities. A possible solution to overcome those difficulties is to apply the position-based dynamics [6], [7], which uses the difference between the deformed shape and the deformation-free shape to drive unstructured particle motion. This method is fast and robust, but it cannot produce many interesting elastic material behaviors, especially the *Poisson effect*.

Recently, peridynamics has gained its popularity in meshless simulation of discontinuous deformation, such as fractures [8]. The original *bond-based peridynamics* model [9] assumes that the force between two neighboring particles depends on the two particles only. Silling and colleagues [10] generalized this model later into *state-based peridynamics*, by defining the force as a function of all of the particles in the neighborhood. While existing peridynamics simulators have demonstrated their potentials in elastoplastic simulation, they typically have to use very small time steps as shown in [11], making them inefficient and less attractive than they might otherwise be.

Motivated by the recent success of peridynamics and constraint-based techniques, we present a fast, robust, and meshless solver under the state-based peridynamics framework to simulate versatile elastoplastic materials. Since peridynamics uses integrals rather than partial derivatives to construct its dynamic equation, the property allows us to design a specific constitutive material model, under which the simulator can be solved in a projective dynamics way [12], [13]. Toward the development of our simulator, we made the following technical contributions.

- **Elasticity**. We present an iterative method based on implicit time integration, to efficiently simulate our new constitutive material model. This method can handle not only bulky bodies, but also thin shells and rods.

- *X. He is with the State Key Lab. of CS, Institute of Software, Chinese Academy of Sciences and the University of Chinese Academy of Sciences, Beijing, China, 100190.*
  E-mail: xiaowei@iscas.ac.cn
- *H. Wang is with the Ohio State University.*
  E-mail: whmin@cse.ohio-state.edu
- *E. Wu is with the State Key Lab. of CS, Institute of Software, Chinese Academy of Sciences and the University of Macau.*
  E-mail: ehwu@umac.mo

Manuscript received X X, XXXX; revised X X, XXXX.

- **Plasticity**. We propose a novel plastic deformation technique for our peridynamics simulator, by modifying reference positions to reflect permanent shape changes. This technique is equipped with the Drucker-Prager yield criterion formulated under peridynamics, and a novel way to estimate reference positions of the particles newly added into a neighborhood.
- **Viscoelastic and granular flows**. We demonstrate the ability of our simulator in animating viscoelastic and granular flows. This is achieved by using position-based constraints for incompressibility and particle redistribution, and spatially varying stiffness models for cohesion and friction.

The results demonstrate that our simulator can faithfully simulate elastoplastic bodies, shells, rods, viscoelastic fluids, and granular materials. Since the simulator handles all these materials in a purely meshless framework, it becomes straightforward to make material property transitions, as shown in Figure 19. The simulator can robustly handle large deformations and time steps, such as $h = 1/30$s in elastic body simulation. Finally, the whole simulator is efficient and it can be easily parallelized. So we expect to see a significant performance improvement on the GPU in the near future.

2 Other Related Work

Smoothed particle hydrodynamics. The use of smoothed particle hydrodynamics (SPH) in the simulation of Newtonian fluids has been widely studied in computer graphics. A detailed survey on this topic can be found in [14]. Meanwhile, researchers have explored the use of SPH in other simulation problems as well. An early SPH-based technique for animating deformable bodies was proposed by Desbrun and Gascuel [15]. Later Pavia and colleagues [16] used SPH to simulate non-Newtonian fluid flows, such as melting. Solenthaler and collaborators [3] developed a unified SPH model for simulating both fluids and deformable bodies. To address the rotational invariance issue, Becker and colleagues [17] presented a co-rotational SPH formulation for deformable bodies. Gerszewski and collaborators [18] extended a SPH solver to model elastoplastic materials. Alduán and Otaduy [19] used a predictive-corrective algorithm to calculate friction and cohesion forces in the simulation of granular materials. Also under the SPH framework, Jones and collaborators [20] introduced an embedded space approach for elastoplastic deformation, without thin features.

Particle-in-cell and its extensions. Particle-in-cell (PIC) methods and its extensions, including the material point (MPM) method [21], the fluid-implicit-particle (FLIP) method [22] and the affine particle-in-cell method [23], are commonly used in computer graphics to simulate granular materials, such as sand and snow. Additionally, researchers have also investigated the use of these methods in the simulation of melting and solidifying materials [24], non-Newtonian viscoplastic materials under hyperelastic models [25], granular materials [26] and viscoelastic fluids, foams and sponges [27]. The major difference between MPM and our method is that MPM calculates derivative terms, e.g., the deformation gradient, on a background grid while our method is purely Lagrangian and calculates all physical quantities on particles. This feature makes MPM better suited for modeling volumetric objects. As far as we know, how to model codimensional elastic objects like thin surfaces or curves under a single MPM framework remains unclear yet. Besides, it is also not an easy task for MPM to handle extremely large deformations, e.g., recovering a bunny from flat to its initial state. One remedy is by applying a hybrid method which integrates a Lagrangian framework into MPM, as was done recently in [28], [29]. Fortunately, our purely meshless framework can uniformly model materials with different dimensions and handle large deformations and timesteps.

Position based dynamics and projective dynamics. Position based dynamics (PBD) is different from traditional simulation approaches in that it uses geometric constraints rather than forces to model elastic behaviors. This characteristics enables PBD methods to robustly use large time steps in the simulation of mass-spring systems [6] and tetrahedral meshes [7]. Later, Macklin and Müller [30] extended the PBD idea to simulate incompressible fluid dynamics by unstructured particles as well. Based on all of these PBD methods, Macklin and colleagues [31] built a GPU-based system using both structured particles (as meshes) and unstructured particles. One problem associated with PBD is that its result is controlled by the number of iterations and the mesh resolution, rather than a model with physical meanings. Bouaziz and collaborators [13] solved this problem, by defining geometric constraints as strain energies in a quadratic form. This helps them develop a fast and robust projective dynamics simulator, which iteratively solves a local constraint step and a global linear system step. Recently, Tournier and colleagues [32] studied how to speed up constrained simulation of stiff deformable objects, at the expense of a larger system size.

3 Background

Let $\mathcal{H}$ be a 3D spherical neighborhood of radius r centered at the origin and $\mathcal{L}_m$ be the set of order-m tensors. State-based peridynamics [10] defines an order-m state as a function:

$$\underline{\mathbf{A}}\langle\cdot\rangle : \mathcal{H} \to \mathcal{L}_m. \tag{1}$$

For example, given an point $\mathbf{x}_i$ and any point $\mathbf{x}_j$ in its neighborhood, we have $\xi = \mathbf{x}_j - \mathbf{x}_i \in \mathcal{H}$ and we can formulate a *reference position vector state* of particle i as:

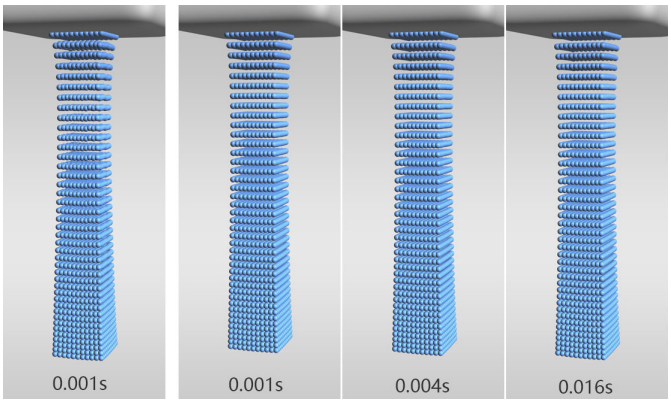
$$\underline{\mathbf{X}}\langle\xi\rangle = \mathbf{x}_j - \mathbf{x}_i, \quad \text{for } \xi \in \mathcal{H}. \tag{2}$$

For simplicity, we drop subscript i from state notations. Let $\mathbf{y}_i = \phi(\mathbf{x}_i)$ and $\mathbf{y}_j = \phi(\mathbf{x}_j)$ be the particle positions after deformation, we can also formulate a *deformation vector state* as:

$$\underline{\mathbf{Y}}\langle\xi\rangle = \mathbf{y}_j - \mathbf{y}_i, \quad \text{for } \xi \in \mathcal{H}. \tag{3}$$

