Simulating Fractures with Bonded Discrete Element Method

Jia-Ming Lu, Chen-Feng Li, Geng-Chen Cao, and Shi-Min Hu, *Senior Member, IEEE*

Abstract—Along with motion and deformation, fracture is a fundamental behaviour for solid materials, playing a critical role in physically-based animation. Many simulation methods including both continuum and discrete approaches have been used by the graphics community to animate fractures for various materials. However, compared with motion and deformation, fracture remains a challenging task for simulation, because the material's geometry, topology and mechanical states all undergo continuous (and sometimes chaotic) changes as fragmentation develops. Recognizing the discontinuous nature of fragmentation, we propose a discrete approach, namely the Bonded Discrete Element Method (BDEM), for fracture simulation. The research of BDEM in engineering has been growing rapidly in recent years, while its potential in graphics has not been explored. We also introduce several novel changes to BDEM to make it more suitable for animation design. Compared with other fracture simulation methods, the BDEM has some attractive benefits, e.g. efficient handling of multiple fractures, simple formulation and implementation, and good scaling consistency. But it also has some critical weaknesses, e.g. high computational cost, which demand further research. A number of examples are presented to demonstrate the pros and cons, which are then highlighted in the conclusion and discussion.

Index Terms—physically based animation, solid material, fracture, fragmentation, particle-based method

1 Introduction

Fragmentation processes are found almost everywhere in our daily life, such as breaking eggs, cutting cheese, and tearing paper. Common to all fracturing phenomena are the significant changes to the geometry, topology and mechanical status of material, which occur simultaneously and make fragmentation one of the most challenging physical problems to simulate in both graphics and engineering communities. A variety of numerical approaches have been developed for simulating fractures and depending on how the mechanical behaviour of material is represented, they can be roughly classified into two categories: the continuum methods and the discrete methods.

Real-world materials are composed of atoms or molecules, which are linked by various microscopic forces to form into different microstructures. The Continuum Methods (CMs) ignore such microstructures and instead treat various materials as different continua, whose mechanical responses are represented by specific force-deformation relations. In the continuum constitutive model, the force and deformation in a material are described by stress and strain respectively, which are sequentially used to express its equilibrium condition as the momentum conservation equation. As strain is defined as the derivative of motion in the continuum model, the associated momentum conservation equation is presented as a Partial Differential Equation (PDE), whose solution can be approximately obtained by using such sophisticated numerical schemes as the Finite Element Method (FEM) [1], the Extended Finite Element Method (XFEM) [2], and the meshfree (or meshless) method. FEM and XFEM both have well established generic formulations, while the meshfree (or meshless) method refers to all numerical schemes that do not rely on a mesh/grid discretization, such as the Moving Least Square (MLS) method [3], the Smoothed-Particle Hydrodynamics (SPH) method [4], and the Material Point Method (MPM) [5]. Despite the distinctness of various CM schemes, they all discretize the material medium in one way or another and subsequently transform its governing PDEs into a set of linear algebraic equations in order to obtain the numerical solution.

The Discrete Methods (DMs), such as the mass-spring method, the Bonded Discrete Element Method (BDEM) [6], and the peridynamics method [7], directly mimic the microstructure of material with a lattice or network configuration consisting of nodes and connectors. The mechanical response of material is thus represented by the interactions between nodes and connectors, whose behaviours follow certain preset rules, e.g. Newton's laws. DMs typically avoid the use of such derivative quantities as strain and stress gradient, while the linear algebraic governing equations are directly established according to the discrete constitutive model. Without additional numerical discretization, the DM formulations are simpler than the CM formulations and they are also easier to implement. However, as standard material testing and characterization are all based on the continuum constitutive model, the model parameters in DMs are often more difficult to determine than the material parameters used in CMs, while the model response of DMs is sometimes sensitive to the model configuration. Such input and output model uncertainties have made DMs less attractive than CMs in engineering applications.

In computer graphics, the modelling of solid materials was once dominated by DMs, especially the mass-spring method [8]. But in the past two decades and with the continuous increase of computing power, a variety of CMs initially developed for engineering simulations have been introduced into and adapted for graphics simulations. Reflected in the most recent literature, CMs are more widely used by the graphics community for the simulation of solid

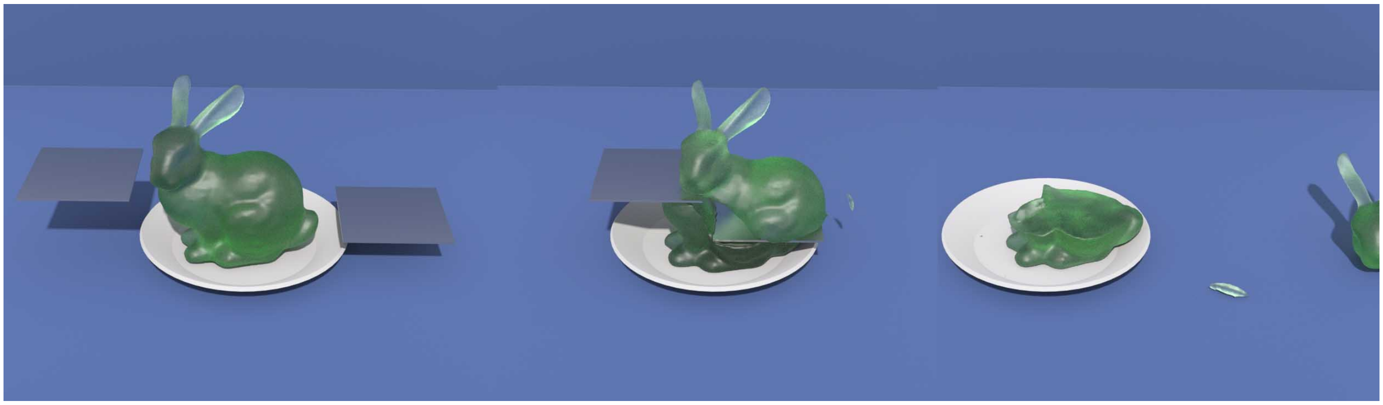


Fig. 1: **Bunny Jell-O.** A bunny Jell-O with 1.26 M discrete elements is cut by two blades, where all bonds are set with the same parameters to simulate an elastic body.

materials with the presence of fragmentation, achieving attractive results for brittle fracture [9] [10] [11] [12], ductile fracture [13] [14], thin-shell objects [15], and rigid bodies [16] etc. Despite these great successes with CMs, it is worth noting that fracturing is by nature a discontinuous process with co-occurrence of time-varying geometry and topology changes to the material medium. As a result, it is always a challenging job to simulate fragmentation processes with CMs, which involve complicated formulation and implantation. Besides, depending on the specific application, the efficiency, robustness and flexibility of fracture simulation using CMs can be limited.

Recognizing the present challenges associated with fracture simulation in graphics, the subject is revisited in this study by using a DM approach, namely the Bonded Discrete Element Method (BDEM). We do not argue to substitute the existing CM approaches (no method is the silver bullet for everything) that have been prevailing in recent years with tremendous success in simulating fragmentation processes in various solids. Instead, our intention is to bring attention to BDEM as an alternative approach to simulating fractures in solid materials by demonstrating its potential benefits as well as current weaknesses, and thereby to stimulate further research in the graphics community.

The BDEM shares a similar discrete configuration as the classic mass-spring model, with the material explicitly represented by a set of connected nodes. In BDEM, the nodes are referred to as discrete elements, which are usually defined as rigid balls, and the nodal connections are referred to as bonds, which are defined between neighbouring elements if and only if they are in direct contact with each other. There are however two fundamental differences between BDEM and the mass-spring model. First, in the mass-spring method, a point mass is an ideal point without volume, and it has 3 Degrees of Freedom (DOF) corresponding to the translational motions in three Cartesian directions. In BDEM, a discrete element is a shaped entity with both mass and volume, and it has 6 DOFs corresponding to both translational and rotational motions. Secondly, in the mass-spring method, many different types of springs are set to handle such mechanical effects as shearing, bending and twisting, and together these springs form into a complex network. As a result, the overall system response is sensitive to the spring network configuration, affecting its flexibility and repeatability. In BDEM, the bonds are defined between adjacent elements if and only if they are in direct contact, and the bond reaction follows standard force-displacement relations that react to relative translational and rotational motions between adjacent elements. As a result, the BDEM configuration is fully determined by the discrete element packing and has good simulation consistency. Our main contributions include:

- The BDEM is introduced for graphics simulation of solid materials with the presence of fractures. The BDEM is simple in formulation and easy to implement. A 99-line C++ code is provided in the Appendix for reference.
- The BDEM is reformulated by using quaternion to represent rotation and set the degrees of freedom exclusively on the elements, which significantly improves the simulation efficiency.
- A BDEM oriented marching cubes algorithm is developed to achieve accurate and robust surface reconstruction for complex fragmentation processes without scarifying the computational efficiency.
- We explore the model settings to simulate a range of fragmentation processes including cutting, breaking, cracking, crumbling and impacting etc. and in a range of materials including granular matter, elastic body, stiff thin-shell, anisotropic material and woven textile etc. The results show that the proposed BDEM approach is extremely versatile in simulating fractures.

2 Related Works

The research works related to fracture simulation in computer graphics are reviewed in this section. As summarized in Table 1, we divide the simulation approaches into two groups: continuum and discrete methods, which are briefly recapped in § 2.1 and § 2.2, respectively.

2.1 Continuum Methods

For the simulation of solid materials, continuum methods have been introduced into the graphics community for at

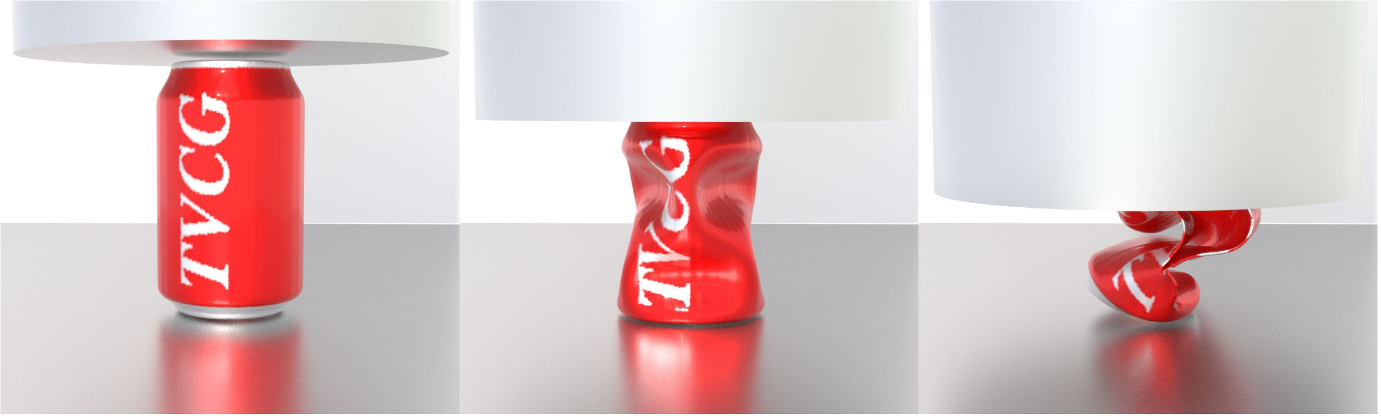


Fig. 2: **Coke Can.** A coke can wiht 495 K discrete elements is pressed to buckle by a hydraulic press, where the thin-shell wall is simulated with three layers of elements.

