Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 416 (2023) 116370
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

Revolutionizing granular matter simulations by high-performance ray tracing discrete element method for arbitrarily-shaped particles

Shiwei Zhao*, Jidong Zhao*

Department of Civil and Environmental Engineering, The Hong Kong University of Science and Technology, Hong Kong, China

Received 25 May 2023; received in revised form 9 August 2023; accepted 10 August 2023
Available online xxxx

Abstract

It remains a challenge for discrete element method to simulate large-scale systems with arbitrarily shaped particles. This paper presents a novel approach called Ray Tracing Discrete Element Method (RTDEM) to offer ultra-high efficiency in simulating particles with arbitrary shapes. In RTDEM, we use a triangular mesh to represent the surface of a particle for greater flexibility in shape representation. Template meshes are further introduced to group particles of different shapes to save computing memory. Key to RTDEM are innovative ray tracing-based algorithms specially designed for accurate and efficient contact detection and resolution between arbitrarily shaped particles. Based on the RT algorithms, discrete potential field functions are further proposed for the discrete mesh vertices and facets to define energy-conservative contact models that include vertex-potential, facet-potential models, and their combination. The integration of these key elements into RTDEM offers a whole new DEM framework that readily empowers any parallel computing architecture for high acceleration on granular media modeling. It matches particularly well with the Graphics Processing Units (GPUs) to fully unleash the computing power of ray tracing cores for large-scale DEM simulations of arbitrarily shaped particles, even on a personal computer with a common gaming GPU card.

Keywords: Discrete element method; Discrete potential field; Arbitrary shape; Ray tracing

1. Introduction

The shape of particles plays a crucial role in affecting various aspects of the physical properties and mechanical behaviors of granular materials under static and dynamic conditions [1,2]. The flow of a granular system in the presence of interstitial fluids can be significantly influenced by particle shape, as exemplified by the bed load sediment transport phenomenon [3]. Therefore, quantifying and understanding the role of particle shape in relation with the mechanical responses of granular media has become a focus of recent interest in the community, in particular in numerical modeling. Among many numerical tools, the discrete element method (DEM) has been widely used to explore the microscopic origins of the mechanical behavior of granular materials. While conventional DEM commonly employs circular or spherical particles for efficient simulations, variants of DEM have recently been developed to model particles with different shapes [4]. They collectively help to gain new insights into the

* Corresponding authors.
E-mail addresses: ceswzhao@ust.hk (S. Zhao), jzhao@ust.hk (J. Zhao).

https://doi.org/10.1016/j.cma.2023.116370
0045-7825/© 2023 Elsevier B.V. All rights reserved.

underlying physics of large granular systems and aid in designing and optimizing systems in civil, chemical, and mining engineering.

The introduction of irregular or realistic complex particle shapes requires the development of computationally efficient contact models for large-scale simulations. Most existing studies have focused on the use of convex shapes, such as ellipsoids [5,6], superellipsoids [7,8], poly-superellipsoids [9], convex Fourier series-representation shapes [10], convex/dilated polyhedra [11,12], and convex NURBS [13]. These shapes allow for efficient interparticle contact detection and resolution using convex-optimization algorithms [14], such as the simplex-based GJK algorithm [15] and LM algorithm [16]. However, these algorithms are not applicable to arbitrarily shaped particles (e.g., general concave polyhedra [17], Fourier spectrum-based shapes [18] and spherical harmonics-based shapes [19]) for which brute-force algorithms are generally needed. For example, possible face–face, vertex-face and face-edge pairs are traversed for contact detection among polyhedra in early studies, e.g., [20,21] and among others. The algorithms proposed for contact problems in the finite element method have also garnered increasing attention in interparticle contact detection and resolution within the framework of DEM. Notably, the master-master algorithms [22] have been successfully employed in DEM such as [17], while the node-to-surface algorithms [23] have been utilized in works, e.g., [24]. These brute-force algorithms are flexible in dealing with any shape representations but frequently sacrifice computational efficiency and robustness. Interested readers are referred to a recent review on computational modeling of granular matter [25].

Energy-conserving contact models (e.g., volume-based contact models [26] for convex polyhedra) have been extended for arbitrarily shaped particles in conjunction with the brute-force algorithms [27], lending remarkable improvements on numerical stability. Specifically, when arbitrarily shaped particles are represented by triangular (or tetrahedral) meshes, there are two prevailing approaches for contact detection and resolution associated with energy-conserving contact models. The first approach is to compute the overlap volume between the two meshes by discretizing the particles into tetrahedra and resolving the inter-tetrahedron overlap one by one. This approach is good for numerical stability when using a volume-based contact model, but it is the most computationally intensive. The second approach avoids the computation of overlap volume by computing the intersection line between the two meshes [27]. This may significantly improve computational efficiency because of dimensional degeneration from the inter-tetrahedron overlap to the inter-triangle intersection and thus reduced computational cost.

Similar to energy-conserving contact models, nodal penetration potential within a particle surface is introduced in DEM variants such as LS-DEM [24] and SDF-DEM [28] based on the node-to-surface algorithms [23]. In these variants, one particle of a contacting pair is represented by level-set (LS) or signed distance field (SDF) functions, while the other has a surface with discretized nodes. This approach can further reduce the inter-surface (triangle) intersection problem to the node-surface (triangle) potential problem. The critical step is to compute the potential of the node within another particle. One direct option is to find a proper vertex-triangle pair, such as the one with the shortest distance, which is called direct computing. However, all vertex-triangle distances require explicit comparison, which is an outstanding issue. An alternative approach is to define an explicit potential function, thereby no need of explicit comparison. Typically, one can define a potential function with a radial distance between particle surface and center when the surface function is analytically given. Nevertheless, the radial distance-based potential is less accurate, which may cause possible issues of numerical stability. Moreover, it is non-trivial to have an elegant and efficient mathematical description of arbitrary particle shapes. Although the spherical harmonics (SH) function has been successfully employed in DEM, it is highly computationally expensive to solve a potential function based on an SH function [28].

To mitigate the above issues on both numerical efficiency and robustness of modeling arbitrarily shaped particles in DEM, previous studies [24,28] use LS and SDF functions to pre-cache the possible potential values within a particle. Hence, the contact problem becomes to query the potential value of a node within another particle. Since the pre-cached potential values are arranged at a lattice grid to facilitate the query procedure with a lookup-table algorithm, the potential value at the node can be interpolated in terms of the values of the grid nodes in the vicinity of the node. However, the allocation and access of the pre-cached data may cause efficiency issues associated with computer memory. It is less suitable for parallel computing in advanced computing systems, such as modern GPUs. Specifically, the lookup-table algorithm will cause performance degradation due to the bottleneck of data communication on GPUs, while the powerful GPU cores will not be well leveraged because of less computation. Therefore, it deserves efforts to propose a revolutionary DEM framework that is accurate, efficient and robust for modeling arbitrarily shaped particles in large-scale simulations.

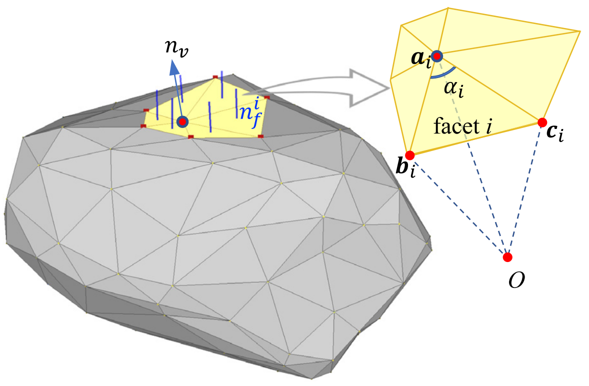

Fig. 1. A triangular mesh and its geometric measures facet normal $\boldsymbol{n}_f$ and vertex normal $\boldsymbol{n}_v$.

This study proposes a paradigm-shifting approach for modeling arbitrarily shaped particles represented by flexible triangular meshes in DEM. It features a ray tracing-based algorithm for contact detection and resolution in conjunction with discrete potential field functions, which overcomes the outstanding issues associated with the direct computing or pre-caching approaches in the previous studies. The framework of the proposed Ray Tracing Discrete Element Method (RTDEM) is outlined in Section 2, followed by the implementation of hardware acceleration with ray tracing cores on GPUs in Section 3. A set of numerical examples is then showcased to verify the implementation and demonstrate the performance of RTDEM in Section 4. Finally, conclusions are summarized in Section 5.

2. Ray tracing discrete element method

2.1. Triangularization of arbitrarily shaped particle

The surface of a particle, regardless of its complexity, can be effectively represented by a triangular mesh with a prescribed resolution. For simplicity, we use the numbers of vertices N_v and facets N_f to quantify the resolution of a particle surface. The triangular facet is defined by three vertices $\boldsymbol{a}$, $\boldsymbol{b}$ and $\boldsymbol{c}$, as shown in Fig. 1, such that the outward normal of the facet is given by

$$\boldsymbol{n}_f = \frac{(\boldsymbol{b}-\boldsymbol{a})\times(\boldsymbol{c}-\boldsymbol{a})}{\|(\boldsymbol{b}-\boldsymbol{a})\times(\boldsymbol{c}-\boldsymbol{a})\|} \tag{1}$$

An area vector $\boldsymbol{S}$ at a vertex is defined by weighting the area vectors of the adjacent facets of the vertex, given by

$$\boldsymbol{S} = \sum_{i=1}^{N_{af}} A_i \boldsymbol{n}_f^i \tag{2}$$

where A_i and $\boldsymbol{n}_f^i$ are the area and outward normal of the ith adjacent facet at the vertex, respectively; N_{af} is the number of adjacent facets. Another geometric measure is the vertex normal $\boldsymbol{n}_v$, defined as an angle weighted pseudonormal that was proposed to compute signed distance for polyhedra in the literature [29], i.e.,

$$\boldsymbol{n}_v = \frac{\sum_{i=1}^{N_{af}} \alpha_i \boldsymbol{n}_f^i}{\|\sum_{i=1}^{N_{af}} \alpha_i \boldsymbol{n}_f^i\|} \tag{3}$$

where α_i is the angle of the ith facet at the vertex; $\boldsymbol{n}_f^i$ is the outward normal of the ith facet; N_{af} is the number of adjacent facets at the vertex.

The mesh-based representation of particle shape offers great flexibility for modeling arbitrarily shape particles that can be either generated by numerical algorithms such as spherical harmonics [30] or reconstructed from X-ray CT scanned images [19]. Moreover, the discrete vertices can serve as Lagrangian points in immersed boundary techniques when coupling DEM with computational fluid dynamics (CFD) for particle–fluid interactions [31]. Note that the terms “particle surface”, “triangular mesh” and “mesh” will be used interchangeably throughout the presentation.

2.2. Inertia properties

Mass m, centroid $\boldsymbol{x}^p$ and moment of inertia $\boldsymbol{J}$ are basic inertia properties used to resolve particle motion. These measures can be computed by decomposing the particle into tetrahedra, each with a triangular facet of the particle surface as its base and a reference point (e.g., the origin) as its apex. The inertia properties of a particle can then be determined by summing the measures of the individual tetrahedra, i.e.,

$$m = \rho \sum_{i=1}^{N_f} V_i \tag{4}$$

$$\boldsymbol{x}^c = \frac{\sum_{i=1}^{N_f} (\boldsymbol{a}_i + \boldsymbol{b}_i + \boldsymbol{c}_i) V_i}{4 \sum_{i=1}^{N_f} V_i} \tag{5}$$

$$\boldsymbol{J} = \sum_{i=1}^{N_f} \boldsymbol{J}^{(i)} V_i \tag{6}$$

where ρ is mass density; $\boldsymbol{a}_i, \boldsymbol{b}_i$ and $\boldsymbol{c}_i$ are the position vectors of the three vertices of the ith triangular facet (i.e., base of the ith tetrahedron), respectively; N_f is the total number of triangular facets of the particle surface. The signed volume of the ith tetrahedron is given by

$$V_i = \frac{1}{6}(-c_0 b_1 a_2 + b_0 c_1 a_2 + c_0 a_1 b_2 - a_0 c_1 b_2 - b_0 a_1 c_2 + a_0 b_1 c_2). \tag{7}$$

Note that the three vertices $\boldsymbol{a}_i, \boldsymbol{b}_i$ and $\boldsymbol{c}_i$ are counterclockwise labeled to ensure an outward normal of the facet. The moment of inertia of the ith tetrahedron is obtained with respect to the mass centroid, i.e.,

$$\boldsymbol{J}^{(i)} = \begin{bmatrix} I_{11}^{(i)} + I_{22}^{(i)} & -I_{01}^{(i)} & -I_{02}^{(i)} \\ -I_{01}^{(i)} & I_{00}^{(i)} + I_{22}^{(i)} & -I_{12}^{(i)} \\ -I_{02}^{(i)} & -I_{12}^{(i)} & I_{00}^{(i)} + I_{11}^{(i)} \end{bmatrix} \tag{8}$$

with

$$I_{\alpha\beta}^{(i)} = \frac{\rho}{20}(\boldsymbol{h} - x_\alpha^p \mathbf{1}) \cdot (\boldsymbol{t} - 4 x_\beta^p \mathbf{1}) \tag{9}$$

$$\boldsymbol{h} = [a_\alpha, b_\alpha, c_\alpha] \tag{10}$$

$$\boldsymbol{t} = [2a_\beta + b_\beta + c_\beta, a_\beta + 2b_\beta + c_\beta, a_\beta + b_\beta + 2c_\beta] \tag{11}$$

where α and β are indices in [0,1,2] for the corresponding components; ρ is the mass density. The principal moments of inertia can be obtained by eigen-decomposition of $\boldsymbol{J}^{(i)}$, where the eigenvectors $(\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3)$ and eigenvalues (J_1, J_2, J_3) are principal axes and the corresponding principal moments of inertia, respectively.

2.3. Template mesh

A template mesh is introduced to facilitate shape transformation and save computing memory. Its mass centroid serves as the local origin and its principal axes as the local Cartesian axes. Consequently, the mesh can be transformed to its local configuration by:

$$\boldsymbol{x}_v^{\prime j} = \boldsymbol{M}^{-1}(\boldsymbol{x}_v^j - \boldsymbol{x}^p) \tag{12}$$

where $\boldsymbol{x}_v^{\prime j}$ and $\boldsymbol{x}_v^j$ are the new and original position vectors of the jth vertex of the mesh, respectively; $\boldsymbol{M}$ is the rotation matrix determined by the principal axes, i.e.,

$$\boldsymbol{M} = |\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3|. \tag{13}$$

Note that the sequence of $(\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3)$ in $\boldsymbol{M}$ may vary.

2.4. Shape transformation

Shape transformation, such as scale, shear, rotation and translation, can be applied to a template mesh, which facilitates memory saving by sharing the mesh data with particles that may have different positions, orientations or shapes. Moreover, this approach avoids accumulative errors in particle shapes during the integration of particle motion. In this work, transformation matrices $\mathcal{S}$, $\mathcal{R}$, and $\mathcal{T}$ are used to account for scale, rotation, and translation, respectively, given by,

$$\mathcal{S}=\begin{bmatrix} S_x & 0 & 0 & 0\\ 0 & S_y & 0 & 0\\ 0 & 0 & S_z & 0\\ 0 & 0 & 0 & 1\end{bmatrix},\quad \mathcal{R}=\begin{bmatrix} R_{xx} & R_{xy} & R_{xz} & 0\\ R_{yx} & R_{yy} & R_{yz} & 0\\ R_{zx} & R_{zy} & R_{zz} & 0\\ 0 & 0 & 0 & 1\end{bmatrix},\quad \mathcal{T}=\begin{bmatrix} 0 & 0 & 0 & T_x\\ 0 & 0 & 0 & T_y\\ 0 & 0 & 0 & T_z\\ 0 & 0 & 0 & 1\end{bmatrix} \tag{14}$$

The overall transformation is given by the following matrix

$$\boldsymbol{TRS}=\begin{bmatrix} S_xR_{xx} & S_yR_{xy} & S_zR_{xz} & T_x\\ S_xR_{yx} & S_yR_{yy} & S_zR_{yz} & T_y\\ S_xR_{zx} & S_yR_{zy} & S_zR_{zz} & T_z\\ 0 & 0 & 0 & 1\end{bmatrix} \tag{15}$$

The particle surface $\boldsymbol{x}^s$ pertaining to a template mesh is given by

$$\boldsymbol{x}^s = s\boldsymbol{R}'\boldsymbol{x}^m + \boldsymbol{x}^p \tag{16}$$

where $\boldsymbol{x}^m$ is a surface point on the template mesh; $\boldsymbol{x}^p$ is particle position (center of mass); $\boldsymbol{R}'$ is a 3×3 matrix of particle orientation; s is a scale factor for isotropic scale (i.e., $S_x = S_y = S_z = s$). In the presence of a scale factor, the basic particle geometric/inertia measures are given by

$$m' = ms^3, \quad \boldsymbol{J}' = \boldsymbol{J}s^5 \tag{17}$$

Note that particle orientation is often represented by a quaternion $\boldsymbol{q}(w, x, y, z)$ to facilitate the implementation in rotation integration, thus Eq. (16) is rewritten as

$$\boldsymbol{x}^s = s\boldsymbol{q}\boldsymbol{x}^m + \boldsymbol{x}^p \tag{18}$$

2.5. Ray tracing contact detection

2.5.1. General contact detection algorithm

Ray tracing is a widely used technique to enhance realistic visualization in computer graphics and gaming by rendering images and videos. In practical rendering, millions of rays are cast into a model scene, and their paths change via reflection and/or refraction when colliding with an object, similar to the real world. Taking a similar approach, ray tracing has been employed for contact detection in computer graphics [32]. In this work, we propose a ray tracing-based algorithm for contact detection and resolution in DEM simulations of arbitrarily shaped particles.

