StarDEM: efficient discrete element method for star-shaped particles

Camille Schreck, Sylvain Lefebvre, David Jourdan, Jonàs Martínez

Université de Lorraine, CNRS, Inria, LORIA

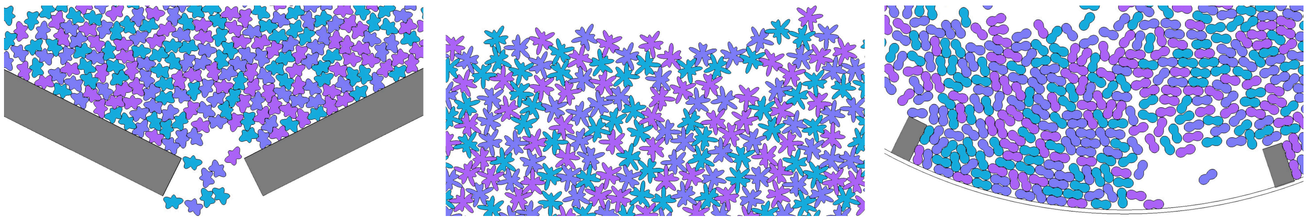

Figure 1: *Our StarDEM simulation: bridges forming when particles discharge through a narrow opening, packing and rotating drum.*

Abstract

Granular materials composed of particles with complex shapes are challenging to simulate due to the high number of collisions between the particles. In this context, star shapes are promising: they cover a wide range of geometries from convex to concave and have interesting geometric properties. We propose an efficient method to simulate a large number of identical star-shaped particles. Our method relies on an effective approximation of the contacts between particles that can handle complex shapes, including highly non-convex ones. We demonstrate our method by implementing it in a 2D simulation using the Discrete Element Method, both on the CPU and GPU.

CCS Concepts

• ***Computing methodologies*** $\rightarrow$ ***Physical simulation; Collision detection;***

1. Introduction

Granular materials, composed of discrete particles like sand or gravel, are of particular interest in material science as their material behavior still needs to be fully understood; notably, understanding how the particles' shape influences the material behavior remains an open research problem. Recently, granular materials whose particles have been fabricated with a specific shape have gained attention, as they have been shown to offer applications in soft robotics and architecture [DM16; HOBD21]. In that context, an efficient and accurate simulation method would avoid the need for time-consuming and expensive experiments with fabricated particles while allowing researchers to explore the properties of those materials. In order to capture the behavior caused by specific shapes, simulations must operate at the particle level, dealing with a large number of collisions. While numerous methods focus on spherical particles, fewer have studied the simulation of non-spherical ones. Detecting collisions and handling contacts for non-spherical geometries is typically more resource-intensive compared to spheres.

This paper aims to propose a simulation method dedicated to such granular materials composed of identical fabricated particles. For this, we need not only a fast simulation, as experimenting may require a large number of simulations, but we also need to be able to represent a broad range of complex particle geometries–in particular, the non-convex ones, as shown in Figure 1, often exhibit intriguing behavior. Many previous works focus on convex or simple shapes and lack the expressivity to explore more complex shapes. Star shapes have been shown to be a useful compact representation of an extensive range of non-convex shapes. They also offer distinctive geometric properties that can be used to speed up collision handling. A few works [LCH20; Wan*21; LZZH22] simulate star-shaped rigid particles. However, their evaluation of the overlap between two particles, while effective for close-to-spherical particles (usually representing natural grains or rocks), is not well-suited for strong concavities or elongated particles. Our *StarDEM* method focuses on the stable and efficient simulation of star-shaped particles and can handle a large variety of non-spherical particles, including highly non-convex geometries.