TABLE 1: Fracture simulation methods in computer graphics.

Types	Methods	Mesh-based	Need remeshing	Fracture Type[1]	Special Materials	Representative References
CMs	FEM	✓	✓	B&D	multi-layer thin sheet	[9] [13]
	XFEM	✓		B&D	thin-shells	[17] [18]
	BEM	✓	✓	B	low-resolution mesh	[11] [19]
	MLS			B&D	melting objects	[10] [20]
	MPM			B&D	snow,foam,wet sand	[21] [22] [23]
DMs	Mass-spring	✓		B	cloth	[24] [25]
	Peridynamics			B	glass	[26] [27]
	BDEM	✓		B	rope,cloth,glass	None

[1] B indicates brittle fracture and D indicates ductile fracture.

least three decades, and viscoelasticity, plasticity and fractures are among the early physical phenomena attempted [28]. Seminar works were presented in [9] [13], where the fragmentation processes in both brittle and ductile materials are simulated with FEM aided by adaptive remeshing. These early successes promoted the popularity of FEM in fracture simulation, and further developments have been made in realtime simulation [29] [30], realtime interactive applications [31], rigid body interaction [16], and thin plates [15]. For graphics applications, the most demanding challenge faced by FEM fracture simulation is adaptive remeshing to capture discontinuities caused by crack propagation, which if not appropriately handled can catastrophically affects the simulation efficiency and robustness. A number of highly effective but application-dependent approaches have been proposed to address this problem [32] [33] [34] [35] [36] [37].

Instead of capturing the fracture propagation with adaptive remeshing, another way to represent the stress field evolution caused by fragmentation is through element enrichment using specially designed shape functions and the corresponding technique is generally referred to as XFEM in engineering [38]. Enrichment textures on regular 2D grids were used by [17] as enrichment basis functions to simulate cutting and fracturing of thin-shells. Discontinuities are represented as edge path between texture pixels and highly detailed crack path can be produced using low-resolution simulation meshes. An XFEM-based cutting algorithm was proposed by [18] with fully implicit interaction using shifted sign enrichment and specialized quadrature rules for discontinuous integrands.

The Boundary Element Method (BEM) is another well established CM approach for solving linear partial differential equations in the boundary integral form, and it has also been successfully applied in graphics for fracture simulation. BEM-based rigid body fracture simulation was presented by [11] with only surface representations. Low-resolution simulation mesh with high-resolution crack-front was used by [19] to produce detailed fracture surfaces with low computational cost. The efficiency is further improved in their later work [12] with a fast approximation method. However, owing to the boundary integral formulation requirement, BEM is generally limited to linear elasticity and brittle fracture.

Besides the grid-based methods, particle-based CM approaches have also been used in computer graphics to simulate fractures in various solid materials. Early attempts includes MLS methods to simulating melting objects [10] and dynamic sampling for fracture of a wide range of materials [20]. The most successful approaches in this category is MPM [21] [22] [23] [14]. A main focus for MPM fracture simulation has been on the integration of multiple physics phenomena, such as fractures in snow [21], foam [22], and wet sand [23]. More recently, fracture with large elastoplastic deformation was animated by the phase-field method based on continuum damage mechanics [14].

2.2 Discrete Methods

The mass-spring model is among the earliest physics-based models that became successful in computer graphics, and it remains widely used in many commercial systems thanks

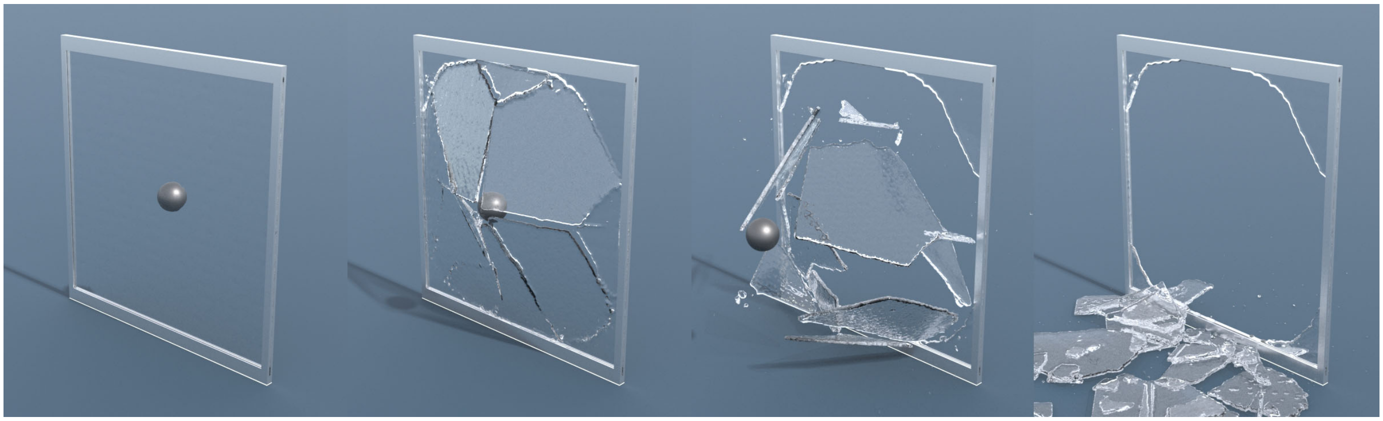


Fig. 3: **Window Breaking.** A glass panel with 521 K discrete elements is penetrated by a bullet, where the glass panel is modelled with five layers of elements and the Young's modulus is set as $E = 10^9$ Pa to simulate thin stiff materials.

to its simplicity and low computational cost. In a mass-spring system, the material is represented by a collection of point masses and mass-less springs, and the material behaviour is determined by the mechanical response of the mass-spring network according to Newton's laws. One of the first successful applications of the mass-spring model is to simulate brittle fracture in an elastic material [8]. Rigid constraints was also applied for fast simulation of the shattering of brittle objects with implicit time integration [39]. Nevertheless, its success is perhaps most noticeable in cloth simulation [24] [25]. Despite the simple formulation and high efficiency, the method also suffers from some critical drawbacks: 1) the model parameters are difficult to set for representing real-world materials, especially when specialized springs are introduced to handle bending and twisting effects; 2) the system behaviour is sensitive to its network configuration, and even becomes unpredictable in some situations.

The Discrete Element Method (DEM) was first proposed by [40] [41] for rock mechanics, and over the years it has become the standard simulation tool for granular matters in engineering. The DEM has also been used by the graphics community to simulate granular materials [42] [43] [44] [45]. Owing to the intrinsic advantages of DEM in dealing with discontinuities, it has been extended to establish the so-called Bonded DEM [6] (BDEM) for the simulation of continuum media. Evidenced by the rapid growth of Itasca's commercial package PFC [46], the BDEM has been increasingly successful in many engineering applications where CMs fail or become too expensive to run. However, there is no exploration of BDEM in graphics applications.

Based on integral equations, peridynamics is a relative new DM approach that is specially designed to simulate deformation with the presence of discontinuities [7]. Its use in computer graphics has also been explored in recent years for brittle fracture [26] and for modeling elastoplastic materials [27]. The key concept of peridynamics is to use integral formulation to avoid the difficulties of computing derivatives at discontinuities. Similar to BDEM, peridynamics also uses points and bonds to simulate continuum solids. A major difference between BDEM and peridynamics in geometry representation is that BDEM stores positions and rotations on elements but peridynamics uses only positions. As a result, peridynamics requires more bonds to model bending and twisting behaviours and it is not straightforward for the bond-based peridynamics to simulate manifolds like rope and cloth.

Researchers have also used rod element to model rope. A Discrete Differential Geometry (DDG) approach was used to compute rod curvatance at point [47]. Cosserat rods was used to simulate rod behaviors which can efficiently model complex bending and torsion effects [48]. This work was later combined with projective dynamics to achieve mesh independence [49]. Cosserat rods were also used to model fibers inside wood [50]. These methods discretize the solid object into rod elements, which differ fundamentally from the discrete element used in BDEM. A rod element consists of a line segment with two points and a quaternion, and it can deform in the form of shearing, stretching, bending and twisting. The deformation of rod element is measured by strain, which is defined following the theory of curves. Strain measures for stretching and shearing are defined by using the difference between tangent vector and normal of rod cross section. Strain measures for bending and twisting are defined on points by using rotations of connected lines by Darboux vector of the curve. Due to the close integration with the theory of curves, the cosserat rods are extremely efficient for modelling slender structures, e.g. a rope, but they are far less effective in describing generic 2D or 3D objects. Used in BDEM, discrete elements are small rigid balls, which can have arbitrary displacements and rotations, and neighbouring discrete elements are connected with bonds. The object deformation is uniformly determined by the states of discrete elements, i.e. their relative translations and rotations, and there is no need to distinguish different deformation modes (e.g. stretching, shearing, bending and twisting) and there is no need to define strains. Compared to the rod element methods, BDEM is equally applicable to generic objects, regardless of their shapes and dimensions. In addition, for a 1D rope, the number of connections and points are similar, but for 2D or 3D objects, the number of connections is usually much more than the number of points. Thus, storing rotation variables on line segments will result in higher computational cost, and it is more cost effective to store the motion states on the points instead of on the connections. Another related work is [51], and similar

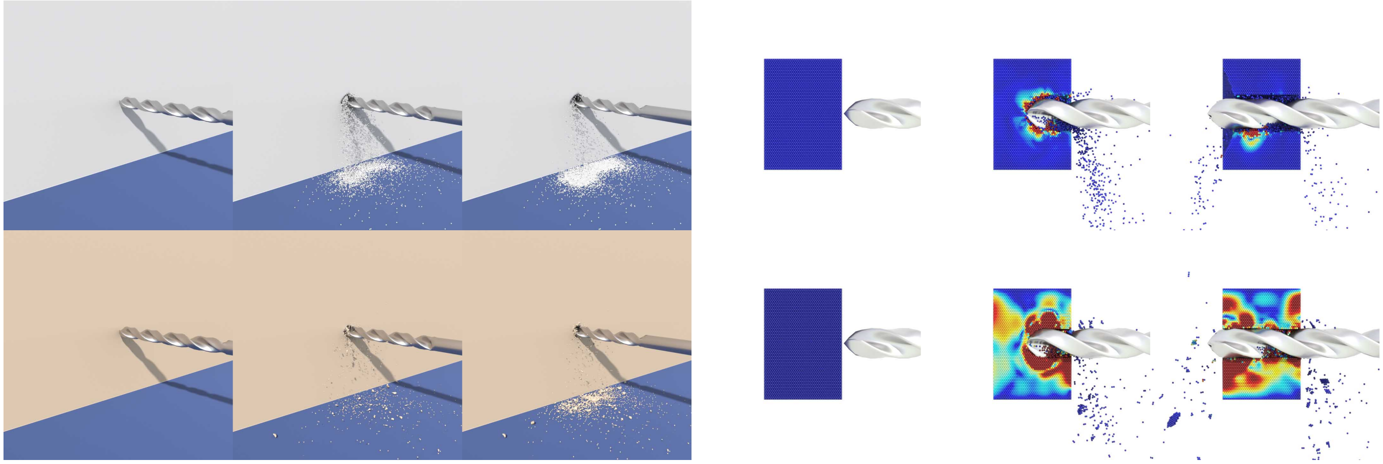


Fig. 4: **Drilling.** An isotropic plaster board and an anisotropic wood sheet are drilled horizontally. These two boards have the same packing with 321 K discrete elements. The bond parameters for the plaster board are set as constants independent from directions, while they vary for the wood sheet dependent on the orientation.

to BDEM it uses particles with orientation to simulate object deformations. But this method is a pure geometric approach based on shape matching, and it does not involve such physical concepts as momentum and stress, which limit its ability to achieve physically realistic effects.

