



THE UNIVERSITY *of* EDINBURGH

Edinburgh Research Explorer

Contact detection between convex polyhedra and superquadrics in discrete element codes

Citation for published version:

Peng, D & Hanley, KJ 2019, 'Contact detection between convex polyhedra and superquadrics in discrete element codes', *Powder Technology*. <https://doi.org/10.1016/j.powtec.2019.07.082>

Digital Object Identifier (DOI):

[10.1016/j.powtec.2019.07.082](https://doi.org/10.1016/j.powtec.2019.07.082)

Link:

[Link to publication record in Edinburgh Research Explorer](#)

Document Version:

Peer reviewed version

Published In:

Powder Technology

General rights

Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy

The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.



Contact detection between convex polyhedra and superquadrics in discrete element codes

Di Peng, Kevin J. Hanley*

*School of Engineering, Institute for Infrastructure and Environment, The University of Edinburgh,
Edinburgh EH9 3FG, United Kingdom*

Abstract

Particle shape substantially influences the bulk behaviour of granular systems. Hence, much scientific interest has been devoted to the adoption of non-spherical fundamental particles in discrete element method simulations. Two examples of such particles are polyhedra, which are highly angular, and superquadrics, which are best suited to simulate rounded particles. It is desirable to use both types of particle together in a simulation to capture the broadest possible range of particle shapes. In this paper, a novel contact detection algorithm is presented for a convex polyhedron and superquadric. This algorithm was implemented in a C++ code which was used to verify the correctness of the algorithm and evaluate its efficiency using the Monte Carlo method. The proposed contact detection algorithm is particularly efficient for many-faceted polyhedra as the effect of increasing the number of faces on the evaluation time is small.

Keywords: discrete element method (DEM), nonspherical particle, polyhedron, superquadric, numerical simulation

1. Introduction

Since its formulation by Cundall and Strack [1] in the 1970s, the discrete element method (DEM) has become extremely popular as it allows exploration of the micro-scale behaviour that determines the bulk-scale response of any particulate system. The most commonly used

*Corresponding author.

Email addresses: `di.peng@ed.ac.uk` (Di Peng), `k.hanley@ed.ac.uk` (Kevin J. Hanley)

1 particle shape in DEM simulations is the sphere, which is both conceptually and computa-
2 tionally simple. The large majority of commercial codes such as PFC [2] and EDEM [3],
3 along with open-source codes such as YADE [4] and LIGGGHTS [5], adopt spheres as the
4 basic shape of element. However, in reality many particles are poorly described by spheres
5 and some problems cannot be adequately addressed without considering particle shape more
6 accurately.

7 In recognition of the importance of particle shape, there has been a major increase
8 in scientific interest in recent years in DEM modelling of non-spherical particle systems [6].
9 Many approaches have been proposed. The simplest is implementation of a rolling resistance
10 model at the inter-particle contacts. As particle non-sphericity restricts its rotation to some
11 extent, rolling resistance is capable of capturing a subset of the physics of non-spherical
12 particles but some behaviour is poorly captured. The most common approach for simulating
13 non-spherical particles, adopted in the commercial codes PFC [2] and EDEM [3], is the multi-
14 sphere method in which spheres are clumped together to create irregularly shaped clusters,
15 e.g., [7]. However, many spheres may be required to obtain a reasonable approximation of a
16 real particle's shape [8, 9]. An alternative is to replace the fundamental spheres with other
17 shapes such as polyhedra, ellipsoids [10, 11], or superquadrics [12] which are a generalisation
18 of ellipsoids.

19 Compared to spheres, accurately resolving the contacts between complex fundamental
20 particles such as superquadrics or polyhedra is difficult and computationally expensive. Two
21 approaches are commonly used: discrete function representation (DFR) and continuous func-
22 tion representation (CFR) [13]. DFR [14, 15] is based on discretising the surface of a particle
23 into a set of nodes. CFR detects and evaluates interparticle contacts through direct manipu-
24 lation of the equations which mathematically describe a particle's shape. CFR has two sub-
25 types: analytical CFR is used to resolve contacts between polyhedra [16, 17] while numerical
26 CFR is used for resolving contacts between particles with curved surfaces [10, 12, 18, 19, 20].
27 These CFR approaches tend to be applicable to particles only of specific shapes, e.g., cylin-
28 ders [21] or convex particles represented using the potential particle approach [22, 23]. There
29 are established analytical CFR methods for detecting and evaluating contacts between poly-

1 hedra, e.g., the 3DEC [24] code adopts Cundall’s contact plane (CP) method [25]. PFC v6.0
2 allows the simulation of systems of mixed spheres and polyhedra using a variant of the con-
3 tact detection algorithm proposed by Nezami et al. [26]. Overlaps between ellipsoids may be
4 detected efficiently using CFR-based algorithms, e.g., [27, 28]. A CFR approach for contact
5 resolution of convex superquadrics [29] has been implemented in LIGGGHTS [5].

6 There are also methods which, in principle, can be applied to any shape of particle. Feng
7 et al. [30] proposed a method of determining contact orientation based on the potential
8 field which is related to particle shape. Dong et al. [31] proposed a contact resolution
9 method based on orientation discretisation: the contact region is divided into cells and each
10 cell is categorised as inside or outside of the particle intersection. Their specific approach
11 requires a pre-computed database. The Gilbert-Johnson-Keerthi (GJK) algorithm [32]
12 is used in computer graphics for detecting penetration or finding the minimum distance
13 between two objects. The original GJK algorithm has been optimised both for speed [33, 34,
14 35] and for specific shapes such as polyhedra [36]. However, the GJK algorithm is accurate
15 only for shapes with flat surfaces; for shapes with curved surfaces such as superquadrics,
16 discretisation is necessary.

17 While there are established approaches for polyhedron–polyhedron (PH–PH) and superquadric–
18 superquadric (SQ–SQ) contact detection, an open problem is contact detection in mixed
19 polyhedron–superquadric (PH–SQ) systems. Such simulations are desirable as many real
20 systems contain a broad range of particle shapes. Superquadrics are appropriate for sim-
21 ulating rounded particles but cannot capture the high degree of angularity provided by
22 polyhedra. Hence, having both shapes available in a single simulation gives a broader range
23 of possible particle shapes than solely polyhedra or superquadrics.

24 In this paper, a novel contact detection algorithm for a convex polyhedron and su-
25 perquadric is presented which is efficient and reasonably simple. An associated contact
26 evaluation method has also been proposed to determine the contact normal and interparti-
27 cle overlap. The contact detection algorithm has been implemented in a C++ code, enabling
28 verification of the algorithm and quantification of its run time.

1 2. Development of convex PH–SQ contact detection algorithm

2 In Sections 2.1 and 2.2, the superquadric (SQ) and polyhedron (PH) particles are re-
3 spectively introduced and an existing algorithm for SQ–SQ contact detection is presented.
4 The newly developed algorithm for PH–SQ contact detection is then described in Section
5 2.3 based on this prior knowledge.

6 2.1. Definition of and contact detection between convex superquadrics

7 The surface of a SQ particle may be written as a function

$$F(\mathbf{X}) = F(x, y, z) = \left(\left| \frac{x}{a} \right|^{n_2} + \left| \frac{y}{b} \right|^{n_2} \right)^{\frac{n_1}{n_2}} + \left| \frac{z}{c} \right|^{n_1} - 1 = 0 \quad (1)$$

8 where \mathbf{X} is a point defined by three coordinates (x, y, z) in 3D space [29]. a, b, c are shape
9 parameters (half-lengths of the particle along the x, y, z axes) and n_1, n_2 are blockiness
10 parameters which define a SQ’s shape. $n_1 = n_2 = 2$ corresponds to an ellipsoid; in addition,
11 if $a = b = c$, the ellipsoid becomes a sphere. The LIGGGHTS implementation of SQs
12 requires $n_1, n_2 \geq 2$ [37]. This prohibition of non-convex SQs is to avoid issues with multiple
13 contacts between two SQs.

