SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

January 2023, Vol. 66 112108:1–112108:14
https://doi.org/10.1007/s11432-021-3464-2

An energy constraint position-based dynamics with corrected SPH kernel

Wei CAO[1,3†], Luan LYU[2,3†], Zhixin YANG[2,3*] & Enhua WU[3,4,5*]

[1]*College of Computer Science and Technology, China University of Petroleum, Qingdao* 266580*, China;*
[2]*State Key Laboratory of Internet of Things for Smart City, University of Macau, Macao* 999078*, China;*
[3]*Faculty of Science and Technology, University of Macau, Macao* 999078*, China;*
[4]*State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing* 100864*, China;*
[5]*Guangzhou Greater Bay Area Virtual Reality Research Institute, Guangzhou* 510000*, China*

Received 22 September 2021/Revised 3 January 2022/Accepted 4 March 2022/Published online 23 December 2022

Abstract We introduce an improved position-based dynamics method with corrected smoothed particle hydrodynamics (SPH) kernel to simulate deformable solids. Using a strain energy constraint that follows the continuum mechanics, the method can maintain the efficiency and stability of the position-based approach while improving the physical plausibility of the simulation. We can easily simulate the behavior of anisotropic and plastic materials because the method is based on physics. Unlike the previous position-based simulations of continuous materials, we use weakly structured particles to discretize the model for the convenience of deformable object cutting. In this case, a corrected SPH kernel function is adopted to measure the deformation gradient and calculate the strain energy on each particle. We also propose a solution for the interparticle inversion and penetration in large deformation. To perform complex interaction scenarios, we provide a simple method for collision detection. We demonstrate the flexibility, efficiency, and robustness of the proposed method by simulating various scenes, including anisotropic elastic deformation, plastic deformation, model cutting, and large-scale elastic collision.

Keywords position-based dynamics, deformable solids, continuum mechanics

Citation Cao W, Lyu L, Yang Z X, et al. An energy constraint position-based dynamics with corrected SPH kernel. Sci China Inf Sci, 2023, 66(1): 112108, https://doi.org/10.1007/s11432-021-3464-2

1 Introduction

The dynamic simulation of deformable solids is a classic topic in computer graphics. In recent years, physical-based methods have achieved remarkably realistic simulation effects, such as the material point method [1–3]. However, these methods come at the cost of increased computational complexity and time consumption. Müller et al. [4] proposed a position-based dynamics (PBD) approach that can simulate cloth movement in real time. The approach works on the position of vertices or particles in deformable objects directly; thus, it is more controllable and faster than the traditional physical-based methods. Subsequently, the approach has been widely investigated and used in computer games, the visual effects industry, and many other interactive applications. Nevertheless, most existing position-based approaches are geometrically motivated, which cannot handle complex material behavior. Therefore, we aim to develop a position-based method that improves the computation speed while retaining the physical properties of materials.

Several studies have combined the continuum mechanics with the position-based approach to simulate many physical phenomena [5–7]. Instead of using geometric distance to constrain the particle positions like the traditional PBD method, they used potential energy as the constraint, which enables the simulation of the physical properties of different materials by choosing the constitutive model. Most previous studies adopted mesh-based discretization methods, such as the finite element method (FEM), which can directly calculate the tensor variables on discrete elements through the shape function. However, object

* Corresponding author (email: zxyang@um.edu.mo, ehwu@um.edu.mo)
† Cao W and Lyu L have the same contribution to this work.

cutting and fracture simulation will be confronted with sophisticated topological changes. To address the aforementioned problem, the particle-based method is employed.

In this study, we develop a particle-based method with a combination of PBD and continuum mechanics. We use strain energy to constrain the particle positions within an object, where the weakly structured particles discretize the model for the convenience of deformable object cutting. Compared with the mesh-based method, the interpolation of the tensor variables into discrete particles is challenging for the particle-based method. Inspired by the smoothed particle hydrodynamics (SPH) method [8], we adopt a corrected SPH interpolation scheme to compute the deformation gradient located in potential energy. In comparison with the PBD approach based on FEM [5], a further advantage of our method is that when the number of particles is the same in the two methods, our method considerably reduces the number of constraints, thus improving the solving speed. Because of the continuum mechanics, we can easily simulate anisotropic materials. We also propose a solution for the interparticle inversion and penetration in large deformation. Moreover, both self-collision and collision between objects are considered.

In summary, our contributions are as follows.

- An energy constraint of continuum mechanics is proposed in combination with particle-based discretization, which makes the PBD method follow the physical properties of the object during deformation while at the same time dealing with the cutting problem more conveniently.
- A corrected SPH kernel function is introduced to solve the interpolation problem of the deformation gradient on each particle, which can preserve both linear and angular momenta.
- A technique is developed to solve the problem of particle inversion and penetration in large deformation.

2 Related work

In the past two decades, PBD has been a well-known approach in deformable object simulation research and is widely used in interactive applications, such as computer games and virtual reality. Compared with traditional physical-based methods, PBD is more appealing because of its simplicity and robustness. Bender et al. [9, 10] presented a survey of position-based simulation methods. Müller et al. [4] first proposed a complete PBD framework. Then, they utilized a distance constraint to immediately manipulate the positions of vertices within an object, which simplified the dynamic solution process compared with the previous methods. Macklin and Müller [11] applied PBD to incompressible fluid simulation because of its unconditionally stable time integration. Macklin et al. [12] improved the PBD by using a parallel constraint solver and simulated a wide range of interacting scenes in real time.

However, the constraints utilized in the previous studies are geometrically driven, which are unsuitable for simulating complex physical phenomena. To solve the problem, Bender et al. [5] combined a continuum-based formulation with the PBD approach to computing the strain energy. They implemented plasticity and anisotropy simulations of cloth and volumetric bodies. Bouaziz et al. [6] also presented a set of continuum-based energy potentials and replaced the Gauss-Seidel solver in the PBD method with the Jacobi solver. Same as the previous study, they adopted the FEM for discretization and further realized the fracture animation of cloth. Müller et al. [13] proposed constraints for the entries of Green's strain tensor instead of constraining the distances between particles. They applied the constraints to both triangular and tetrahedral meshes, enabling the PBD method to flexibly control the strain direction and achieve the simulation of anisotropic materials. Macklin et al. [14] presented an extended PBD algorithm (XPBD) that employs a compliant constraint formulation, which corresponds to elastic potential energy. In comparison with PBD, XPBD frees the deformable object stiffness from the control of the time step and iteration count to realize the simulation of arbitrary elastic materials. Cetinaslan [7] introduced a modified Morse potential to the parallel XPBD framework.

Although the aforementioned methods introduced strain or potential energy to enhance the physical properties of simulation, they adopted mesh or structured particles for discretization, which is unsuitable for fracture animation. For example, Ref. [6] simulated the fracture of cloth based on two-dimensional triangular meshes. He and colleagues [15] presented a meshless peridynamics framework built based on [6] to simulate versatile elastoplastic materials. By contrast, the difficulty in adjusting the structure of tetrahedral meshes in fracture will increase in the three-dimensional case. Therefore, we prefer to utilize weakly structured particles to discretize the 3D model into the PBD method. In combination with the continuum mechanics, a challenge for the unstructured particle approach is how to project the tensor variables onto discrete particles.