3 Bonded Discrete Element Model

In this section, we explain the BDEM approach for the simulation of solid materials with the presence of fractures. To make BDEM fit for graphics applications, we also introduce several novel changes to improve its simulation performance and to capture the detailed geometric features during fragmentation processes.

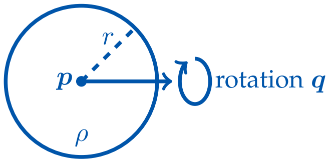


Fig. 5: Discrete element e

In BDEM, a solid material is represented by a collection of connected discrete particles. The discrete particles are termed as discrete elements, and they usually take the form of rigid balls. As shown in Figure 5, a discrete element e can be described by two fixed properties, radius r and density ρ, and two state parameters, position $\boldsymbol{p}$ and rotation $\boldsymbol{q}$. The proposed BDEM framework comprises three technical parts: a bond model to represent the deformation of solid materials, a bond fracture model to capture the fragmentation process, and a contact model to account for collisions between material parts, and they are explained in § 3.1, § 3.2 and § 3.3, respectively.

3.1 Bond Model

In BDEM, the bond model describes the behaviour of the inter-element connections. Using a bond model, the force and torque in the material are calculated based on the position and rotation of discrete elements. The torque is typically accumulated by computing the rotation increment from the angular velocity [6], for which an external increment vector needs to be stored on bonds. Another way to compute the torque is through Euler angles [52], whose formulation and implementation are more involved and due to the accumulation of numerical errors, the results can vary depending on the rotation order selection. To avoid the cost of additional storage and to improve the accuracy and efficiency, we establish the bond model by using quaternion to represent the rotation of discrete element. Compared with other rotation representation methods such as rotation matrix and Euler angles, the quaternion representation has the advantages of lower computational cost, smoother interpolation and avoiding gimbal lock.

For the i-th discrete element, let r_i denote its radius and ρ_i material density, and its state is defined as

$$\boldsymbol{s}_i = (\boldsymbol{p}_i, \boldsymbol{q}_i), \tag{1}$$

where $\boldsymbol{p}_i$ represents the position of the discrete element i, and $\boldsymbol{q}_i$ is a unit quaternion representing the rotation of the discrete element i. As shown in Figure 7, two discrete elements i and j are connected by a bond, whose state is fully determined by the element states $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$. As no dynamic data are stored on bonds, the DOFs of the system is only related to the discrete elements $i = 1, \cdots, N$. In the bond model, a bond has two material properties: Young's modulus E and shear modulus G. Note that the bond parameters here are microscopic parameters, and they are not necessarily identical to the macroscopic material properties. Indeed, the accurate determination of microscopic BDEM parameters is still a challenge in engineering applications, and usually requires calibrations. In a classical experiment of BDEM [6], the micro-macro difference of Young's modulus is found to be 2.7-5.0%. Some recent studies of BDEM calibrations [53], [54], [55] show that using macroscopic parameters as bond parameters causes an error of 22-38%. However, the requirement for accuracy is very different in graphics applications. In our experiment, directly using macro parameters have yielded satisfactory macroscopic results. Detailed in § 5.1, a three-point bending test shows

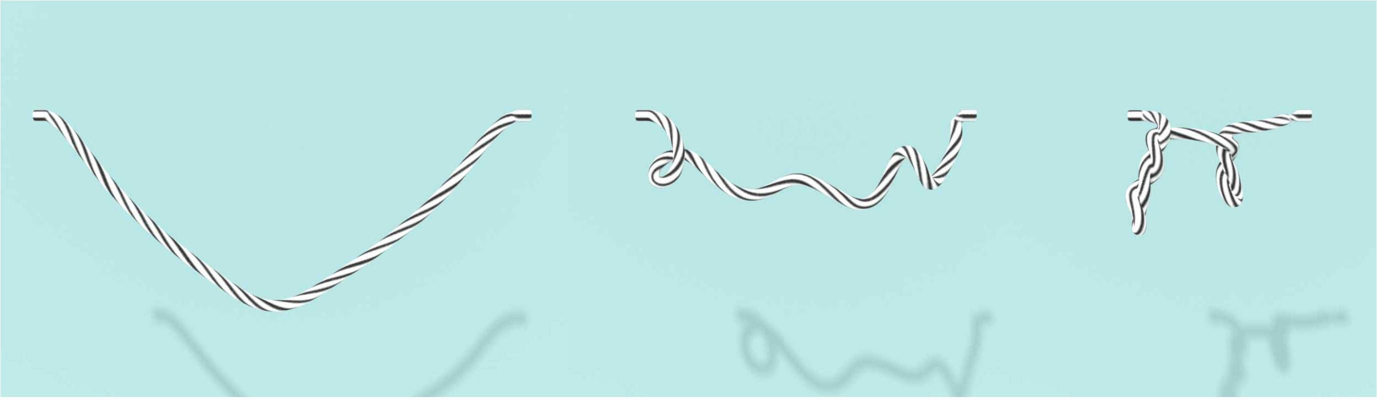


Fig. 6: **Rope.** A rope with 78 K elements is continuously twisted, where a simple rope-shaped element packing is sufficient for our model to correctly handle the associated complex bending and torsion behaviors.

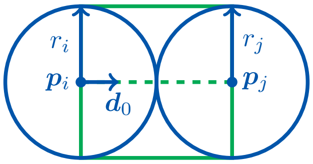


Fig. 7: Two discrete elements connected by a bond.

that the error between micro and macro parameters is less than 28%. Therefore, based on the literature studies and our own tests, we simply treat the bond parameters as macroscopic material properties.

Other bond parameters include the cross section radius r_0, the relaxed length l_0 and the initial direction $\boldsymbol{d}_0$, and they are all determined by the element states as expressed below:

$$r_0 = \frac{r_i + r_j}{2}, l_0 = r_i + r_j, \boldsymbol{d}_0 = \frac{\boldsymbol{p}_j - \boldsymbol{p}_i}{|\boldsymbol{p}_j - \boldsymbol{p}_i|}. \tag{2}$$

A bond has four deformation modes: stretch, shear, and bend and twist, which are detailed in § 3.1.1, § 3.1.2 and § 3.1.3, respectively.

3.1.1 Stretch

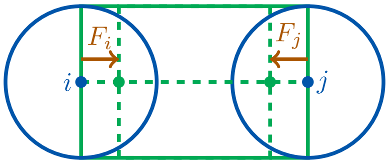


Fig. 8: A stretched bond.

As shown in Figure 8, when the bond is stretched (or shortened), the bond behaves like an elastic spring, and the resulting force will pull (or push) the bond back to its relaxed state. The bond stiffness is k_n is given by

$$k_n = \frac{ES}{l_0}, \tag{3}$$

where E is the Young's modulus of bond and $S = \pi r_0^2$ is the cross section area of the bond. The stretching force is given by:

$$\boldsymbol{F}_{\text{stretch}} = k_n(l - l_0)\boldsymbol{n}. \tag{4}$$

where $l = |\boldsymbol{p}_i - \boldsymbol{p}_j|$ is the current bond length and $\boldsymbol{n} = (\boldsymbol{p}_i - \boldsymbol{p}_j)/l$ the normal direction.

3.1.2 Shear

A pure shear deformation mode is shown in Figure 10, where the discrete element j moves away from its original position (dashed line), creating a shear deformation in the bond (i, j). It is noted that for both discrete elements i and j, their cross-section directions remain unchanged. This situation is geometrically equivalent to the case where discrete elements i and j are under the same self-rotation. In an arbitrary deformation status, the discrete elements i and j can rotate differently, and as a result the two cross-sections of elements i and j can have different directions:

$$\boldsymbol{d}_i = \boldsymbol{q}_i \otimes \boldsymbol{d}_0 \otimes \overline{\boldsymbol{q}_i}, \boldsymbol{d}_j = \boldsymbol{q}_j \otimes \boldsymbol{d}_0 \otimes \overline{\boldsymbol{q}_j}, \tag{5}$$

where $\boldsymbol{d}_i$ denotes the direction of bond at the cross-section of element i, $\boldsymbol{d}_j$ the direction of bond at the cross-section of element j, $\boldsymbol{q}_i$ the rotation quaternion of element i, $\boldsymbol{q}_j$ the rotation quaternion of element j, $\overline{\boldsymbol{q}_i}$ the conjugate of $\boldsymbol{q}_i$, $\overline{\boldsymbol{q}_j}$ the conjugate $\boldsymbol{q}_j$, and $\otimes$ the Hamilton product for quaternions. Thus, the shear direction $\boldsymbol{d}_{ij}$ and the tangential unit vector $\boldsymbol{t}$ can be calculated as:

$$\boldsymbol{d}_{ij} = \frac{\boldsymbol{d}_i + \boldsymbol{d}_j}{2}, \tag{6}$$

and

$$\boldsymbol{t} = \frac{\boldsymbol{d}_{ij} - (\boldsymbol{d}_{ij} \cdot \boldsymbol{n})\boldsymbol{n}}{|\boldsymbol{d}_{ij} - (\boldsymbol{d}_{ij} \cdot \boldsymbol{n})\boldsymbol{n}|}. \tag{7}$$