Using $\underline{\mathbf{X}}$ and $\underline{\mathbf{Y}}$, we can calculate the deformation gradient tensor at $\mathbf{x}_i$ as: $\bar{\mathbf{F}} = (\underline{\mathbf{Y}} * \underline{\mathbf{X}})(\underline{\mathbf{X}} * \underline{\mathbf{X}})^{-1}$. Here the tensor product $\underline{\mathbf{A}} * \underline{\mathbf{B}}$ is defined as a reduction over the spherical volume:

$$\underline{\mathbf{A}} * \underline{\mathbf{B}} = \int_{\mathcal{H}} \underline{w}\langle\xi\rangle\, \underline{\mathbf{A}}\langle\xi\rangle \otimes \underline{\mathbf{B}}\langle\xi\rangle\, d\xi, \tag{4}$$



(a) MLS-based (b) Our method

Fig. 1. Beam test with three different timesteps ($0.001s$, $0.004s$ and $0.016s$). Unlike the MLS-based method [20], which is only stable at a small timestep $0.001s$ (a), our method is stable and consistent for all three timesteps (b).

where $\underline{w}\langle\xi\rangle$ is a weight function and $\otimes$ is the dyadic product operator. Intuitively, this calculation is similar to how shape matching [33] estimates linear transformation in a closed form. The difference is that shape matching tries to directly enforce rigid transformation in a position-based way, so it fails to produce many interesting deformation effects, such as Poisson effect. In contrast, state-based peridynamics uses a constitutive model to compute a force vector state $\underline{\mathbf{T}}$ from $\underline{\mathbf{Y}}$, and then guide the motion of particle i as:

$$\rho_i \ddot{\mathbf{y}}_i = \int_{\mathcal{H}} \left\{ \underline{\mathbf{T}}_i\langle\xi\rangle - \underline{\mathbf{T}}_j\langle -\xi\rangle \right\} d\xi + \mathbf{b}, \tag{5}$$

in which ρ_i is the density at point $\mathbf{x}_i$ and $\mathbf{b}$ represents body forces. Peridynamic constitutive models can be designed to match many hyperelastic constitutive models under the classical elasticity theory, as shown in [10].

4 Elastic Simulation

The main question involved in peridynamics is how to construct the constitutive model for elasticity. In this section, we present the design of a constitutive model using a quadratic elastic energy density function. This unique model allows us to solve peridynamics under implicit time integration in a projective dynamics fashion [12], [13], [34].

4.1 Projective Elastic Model Based on Integration

In continuum mechanics, the elastoplastic energy density function based on partial derivatives can be defined as [21],

$$\Psi = \mu \left\| \mathbf{F}_E - \mathbf{R}_E \right\|_F^2 + \frac{\lambda}{2}(J_E - 1)^2, \tag{6}$$

where $\mathbf{F}_E$ is the elastic part of the deformation gradient tensor $\mathbf{F}$, $J_E = \det \mathbf{F}_E$, $\mathbf{R}_E$ is the rotation, λ and μ are the Lamé parameters whose values may depend on the plastic part of the deformation gradient. This definition works fine for bulky solids. However, it has two limitations. One is that we need polar decomposition and special treatment to address singularity issues. Another reason is that it is not suitable for modeling bending elasticity, which is handled by a hinge-edge energy model in [13] instead. To address above problems, we aim to devise a new elastoplastic energy density function based on integration, i.e., integrate the

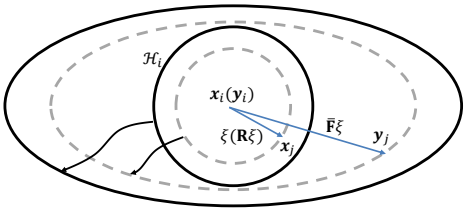


Fig. 2. Illustration of a sphere being symmetrically mapped into an ellipsoid.

energy stored on each pair of neighboring particles. Inspired by the finite element theory, we assume that all of the particles in a neighborhood are deformed by the same deformation gradient tensor $\bar{\mathbf{F}}$. In that case, the ideal position of a neighboring particle $\mathbf{x}_j$ can be obtained from a new deformation vector state: $\underline{\mathbf{Y}}^* = \bar{\mathbf{F}}\xi$, as shown in Figure 2. According to peridynamics, the first term of Equation 6 can be reformulated as

$$\Psi^{dev} = \int_{\mathcal{H}} \underline{w}\langle\xi\rangle \left(\mu \left\| \frac{\bar{\mathbf{F}}\xi - \mathbf{R}\xi}{|\underline{\mathbf{X}}|} \right\|^2 \right) d\xi \tag{7}$$

where we have ignored the subscripts $(\cdot)_E$ and $(\cdot)_F$. By looking into Figure 2, the term $\left\|\bar{\mathbf{F}}\xi - \mathbf{R}\xi\right\|$ actually measures the distance from the reference position $\mathbf{x}_j$ to its deformed position $\mathbf{y}_j$. If all the neighbors have already be transformed to the current configuration, the value of $\left\|\bar{\mathbf{F}}\xi - \mathbf{R}\xi\right\|$ can be approximated as $\left|\left\|\bar{\mathbf{F}}\xi\right\| - \left\|\mathbf{R}\xi\right\|\right|$ which has the advantage to remove the need to precompute the rotation matrix. Therefore, we can define our peridynamics-based elastic energy density function as

$$\Psi = \int_{\mathcal{H}} \underline{w}\langle\xi\rangle \left(\mu \underline{E}^{dev}\langle\xi\rangle + \frac{\lambda}{2} \underline{E}^{iso}\langle\xi\rangle \right) d\xi, \tag{8}$$

where $\underline{E}^{\mathtt{dev}} = \left(|\underline{\mathbf{Y}}^*|/|\underline{\mathbf{X}}| - 1\right)^2$, modeling the deviatoric part of the deformation energy, $\underline{E}^{\mathtt{iso}} = \left(|\underline{\mathbf{Y}}|/|\underline{\mathbf{X}}| - 1\right)^2$, modeling the isotropic part. The term $\underline{E}^{\mathtt{iso}}$ is designed to approximately model the second part of Equation 6 with spring-like forces.

To derive the elastic force density from $E^{\mathtt{dev}}$, we can calculate the following force vector state $\underline{\mathbf{T}}^{\mathtt{dev}}$ as:

$$\underline{\mathbf{T}}^{\mathtt{dev}} = -\mu \underline{w} \frac{\partial \underline{E}^{\mathtt{dev}}}{\partial \mathbf{y}_i} = -\frac{2\mu \underline{w}}{|\underline{\mathbf{X}}|^2} \left(1 - \frac{|\underline{\mathbf{X}}|}{|\underline{\mathbf{Y}}^*|} \right) \underline{\mathbf{Y}}^* \frac{\partial \underline{\mathbf{Y}}^*}{\partial \mathbf{y}_i}. \tag{9}$$

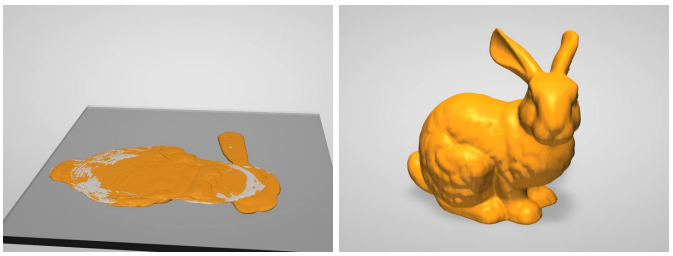
When the neighborhood is small and the deformation field is smooth, we have $\underline{\mathbf{Y}}^* \approx \underline{\mathbf{Y}}$. Based on this approximation, we drop the remaining derivative and reorganize Equation 9 into:

$$\underline{\mathbf{T}}^{\mathtt{dev}} \approx \frac{2\mu \underline{w}}{|\underline{\mathbf{X}}|^2} \left(\underline{\mathbf{Y}} - |\underline{\mathbf{X}}| \left(\mathtt{dir}\, \underline{\mathbf{Y}}^* \right) \right), \tag{10}$$

in which $\mathtt{dir}\, \underline{\mathbf{Y}}^*$ gives the normalized direction of $\underline{\mathbf{Y}}^*$. The reason we formulate the elastic force vector state into Equation 10 is to have two components: $\underline{\mathbf{Y}}$ and $\mathtt{dir}\, \underline{\mathbf{Y}}^*$. The deformation state $\underline{\mathbf{Y}}$ is a linear function of $\mathbf{y}_i$ and $\mathbf{y}_j$, while $\mathtt{dir}\, \underline{\mathbf{Y}}^*$ depends on all of the particles in the neighborhood. Similarly, we can also derive the elastic force density from $E^{\mathtt{iso}}$ as