14 If $F(\mathbf{X}) < 0$, \mathbf{X} lies within the particle; if $F(\mathbf{X}) > 0$, \mathbf{X} is outside of the particle. The
15 special case $F(\mathbf{X}) = 0$ means that \mathbf{X} lies on the particle’s surface. Consider two SQs A and
16 B that are potentially in contact. The surfaces of both particles can be written as $F_A(\mathbf{X})$
17 and $F_B(\mathbf{X})$, or simply F_A and F_B . Contact detection between A and B is equivalent to the
18 following constrained optimisation [23]:

$$\text{Minimise } F_A + F_B \text{ while } F_A - F_B = 0 \quad (2)$$

19 Podlozhnyuk et al. [29] substituted Eq. (1) into Eq. (2), and used the Lagrange multiplier
20 method to obtain a solution. Suppose the Lagrange function

$$L(x, y, z, \lambda) = F_A + F_B + \lambda(F_A - F_B). \quad (3)$$

1 Thus,

$$\nabla L = \mathbf{0} \iff \nabla(F_A + F_B) = -\lambda \nabla(F_A - F_B) \iff (1 + \lambda) \nabla F_A + (1 - \lambda) \nabla F_B = \mathbf{0} \quad (4)$$

2 By combining Eq. (4), which provides three equations, with $\frac{\partial L}{\partial \lambda} = 0$, or equivalently $F_A -$
 3 $F_B = 0$, a set of four equations is obtained:

$$\Phi = \begin{bmatrix} (1 + \lambda) \frac{\partial F_A}{\partial x} + (1 - \lambda) \frac{\partial F_B}{\partial x} \\ (1 + \lambda) \frac{\partial F_A}{\partial y} + (1 - \lambda) \frac{\partial F_B}{\partial y} \\ (1 + \lambda) \frac{\partial F_A}{\partial z} + (1 - \lambda) \frac{\partial F_B}{\partial z} \\ F_A - F_B \end{bmatrix} = \mathbf{0} \quad (5)$$

4 There must be a solution $(x_0, y_0, z_0, \lambda_0)$ to Eq. (5). The multi-unknown Newton's method
 5 can be used to obtain this solution:

$$\begin{bmatrix} x_{n+1} \\ y_{n+1} \\ z_{n+1} \\ \lambda_{n+1} \end{bmatrix} = \begin{bmatrix} x_n \\ y_n \\ z_n \\ \lambda_n \end{bmatrix} - \mathbf{J}^{-1} \Phi \quad (6)$$

6 where \mathbf{J} is the Jacobian of Φ . The detailed solution procedure can be found in Podlozhnyuk
 7 et al. [29], resulting in the point (x_0, y_0, z_0) , which is the midpoint of two points \mathbf{X}_A and
 8 \mathbf{X}_B : the closest point to the centre of SQ B on SQ A 's surface and the closest point to the
 9 centre of SQ A on SQ B 's surface, respectively. We can evaluate whether or not (x_0, y_0, z_0)
 10 is the contact point with the following criterion:

$$F_A(x_0, y_0, z_0) < 0 \quad \text{and} \quad F_B(x_0, y_0, z_0) < 0 \quad (7)$$

11 If Eq. (7) is true, A and B are in contact; otherwise the two SQs are not in contact. If
 12 F_B were substituted by the function of a planar wall, SQ-wall contact detection can be
 13 accomplished in a similar manner to find the point on the surface of the SQ which is closest

1 to the wall, or the deepest point beneath the wall surface if the SQ contacts the wall [29].

2 2.2. Definition of and points within convex polyhedra

3 The surface function describing a polyhedron is given as Eq. (8) [22]:

$$F(\mathbf{X}) = F(x, y, z) = \sum_{i=1}^n \langle f_i(x, y, z) \rangle = \sum_{i=1}^n \langle A_i x + B_i y + C_i z - D_i \rangle = 0 \quad (8)$$

where n is the total number of faces comprising the PH, $f_i(x, y, z)$ is the function of the i^{th} plane of the PH with outer normal vector (A_i, B_i, C_i) , and D_i is the perpendicular distance from the origin to the i^{th} plane of the PH. The Macaulay brackets in Eq. (8) indicate that positive terms retain their values while negative terms are set to zero. Eq. (8) is equivalent to

$$F(\mathbf{X}) \leq 0 \iff \forall f_i(\mathbf{X}) \leq 0 \quad (9a)$$

$$F(\mathbf{X}) > 0 \iff \exists f_i(\mathbf{X}) > 0. \quad (9b)$$

4 Eq. (9a) means that if a point is within a PH, all PH surface functions must be negative
5 at that point. Conversely, if a point makes one or more PH surface functions positive, that
6 point must be outside the PH (Eq. (9b)). Thus Eq. (9a) can be used to confirm a possible
7 contact point, while Eq. (9b) can be used to eliminate invalid candidate contact points. Eq.
8 (9) is used extensively in the algorithm presented in Section 2.3.

9 2.3. Resolving contact or non-contact between a PH and a SQ

10 In total, there are six distinct cases that must be considered in this algorithm: three
11 cases in which the PH and SQ are in contact, and three in which they are not. We have
12 chosen a triangular prism with five faces as the PH and an ellipsoid as the SQ for most of
13 the figures in Section 2.3. After the six cases have been described, a flowchart is shown as
14 Fig. 7 which summarises the steps in the algorithm.

2.3.1. Defining the ‘Initial Face’ (IF) and positive/negative faces

We define the initial face (IF) as the face of the PH whose outer normal vector has the maximum dot product, i.e., is most closely aligned, with the vector joining the centroids of the PH and SQ. An example IF is shown in Fig. 1(a). The IF is used in the first step to judge whether or not the particles are in contact. Positive/negative faces are assigned that status by their positions relative to a fixed point \mathbf{X} . Each face lies on an infinite plane; if the point \mathbf{X} is above that plane, where the outer normal vector indicates the positive direction, that face is positive for point \mathbf{X} . Conversely, if \mathbf{X} is beneath the plane, the face is negative for \mathbf{X} . In order to clearly present a 3D shape on paper, only three of the faces of the triangular prism PH are labelled as positive or negative in the figures in this section; the other faces are always negative.

In this algorithm, the SQ–planar wall contact detection approach summarised in Section 2.1 [29] is applied to one or more specific planes enclosing faces of the PH. This yields one or more points on the surface of the SQ which are potentially contact points.

2.3.2. Case 1: Positive IF

Consider Fig. 1(a) in which the PH and SQ are not in contact. The point highlighted on the surface of the SQ has been identified as a potential contact point based on Section 2.1. However, this point makes the IF positive. Based on Eq. (9b), there must be no contact and the contact detection can be terminated. Case 1 is highly efficient as there is no need to check any PH faces apart from the IF.

2.3.3. Case 2: All faces negative

Case 2, in which all faces are negative, is shown in Fig. 1(b). The highlighted point must be the contact point according to Eq. (9a). Case 2 is somewhat less efficient than Case 1 as the positive/negative assessment must be made for all faces of the PH, not only the IF.