As shown in Fig. 2, a ray with a length of l^i will be cast along the opposite direction of the normal at a vertex i, i.e., $-\boldsymbol{n}_v^i$ defined in Eq. (3). The ray length can be determined by

$$l^i = \|\boldsymbol{x}_h^i - \boldsymbol{x}_v^i\| \tag{19}$$

where $\boldsymbol{x}_h^i$ is the hit point on the particle surface, i.e., the intersection point between the ray and a hit triangular facet (see Fig. 2a); $\boldsymbol{x}_v^i$ is the position vector of the vertex i. However, it is not necessary for rays to hit their own particle surface. Thus, the ray length is scaled down by a weight w (see Fig. 2b). Moreover, the ray lengths can be the same for all vertices of a particle (see Fig. 2c). For simplicity, the single ray length can be defined as the average of three semi-lengths (l_x, l_y, l_z), i.e.,

$$l = \frac{1}{3}(l_x + l_y + l_z) \tag{20}$$

Note that the semi-lengths can be defined with respect to the axis-aligned bounding box (AABB) or the principal axes of a particle. Three-dimensional ray casting is showcased for different shapes in Fig. 3.

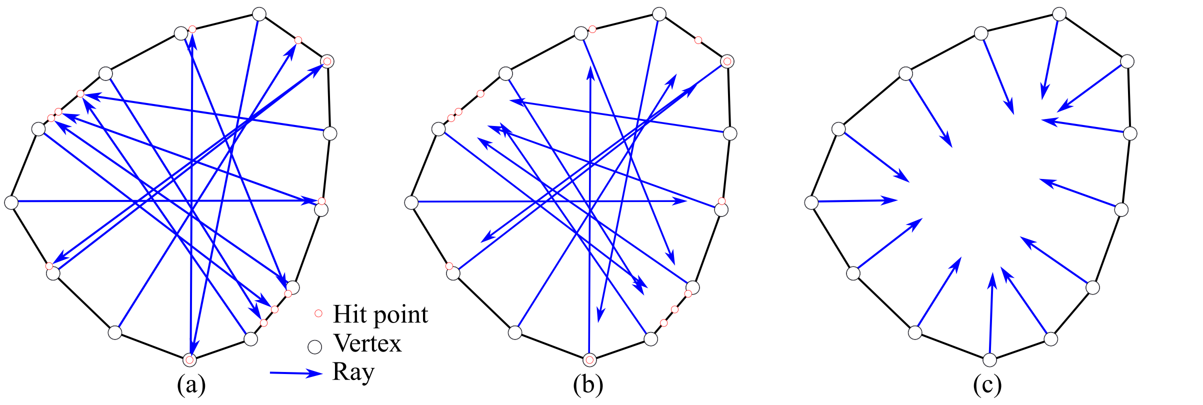

Fig. 2. Two-dimensional illustration of ray casting inside a triangular mesh with: single ray length (a) and different ray lengths l^i (b) and wl^i (c).

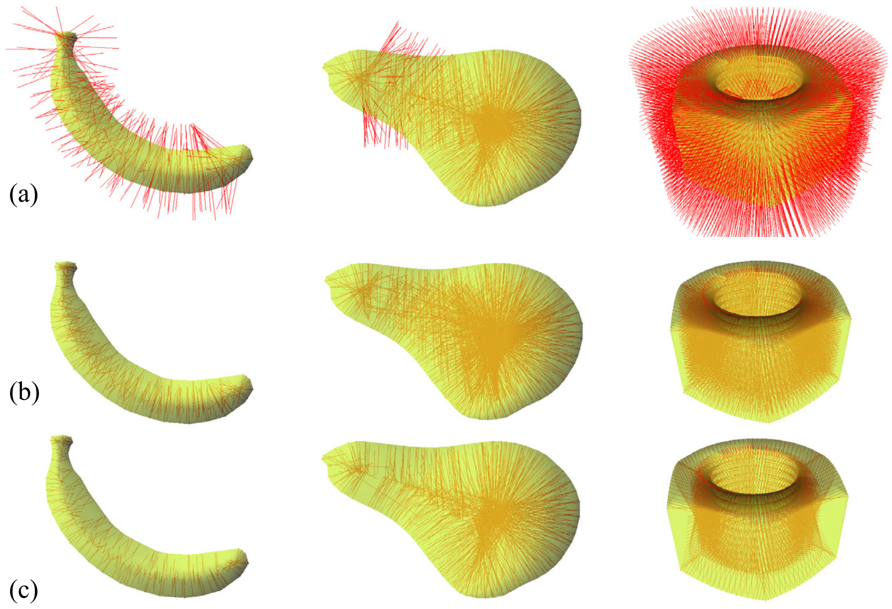

Fig. 3. Examples of different ray casting lengths for banana, pear and nut: (a) $0.5l$, (b) $0.9l^i$ and (c) $0.5l^i$.

Given two particles, as shown in Fig. 4(a), contacts occur once there are rays from one particle hitting the facets of another particle. To facilitate the contact detection between rays and facets, a bounding volume hierarchy (BVH) is introduced to organize the axis-aligned bounding box (AABB) of each facet. Hence, the ray-facet contact detection becomes to examine the intersection of a ray and an AABB first, and another exact intersection computation is then applied to the ray-facet pair if the ray collides the AABB. This procedure can be efficiently implemented via an acceleration structure elaborated in the following subsection.

Furthermore, it is important to address the occurrence of an extreme configuration, as depicted in Fig. 4(b), which can arise during simulations. In this configuration, one particle, denoted as Particle A, can penetrate another elongated or thin particle, denoted as Particle B, due to a significant relative velocity. This particular scenario can lead to numerical issues and potentially cause the simulation to crash in SDF-DEM and LS-DEM methods. In SDF-DEM and LS-DEM, the vertices D and E of Particle A can be identified as being outside Particle B based on their SDF or LS values. As a result, these vertices are ignored, resulting in no repulsive forces being applied. On the other hand, the repulsive forces acting on vertices C and F are significantly underestimated due to their proximity to the surface of Particle B. Consequently, when such a configuration occurs, Particle A may effectively

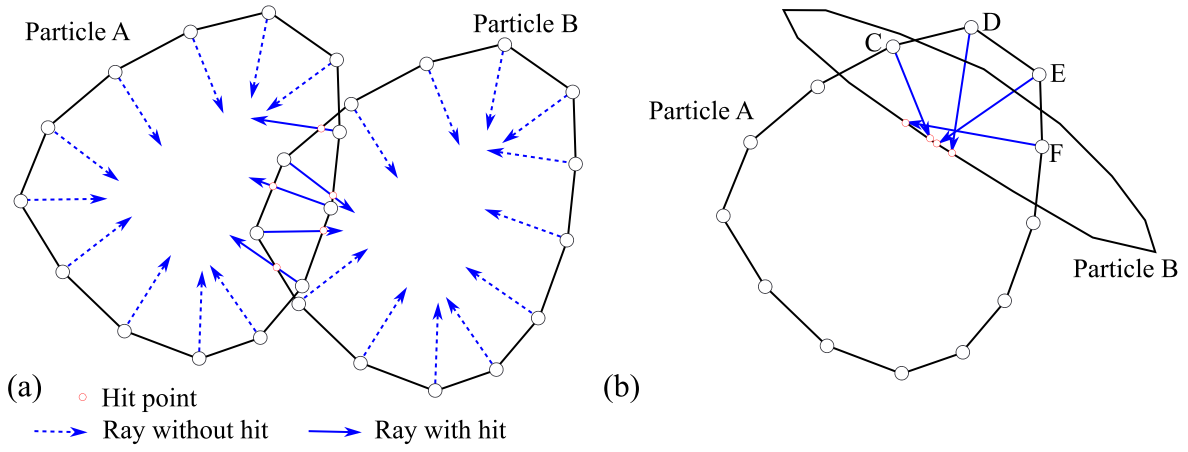

Fig. 4. (a) Illustration of ray tracing contact detection and resolution; (b) an extreme configuration that may be encountered in a simulation.

pass through Particle B without the expected repulsive forces in SDF-DEM and LS-DEM. In contrast, our RTDEM framework offers a robust solution to this numerical problem without requiring any additional implementation. In RTDEM, the hit point occurs only on the inward side of a facet. For example, the ray at vertex D ignores the first hit of the surface of Particle B and eventually collides with the inner side of Particle B at another facet. The inner side of Particle B satisfies the following condition,

$$\boldsymbol{n}_r \cdot \boldsymbol{n}_f > 0 \tag{21}$$

where $\boldsymbol{n}_r$ is the ray direction, and $\boldsymbol{n}_f$ is the outward normal of a hit facet given in Eq. (1).

2.5.2. *Acceleration structure*

The most computationally intensive aspect of ray tracing is collision detection between rays and objects. To address this challenging task, a sophisticated data structure is required to organize spatial objects. This structure is commonly known as an acceleration structure in computer graphics. When dealing with objects that have complex shapes, their surfaces are often represented by triangular meshes, allowing for arbitrary shapes. As a result, collision detection between a ray and an object is simplified to checking for a possible intersection between the ray and the triangular mesh. Our RTDEM utilizes a workflow similar to the general collision detection in ray tracing. Therefore, the contact detection can be easily implemented using state-of-the-art algorithms for ray tracing, including acceleration structures and hardware acceleration.

An acceleration structure can be established as a bounding volume hierarchy (BVH) tree, which has been successfully applied in computer graphics [33] and computational mechanics [34] to improve computational efficiency of contact detection in multi-body systems. As shown in Fig. 5, we introduce two levels (bottom level and top level) of sub-BVH trees to facilitate the management of an acceleration structure. For the bottom level, aligned axis bounding boxes (AABB) of all triangular facets for a particle surface are grouped into a BVH tree. For the top level, AABBs of all particles are then grouped into another BVH tree. Using mesh templates, the particles having the same mesh template can share a bottom-level BVH tree but with different scale, rotation and/or translation. In other words, a bottom-level BVH can be established with respect to a mesh template, and for a particle with given scale, rotation and translation, the corresponding BVH tree can be readily inserted into the top-level BVH tree, which is called as a particle instance (of the template mesh).

It is assumed that particle shapes remain unchangeable in a simulation. Hence, a particle instance requires updating once either particle position or particle orientation updates during particle dynamics. With the two-level BVH scheme, it is remarkably efficient to update the entire acceleration structure based on its previous data, especially for the case of small particle motion that holds generally for one timestep simulation in DEM. It is noteworthy that the performance of an acceleration structure may degrade considerably after many updates, in which case one may reestablish a fresh acceleration structure instead of updating it based on the previous data. The detailed implementation is not introduced here to avoid possible distraction. Indeed, there are well documented algorithms to implement a BVH tree on GPUs, and interested readers are referred to the literature [35,36].

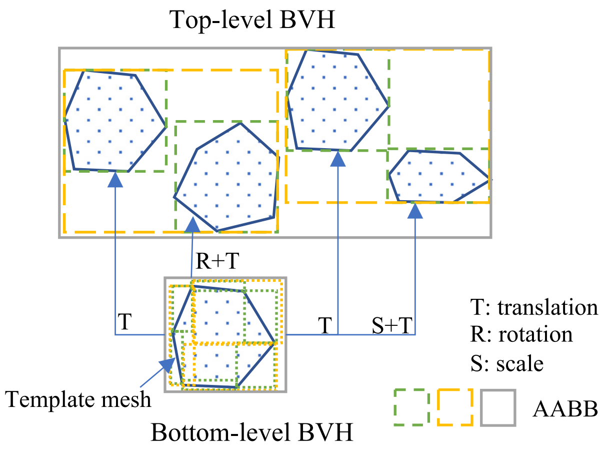

Fig. 5. Two-dimensional illustration of the proposed two-level acceleration structure with template meshes.

2.6. *Discrete potential field functions*

To introduce energy-conservative contact models for numerical stability [27], it is necessary to define a potential field function properly. Similar ideas have also been proposed for contact problems in finite element method, e.g., [37]. In previous DEM variants such as LS-DEM [24] and SDF-DEM [28], both level-set and signed distance field functions can be regarded as general potential field functions, which are continuously defined in space. Specifically, given a particle surface S, a potential field function $W(S, \boldsymbol{x}_p)$ can be defined so that the potential value at a point $\boldsymbol{x}_p$ is the minimum with respect to the surface (see Fig. 6a). Typically, such a continuous potential field function is implicit and requires pre-caching at a spatial lattice grid to facilitate lookup-table operations, as implemented in LS-DEM and SDF-DEM.

In this work, instead of defining a potential field with respect to the whole surface, we propose two discrete potential field functions W_v and W_f with respect to the ith vertex or facet, respectively, as illustrated in Fig. 6b and c.

$$W_v = W(\boldsymbol{x}_v^i, \boldsymbol{x}_p), \boldsymbol{x}_p \in \{\boldsymbol{x}_v^i + \lambda \boldsymbol{n}_v^i | i = 1, \dots, N_v\} \tag{22}$$

$$W_f = W(\boldsymbol{x}_f^i, \boldsymbol{x}_p), \boldsymbol{x}_p \in \{\boldsymbol{x}_f^i + \lambda \boldsymbol{n}_f^i | i = 1, \dots, N_f\} \tag{23}$$

where N_v and N_f are vertex number and facet number of a particle surface, respectively; $\boldsymbol{n}_v^i$ and $\boldsymbol{n}_f^i$ are the ith vertex normal given in Eq. (3) and ith facet normal given in Eq. (1), respective; λ is an arbitrary constant that locates the point $\boldsymbol{x}_p$; $\boldsymbol{x}_v^i$ is the position of the ith vertex; $\boldsymbol{x}_f^i$ is a projection point of $\boldsymbol{x}_p$ on the ith facet. These two discrete potential field functions can be superimposed to approximate the continuous potential field function, i.e.,

$$W(S, \boldsymbol{x}_p) \simeq W(\boldsymbol{x}_v^i, \boldsymbol{x}_p) + W(\boldsymbol{x}_f^i, \boldsymbol{x}_p), S \in \{\boldsymbol{x}_v^i | i = 1, \dots, N_v\} \cup \{\boldsymbol{x}_f^i | i = 1, \dots, N_f\}. \tag{24}$$

Note that for a given point $\boldsymbol{x}_p$, W_v and W_f are defined mutually exclusive, i.e., $\{\boldsymbol{x}_v^i + \lambda \boldsymbol{n}_v^i | i = 1, \dots, N_v\} \cap \{\boldsymbol{x}_f^i + \lambda \boldsymbol{n}_f^i | i = 1, \dots, N_f\} = \emptyset$. Compared with potential fields based on level-set functions or signed distance functions, the discrete potential field functions define potential values at each individual vertex or facet only. This feature significantly reduces computing memory usage because no pre-cached potential values are required to be distributed within a particle.