As a typical meshless method, SPH is well-known in the field of computer graphics. Early SPH methods were mainly concerned with fluid. Ihmsen et al. [16] provided a detailed survey on this topic. Desbrun and Gascuel [17] extended SPH to animate deformable bodies. However, the standard SPH interpolation does not satisfy rotational invariance, which is fatal to solid simulation. Therefore, some correction of SPH is usually required when simulating soft body deformation. Solenthaler et al. [18] utilized SPH with a unified particle model to simulate fluid-solid interactions. Becker et al. [19] presented a corotational formulation integrated with SPH for deformable solids. Gerszewski et al. [20] modified the SPH for animating elastoplastic materials. Within the SPH framework, Jones et al. [21] introduced an embedded space for elastoplastic simulation. Recently, in the field of physics and engineering, Refs. [22,23] proposed improved SPH methods to solve large deformation problems. Similarly, the moving least squares (MLS) is another well-known approach for meshless solid simulation, and it can be used to correct the SPH kernel. Müller et al. [24] achieved point-based animation by using MLS to interpolate the tensor field. Martin et al. [25] chose both linear and quadratic generalized MLS to generate the unified simulation for rods, shells, and solids. Chen et al. [26] achieved unified multiphase continuum simulation and interaction by using the MLS reproducing kernel (MLSRK). Several other meshless methods, such as sparse frame-based models [27, 28] and elastic weighting mechanism [29], also obtained good simulation results. Inspired by meshless solid simulation and considering the balance between efficiency and accuracy, we adopt a corrected SPH kernel to calculate the deformation gradient on particles.

To solve the element inversion problem in large deformation, Irving and Teran et al. [30, 31] proposed the invertible finite element (IFE) approach. McAdams et al. [32] presented a stable solution by careful treatment of linearization. Stomakhin et al. [33] improved the IFE framework by extrapolating the polynomial of the uninverted part in the singular value space. Bender and colleagues [5] also employed the IFE approach for inversion handling. Smith et al. [34] focused on flesh simulation, proposed a novel Neo-Hookean elastic model robust to kinematic rotations and inversions, and adopted mesh-based methods.

Several anisotropic methods for the simulations using the strain energy function to calculate deformation have been proposed successively. Irving et al. [30] accomplished transverse isotropy through an extra stress term. Based on their work, Xu et al. [35] introduced a principal stretch method to design orthotropic materials. Xu and colleagues [36] focused on modeling general hyperelastic materials with peridynamics and added an anisotropic kernel. Clyde et al. [37] developed a novel orthotropic hyperelastic constitutive model for woven fabrics.

3 Background

The PBD method has advantages in simulating large-scale interactive scenes because of its simplicity and stability. Most of its early studies were geometrically motivated and could not describe the complex physical behavior of deformable objects. As a result, some PBD methods combining continuum mechanics have emerged. This section will provide an overview of the PBD approach and some basic notations in continuum mechanics.

3.1 PBD framework

Our work is conducted within the PBD framework. We suppose that a deformable object is discretized into n vertices with the positions $\boldsymbol{x} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ and masses $m = \{m_1, \ldots, m_n\}$. The PBD algorithm mainly consists of three steps. First, the position $\boldsymbol{x}^{n+1}$ of each vertex is predicted by a time integration scheme. Then, a set of bilateral constraints $C(\boldsymbol{x}) = 0$ are defined to modify the predicted vertex positions. Finally, the vertex velocities $\boldsymbol{v}^{n+1}$ are updated with the modified positions. The core of the PBD approach lies in the definition of position constraints. The purpose here is to determine the position corrections $\Delta\boldsymbol{x}$ that satisfy $C(\boldsymbol{x} + \Delta\boldsymbol{x}) = 0$. The equation can be expanded by the Taylor polynomial, as follows:

$$C(\boldsymbol{x} + \Delta\boldsymbol{x}) \approx C(\boldsymbol{x}) + \nabla_x C^{\mathrm{T}}(\boldsymbol{x})\Delta\boldsymbol{x} = 0. \tag{1}$$

From this, the position correction of each vertex can be computed as follows:

$$\Delta\boldsymbol{x}_i = w_i \nabla_{\boldsymbol{x}_i} C(\boldsymbol{x})\lambda, \tag{2}$$

where $w_i = 1/m_i$ is the inverse vertex mass. λ is a Lagrange multiplier calculated by substituting (2) into (1), as follows:

$$\lambda = -\frac{C(\boldsymbol{x})}{\sum_j w_j \left|\nabla_{\boldsymbol{x}_j} C(\boldsymbol{x})\right|^2}, \tag{3}$$

where $\sum_j$ represents all of the vertices in the constraint that are associated with $\boldsymbol{x}_i$. The conservations of linear and angular momenta are implicit in the aforementioned constraints.

Since the traditional PBD constraint is geometrically driven, a stiffness parameter is also needed to control the elastic property. The simplest method for merging stiffness is to multiply the position correction $\Delta\boldsymbol{x}_i$ by the stiffness parameter $k_i \in [0, 1]$ [4]. However, stiffness is dependent on the simulation time step and iteration count. To overcome this limitation, XPBD [14] provided a total Lagrange multiplier $\Delta\lambda$ instead of λ in (3), which is defined as follows:

$$\Delta\lambda = -\frac{C(\boldsymbol{x}) + \tilde{\alpha}\lambda}{\sum_j w_j \left|\nabla_{\boldsymbol{x}_j} C(\boldsymbol{x})\right|^2 + \tilde{\alpha}}, \tag{4}$$

where $\tilde{\alpha} = \alpha/\Delta t^2$ is related to the stiffness expression considering the time step Δt. Afterward, the position correction can be reformulated as follows:

$$\Delta\boldsymbol{x}_i = w_i \nabla_{\boldsymbol{x}_i} C(\boldsymbol{x}) \Delta\lambda. \tag{5}$$

Similar to the vertex position x, λ is updated with $\Delta\lambda$ in each time step. We refer the readers to [14] for more details.

3.2 Notations in continuum mechanics

To make the simulation more physically plausible, our work is combined with continuum mechanics. Let $\boldsymbol{X} \in \mathbb{R}^3$ denote the particle position in the undeformed configuration and let $\boldsymbol{x} \in \mathbb{R}^3$ denote the particle position in the deformed configuration. The deformation gradient $\boldsymbol{F} = \partial\boldsymbol{x}/\partial\boldsymbol{X}$ is the Jacobian matrix of deformation mapping, which distinguishes shape changes from the displacement of the deformable objects. The properties of materials can be expressed by the strain energy $\Psi(\boldsymbol{F})$ and stress tensor $\boldsymbol{P}(\boldsymbol{F})$, functions of the deformation gradient, where $\boldsymbol{P}(\boldsymbol{F}) = \partial\Psi(\boldsymbol{F})/\partial\boldsymbol{F}$. We provide a detailed definition of $\Psi(\boldsymbol{F})$ and $\boldsymbol{P}(\boldsymbol{F})$ in the subsequent section according to the specific situation.

4 Our method

In this section, we will introduce our particle-based method to the PBD framework. We employ the corrected SPH kernel to compute the deformation gradient on each particle. The method follows the continuum mechanics by using an energy constraint, making it easier for us to simulate anisotropic materials. The inversion and collision problems are also considered.

4.1 Corrected SPH interpolation

Inspired by [5], we use the strain energy to replace the previous geometric constraints, thus following the continuum mechanics. For this reason, the position constraint on each particle is defined as $C(\boldsymbol{x}) = \Psi(\boldsymbol{F}) = 0$. $\Psi(\boldsymbol{F})$ is the energy density function. The common constitutive models used to formulate $\Psi(\boldsymbol{F})$ include the corotated linear, Saint Venant Kirchhoff (StVK), and Neo-Hookean models. The StVK and Neo-Hookean are nonlinear models that exhibit more complex deformation behavior but require more computation time. All of the aforementioned models can be directly applied to our algorithm framework. Most of our elastic simulations adopt the Neo-Hookean constitutive model in the following form:

$$\Psi(\boldsymbol{F}) = \frac{\mu}{2}(\mathrm{tr}(\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F}) - 3) - \mu \ln J + \frac{\lambda}{2}\ln^2 J, \tag{6}$$

where $\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F}$ is the right Cauchy-Green strain tensor. $J = \det(\boldsymbol{F})$ is the determinant of the deformation gradient that represents the ratio of the unit volume changes from the undeformed domain to the deformed domain. μ and λ are Lamé coefficients consisting of Young's modulus E and Poisson's ratio ν, which can be expressed as follows:

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)}, \quad \mu = \frac{E}{2(1+\nu)}.$$