2.3.4. Cases 3 & 4: Negative IF, resolve through switch to ‘Positive Face’ (PF)

Cases 3 (non-contact) and 4 (contact) arise when the IF is negative for the potential contact point denoted as \mathbf{X}_0 but one other face is positive. The scenario in which more

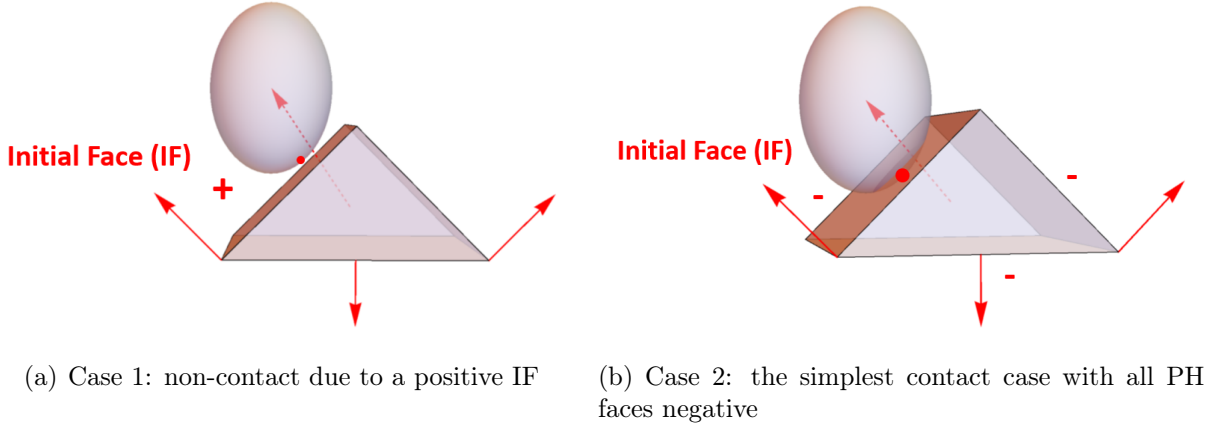


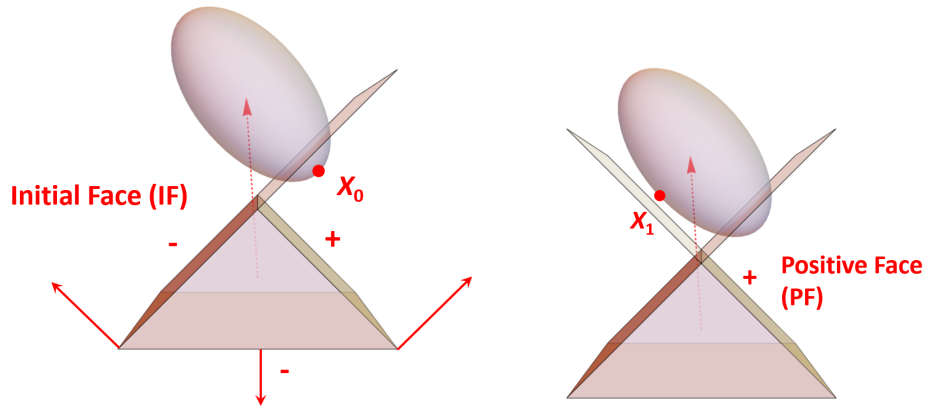
Figure 1: The simplest (non-)contact cases for face contact of a PH and SQ.

1 than one face is positive is addressed in Section 2.3.5. Fig. 2(a) illustrates Case 3: \mathbf{X}_0
 2 makes one face, which is not the IF, positive. In this situation, we switch from the IF to
 3 this newly identified positive face (PF) and repeat the same procedure as before. A new
 4 potential contact point \mathbf{X}_1 is identified by applying the procedure outlined in Section 2.2
 5 to the plane which includes the PF (Fig. 2(b)). Since the PF remains positive, the same
 6 conclusion of no contact is drawn, based on Eq. (9b), as for Case 1.

7 Case 4 is the analogue of Case 3 for which there is a PH–SQ contact. This is shown in
 8 Fig. 3. As usual, the IF is found initially, resulting in a potential contact point \mathbf{X}_0 (Fig.
 9 3(a)). Since \mathbf{X}_0 makes a face other than the IF positive, it cannot be the contact point but
 10 another point on the SQ surface may still be in contact with the PH. Attention is switched
 11 from the IF to the PF (Fig. 3(b)). The potential contact point \mathbf{X}_1 makes all PH faces
 12 negative. As concluded for Case 2, \mathbf{X}_1 is the contact point.

13 2.3.5. Cases 5 & 6: Negative IF, search and return for edge/vertex contact

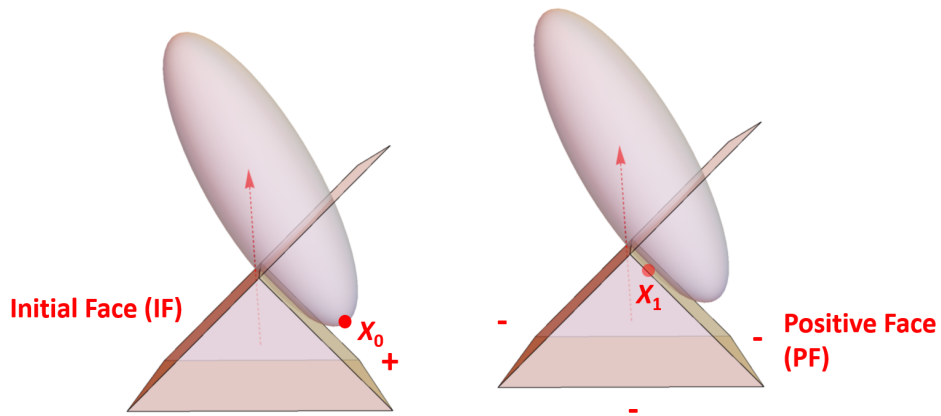
14 In Cases 3 and 4, an unambiguous assessment of contact/non-contact could be made
 15 after switching from the initial face to the one positive face. However, one could imagine a
 16 scenario in which the contact status remains uncertain after switching to the one PF. For
 17 example, resolving a contact between a SQ and the edge or vertex of a PH, rather than
 18 its face, cannot be adequately resolved using Cases 1–4. Furthermore, Cases 3 and 4 are



(a) Positive and negative faces and potential contact point X_0 based on the IF

(b) After switching attention to the PF

Figure 2: Non-contact of a PH and SQ determined by switching from the IF to the PF (Case 3).



(a) Positive and negative faces and potential contact point X_0 based on the IF

(b) Contact point X_1 found after switching to the PF

Figure 3: Contact point X_1 found after switching from the IF to the PF (Case 4).

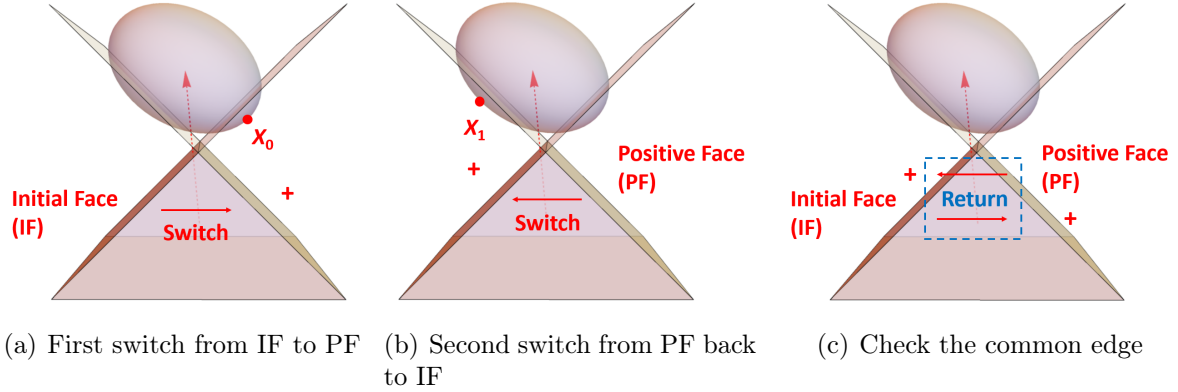


Figure 4: SR method for determining SQ contact with a PH edge/vertex (Case 5: non-contact).

