Theoretical and Applied Fracture Mechanics 72 (2014) 136–149

Contents lists available at ScienceDirect

Theoretical and Applied Fracture Mechanics

journal homepage: www.elsevier.com/locate/tafmec

A multi-shell cover algorithm for contact detection in the three dimensional discontinuous deformation analysis

Wei Wu [a,b,*], Hehua Zhu [a], Xiaoying Zhuang [a,c,*], Guowei Ma [b], Yongchang Cai [a]

[a] State Key Laboratory for Disaster Reduction in Civil Engineering, Department of Geotechnical Engineering, College of Civil Engineering, Tongji University, Shanghai 200092, China
[b] School of Civil and Resource Engineering, The University of Western Australia, M051, 35 Stirling Highway, Crawley, WA 6009, Australia
[c] Institute of Structural Mechanics, Bauhaus-University Weimar, Marienstrasse 15, 99423, Germany

ARTICLE INFO

Article history:
Available online 13 April 2014

Keywords:
Three dimensional discontinuous deformation analysis (3D DDA)
Neighbor searching
Contact detection
Computational cost

ABSTRACT

In three dimensional discontinuous deformation analysis (3D DDA), the contact detection between blocks is the most expensive part in terms of the total computational cost. The detection normally comprises two stages, namely the search of neighboring blocks and the identification of contact modes. The first stage aims to find out all possible neighboring blocks and the second is to identify the exact contact modes between of neighboring blocks from their vertices, edges and faces. In this paper, an efficient and robust spatial contact detection algorithm is presented linking the above mentioned two stages using a novel multi-shell cover (MSC) system and decomposition of geometrical sub-units. The present MSC method greatly reduces the contact detection volume and iterations. This paper also provides a unified formula of vertex to face and edge to edge contacts. The proposed method is implemented in a 3D DDA computer program. Numerical examples are tested and the results show improved computational efficiency comparing with existing methods.

1. Introduction

The numerical methods for rock mechanics achieved great developments in the past three decades as they provide complementary guidance and predictive information that cannot be achieved by the traditional empirical ways. The most widely used continuum approaches at present are the finite element method (FEM), the finite difference method (FDM), and the boundary element method (BEM). They continue to be the mainstream in the numerical analysis of substantial rock engineering problems. However, they are faced with difficulties in dealing with problems with numerous moving boundaries, because the mesh has to be updated to conform to the evolving geometry at each time step [12]. To remove this difficulty, a variety of innovative numerical methods have been developed including the meshless methods [28–35] and the extended finite element methods [36,37], etc. However, these methods are mainly designed for dealing with material failure process. For the modeling of discrete block systems where the kinematics and removability of the blocks are concerned, the discontinuous deformation analysis (DDA) based on explicit dynamics is more suitable [1].

DDA is a discontinuum-based numerical method for blocky systems with large displacements [1,2]. Shi developed 2D and 3D DDA methods for dynamic discontinuous simulation [3–5]. The key of large displacement simulation in DDA is the contact theory including neighbor searching, contact pattern detection and contact force calculation. Contact detection process is to find candidate neighboring blocks and identify contact groups of vertices, edges and facets (geometrical elements) in a multi-block system. Contact relations between discontinuous objects are complicated both in geometry and algebra, especially for 3D cases. It requires high computational cost in contact detection of large scale problem. DDA can be classified as a type of Discrete Element Method (DEM). For 3D particle geometries, contact detection of DEM can take up to 80% of the total analysis time [19]. Thus, the computational speed and stability of 3D DDA are strongly dependent on the efficiency and robustness of contact searching and definition algorithm, which affects the simultaneous equilibrium equations solved in each step of open–close iteration (OCI) [21]. It should be also mentioned that the generation of blocks in the preprossessing can also be time consuming. The computational cost of model setup and geometry operation for numerical methods in terms of accuracy and adaptability with respect to input data has already attracted much attention. Recently, there are intense research interests focusing on incorporating the geometry and field interpolation with by the same splines, which is known as isogeometric

[*] Corresponding authors. Address: Yantu Buiding, 1239 Siping Rd, Shanghai 20092, China. Tel.: +86 65983982 (X. Zhuang).