Then, the corresponding first Piola-Kirchhoff stress is calculated as follows:

$$\boldsymbol{P}(\boldsymbol{F}) = \mu\boldsymbol{F} - \mu\boldsymbol{F}^{-\mathrm{T}} + \lambda \ln J\boldsymbol{F}^{-\mathrm{T}}.$$

By contrast, the deformation gradient of the mesh-based method can be calculated directly by using its edge vectors derived from each element. Meanwhile, for the particle-based method, computing the deformation gradient on each discrete particle is difficult. Here, we adopt the idea of SPH [8] and use the corrected SPH kernel to calculate the deformation gradient for each particle, as follows:

$$\boldsymbol{F}_i = \sum_j V_j(\boldsymbol{u}_j - \boldsymbol{u}_i) \otimes \tilde{\nabla}\tilde{W}_{ij}(\boldsymbol{X}), \tag{7}$$

where V is the initial particle volume and $\boldsymbol{u} = \boldsymbol{x} - \boldsymbol{X}$ is the particle displacement. In this case, we use the corrected gradient of the corrected kernel $\tilde{\nabla}\tilde{W}_{ij}$ to make the deformation gradient rotation aware.

First, the kernel function W_{ij}, which interpolates the quantities into a specific particle from its surrounding particles, is defined as follows:

$$W_{\text{poly6}}(\boldsymbol{r}, h) = \begin{cases} \frac{315}{64\pi h^9}(h^2 - r^2)^3, & 0 \leqslant r \leqslant h, \\ 0, & \text{otherwise}, \end{cases}$$

where $r = |\boldsymbol{r}| = \|\boldsymbol{X}_j - \boldsymbol{X}_i\|_2$ is the radius of the kernel function and $h = 0.0625$ in our work. Then, the gradient of the kernel function is derived as follows:

$$\nabla W_{\text{poly6}}(\boldsymbol{r}, h) = \begin{cases} -\frac{945}{32\pi h^9}(h^2 - r^2)^2\boldsymbol{r}, & 0 \leqslant r \leqslant h, \\ 0, & \text{otherwise}. \end{cases}$$

However, the aforementioned kernel function cannot preserve the angular momentum. Hence, we make corrections to both the kernel function and its gradient. The corrected kernel is computed as follows:

$$\nabla\tilde{W}_{ij}(\boldsymbol{X}) = \frac{\nabla W_{ij}(\boldsymbol{X}) - \gamma_i(\boldsymbol{X})}{\sum_j V_j W_{ij}(\boldsymbol{X})},$$

where $\gamma_i(\boldsymbol{X}) = \sum_j V_j \nabla W_{ij}(\boldsymbol{X}) / \sum_j V_j W_{ij}(\boldsymbol{X})$.

The corrected gradient is formulated as follows:

$$\tilde{\nabla}\tilde{W}_{ij}(\boldsymbol{X}) = \boldsymbol{L}_i \nabla\tilde{W}_{ij}(\boldsymbol{X}), \tag{8}$$

where the correction matrix $\boldsymbol{L}_i = (\sum_j V_j \nabla\tilde{W}_{ij}(\boldsymbol{X}) \otimes (\boldsymbol{X}_j - \boldsymbol{X}_i))^{-1}$.

With the deformation gradient discretized into each particle, we can use the energy function to calculate the distance constraint of the particles. The constraint gradient is computed as follows:

$$\nabla_{\boldsymbol{x}_k} C_i(\boldsymbol{x}) = \nabla_{\boldsymbol{x}_k} \Psi_i(\boldsymbol{F}_i) = \boldsymbol{P}_i(\boldsymbol{F}_i)\nabla_{\boldsymbol{x}_k}\boldsymbol{F}_i, \tag{9}$$

where the gradient of deformation gradient is derived as follows:

$$\nabla_{\boldsymbol{x}_k}\boldsymbol{F}_i = \begin{cases} -\sum_j V_j \tilde{\nabla}\tilde{W}_{ij}(\boldsymbol{X}), & k = i, \\ V_j \tilde{\nabla}\tilde{W}_{ij}(\boldsymbol{X}), & k = j. \end{cases}$$

Here, we use the total Lagrange multiplier based on the idea of XPBD [22]:

$$\Delta\lambda_i = -\frac{\Psi_i(\boldsymbol{F}_i) + \tilde{\alpha}_i\lambda_i}{\sum_j w_j \left|\nabla_{\boldsymbol{x}_j}\Psi_i(\boldsymbol{F}_i)\right|^2 + \tilde{\alpha}_i}, \tag{10}$$

where $\tilde{\alpha} = 1/\Delta t^2$. Then, by substituting (9) and (10) into (5), the position correction can be reformulated as follows:

$$\Delta\boldsymbol{x}_i = \sum_j (\Delta\lambda_j \boldsymbol{P}_j(\boldsymbol{F}_j)\tilde{\nabla}\tilde{W}_{ji}(\boldsymbol{X})V_i - \Delta\lambda_i \boldsymbol{P}_i(\boldsymbol{F}_i)\tilde{\nabla}\tilde{W}_{ij}(\boldsymbol{X})V_j). \tag{11}$$

Notably, in simulating pure deformation, the kernel function only needs to be calculated once in the initial configuration; thus, there is no additional computation cost to the PBD algorithm. The flow of our algorithm is summarized in Algorithm 1.

Algorithm 1 Simulation steps of our energy constraint PBD

```
 1: $v^{n+1} \leftarrow v^n + \Delta t M^{-1} f^n_{\text{ext}}$;
 2: $x^{n+1} \leftarrow x^n + \Delta t v^{n+1}$;
 3: Perform collision detection and response;
 4: for iter := 1 to maxiterations do
 5:     for each energy constraint $C(x)$ do
 6:         Compute deformation gradient $F$ (Eq. (7));
 7:         Compute the total Lagrange multiplier $\Delta\lambda$ (Eq. (10));
 8:         $\lambda^{n+1} \leftarrow \lambda^{n+1} + \Delta\lambda$;
 9:         Determine the position changes $\Delta x$ (Eq. (11));
10:         $x^{n+1} \leftarrow x^{n+1} + \Delta x$;
11:     end for
12:     for each collision constraint do
13:         Compute the position changes $\Delta x^{\text{col}}$ (Eq. (12));
14:         $x^{n+1} \leftarrow x^{n+1} + \Delta x^{\text{col}}$;
15:     end for
16: end for
17: Perform object cutting;
18: $v^{n+1} \leftarrow \frac{1}{\Delta t}(x^{n+1} - x^n)$.
```

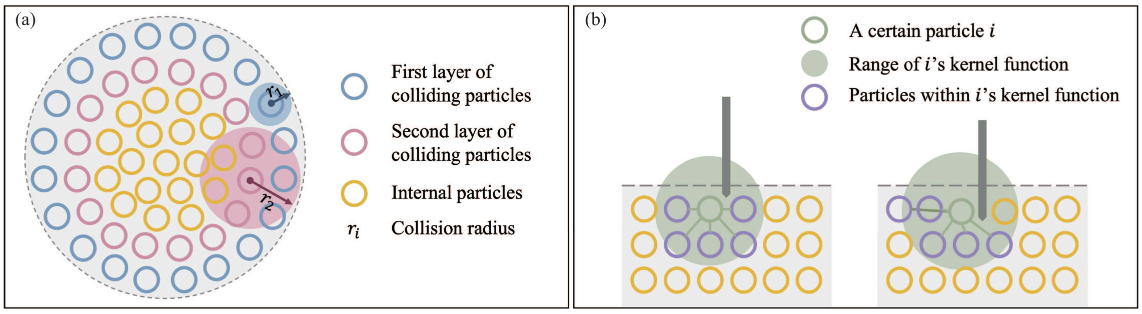
(a)
r_1
r_2
First layer of colliding particles
Second layer of colliding particles
Internal particles
r_i
Collision radius
(b)
A certain particle i
Range of i's kernel function
Particles within i's kernel function

Figure 1 (Color online) Particle structure. (a) Colliding particles setting; (b) change in the relationship between particles within a kernel function during the cutting process.