1 inapplicable if there is more than one positive face. In these situations, Cases 5 (non-contact)
 2 and 6 (contact) apply.

3 Consider Fig. 4(a). As for Cases 3 and 4, a potential contact point \mathbf{X}_0 is found initially
 4 using the IF obtained by joining the centroids of the two shapes. Since this makes the
 5 rightmost face positive, another potential contact point \mathbf{X}_1 is found based on the PF (Fig.
 6 4(b)). However, \mathbf{X}_1 makes the PF negative but the IF positive, thus creating an endless
 7 alternating cycle between these two faces. This is resolved by checking whether the edge
 8 common to the two faces contacts the SQ (Fig. 4(c)). Fig. 5 is a similar scenario in which
 9 there is indeed a contact with the common edge. This is termed the ‘search and return’
 10 (SR) method for contact determination. This also allows the detection of contacts between
 11 a SQ and a vertex of a PH: since a vertex belongs to multiple edges of a PH, contact can
 12 be assessed using any of the edges linking to it.

13 In Figs. 4 and 5, the return appears after only one switch from IF to another face.
 14 However, more iterations may be needed before a return appears, e.g., Fig. 6(a). This is
 15 a limitation of the simple SR method. Furthermore, consider Fig. 6(b) which shows a 2D
 16 projection of a cuboidal SQ contacting a many-faceted PH. Starting from the IF in the usual
 17 manner, both the IF and all of the faces surrounding it are negative. However, since one
 18 face of the PH remains positive, \mathbf{X}_0 cannot be definitively identified as the contact point.
 19 The SR method described above needs to be amended to account for this general situation.

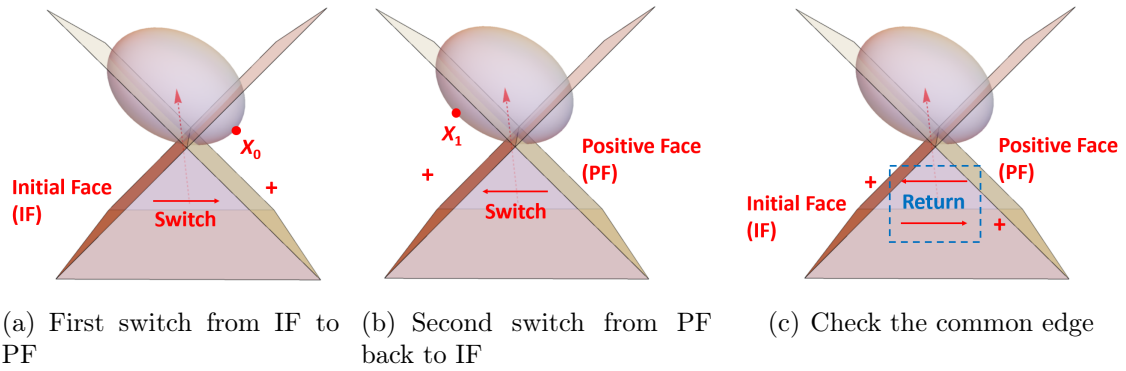


Figure 5: SR method for determining SQ contact with a PH edge/vertex (Case 6: contact found).

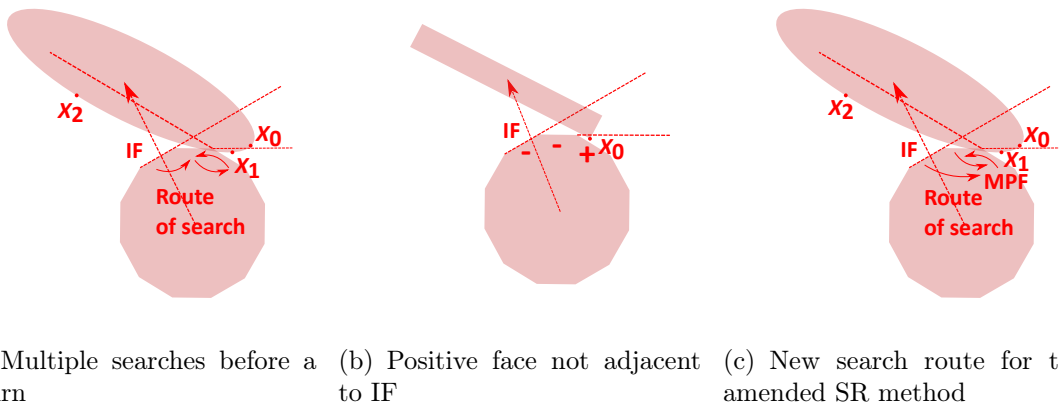


Figure 6: Extreme PH-SQ contact situations which cannot be addressed using the simple SR method.

1 This is simply done. As before, start from the IF and find \mathbf{X}_0 . The check of plane
2 function values with \mathbf{X}_0 is expanded to all PH faces, rather than those faces adjacent to
3 the IF. The face with maximum positive value (termed the ‘maximal positive face’ or MPF)
4 becomes the starting point for the next search. This amended initial search procedure is
5 suitable for Fig. 6(a), as demonstrated in Fig. 6(c), and is also applicable to all other
6 scenarios, including Fig. 6(b). As the SQ surface is considered in finding the MPF, this
7 amended SR method also improves efficiency by reducing the scope of the search to the area
8 in which contact is most likely. Only neighbouring faces need to be checked in the following
9 searches to find a return.

10 There is a subtle difference to note between contacts with the faces of PHs, e.g., Fig.
11 1(b) or Fig. 3, and contacts with the edges or vertices of PHs, e.g., Fig. 5 or Fig. 6. Face
12 contacts yield a contact point on the surface of the SQ while edge/vertex contacts yield a
13 contact point which is on an edge of the PH.

14 This algorithm is only applicable to a convex PH and SQ: the outer normal vector is
15 used in the approach for PHs, while the SQs must be convex to avoid the possibility of
16 multiple contacts as the algorithm presented in this section gives at most one contact point.
17 Furthermore, the overlap between a PH and a SQ should be small to ensure accuracy. This
18 is because an identified contact point is always located on the surface of the PH or SQ; as
19 overlaps become large, the validity of choosing a surface point as the single contact point
20 becomes questionable.

21 The complete algorithm is summarised in Fig. 7. The six cases leading to termination
22 of the algorithm are shown in this flowchart. For odd cases, a contact is not found; for even
23 cases, a contact is found.

24 *2.4. Contact evaluation based on the contact point*

25 If a contact exists, a single contact point \mathbf{X}_c is found by following the algorithm in Fig.
26 7. The next step is to determine the contact normal and the interparticle overlap, both of
27 which are necessary to calculate a contact force. It is noted that these quantities are not
28 well defined in general for irregularly shaped particles [21]; however, the approach proposed