For a colliding ray-facet pair, a general discrete potential field function $W(\boldsymbol{x}^v, \boldsymbol{x})$ is defined by

$$W(\boldsymbol{x}^{vi}, \boldsymbol{x}) = k \frac{\boldsymbol{S}^i \cdot \boldsymbol{n}_v^i}{\vartheta S_p} d^{\vartheta} \tag{25}$$

where k is a parameter of material stiffness; $\boldsymbol{S}^i$ is the area vector at the vertex i, given in Eq. (2); S_p is the surface area of the given particle; $\boldsymbol{v}_v^i$ is the vertex normal defined in Eq. (3), and ϑ is the potential order. d is the distance

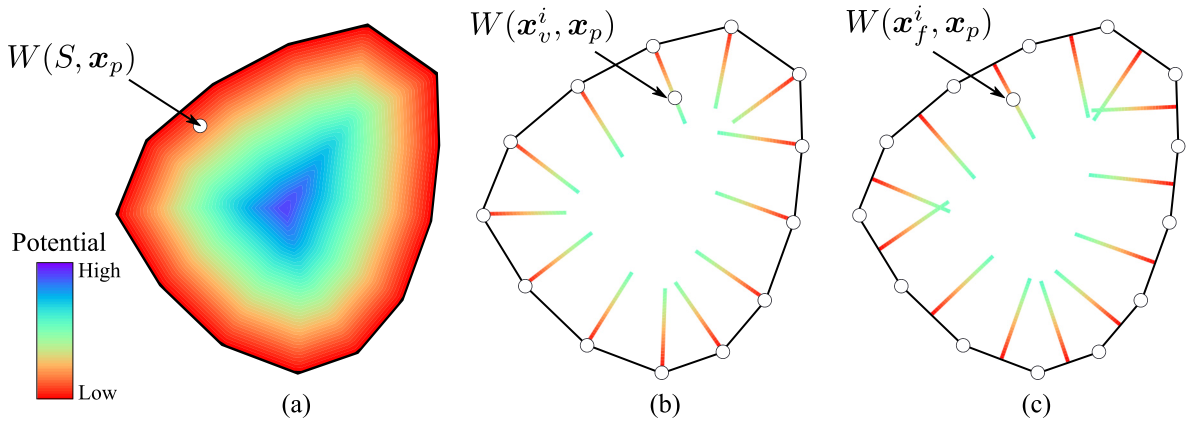
$W(S, \boldsymbol{x}_p)$
Potential
High
Low
(a)
$W(\boldsymbol{x}_v^i, \boldsymbol{x}_p)$
(b)
$W(\boldsymbol{x}_f^i, \boldsymbol{x}_p)$
(c)

Fig. 6. Exemplified potential field functions: (a) continuous $W(S, \boldsymbol{x}_p)$ against the whole surface, (b) discrete $W(\boldsymbol{x}_v^i, \boldsymbol{x}_p)$ against vertex i and (c) discrete $W(\boldsymbol{x}_f^i, \boldsymbol{x}_p)$ against facet i.

given by

$$d = \begin{cases} (\boldsymbol{x}_v^i - \boldsymbol{x}_h^i) \cdot \boldsymbol{n}_v^i & \text{hit point against vertex} \\ (\boldsymbol{x}_h^i - \boldsymbol{x}_v^i) \cdot \boldsymbol{n}_h^i & \text{vertex against hit facet} \end{cases} \tag{26}$$

where $\boldsymbol{n}_h^i$ is the surface normal at the hit point $\boldsymbol{x}_h^i$, equal to the facet normal $\boldsymbol{n}_f$ of the hit facet. Note that Eq. (26) specifically addresses point-vertex and point-facet interactions, omitting explicit consideration of other primitive interactions such as edge-edge and edge-facet interactions (see e.g., [34]). Instead, the primitives (edges and facets) surrounding a point (i.e., a vertex from the other particle surface) are collectively incorporated by being combined with the point using a local geometric measure, as defined in Eq. (34). Moreover, both point-vertex and point-facet interactions can be unified into a point-surface interaction, which can be described by a continuous potential field as expressed in Eq. (24). This enables a comprehensive treatment of the interactions between points and surfaces within the model, encompassing the combined effects of vertices, facets and their respective geometric measures. Therefore, the two discrete potential field functions are rewritten as

$$W(\boldsymbol{x}_v^i, \boldsymbol{x}_h^i) = k \frac{\boldsymbol{S}^i \cdot \boldsymbol{n}_v^i}{\vartheta S_p} ((\boldsymbol{x}_v^i - \boldsymbol{x}_h^i) \cdot \boldsymbol{n}_v^i)^{\vartheta} \tag{27}$$

$$W(\boldsymbol{x}_f^i, \boldsymbol{x}_v^i) = k \frac{\boldsymbol{S}^i \cdot \boldsymbol{n}_v^i}{\vartheta S_p} ((\boldsymbol{x}_h^i - \boldsymbol{x}_v^i) \cdot \boldsymbol{n}_h^i)^{\vartheta} \tag{28}$$

Their derivatives with respect to $\boldsymbol{x}$ are given by

$$\frac{\partial W(\boldsymbol{x}_v^i, \boldsymbol{x}_h^i)}{\partial \boldsymbol{x}_h} = -k \frac{\boldsymbol{S}^i \cdot \boldsymbol{n}_v^i}{S_p} ((\boldsymbol{x}_v^i - \boldsymbol{x}_h^i) \cdot \boldsymbol{n}_v^i)^{(\vartheta-1)} \boldsymbol{n}_v^i \tag{29}$$

$$\frac{\partial W(\boldsymbol{x}_f^i, \boldsymbol{x}_v^i)}{\partial \boldsymbol{x}_v} = -k \frac{\boldsymbol{S}^i \cdot \boldsymbol{n}_v^i}{S_p} ((\boldsymbol{x}_h^i - \boldsymbol{x}_v^i) \cdot \boldsymbol{n}_h^i)^{(\vartheta-1)} \boldsymbol{n}_h^i \tag{30}$$

2.7. Contact model

2.7.1. Normal contact force

Given a scalar potential field $W(\boldsymbol{x})$, a normal contact force $\boldsymbol{f}^n$ can be obtained by

$$\boldsymbol{f}^n = -\frac{\partial W(\boldsymbol{x})}{\partial \boldsymbol{x}} \tag{31}$$

which yields an energy-conserving contact model as introduced in the literature [27]. In previous studies, e.g., [24,28], the node-to-surface algorithm was employed to solve contact forces. Specifically, a node (or vertex)

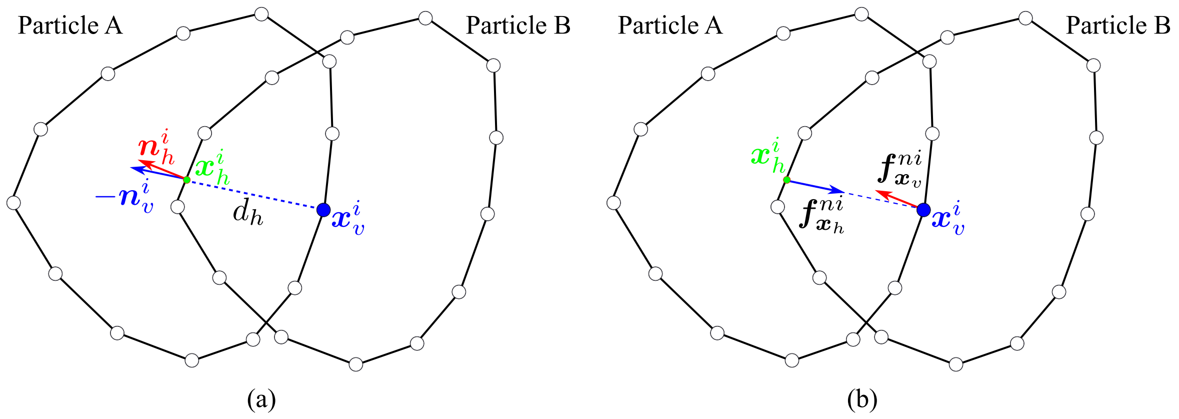

Fig. 7. (a) Contact geometric measures and (b) potential-based contact forces .

from one particle surface is regarded as an intruder, as shown in Fig. 7, and the potential value of this intruder is required against the surface of another particle, typically by looking up a pre-cached table. Therefore, the pre-cached table will store both the potential and its gradient.

With our proposed discrete potential field functions $W(\boldsymbol{x}_v^i, \boldsymbol{x}_h^i)$ and $W(\boldsymbol{x}_f^i, \boldsymbol{x}_v^i)$ given in Eqs. (27) and (28), the potential values at the ith vertex $\boldsymbol{x}_v^i$ and its hit point $\boldsymbol{x}_h^i$ are employed to compute the contact force. Therefore, the normal contact forces at the vertex $\boldsymbol{x}_v^i$ and the hit point $\boldsymbol{x}_h^i$ are given by

$$\boldsymbol{f}_{\boldsymbol{x}_h}^{ni} = -\frac{\partial W(\boldsymbol{x}_v^i, \boldsymbol{x}_h^i)}{\partial \boldsymbol{x}_h} = k_n \gamma^i d_h^{(\vartheta-1)} \boldsymbol{n}_v^i \tag{32}$$

$$\boldsymbol{f}_{\boldsymbol{x}_v}^{ni} = -\frac{\partial W(\boldsymbol{x}_f^i, \boldsymbol{x}_v^i)}{\partial \boldsymbol{x}_v^i} = k_n \gamma^i (d_h \boldsymbol{n}_h^i \cdot (-\boldsymbol{n}_v^i))^{(\vartheta-1)} \boldsymbol{n}_h^i \tag{33}$$

with

$$\gamma^i = \frac{\boldsymbol{S}^i \cdot \boldsymbol{n}_v^i}{S_p} \tag{34}$$

where k_n is the normal contact stiffness; d_h is the hit distance; γ^i is a local geometric measure for normalized facet area projection, which can be regarded as a weight for the overall normal contact force. Therefore, there are three options to define a normal contact force, i.e.,

$$\boldsymbol{f}_n^i = \boldsymbol{f}_{\boldsymbol{x}_h}^{ni} \tag{35}$$

$$\boldsymbol{f}_n^i = -\boldsymbol{f}_{\boldsymbol{x}_v}^{ni} \tag{36}$$

$$\boldsymbol{f}_n^i = \boldsymbol{f}_{\boldsymbol{x}_h}^{ni} - \boldsymbol{f}_{\boldsymbol{x}_v}^{ni} \tag{37}$$

where the third equation is a linear combination of contact forces at $\boldsymbol{x}_v^i$ and $\boldsymbol{x}_h^i$ for simplicity. For the sake of clarity, we denote these three normal contact forces as the *vertex-potential normal contact force*, *facet-potential normal contact force*, and *combined normal contact force*. Note that the contact force is applied directly to Particle B, while its reaction is applied to Particle A. The direction of normal contact force is taken as the contact normal $\boldsymbol{n}_c$, and the contact point $\boldsymbol{x}_c^i$ can be defined as

$$\boldsymbol{x}_c^i = \boldsymbol{x}_v^i + 0.5 d_h \boldsymbol{n}_c^i \tag{38}$$

Remarks. The proposed contact model can be considered as a type of multi-point contact model, as it introduces discrete vertices of a particle surface, distinguishing it from conventional single-point contact models. To quantify the overall behavior of multi-point contacts, an equivalent single-point contact model is proposed, following a power law relationship given by

$$F_n = K_n d^{\alpha} \tag{39}$$

where F_n is the equivalent normal contact force (or resultant contact force given by $F_n = \|\sum_i \boldsymbol{f}_n^i\|$); K_n is deformation-independent contact stiffness; α is a constant, and d is the overall penetration depth at the particle scale. Note that d may not have a direct relationship with the local penetration depth (the hit distance d_h) at each vertex. Notably, Eq. (39) resembles a linear spring model when α is equal to 1, or the Hertzian model when α is equal to $\frac{3}{2}$. In the case of our multi-point contact model, the contact force $\boldsymbol{f}_n^i$ is assumed to be independent of the local penetration depth for a model parameter ϑ set to 1. However, this assumption can lead to numerical issues in situations where penetrated vertices reach a constant number. Hence, ϑ is set to $\frac{3}{2}$ or 2 in the simulations of this work.

2.7.2. *Tangential contact force*

As for the tangential contact force, we use a linear contact model in conjunction with the Coulomb sliding model. The computation of the tangential contact force follows a conventional implementation. The relative displacement $\boldsymbol{s}$ at the contact point $\boldsymbol{x}^c$ is first solved by

$$\boldsymbol{s} = \big(\boldsymbol{v}_{AB} - (\boldsymbol{v}_{AB} \cdot \boldsymbol{n}_c)\boldsymbol{n}_c\big)\Delta t \tag{40}$$

where $\boldsymbol{n}_c$ is the contact normal, equal to $-\boldsymbol{n}_v$; $\boldsymbol{v}_{AB}$ is the relative velocity at the contact; Δt is the timestep. For consistency, the particle that has a ray cast is denoted as particle A, while the particle with facets being hit is denoted as particle B. Thus, the relative velocity $\boldsymbol{v}_{AB}$ is given by

$$\boldsymbol{v}_{AB} = \boldsymbol{v}_B + \boldsymbol{\omega}_B \times (\boldsymbol{x}^c - \boldsymbol{x}_B^p) - \boldsymbol{v}_A - \boldsymbol{\omega}_A \times (\boldsymbol{x}^c - \boldsymbol{x}_A^p) \tag{41}$$

where $\boldsymbol{v}, \boldsymbol{\omega}$ and $\boldsymbol{x}^p$ are the translational velocity, rotational velocity and particle position, respectively; and the subscripts A and B denote the corresponding particle A and B, respectively. Therefore, the increment of tangential contact force $\Delta \boldsymbol{f}^i$ is given by a linear displacement–force law, i.e.,

$$\Delta \boldsymbol{f}^i = k_t \boldsymbol{s} \tag{42}$$

where k_t is the tangential contact stiffness. Unlike the computation of normal contact force, the previous tangential contact force required to be rotated to the current tangential plane. Accordingly, two axes orthogonal with the contact normal, namely orthonormal axis $\boldsymbol{n}^{oa}$ and twist axis $\boldsymbol{n}^{ta}$, are introduced to facilitate the force rotation, defined as

$$\boldsymbol{n}^{oa} = \boldsymbol{n}_c' \times \boldsymbol{n}_c \tag{43}$$

$$\boldsymbol{n}^{ta} = \frac{\Delta t}{2}[\boldsymbol{n}_c' \times (\boldsymbol{\omega}_B + \boldsymbol{\omega}_A)] \cdot \boldsymbol{n}_c' \tag{44}$$

where $\boldsymbol{n}^{c'}$ is the previous contact normal. Then, the previous tangential contact force $\boldsymbol{f}_t'$ is rotated twice as follows

$$\boldsymbol{f}_t^{r1} = \boldsymbol{f}_t' - \boldsymbol{f}_t' \times \boldsymbol{n}^{oa} \tag{45}$$

$$\boldsymbol{f}_t^{r2} = \boldsymbol{f}_t^{r1} - \boldsymbol{f}_t^{r1} \times \boldsymbol{n}^{ta} \tag{46}$$

Alternatively, the previous tangential contact force $\boldsymbol{f}_t'$ can be simply projected onto the current contact plane, i.e.,

$$\boldsymbol{f}_t^{r2} = \boldsymbol{f}_t' - (\boldsymbol{f}_t' \cdot \boldsymbol{n}_c)\boldsymbol{n}_c \tag{47}$$

Therefore, the tangential contact force $\boldsymbol{f}_t^i$ is obtained as

$$\boldsymbol{f}_t^i = \boldsymbol{f}_t^{r2} + \Delta \boldsymbol{f}^i \tag{48}$$

with its magnitude subjected to the Coulomb condition, i.e.,

$$\|\boldsymbol{f}_t^i\| = \min\{\|\boldsymbol{f}_t^i\|, \mu\|\boldsymbol{f}_n^i\|\} \tag{49}$$

where μ is the coefficient of friction. Note that the tangential contact force $\boldsymbol{f}_t^i$ is not necessarily parallel to the relative tangential velocity at contact in a three-dimensional space [38].