Hence, the shear angle is

$$\theta_{ij} = \langle \boldsymbol{d}_{ij}, \boldsymbol{n} \rangle. \tag{8}$$

Therefore, when a bond is sheared, the corresponding forces and moments can be calculated as follows:

$$\boldsymbol{F}_{shear} = k_s sin(\theta_{ij})\boldsymbol{t}, \tag{9}$$

$$\boldsymbol{M}_{shear} = \frac{l}{2}\boldsymbol{n} \times \boldsymbol{F}_s, \tag{10}$$

where the shear stiffness is determined by the shear modulus G of bond and its cross section area S:

$$k_s = GS. \tag{11}$$

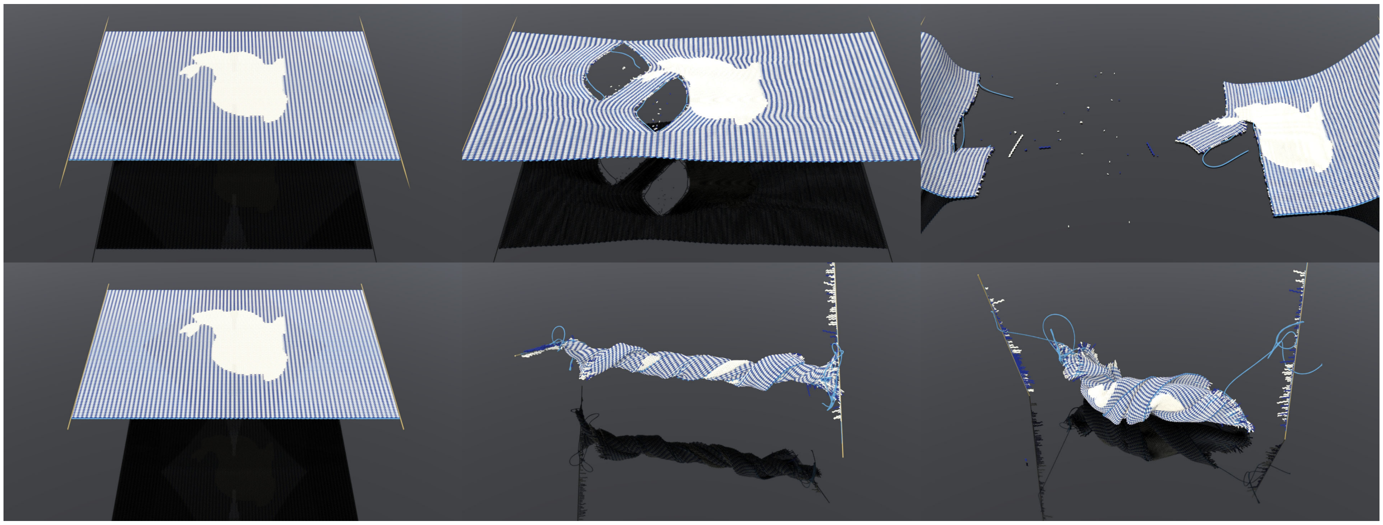


Fig. 9: **Textile.** A textile with 120 K discrete elements is torn and twisted into pieces, where the element packing is constructed with warp and weft to closely capture the behaviour of textile.

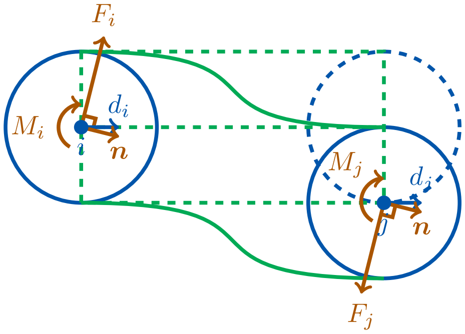


Fig. 10: A sheared bond.

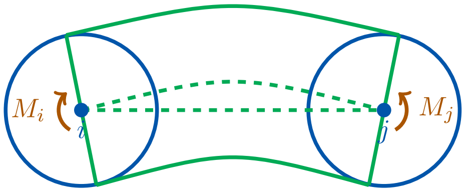


Fig. 11: A bended bond.

3.1.3 Bend and Twist

As shown in Figure 11, by removing the rotations with shear deformation which have same cross-section direction, bend and twist of the bond can be extracted as

$$\boldsymbol{q}_{ij} = \boldsymbol{q}_j \otimes \overline{\boldsymbol{q}_i}. \tag{12}$$

It can also be expressed as a rotation axis $\boldsymbol{a}$ and a rotation angle θ:

$$\boldsymbol{a} = \frac{\Im(\boldsymbol{q}_{ij})}{|\Im(\boldsymbol{q}_{ij})|}, \theta = 2 \arccos \Re(\boldsymbol{q}_{ij}), \tag{13}$$

where $\Im(\boldsymbol{q}_{ij})$ denotes the imaginary part of $\boldsymbol{q}_{ij}$, and $\Re(\boldsymbol{q}_{ij})$ the real part. The twisting and bending moments are

$$\boldsymbol{M}_t = k_t(\theta \boldsymbol{a} \cdot \boldsymbol{n})\boldsymbol{n}, \tag{14}$$

$$\boldsymbol{M}_b = k_b(\theta \boldsymbol{a} - (\theta \boldsymbol{a} \cdot \boldsymbol{n})\boldsymbol{n}), \tag{15}$$

with the twisting stiffness and bending stiffness defined as:

$$k_t = \frac{GJ}{l_0}, k_b = \frac{EI}{l_0}, \tag{16}$$

where $I = \pi r_0^4/4$ and $J = \pi r_0^4/2$ are the second moments of area for bending and twisting, respectively.

3.2 Fracture Model and Fracture Surface Visualization

In BDEM, the fragmentation is represented by the breakage of bond, which typically follows a stress-based criterion [6]. Specifically, the tensile and shear stresses in the bond are calculated as:

$$\sigma = \frac{|\boldsymbol{F}_n|}{S} + \frac{|\boldsymbol{M}_b| r_0}{I}, \tag{17}$$

$$\tau = \frac{|\boldsymbol{F}_s|}{S} + \frac{|\boldsymbol{M}_t| r_0}{J}. \tag{18}$$

When the stress in the bond exceeds the corresponding strength, i.e. $\sigma > \sigma_c$ or $\tau > \tau_c$ with σ_c and τ_c denoting the tensile and shear strengths, respectively, the bond breaks. To represent the natural variability of real-world materials, some randomness is introduced to the bond strength following the Weibull distribution.

The BDEM simulation does not explicitly capture the fracture surface, and hence a separate surface reconstruction step is needed for visualization. Given the particle-based configuration of BDEM, a natural option for surface reconstruction is to follow the idea of the marching cubes algorithm [56], which works well for point cloud data. A direct application is to treat the positions of discrete elements as point cloud data and apply marching cubes to extract fracture surfaces. But as the fragmentation process tends to create closely located fracture surfaces, large amount of tiny fragments, and separate surfaces that are in contact during collision, severe defects are observed with the direction application of marching cubes algorithm. A schematic illustration of some failure cases is given in Figure 13, where the marching cubes algorithm cannot correctly capture the surfaces during fracture propagation (Figure 13(a)) and

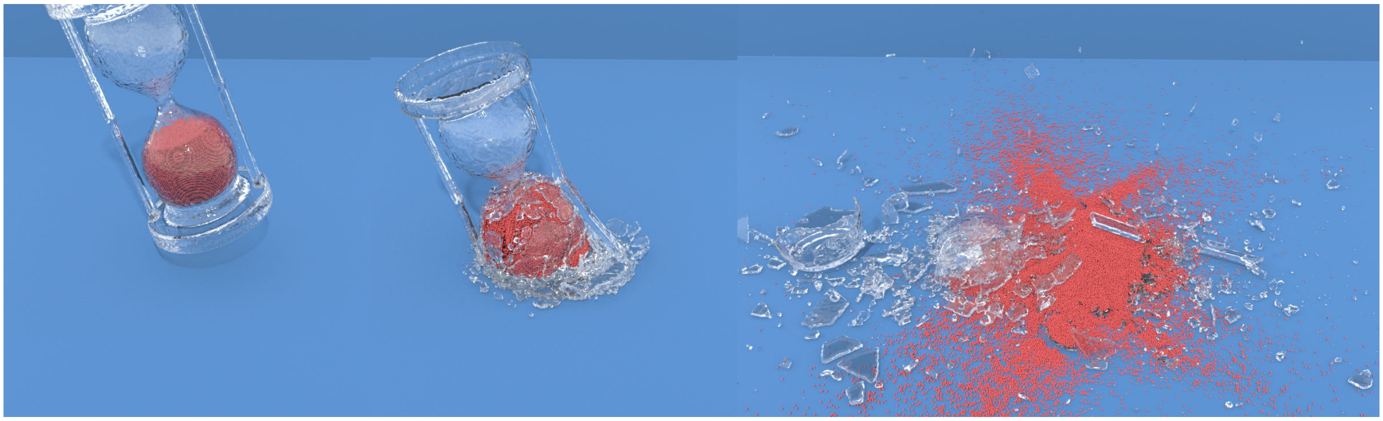
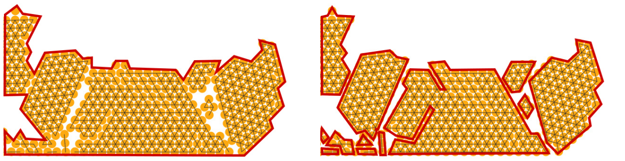


Fig. 12: **Hourglass.** A hourglass with 966 K discrete elements falls onto the ground and breaks into pieces, showing that our method can robustly handle the coupling between granular matter and solid objects.



(a)

(b)

(c)

(d)

Fig. 13: Comparison of fracture surface capture. Yellow dots indicate discrete elements, thin black segments indicate bonds, and thick red lines indicate reconstructed surfaces. (a) Closely located fracture surfaces are missed by the naive marching cubes algorithm. (b) Closely located fracture surfaces are correctly captured by our visualization algorithm. (c) Wrong surfaces are generated by the naive marching cubes algorithm for separate objects contacting with each other. (d) Correct surfaces are generated by our visualization algorithm for separate objects contacting with each other.

during collision of discrete elements (Figure 13(c)). Hence, a BDEM oriented marching cubes algorithm is proposed to address the defects of surface reconstruction.

Fracture Points Instead of using positions of elements, we use fracture points to form the implicit surface. We assume that the bond does not disappear when it breaks, but is divided into two segments and remains on the discrete element. Each fracture point comprises a fracture position $\boldsymbol{p}_f$ of the bond and a vector $\boldsymbol{n}_f$ representing the direction of cross section of bond, which can also be seen as the direction of fracture surface at $\boldsymbol{p}_f$. As shown in Figure 15a, broken bonds are colored in red, fracture points are colored in black, and the dashed line indicates the fracture surface formed by fracture points.

Grouped SDFs To prevent close but disconnected elements from sticking together like in Figure 13(c), we use the topology information to extract fracture-labeled signed distance functions. A Depth-First Search (DFS) is performed to group elements into different connected parts to distinguish different pieces of fragments as shown in Figure 15b. For each fragment k (elements in the same group G_k), we can define a signed distance function (SDF) based on fracture points in G_k:

$$F_k(\boldsymbol{p}) = \frac{\sum_{i\in G_k}(\boldsymbol{p}-\boldsymbol{p}_i)\cdot \boldsymbol{n}_i W_i}{\sum_{i\in G_k} W_i}, \tag{19}$$

where $\boldsymbol{p}$ is position, and the weights W_i are computed using Gaussian function based on the distance between $\boldsymbol{p}$ and the fracture position $\boldsymbol{p}_i$. It is noted that these SDFs are implicitly defined based on fracture points without causing additional storage overhead.