4.2 Collision and inversion handling

Concerning collision detection, we examine two layers of particles on the surface. If we only consider the collision of surface particles, then the time step will be limited to a small setting, which can be solved by adding a layer of colliding particles. We set a collision radius r_i for each colliding particle. Figure 1(a) shows the schematic of colliding particles. In the case of a collision between objects, we impose a simple constraint $C_i^{\text{col}}(\boldsymbol{x}) = \|\boldsymbol{x}_i - \boldsymbol{x}_j\| - d$. We can detect the collision of particles if the distance between two particles is less than the sum of their collision radii $d = r_i + r_j$. Collision correction is calculated as follows:

$$\Delta \boldsymbol{x}_i^{\text{col}} = \sum_j \frac{w_i}{w_i + w_j}(d - \|\boldsymbol{x}_i - \boldsymbol{x}_j\|)\frac{\boldsymbol{x}_i - \boldsymbol{x}_j}{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|}. \tag{12}$$

Collision correction is used to correct the particle position, as shown in Lines 13 and 14 of Algorithm 1. Based on the collision between objects, we utilize an intuitive technique for self-collision. Self-collision detection is triggered only if the initial distance between two particles is greater than a given value.

Because the linear constitutive model can inherently process the inversion problem, we directly combine the linear and nonlinear models for handling inversion. To avoid the rotation aliasing problem in the linear model, we adopt the corotated linear constitutive model for the inversion part. The energy density function is defined as follows:

$$\Psi(\boldsymbol{F}) = \mu \|\boldsymbol{F} - \boldsymbol{R}\|_F^2 + \frac{\lambda}{2}\text{tr}^2\left(\boldsymbol{R}^{\mathrm{T}}\boldsymbol{F} - \boldsymbol{I}\right).$$

Then, the corresponding first Piola-Kirchhoff stress is calculated as follows:

$$\boldsymbol{P}(\boldsymbol{F}) = 2\mu(\boldsymbol{F} - \boldsymbol{R}) + \lambda\text{tr}(\boldsymbol{R}^{\mathrm{T}}\boldsymbol{F} - \boldsymbol{I})\boldsymbol{R},$$

where the deformation gradient $\boldsymbol{F} = \boldsymbol{RS}$ is decomposed into a rotation matrix $\boldsymbol{R}$ and a symmetric tensor $\boldsymbol{S}$. We assess the inversion by the singular value decomposition of $\boldsymbol{F} = \boldsymbol{U}\hat{\boldsymbol{F}}\boldsymbol{V}^{\mathrm{T}}$ and check the sign of the

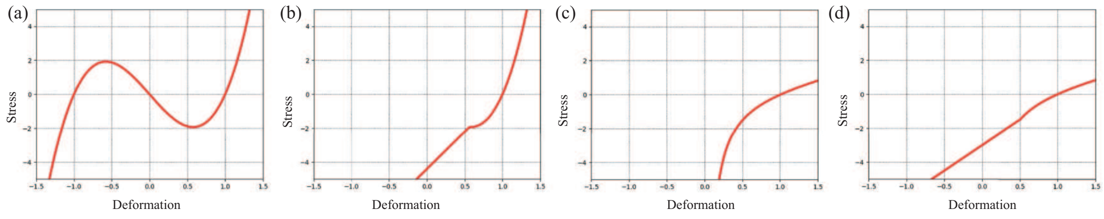

Figure 2 (Color online) Uniaxial stress-strain curve. (a) StVK model; (b) combination of the StVK and corotated linear models; (c) Neo-Hookean model; (d) combination of the Neo-Hookean and corotated linear models.

singular value in $\hat{\boldsymbol{F}}$. When the StVK elastic body is compressed, its restorative force reaches a maximum as the compression reaches a critical threshold ($\approx 58\%$ of the undeformed configuration). When the singular value becomes less than the critical compression threshold, the restorative force decreases and an inversion occurs. Then, we utilize the corotated linear model to prevent the volume from inverting. We apply an analogous modification to the Neo-Hookean model. The comparison of the stress-strain curves of different constitutive models is shown in Figure 2. Notably, the Neo-Hookean model exhibits a stronger reaction to extreme compression than the two other models. However, in the case where the deformation is inverted, the reaction is unclear. Conversely, the StVK model exhibits more resistance against stretching than the two other models.

4.3 Anisotropy and plasticity simulation

Because of the continuum mechanics, we can easily simulate complex material behavior by tuning the strain energy function. Inspired by Xu and colleagues [35], we decompose the energy function into the isotropic and anisotropic energy terms, $\Psi(\boldsymbol{F}) = \Psi^{\text{iso}}(\boldsymbol{F}) + \Psi^{\text{ortho}}(\boldsymbol{F})$, for the anisotropy simulation. We directly employ the Neo-Hookean model of (6) for the isotropic elasticity term. To keep the material behavior of the anisotropic term consistent with that of the isotropic term, we use a simplified Neo-Hookean model to simulate the anisotropic elasticity term. In this study, we mainly consider the orthotropic case. Hence, assuming the orthogonal material directions are $\boldsymbol{b}_i$ with $i = 1, 2, 3$, the orthotropic strain energy can be computed as follows:

$$\Psi^{\text{ortho}}(\boldsymbol{F}) = \sum_i \omega_i(\xi_i), \quad \omega_i(\xi_i) = \beta_i \left(\frac{\xi_i^2 - 1}{2} - \ln \xi_i \right),$$

where $\beta_i = \|\boldsymbol{F}\boldsymbol{b}_i\|_2$. ξ_i is the stiffness coefficient associated with the orthotropic direction $\boldsymbol{b}_i$. Then, the first Piola-Kirchhoff stress tensor is calculated as follows:

$$\boldsymbol{P}^{\text{ortho}}(\boldsymbol{F}) = \sum_{i=1}^{3} \frac{\omega_i'(\xi_i)}{\xi_i} \boldsymbol{F}\boldsymbol{b}_i \otimes \boldsymbol{b}_i,$$

where $\omega_i'(\xi_i) = \beta_i(\xi_i - 1/\xi_i)$, $\boldsymbol{b}_i \otimes \boldsymbol{b}_i = \boldsymbol{b}_i\boldsymbol{b}_i^{\text{T}}$. We determine that β_i controls the scaling of $\omega_i'(\xi_i)$, with $i = 1$ representing the transversely isotropic material and $i = 2$ representing the orthotropic material.

For plastic deformation, we adopt the StVK constitutive model because its stress increases with tensile deformation rapidly, thereby easily reaching the plastic yield criterion (as shown in Figure 2). The energy density function of the StVK constitutive model is formulated as follows:

$$\Psi\left(\boldsymbol{F}\right) = \mu \left\|\epsilon\right\|_F^2 + \frac{\lambda}{2} \text{tr}^2\left(\epsilon\right),$$

where $\epsilon = \frac{1}{2}(\boldsymbol{F}^{\text{T}}\boldsymbol{F} - \boldsymbol{I})$ is the Green strain tensor. Then, the corresponding first Piola-Kirchhoff stress is calculated as follows:

$$\boldsymbol{P}(\boldsymbol{F}) = \boldsymbol{F}(2\mu\epsilon + \lambda \text{tr}(\epsilon)\boldsymbol{I}).$$

We follow the process in [5] and decompose the strain into two components, that is, $\epsilon = \epsilon_{\text{elas}} + \epsilon_{\text{plas}}$. By using the von Mises yield criterion, plastic deformation occurs when the Frobenius norm of elastic strain exceeds a critical value. Thereafter, the plastic strain will absorb part of the elastic strain. As a result, the deformable object cannot be restored to its initial state even without external force.