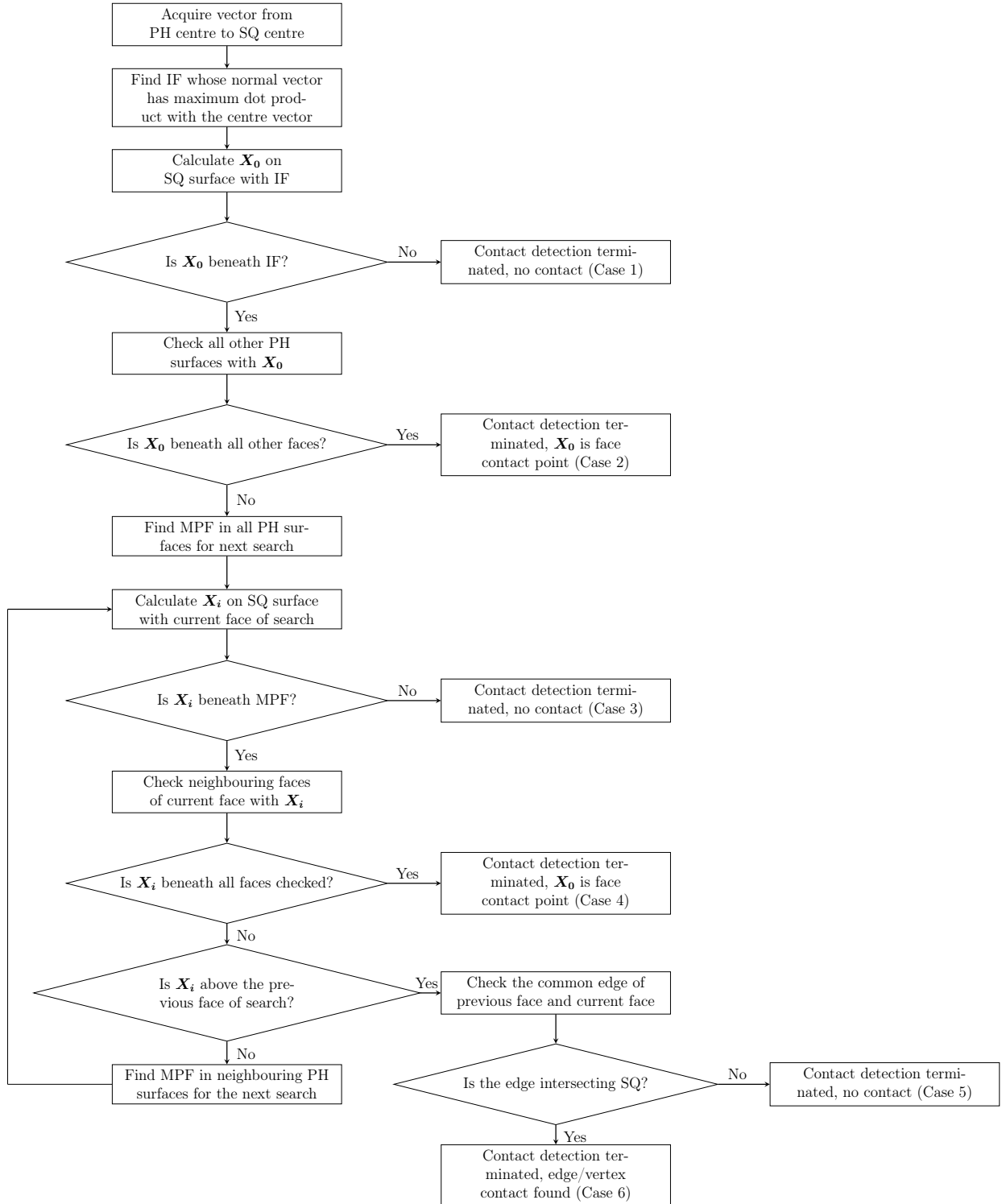


Figure 7: Flowchart showing the complete PH-SQ contact detection algorithm.

1 here gives physically reasonable values. Three cases must be considered separately: face
 2 contact, edge contact and vertex contact.

3 For face contact, the contact point lies on a single face of the PH which is therefore the
 4 contact plane. The normal vector of the contact plane is

$$\mathbf{n} = \mathbf{n}_c \quad (10)$$

5 where \mathbf{n}_c is the normal vector of the PH face which contains the contact point \mathbf{X}_c . The
 6 interparticle overlap is the distance between \mathbf{X}_c and the contact plane (Fig. 8(a)). This
 7 method originates from Feng et al. [30].

8 For edge contact, it would not be physically meaningful to adopt either of the PH faces
 9 forming the edge as the contact plane. Hence, a virtual contact plane is sought. Its normal
 10 vector is chosen to be the normalised sum of the normal vectors of the two faces forming
 11 the edge:

$$\mathbf{n} = \frac{\mathbf{n}_1 + \mathbf{n}_2}{\|\mathbf{n}_1 + \mathbf{n}_2\|} \quad (11)$$

12 where \mathbf{n}_1 and \mathbf{n}_2 are the normal vectors of the two faces forming the edge. The contact
 13 plane passes through \mathbf{X}_c with normal vector \mathbf{n} . The overlap is computed between the SQ
 14 and the contact plane [29] (Fig. 8(b)).

15 For vertex contact, a virtual contact plane is again required. Its normal vector is chosen
 16 to be the normalised sum of the direction vectors \mathbf{l} of all n edges meeting at the vertex:

$$\mathbf{n} = \frac{\sum_{i=1}^n \mathbf{l}_i}{\|\sum_{i=1}^n \mathbf{l}_i\|} \quad (12)$$

17 \mathbf{l} always points towards the vertex contact. This notation emphasises the adoption of the
 18 direction vectors of edges rather than normal vectors of faces for the edge contact case. As
 19 for the edge contact case, the overlap is computed between the SQ and the contact plane [29]
 20 (Fig. 8(c)).

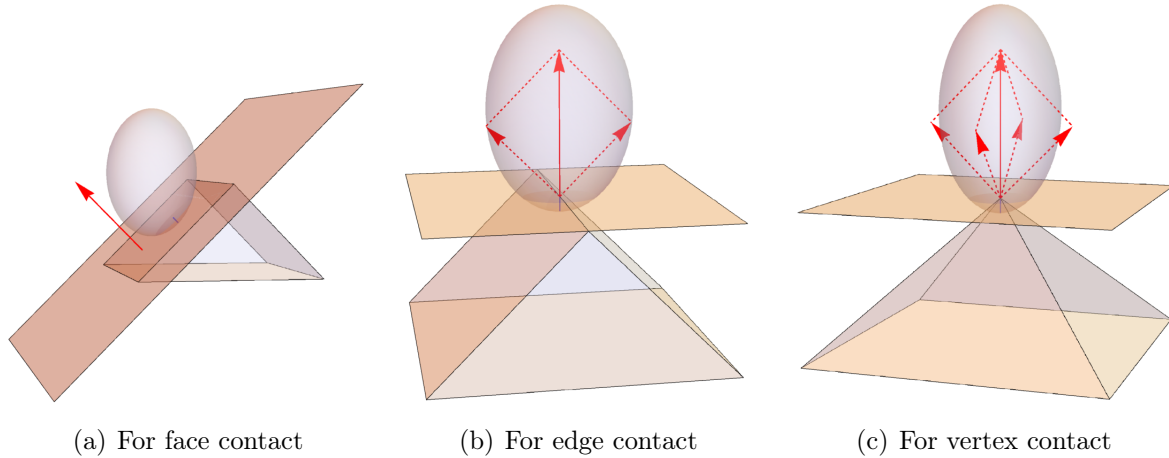


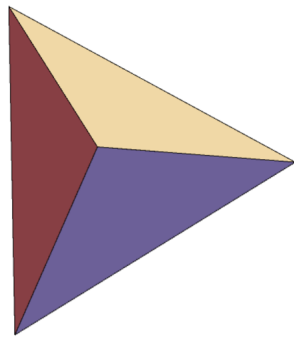
Figure 8: Contact evaluation for three different contact cases.