2.8. *Motion integration*

Particle motion follows the Newton-Euler equations that are shown here for the sake of presentation, i.e.,

$$m\frac{d\boldsymbol{v}}{dt} = \boldsymbol{F}^{(b)} + \boldsymbol{F}^{(d)} + \boldsymbol{F}^{(c)} \tag{50}$$

$$\boldsymbol{J}\frac{d\boldsymbol{\omega}}{dt} + \boldsymbol{\omega} \times (\boldsymbol{J}\boldsymbol{\omega}) = \boldsymbol{T}^{(d)} + \boldsymbol{T}^{(c)} \tag{51}$$

where $\boldsymbol{v}$ and $\boldsymbol{\omega}$ are the translational and rotational velocities of a particle, respectively; m and $\boldsymbol{J}$ are the mass and moment of inertia of the particle, respectively; $\boldsymbol{F}^{(b)}$ is the body force; $\boldsymbol{F}^{(d)}$ and $\boldsymbol{T}^{(d)}$ are damping force and torque, respectively; $\boldsymbol{F}^{(c)}$ and $\boldsymbol{T}^{(c)}$ are the resultant contact force and torque on the particle, respectively, given by

$$\boldsymbol{F}^{(c)} = \sum_{i=1}^{N_{cv}^A}\left(-\boldsymbol{f}_n^i + \boldsymbol{f}_t^i\right) - \sum_{B\in N_p}\sum_{i=1}^{N_{cv}^B}\left(-\boldsymbol{f}_n^i + \boldsymbol{f}_t^i\right) \tag{52}$$

$$\boldsymbol{T}_{(c)} = \sum_{i=1}^{N_{cv}^A}(\boldsymbol{x}_c^i - \boldsymbol{x}_A) \times (-\boldsymbol{f}_n^i + \boldsymbol{f}_t^i) - \sum_{B\in N_p}\sum_{i=1}^{N_{cv}^B}(\boldsymbol{x}_c^i - \boldsymbol{x}_A) \times (-\boldsymbol{f}_n^i + \boldsymbol{f}_t^i) \tag{53}$$

where N_{cv}^A and N_{cv}^B are the numbers of vertices within the contact overlap for particles A and B, respectively; N_p is the neighbor number of particle A; $\boldsymbol{x}_A$ is the position vector of particle A.

Note that the Euler equation, i.e., Eq. (51), requires a body-fixed reference frame. In general, it is preferable to align the local axes with the principal axes of the particle, which facilitates the computation of particle rotation. Moreover, particle orientation is represented by a quaternion as mentioned in Section 2.4. The Newton-Euler equations are solved explicitly by a canonical Verlet scheme for simplicity in this work. The detailed implementation is not presented here for brevity, while interested readers are referred to the literature [9,39].

3. Hardware acceleration

The proposed computational framework is well-suited for parallel computing on conventional multi-core or multi-node CPU computing systems. Moreover, it introduces a novel paradigm of efficient parallel computing by largely leveraging the hardware of modern GPUs. Unlike conventional GPGPU computing that only uses shader cores (such as CUDA cores on NVIDIA's GPUs), RTDEM employs the latest hardware, i.e., ray tracing cores, to compute contact detection and resolution between arbitrarily shaped particles. This cutting-edge technology makes RTDEM a next-generation tool for large-scale DEM simulations with arbitrarily shaped particles using economically available computing resources, such as personal computers with a common gaming graphic card.

3.1. Ray tracing cores on GPUs

In RTDEM, many rays will be cast from particle vertices and then queried against an acceleration structure. This task is most often computationally time-consuming but can be generally solved in parallel on multi-core CPUs or shaders of GPUs. In addition to GPU shaders (e.g., CUDA cores on NVIDIA GPUs) for general-purpose computing, RTDEM benefits remarkably from ray tracing cores that were first introduced in NVIDIA's Turing architecture in 2019 to accelerate ray tracing tasks [40]. The ray tracing cores have been successfully utilized for general-purpose computing such as locating particle position in unstructured meshes [41], accelerating particle-based numerical simulations [42]. A ray tracing core has two individual physical units: box intersection and ray-triangle intersection evaluators. The box intersection evaluator checks if there is an intersection between a ray and an AABB, which can be regarded as a broad phase of contact detection. The ray-triangle intersection evaluator is a narrow phase of contact detection, which computes the intersection point of a ray and a triangular facet and returns the hit distance d_h.

3.2. Implementation on GPUs

RTDEM has been implemented using GPU acceleration, including general acceleration with shaders (such as CUDA acceleration of NVIDIA GPUs) and ray tracing (RT) acceleration with RT cores. The overall workflow of RTDEM is outlined in Algorithm 1. GPU shader acceleration has been implemented using CUDA 11.5, while GPU RT acceleration has been implemented using OptiX 7.5. All simulations in the next section were carried out on a Linux/Ubuntu 18.04 Desktop with an NVIDIA Geforce RTX 2080 Ti GPU.

Algorithm 1: Pseudocode of the RTDEM workflow.

```
   Input: Simulation time-steps;
1  foreach time-step do
2      foreach vertex of each particle {GPU shader acceleration} do
3          computing ray position and direction;
4          updating the acceleration structure;
5      foreach ray {GPU RT acceleration} do
6          computing hit distance;
7      foreach vertex of each particle {GPU shader acceleration} do
8          computing contact force;
9      foreach particle {GPU shader acceleration} do
10         integrating particle motion;
```

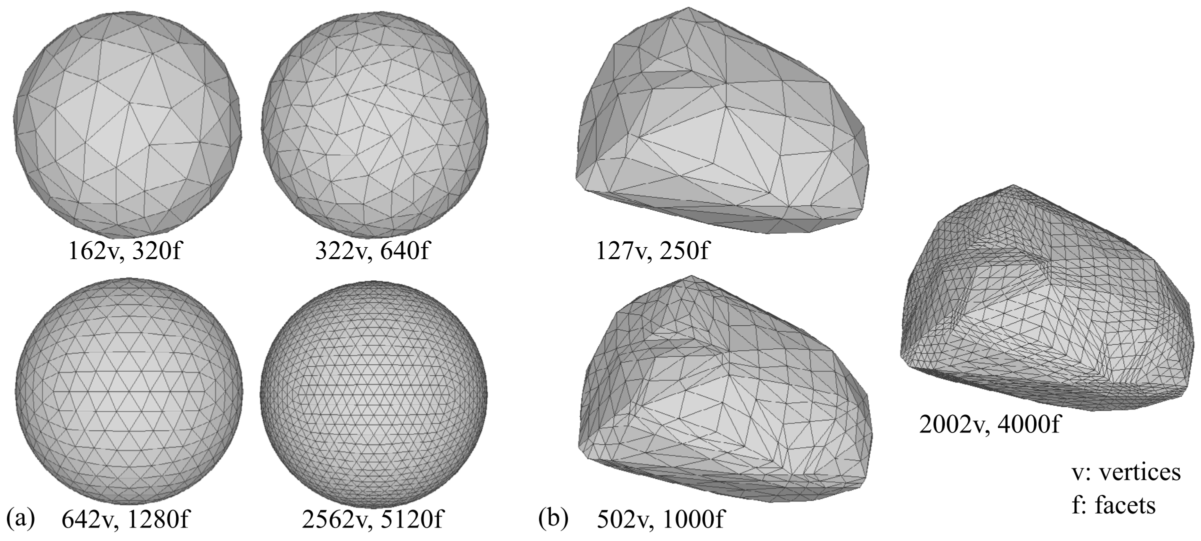

Fig. 8. Exemplified shapes with different mesh resolutions: (a) sphere and (b) sand.

4. Numerical examples

4.1. Particle pair test

In this section, we examine the discrete potential-based contact models, namely the *vertex-potential normal contact force*, *facet-potential normal contact force*, and *combined normal contact force* given in Eqs. (35), (36), and (37), respectively, for both spherical and non-spherical particles. We then perform a sensitivity analysis of mesh resolutions. To this end, we prepare two groups of template meshes with different resolutions, as illustrated in Fig. 8. The spherical particle has a radius of 5 m, while the non-spherical (sand) particle has an equivalent radius of 5 m with the same volume as a sphere. In the particle pair test, two particles are vertically stacked without initial contact, and the top particle moves downwards with a constant velocity, while the bottom particle remains fixed. The repulsive force F_n, i.e., equivalent normal contact force of classical single-point contact models, e.g., Eq. (39), is used to examine the holistic behavior of multi-point contacts between the two particles. The normal contact stiffness k_n is set to 1×10^6 N/m$^{0.5}$ for $\vartheta = \frac{3}{2}$ and 1×10^6 N/m for $\vartheta = 2$. The normal contact forces F_n with different models are normalized by the corresponding normal contact stiffness k_n.

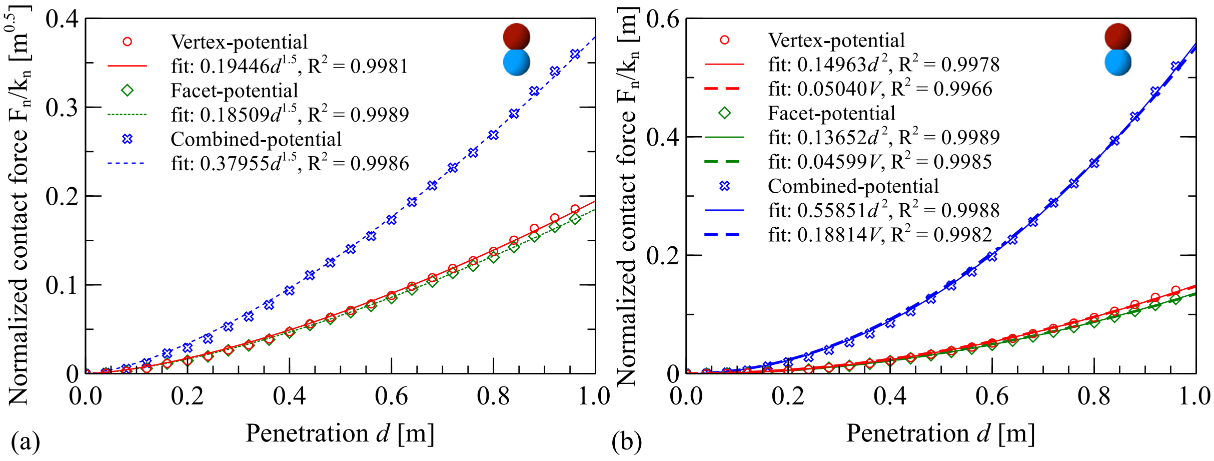

Fig. 9. Variation of normal contact force F_n normalized by normal contact stiffness k_n with penetration depth d for spheres (2562 vertices and 5120 facets for each): (a) $\vartheta = \frac{3}{2}$ and (b) $\vartheta = 2$. Note that V in (b) is the theoretical overlap volume between the contacting spheres.

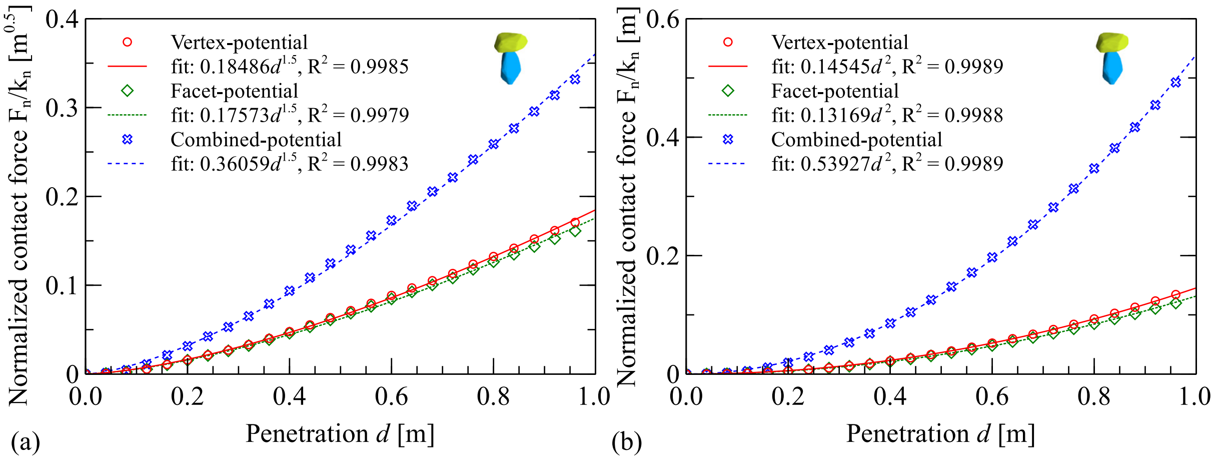

Fig. 10. Variation of normal contact force F_n normalized by normal contact stiffness k_n with penetration depth d for sands (2002 vertices and 4000 facets for each): (a) $\vartheta = \frac{3}{2}$ and (b) $\vartheta = 2$.

4.1.1. Test 1: different contact models

We use the highest-resolution template meshes for the tests, namely a sphere with 2562 vertices and 5120 facets, and sand with 2002 vertices and 4000 facets. The variations of F_n/k_n are plotted in Figs. 9 and 10 for sphere-sphere and sand-sand contact tests, respectively. The results show that there is no significant deviation between the vertex-potential and facet-potential contact forces. To quantify the holistic behavior of the proposed multi-point contact model, a power law fitting equation is introduced below based on the single-point contact model in Eq. (39).

$$F_n/k_n = ad^{\vartheta} \tag{54}$$

where a is a fitting parameter that satisfies $K_n = ak_n$; α in Eq. (39) is set to ϑ, the model parameter of our multi-point contact model. The results show that the fitting Eq. (54) is suitable for fitting the force–displacement relation regardless of particle shape. It is worth mentioning that the local contact force f_n^i is proportional to the $\vartheta - 1$ order of the local penetration d_h, referring to Eqs (32) and (33), while the overall contact force F_n is proportional to the ϑ order of the overall penetration d.

The proposed discrete potential-based contact model has two noteworthy features. Firstly, for $\vartheta = \frac{3}{2}$, the contact model resembles the Hertzian model, where the contact force is proportional to the $\frac{3}{2}$ order of the penetration.

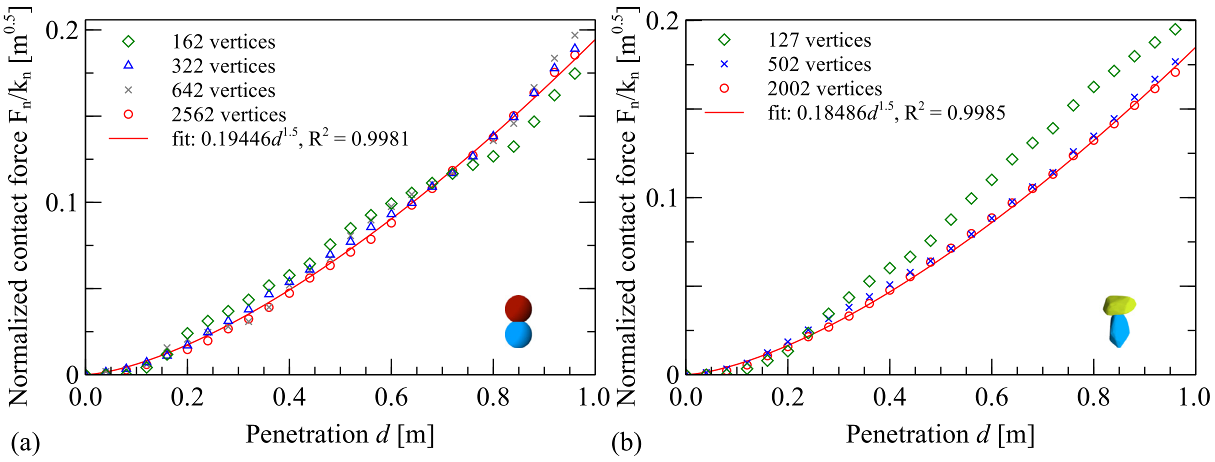

Fig. 11. Normal contact force varying with penetration depth for different vertex numbers: (a) sphere and (b) sand.

Secondly, for $\vartheta = 2$, the contact model resembles a volume-based contact model, especially for sphere-sphere contact. To verify this feature, we introduce the theoretical overlap volume of two identical spheres given by:

$$V = \frac{2\pi}{3}(3r - d)d^2 \tag{55}$$

where r is the radius, and d is the penetration depth. As shown in Fig. 9b, the contact force can be fitted with a linear relation with the overlap volume V.