Table 1 Parameters and timings[a)]

Experiment	♯Particle	ms/f	col(ms)	Δt/f	S/f	iter	Elastic model	(E, ν)
Tori (Figure 3)	50544	20.200	4.266	1/60	3	8	Neo-Hookean & Corotated	(3000, 0.4)
Bunny (Figure 4)	41555	13.620	2.994	1/60	2	8	Neo-Hookean & Corotated	(500, 0.4)
Cow (Figure 5 bottom)	188257	27.750	–	1/90	2	8	Neo-Hookean & Corotated	(2000, 0.4)
Cow [5] (Figure 5 top) (1174842 tetrahedra)	187382	178.820	–	1/90	2	8	Neo-Hookean & Corotated	(2000, 0.4)
Towel (Figure 6)	8404	11.952	–	1/60	4	10	Neo-Hookean & Corotated	(50, 0.4)
Metal (Figure 7)	4875	15.978	7.200	1/60	4	8	StVK & Corotated	(2000, 0.4)
Sausages (Figure 8)	38448	21.676	5.683	1/60	3	8	Neo-Hookean & Corotated	(2000, 0.4)

a) "♯Particle" is the number of particles, "ms/f" is the total operation time per frame, "col" is the collision time per frame, "Δt/f" is the time step per frame, and "S/f" is the number of substeps per time step.

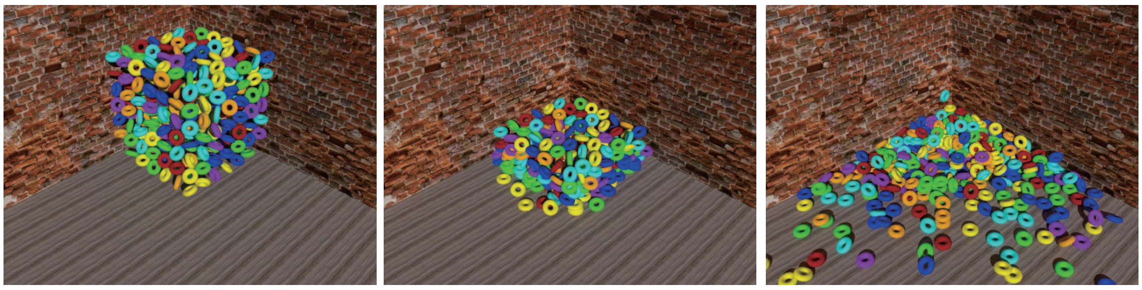

Figure 3 (Color online) Simulation of 324 elastic tori containing 50544 particles falling to the ground along the walls.

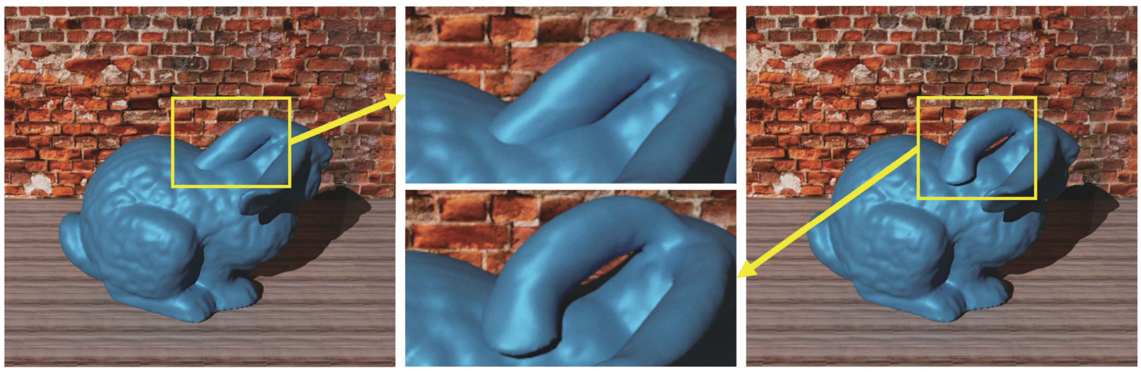

Figure 4 (Color online) Self-collision detection of the bunny model falling to the ground. The images are selected from the same frame of animation. The one on the left has no self-collision detection function, whereas the one on the right has a self-collision detection function.

4.4 Object cutting

Because of our weakly structured particle model, we can easily deal with the fracture during object cutting. For two adjacent particles, we assume that an edge connects them. The edge will be disconnected when it intersects the cutting tool. In other words, the two particles are not acting on each other's kernel functions anymore. At the same time, to enhance stability, we will select and add an additional nearest particle to the kernel constraint. Figure 1(b) shows the operation that regulates the relationship between particles. In this case, the corrected kernel function in Subsection 4.1 needs to be recomputed.

5 Experimental results

Our implementations were executed on a 3.6-GHz Intel Core i7-6850K CPU and NVIDIA GeFore GTX 1080 GPU. Rendering was done in Houdini. The model size, computational efficiency, and parameter settings are given in Table 1 (Figures 3–8). In addition to the cow compression experiment shown in Figure 5, the dynamic process of deformation is fast because of the large strain energy, such that we can reduce the time step of each frame to $\Delta t = 1/90$ for observation. We use a fixed time step of $\Delta t = 1/60$ per frame for all other experiments. In traditional PBD approaches, large iteration counts will make the

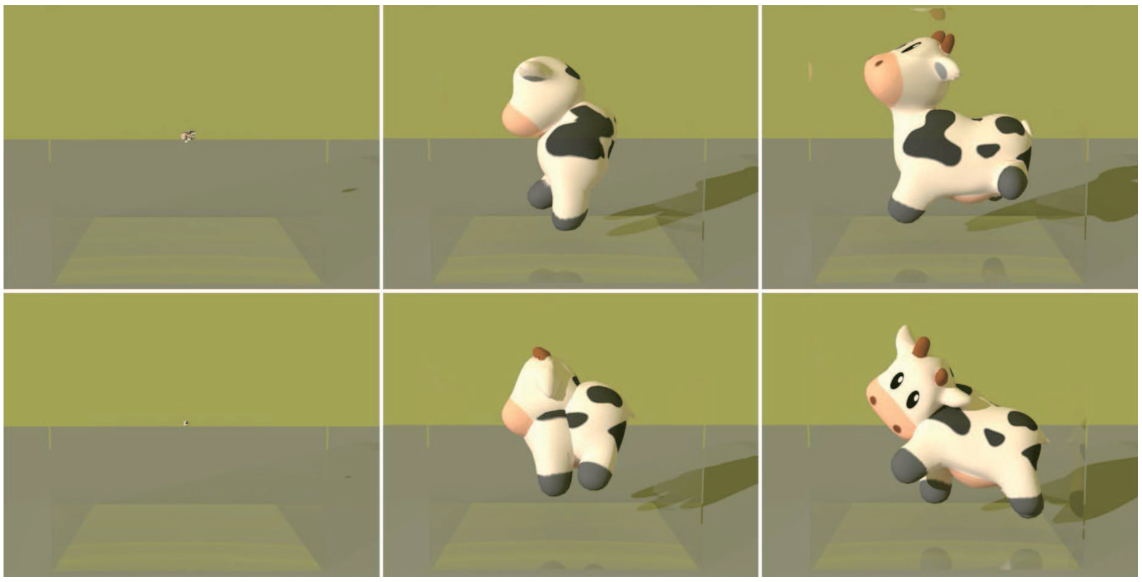

Figure 5 (Color online) A cow model recovering from a completely compressed ball. The top row is the result of the method in [5], whereas the bottom row is the result of our method.