E-mail address: xiaoying.zhuang@gmail.com (X. Zhuang).

http://dx.doi.org/10.1016/j.tafmec.2014.03.004
0167-8442/© 2014 Elsevier Ltd. All rights reserved.

analysis [38]. In comparison, the generation of blocks in DDA is lacking studies in model generation [39–45].

In contact detection, the initial contact types between blocks (vertex-to-vertex, vertex-to-edge, vertex-to-face, edge-to-edge, edge-to-face and face-to-face) are determined and translated to sub-contact types (vertex-to-face and edge-to-edge) [6,7]. Contact springs are added according to geometry and state (open, sliding or locked), contact forces and status are calculated in a series of open-close iterations (OCI).

Neighbor searching is the first step in contact detection. Apparently, a search algorithm over the whole domain is inefficient and unnecessary. In neighbor searching, algorithms for reducing search domain are used to find neighboring discontinuous blocks. Indeed, neighbor searching is an indispensable process not only for DDA and DEM but also for molecular dynamics and other discontinues methods. This is due to that fact that an automatic search technique is necessary for large-scale large displacement simulation.

Various efforts have been devoted to the development of efficient neighbor search method. Greengard [8] made extensive uses of a body-based cell approach in conjunction with octrees. They applied this method in molecular dynamics analysis. Williams [9] presented an algorithm for contact resolution with spatial heapsort and applied it to 2D DEM simulation. Munjiza [10] provided a spatial hashing algorithm based on no binary search (NBS) and tested it with large-scale discrete element simulations of bodies of similar size. Perkins [11] developed a spatial sorting based neighbor search algorithm named double-ended spatial sorting (DESS). The DESS algorithm reduces the effects of variation in size by using a spatial sorting technique applied to both ends of the object's projection along each orthogonal axis.

Contact pattern identification is the second step in contact detection. After neighbor searching calculation, specific contact pattern and contact position between two contacting blocks are ascertained. The common plane (CP) method and the geometric analysis are the two classical approaches for contact pattern identification [22,26].

Common plane (CP) method is a widely used contact pattern identification algorithm used in the DEM proposed by Cundall [14]. A common-plane bisects the space between a pair of contacting polyhedra. Both polyhedra will intersect the CP if they are in contact to each other and vice versa. Various methods have been proposed based on CP algorithms. An algorithm known as GJK, developed by Gilbert, Johnson and Keerthi, was used to compute the Euclidean distance between a pair of objects represented by convex polyhedron in three dimensional space [17]. Ong improved GJK algorithm in speed with the additional structural information on the objects. They modified GJK to apply it on objects containing complex vertices or faces [17]. Mirtich [18] provided a collision detection algorithm named Voronoi-clip for polyhedra objects specified by a boundary representation. V-clip tracks the closest pair of features between convex polyhedra in impulse-based simulation in which all contact interactions between bodies are affected through collisions. Nezami [19] introduced an efficient algorithm named the fast common plane (FCP) method to find the common plane between two polyhedrons. Properties of the CP are utilized to limit its position and reduce iterations. On this basis, also the shortest link method (SLM) was developed by Nezami to obtain the shortest link and the CP in a more efficient way [20]. Nezami pointed that the perpendicular bisector plane (PBP) of the shortest link between two particles is the CP. One disadvantage of the CP method is the limitation of convex blocks only. In case of concave blocks, error detection will occur. This reduces the usefulness of the method for substantial rock analysis since it is not appropriate to assume all blocks are concave.

On the other hand, developing algorithms calculating contact patterns among blocks directly based on their geometrical characteristics is supposed to be another way to solve the disadvantage of CP method [22]. In geometric algorithms, a polyhedron is discretized into geometric elements including vertexes, edges and facets. The 3D contact detection of polyhedral blocks includes vertex-to-vertex, vertex-to-edge, vertex-to-face, edge-to-edge, edge-to-face and face-to-face contacts [21]. Intuitively, all other contact patterns, such as vertex-to-vertex, vertex-to-edge, face-to-face, edge-to-face and edge-to-edge, respectively, can be converted into vertex-to-face contacts [15]. Wu [13], Jiang [6,23] and Liu [24] proposed a vertex-to-face contact algorithm based on geometric analysis for 3D DDA. Keneti and Jafari [25] developed a new contact detection algorithm considering main planes and dominant contacts to identify contact points and types. Beyabanaki and Mikola [16] offered an approach to identify the contact pattern between two blocks using a closest point searching algorithm as well as other 3D contact algorithms for improved efficiency [27].