Our contributions are the following: (1) We propose to use a first-order approximation of the distance to efficiently estimate the distance between a point and the boundary of a star-shape. (2) We implement an efficient 2D simulation on CPU and GPU dedicated to

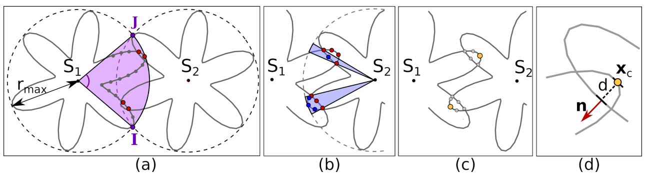

Figure 2: *Contact detection: (a) Nodes from S_1 between the intersection points of the bounding circles are tested to find the ones inside S_2 (in red). (b) Nodes from S_2 corresponding to each contact are identified (in blue). (c) For each distinct contact, the deepest node (yellow) is selected as the contact point $\boldsymbol{x}_c$. (d) The corresponding overlap depth d and the normal $\boldsymbol{n}$ are computed.*

this fabricated granular material, particularly materials composed of highly non-convex particles. We provide open-source code for our implementation here https://github.com/schreckc/StarDEM.

2. Related work

In Computer Graphics, granular materials are often simulated as a continuum (e.g., the Material Point Method [DB16]). The material's macroscale behavior is homogenized, so there is no need to represent each grain individually. However, to capture new behaviors induced by different shapes, it is first necessary to simulate each distinct particle and handle the collision between them at a small scale.

The Discrete Element Method (DEM) is an efficient way of simulating a large number of particles, each being represented as a distinct object with a position and orientation. In the original paper from Cundall et al. [CS79], rigid spheres are simulated using soft collisions: the particles are allowed to overlap slightly. Given two spheres S_1 and S_2 of centers $\boldsymbol{c}_1, \boldsymbol{c}_2$ and radii r_1, r_2, the normal force between them depends on the depth of the overlap $d = r_1 + r_2 - ||\boldsymbol{c}_1 - \boldsymbol{c}_2||$, and the contact normal $\vec{\boldsymbol{n}} = (\boldsymbol{c}_1 - \boldsymbol{c}_2)/||\boldsymbol{c}_1 - \boldsymbol{c}_2||$. The normal force can then be computed as $\vec{\boldsymbol{F}}_n = k_n\, d\, \vec{\boldsymbol{n}}$ where k_n is a stiffness coefficient. Friction forces can also be computed, as well as tangential forces, which come from the difference of tangential speed between the two particles at contact point $\boldsymbol{x}_c$ (the center of the overlap). We refer the reader to [CS79] for additional details. The force $\vec{\boldsymbol{F}}_{12}$ of S_2 over S_1 is then applied at $\boldsymbol{x}_c$, and the resulting force and momentum on the gravity center of S_1 are $\vec{\boldsymbol{F}} = \vec{\boldsymbol{F}}_{12}$ and $\vec{\boldsymbol{M}} = \vec{\boldsymbol{F}}_{12} \times (\boldsymbol{x}_c - \boldsymbol{c}_1)$. The forces are applied similarly on S_2 with $\vec{\boldsymbol{F}}_{21} = -\vec{\boldsymbol{F}}_{12}$. Finally, the particles are advected using an explicit integration scheme.

Over the years, many papers have expanded on the original DEM, and in particular, some extend the method to non-spherical smooth particles; we refer the reader to [Fen23] for an in-depth recent review. The most prominent method represents the particle shape as a clump of spheres [BYM05]. This method takes advantage of the efficiency of treating collisions between spheres. However, accurately representing smooth, complex shapes requires a high number of spheres per particle, making the method costly in these cases. Other methods using implicit functions, such as super ellipsoid and Signed Distance Fields (SDF), have also been investigated. Among them are a few works focusing on star-shaped particles whose radial function is represented by spherical harmonics (or Fourier series in 2D). Star shapes represent an extensive range of nontrivial non-convex geometry (see, for example, Figure 1) that can lead to interesting physical behavior.