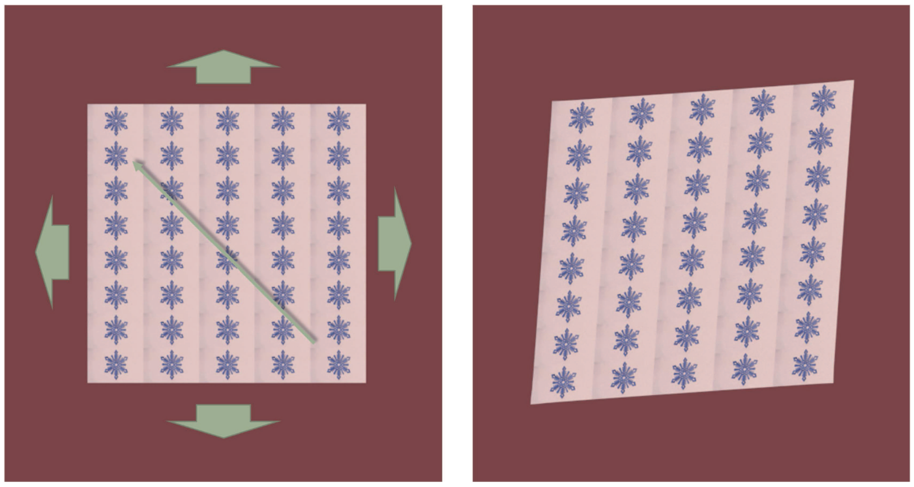

Figure 6 (Color online) A towel model is stretched by the outward tensile forces that are perpendicular to the edges. The anisotropic direction is set along a 45° angle.

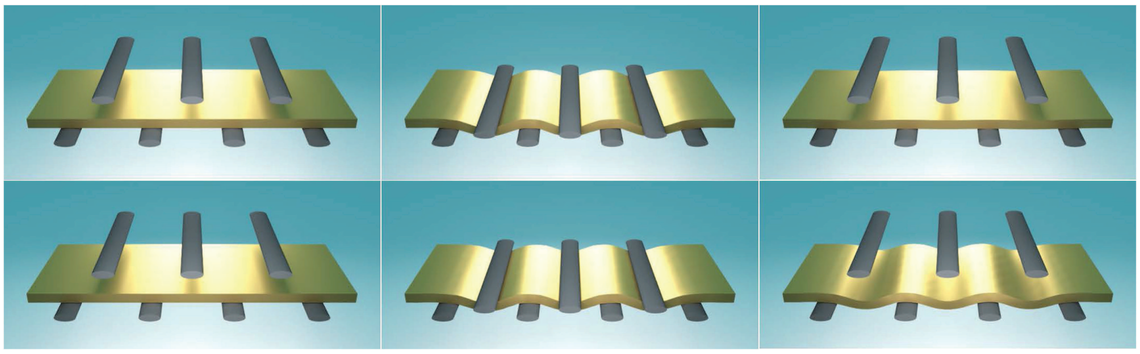

Figure 7 (Color online) Comparison of the deformation between elastic and plastic materials, which simulates the process of heavy objects from falling to bouncing. The top row is elastic material, whereas the bottom row is plastic material.

deformable object stiff. However, our method can control the stiffness by using a strain energy function.

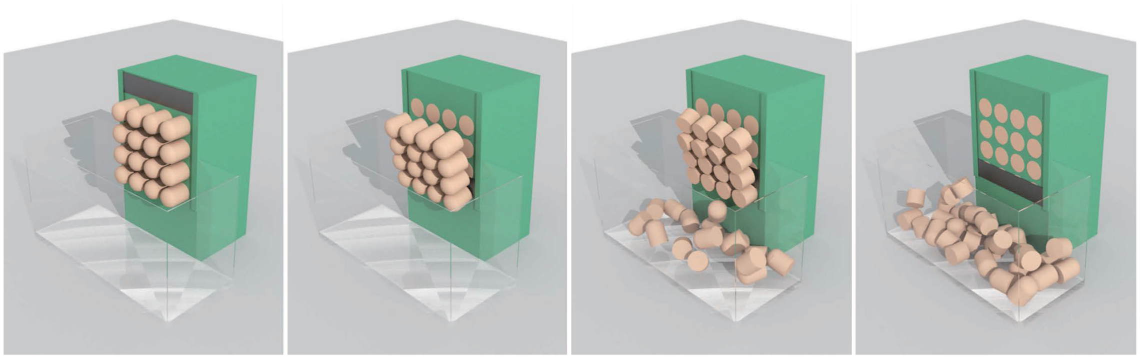

Figure 8 (Color online) Sausages are pushed out of the machine and cut with a knife, with the pieces falling into a glass box. Collision detection is also performed in this scenario.

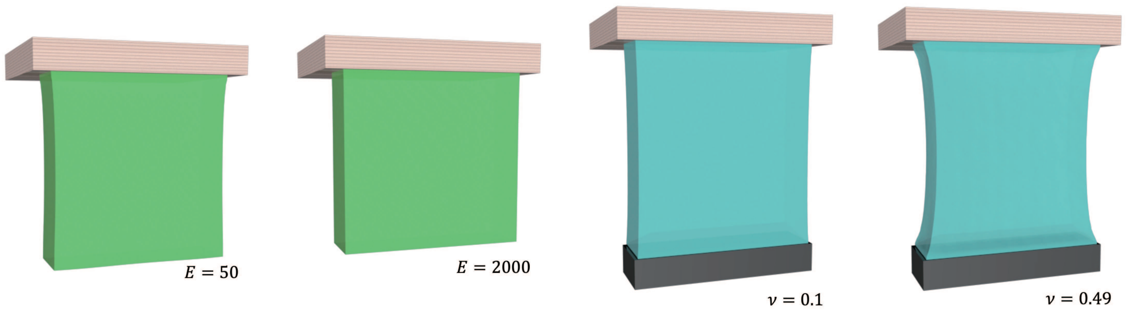

Figure 9 (Color online) Testing the effect of Young's moduli E and Poisson's ratios ν on the elastic deformation algorithm.

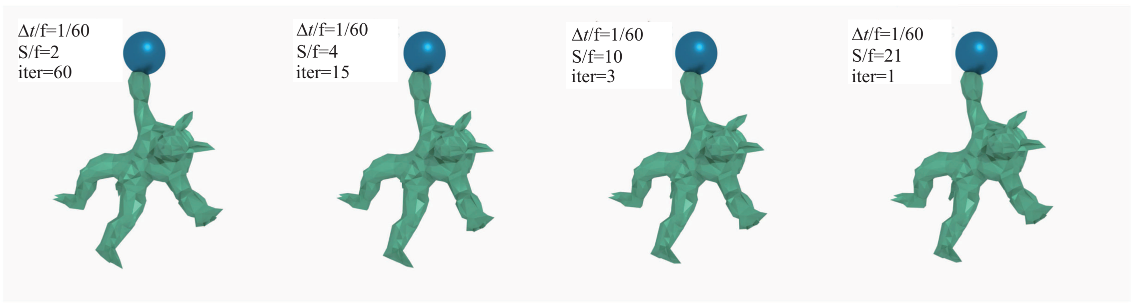

Figure 10 (Color online) The parameter setting of time steps and the iterations of PBD solver to capture similar simulation results. The armadillo is fixed with one hand and subjected to gravity.

Parameter testing. Because of the continuum mechanics, we can simulate more complex material behavior according to the strain energy functions. By improving the constraint dependence through XPBD, the stiffness coefficients can control the elastic strength of a deformable object more effectively. We show the parameter testing in Figure 9 with different Young's moduli E and Poisson's ratios ν. In Young's modulus test, the object with its top fixed is affected by gravity, which is a volume force. We observe that the larger Young's modulus E is, the stiffer the object is. In the Poisson's ratio test, the object is pulled by a surface force at the bottom. We observe that the larger the Poisson's ratio ν is, the more the contraction in the direction perpendicular to the tension is. Moreover, the surface force produces a more pronounced deformation than the volume force. We also test the relationship between the time step and the iterations of the PBD solver. Figure 10 shows the parameter setting used to achieve similar simulation effects. Meanwhile, Table 2 shows the corresponding implementation times. The simulation time step is equal to "Δt/f" divided by "S/f". We can conclude from the data shown in the table that the simulation speed can be improved by appropriately choosing a smaller time step and fewer iterations.