4.1.2. Test 2: effect of mesh resolution

The vertex-potential contact model with $\vartheta = \frac{3}{2}$ is selected to conduct sensitivity analysis on the resolution of a template mesh. As shown in Fig. 11, the template meshes with 322 vertices and 502 vertices are sufficiently fine to produce quantitatively comparable results with high-resolution meshes for the testing sphere and sand cases, respectively. It suggests that our proposed multi-point contact model is not sensitive to the mesh resolution, benefiting from the introduction of a local geometric measure γ in Eq. (34) that helps weight the contact forces adaptively based on the mesh resolution. It should be noted that the resolution sensitivity may be shape-dependent, and the uniformity of facets may also influence the sensitivity to some extent, as reported in the literature [28]. A comprehensive discussion of this topic is beyond the scope of this work. Interested readers may refer to the literature [43] for a detailed discussion on how to reasonably reduce the vertex number of a triangular mesh.

4.2. Nut spinning into a bolt

The proposed discrete potential-based contact model and the overall RTDEM will be further verified for a dynamic problem: a frictionless nut spinning into a bolt under gravity. During the nut rotation, it involves complicated contact detection within the nut-bolt system, which has been selected as a benchmark for multi-body dynamic simulations in previous studies [28]. With the energy conservation equation of the nut $mgh = 0.5mv_z^2 + 0.5I_z\omega_z^2$ (m: mass; I_z: principal moment of inertia around the vertical axis; v_z: vertical velocity; ω_z: rotational velocity; g: gravitational acceleration, 10 m/s^2 for the default value in this work; h: vertical displacement) and the constraint equations $dh/dt = v_z$ and $2\pi/\omega_z = H/v_z$, the analytical rotational velocity ω_z is given by

$$\omega_z = \frac{2\pi mgHt}{4\pi^2 I_z + mH^2} \tag{56}$$

where H is the vertical displacement of the nut rotating 2π around the bolt.

The simulation setup for the dynamic problem of a frictionless nut spinning into a bolt under gravity is illustrated in Fig. 12. The bolt is fixed, and the nut rotates freely driven by its self-weight. The nut has a mass m of 38.95 kg and a principal moment of inertia I_z of 0.4371 kg m^2. The vertical span between two neighboring bolt tracks H

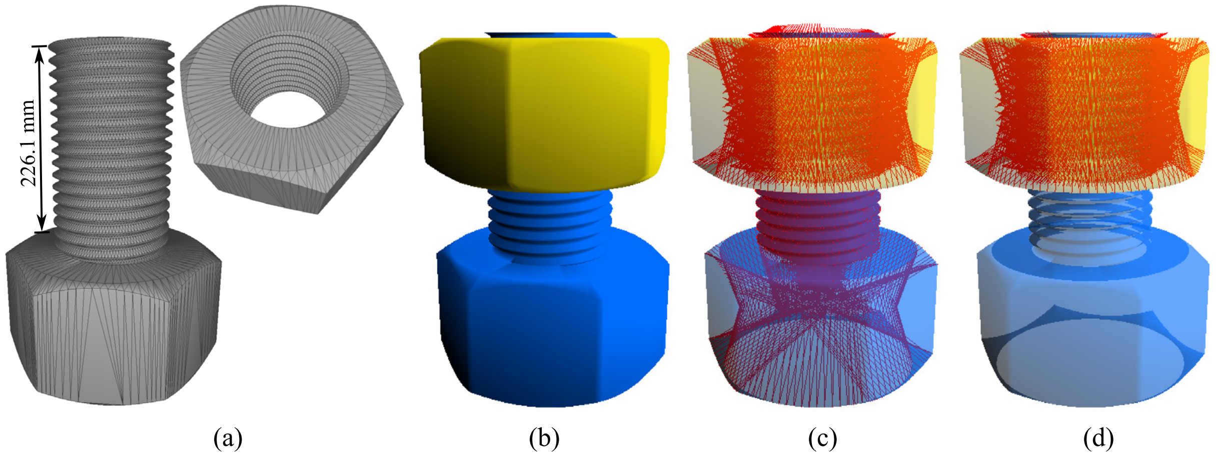

Fig. 12. Setup of the nut spinning test: (a) a nut with 4690 vertices and 9380 facets, and a bolt with 8776 vertices and 17 548 facets; (b) initial configuration; (c) both nut and bolt with ray casting and (d) only the nut with ray casting (The red lines for rays). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

is 15.07 mm, resulting in an analytical rotational acceleration of 2.137 rad/s^2. We utilize the three potential-based contact force models, i.e., *vertex-potential*, *facet-potential*, and *combined-potential* models, in conjunction with two ϑ values ($\frac{3}{2}$ and 2). In the simulation, no friction is applied to any contact, and a small local damping coefficient of 0.025 is used to ensure numerical stability for a given time step of 1×10^{-4} s. The normal contact stiffness k_n is set to 1×10^6 N/m$^{\frac{1}{2}}$ and 1×10^7 N/m for $\vartheta = \frac{3}{2}$ and $\vartheta = 2$, respectively. Note that the value of k_n will not influence the result, except for numerical stability, since it vanishes in the analytical solution.

As shown in Fig. 13, the facet-potential contact model outperforms the other two models, yielding nearly consistent results with the analytical solution. The combined-potential contact model performs better than the vertex-potential model in most cases. Moreover, the results of only nut casting rays are nearly identical to those of both nut and bolt casting rays. The average simulation speeds are 15,813 steps/s and 9724 steps/s for the only-nut-casting and both-nut-and-bolt-casting cases, respectively, resulting in a speedup of 250 $\sim$ with respect to previous SDF-DEM simulations [28] with 5196 and 8170 facets for the nut and the bolt, respectively (almost half problem size of this work). It is important to acknowledge that comparing the performance of RTDEM and SDF-DEM based on different computing architectures, one on GPU and the other on CPU, may not provide a fair evaluation. Therefore, it would be more meaningful to conduct a comprehensive comparison on the same GPU computing architecture. However, such a comparison is beyond the scope of this work. Interested readers are referred to the literature [28] for a comprehensive investigation of computational performance of SDF-DEM on a CPU computing architecture. Moreover, it suggests that casting rays from only one of the contacting particles can significantly improve the computational performance with comparable results to casting rays from both particles. However, for $\vartheta = \frac{3}{2}$, there is a moderate difference between simulation results of the two ray-casting cases when using the vertex-potential or combined-potential contact models. Furthermore, the matter of selecting which particle to employ for ray casting in many-particle systems is still an open question that we intend to address in future research. This issue is analogous to the one encountered in the conventional master–slave particle scheme when utilizing node-to-surface algorithms. It is noteworthy that previous studies, such as [28], have demonstrated that swapping the roles of master and slave particles results in a negligible deviation in the contact force. This negligible deviation does not significantly affect the mechanical response of the particle pair, provided that the mesh resolution is adequate. In this work, however, we will use both contacting particles for ray casting to eliminate potential numerical issues caused by using single particle for ray casting. In summary, the facet-potential contact model is the most robust, regardless of the ray-casting cases and the model order parameter ϑ.

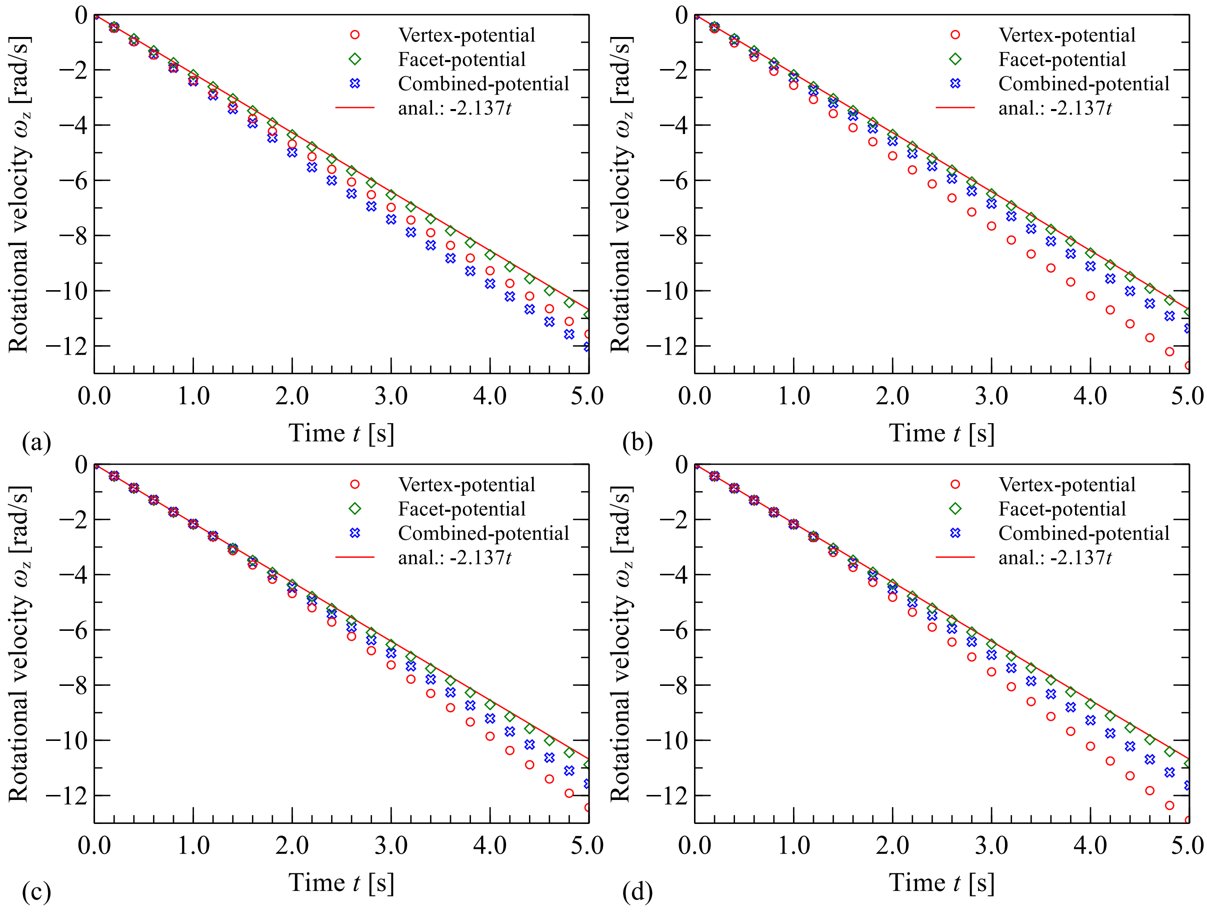

Fig. 13. Rotational velocity of the nut driven by its self weigh with: both nut and bolt casting rays for (a) $\vartheta = \frac{3}{2}$ and (b) $\vartheta = 2$, and only the nut casting rays for (c) $\vartheta = \frac{3}{2}$ and (d) $\vartheta = 2$.

4.3. Granular packing

This section examines the robustness and performance of our RTDEM in simulating frictional granular systems with respect to varying particle shape. The template meshes (127v and 502v) of sand particle illustrated in Fig. 8 are adopted to investigate the effect of mesh resolution, and another two template meshes, i.e., banana and pear, are prepared as illustrated in Fig. 14. Both the vertex-potential and facet-potential contact models are employed, and the contact model parameter ϑ is set to $\frac{3}{2}$. We only show snapshots of the packing configurations for simulations with the vertex-potential contact model to observe the numerical stability. Kinetic energy and elapsed time in computation are then quantitatively examined for simulations with both contact models.

4.3.1. Packing protocol

The following simulation protocol is employed for granular packing unless stated otherwise. The initial packing configuration is set up at a $5 \times 5 \times 20$ lattice grid with a gap of 0.11 m, and then particles deposit freely into a box under gravity. The box has a square base of 1.42×1.42 m^2. For particle–particle contacts, the normal contact stiffness k_n is set to 1×10^6 N/m$^{\frac{1}{2}}$, and the tangential contact stiffness k_s is set to 1×10^5 N/m. For the particle-box contacts, k_n and k_s are 1×10^8 N/m$^{\frac{1}{2}}$ and 1×10^5 N/m, respectively. Both coefficients of friction for particle–particle and particle-box contacts are set to 0.3. The material mass density of particles is 2650 kg/m^3. A local damping coefficient of 0.3 is used to facilitate energy dissipation. The packing simulation runs for 30 000 steps with a fixed timestep of 1×10^{-4} s.

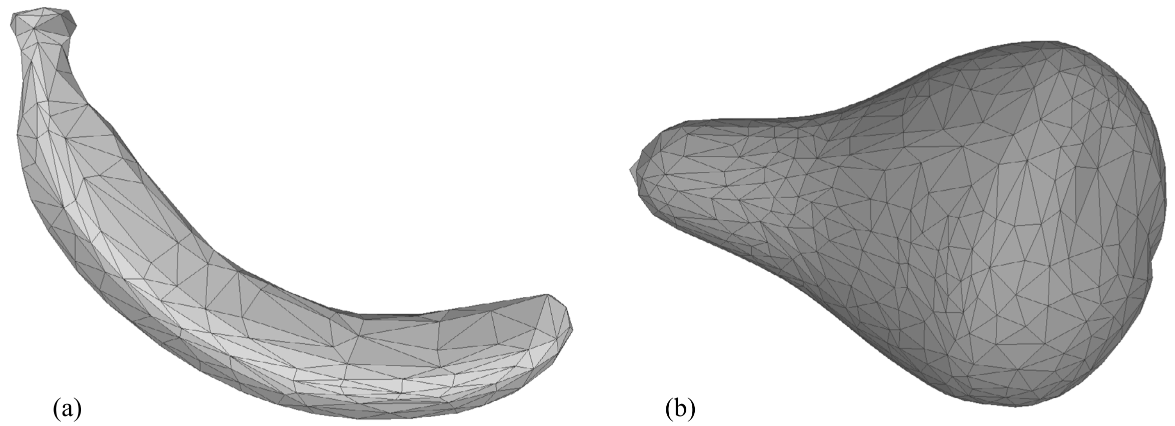

Fig. 14. Template meshes: (a) a banana with 252 vertices and 500 facets, (b) a pear with 750 vertices and 1496 facets.

4.3.2. Packing of sand particles with different mesh resolutions

Fig. 15 shows the packing configurations of 500 sand particles with an equivalent size of 5.0 cm. The packings use the 127v and 502v meshes, which have over 63,500 and 251,000 vertices, respectively. Although there is a moderate difference between the final configurations for the two mesh resolutions, the packing simulation is numerically stable for the mesh with only 127 vertices.

4.3.3. Packing of bananas and pears

Figs. 16 and 17 show the packing configurations of 500 bananas (with an equivalent size of 5.2 cm) and 500 pears (with an equivalent size of 8.3 cm), respectively. The packings have over 126,000 and 375,000 vertices, respectively. The packing simulations for these two distorted shapes are numerically stable. Moreover, we simulate a packing with a mixture of 400 bananas and 100 pears, and the configurations are shown in Fig. 18. No numerical instability has been observed so far, confirming the robustness of RTDEM in simulating elongated particles that can result in extreme configurations, as depicted in Fig. 4(b).

4.3.4. Packing of sand particles with varying sizes

Two more examples of packing sand particles (using the mesh template 127v) are showcased to further examine the robustness of RTDEM in simulating polydisperse granular systems with varying particle size. One example has grain sizes uniformly distributed between 2.5 and 5.0 cm (denoted as gsd packing), and the other has binary sizes of 1.0 and 5.0 cm (denoted as bi-mixture packing). For the bi-mixture packing, the normal contact stiffness k_n is set to 1×10^5 N/m$^{\frac{1}{2}}$ and 1×10^4 N/m$^{\frac{1}{2}}$ for the particle-box contacts and the inter-particle contacts of fine particles, respectively. Moreover, there are 100 coarse particles and 9375 fine particles. The rest of the simulation setup is the same as the previous examples. Figs. 19 and 20 show the packing configurations at different time instants for the gsd packing and the bi-mixture packing, respectively. It is observed that the simulations are numerically stable during the course of packing.

4.3.5. Energy dissipation during packing

The numerical stability has been observed in the simulations above. To quantify the stability, the variation of kinetic energy $\sum(mv^2/2 + I\omega^2/2)$ is tracked and normalized by the initial potential energy $\sum(mgh)$ (h is the particle height above the box base). Fig. 21 shows the variations of kinetic energy for all the simulations. The kinetic energy evolves almost the same for the vertex-potential and facet-potential contact models, but there is a significant deviation at the peak for a low resolution of particle mesh, such as sand-127v. Moreover, the kinetic energy exhibits a single-peak evolution during packing, except for the packing of pears, implying that the evolution shape of kinetic energy is somewhat related to particle shape. However, a comprehensive investigation of the effect of particle shape is not presented here to avoid any possible distractions.