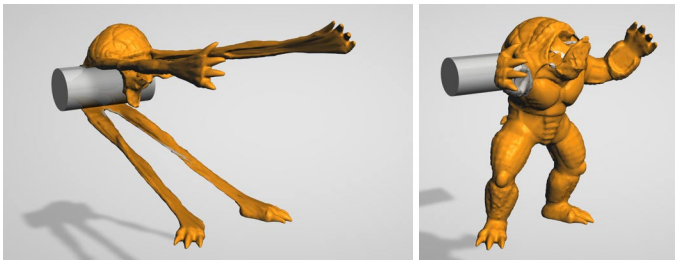
$$\underline{\mathbf{T}}^{\mathtt{iso}} = \frac{\lambda \underline{w}}{|\underline{\mathbf{X}}|^2} \left(\underline{\mathbf{Y}} - |\underline{\mathbf{X}}| \left(\mathtt{dir}\, \underline{\mathbf{Y}} \right) \right). \tag{11}$$

The difference between Equation 11 and 10 is that the term $|\underline{\mathbf{X}}| \left(\mathtt{dir}\, \underline{\mathbf{Y}} \right)$ in Equation 11 only depends on $\mathbf{y}_i$ and $\mathbf{y}_j$,

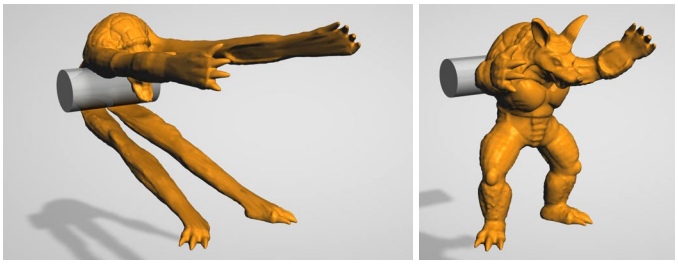


(a) Flattened shape (b) Recovered shape

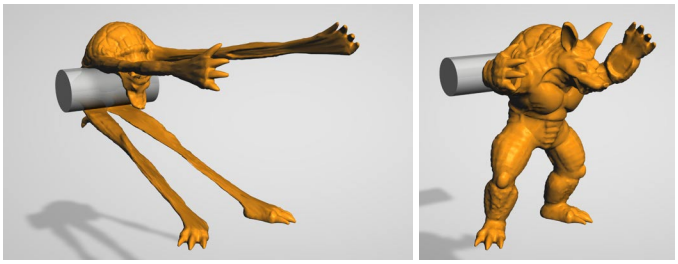
Fig. 3. The bunny example. Our method can recover the bunny from its degenerated shape as shown in (a), with a simple treatment to prevent shape inversion



(a) The result of the mass-spring system



(b) The result of shape matching



(c) The result of our method

Fig. 4. The armadillo example. Unlike the mass-spring system in (a) or shape matching in (b), our method can properly handle both inverted regions and the Poisson effect, as shown in (c).

while $|\underline{\mathbf{X}}|$ $(\mathtt{dir}\,\underline{\mathbf{Y}}^*)$ depends on all the neighbors. Therefore, the direction of $\underline{\mathbf{T}}^{\mathtt{iso}}$ is along the line of $\mathbf{y}_i$ and $\mathbf{y}_j$, which has a similar effect as the normal stress in continuum mechanics. Equivalently, $\underline{\mathbf{T}}^{\mathtt{dev}}$ has a similar effect as the shear stress.

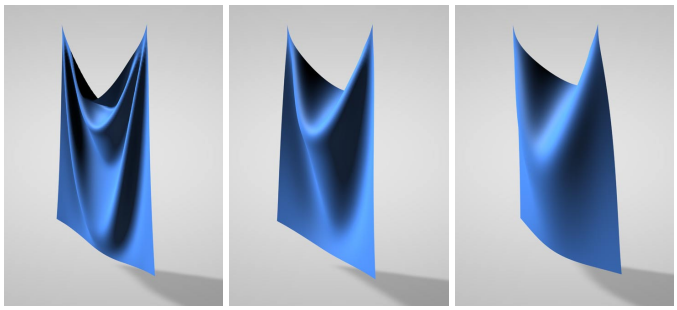
Time integration. Based on implicit Euler integration, we can discretize the update function of particle i into a nonlinear system:

$$\rho_i(\mathbf{y}_i^{t+1} - \mathbf{y}_i^*) = h^2 \sum_j V_j \left(\underline{\mathbf{T}}_i^{t+1} \langle \xi \rangle - \underline{\mathbf{T}}_j^{t+1} \langle -\xi \rangle\right), \quad (12)$$

in which $\mathbf{y}_i^*$ is the expected particle position under momentum and other operations, h is the time step, and V_j is the volume for a particle j in the neighborhood $\mathcal{H}_i$. Similar to projective dynamics [13], we solve this nonlinear system by iterating a local step and a global step. In the local step, we use the current result to evaluate $(\mathtt{dir}\,\underline{\mathbf{Y}}^*)\langle\xi\rangle$ for every particle and its neighbor. In the global step, we then treat these direction terms as constants and solve the remaining linear system to get a new result for the next iteration. Unlike many existing techniques, our method does not need polar decomposition to extract the rotation matrix in the iterations, so its local step is very inexpensive. If the deformation is elastic only and the neighborhood does not change over time, we can further speed up the global step by pre-factorizing the constant system matrix, as did in [13].

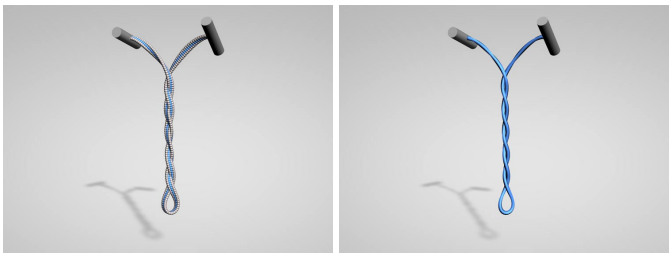
Comparison to a MLS-based method. The difference between using Equation 8 and using the force formulas based on MLS [20] is that Equation 8 avoids the need to explicitly extract a rotation matrix for each particle. Although applying the singular value decomposition (SVD) for a small size matrix to extract the rotation matrix is not a bottleneck when done properly, it is preferable to avoid implementing SVD since it is not necessary for achieving desired behaviors [27]. For our purely meshless method, the advantage of avoiding SVD is twofold: our method can have a better stability for simulation with large timesteps. Besides, it can increase the diversity of elastoplastic materials our method can model. By choosing the appropriate parameters, Figure 1 shows that our method is able to create a similar result as the MLS-based method [20] with a small timestep($\Delta t = 0.001s$). However, if we choose large timesteps, the MLS-based method fails to create stable results, possibly because the singular value decomposition is sensitive to the particle distribution. Besides, rather than mapping a sphere into an ellipsoid, our force formulas map a sphere into a more complex and possibly discontinuous surface. Therefore, it is easy for us to introduce other conditions to model more complex elastoplastic materials, such as the modeling of dry/wet granular materials demonstrated in Section 7. As far as we know, it is not clear how Jones and colleagues's work [20] can be extended to simulate granular materials or lower-dimensional objects, such as cloth.

Motivated by the finite element method [35], [36], we prevent shape inversion by permuting the first two rows of the deformation gradient tensor $\bar{\mathbf{F}}$ if its determinant is less than 0. Figure 3 shows how this simple treatment can help recover the bunny from its degenerated shape. Figure 4 also compares the result of our method with the results of the mass-spring system and shape matching [33]. It shows that the mass-spring system cannot fix inverted regions, as expected. Shape matching does not have this issue, but it fails to produce the Poisson effect as shown in Figure 4b. In contrast, our method generates a more physically plausible result as Figure 4c shows. Besides, we can also conveniently simulate lower-dimensional objects, such as a cloth with different bending stiffness by adjusting the value of μ, as Figure 5 shows. We note that twisting deformation cannot be simulated if a rod is represented by a single list of particles, due to the singularity in $\bar{\mathbf{F}}$. Our solution is to expand a rod by adding ghost particles, as shown in Figure 6a. Ghost particles are treated in the same way as original particles, except that they do not participate in collision handling. Unfortunately, we cannot further separate bending deformation from twisting deformation in rod simulation at this



(a) μ=λ/10 (b) μ=λ/2 (c) μ=λ

Fig. 5. The cloth example. Our method can simulate different bending stiffness behaviors, by adjusting the value of μ.



(a) Ghost particles rendered (b) Ghost particles hidden

Fig. 6. The rod example. Our method needs ghost particles to properly simulate twisting deformation of a rod, where ghost particles are rendered in grey while real particles are rendered in blue, as shown in (a)

time. So modifying μ will cause both bending and twisting resistance to change.