Accuracy and robustness. In essence, SPH interpolation is similar to MLS with constant basis functions, which cannot handle the rotation of objects. Usually, MLS adopts a linear or quadratic basis to ensure linear consistency; however, it will reduce the computational efficiency. To maintain the efficiency of the

Table 2 We test the effect of time steps and PBD iterations on the implementation time[a)]

Δt/f	S/f	Iterations of PBD solver	Simulation time (s)
1/60	2	60	12.763
1/60	4	15	8.171
1/60	10	3	5.275
1/60	21	1	6.291

a) "Δt/f" is the time step per frame. "S/f" is the number of substeps per time step. The last column shows the simulation time of 500 frames.

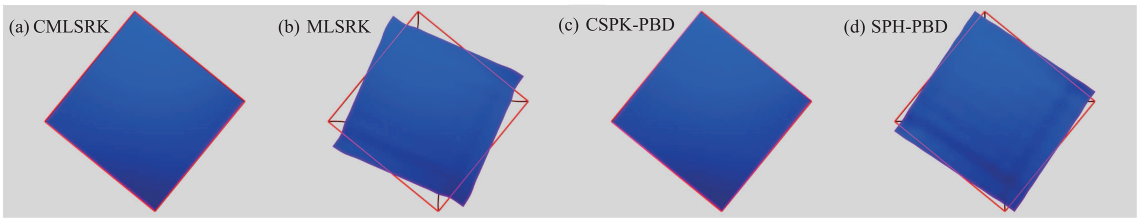

Figure 11 (Color online) Rotation test of our SPH-PBD method and the MLSRK method [26]. (a) and (b) are the MLSRK method with a constant basis, and we add our corrected kernel function into (a). (c) and (d) are our SPH-PBD method with and without the kernel correction, respectively.

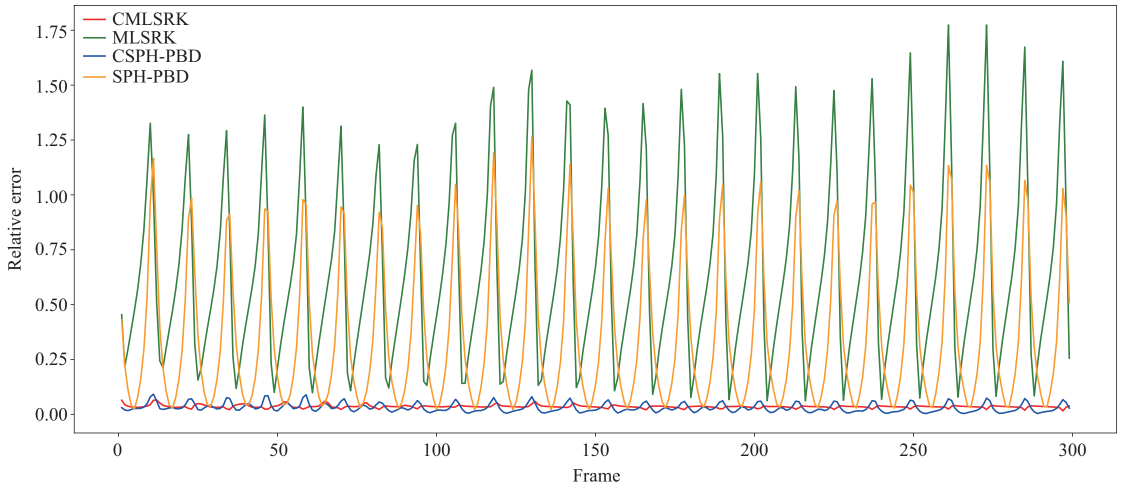

Figure 12 (Color online) Comparison of relative error between our SPH-PBD method and the MLSRK method [26] with different kernels.

standard SPH interpolation while enhancing its rotational invariance, we add a correction to its kernel. Figures 11 and 12 show the comparison of the accuracy between our proposed method and MLSRK in [26]. We regard the MLSRK with a linear basis as the baseline and compare the rotation errors of four different kernels. We determine that our correction function is effective and has good compatibility. We also compare the robustness of the two methods shown in Figure 13. When their velocities are large, their simulation collapsed, indicating that the MLSRK is unstable under extremely large deformation. Table 3 shows the comparison of the execution time of our corrected SPH kernel and the MLS kernel with a linear basis. Results show that our proposed SPH-PBD method is more suitable for the real-time simulation of large-scale interactive scenes than the MLS kernel with a linear basis.

Collision and inversion handling. Particle structures make collision detection easier, where we can use some intuitive techniques for collision processing. The following experiments demonstrate the effectiveness of our method. We simulate a complex collision situation in Figure 3, in which 324 elastic torus models with 50544 particles fall to the ground. Our approach requires 20.2 ms per time step on average and achieves a near real-time visual effect. In Figure 4, we simulate the animation of the bunny model falling to the ground and compare the results with and without self-collision detection.

To demonstrate the stability of our algorithm, we simulate a completely degenerate configuration in Figure 5. The figure shows a cow model recovering from an extremely compressed ball in a vacuum. Notably, the algorithm is stable under high stress levels and deals well with the volume inversion problem.

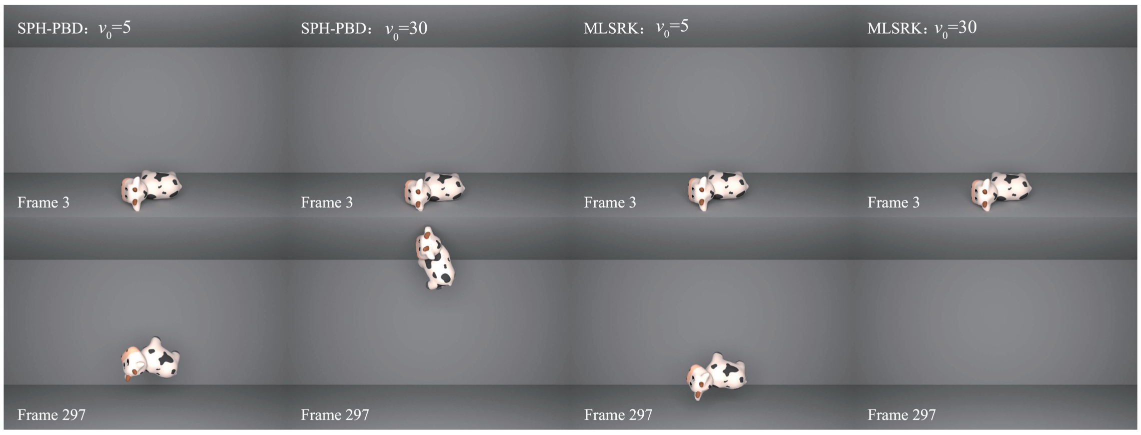

Figure 13 (Color online) Comparison of robustness between our SPH-PBD method and the MLSRK method [26]. We simulate the animation of a cow placed on the ground with an initial velocity v_0 and then bounced up.

Table 3 Time comparison between the corrected SPH kernel and the MLS kernel with linear basis. The performance was tested on both CPU (run 500 times) and GPU (run 3000 times) with 17325 particles.

	CPU (s)	GPU (s)
Corrected SPH kernel	1.645	0.815
MLS kernel with linear basis	4.688	0.929

Furthermore, we compare our method with [5] and obtain similar results. The data shown in Table 1 indicate that our method is faster than that of [5] when the models have the same number of particles because their energy constraints act on the elements, which are usually several times the number of particles. By contrast, our energy constraints act directly on the particles.

Anisotropy and plasticity. Complex material behavior such as anisotropy and plasticity, can be easily achieved by designing the strain energy and stress. Figure 6 shows the simulation of an anisotropic towel model. We add an extra energy term in the 45° direction, which makes the material stiffer in that direction. Then, we apply outward tensile forces perpendicular to the edges on the sides of the towel, resulting in an oblique deformation. Notably, the direction perpendicular to the set direction is more likely to produce tensile deformation.