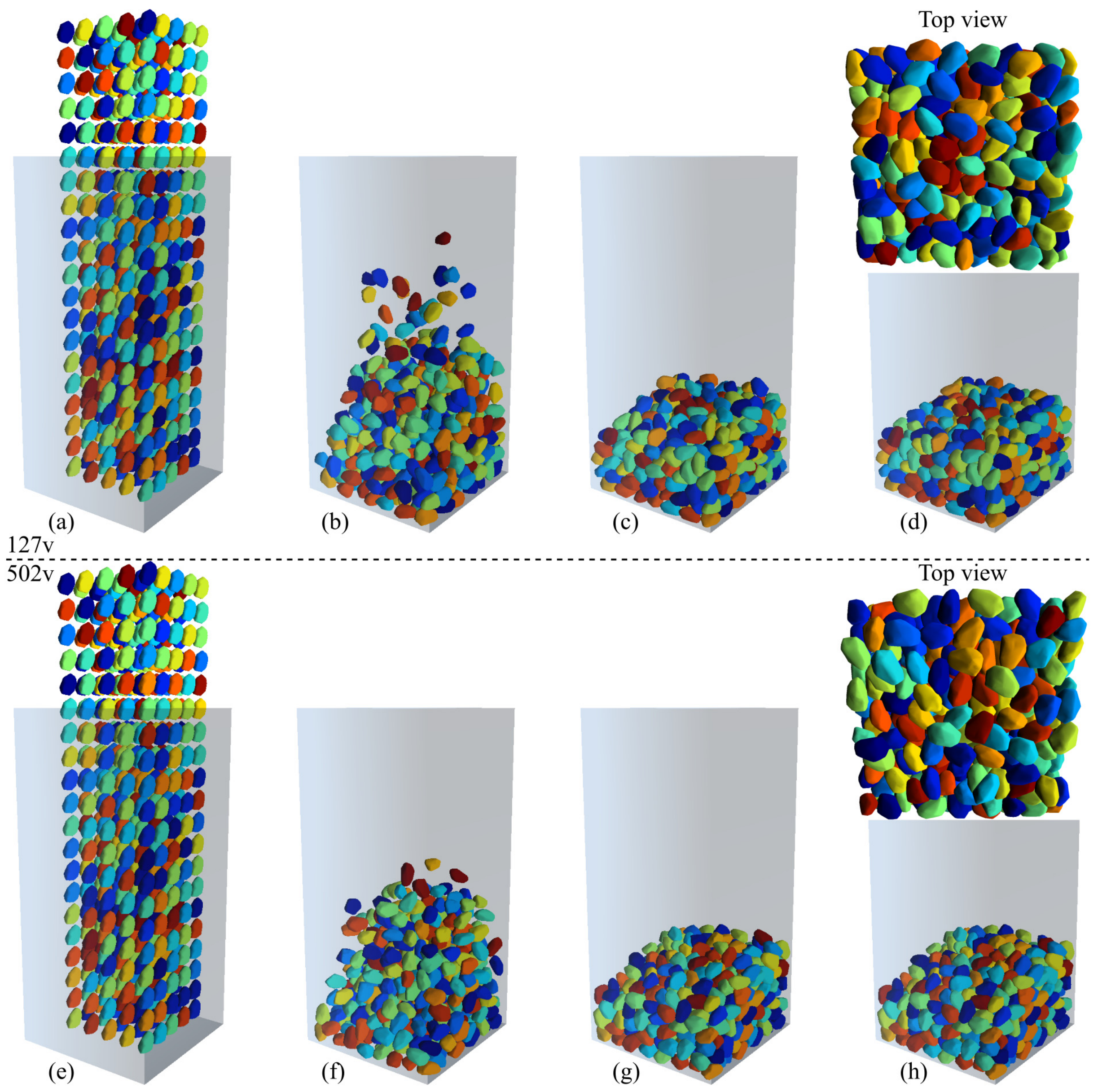

Fig. 15. Packing of sand particles within a box at different time instants: (a) 127v and (e) 502v at $t = 0$ s, (b) 127v and (f) 502v at $t = 1$ s, (c) 127v and (g) 502v at $t = 2$ s and (d) 127v and (h) 502v at $t = 3$ s. Note: 127v for 127 vertices and 502v for 502 vertices.

4.3.6. Computational efficiency

Computational efficiency is another important indicator for evaluating the performance of RTDEM. The elapsed time for all simulations is summarized in Table 1, where the problem size is roughly quantified by the total vertex number to allow for an intuitive comparison between simulations. Note that particle shape and size distribution are also major factors influencing computational efficiency. Based on Table 1, we can conclude that the vertex-potential contact model is slightly more computationally efficient than the facet-potential contact model for most of the simulations. The reason is that the facet-potential contact model requires extra computation to access the facet normal. However, considering the overall performance, including accuracy, robustness, and efficiency, it is recommended to use the facet-potential contact model.

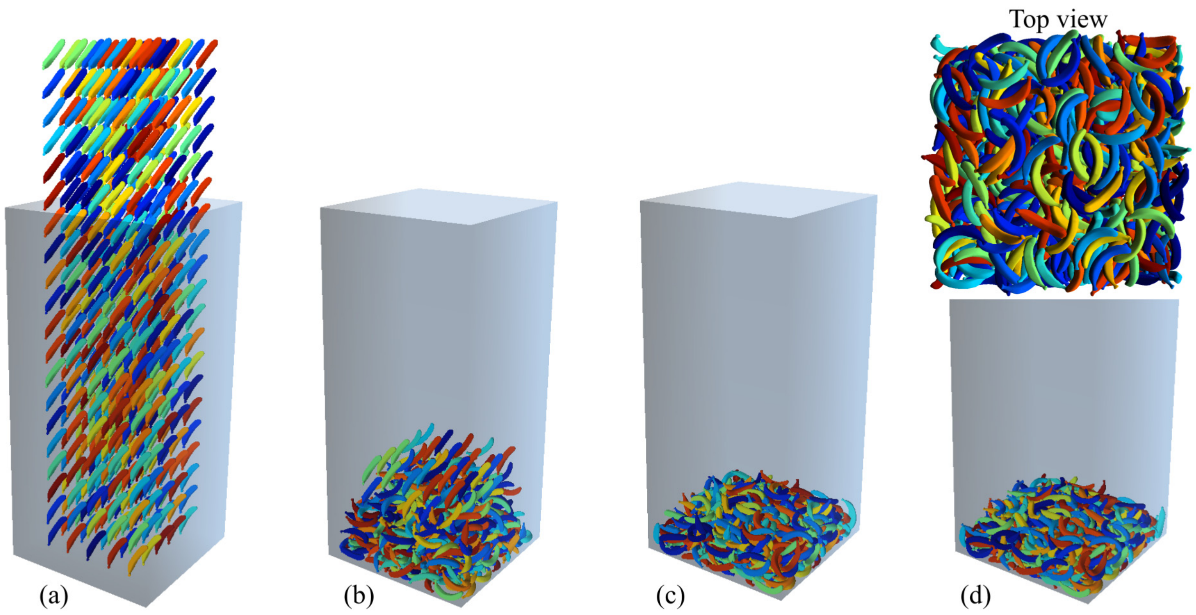

Fig. 16. Packing of bananas within a box at different time instants: (a) $t = 0$ s, (b) $t = 1$ s, (c) $t = 2$ s and (d) $t = 3$ s.

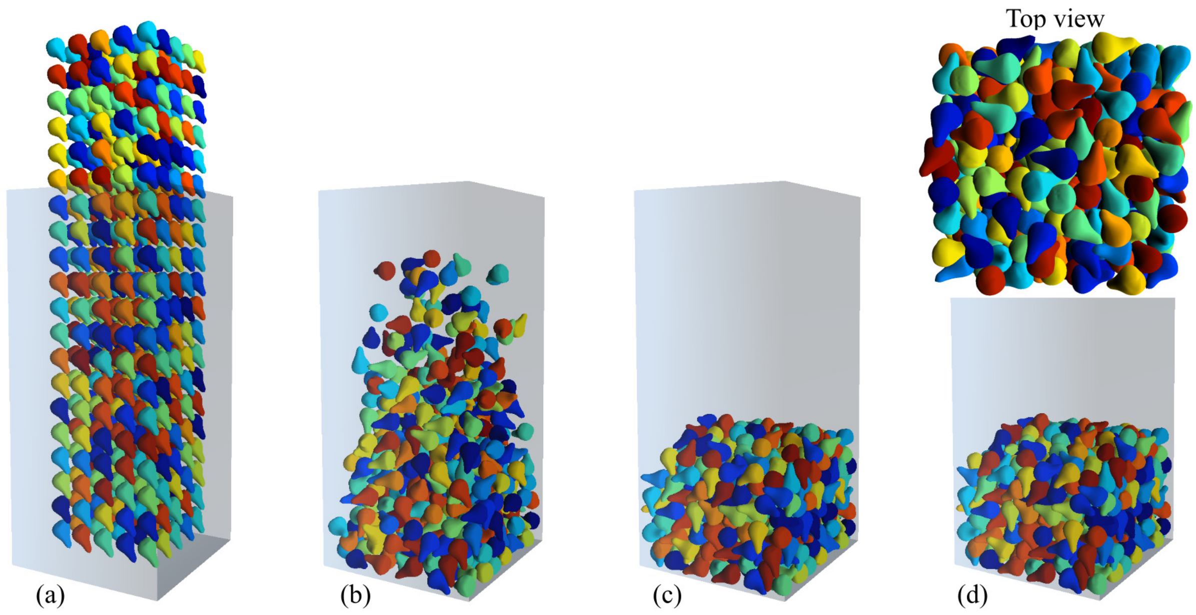

Fig. 17. Packing of pears within a box at different time instants: (a) $t = 0$ s, (b) $t = 1$ s, (c) $t = 2$ s and (d) $t = 3$ s.

4.3.7. Packing of bi-mixture with an extreme size ratio

The proposed computational framework is able to efficiently simulate dynamic problems of both mono-disperse and poly-disperse particle systems with arbitrarily shapes as demonstrated above. An additional extreme case is presented here to further showcase the performance of the proposed framework. In the simulation, 0.2 million fine particles deposit above a coarse particle in a box. The equivalent sizes of the coarse and fine particles are 1 m and 1 cm, respectively, resulting in an extreme size ratio of 100 : 1. All particles have the same shape with the

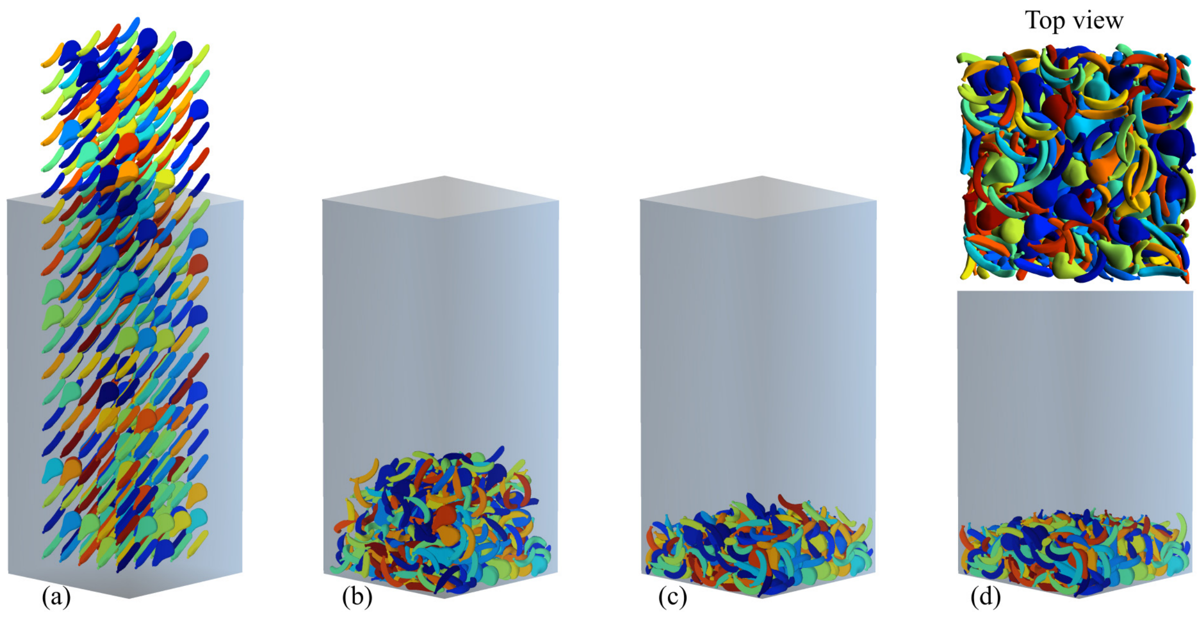

Fig. 18. Mixture packing of bananas and pears within a box at different time instants: (a) $t = 0$ s, (b) $t = 1$ s, (c) $t = 2$ s and (d) $t = 3$ s.

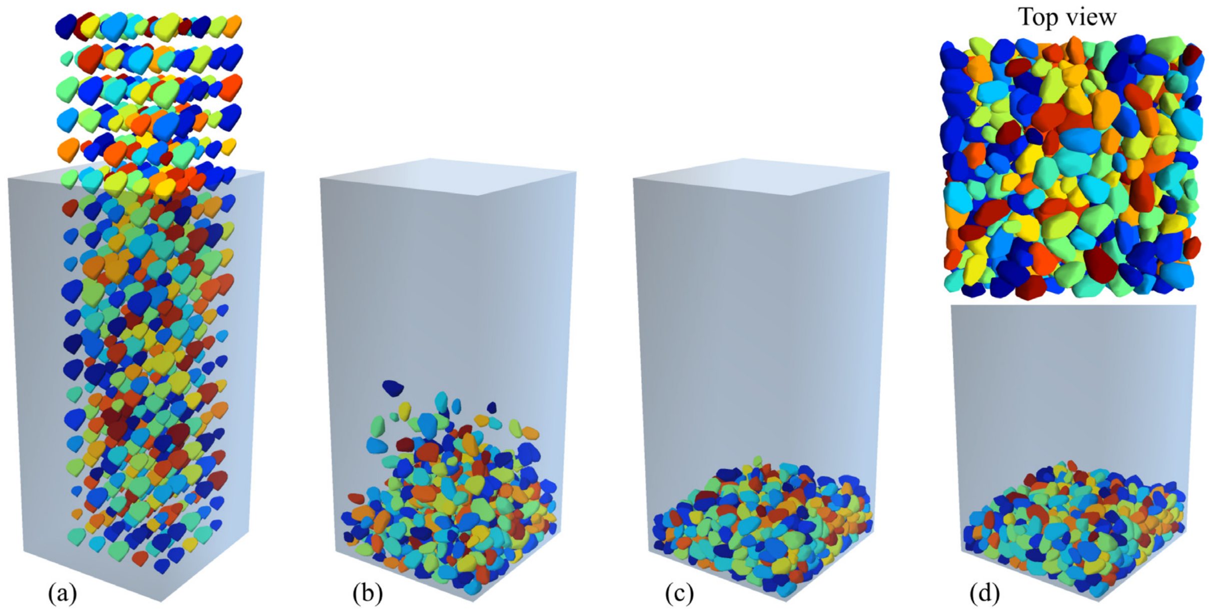

Fig. 19. Packing of sand particles with a uniform grain size distribution (gsd packing) within a box at different time instants: (a) $t = 0$ s, (b) $t = 1$ s, (c) $t = 2$ s and (d) $t = 3$ s.

sand127v template mesh. The facet-potential contact model is adopted with $\vartheta = \frac{3}{2}$. The normal contact stiffnesses are 1×10^8 N/m$^{\frac{1}{2}}$ and 1×10^4 N/m$^{\frac{1}{2}}$ for the coarse and fine particles, respectively, while the tangential contact stiffnesses are 1×10^5 N/m and 1×10^4 N/m, respectively. The other simulation parameters are the same as in the previous simulations. The simulation runs for 30,000 steps with a fixed timestep of 1×10^{-4} s, and the snapshots of particle configuration are shown in Fig. 22. The entire simulations takes 29.4 min on a gaming NVIDIA RTX 2080