4.2 Analysis and Evaluation

In this subsection, we will analyze the convergence and the performance of our projective peridynamics method. We will also discuss implementation details related to them.

Convergence condition. If we use only spring forces in simulation, it is straightforward to see that $|\underline{\mathbf{X}}|\,(\mathtt{dir}\,\underline{\mathbf{Y}})$ is the geometric projection of $\underline{\mathbf{Y}}$ into a deformation-free space. In that case, our method is identical to projective dynamics and it is guaranteed to converge weakly as shown in [12], [13]. This conclusion is no longer valid, when the direction term contains $|\underline{\mathbf{X}}|\,(\mathtt{dir}\,\underline{\mathbf{Y}}^*)$. Fortunately, if we consider the nonlinear system in Equation 12 as a nonlinear optimization problem for finding optimal $\mathbf{y}_i$, we can treat our method fundamentally as a preconditioned gradient descent method with the system matrix acting as the preconditioner. Since the matrix is strictly diagonally dominant, it is positive definite and we can guarantee the convergence of the method by using a sufficiently small step length $\alpha^{(k)}$:

$$\mathbf{y}_i^{(k+1)} = \mathbf{y}_i^{(k)} + \alpha^{(k)}\left(\hat{\mathbf{y}}_i^{(k)} - \mathbf{y}_i^{(k)}\right), \tag{13}$$

in which $\mathbf{y}_i^{(k)}$ is the particle position result in the k-th iteration and $\hat{\mathbf{y}}_i^{(k)}$ is the solution to the linear system after the k-th iteration. In our system, we use the standard backtracking line search method to find $\alpha^{(k)}$, by evaluating the Armijo-Goldstein condition.

Performance evaluation. Given the proposed elastic simulation technique, our next question is: *how should we get the linear system solved in the global step*? Based on previous research on projective dynamics, here we test three different solvers on a single CPU core, including the direct solve using Cholesky factorization [13], one Jacobi iteration, and one Jacobi iteration with Chebyshev acceleration [34]. For comparison purposes, we also evaluate the performance of solving the global system in a position-based way. To do that, we simply calculate particle positions suggested by each neighborhood according to Equation 10, and then use their averages to update particle positions in one global step. Similar to [34], we define the error as the residual magnitude of the nonlinear system in Equation 12. Figure 7 shows that the direct solve performs the best on the CPU. This is not surprising, since the particle neighborhood does not change in elastic body simulation and the matrix factorization cost can be removed from the running cost. However, when the neighborhood changes as in plastic or fluid simulation, or when the simulator runs on the GPU, the Jacobi+Chebyshev method should outperform the others. So we choose it as the default solver for the global step in our elastic simulation component.

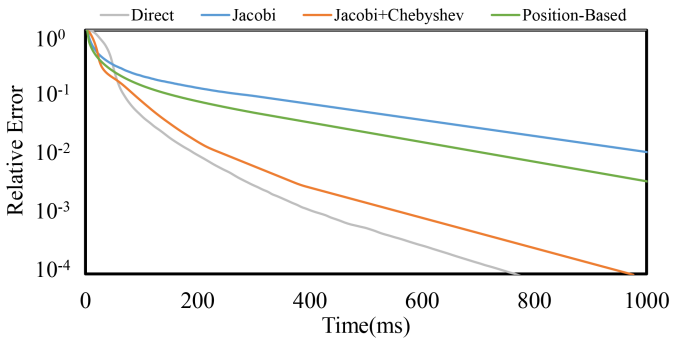


Fig. 7. The convergence rates of the simulator using different methods for the global step, including direct solve, Jacobi solve, Jacobi+Chebyshev solve, and position-based solve. The direct method runs the fastest on the CPU, while the Jacobi+Chebyshev method is expected to run the fastest on the GPU.

5 Plastic Simulation

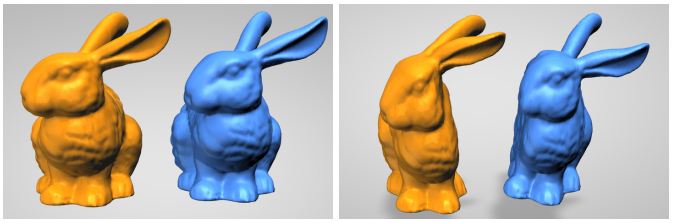
To model plastic deformation and its hysteresis effect, many existing techniques [21], [37] choose to extract a plastic component from the deformation gradient and then accumulate it over time. This approach fails to work, when the deformation gradient is singular and the plastic component is not unique [20]. It is not so compatible with the structure of our method either, which relies heavily on the particle positions, rather than the deformation gradient. So our idea here is to directly modify the reference position state $\underline{\mathbf{X}}$ over time, which accounts for the permanent shape change under plastic deformation. Let $\underline{\mathbf{D}}_\mathrm{e}$ and $\underline{\mathbf{D}}_\mathrm{p}$ be the elastic and plastic displacement states decomposed from the co-rotational displacement state: $\underline{\mathbf{D}} = \underline{\mathbf{Y}} - \bar{\mathbf{R}}\underline{\mathbf{X}} = \underline{\mathbf{D}}_\mathrm{e} + \underline{\mathbf{D}}_\mathrm{p}$, we can update the reference position vector state as: $\underline{\mathbf{X}} \leftarrow \bar{\mathbf{R}}\underline{\mathbf{X}} + \underline{\mathbf{D}}_\mathrm{p}$, in which $\bar{\mathbf{R}}$ is the rotation matrix extracted from $\bar{\mathbf{F}}$.

To calculate the plastic displacement, we must decide the yield criterion. Here we choose the Drucker-Prager yield criterion:

$$\sqrt{J_2} \le A + BI_1, \tag{14}$$

(a) Dropping under the gravity (b) Dripping started (c) Dripping ended

Fig. 8. The honey example. By redistributing the particles, our method can be extended to model homogeneous viscoelastic fluids, such as honey.



(a) Before collision (b) After collision

Fig. 9. The colliding bunny example. This example demonstrates the permanent plastic deformation of two bunnies after they hit together.

where I_1 is the first invariant of the Cauchy stress, J_2 is the second invariant of the deviatoric part of the Cauchy stress, and A and B are constants. Under the Mohr-Coulomb theory, we express A and B as functions of the internal friction angle ϕ and the cohesion c:

$$A = \frac{6c\cos\phi}{\sqrt{3}\,(3+\sin\phi)}, \quad B = \frac{2\sin\phi}{\sqrt{3}\,(3+\sin\phi)}. \tag{15}$$

The question is how this yield criterion can be applied to determine the plastic displacement between any two particles. Let $\underline{e} = |\underline{\mathbf{Y}}| \,/\, |\underline{\mathbf{X}}| - 1$ be a scalar state describing the Cauchy strain of the spring connecting the particles. We decompose it into an isotropic part and a deviatoric part:

$$e^{\mathtt{iso}} = \int_{\mathcal{H}} \underline{w}\,\langle\xi\rangle\,\underline{e}\,\langle\xi\rangle\,d\xi, \quad \underline{e}^{\mathtt{dev}} = \underline{e} - e^{\mathtt{iso}}, \tag{16}$$

and estimate I_1 and J_2 as:

$$I_1 = -\lambda e^{\mathtt{iso}}, \quad J_2 = \mu \int_{\mathcal{H}} \underline{e}^{\mathtt{dev}}\,\langle\xi\rangle\,\underline{e}^{\mathtt{dev}}\,\langle\xi\rangle\,d\xi. \tag{17}$$

Given I_1 and J_2, we evaluate whether the yield criterion is violated. If so, we compute a set of two parameters:

$$\{\alpha,\beta\} = \begin{cases} \left\{0, \frac{\sqrt{J_2}-A-BI_1}{\sqrt{J_2}}\right\}, & \text{if } A+BI_1 > 0, \\ \left\{\frac{A+BI_1}{BI_1}, 1\right\}, & \text{otherwise,} \end{cases} \tag{18}$$

and use them to calculate the plastic displacement state as: $\underline{\mathbf{D}}_{\mathtt{p}} = \alpha\mathbf{D}^{\mathtt{iso}} + \beta\underline{\mathbf{D}}^{\mathtt{dev}}$. Here $\mathbf{D}^{\mathtt{iso}}$ and $\underline{\mathbf{D}}^{\mathtt{dev}}$ are the isotropic and deviatoric parts of $\underline{\mathbf{D}}$, respectively:

$$\mathbf{D}^{\mathtt{iso}} = \int_{\mathcal{H}} \underline{w}\,\langle\xi\rangle\,\underline{\mathbf{D}}\,\langle\xi\rangle\,d\xi, \quad \underline{\mathbf{D}}^{\mathtt{dev}} = \underline{\mathbf{D}} - \mathbf{D}^{\mathtt{iso}}. \tag{19}$$

It is straightforward to see from Equation 18 that $\alpha, \beta \in [0,1]$. So $\underline{\mathbf{D}}_{\mathtt{p}}$ cannot exceed $\underline{\mathbf{D}}$, as in the real world. When $A + BI_1 = \sqrt{J_2}$, the criterion is just violated and we should have $\alpha = \beta = 0$. If $A + BI_1 > 0$, we assume that plastic deformation exists only in the deviatoric part, so $\alpha = 0$. The transition happens when $A + BI_1 \leq 0$ and the criterion is always violated. Since we cannot have $\beta > 1$, we assume that plastic deformation occurs in the isotropic part as well. These thoughts guide us to form the calculation of α and β as shown in Equation 18. Figure 9 demonstrates the effectiveness of our plastic model, where we have setup this experiment with $\phi = 0$ and $c = 0.005$.