We compare the elastic and plastic materials, as shown in Figure 7. Heavy objects are dropped onto metal-like materials. After the objects bounce, the elastic material will return to its original shape, whereas the plastic material will keep its bent shape. The deformation behavior of anisotropic and plastic materials can be easily simulated by using the existing approaches.

Object cutting. As shown in Figure 8, we simulate a sausage cutting scenario in the food industry. The sausages are pushed out of the machine and cut with a knife, with the pieces falling into a glass box. At the same time, collision detection is performed in this scenario.

6 Conclusion and future work

In this study, we developed an improved PBD method based on continuum mechanics to simulate deformable objects. To reduce the dependence of the stiffness coefficient on the time step and iteration count, we applied a total Lagrange multiplier based on the idea of XPBD to act on the position constraints. The method is discretized with weakly structured particles. For the deformation gradient used to calculate the strain energy, we employed a corrected SPH kernel function to interpolate it into discrete particles. Based on the improved PBD framework, we also introduced some simple methods to process the collision and inversion problems. Because of the energy representation of the constraints, we directly used existing approaches to simulate anisotropic and plastic deformation. Furthermore, we examined the object cutting scenario, which can be more easily handled by our proposed method than by mesh-based methods.

At present, our cutting simulation only involves the structural change of the particle model. We plan to introduce fracture energy [2] to assess the crack origin and evolution direction in the future. Consequently, more complex fracture scenarios will be simulated.

Acknowledgements This work was supported by Fund of China University of Ptroleum (Grant No. 21CX06042A), National Natural Science Foundation of China (Grant Nos. 62072449, 61972271), Macau FDCT Grant (Grant No. 0018/2019/AKP), Science and Technology Development Fund, Macau SAR (Grant Nos. 0018/2019/AKP, SKL-IOTSC(UM)-2021-2023), Guangdong Science and Technology Department (Grant Nos. 2018B030324002, 2020B1515130001), Zhuhai Science and Technology Innovation Bureau (Grant No. ZH22017002200001PWC), Jiangsu Science and Technology Department (Grant No. BZ2021061), University of Macau (Grant No. MYRG2020-00253-FST). The authors would like to thank anonymous reviewers for their valuable comments.

References

1 Ding M, Han X, Wang S, et al. A thermomechanical material point method for baking and cooking. ACM Trans Graph, 2019, 38: 1–14

2 Wolper J, Fang Y, Li M, et al. CD-MPM: continuum damage material point methods for dynamic fracture animation. ACM Trans Graph, 2019, 38: 1–15

3 Wolper J, Chen Y, Li M, et al. AnisoMPM: animating anisotropic damage mechanics. ACM Trans Graph, 2020, 39: 1–16

4 Müller M, Heidelberger B, Hennix M, et al. Position based dynamics. J Visual Commun Image Represent, 2007, 18: 109–118

5 Bender J, Koschier D, Charrier P, et al. Position-based simulation of continuous materials. Comput Graph, 2014, 44: 1–10

6 Bouaziz S, Martin S, Liu T, et al. Projective dynamics: fusing constraint projections for fast simulation. ACM Trans Graph, 2014, 33: 1–11

7 Cetinaslan O. Parallel XPBD simulation of modified morse potential-an alternative spring model. In: Proceedings of Eurographics Symposium on Parallel Graphics and Visualization, 2019

8 Bonet J, Lok T S. Variational and momentum preservation aspects of smooth particle hydrodynamic formulations. Comput Methods Appl Mech Eng, 1999, 180: 97–115

9 Bender J, Müller M, Otaduy M A, et al. A survey on position-based simulation methods in computer graphics. Comput Graphics Forum, 2014, 33: 228–251

10 Bender J, Müller M, Macklin M. Position-based simulation methods in computer graphics. In: Proceedings of Eurographics (tutorials), 2015. 8

11 Macklin M, Müller M. Position based fluids. ACM Trans Graph, 2013, 32: 1–12

12 Macklin M, Müller M, Chentanez N, et al. Unified particle physics for real-time applications. ACM Trans Graph, 2014, 33: 1–12

13 Müller M, Chentanez N, Kim T Y, et al. Strain based dynamics. In: Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2015. 149–157

14 Macklin M, Müller M, Chentanez N. XPBD: position-based simulation of compliant constrained dynamics. In: Proceedings of the 9th International Conference on Motion in Games, 2016. 49–54

15 He X, Wang H, Wu E. Projective peridynamics for modeling versatile elastoplastic materials. IEEE Trans Visual Comput Graph, 2017, 24: 2589–2599

16 Ihmsen M, Orthmann J, Solenthaler B, et al. SPH fluids in computer graphics. In: Proceedings of the 35th Annual Conference of the European Association for Computer Graphics, 2014

17 Desbrun M, Gascuel M P. Smoothed particles: a new paradigm for animating highly deformable bodies. In: Proceedings of the Eurographics Workshop on Computer Animation and Simulation'96. Berlin: Springer-Verlag, 1996. 61–76

18 Solenthaler B, Schläfli J, Pajarola R. A unified particle model for fluid-solid interactions. Comp Anim Virtual Worlds, 2007, 18: 69–82

19 Becker M, Ihmsen M, Teschner M. Corotated SPH for deformable solids. In: Proceedings of the 5th Euro-graphics Conference on Natural Phenomena, 2009. 27–34

20 Gerszewski D, Bhattacharya H, Bargteil A W. A point-based method for animating elastoplastic solids. In: Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2009. 133–138

21 Jones B, Ward S, Jallepalli A, et al. Deformation embedding for point-based elastoplastic simulation. ACM Trans Graph, 2014, 33: 1–9

22 Islam M R I, Peng C. A stabilized total-Lagrangian SPH method for large deformation and failure in geomaterials. 2019. ArXiv:1907.06990

23 Shutov A, Klyuchantsev V. On the application of SPH to solid mechanics. J Phys-Conf Ser, 2019, 1268: 012077

24 Müller M, Keiser R, Nealen A, et al. Point based animation of elastic, plastic and melting objects. In: Proceedings of Symposium on Computer Animation, 2004

25 Martin S, Kaufmann P, Botsch M, et al. Unified simulation of elastic rods, shells, and solids. ACM Trans Graph, 2010, 29: 1–10

26 Chen X S, Li C F, Cao G C, et al. A moving least square reproducing kernel particle method for unified multiphase continuum simulation. ACM Trans Graph, 2020, 39: 1–15

27 Gilles B, Bousquet G, Faure F, et al. Frame-based elastic models. ACM Trans Graph, 2011, 30: 1–12

28 Faure F, Gilles B, Bousquet G, et al. Sparse meshless models of complex deformable solids. ACM Trans Graph, 2011, 30: 1–10

29 Luo R, Xu W, Wang H, et al. Physics-based quadratic deformation using elastic weighting. IEEE Trans Visual Comput Graphics, 2018, 24: 3188–3199

30 Irving G, Teran J, Fedkiw R. Invertible finite elements for robust simulation of large deformation. In: Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2004. 131–140

31 Teran J, Sifakis E, Irving G, et al. Robust quasistatic finite elements and flesh simulation. In: Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2005. 181–190

32 McAdams A, Zhu Y, Selle A, et al. Efficient elasticity for character skinning with contact and collisions. ACM Trans Graph, 2011, 30: 1–12

33 Stomakhin A, Howes R, Schroeder C, et al. Energetically consistent invertible elasticity. In: Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2012. 25–32

34 Smith B, Goes F D, Kim T. Stable Neo-Hookean flesh simulation. ACM Trans Graph, 2018, 37: 1–15

35 Xu H, Sin F, Zhu Y, et al. Nonlinear material design using principal stretches. ACM Trans Graph, 2015, 34: 1–11

36 Xu L, He X, Chen W, et al. Reformulating hyperelastic materials with peridynamic modeling. Comput Graph Forum, 2018, 37: 121–130

37 Clyde D, Teran J, Tamstorf R. Modeling and data-driven parameter estimation for woven fabrics. In: Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation, New York, 2017