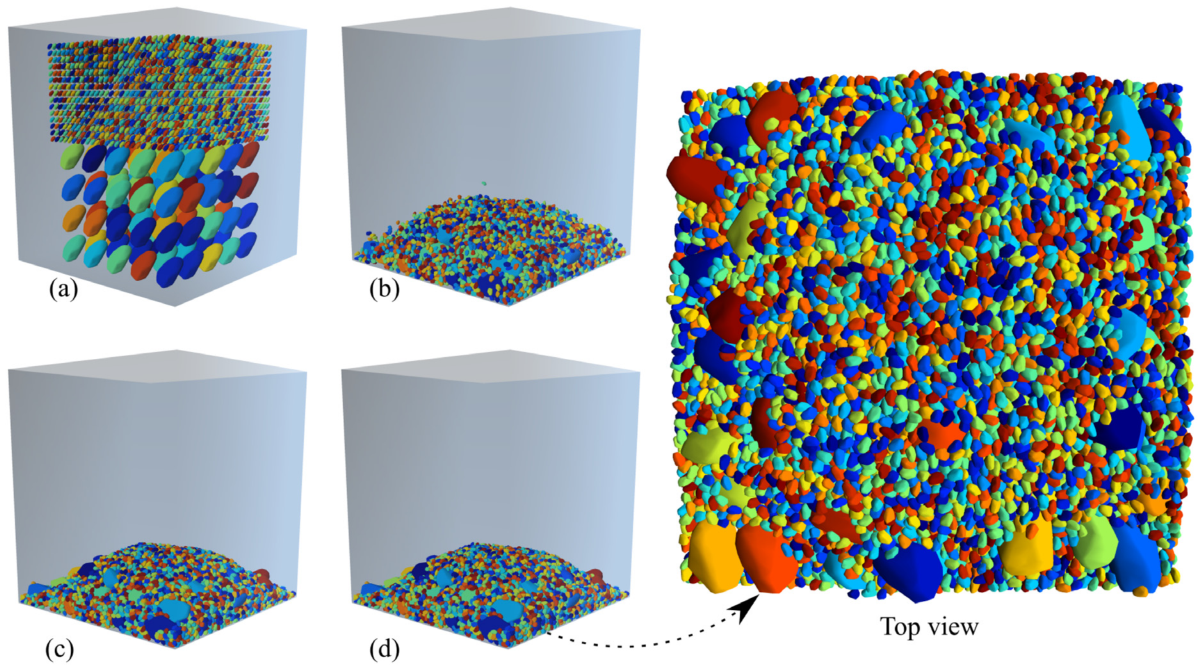

Fig. 20. Packing of bi-mixture sand particles (bi-mixture packing) within a box at different time instants: (a) $t = 0$ s, (b) $t = 1$ s, (c) $t = 2$ s and (d) $t = 3$ s.

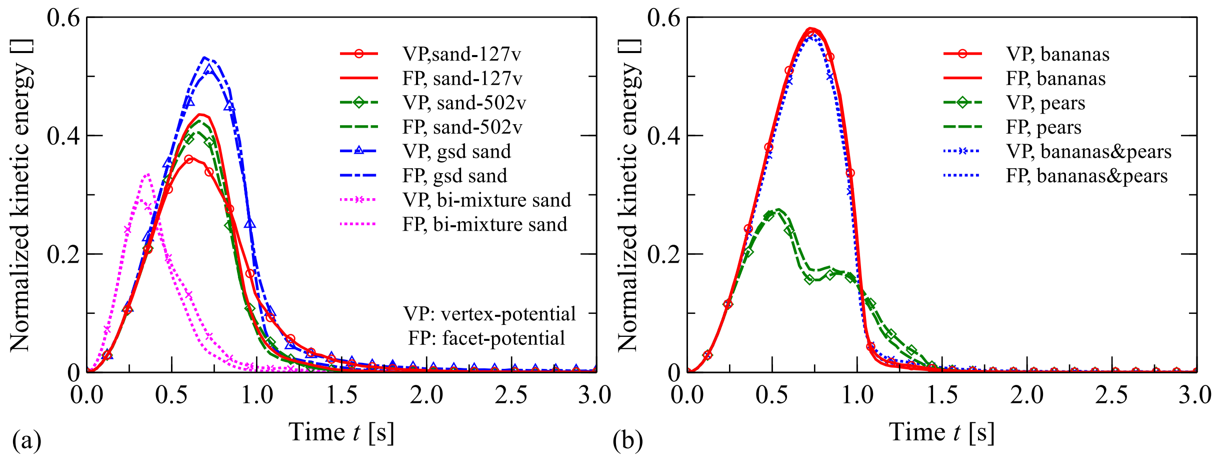

Fig. 21. Kinetic energy normalized by the initial gravitational potential energy during packing of: (a) sand particles and (b) bananas and pears.

Ti GPU (with 11 GB memory), and the GPU memory usage is 2323 MB. These results suggest that the proposed framework offers an efficient solution for modeling arbitrarily shaped particles, even with a large size ratio.

4.4. Column collapse

Experimental tests of column collapse are carried out to validate the corresponding simulations in RTDEM. Two groups of granular columns are prepared with mono-disperse particles: one with 1 540 sand particles as introduced in previous sections and the other with 728 poly-superellipsoidal particles. A poly-superellipsoidal particle has a

Table 1
Elapsed time of RTDEM simulations for granular packing on an NVIDIA RTX 2080 Ti GPU.

Test	Total vertex #	Virtual time [s]	Elapsed time [s]	
			Vertex-potential	Facet-potential
Sand, 127v	63 500	3	3.53	3.61
Sand, 502v	251 000	3	7.68	7.75
Bananas	126 000	3	5.39	5.43
Pears	375 000	3	10.68	10.66
Bananas & pears	175 800	3	7.29	7.52
Gsd sand	63 500	3	3.58	3.69
Bi-mixture sand	1 203 325	3	45.87	44.84

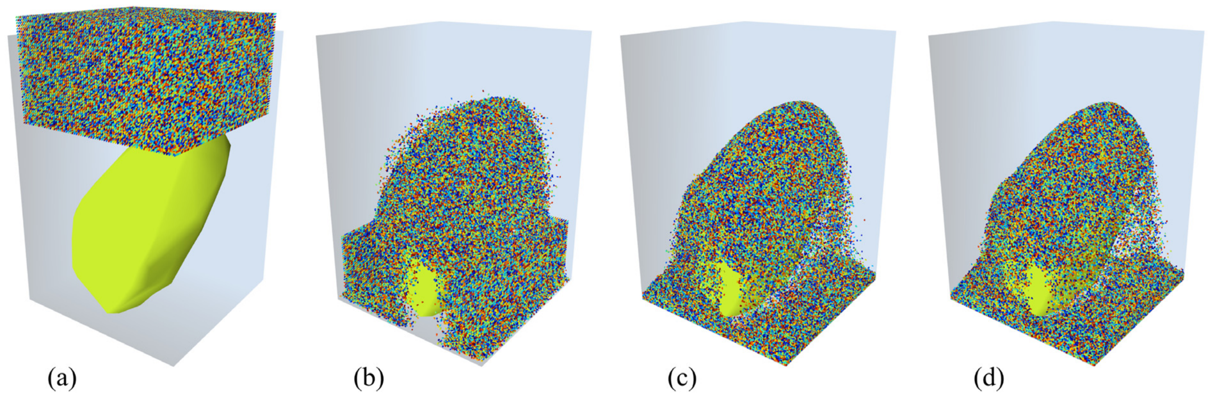

Fig. 22. Packing of bi-mixture particles with a size ratio of 100 : 1 within a box at different time instants: (a) $t = 0$ s, (b) $t = 1$ s, $t = 2$ s and $t = 3$ s.

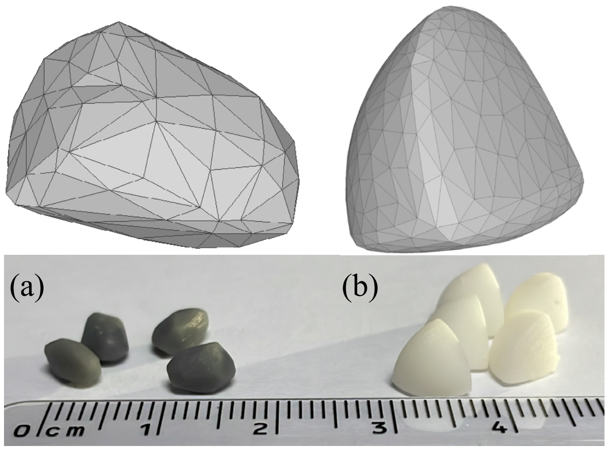

Fig. 23. Template meshes (upper) and 3D printed particles (lower): (a) sand with 127 vertices and 250 facets; (b) poly-superellipsoid with 475 vertices and 946 facets.

smooth surface, as described mathematically in the literature [9]. Particles are 3D-printed with a resolution of 50 μm using a Formlabs Form 3L printer. Fig. 23 shows the template meshes and their 3D-printed samples.

The 3D-printed particles are poured into an acrylic container with a base of 4 cm × 3.5 cm. The granular column collapses to approach an equilibrium state (final configuration) after removing a side wall of the container. Both initial and final configurations are plotted in Fig. 24. It can be seen that the poly-superellipsoidal particles run out further than the sand particles due to their smoother surfaces. The slopes are 27.7° and 21.9° for sand and

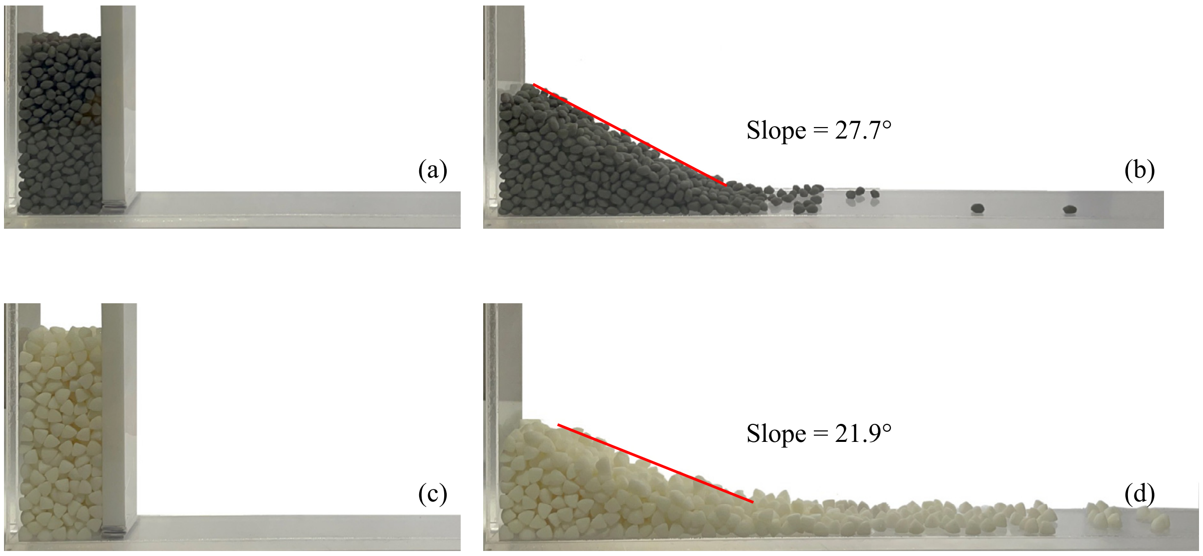

Fig. 24. Column collapse of 3D printed particles: (a) initial and (b) final configurations for sand particles; (c) initial and (d) final configurations for poly-superellipsoidal particles.

poly-superellipsoidal particles, respectively. The final profiles are taken as benchmarks to validate the numerical simulations. Note that it is inevitable that there can be a moderate deviation between the profiles of experimental repetitions.

In the proposed numerical model of RTDEM, model parameters such as contact stiffness, friction coefficient and damping coefficient may influence the final profile of a granular column after collapse. However, a comprehensive parametric study is not presented here to avoid any possible distractions. Moreover, an elaborated calibration on those model parameters is required to quantitatively reproduce the experimental results, e.g., using an iterative Bayesian filtering framework [44], which is beyond the scope of this work. For simplicity, the friction coefficient μ is taken as a variable for sensitive analysis, while the values of the other model parameters are selected as follows: mass density 1200 kg/m^3, normal and tangential contact stiffness $k_n = 5 \times 10^2$ N/m$^{\frac{1}{2}}$, $k_s = 1 \times 10^4$ N/m for inter-particle contacts and $k_n = 5 \times 10^3$ N/m$^{\frac{1}{2}}$, $k_s = 1 \times 10^5$ N/m for particle-box contacts, and a damping coefficient of 0.1. The vertex-potential contact model with $\vartheta = \frac{3}{2}$ is adopted. Each simulation runs for 100 000 steps with a fixed timestep of 1×10^{-5} s.

Fig. 25 shows the snapshots of the sand column during collapsing. Two friction coefficients $\mu = 0.3$ and $\mu = 0.2$ are adopted, yielding two slopes of 29.4° and 26.3°, respectively, at the final state. The final profile of the simulation with $\mu = 0.2$ closely resembles that of the experiment. Fig. 26 shows the configurations of poly-superellipsoidal particles at different time instants. The final slopes are 25.2° and 21.0° with friction coefficients of $\mu = 0.2$ and $\mu = 0.15$, respectively. It is evident that the simulated profile is expected to match the experimental one with a slightly increased friction coefficient from 0.15. In summary, RTDEM is capable of reproducing experimental results without requiring significant effort to tune the model parameters.

5. Conclusions

Our proposed computational framework is shown capable of efficiently modeling arbitrarily shaped particles. This framework has three major features: (1) Contact detection and resolution are solved by our proposed ray-tracing approach; (2) Three potential-based contact models, including the vertex-potential, the facet-potential, and their combined models, make the simulations numerically stable, with the facet-potential model being the most robust; (3) The potential-based models can be seamlessly combined with the ray-tracing contact detection, making the overall framework suitable for parallel computing in advanced computing systems, such as GPUs. Specifically, both shader and RT cores on GPUs can be fully leveraged to accelerate simulations, making efficient use of the computing

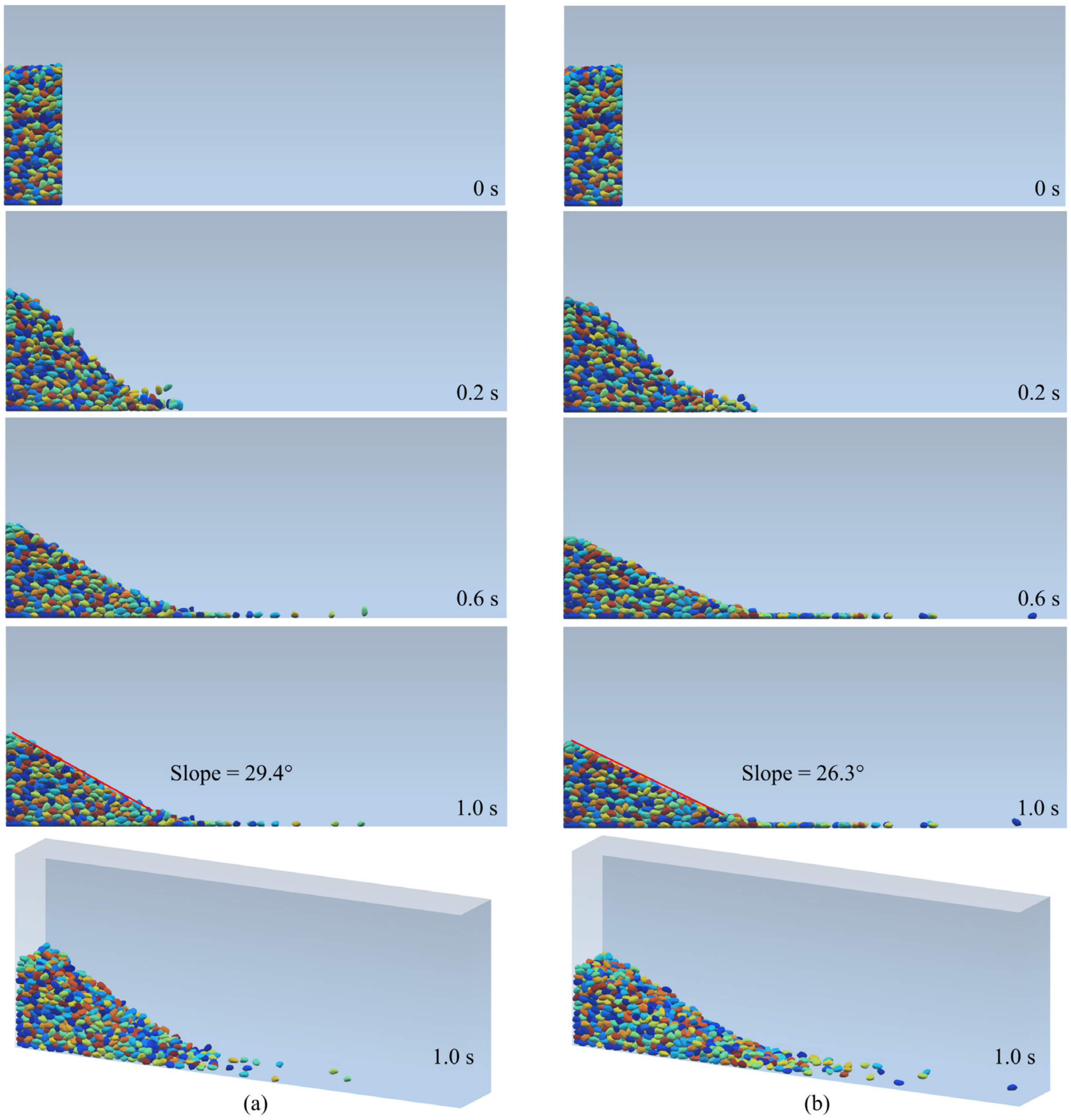

Fig. 25. Configurations of sand particles in simulations of column collapse with different friction coefficients: (a) $\mu = 0.3$ and (b) $\mu = 0.2$.

capacity of GPUs. While the implementation of RTDEM is for GPUs, the proposed framework is compatible with conventional multi-core or multi-node CPU computing systems, as well as with FPGA computing systems.