A star-shape is defined as a shape containing any segment between the shape's center and any boundary point. A 2D star-shape with a smooth boundary can then be defined using a center $\boldsymbol{c}$ and a differentiable 2π-periodic radial function $r(\alpha)$ associating a radius to an angle α around the center. Note that for particles that are not symmetrical around c, the gravity center does not necessarily coincide with the star-shape's geometric center $\boldsymbol{c}$. The boundary of the shape can be implicitly defined as

$$\{\boldsymbol{p}|f(\boldsymbol{p}) = 0\} \text{ with } f(\boldsymbol{p}) = ||\boldsymbol{p} - \boldsymbol{c}|| - r(\alpha(\boldsymbol{p})) \tag{1}$$

where $\alpha(\boldsymbol{p})$ is the angle formed by $(\boldsymbol{p} - \boldsymbol{c})$ and the direction corresponding to $\alpha = 0$. Thus, a point $\boldsymbol{p}$ is inside the shape if $f(\boldsymbol{p}) < 0$ and outside if $f(\boldsymbol{p}) > 0$. The boundary can be sampled by uniformly sampling the angle α:

$$\{r_i\} = \{r(\alpha_i) \mid \alpha_i = i\, 2\pi/n \text{ for } i = 0, \ldots, n-1\}. \tag{2}$$

These properties enable efficient algorithms to detect contacts between two star shapes [CH21; GB13] by finding the sampled nodes from one particle inside the other.

Only a few articles tackle the DEM simulation of star-shaped particles. Closer to us, Wang et al. [Wan*21] consider a contact detection scheme for 3D star shapes. They propose to compute the overlap depth using a radial distance, which is a decent approximation of the Euclidean distance between one point and the boundary of a star-shape only if the shape is close enough to spherical (see Figure 3). The work from Lai et al. [LZZH22] models more general shapes with SDF, but in the case of spherical harmonics, they suffer from the same issue as they approximate the SDF by the radial distance. Finally, Lai et al. [LCH20] propose a closed-form optimization to find the overlap and intersection points between the boundaries of two star shapes in 2D. However, their work does not treat cases where there are several intersections between two particles, which is often the case with non-convex shapes.

3. Method

First-order approximation distance. To compute the overlap, we have to evaluate the distance d between a point and the boundary of a star-shape S defined by a center $\boldsymbol{c}$ and the radial function r. Previous work uses the radial distance along the ray through $\boldsymbol{p}$ and the center: $d_r = f(\boldsymbol{p})$. This is a reasonable approximation as long as the normal of the boundary is close to the direction of the ray from the center. Unfortunately, this is not the case with an elongated or highly non-convex particle like the one shown in Figure 3.

We propose to use instead the first-order approximation of the distance used by Taubin [Tau94] to rasterize an implicit curve. This approximation uses the first-order Taylor expansion of the implicit function. In our case, the boundary is defined by the implicit function f (Equation 1). The distance between a point $\boldsymbol{p}$ and the boundary can be approximated by: $d_{fo} = \frac{f(\boldsymbol{p})}{||\nabla f||}$ with ∇f being the analytical gradient of f. This distance is asymptotically equivalent to the Euclidean distance: the error with the Euclidean distance approaches zero when $\boldsymbol{p}$ approaches the boundary. So the error is small when close to the boundary (Figure 3), which is where the soft collisions are computed given that they are supposed only to allow small overlap.

This first-order approximation distance between a point and the boundary of S can thus be computed using only one evaluation of f and its gradient. The normal of the surface close to the point $\boldsymbol{p}$ can then be approximated by the normalized gradient $\frac{\nabla f}{||\nabla f||}$. The first-order and radial distances are both evaluated in constant time. The former is only slightly more costly to evaluate than the latter. Indeed, it is the radial distance divided by the gradient norm of f, which is also used to compute the normal. Even though higher-order approximations of d are possible [Tau94], we found our simulation stable by using the first-order only.

Collision detection. We use a collision detection scheme similar to Wang et al. [Wan*21]. A bounding circle is centered on the geometric center of the particle. Its radius is the maximal radius of the particle. If an intersection exists between the bounding circles of two particles S_1 and S_2, the region of possible contact is restricted to the intersection area of the two disks. We denote the intersection points between the two circles $\boldsymbol{I}$ and $\boldsymbol{J}$ (see Figure 2).