Modified Marching Cubes With grouped SDFs, we adapt the marching cubes procedure to generate triangle surfaces. According to the algorithm, the space is divided into small cubes. We need to calculate the values on the eight vertices of the cube, calculate the triangular faces in the cube, and then move to the next cube and repeat this process. According to our grouped SDF, the distance value needs to be calculated for each group on the vertex, and triangles are calculated separately for each group. As the fracture surface only occupies a small part of the space, we only deal with the cubes near fracture points and calculate the distance values of the corresponding groups, saving calculations. The computational cost of our method is linearly related to the size of the fracture surface and can accurately capture the detailed geometric features. In addition, we also propose a method to deal with fracture surface in the same group. For a certain point $\boldsymbol{p}$ in the space, we divide the fracture points in the neighborhood δ into several subgroups according to the connection relationship in the neighborhood δ, calculate the distance values separately, and take the smallest one as the final distance value, as shown in Figure 15c. Since the point is at least outside of one of the subgroups, the gap detection is guaranteed.

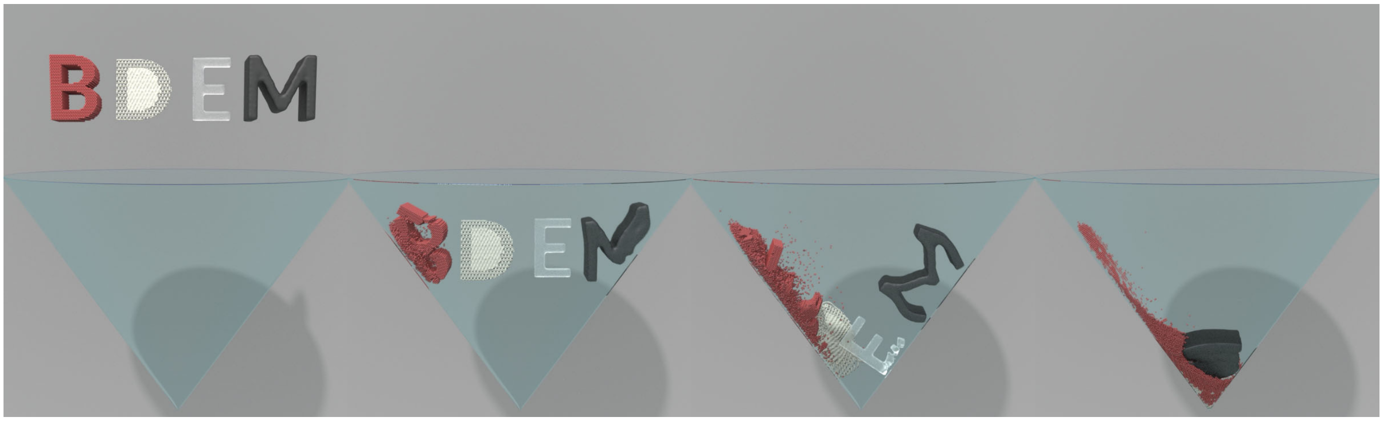


Fig. 14: **Coupling.** The falling of B, D, E and M, which are made from granules, textile, rigid thin sheet, and soft elastic material, respectively, showing that our method can uniformly simulate the behaviour of various materials and the complex coupling with the presence of fractures.

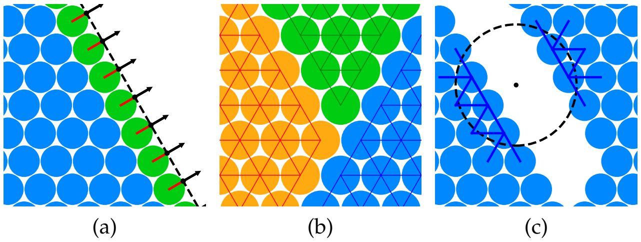

(a) (b) (c)

Fig. 15: The process of our fracture surface visualization method. (a) The black dots with directions indicate the fracture points. (b) Elements are divided into several groups by connection relationship. (c) The fracture surfaces in the same group of elements are correctly processed by our algorithm.

3.3 Contact Model

A classic contact model [57] is used in the proposed BDEM to deal with interactions between separated discrete elements. A small overlap is formed when two separated discrete elements collide with each other and as a result of the collision, the interaction comprises a normal force (repulsion) and a tangential force (friction). The contact model can process uniformly granular-granular, continuum-continuum and granular-continuum couplings.

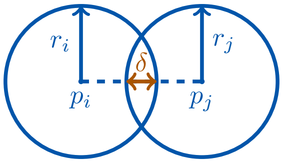


Fig. 16: Two discrete elements in contact.

As shown in Figure 16, two discrete elements i and j are in contact with an overlap

$$\delta_{ij} = (r_i + r_j) - |\boldsymbol{p}_i - \boldsymbol{p}_j|, \tag{20}$$

where r_i and r_j denote their radii, and $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$ their positions. The contact force is decomposed into a normal repulsion force and a tangential friction force:

$$\boldsymbol{F}_{\text{contact}} = \boldsymbol{F}_r + \boldsymbol{F}_f. \tag{21}$$

The normal repulsion force is computed by using a linear elastic model:

$$\boldsymbol{F}_r = k_r \delta \boldsymbol{n}, \tag{22}$$

where k_r is the normal repulsion stiffness and unit normal vector $\boldsymbol{n} = (\boldsymbol{p}_i - \boldsymbol{p}_j)/|\boldsymbol{p}_i - \boldsymbol{p}_j|$.

To model friction, we use an accumulated tangential spring $\boldsymbol{\xi}$ between discrete elements i and j. When two elements start to contact, a tangential spring is created between i and j with initial value $\mathbf{0}$. In the following time step, if i and j are not in contact, we remove the tangential spring. If they are still in contact, as the contact direction can be rotated, we first project the spring to current tangential plane through

$$\boldsymbol{\xi}_t' = \boldsymbol{\xi}_{t-1} - (\boldsymbol{\xi}_{t-1} \cdot \boldsymbol{n})\boldsymbol{n}, \tag{23}$$

where $\boldsymbol{\xi}_{t-1}$ denotes the friction spring in the last time step, and $\boldsymbol{\xi}_t'$ denotes the projected spring. Then we accumulate the tangential spring by

$$\boldsymbol{\xi}_t = \boldsymbol{\xi}_t' + \boldsymbol{v}_t \Delta t \tag{24}$$

where the relative tangential velocity is given by

$$\boldsymbol{v}_t = (\boldsymbol{v}_i - \boldsymbol{v}_j) - ((\boldsymbol{v}_i - \boldsymbol{v}_j) \cdot \boldsymbol{n})\boldsymbol{n}. \tag{25}$$

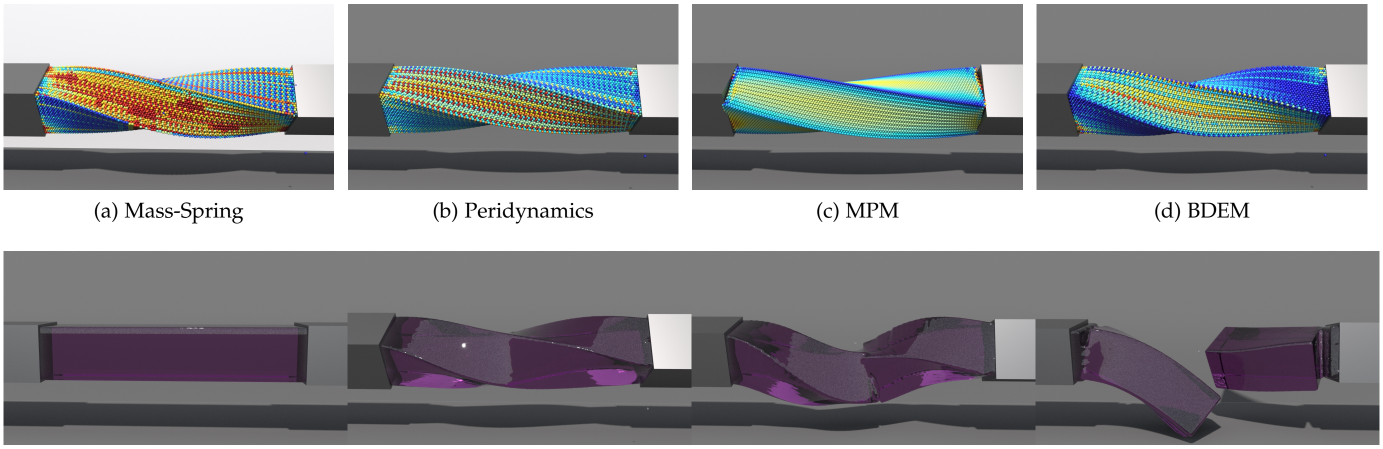
The friction force is first computed by

$$\boldsymbol{F}_f = -k_t \boldsymbol{\xi}, \tag{26}$$

where k_t is the tangential stiffness. According to Coulomb's law of friction, the tangential friction force $\boldsymbol{F}_f$ has relationship with the normal repulsion force $\boldsymbol{F}_r$: for static friction, $|\boldsymbol{F}_f| \leq \mu_s |\boldsymbol{F}_r|$ and for dynamic friction, $|\boldsymbol{F}_f| = \mu_d |\boldsymbol{F}_r|$. The parameters μ_s and μ_d are the coefficients of static and dynamic friction, where $\mu_d \leq \mu_s$ in general. As the tangential spring accumulates, when $|\boldsymbol{F}_f| \geq \mu_s |\boldsymbol{F}_r|$, the dynamic friction is active. Considering the current direction of $\boldsymbol{F}_f$ is $\boldsymbol{t}$, the dynamic tangential spring and dynamic friction force are

$$\boldsymbol{\xi} = -\frac{\mu_d |\boldsymbol{F}_r| \boldsymbol{t}}{k_t}, \boldsymbol{F}_f = \mu_d |\boldsymbol{F}_r| \boldsymbol{t}. \tag{27}$$

For simplicity, we assume $k_r = k_n$ and $\mu_s = \mu_d = \mu$.



(a) Mass-Spring

(b) Peridynamics

(c) MPM

(d) BDEM

(e) Beam twisting with BDEM

Fig. 17: **Beam Twisting.** The first row shows the stress visualization of different methods in a beam twisting test. The second row shows the fragmentation simulation results of BDEM.

4 Model Configuration and Implementation

The formulations for the proposed BDEM approach for fracture simulation are explained in § 3. In this section, we further explain the model configuration, the handling of some numerical issues, and the simulation workflow.

4.1 Model Configuration

The BDEM model represents a solid material with a collection of densely packed discrete elements, where only adjacent elements are connected with bonds. The macroscopic properties of solid materials are determined by the packing configuration of discrete elements and the microscopic parameters of elements and bonds. With respect to the packing configuration, a random close packing with different size of elements is often adopted in engineering applications. The direction bias of this method is smaller, which is conducive to accurate calculations in engineering. But this method requires complicated and time-consuming operations to initialize packing. Our BDEM model does support the use of discrete elements of different sizes as shown in Figure 18, where a three-point bending beam is represented with random close packing of discrete elements at different sizes ranging from 0.9 to 4.0. But for efficiency reasons, we adopt a simple hexagonal closest packing with the same element size in this study. With this packing method, BDEM shows good scaling consistency and material properties can be easily adapted as the simulation resolution refines.