3. Implementation and verification of the contact detection algorithm

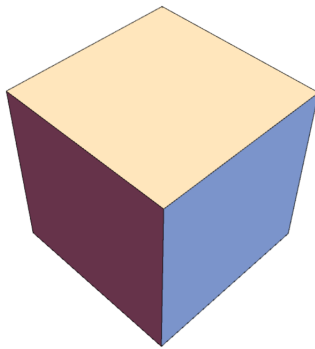
3.1. Implementation of algorithm

LIGGGHTS-PUBLIC v3.8.0 [37] implements the contact detection approach described in Section 2.1 for SQ–SQ, SQ–planar wall and SQ–line segment contacts. The LIGGGHTS implementation was used as a basis for the code implementation described in this paper. Since the algorithm requires knowledge of neighbouring faces, it is beneficial to build a relationship map of PH surface faces into the implementation. A graph model in which nodes are PH surface faces and links are relationships between these faces was adopted to satisfy this requirement. Two faces link only when they have a common edge, not if they only share a vertex. Fig. 9 shows three simple examples of this graph model. All faces are linked to each other for a tetrahedron; for a cube, each face is linked to four other faces; for a regular pyramid, any side face links to all other faces except the face directly opposite, while the square bottom face links to all faces.

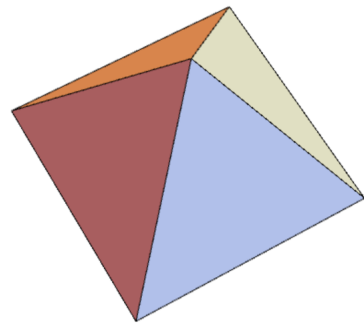
This PH model was implemented in C++ by creating a class which includes a set of planes enclosing the PH faces (stored using the outer normal vector and one specific point on each plane) and a 2D adjacency array to store the relationships between faces/planes. This is an easy way to model a static graph [38]. Within this class, the vertices are stored in a disordered array list. The vertices are traversed only when edge/vertex contact judgement is



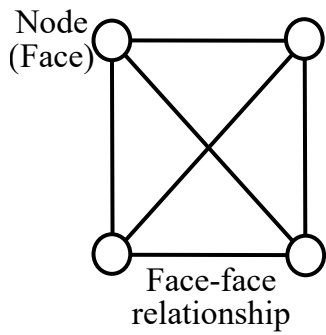
(a) Tetrahedron



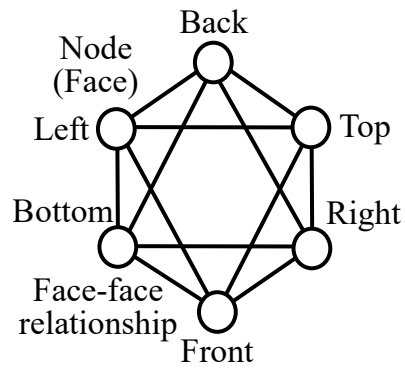
(b) Cube



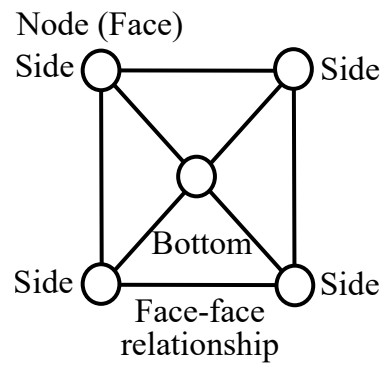
(c) Regular pyramid



(d) Tetrahedron model



(e) Cube model



(f) Regular pyramid model

Figure 9: Describing PHs using a graph model.

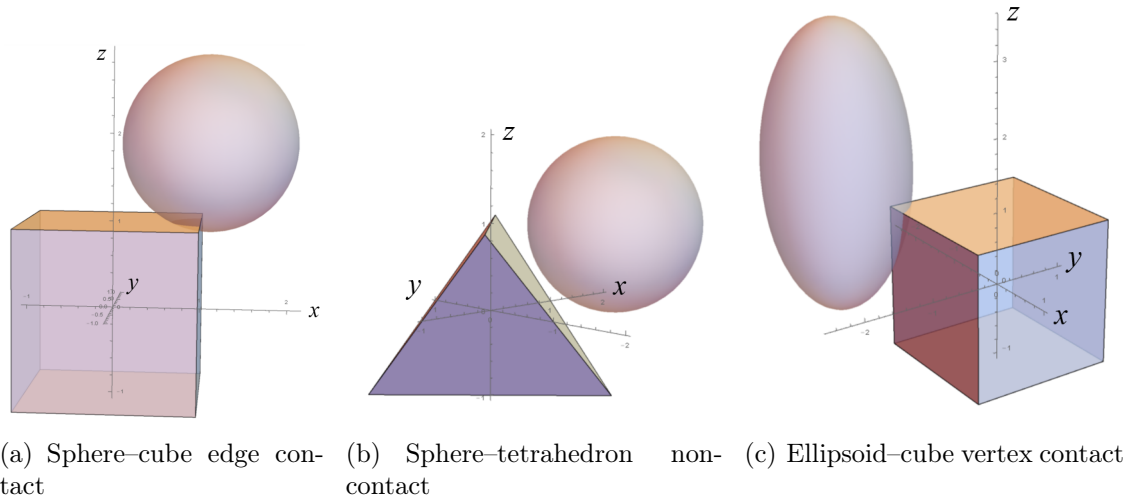


Figure 10: Three examples of verification cases.

1 needed. This traversal was not optimised to improve upon the time complexity of $O(n)$ [39]
 2 for the naive implementation which was considered acceptable. This class was combined
 3 with the SQ implementation from LIGGGHTS to create a standalone program for verifying
 4 one PH contacting one SQ.

5 3.2. Verification of algorithm

6 Initially, spheres, the simplest SQs, were placed in a wide range of contacting and non-
 7 contacting positions relative to a cube, e.g., Fig. 10(a). Contact/non-contact was determined
 8 using the code described in Section 3.1. The contact points identified were compared with
 9 the analytical contact points to confirm that the correct results were obtained. A wide
 10 range of other (non-)contact cases were checked, e.g., sphere-tetrahedron (Fig. 10(b)) and
 11 ellipsoid-cube (Fig. 10(c)), which also gave the expected results.

12 After successfully carrying out this initial stage of verification, the Monte Carlo method
 13 was adopted to randomly position a spherical SQ of radius 1 m in a large cubical space (edge
 14 length 6 m) containing a smaller cubical PH of edge length 2 m. The centre of the SQ could
 15 be any point within the $6\text{ m} \times 6\text{ m} \times 6\text{ m}$ cubical space shown in pink in Fig. 11. The
 16 cubical PH is shown in green in this figure. The centres of any spheres which contact the
 17 cube must lie within the round-cornered cubic space shown in yellow in Fig. 11. 1000000

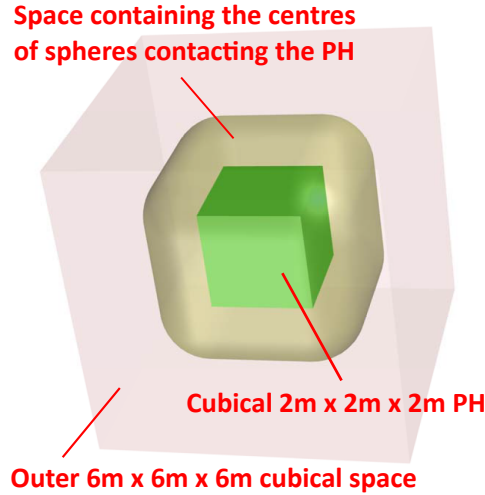


Figure 11: Setup for the Monte Carlo verification.

1 points were randomly generated within the pink cube as SQ centre points. This led to three
 2 possible outcomes for each point, with reference to Fig. 11:

- 3 1. The centre was inside the green PH. These points were neither identified as contacts
 4 nor as non-contacts but were counted separately as ‘Case 8’.
- 5 2. The centre was inside the yellow round-cornered cubic space but not inside the green
 6 PH. These are contact points, i.e., Cases 2, 4 or 6.
- 7 3. The centre was outside the yellow region, i.e., no contact (Cases 1, 3 or 5).