New Particle Neighbors. For large plastic deformations, the neighborhood of a particle will change. Theoretically speaking, we should perform the new neighborhood search on particle's new rest positions. Since new rest positions are never known, we choose to perform the neighborhood search on deformed positions at the beginning of each time step instead. Now we must update the reference position state $\underline{\mathbf{X}}$, if particle $\mathbf{x}_j$ is newly added into the neighborhood of particle $\mathbf{x}_i$.

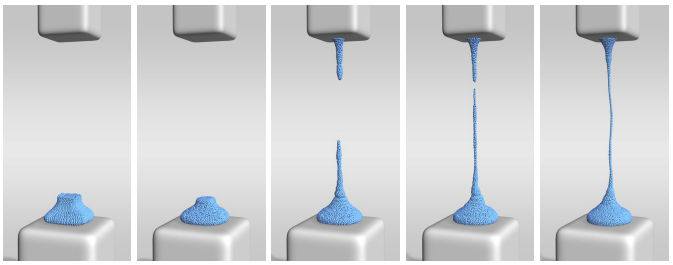
Given particle i's old neighbors, we can use them to calculate the deformation gradient $\bar{\mathbf{F}}$, and then estimate the reference position of particle j by inverse transformation:

$$\underline{\mathbf{X}}\,\langle\xi\rangle \approx \bar{\mathbf{F}}^{-1}(\mathbf{y}_j - \mathbf{y}_i) = (\underline{\mathbf{X}} * \underline{\mathbf{Y}})(\underline{\mathbf{Y}} * \underline{\mathbf{Y}})^{-1}(\mathbf{y}_j - \mathbf{y}_i). \tag{20}$$

This method, however, can cause newly added particles to be projected onto a plane or a line, if $\bar{\mathbf{F}}$ is nearly singular. To address this issue, we apply singular value decomposition on $\bar{\mathbf{F}}$ and clamp its singular values to $[1+e^{\mathtt{iso}} - \|\underline{e}^{\mathtt{dev}}\|, 1+e^{\mathtt{iso}} + \|\underline{e}^{\mathtt{dev}}\|]$, which specifies the range of acceptable stretching ratios. Here $\|\cdot\|$ is a magnitude reduction operator applied on a scalar state [10]. After that, we use the modified $\bar{\mathbf{F}}$ to perform the computation in Equation 20 instead. The evaluation of large plastic deformation can be found in Section 7.

6 Viscoelastic Fluid Simulation

Homogeneous viscoelastic fluids, such as honey shown in Figure 8, can be simulated by our method as well. To do so, we define Lamé parameters as constants and we set the



(a) $\delta = 0$ (b) $\delta = 0.25$ (c) $\delta = 0.5$ (d) $\delta = 0.75$ (e) $\delta = 1$

Fig. 10. Dripping patterns. When using a larger δ, our particle redistribution approach can more effectively avoid tensile instability and preserve thin features as shown in (e).

internal friction angle ϕ to zero. We enforce incompressibility by applying position-based density constraints on particles as did in [30] and implement an implicit version of the viscous force [38]. In addition, we need to redistribute particles to avoid tensile instability, which is a common problem in particle-based fluid simulation [39]. One possible solution is to re-sample materials with particles as [40]. However, re-sampling without losing the total mass is never an easy task to do. In smoothed particle hydrodynamics (SPH), researchers [41], [42] pointed out that particle redistribution can be considered as minimizing $\|\nabla\rho\|^2$. So we propose to formulate it as a positional constraint as well and enforce it in a position-based way. Here we define the particle distribution constraint on the i-th particle as:

$$C_i = \|\nabla_i \rho\|^2 = 0, \tag{21}$$

in which $\nabla_i \rho = m \sum_j \nabla_i W_{ij}$, m is the particle mass, and W is the Spiky kernel function under the SPH framework [38]. According to [30], we calculate the moving distance in one Newton iteration as:

$$\tau_i = \frac{C_i}{\sum_l \|\nabla_l C_i\|^2 + \varepsilon}, \tag{22}$$

where ε is a relaxation variable and $\nabla_l C_i$ is:

$$\nabla_l C_i = \begin{cases} \nabla_i \nabla_i \rho \cdot \nabla_i \rho, & \text{if } l = i, \\ -\nabla_j \nabla_i \rho \cdot \nabla_i \rho, & \text{if } l = j, \end{cases} \tag{23}$$

Given τ_i, we can then update the position of particle i by:

$$\mathbf{y}_i \leftarrow \mathbf{y}_i + \delta \sum_j \frac{\tau_i + \tau_j}{2} \nabla_i W_{ij}, \tag{24}$$

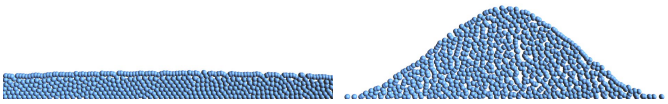
in which $\delta \in [0, 1]$ controls the strength of particle redistribution. We typically take $3 \sim 10$ iterations with Equation 24 to get a good particle distribution according to the average error. Figure 10 demonstrates the effect of different δ on the dripping pattern of a viscoelastic fluid. It shows that by using a larger δ, particle redistribution can preserve thin features better. To model the sticking behavior of the fluid to the top ceiling, we seed ghost particles near the boundary of the top ceiling and treat them similarly as those fluid particles, except that the positions of ghost particles keep fixed during the whole simulation.

7 Simulation of Granular Materials

Unlike homogeneous materials, granular materials, such as sand and snow, can exhibit complex deformation behaviors under different moisture, temperature, or granularity conditions [43]. In this section, we will discuss common characteristics of granular materials and how to achieve their effects by our simulation method.

(a) $\vartheta = 0.01$ (b) $\vartheta = 0.1$

Fig. 11. The wet sand example. By adjusting the water saturation coefficient to control the magnitude of the cohesion stiffness, our simulator can animate wet sands with different cohesion effects.



(a) Without our static friction model (b) With our static friction model

Fig. 12. The 2D dry sand example. Applying frictional stiffness alone cannot help the simulator get dry sands piled up properly as shown in (a). Fortunately, our static friction model solves this problem as shown in (b). Both are solved by 5 Jacobi iterations with Chebyshev acceleration.

7.1 Cohesion and Friction Stiffness

When we simulate granular materials, we no longer use constant Lamé parameters. Instead, we split them into the sum of a friction stiffness $k_\mathtt{f}$ and a cohesion stiffness $k_\mathtt{c}$, e.g., $\lambda = k_\mathtt{f} + k_\mathtt{c}$. Similar to [21], we define the friction stiffness as a function of the particle density ρ:

$$k_\mathtt{f}(\rho) = \begin{cases} K_\mathtt{f} e^{H_\mathtt{f}(\rho/\rho_0 - 1)}, & \text{if } \rho \in [\rho_0, +\infty), \\ 0, & \text{otherwise}, \end{cases} \tag{25}$$

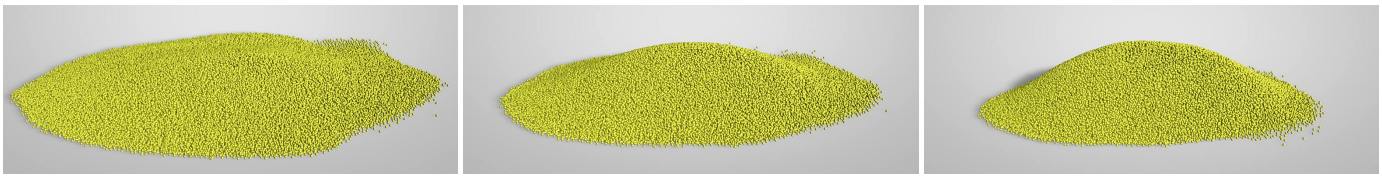
in which ρ_0 is the reference density, $K_\mathtt{f}$ is a reference friction, and $H_\mathtt{f}$ is the hardening coefficient. Intuitively, Equation 25 means if the granular density is lower than the reference density, the friction force should disappear. This treatment is based on the fact that friction forces can occur only when particles are in close contact.