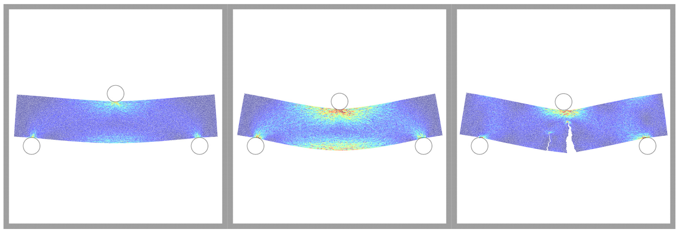


Fig. 18: Bending test with random close packing of different sizes of elements (element size 0.9 to 4.0).

In the BDEM model, each discrete element has two parameters including density ρ and radius r, and each bond has four parameters including Young's modulus E, shear modulus G, tensile strength σ_c and shear strength τ_c. All other algorithmic parameters are readily derived from these six basic parameters and the packing configuration, and they include $m_i, l_0, r_0, S, k_n, k_s, k_t, k_b$. By adjusting the six basic model parameters, the behaviours of a wide range of solid materials can be captured by the proposed BDEM approach. Granular materials can be easily modeled in BDEM without bond connections. Isotropic elastic materials can be modeled with the same parameters set for all bonds, with the material stiffness controlled by E and G and the fragility controlled by σ_c and τ_c. In addition, by purposely setting the bond parameters, BDEM is also capable of simulating complex materials with substructures. More details on parameter setting are given in § 5 in conjunction with the relevant examples.

4.2 Simulation Workflow

The simulation workflow is summarized in Algorithm 1, and a 99-line C++ demo code is provided in Appendix. To update the rotation with time step Δt, the rotation increment is computed as $\boldsymbol{w}\Delta t$ and we use a quaternion $\Delta\boldsymbol{q}$ to represent it. The rotation of element is updated with rotation increment by $\boldsymbol{q} \leftarrow \Delta\boldsymbol{q} \otimes \boldsymbol{q}$. As shown in lines 6-10, we use a velocity verlet scheme to update positions and a similar form for rotations. Like other explicit methods, the time step of BDEM needs to be sufficiently small for simulation stability. We used a method based on eigenvalues to estimate the critical time step [58]. The critical time step has a relation between the largest eigenvalue λ of the system of $T = 2/\sqrt{\lambda}$. In our BDEM method, the largest eigen value of the system is roughly

$$\lambda = \frac{k}{m}N, \tag{28}$$

where m and k represent the effective mass and stiffness respectively, and N is the largest eigen value of the Laplacian matrix of the connections between bonds. With hexagonal

TABLE 2: Model settings

Demo	N	s/frame	Δt	FPS	r	ρ	E	G	σ	τ	μ
Beam Bending	24K	68.7	1.22×10^{-6}	100	0.00125	2.71×10^3	10^8	3×10^7	8×10^6	1×10^7	0.5
Beam Twisting	69K	9.3	3.01×10^{-6}	100	0.0025	2.6×10^3	10^6	4×10^6	2×10^4	2.5×10^4	0.7
Paper	46K	13.4	1.04×10^{-6}	100	0.001	2.6×10^3	10^6	3×10^5	8×10^4	1×10^5	0.7
Rope	78K	21.6	6.35×10^{-6}	100	0.001	5×10^3	10^6	3×10^5	∞	∞	0.3
Coke Can	495K	297.8	1.01×10^{-6}	100	0.0005	5×10^3	10^7	3×10^6	2×10^6	2.5×10^6	0.3
Bunny Jell-O	1.26M	284.8	3.13×10^{-6}	100	0.0006	2.6×10^3	10^6	3×10^5	2.4×10^5	3×10^5	0.7
Drilling	312K	520.4	7.24×10^{-7}	100	0.0005	4×10^3	10^7	3×10^6	2×10^4	2×10^4	0.5
Textile	120K	132.4	6.39×10^{-7}	100	0.001	50	10^5	3×10^4	8×10^5	∞	0.7
Hourglass	966K	706.5	1.01×10^{-7}	480	0.0005	5×10^3	10^9	3×10^8	2×10^6	2.5×10^6	0.5
Window Breaking	521K	734.7	1.01×10^{-7}	480	0.0005	5×10^3	10^9	3×10^8	2.5×10^6	2.5×10^6	0.3
Coupling	547K	106.8	2.02×10^{-6}	100	0.001	5×10^3	10^7	3×10^6	5.6×10^5	7.0×10^5	0.3

packing, $N \approx 9$ in 2D and $N \approx 293$ in 3D. Besides, $k = k_n \approx E$ in 2D and $k = k_n \approx \frac{\pi}{2}Er$ in 3D, in which E is the Young's modulus and r is the average radius of elements. In our implementation, a viscous damping term linearly proportional to the relative velocity is added to help stabilize the simulation, and we found that the time step of 0.2-0.5T is sufficient for stable simulation in all our experiments.

ALGORITHM 1: BDEM Fracture Simulation Workflow

```
Initialize discrete elements with position $\boldsymbol{p}_0$ and
  rotation $\boldsymbol{q}_0$;
foreach element i and neighbor element j do
    Initialize bond $(i, j)$ with length $l_0$, radius $r_0$ and
      direction $\boldsymbol{d}_0$;
end
repeat
    $\boldsymbol{v}(t + \frac{1}{2}\Delta t) \leftarrow \boldsymbol{v}(t) + \frac{1}{2}\Delta t \boldsymbol{M}^{-1}\boldsymbol{F}(t)$ ;
    $\boldsymbol{w}(t + \frac{1}{2}\Delta t) \leftarrow \boldsymbol{w}(t) + \frac{1}{2}\Delta t \boldsymbol{I}^{-1}\boldsymbol{T}(t)$;
    $\boldsymbol{p}(t + \Delta t) \leftarrow \boldsymbol{p}(t) + \Delta t \boldsymbol{v}(t + \frac{1}{2}\Delta t)$;
    compute $\Delta\boldsymbol{q}(t)$ with $\boldsymbol{w}(t + \frac{1}{2}\Delta t)$;
    $\boldsymbol{q}(t + \Delta t) \leftarrow \Delta\boldsymbol{q}(t) \otimes \boldsymbol{q}(t)$ ;
    compute $\boldsymbol{F}_{\text{contact}}(t + \Delta t)$ and $\boldsymbol{T}_{\text{contact}}(t + \Delta t)$;
    compute $\boldsymbol{F}_{\text{bond}}(t + \Delta t)$ and $\boldsymbol{T}_{\text{bond}}(t + \Delta t)$;
    foreach bond pair i, j do
        measure stress $\sigma$, $\tau$ and check breakage;
    end
    $\boldsymbol{F}(t + \Delta t) \leftarrow$
      $\boldsymbol{F}(t+\Delta t) + \boldsymbol{F}_{\text{contact}}(t+\Delta t) + \boldsymbol{F}_{\text{bond}}(t+\Delta t) + \boldsymbol{Mg}$;
    $\boldsymbol{T}(t + \Delta t) \leftarrow$
      $\boldsymbol{T}(t + \Delta t) + \boldsymbol{T}_{\text{contact}}(t + \Delta t) + \boldsymbol{T}_{\text{bond}}(t + \Delta t)$;
    $\boldsymbol{v}(t + \Delta t) \leftarrow \boldsymbol{v}(t + \frac{1}{2}\Delta t) + \frac{1}{2}\boldsymbol{M}^{-1}\boldsymbol{F}(t + \Delta t)\Delta t$;
    $\boldsymbol{w}(t + \Delta t) \leftarrow \boldsymbol{w}(t + \frac{1}{2}\Delta t) + \frac{1}{2}\boldsymbol{I}^{-1}\boldsymbol{T}(t + \Delta t)\Delta t$;
until end of simulation;
```

In addition to choosing suitable time steps, the BDEM simulation workflow can be easily accelerated through parallel computing techniques. Both CPU multi-threading and GPU computing can be used to speed up the simulation, where the calculations occurring at the discrete elements can all be performed in parallel. Besides, we divide the space into a set of uniform cells to speed up neighbourhood searching, which is a commonly used technique for particle-based simulations.

5 Results

To demonstrate the ability of BDEM for fracture modeling, a series of test examples are presented in this section including different types of solid materials and fragmentation processes. We use CUDA/C++ and Taichi programming language [59] to implement our model. The beam bending experiment was carried out on a CPU of Intel Core i7-8700, and the other experiments were carried out on a NVIDIA GEFORCE 1080 Ti GPU. Model parameters are summarized in Table 2.

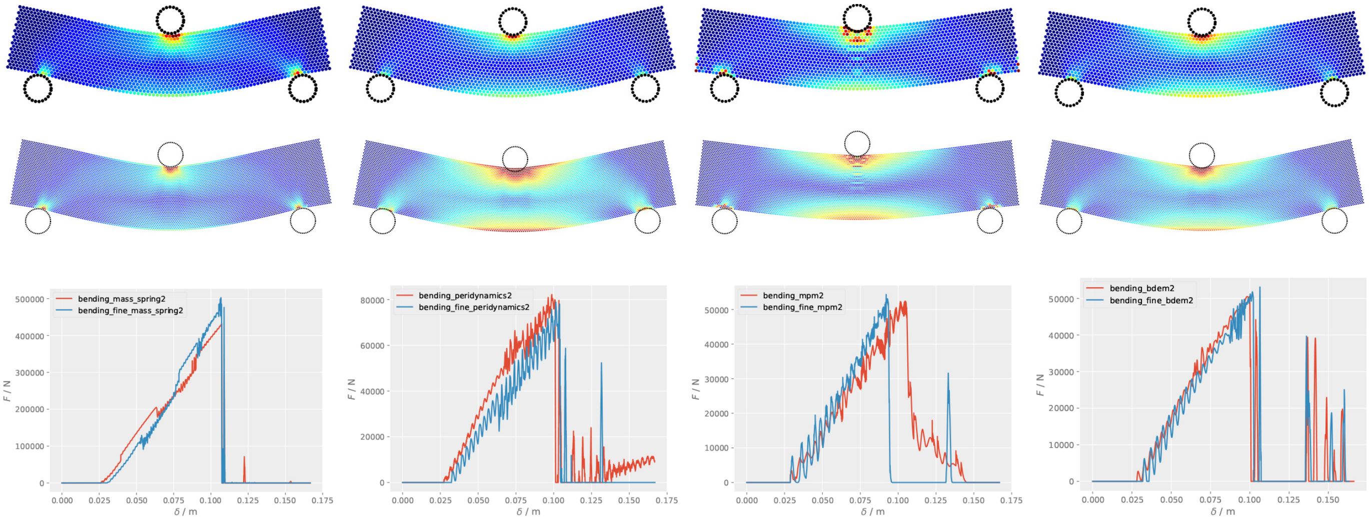
5.1 Comparison: Bending and Twisting

As shown in Figure 19, several different fracture simulation methods including Mass-Spring, Peridynamics, MPM and BDEM are compared via a beam bending test. In these comparisions, we use a classical 2-ring mass-spring system, a bond-based peridynamics method [26] and a local phase-field fracture for MPM described in CDMPM [14]. The first and second row in Figure 19 show the fracture effects obtained using different methods in low and high resolutions, respectively. The load-displacement curves of the beam bending test are plotted in the third row, where the red and blue lines show the results of low and high resolutions, respectively. From the the load-displacement curve, the macroscopic material properties can be calculated as:

$$E_{\text{macro}} = \frac{l^3}{4bh^3}\frac{\Delta F}{\Delta \delta}, \sigma_{\text{macro}} = \frac{3lF_{\text{top}}}{2bh^2} \tag{29}$$

where l, b and h denotes the length, thickness and height of the beam, F denotes the load, and δ is the displacement in the center of the beam. $\frac{\Delta F}{\Delta \delta}$ indicates the slope of the curve, and F_{top} represents the load at the highest point of the curve. The micro- and macro-parameter settings are listed in Table 3 together with the time stepping and calculation efficiency. The experimental results show that the BDEM method has good scaling consistency compared to Mass-Spring and Peridynamics, and the error between microscopic and macroscopic parameters is 10% for Young's modulus and 28% for bending strength. From Table 3, we can also find that Mass-Spring and Peridynamics generally runs faster than BDEM, because they can simulate stiffer materials with the same micro parameters. For BDEM, the microscopic parameters are closer to the macroscopic parameters, so a smaller time step is required to simulate stiff materials.