The above studies mainly focus on the algorithms to select contact candidate blocks and contact patterns precisely and efficiently. After contact detection, initial contact states are set by adding penalty springs and/or frictional force [21]. Then open–close iteration (OCI) process in which an open or closed state of contact is to determine the contact states of the detected contacts in the system.

Fig. 1 shows the flowchart of a DDA code, including a geometry analysis algorithm.

As shown in Fig. 1, neighbor searching and contact pattern identification are collectively referred as contact detection which is a key process in the DDA formulation. In rock engineering, such as the cross section of a tunnel or side slopes may contain thousands or more rock blocks that govern the overall stability of rock mass. In 3D DDA, computational cost increases exponentially with respect to the number of block quantity. And most of the computational time is spent in neighbor searching and contact pattern identification.

There are two types geometrical searching loops in contact detection, namely neighbor searching loops between blocks and geometrical loops between geometrical elements. In searching loops between blocks, distances calculation of block centroids and boundaries are repeated in every time step. However, in the loops between vertices, edges and faces of blocks, contact relation judgment is complex and time-consuming. Apart from distance judgment, three dimensional projections, entry face searching and invasion judgment may occur in contact pattern identification process. Thus the contact pattern identification costs more CPU time than those in block neighbor searching generally.

In presently available algorithms, neighbor searching and contact pattern identification are the two independent stages in computations. A neighboring block pair should be detected first, and then followed by the identification of contact patterns between geometrical elements (vertices, edges and faces) of the two blocks. Due to the complexity of contact pattern identification, the more geometrical elements included in a neighboring, the higher is the computing cost.

Therefore, to reduce the unnecessary neighbor searching loops and narrow the search domain of geometrical loops will greatly improve the computational efficiency in contact detection. In this paper, a new spatial contact detection method bridging between the neighbor searching and contact pattern identification using a multi-shell cover system is presented. The multi-sphere cover system is a series of cubes or spheres used for neighbor searching. Instead of using a single circumscribed sphere or cubic shell to reduce the volume of contact detection shell, the present multi-sphere divides geometrical elements into several groups. In other words, each shell in this system is linked to a subgroup of the vertices, edges and facets of a single block. In this way, the quantity