Cohesion forces in wet granular materials are caused by liquid capillary bridges among adjacent grains [43]. Since it is difficult to integrate this theoretical concept into the development of our simulator, we propose to build an empirical relationship between the cohesion stiffness $k_\mathtt{c}$ and the particle density ρ. Inspired by the exponential capillary force model [44], we define the cohesion stiffness $k_\mathtt{c}$ as:

$$k_\mathtt{c}(\rho) = \begin{cases} K_\mathtt{c}, & \text{if } \rho \in [\rho_0, +\infty), \\ K_\mathtt{c} e^{H_\mathtt{c}(1 - \rho/\rho_0)}, & \text{if } \rho \in [\rho_1, \rho_0), \\ 0, & \text{otherwise}. \end{cases} \tag{26}$$

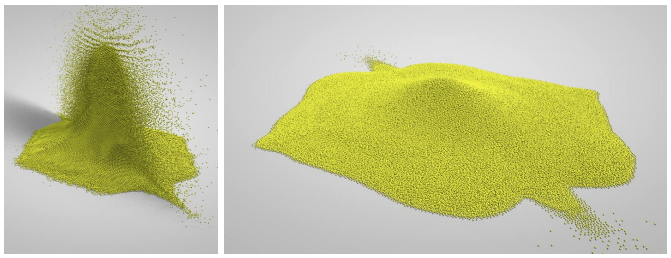
where $K_\mathtt{c}$ is the reference cohesion, $H_\mathtt{c}$ is an exponential falloff coefficient, and ρ_1 is the debonding density. In our simulation, we define $H_\mathtt{c}$ and ρ_1 as:

$$H_\mathtt{c} = H_0 \vartheta^{-1/2}, \quad \rho_1 = (1 - b\vartheta)\rho_0, \tag{27}$$



(a) $\phi = 15$ (b) $\phi = 30$ (c) $\phi = 45$

Fig. 13. The piled dry sand example. Our simulator uses the internal friction angle to control how tall dry sands can get piled up.



(a) Colliding sands (b) Sand at rest

Fig. 14. The colliding sand example. Our static friction model does not cause sticky artifacts, as this example shows.

in which ϑ is the water saturation, H_0 is the reference coefficient value, and b is a scaling variable typically set to 1. Figure 11 illustrates cohesion effects in the simulation of wet sands, when using different water saturation ϑ values.

7.2 Static Friction

The iterative nature of our projective simulator allows us to recalculate the stiffness $k_{\mathbf{f}}$ in every iteration. However, if not using a large number of iterations, it is difficult to achieve static friction among dry sands as did in [45], which solved a linear complementarity problem under unilateral constraints. In fact, even if we use 100 iterations, dry sands still cannot stack on each other as our experiment shows. To address this issue, we propose to simply check if the friction force is strong enough to push particle i back:

$$\left(\mathbf{y}_i^{t+1} - \mathbf{y}_i^*\right) \cdot \left(\mathbf{y}_i^t - \mathbf{y}_i^*\right) \geq \eta \left\|\mathbf{y}_i^* - \mathbf{y}_i^t\right\|^2, \qquad (28)$$

where η is a static friction coefficient typically set to 0.8 in our experiment. If the condition in Equation 28 is satisfied, we assume static friction happens and we set $\mathbf{y}_i^{t+1} = \mathbf{y}_i^t$. By using this method, our simulator can effectively address the sliding issue under static friction as shown in Figure 12b. We note that although this method tries to eliminate the relative motion between two particles, it does not cause particles to be mutually locked as sticky artifacts, as the colliding example shown in Figure 14.

8 Results and Discussions

We implemented our system and test our examples on an Intel Core i7-2600 3.4GHz processor. We use the OpenMP library to parallelize the system and the Eigen library for direct linear solve. For rendering viscoelastic fluids, we use the particle skinning method [46] to reconstruct the fluid surface mesh. Table 1 summarizes the statistics and the timings of our examples.

Fig. 15. The falling sand example. This high-resolution example demonstrates the ability of our simulation system in generating interesting cracking patterns during the sand falling process.

Implementation. During our implementation, we use the Z-index sort method [47] to renew particle neighborhoods, every time particle positions get changed. In our experiment, a neighborhood contains 30 neighbors. Depending on the type of the material, the simulator selects and runs any combination of three simulation components: elasticity, plasticity, and particle redistribution. The convergence criteria in our simulation uses both the residual error and the maximum number of iterations. The system offers different options for solving the global linear system, as described in Subsection 4.2.

We implement both slip and no-slip solid boundary conditions. To model slip boundary conditions, we simply project particles out of the solids. To model no-slip boundary conditions, we sample ghost particles around solid boundaries as in [48] and simulate them together with other particles, except that their positions are never updated. We typically impose a non-slip boundary condition on the ground floor, and slip boundary conditions on the rest of the solids.

To prevent particles from getting too close to each other, we apply a repulsive distance constraint as in [31]. Nearby particles can be easily detected from the particle neighborhood renewed at the beginning of each time step. We note that the renewed particle neighborhood should not cause the reference position vector state to be updated, if the deformation is elastic only.

An important strength of our method is that it is highly

Name	#Particles	h (s)	$t_{\mathtt{n}}$ (ms)	t_{ρ} (ms)	$t_{\mathtt{d}}$ (ms)	$t_{\mathtt{p}}$ (ms)	$t_{\mathtt{e}}$ (ms)	$t_{\mathtt{sum}}$ (ms)
Bunny (Figure 3)	25K	1/30	-	-	-	-	36.2	41.1
Armadillo (Figure 4)	29K	1/30	-	-	-	-	44.2	49.5
Honey (Figure 8)	72K	1/250	34.0	26.9	32.4	4.9	24.0	125.4
Two bunnies (Figure 9)	54K	1/250	-	22.1	-	4.1	23.7	53.6
Dry sand (Figure 14)	1,177K	1/500	481.0	340.2	-	81.7	124.5	1081.7
Wet sand (Figure 15)	1,325K	1/500	491.0	395.5	-	42.5	390.5	1492.3
Dambreak (Figure 16)	312K	1/500	127.1	122.5	-	19.3	118.2	412.0
Dripping (Figure 17)	564K	1/250	221.9	211.0	253.1	38.2	233.0	982.2
Coupling (Figure 18)	49K	1/1000	19.2	21.5	17.6	3.1	20.2	85.7
Dress (Figure 19)	25K	1/250	9.8	9.4	11.3	1.9	10.5	43.8

TABLE 1
Statistics and timings of our examples, The timings include the neighborhood update cost $t_{\mathtt{n}}$, the particle redistribution cost $t_{\mathtt{d}}$, the incompressibly enforcement cost t_{ρ}, the elastic solver cost $t_{\mathtt{e}}$, the plastic solver cost $t_{\mathtt{p}}$, and the total cost per time step $t_{\mathtt{sum}}$. Note that '-' means the corresponding part is not solved.