To further compare the performance of different methods, a 3D beam twisting experiment is conducted as shown



(a) Mass-Spring

(b) Peridynamics

(c) MPM

(d) BDEM(Ours)

Fig. 19: **Beam Bending.** Comparison of different methods in a three-point bending test. The first row shows the stress distribution obtained from a low-resolution simulation, the second row shows the stress distribution obtained from a high-resolution simulation, and the third row shows the load-displacement curves.

TABLE 3: Comparison of different methods in three point bending test.

Method	E_{micro}	G_{micro}	σ_{micro}	E_{macro}	σ_{macro}	error	Δt	T
Mass-Spring (L)	1.0×10^7	-	1.25×10^6	7.9×10^7	1.4×10^7	690%/1020%	2.0×10^{-5}s	1.2s
Mass-Spring (H)	1.0×10^7	-	1.25×10^6	1.0×10^8	1.6×10^7	900%/1180%	1.0×10^{-5}s	4.7s
Peridynamics (L)	1.0×10^7	-	1.25×10^6	1.8×10^7	2.6×10^6	80%/108%	2.0×10^{-5}s	0.8s
Peridynamics (H)	1.0×10^7	-	1.25×10^6	1.7×10^7	2.4×10^6	70%/92%	1.0×10^{-5}s	4.5s
MPM (L)	1.0×10^7	4.0×10^6	1.25×10^6	1.2×10^7	1.6×10^6	20%/28%	1.0×10^{-4}s	0.12s
MPM (H)	1.0×10^7	4.0×10^6	1.25×10^6	1.0×10^7	1.6×10^6	0%/28%	5.1×10^{-5}s	0.58s
BDEM (L)	1.0×10^7	4.0×10^6	1.25×10^6	1.1×10^7	1.6×10^6	10%/28%	2.0×10^{-5}s	1.1s
BDEM (H)	1.0×10^7	4.0×10^6	1.25×10^6	1.1×10^7	1.6×10^6	10%/28%	1.0×10^{-5}s	5.0s

[*] H indicates high resolution and L low resolution.

[**] In Peridynamis, the micro bond parameters can be calculated by macro parameter E. The parameters in the table represent the macro E we used. The real bond parameter is smaller.

[***] The error column shows the error between the macro parameters and the micro parameters of E and σ. The error is calculated by $\frac{A_{\text{macro}} - A_{\text{micro}}}{A_{\text{micro}}}$ where A indicates E or σ.

in Figure 17, where the first row plots the stress distribution obtained by Mass-Spring, Peridynamcis, MPM and BDEM, respectively, and the second row shows the fragmentation visualization of BDEM. The comparison shows the Mass-Spring model generates a noisy and non-physical result, Peridynamics and BDEM models obtain similar stress distributions, and the MPM result appears to be the smoothest and most accurate. Note that the results of both Peridynamics and BDEM are affected by the structured element/point packing, and therefore appearing to be unsymmetrical. At an extra cost of initialization, a random packing scheme will resolve the issue. This is a limitation of the simple hexagonal closest packing scheme adopted in this work.

5.2 Elasticity: Bunny Jell-O

The fragmentation of elastic materials is first tested. Figure 1 shows a bunny-shaped Jell-O cut by two blades. To model the rigid boundaries (e.g. the two blades), we simply sample the boundaries with discrete elements of infinite mass, which allows the rigid boundary conditions to be automatically handled by the built-in collision search algorithm and contact model.

5.3 Thin-shell Objects: Paper, Coke Can and Glass

The second group of tests concern the fracture phenomena in thin-shell objects, which often require specialized mathematical and numerical formulations to simulate when using a CM approach. Thin-shell objects can be readily handled in BDEM by representing the object with a few layers of discrete elements, without the need of any additional treatment or modification. Figure 20 shows two different fragmentation processes of paper sheets with or without a hole in the center. The stress distribution is also visualized. Two more complicated examples are shown in Figure 2, where a coke can is squashed to buckle, and Figure 3, where a glass panel is struck by a bullet.

5.4 Drilling

In this example, we consider a drilling process. As a large amount of small fragments are continuously generated during drilling, its simulation is challenging for both CM and

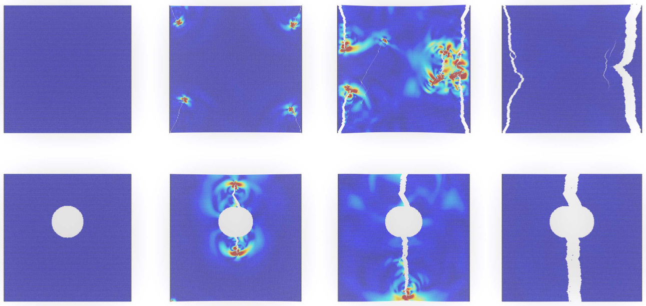


Fig. 20: **Paper.** Two sheets of paper with or without a hole at the center are torn apart. The color indicates the stress inside the paper sheet.

other DM approaches. As shown in Figure 4, an isotropic plaster board and an anisotropic wood board are drilled horizontally, creating dust and tiny fragaments that fall onto the ground. To represent the anisotropy of natural wood, whose stiffness and strength differ in different directions, we simply set the bond parameters differently based on its direction. For example, the Young's modulus of wood is set as

$$E_{\text{wood}} = \left(1 + \lambda \left(\boldsymbol{d}_g \cdot \boldsymbol{d}_b\right)^4\right) E, \tag{30}$$

where $\boldsymbol{d}_g$ is the wood growth direction, $\boldsymbol{d}_b$ is the direction of the bond, λ is a user-specified parameter and E is the Young's modulus in the direction perpendicular to the wood growth.

5.5 Rope and Textile

Rope and textile are flexible materials made by creating an interlocking network of yarns or threads, and their deformation and fracture behaviours differ depending on both the properties of yarns and the weaving patterns. BDEM provides a good framework to model rope and textile at the yarn level. As shown in Figure 6 and Figure 9, the yarns in rope and textile can be easily modelled by using discrete elements and bonds, where the properties of yarns are represented by the parameters of discrete elements and bonds and the weaving patterns are represented by the packing configuration of discrete elements. The complex interaction between yarns are automatically handled by the built-in collision and contact models, without the need of any additional treatment.

5.6 Coupling

The coupling between different types of materials can be easily handled by the proposed BDEM approach, thanks to its flexibility to represent various solid continua and the uniform treatment of interactions. Figure 12 shows a hourglass dropped and broken on the floor, which demonstrates the coupling between granules and solids. Figure 14 shows the interaction between four 3D letters: B, D, E and M, which are made from granules, textile, rigid thin sheet, and soft elastic material, respectively.

6 Conclusion and Discussion

In this work, we propose a BDEM approach to simulating fracture phenomena in solid materials. One of the most attractive advantages of BDEM is its ability to handle multi-fractures involving large number of fragments. Other advantages of BDEM include its simplicity and flexibility. With well-defined and clearly structured bond and contact models, the implementation of BDEM is simple, and it can simulate a wide range of solid materials without the need of any additional treatment. In addition, with the same packing configuration, BDEM exhibits good scaling consistency, which is a useful feature for practical animation design.

However, the BDEM approach is not without limitations, at least in its current form. First, due to the small time step required in the explicit time integration, the computational cost of BDEM is relatively high, and this can be a major hurdle for its practice use. Although the limitation of high computational cost is partially relieved by BDEM's good scaling consistency, further research is needed in order to fundamentally overcome this issue. One possible solution to this problem is to develop implicit schemes with adaptive time stepping. All materials considered in the present work are elastic solids, and it would be an interesting future work to extend the proposed BDEM framework to model plastic solids and to support fluid-solid coupling, which will greatly enhance its animation capacity. For best accuracy, a random packing of discrete elements is generally desirable for BDEM simulation, and therefore further research to efficiently generate randomly packed discrete element models is beneficial.

Acknowledgments

This work was supported by China's National Key Technology R&D Program (Project Number 2017YFB1002701), National Natural Science Foundation of China (Project Number 61521002), Research Grant of Beijing Higher Institution Engineering Research Center and Tsinghua-Tencent Joint Laboratory for Internet Innovation Technology. We would like to thank all the anonymous reviewers for their suggestions.

References

[1] O. C. Zienkiewicz and R. L. Taylor, *The finite element method for solid and structural mechanics*. Elsevier, 2005.

[2] N. Moës, J. Dolbow, and T. Belytschko, "A finite element method for crack growth without remeshing," *International journal for numerical methods in engineering*, vol. 46, no. 1, pp. 131–150, 1999.

[3] D. Levin, "The approximation power of moving least-squares," *Mathematics of computation*, vol. 67, no. 224, pp. 1517–1531, 1998.

[4] R. A. Gingold and J. J. Monaghan, "Smoothed particle hydrodynamics: theory and application to non-spherical stars," *Monthly notices of the royal astronomical society*, vol. 181, no. 3, pp. 375–389, 1977.

[5] D. Sulsky, S.-J. Zhou, and H. L. Schreyer, "Application of a particle-in-cell method to solid mechanics," *Computer physics communications*, vol. 87, no. 1-2, pp. 236–252, 1995.

[6] D. O. Potyondy and P. Cundall, "A bonded-particle model for rock," *International journal of rock mechanics and mining sciences*, vol. 41, no. 8, pp. 1329–1364, 2004.

[7] S. A. Silling and E. Askari, "A meshfree method based on the peridynamic model of solid mechanics," *Computers & structures*, vol. 83, no. 17-18, pp. 1526–1535, 2005.

[8] A. Norton, G. Turk, B. Bacon, J. Gerth, and P. Sweeney, "Animation of fracture by physical modeling," *The visual computer*, vol. 7, no. 4, pp. 210–219, 1991.

[9] J. F. O'Brien and J. K. Hodgins, "Graphical modeling and animation of brittle fracture," in *Proceedings of ACM SIGGRAPH 1999*, 1999, p. 137–146.

[10] 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 the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 2004, pp. 141–151.