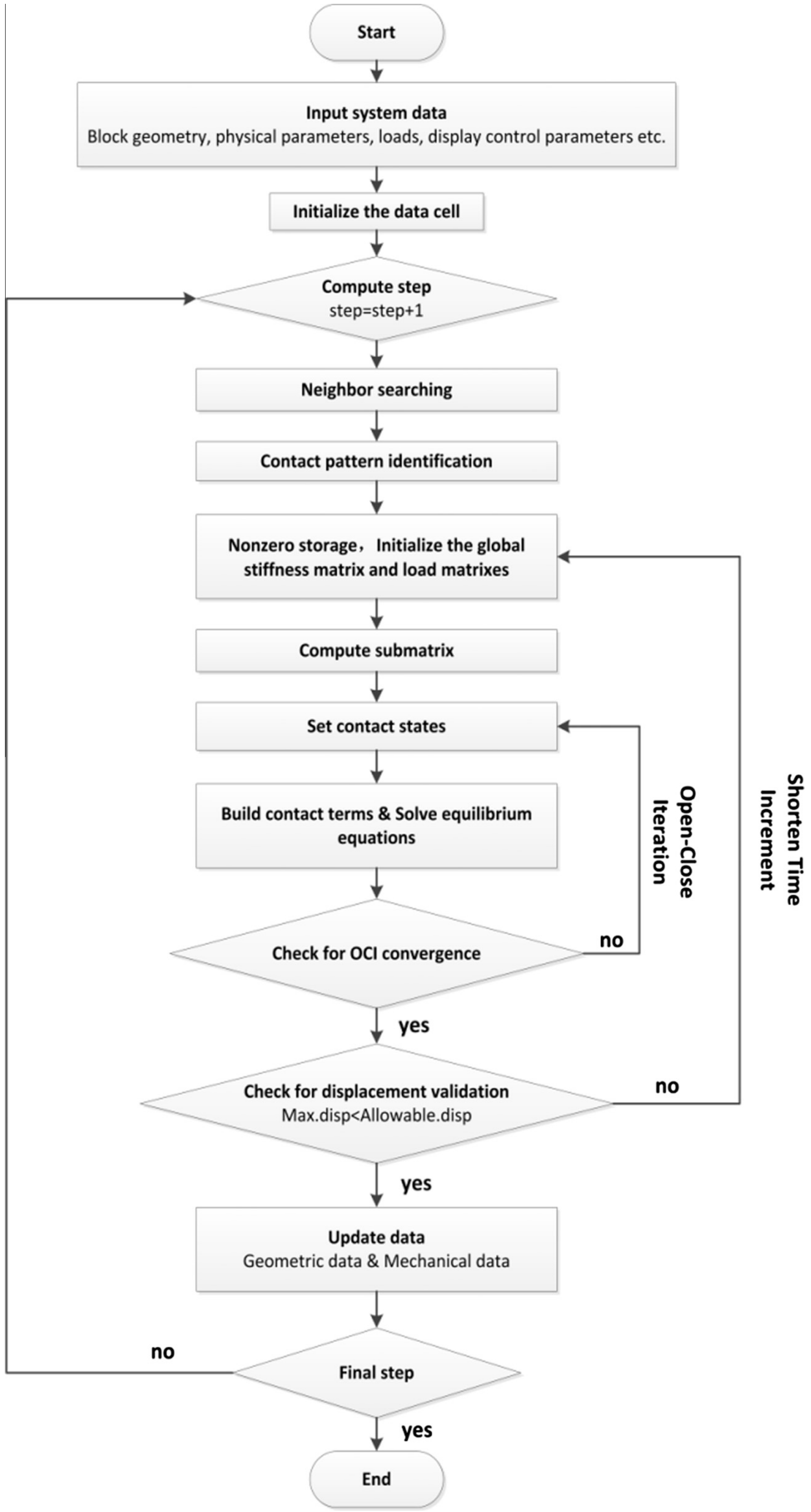

Start
Input system data
Block geometry, physical parameters, loads, display control parameters etc.
Initialize the data cell
Compute step
step=step+1
Neighbor searching
Contact pattern identification
Nonzero storage，Initialize the global stiffness matrix and load matrixes
Compute submatrix
Set contact states
Build contact terms & Solve equilibrium equations
Check for OCI convergence
no
Open-Close Iteration
yes
Check for displacement validation
Max.disp<Allowable.disp
no
Shorten Time Increment
yes
Update data
Geometric data & Mechanical data
Final step
no
yes
End

Fig. 1. Flowchart of a DDA code.

of searching loops between geometrical elements decreases. To highlight the features of this algorithm, we will compare the present method with other spatial sphere and cubic cell based detection algorithms. Finally, a series of numerical tests are performed to compare the performance of the new algorithm with other existing algorithms.

2. 3D DDA formulation

In discontinuous deformation analysis, the computational *time steps* are used for both statics and dynamics [6]. The large deformation in one block and the large relative movements between blocks are accumulated through many time steps [7]. The condition of linear elasticity, uniform stresses and strains are assumed for the whole process of simulation. The motion and deformation of each block are defined by 12 independent deformation variables in the deformation matrix denoted as

$$[D_i] = [u_0, v_0, w_0, \alpha_0, \beta_0, \gamma_0, \varepsilon_x, \varepsilon_y, \varepsilon_z, \gamma_{xy}, \gamma_{yz}, \gamma_{zx}] \tag{1}$$

where $\varepsilon_x, \varepsilon_y, \varepsilon_z, \gamma_{xy}, \gamma_{yz}, \gamma_{zx}$ are the three normal strains and three shear strains of the block, (u_0, v_0, w_0) and $(\alpha_0, \beta_0, \gamma_0)$ are the x-, y- and z- normal displacement and translations of the block's center of mass, respectively, (x_0, y_0, z_0) are rotations of the block along the x-, y- and z-axes. The x-, y-, z-displacements (u, v, w) of an arbitrary point (x, y, z) in the block can be expressed in terms of the deformation variables of $[D_i]$ as