8 The code described in Section 3.1 was applied to each SQ centre point generated, leading
 9 to one of these three outcomes for each point. Analytically, the frequency with which these
 10 outcomes were observed must be proportional to the volumes of the three regions on Fig. 11.
 11 $V_{green} = 2^3 = 8 \text{ m}^3$, $V_{yellow} \approx 47.038 \text{ m}^3$, and $V_{pink} = 6^3 - V_{green} - V_{yellow} \approx 160.962 \text{ m}^3$. This
 12 is exactly what was observed, as shown in Table 1. The difference between the analytical
 13 and Monte Carlo results is less than 0.05%, further demonstrating the reliability of this
 14 algorithm for resolving PH–SQ contact.

Table 1: Comparison of the Monte Carlo verification with the analytical result based on volumes.

	Cases 1/3/5 (no contact)	Cases 2/4/6 (contact)	Case 8	Total
Monte Carlo count	745619	217499	36882	1000000
Monte Carlo (%)	74.56	21.75	3.69	100.00
Analytical volume (m ³)	160.962	47.038	8	216
Analytical %	74.52	21.78	3.70	100.00

1 4. Computational efficiency of algorithm

2 After the code had been verified, it could be used to assess the computational efficiency of
3 the contact detection algorithm. A similar Monte Carlo approach was used for this purpose
4 with the centre points of 10000 spherical SQs being randomly located within the space
5 shown on Fig. 11. Instead of only a cubical PH, all five convex regular polyhedra were used
6 for assessing computational efficiency: a tetrahedron, cube, octahedron, dodecahedron and
7 icosahedron which have 4, 6, 8, 12 and 20 faces, respectively. Sphere centres within the PHs
8 (Case 8) were disregarded for this benchmarking exercise.

9 Each determination of contact/non-contact was repeated 5000 times to ensure robust
10 data were gathered for total elapsed time. Times were measured using the *ctime* library [40];
11 this was confirmed to give almost identical results to class *std::chrono::system_clock* [41].
12 All benchmarks were run on a laptop computer with an Intel Core i7-6600U 2.60 GHz CPU.

13 The results are shown in Fig. 12. The elapsed calculation times are categorised by case
14 and accumulated curves are adopted for representation. It is noted that, for certain PHs,
15 some cases cannot occur, e.g., Case 4 for a cube. For any PH, Cases 1 and 2 are clearly the
16 fastest, then Cases 3 and 4 where present, and finally Cases 5 and 6 are the slowest by a
17 significant margin, which is expected from Fig. 7. The odd, non-contact cases are generally
18 slightly faster than the corresponding contact cases, as termination of the algorithm occurs
19 sooner for the non-contact cases than for the contact cases.

20 The effect of increasing the number of PH faces (and also edges and vertices) is perhaps
21 not as pronounced as might be expected. Comparing the graphs for the five different poly-
22 hedra on Fig. 12, there is a small time penalty associated with increasing the number of

1 faces but this is modest. This algorithm is therefore particularly efficient for polyhedra with
2 a large number of faces.

3 **5. Conclusions**

4 This paper presents a novel contact detection algorithm for a convex polyhedron (PH)
5 and superquadric (SQ) in DEM. This algorithm is based on contact detection between a SQ
6 and a planar wall, and terminates in one of six cases for each contact event: three cases in
7 which the PH and the SQ are in contact, and three in which they are not. This contact
8 detection algorithm has been implemented as a standalone program, based on the existing
9 SQ implementation in LIGGGHTS [37].

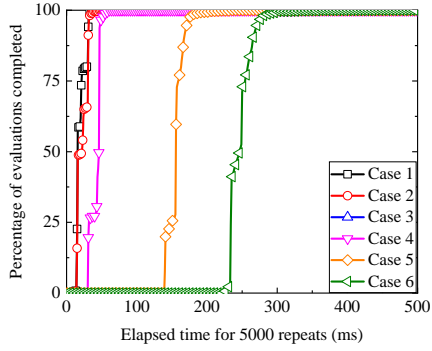
10 This program was used to verify the correctness of the algorithm and evaluate its effi-
11 ciency using the Monte Carlo method. The evaluation time largely depends on the point
12 at which the algorithm terminates. Non-contact cases are generally slightly faster than the
13 equivalent contact cases. The effect of increasing the number of PH faces on the evaluation
14 time is small, indicating that the proposed algorithm is particularly efficient for many-faceted
15 PHs.

16 **Acknowledgements**

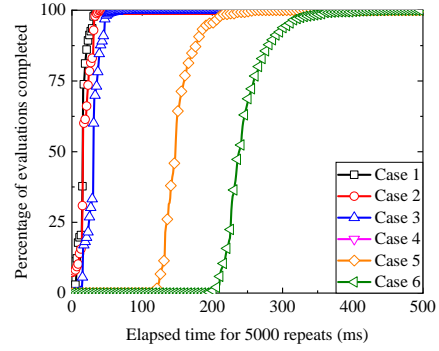
17 Dr Andrew Firth from the School of Engineering at The University of Edinburgh is
18 acknowledged. Funding from the University of Edinburgh Global Research Scholarship and
19 the China Scholarship Council (No. 201606260230) is also acknowledged.

20 **References**

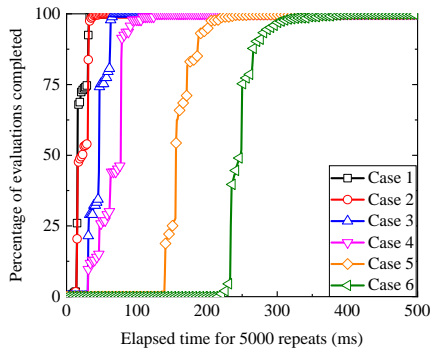
- 21 [1] P. A. Cundall, O. D. L. Strack, A discrete numerical model for granular assemblies, *Géotechnique* 29 (1)
22 (1979) 47–65.
- 23 [2] Itasca Consulting Group, Inc., Minneapolis: Itasca, PFC - Particle Flow Code, Ver. 6.0 (2018).
- 24 [3] DEM Solutions Ltd., <https://www.edemsimulation.com/>, EDEM 2019 User Guide (2019).
- 25 [4] V. Šmilauer et al, Reference manual, in: *Yade Documentation 2nd ed*, The Yade Project, 2015,
26 <http://yade-dem.org/doc/>. doi:10.5281/zenodo.34045.



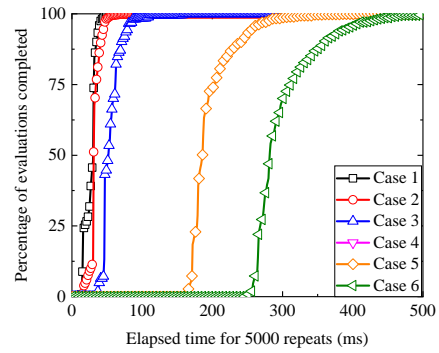
(a) Tetrahedron



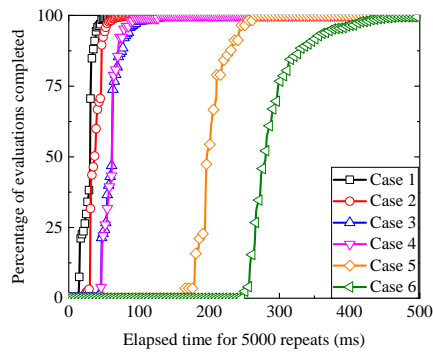
(b) Cube



(c) Octahedron



(d) Dodecahedron



(e) Icosahedron

Figure 12: Evaluation times, subdivided by case, for different types of PH.