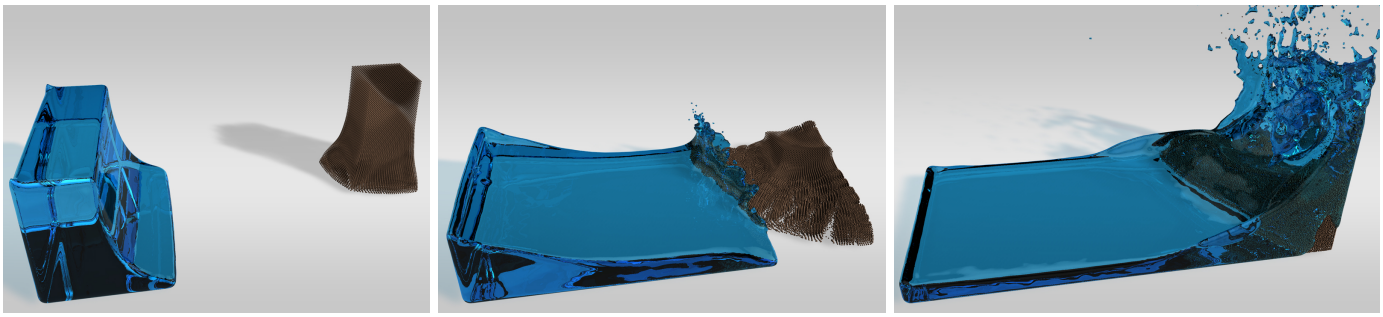
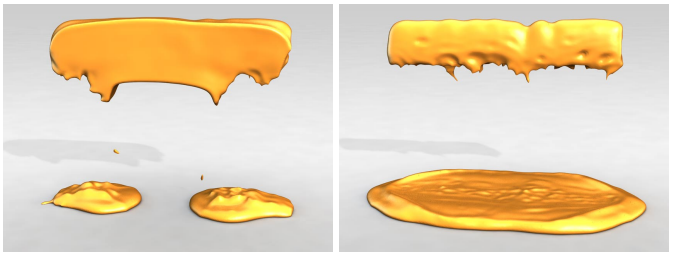


Fig. 16. The dambreak example. This example shows the interaction between a viscous liquid and wet sands.



(a) $\delta = 0$ (b) $\delta = 0.5$

Fig. 17. Another dripping example. This example demonstrates how a bulk of viscoelastic material drips differently under the gravity with a different value of δ.

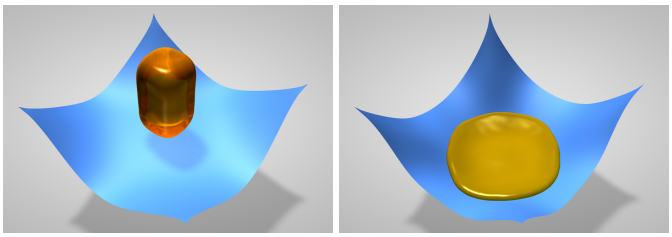


Fig. 18. The liquid-cloth coupling example. Our method can uniformly model the interaction between a cloth and a viscoelastic fluid.

compatible with GPU acceleration. Our fluid simulation component, built upon the position-based framework, can be easily implemented on the GPU. Although the elastic simulation component needs to solve a linear system, it can be implemented using the Jacobi+Chebyshev method on the GPU as well, as suggested by Wang [34]. We expect to see a significant performance gain, once we transfer our simulator to the GPU in the near future.

Examples. Our examples demonstrate that the simulator can effectively simulate a variety of elastoplastic materials. Figure 13 shows the use of the internal friction angle to control how tall dry sands can be piled up. Figure 15 reveals the ability of our simulator in handling a large number of particles and producing interesting cracking patterns as sands fall. Figure 17 shows that the simulator can successfully present different behaviors of viscoelastic fluids by adjusting the value of δ.

As a unified system, our meshless simulator can easily model the interaction of different materials. For example, Figure 16 demonstrates the interaction between a viscous liquid and wet sands. In this example, we only solve elasticity and plasticity for wet sands while solve incompressibility for both materials. Figure 18 models the process of an viscoelastic fluid falling onto a cloth. To prevent fluid particles from entering the other side of the cloth, we take an extra step to correct the positions of fluid particles by checking the relative position of the fluid particles and the cloth. Besides, our meshless simulator can easily handle the transition among different material properties as well. The bottom part of a dress shown in Figure 19a can be simulated as a viscoelastic fluid first, and then as an elastic thin shell again after solidification.

Limitations. The projective nature of our simulator requires it to define the elastic constitutive model in a specific form. This limits its ability in handling generic hyperelastic constitutive models under the classical elasticity theory, as in [10]. This also means the connection between existing viscoelastic models and the viscoelastic fluids animated by our simulator is not straightforward. Currently, we use the particle redistribution process to achieve the surface tension effect, which cannot be controlled separately. A potential solution to this problem is to add ghost air particles outside

(a) Before simulation (b) After simulation

Fig. 19. The dress example. Our simulator can simultaneously animate the top part of the dress as an elastic shell and the bottom part as a viscoelastic fluid .

of the free surface, for guiding particle redistribution only. Our current system does not allow bending stiffness to be controlled separately from twisting stiffness of a rod as well. Although the system can robustly handle very large time steps, such as $h = 1/30$s, it is usually not preferred to do so, due to the resulting artificial damping, erroneous friction, or tensile instability issues. Finally, current state-of-art surface reconstruction methods have difficulties in reconstructing thin features, such as filaments. To reconstruct these features, we applied a post-processing to shrink the mesh surface after reconstruction, which might cause some volume-loss artifacts in certain examples.

9 Conclusions and Future Work

In this paper, we demonstrate that the idea of peridynamics and projective dynamics can be combined together to form a novel simulation technique for unified particle simulation of many elastoplastic materials, including granular material behaviors. This technique is efficient and robust against large deformations and time steps. But just like the original projective dynamics technique, it is restricted by the types of the materials it can handle.

The first thing listed in our future plan is to develop and test our simulator on the GPU. We then would like to explore the generalization of our constitutive model, to make it suitable for producing more hyperelastic and viscoelastic material behaviors. Currently, our elastic simulation component is designed in a projective dynamics way, while our fluid simulation component is designed in a position-based way. So it will be interesting to know whether the algorithm structure can be formulated in a more uniform style.

10 Acknowledgement

We would like to thank anonymous reviewers for their constructive comments, Miaojun Yao, Xiaofeng Wu and Jing Zhao for valuable suggestions that helped to improve the manuscript. The project was supported by the National Key R&D Program of China (No.2017YFB1002700), the National Key Technology R&D Program (No.2015BAK01B06), the National Natural Science Foundation of China (No.6140051239, 61572479, 61672502, 61632003), Macao FDCT Fund (068/2015/A2, 136/2014/A3) and UMRG (MYRG2014-00139-FST). The second author would also like to thank Adobe and NVIDIA research gifts and the NSF grant IIS-1524992.

References

[1] M. Müller, R. Keiser, A. Nealen, M. Pauly, M. Gross, and M. Alexa, "Point based animation of elastic, plastic and melting objects," in *Proceedings of SCA*, 2004, pp. 141–151.

[2] S. Martin, P. Kaufmann, M. Botsch, E. Grinspun, and M. Gross, "Unified simulation of elastic rods, shells, and solids," *ACM Trans. Graph. (SIGGRAPH)*, vol. 29, no. 4, pp. 39:1–39:10, Jul. 2010.

[3] B. Solenthaler, J. Schläfli, and R. Pajarola, "A unified particle model for fluid–solid interactions," *Computer Animation and Virtual Worlds*, vol. 18, no. 1, pp. 69–82, 2007.

[4] M. Pauly, R. Keiser, B. Adams, P. Dutré, M. Gross, and L. J. Guibas, "Meshless animation of fracturing solids," *ACM Trans. Graph. (SIGGRAPH)*, vol. 24, no. 3, pp. 957–964, Jul. 2005.

[5] N. Liu, X. He, S. Li, and G. Wang, "Meshless simulation of brittle fracture," *Computer Animation and Virtual Worlds*, vol. 22, no. 2-3, pp. 115–124, 2011.

[6] M. Müller, B. Heidelberger, M. Hennix, and J. Ratcliff, "Position based dynamics," *J. Vis. Comun. Image Represent.*, vol. 18, no. 2, pp. 109–118, Apr. 2007.

[7] M. Müller, N. Chentanez, T. Kim, and M. Macklin, "Strain based dynamics," in *Proceedings of SCA*, 2014, pp. 21–23.

[8] J. A. Levine, A. W. Bargteil, C. Corsi, J. Tessendorf, and R. Geist, "A peridynamic perspective on spring-mass fracture," in *Proceedings of SCA*, 2014, pp. 47–55.

[9] S. A. Silling, "Reformulation of elasticity theory for discontinuities and long-range forces," *Journal of the Mechanics and Physics of Solids*, vol. 48, no. 1, pp. 175–209, 2000.

[10] S. A. Silling, M. Epton, O. Weckner, J. Xu, and E. Askari, "Peridynamic states and constitutive modeling," *Journal of Elasticity*, vol. 88, no. 2, pp. 151–184, 2007.

[11] M. Tupek, J. Rimoli, and R. Radovitzky, "An approach for incorporating classical continuum damage models in state-based peridynamics," *Computer methods in applied mechanics and engineering*, vol. 263, pp. 20–26, 2013.