[11] Y. Zhu, R. Bridson, and C. Greif, "Simulating rigid body fracture with surface meshes," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 4, pp. 1–11, 2015.

[12] D. Hahn and C. Wojtan, "Fast approximations for boundary element based brittle fracture simulation," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, pp. 1–11, 2016.

[13] J. F. O'Brien, A. W. Bargteil, and J. K. Hodgins, "Graphical modeling and animation of ductile fracture," in *ACM Transactions on Graphics (TOG)*, vol. 21, no. 3, Jul. 2002, p. 291–294.

[14] J. Wolper, Y. Fang, M. Li, J. Lu, M. Gao, and C. Jiang, "Cd-mpm: Continuum damage material point methods for dynamic fracture animation," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 4, pp. 1–15, 2019.

[15] O. Busaryev, T. K. Dey, and H. Wang, "Adaptive fracture simulation of multi-layered thin plates," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 4, pp. 1–6, 2013.

[16] Z. Bao, J.-M. Hong, J. Teran, and R. Fedkiw, "Fracturing rigid materials," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 2, pp. 370–378, 2007.

[17] P. Kaufmann, S. Martin, M. Botsch, E. Grinspun, and M. Gross, "Enrichment textures for detailed cutting of shells," in *ACM SIGGRAPH 2009 papers*, 2009, pp. 1–10.

[18] D. Koschier, J. Bender, and N. Thuerey, "Robust extended finite elements for complex cutting of deformables," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, pp. 1–13, 2017.

[19] D. Hahn and C. Wojtan, "High-resolution brittle fracture simulation with boundary elements," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 4, pp. 1–12, 2015.

[20] M. Pauly, R. Keiser, B. Adams, P. Dutré, M. Gross, and L. J. Guibas, "Meshless animation of fracturing solids," *ACM Transactions on Graphics (TOG)*, vol. 24, no. 3, pp. 957–964, 2005.

[21] A. Stomakhin, C. Schroeder, L. Chai, J. Teran, and A. Selle, "A material point method for snow simulation," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 4, pp. 1–10, 2013.

[22] Y. Yue, B. Smith, C. Batty, C. Zheng, and E. Grinspun, "Continuum foam: A material point method for shear-dependent flows," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 5, pp. 1–20, 2015.

[23] A. P. Tampubolon, T. Gast, G. Klár, C. Fu, J. Teran, C. Jiang, and K. Museth, "Multi-species simulation of porous sand and water mixtures," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, pp. 1–11, 2017.

[24] D. Baraff and A. Witkin, "Large steps in cloth simulation," in *Proceedings of ACM SIGGRAPH 1998*, 1998, pp. 43–54.

[25] T. Liu, A. W. Bargteil, J. F. O'Brien, and L. Kavan, "Fast simulation of mass-spring systems," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 6, pp. 1–7, 2013.

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

[27] X. He, H. Wang, and E. Wu, "Projective peridynamics for modeling versatile elastoplastic materials," *IEEE transactions on visualization and computer graphics*, vol. 24, no. 9, pp. 2589–2599, 2017.

[28] D. Terzopoulos and K. Fleischer, "Modeling inelastic deformation: viscolelasticity, plasticity, fracture," in *Proceedings of ACM SIGGRAPH 1988*, 1988, pp. 269–278.

[29] M. Müller, L. McMillan, J. Dorsey, and R. Jagnow, "Real-time simulation of deformation and fracture of stiff materials," in *Computer Animation and Simulation 2001*, 2001, pp. 113–124.

[30] E. G. Parker and J. F. O'Brien, "Real-time deformation and fracture in a game environment," in *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2009, pp. 165–175.

[31] M. Müller and M. H. Gross, "Interactive virtual materials." in *Graphics interface*, vol. 2004, 2004, pp. 239–246.

[32] M. Wicke, D. Ritchie, B. M. Klingner, S. Burke, J. R. Shewchuk, and J. F. O'Brien, "Dynamic local remeshing for elastoplastic simulation," *ACM Transactions on graphics (TOG)*, vol. 29, no. 4, pp. 1–11, 2010.

[33] N. Molino, Z. Bao, and R. Fedkiw, "A virtual node algorithm for changing mesh topology during simulation," *ACM Transactions on Graphics (TOG)*, vol. 23, no. 3, pp. 385–392, 2004.

[34] E. Sifakis, K. G. Der, and R. Fedkiw, "Arbitrary cutting of deformable tetrahedralized objects," in *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 2007, pp. 73–80.

[35] Z. Chen, M. Yao, R. Feng, and H. Wang, "Physics-inspired adaptive fracture refinement," *ACM Transactions on Graphics (TOG)*, vol. 33, no. 4, pp. 1–7, 2014.

[36] T. Pfaff, R. Narain, J. M. De Joya, and J. F. O'Brien, "Adaptive tearing and cracking of thin sheets," *ACM Transactions on Graphics (TOG)*, vol. 33, no. 4, pp. 1–9, 2014.

[37] Y. Wang, C. Jiang, C. Schroeder, and J. Teran, "An adaptive virtual node algorithm with robust mesh cutting," in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Eurographics Association, 2014, pp. 77–85.

[38] T. Belytschko and T. Black, "Elastic crack growth in finite elements with minimal remeshing," *International journal for numerical methods in engineering*, vol. 45, no. 5, pp. 601–620, 1999.

[39] J. Smith, A. Witkin, and D. Baraff, "Fast and controllable simulation of the shattering of brittle objects," in *Computer Graphics Forum*, vol. 20, no. 2, 2001, pp. 81–91.

[40] P. A. CUNDALL, "A computer model for simulating progressive, large-scale movement in blocky rock system," *Proceedings of the International Symposium on Rock Mechanics*, 1971.

[41] P. A. Cundall and O. D. L. Strack, "A discrete numerical model for granular assemblies," *Géotechnique*, vol. 29, no. 1, pp. 47–65, 1979.

[42] N. Bell, Y. Yu, and P. J. Mucha, "Particle-based simulation of granular materials," in *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 2005, pp. 77–86.

[43] I. Alduán, A. Tena, and M. A. Otaduy, "Simulation of high-resolution granular media." in *CEIG*, 2009, pp. 11–18.

[44] W. Rungjiratananon, Z. Szego, Y. Kanamori, and T. Nishita, "Real-time animation of sand-water interaction," in *Computer Graphics Forum*, vol. 27, no. 7, 2008, pp. 1887–1893.

[45] Y. Yue, B. Smith, P. Y. Chen, M. Chantharayukhonthorn, K. Kamrin, and E. Grinspun, "Hybrid grains: adaptive coupling of discrete and continuum simulations of granular media," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 6, pp. 1–19, 2018.

[46] I. ITASCA Consulting Group, "Pfc." [Online]. Available: https://www.itascacg.com/software/PFC

[47] M. Bergou, M. Wardetzky, S. Robinson, B. Audoly, and E. Grinspun, "Discrete elastic rods," in *ACM SIGGRAPH 2008 papers*, 2008, pp. 1–12.

[48] T. Kugelstadt and E. Schömer, "Position and orientation based cosserat rods." in *Symposium on Computer Animation*, 2016, pp. 169–178.

[49] C. Soler, T. Martin, and O. Sorkine-Hornung, "Cosserat rods with projective dynamics," in *Computer Graphics Forum*, vol. 37, no. 8, 2018, pp. 137–147.

[50] T. Hädrich, J. Scheffczyk, W. Palubicki, S. Pirk, and D. L. Michels, "Interactive Wood Fracture," in *Eurographics/ ACM SIGGRAPH Symposium on Computer Animation - Posters*, 2020.

[51] M. Müller and N. Chentanez, "Solid simulation with oriented particles," in *ACM SIGGRAPH 2011 papers*, 2011, pp. 1–10.

[52] D. André, I. Iordanoff, J.-l. Charles, and J. Néauport, "Discrete element method to simulate continuous material by using the cohesive beam model," *Computer methods in applied mechanics and engineering*, vol. 213, pp. 113–125, 2012.

[53] T. Qu, Y. Feng, T. Zhao, and M. Wang, "Calibration of linear contact stiffnesses in discrete element models using a hybrid analytical-computational framework," *Powder Technology*, vol. 356, pp. 795–807, 2019.

[54] T. Qu, Y. Feng, T. Zhao, and M. Wang, "A hybrid calibration approach to hertz-type contact parameters for discrete element models," *International Journal for Numerical and Analytical Methods in Geomechanics*, vol. 44, no. 9, pp. 1281–1300, 2020.

[55] T. Qu, Y. Feng, M. Wang, and S. Jiang, "Calibration of parallel bond parameters in bonded particle models via physics-informed adaptive moment optimisation," *Powder Technology*, vol. 366, pp. 527–536, 2020.

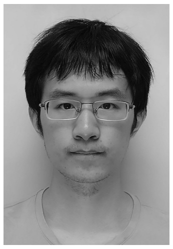
[56] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm," *ACM siggraph computer graphics*, vol. 21, no. 4, pp. 163–169, 1987.

[57] S. Luding, "Introduction to discrete element methods: basic of contact force models and how to perform the micro-macro transition

to continuum theory," *European journal of environmental and civil engineering*, vol. 12, no. 7-8, pp. 785–826, 2008.

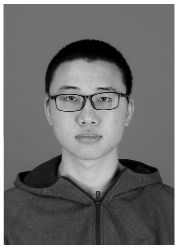
[58] J. Koteras and R. Lehoucq, "Estimating the critical time-step in explicit dynamics using the lanczos method," *International journal for numerical methods in engineering*, vol. 69, no. 13, pp. 2780–2788, 2007.

[59] Y. Hu, T.-M. Li, L. Anderson, J. Ragan-Kelley, and F. Durand, "Taichi: a language for high-performance computation on spatially sparse data structures," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 6, pp. 1–16, 2019.



Jia-Ming Lu received his bachelor's degree in computer science from Tsinghua University in 2016. He is currently a PhD candidate in the Department of Computer Science and Technology, Tsinghua University. His research interests include physically-based simulation and rendering.

Chen-Feng Li received the BEng and MSc degrees from Tsinghua University in 2002, and the PhD degree from Swansea University in 2006. He is currently a Chair Professor at the College of Engineering of Swansea University. Driven by practical engineering challenges arising from civil, aerospace, material, geotechnical and creative industries, his research centres largely on computational engineering, including computational solid mechanics, computational fluid dynamics and computer graphics. Besides the academic position at Swansea University, he also serves as the Editor-in-Chief for the journal Engineering Computations.



Geng-Chen Cao Geng-Chen Cao is a senior student at Tsinghua University. His research interest is computer graphics.

Shi-Min Hu received the Ph.D. degree from Zhejiang University, in 1996. He is currently a Professor with the Department of Computer Science and Technology, Tsinghua University, Beijing, and Director of State Key Laboratory of Virtual Reality Technology and Systems at Beihang University, Beijing. He has authored over 100 papers in journals and refereed conference. His research interests include digital geometry processing, video processing, rendering, computer animation, and computer-aided geometric design. He is the Editor-in-Chief of Computational Visual Media, and on the Editorial Board of several journals, including Computer Aided Design (Elsevier) and Computer & Graphics (Elsevier).