- 1 [5] C. Kloss, C. Goniva, A. Hager, S. Amberger, S. Pirker, Models, algorithms and validation for opensource
2 DEM and CFD-DEM, *Progress in Computational Fluid Dynamics, an International Journal* 12 (2-3)
3 (2012) 140-152.
- 4 [6] G. Lu, J. R. Third, C. R. Müller, Discrete element models for non-spherical particle systems: from
5 theoretical developments to applications, *Chemical Engineering Science* 127 (2015) 425-465.
- 6 [7] D. Markauskas, R. Kačianauskas, A. Džiugys, R. Navakas, Investigation of adequacy of multi-sphere
7 approximation of elliptical particles for DEM simulations, *Granular Matter* 12 (1) (2010) 107-123.
- 8 [8] H. Kruggel-Emden, S. Rickelt, S. Wirtz, V. Scherer, A study on the validity of the multi-sphere discrete
9 element method, *Powder Technology* 188 (2) (2008) 153-165.
- 10 [9] M. Marigo, E. H. Stitt, Discrete element metod (DEM) for industrial applications: comments on
11 calibration and validation for the modelling of cylindrical pellets, *KONA Powder and Particle Journal*
12 32 (2015) 236-252.
- 13 [10] X. Lin, T.-T. Ng, Contact detection algorithms for three-dimensional ellipsoids in discrete element
14 modelling, *International Journal for Numerical and Analytical Methods in Geomechanics* 19 (9) (1995)
15 653-659.
- 16 [11] X. Lin, T.-T. Ng, A three-dimensional discrete element model using arrays of ellipsoids, *Géotechnique*
17 47 (2) (1997) 319-329.
- 18 [12] J. R. Williams, A. P. Pentland, Superquadrics and modal dynamics for discrete elements in interactive
19 design, *Engineering Computations* 9 (2) (1992) 115-127.
- 20 [13] C. Hogue, Shape representation and contact detection for discrete element simulations of arbitrary
21 geometries, *Engineering Computations* 15 (3) (1998) 374-390.
- 22 [14] J. R. Williams, R. O'Connor, A linear complexity intersection algorithm for discrete element simulation
23 of arbitrary geometries, *Engineering Computations* 12 (2) (1995) 185-201.
- 24 [15] J. R. Williams, R. O'Connor, Discrete element simulation and the contact problem, *Archives of Com-
25 putational Methods in Engineering* 6 (4) (1999) 279-304.
- 26 [16] E. G. Nezami, Y. M. A. Hashash, D. Zhao, J. Ghaboussi, Simulation of front end loader bucket-soil
27 interaction using discrete element method, *International Journal for Numerical and Analytical Methods
28 in Geomechanics* 31 (9) (2007) 1147-1162.
- 29 [17] Y. T. Feng, D. R. J. Owen, A 2D polygon/polygon contact model: algorithmic aspects, *Engineering
30 Computations* 21 (2/3/4) (2004) 265-277.
- 31 [18] G. Lu, J. R. Third, C. R. Müller, Critical assessment of two approaches for evaluating contacts between
32 super-quadric shaped particles in DEM simulations, *Chemical Engineering Science* 78 (2012) 226-235.
- 33 [19] G. G. W. Mustoe, M. Miyata, Material flow analyses of noncircular-shaped granular media using
34 discrete element methods, *Journal of Engineering Mechanics* 127 (10) (2001) 1017-1026.

- 1 [20] J. E. Andrade, K.-W. Lim, C. F. Avila, I. Vlahinić, Granular element method for computational particle
2 mechanics, *Computer Methods in Applied Mechanics and Engineering* 241 (2012) 262–274.
- 3 [21] Y. Feng, K. Han, D. Owen, A generic contact detection framework for cylindrical particles in discrete
4 element modelling, *Computer Methods in Applied Mechanics and Engineering* 315 (2017) 632–651.
- 5 [22] C. W. Boon, G. T. Houlsby, S. Utili, A new contact detection algorithm for three-dimensional non-
6 spherical particles, *Powder Technology* 248 (2013) 94–102.
- 7 [23] G. T. Houlsby, Potential particles: a method for modelling non-circular particles in DEM, *Computers
8 and Geotechnics* 36 (6) (2009) 953–959.
- 9 [24] Itasca Consulting Group, Inc., Minneapolis: Itasca, 3DEC - Three-Dimensional Distinct Element Code,
10 Ver. 5.2 (2016).
- 11 [25] P. A. Cundall, Formulation of a three-dimensional distinct element model. Part I. A scheme to detect
12 and represent contacts in a system composed of many polyhedral blocks, *International Journal of Rock
13 Mechanics and Mining Sciences & Geomechanics Abstracts* 25 (3) (1988) 107–116.
- 14 [26] E. G. Nezami, Y. M. A. Hashash, D. Zhao, J. Ghaboussi, Shortest link method for contact detection in
15 discrete element method, *International Journal for Numerical and Analytical Methods in Geomechanics*
16 30 (8) (2006) 783–801.
- 17 [27] W. Xu, H. Chen, Mesostructural characterization of particulate composites via a contact detection
18 algorithm of ellipsoidal particles, *Powder Technology* 221 (2012) 296–305.
- 19 [28] Z. Zhu, W. Xu, H. Chen, The fraction of overlapping interphase around 2D and 3D polydisperse non-
20 spherical particles: Theoretical and numerical models, *Computer Methods in Applied Mechanics and
21 Engineering* 345 (2019) 728–747.
- 22 [29] A. Podlozhnyuk, S. Pirker, C. Kloss, Efficient implementation of superquadric particles in discrete
23 element method within an open-source framework, *Computational Particle Mechanics* 4 (1) (2017)
24 101–118.
- 25 [30] Y. Feng, K. Han, D. Owen, Energy-conserving contact interaction models for arbitrarily shaped discrete
26 elements, *Computer Methods in Applied Mechanics and Engineering* 205 (2012) 169–177.
- 27 [31] K. Dong, C. Wang, A. Yu, A novel method based on orientation discretization for discrete element
28 modeling of non-spherical particles, *Chemical Engineering Science* 126 (2015) 500 – 516.
- 29 [32] E. G. Gilbert, D. W. Johnson, S. S. Keerthi, A fast procedure for computing the distance between
30 complex objects in three-dimensional space, *IEEE Journal on Robotics and Automation* 4 (2) (1988)
31 193–203.
- 32 [33] C. J. Ong, E. G. Gilbert, The Gilbert-Johnson-Keerthi distance algorithm: A fast version for incremen-
33 tal motions, in: *Proceedings of International Conference on Robotics and Automation, Vol. 2, IEEE,*
34 1997, pp. 1183–1189.

- 1 [34] G. V. den Bergen, A fast and robust GJK implementation for collision detection of convex objects,
2 Journal of Graphics Tools 4 (2) (1999) 7–25.
- 3 [35] S. Cameron, A comparison of two fast algorithms for computing the distance between convex polyhedra,
4 IEEE transactions on Robotics and Automation 13 (6) (1997) 915–920.
- 5 [36] S. Cameron, Enhancing GJK: Computing minimum and penetration distances between convex polyhe-
6 dra, in: Proceedings of international conference on robotics and automation, Vol. 4, IEEE, 1997, pp.
7 3112–3117.
- 8 [37] DCS Computing GmbH, <https://www.cfdem.com/media/DEM/docu/Manual.html>, LIGGGHTS(R)-
9 PUBLIC Documentation, version 3.x Edition (2016).
- 10 [38] K. Mehlhorn, Algorithms and Data Structures: The Basic Toolbox, Springer, Berlin, 2008.
- 11 [39] R. Sedgewick, Algorithms, Addison-Wesley Series in Computer Science, Addison-Wesley, Reading,
12 Mass., 1984.
- 13 [40] International Organization for Standardization, Programming languages – C (ISO/IEC 9899:1990)
14 (1990).
- 15 [41] International Organization for Standardization, Information technology – Programming languages –
16 C++ (ISO/IEC 14882:2011) (2011).