As Wang et al. [Wan*21], we then use a nodes-to-surface scheme to compute the possible contacts between S_1 and S_2. We find the sampled nodes (Equation 2) of S_1 that are inside the boundary of S_2 using the sign of f. A ray starting from the center of a star-shape only crosses its boundary once. So we only need to check the boundary nodes $\boldsymbol{x}$ of S_1 corresponding to an angle $\alpha(\boldsymbol{x})$ that belongs to the interval $[\alpha(\boldsymbol{I}), \alpha(\boldsymbol{J})]$ delimited by the intersection points $\boldsymbol{I}$ and $\boldsymbol{J}$ (purple region in Figure 2 (a)). Each set of consecutive nodes inside S_2 corresponds to a distinct contact between S_1 and S_2.

As far as we know, previous works ([LZZH22; Wan*21]) using nodes-to-surface collision methods only use the nodes from S_1 (red nodes in Figure 2) to resolve the collision. However, this leads to an asymmetric collision where the forces change if S_1 and S_2 are reversed. Therefore, we propose also to consider the boundary nodes from S_2. This can be done efficiently using the same geometric properties as above: the first and last S_1's nodes of a distinct contact (the red nodes in Figure 2(a) represent two contacts) define an angular region in S_2 (blue region in (b)). We select the nodes from S_2 inside this region (blue nodes). While this is fast on the CPU, doing this gets costly for the GPU. So, we only consider the red nodes from S_1 for the GPU implementation.

Collision resolution. For each contact between S_1 and S_2, we choose as contact point $\boldsymbol{x}_c$ the deepest node (yellow nodes in Figure 2 (c)) from one of the particles inside the other. The depth d of the overlap for this contact is then the distance between $\boldsymbol{x}_c$ and the boundary of the other particle. d is computed using the first-order approximation distance described above. The normal $\vec{\boldsymbol{n}}$ of the contact is estimated as the gradient of f at point $\boldsymbol{x}_c$. The values $\boldsymbol{x}_c$, d, and $\vec{\boldsymbol{n}}$ computed for each contact are then used as described in Section 2.

Note that while most collision methods use one point of contact for one distinct contact, previous works using nodes-to-surface methods tend to consider all nodes of S_1 inside S_2 as contact points and add forces for each of them. We decided to use the deepest point as it makes the force depend only on the depth of the overlap and has no dependency on the number of sampled nodes inside. However, the relative physical accuracy of the two methods needs further investigation. In addition, the force and momentum only need to be computed once per contact with the deepest point method.

Efficient Implementation. We specifically optimize the implementation of our method for the simulation of granular material composed of identical particles.

Considering that all particles have the same size, we employ a spatial grid with cells of size $2r_{max}$. Each particle is assigned a cell based on its position. For any given particle, we only need to examine the potential collision with particles within a 3x3 cell square surrounding the particle position.

The first-order distance, and thus radial function r and its derivative r', are frequently evaluated and can be costly to compute, for example, when using many Fourier coefficients. To circumvent this issue, we precompute r and r' in lookup tables. This replacement of function evaluation with table access simplifies the process and ensures constant time access. It is important to note that the number of samples for the lookup table differs from those used in the nodes-to-surface method. Given that we are sampling a 1D curve and that we only need to compute and store the tables once, we can afford high-resolution sampling. We use 20000 samples in our examples, ensuring precise and accurate results.

The GPU implementation is similar. We build the grid using atomic operations, creating linked list of particles in each grid entry. A 2D lookup table – a "sprite" of the particle – is employed to check whether points of S_1 are in S_2 and simultaneously retrieve the collision distance and normal.

4. Results

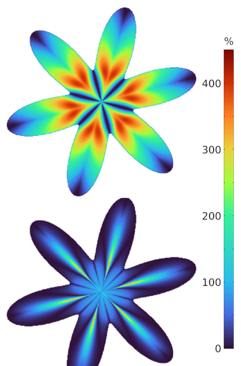

Figure 3: *Distance error: (top) radial, (bottom) first order.*

Our CPU results and timings have been computed on an AMD Ryzen 9 3900X 12-Core processor. We parallelized the simulation using OpenMP. Our examples use particles of radius around 5mm and weight 0.2g. The normal stiffness is 1e3 $\mathrm{N\,m^{-1}}$ and tangent stiffness 0.5e3 $\mathrm{N\,m^{-1}}$. The particle's boundary is sampled with 100 nodes. Our method can use any differentiable r function. In our example, we use Fourier sum to represent r: $r(\theta) = a_0 + \sum_{i=0}^{N} a_i \cos(i\theta) + b_i \sin(i\theta)$. This can be generalized in 3D using spherical harmonics. Please refer to the accompanying video to watch the simulation results described in the examples cited in this section.