[12] T. Liu, A. W. Bargteil, J. F. O'Brien, and L. Kavan, "Fast simulation of mass-spring systems," *ACM Trans. Graph. (SIGGRAPH Asia)*, vol. 32, no. 6, pp. 214:1–214:7, Nov. 2013.

[13] S. Bouaziz, S. Martin, T. Liu, L. Kavan, and M. Pauly, "Projective dynamics: Fusing constraint projections for fast simulation," *ACM Trans. Graph. (SIGGRAPH)*, vol. 33, no. 4, pp. 154:1–154:11, 2014.

[14] M. Ihmsen, J. Orthmann, B. Solenthaler, A. Kolb, and M. Teschner, "SPH fluids in computer graphics," in *Eurographics 2014 - State of the Art Reports*, 2014.

[15] M. Desbrun and M.-P. Gascuel, *Smoothed particles: A new paradigm for animating highly deformable bodies*. Springer, 1996.

[16] A. Paiva, F. Petronetto, T. Lewiner, and G. Tavares, "Particle-based non-Newtonian fluid animation for melting objects," in *Proceedings of SIBGRAPI*, 2006, pp. 78–85.

[17] M. Becker, M. Ihmsen, and M. Teschner, "Corotated SPH for deformable solids." in *Eurographics Workshop on Natural Phenomena*. Citeseer, 2009, pp. 27–34.

[18] D. Gerszewski, H. Bhattacharya, and A. W. Bargteil, "A point-based method for animating elastoplastic solids," in *Proceedings of SCA*, ser. SCA '09. New York, NY, USA: ACM, 2009, pp. 133–138.

[19] I. Alduán and M. A. Otaduy, "SPH granular flow with friction and cohesion," in *Proceedings of SCA*, 2011, pp. 25–32.

[20] B. Jones, S. Ward, A. Jallepalli, J. Perenia, and A. W. Bargteil, "Deformation embedding for point-based elastoplastic simulation," *ACM Trans. Graph.*, vol. 33, no. 2, p. 21, 2014.

[21] A. Stomakhin, C. Schroeder, L. Chai, J. Teran, and A. Selle, "A material point method for snow simulation," *ACM Trans. Graph. (SIGGRAPH)*, vol. 32, no. 4, p. 102, 2013.

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

[23] C. Jiang, C. Schroeder, A. Selle, J. Teran, and A. Stomakhin, "The affine particle-in-cell method," *ACM Trans. Graph.*, vol. 34, no. 4, pp. 51:1–51:10, Jul. 2015.

[24] A. Stomakhin, C. Schroeder, C. Jiang, L. Chai, J. Teran, and A. Selle, "Augmented MPM for phase-change and varied materials," *ACM Trans. Graph. (SIGGRAPH)*, vol. 33, no. 4, p. 138, 2014.

[25] 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.

[26] 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.

[27] 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 SCA*, 2015, pp. 157–163.

[28] F. Zhu, J. Zhao, S. Li, Y. Tang, and G. Wang, "Dynamically enriched mpm for invertible elasticity," 2016.

[29] C. Jiang, T. Gast, and J. Teran, "Anisotropic elastoplasticity for cloth, knit and hair frictional contact," *ACM Trans. Graph. (SIGGRAPH)*, vol. 36, no. 4, 2017.

[30] M. Macklin and M. Müller, "Position based fluids," *ACM Trans. Graph. (SIGGRAPH)*, vol. 32, no. 4, p. 104, 2013.

[31] M. Macklin, M. Müller, N. Chentanez, and T.-Y. Kim, "Unified particle physics for real-time applications," *ACM Trans. Graph. (SIGGRAPH)*, vol. 33, no. 4, p. 153, 2014.

[32] M. Tournier, M. Nesme, B. Gilles, and F. Faure, "Stable constrained dynamics," *ACM Trans. Graph. (SIGGRAPH)*, vol. 34, no. 4, pp. 132:1–132:10, Jul. 2015.

[33] M. Müller, B. Heidelberger, M. Teschner, and M. Gross, "Meshless deformations based on shape matching," *ACM Trans. Graph. (SIGGRAPH)*, vol. 24, no. 3, pp. 471–478, Jul. 2005.

[34] H. Wang, "A Chebyshev semi-iterative approach for accelerating projective and position-based dynamics," *ACM Trans. Graph. (SIGGRAPH Asia)*, vol. 34, no. 6, pp. 246:1–246:9, Oct. 2015.

[35] G. Irving, J. Teran, and R. Fedkiw, "Invertible finite elements for robust simulation of large deformation," in *Proceedings of SCA*, 2004, pp. 131–140.

[36] ——, "Tetrahedral and hexahedral invertible finite elements," *Graph. Models*, vol. 68, no. 2, pp. 66–89, 2006.

[37] B. Zhu, M. Lee, E. Quigley, and R. Fedkiw, "Codimensional non-Newtonian fluids," *ACM Trans. Graph. (SIGGRAPH)*, vol. 34, no. 4, p. 115, 2015.

[38] M. Müller, D. Charypar, and M. Gross, "Particle-based fluid simulation for interactive applications," in *Proceedings of SCA*, 2003, pp. 154–159.

[39] J. J. Monaghan, "SPH without a tensile instability," *Journal of Computational Physics*, vol. 159, no. 2, pp. 290–311, 2000.

[40] N. Chentanez, M. Müller, and M. Macklin, "Real-time simulation of large elasto-plastic deformation with shape matching," in *Proceedings of SCA*. Eurographics Association, 2016, pp. 159–167.

[41] A. Colagrossi, B. Bouscasse, M. Antuono, and S. Marrone, "Particle packing algorithm for SPH schemes," *Computer Physics Communications*, vol. 183, no. 8, pp. 1641–1653, 2012.

[42] X. He, H. Wang, F. Zhang, H. Wang, G. Wang, and K. Zhou, "Robust simulation of sparsely sampled thin features in SPH-based free surface flows," *ACM Trans. Graph.*, vol. 34, no. 1, p. 7, 2014.

[43] S. Herminghaus, "Dynamics of wet granular matter," *Advances in Physics*, vol. 54, no. 3, pp. 221–261, 2005.

[44] V. Richefeu, M. S. El Youssoufi, R. Peyroux, and F. Radjai, "A model of capillary cohesion for numerical simulations of 3D polydisperse granular media," *International Journal for Numerical and Analytical Methods in Geomechanics*, vol. 32, no. 11, pp. 1365–1383, 2008.

[45] R. Narain, A. Golas, and M. C. Lin, "Free-flowing granular materials with two-way solid coupling," in *ACM Trans. Graph. (SIGGRAPH Asia)*, vol. 29, no. 6. ACM, 2010, p. 173.

[46] H. Bhattacharya, Y. Gao, and A. W. Bargteil, "A level-set method for skinning animated particle data," in *Proceedings of SCA*, Aug 2011, pp. 17–24.

[47] M. Ihmsen, N. Akinci, M. Becker, and M. Teschner, "A parallel SPH implementation on multi-core CPUs," *Computer Graphics Forum*, vol. 30, no. 1, pp. 99–112, 2011.

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

Xiaowei He is currently a research assistant professor at the HCI Lab of Institute of Software, CAS. He received his BSc degree in information management and information system and the MSc degrees in biomedical engineering from Peking University, P.R. China, in 2008, 2011, respectively. He is also working towards his Ph.D degree in computer science at the Institute of Software, Chinese Academy of Sciences. His research interests include computer graphics and physical-based simulation.

Huamin Wang is currently an assistant professor in the department of Computer Science and Engineering at the Ohio State University. Before joining OSU, he was a postdoctoral researcher in the department of Electrical Engineering and Computer Sciences at the University of California, Berkeley. He received his Ph.D. degree in Computer Science from Gerogia Institute of Technology in 2009, his M.S. degree from Stanford University in 2004, and his B.Eng. degree from Zhejiang University in 2002.

Enhua Wu received the B.S. degree from Tsinghua University, and the PhD degree from the University of Manchester (UK) in 1984. He is currently a research professor at the State Key Lab. CS, Chinese Academy of Sciences since 1985. He has also been teaching at the University of Macau since 1997, where he is currently an Emeritus Professor, and the Dean of Zhuhai-UM Science and Technology Research Institute. He has been invited to chair or co-chair various conferences or program committees including in recent years SIGGRAPH Asia 2016, ACM VRST2010, 2013, 2015, WSCG2012. He is an associate editor-in-chief of the Journal of Computer Science and Technology, and the associate editor of TVC, VI, CAVW, IJIG, IJSI. His research interests include realistic image synthesis, physically based simulation and virtual reality. He is a member of IEEE and ACM, and a fellow of the China Computer Federation (CCF).