The proposed framework can be extended and enhanced in several key aspects. (1) The current framework necessitates ray casting at each time step. However, a promising avenue for optimization involves the introduction of a local Verlet buffer (skin). The implementation of a Verlet buffer can help reduce the frequency of ray casting, leading to improved computational efficiency without compromising the accuracy of the simulation. (2) While the numerical results demonstrate insensitivity to the mesh resolution or vertex number once an adequate number of

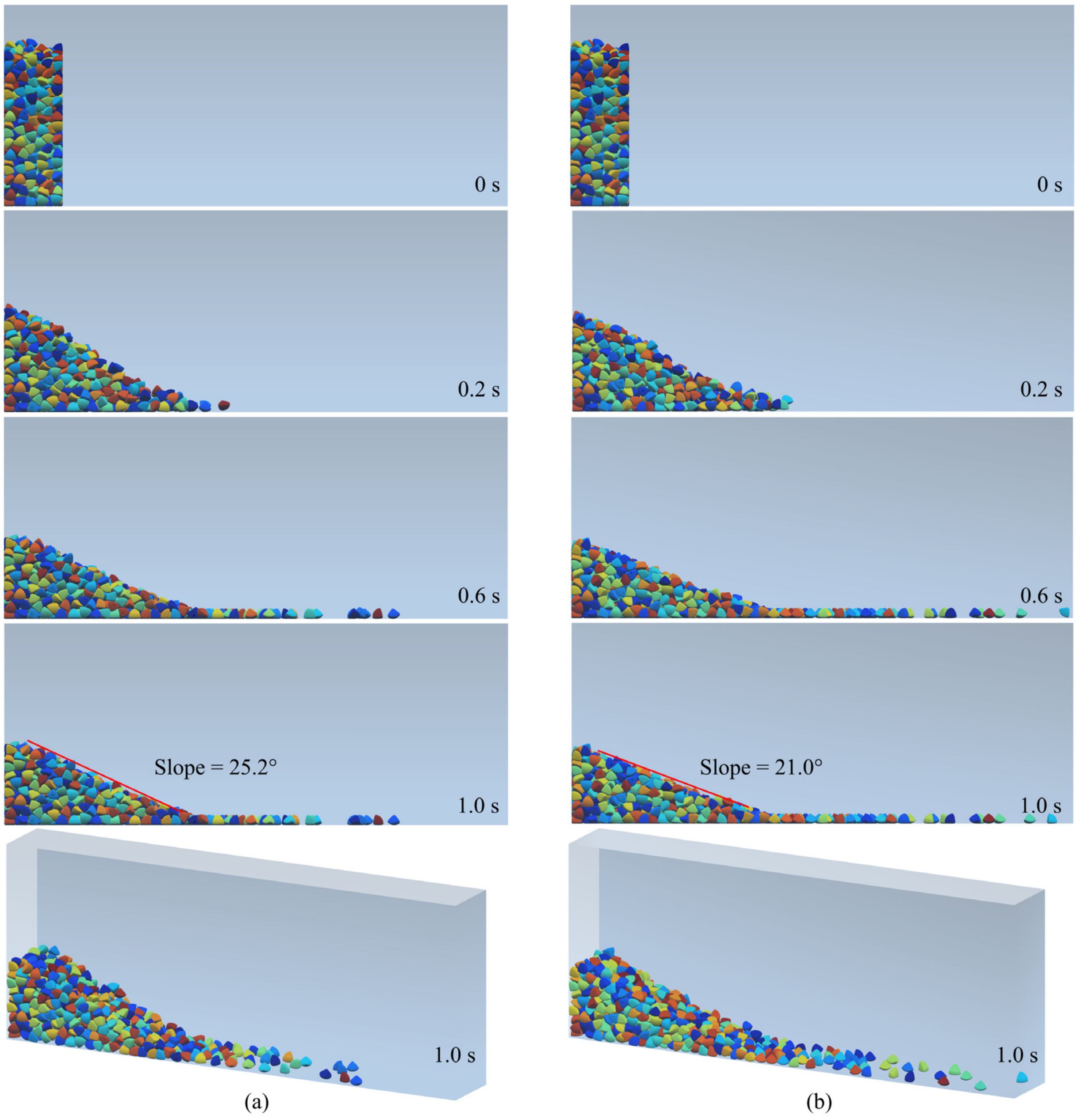

Fig. 26. Configurations of poly-superellipsoidal particles in simulations of column collapse with different friction coefficients: (a) $\mu = 0.2$ and (b) $\mu = 0.15$.

vertices is employed, it is essential to conduct a theoretical analysis to determine such a critical vertex number. Establishing this critical threshold would provide a valuable guideline for selecting an appropriate mesh resolution in simulations using this framework. (3) Further efforts should be dedicated to establishing a theoretical relationship among contact stiffness, mesh resolution, and time step. Strengthening the theoretical foundation of the framework would enable a more rigorous understanding of the interplay between these parameters. This, in turn, would facilitate better-informed choices of contact stiffness, mesh resolution, and time step, leading to improved accuracy and reliability of the simulation results.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgments

This work was financially supported by National Natural Science Foundation of China (Grant Nos. 11972030 and 51909095), Research Grants Council of Hong Kong (by GRF Projects No. 16206322 and No. 16211221, and by CRF Project No. C7082-22G), Guangdong Basic and Applied Basic Research Foundation (2022A1515010848), Science and Technology Program of Guangzhou (202201010392). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the financial bodies.

References

[1] G.-C. Cho, J. Dodds, J.C. Santamarina, Particle shape effects on packing density, stiffness, and strength: natural and crushed sands, J. Geotech. Geoenviron. Eng. 132 (5) (2006) 591–602.
[2] J. Nie, Y. Cui, K. Senetakis, D. Guo, Y. Wang, G. Wang, P. Feng, H. He, X. Zhang, X. Zhang, et al., Predicting residual friction angle of lunar regolith based on Chang'e-5 lunar samples, Sci. Bull. 68 (7) 730–739.
[3] E. Deal, J.G. Venditti, S.J. Benavides, R. Bradley, Q. Zhang, K. Kamrin, J.T. Perron, Grain shape effects in bed load sediment transport, Nature 613 (7943) (2023) 298–302.
[4] Y. Feng, Thirty years of developments in contact modelling of non-spherical particles in dem: a selective review, Acta Mech. Sinica 39 (1) (2023) 722343.
[5] T.-T. Ng, Particle shape effect on macro-and micro-behaviors of monodisperse ellipsoids, Int. J. Numer. Anal. Methods Geomech. 33 (4) (2009) 511–527.
[6] H.B.K. Nguyen, M.M. Rahman, A.B. Fourie, Undrained behaviour of granular material and the role of fabric in isotropic and K0 consolidations: DEM approach, Géotechnique 67 (2) (2016) 153–167.
[7] S. Zhao, N. Zhang, X. Zhou, L. Zhang, Particle shape effects on fabric of granular random packing, Powder Technol. 310 (2017) 175–186.
[8] C. Wellmann, C. Lillie, P. Wriggers, Comparison of the macroscopic behavior of granular materials modeled by different constitutive equations on the microscale, Finite Elem. Anal. Des. 44 (5) (2008) 259–271.
[9] S. Zhao, J. Zhao, A poly-superellipsoid-based approach on particle morphology for dem modeling of granular media, Int. J. Numer. Anal. Methods Geomech. 43 (13) (2019) 2147–2169.
[10] Z. Lai, Q. Chen, L. Huang, Fourier series-based discrete element method for computational mechanics of irregular-shaped particles, Comput. Methods Appl. Mech. Engrg. 362 (2020) 112873.
[11] C. Boon, G. Houlsby, S. Utili, A new algorithm for contact detection between convex polygonal and polyhedral particles in the discrete element method, Comput. Geotech. 44 (2012) 73–82.
[12] L. Liu, S. Ji, A new contact detection method for arbitrary dilated polyhedra with potential function in discrete element method, Internat. J. Numer. Methods Engrg. 121 (24) (2020) 5742–5765.
[13] M.V. Craveiro, A.G. Neto, P. Wriggers, Contact between rigid convex nurbs particles based on computer graphics concepts, Comput. Methods Appl. Mech. Engrg. 386 (2021) 114097.
[14] S. Zhao, J. Zhao, SudoDEM: Unleashing the predictive power of the discrete element method on simulation for non-spherical granular particles, Comput. Phys. Comm. 259 (2021) 107670.
[15] E.G. Gilbert, D.W. Johnson, S.S. Keerthi, A fast procedure for computing the distance between complex objects in three-dimensional space, IEEE J. Robot. Autom. 4 (2) (1988) 193–203.
[16] M.I. Lourakis, A brief description of the levenberg-marquardt algorithm implemented by levmar, Found. Res. Technol. 4 (1) (2005) 1–6.
[17] A.G. Neto, P. Wriggers, Discrete element model for general polyhedra, Comput. Part. Mech. (2022) 1–28.
[18] G. Mollon, J. Zhao, 3D generation of realistic granular samples based on random fields theory and fourier shape descriptors, Comput. Methods Appl. Mech. Engrg. 279 (2014) 46–65.
[19] B. Zhou, J. Wang, Generation of a realistic 3D sand assembly using x-ray micro-computed tomography and spherical harmonic-based principal component analysis, Int. J. Numer. Anal. Methods Geomech. 41 (1) (2017) 93–109.
[20] E.G. Nezami, Y.M.A. Hashash, D. Zhao, J. Ghaboussi, Shortest link method for contact detection in discrete element method, Int. J. Numer. Anal. Methods Geomech. 30 (8) (2006) 783–801.
[21] E. Azéma, F. Radjai, F. Dubois, Packings of irregular polyhedral particles: Strength, structure, and effects of angularity, Phys. Rev. E 87 (6) (2013) 062203.

[22] P. Wriggers, G. Zavarise, On contact between three-dimensional beams undergoing large deflections, Commun. Numer. Methods. Eng. 13 (6) (1997) 429–438.
[23] T.A. Laursen, Computational Contact and Impact Mechanics: Fundamentals of Modeling Interfacial Phenomena in Nonlinear Finite Element Analysis, Springer Science & Business Media, 2003.
[24] R. Kawamoto, E. Andò, G. Viggiani, J.E. Andrade, Level set discrete element method for three-dimensional computations with triaxial case study, J. Mech. Phys. Solids 91 (2016) 1–13.
[25] J. Zhao, S. Zhao, S. Luding, The role of particle shape in computational modelling of granular matter, Nat. Rev. Phys. (2023) 1–21, http://dx.doi.org/10.1038/s42254-023-00617-9.
[26] S. Zhao, X. Zhou, W. Liu, Discrete element simulations of direct shear tests with particle angularity effect, Granul. Matter 17 (2015) 793–806.
[27] Y. Feng, An energy-conserving contact theory for discrete element modelling of arbitrarily shaped particles: Contact volume based model and computational issues, Comput. Methods Appl. Mech. Engrg. 373 (2021) 113493.
[28] Z. Lai, S. Zhao, J. Zhao, L. Huang, Signed distance field framework for unified dem modeling of granular media with arbitrary particle shapes, Comput. Mech. 70 (4) (2022) 763–783.
[29] J.A. Bærentzen, H. Aanaes, Signed distance computation using the angle weighted pseudonormal, IEEE Trans. Vis. Comput. Graphics 11 (3) (2005) 243–253.
[30] X. Wang, Z.-Y. Yin, H. Xiong, D. Su, Y.-T. Feng, A spherical-harmonic-based approach to discrete element modeling of 3d irregular particles, Internat. J. Numer. Methods Engrg. 122 (20) (2021) 5626–5655.
[31] M. Wang, Y. Feng, D. Owen, T. Qu, A novel algorithm of immersed moving boundary scheme for fluid–particle interactions in DEM–LBM, Comput. Methods Appl. Mech. Engrg. 346 (2019) 109–125.
[32] F. Lehericey, V. Gouranton, B. Arnaldi, Gpu ray-traced collision detection for cloth simulation, in: Proceedings of the 21st ACM Symposium on Virtual Reality Software and Technology, 2015, pp. 47–50.
[33] M. Li, D.M. Kaufman, C. Jiang, Codimensional incremental potential contact, ACM Trans. Graph. 40 (4) (2021) 1–24, http://dx.doi.org/10.1145/3450626.3459767.
[34] A.G. Neto, Framework for automatic contact detection in a multibody system, Comput. Methods Appl. Mech. Engrg. 403 (2023) 115703.
[35] T. Karras, T. Aila, Fast parallel construction of high-quality bounding volume hierarchies, in: Proceedings of the 5th High-Performance Graphics Conference, 2013, pp. 89–99.
[36] D. Ströter, J.S. Mueller-Roemer, A. Stork, D.W. Fellner, OLBVH: octree linear bounding volume hierarchy for volumetric meshes, Vis. Comput. 36 (10–12) (2020) 2327–2340.
[37] P. Areias, N. Sukumar, J. Ambrósio, Continuous gap contact formulation based on the screened Poisson equation, Comput. Mech. (2023) 1–17, http://dx.doi.org/10.1007/s00466-023-02309-8.
[38] S. Luding, Introduction to discrete element methods: basic of contact force models and how to perform the micro-macro transition to continuum theory, Eur. J. Environ. Civ. Eng. 12 (7–8) (2008) 785–826.
[39] D. Fincham, Leapfrog rotational algorithms, Mol. Simul. 8 (3–5) (1992) 165–178.
[40] NVIDIA, NVIDIA turing GPU architecture: Graphics reinvented, 2018, URL https://bit.ly/2NGLr5t.
[41] I. Wald, W. Usher, N. Morrical, L. Lediaev, V. Pascucci, RTX beyond ray tracing: Exploring the use of hardware ray tracing cores for tet-mesh point location, in: High Performance Graphics (Short Papers), 2019, pp. 7–13.
[42] S. Zhao, Z. Lai, J. Zhao, Leveraging ray tracing cores for particle-based simulations on gpus, Internat. J. Numer. Methods Engrg. 124 (3) (2023) 696–713.
[43] Y. Feng, An effective energy-conserving contact modelling strategy for spherical harmonic particles represented by surface triangular meshes with automatic simplification, Comput. Methods Appl. Mech. Engrg. 379 (2021) 113750.
[44] H. Cheng, T. Shuku, K. Thoeni, P. Tempone, S. Luding, V. Magnanimo, An iterative bayesian filtering framework for fast and automated calibration of dem models, Comput. Methods Appl. Mech. Engrg. 350 (2019) 268–294.