First-order approximation vs radial distance. We compare the first-order approximation d_{fo} of the Euclidean distance to a numerically computed ground truth d_{gt}. The error $e = |d_{fo} - d_{gt}|/d_{gt}$ stays low close to the boundary (Figure 3, bottom). For example, in the case of the shape in Figure 3 of radius $r_{max} = 1$, the error at a distance < 0.01 of the boundary stays < 0.17.

The radial distance $d_r = f(\boldsymbol{p}) = ||\boldsymbol{p} - \boldsymbol{c}|| - r(\alpha(\boldsymbol{p}))$ is only close to the Euclidean distance where the normal to the boundary is close to the direction of the ray from the center. Compared to the ground truth distance, the error can grow very big for elongated regions even close to the border (Figure 3, top). The error at a distance < 0.01 of the boundary can reach 2.7. As shown in the video, this may cause instabilities in the simulation.

Comparison with a clump of spheres. We implemented Cundall and Strack's method for spheres [CS79] and Bell's method for a clump of spheres [BYM05]. The implementations are the same except for the treatment of the collisions. We compare the methods using packing: 3154 particles fall into a box (Figure 4). Each particle has an area of 30mm^2. We measure the average computation time on one thread to treat the collisions (detection and computation of the overlaps). In the case of spheres (a), using StarDEM with $r(\alpha) = 1$ runs around nine times slower than using direct spheres (9.6ms vs. 1.1ms for one collision step). However, when using a clump of spheres to represent a more complex shape, the cost of using spheres quadratically rises with the number of spheres needed per particle. For a cross shape, using a star-shape with $r(\alpha) = 1 + cos(4\alpha)$ (b), the computation time is 26ms while using a clump of spheres (c) rises to 12ms for 9 spheres and 41ms for 17 spheres. More complicated shapes (e.g. (d), (e)) would require many spheres and an increasing computation cost for a clump of spheres while the timings for our simulation stay similar ((d) 30ms, (e) 29ms).

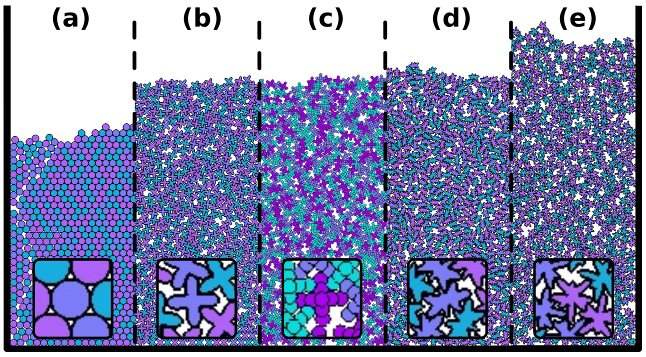

Figure 4: *Same amount of particles of equal area and different shapes, ranging from (a) to (e), after being poured into a box.*

Comparison with [LCH20]. We compared 2D timings with the method from Lai et al. [LCH20]. They report a computation time of around 5s for 1e4 candidate collisions between pairs of particles, resulting in an actual collision about half the time. In comparison, for the last 100 frames of our packing example in Figure 4 (e), we detect 84.9e4 candidate collisions (bounding circles intersecting), resulting in 51e4 pairs actually colliding and 84.7e4 distinct contacts. This is computed in 3s on one thread and 0.6s on 8 threads.

Other results. We demonstrated our method on various examples with different shapes. We use a time step of $\Delta t = 0.1$ms. The reported times for one simulation step are averaged over the whole simulation and computed on 8 threads. The particles used for those times are the ones shown in the pictures–however, the timing is usually quite similar for different shapes. We show a loose packing example (Figure 1 (left)) with particles falling in a box (3154 particles, one simulation step costs in average 8ms). Figure 1 (middle) shows particles inside a rotating drum (900 particles, 2ms). Particles discharging through a narrow gap tend to form bridges as shown in Figure 1 (right) (279 particles, 1ms) –the video also shows the force network created by this example. The last example in the video shows a column collapsing and forming a heap (429 particles, 2ms).

GPU. On an NVidia GTX 4080, a full simulation step with 1 million concave particles in contact takes only 7.5ms (see video).

5. Discussion and Conclusion

While we optimized for identical particles in this paper, the core of our method can also be used for heterogeneous particles. In the case of a high number of different shapes, the memory cost of using lookup tables can become prohibitive (even more so for 2D lookup tables we used for GPU or that would be used in a similar method using rasterized SDF). This issue will become more acute when transitioning to 3D star-shaped particles. In these cases, using the radial function directly instead of LUT would considerably reduce the memory needed at the cost of being more computationally intensive. Indeed, the Fourier representation of star shapes is a convenient way to compactly represent a large range of shapes.

This work's final goal is to simulate manufactured granular materials accurately. This will involve meticulously calibrating physical values to mirror real-world materials and assessing the precision of our chosen parameters. While we are currently using the seminal DEM version–which is simple to implement–we would like to demonstrate our method using more advanced models that may better fit experiments. Notably, using explicit integration imposes a small time step, which is particularly important here, given that the force gets inaccurate if the overlap gets too big.

In contrast to earlier approaches focusing on naturally occurring grains that remain rather close to spherical, our paper uniquely targets the simulation of a large range of complex star-shaped particles, notably including highly non-convex particles.

References

[BYM05] Bell, N., Yu, Y., and Mucha, P. J. "Particle-based simulation of granular materials". *Symposium on Computer Animation*. 2005 2, 4.

[CH21] Capozza, R. and Hanley, K. "A hierarchical, spherical harmonic-based approach to simulate abradable, irregularly shaped particles in DEM". *Powder Technology* 378 (2021), 528–537 2.

[CS79] Cundall, P. A. and Strack, O. D. L. "A discrete numerical model for granular assemblies". *Géotechnique* (1979) 2, 4.

[DB16] Daviet, G. and Bertails-Descoubes, F. "A Semi-Implicit Material Point Method for the Continuum Simulation of Granular Materials". *ACM Trans. Graph.* 35.4 (2016) 2.

[DM16] Dierichs, K. and Menges, A. "Towards an aggregate architecture: designed granular systems as programmable matter in architecture". *Granular Matter* 18 (2016), 1–14 1.

[Fen23] Feng, Y. "Thirty years of developments in contact modelling of non-spherical particles in DEM: a selective review". *Acta Mechanica Sinica* (2023) 2.

[GB13] Garboczi, E. and Bullard, J. "Contact function, uniform-thickness shell volume, and convexity measure for 3D star-shaped random particles". *Powder Technology* 237 (2013), 191–201 2.

[HOBD21] Howard, D., O'Connor, J., Brett, J., and Delaney, G. W. "Shape, size, and fabrication effects in 3D printed granular jamming grippers". *2021 IEEE 4th International Conference on Soft Robotics (RoboSoft)*. IEEE. 2021 1.

[LCH20] Lai, Z., Chen, Q., and Huang, L. "Fourier series-based discrete element method for computational mechanics of irregular-shaped particles". *Comput. Methods Appl. Mech. Eng.* 362 (2020) 1, 2, 4.

[LZZH22] Lai, Z., Zhao, S., Zhao, J., and Huang, L. "Signed Distance Field Framework for Unified DEM Modeling of Granular Media with Arbitrary Particle Shapes". *Comput. Mech.* 70.4 (2022) 1–3.

[Tau94] Taubin, G. "Distance Approximations for Rasterizing Implicit Cuves". *ACM Trans. Graph.* 13.1 (1994), 3–42 2, 3.

[Wan*21] Wang, X. et al. "A spherical-harmonic-based approach to discrete element modeling of 3D irregular particles". *Int. J. Numer. Methods Eng.* 122.20 (2021), 5626–5655 1–3.