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} = [T_i][D_i] \tag{2}$$

where

$$[T_i] = \begin{pmatrix} 1 & 0 & 0 & 0 & (z-z_0) & -(y-y_0) & (x-x_0) & 0 & 0 & \frac{(y-y_0)}{2} & 0 & \frac{(z-z_0)}{2} \\ 0 & 1 & 0 & -(z-z_0) & 0 & (x-x_0) & 0 & (y-y_0) & 0 & \frac{(x-x_0)}{2} & \frac{(z-z_0)}{2} & 0 \\ 0 & 0 & 1 & (y-y_0) & -(x-x_0) & 0 & 0 & 0 & (z-z_0) & 0 & \frac{(y-y_0)}{2} & \frac{(x-x_0)}{2} \end{pmatrix} \tag{3}$$

$[T_i]$ is the displacement transformation matrix of the block. For a system containing n blocks, the simultaneous equations can be presented as

$$\begin{pmatrix} K_{11} & K_{12} & K_{13} & \cdots & K_{1n} \\ K_{21} & K_{22} & K_{23} & \cdots & K_{2n} \\ K_{31} & K_{32} & K_{33} & \cdots & K_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ K_{n1} & K_{n2} & K_{n3} & \cdots & K_{nn} \end{pmatrix} \begin{pmatrix} D_1 \\ D_2 \\ D_3 \\ \vdots \\ D_n \end{pmatrix} = \begin{pmatrix} F_1 \\ F_2 \\ F_3 \\ \vdots \\ F_n \end{pmatrix}. \tag{4}$$

In the coefficient matrix of Eq. (4), each $[K_{ij}]$ is a submatrix. The submatrix $[K_{ii}]$ depends on the material properties of block i, and $[K_{ij}]$ $(i \neq j)$ is determined by the contact condition between block i and block j. $[D]$ and $[F]$ are $12n \times 1$ displacement and force vectors, respectively. In total, the quantity of displacement unknowns is the sum of the degrees of freedom of all the blocks.

Eq. (4) is derived by minimizing the total potential energy Π_p of the block system [5]. The i-th row of Eq. (4) consists of 12 linear equations

$$\frac{\partial^2 \Pi_e}{\partial d_{ri} \partial d_{si}} = 0 \quad r, s = 1, 2, 3, \ldots, 12$$

where r and s are the numbers of independent deformation variables for two contacting blocks respectively.

The elements k_{ij} of $[K_{ij}]$ and f_i of $[F_i]$ can be obtained from Eqs. (5) and (6), respectively that

$$k_{ij} = \frac{\partial^2 \Pi_e}{\partial d_{ri} \partial d_{si}} \quad r, s = 1, 2, 3, \ldots, 12 \tag{5}$$

$$f_i = -\frac{\partial^2 \Pi_\sigma(0)}{\partial d_{ri}} \quad r = 1, 2, 3, \ldots, 12. \tag{6}$$

The contact energy Π_c is divided into three components, namely,

(1) Strain energy of the normal contact spring

$$\Pi_n = \frac{1}{2} k_n (d_n)^2 \tag{7}$$

(2) Strain energy of the shear contact spring

$$\Pi_s = \frac{1}{2} k_s (d_s)^2 \tag{8}$$

(3) Potential energy of the frictional force

$$\Pi_f = F_f d_f = \frac{k_n d_n \tan(\varphi)}{|\vec{l}\,|} \vec{l} \tag{9}$$

where k, d, F and φ are the stiffness of the spring, displacement, the friction force and the friction angle of the block, respectively. Besides, $\vec{l}$ is the sliding direction vector, and the subscripts n, s, and f are the mean normal direction, shear direction, and frictional direction, respectively.

In the "locked" contact state, the contact energy is the sum of the normal and shear spring strain energies. But when the contact state is "sliding" meaning the relative displacement along the tangential direction of two faces, it is the sum of the normal spring strain energy and friction energy. In this paper, the formuation of 3D DDA are adopted mainly from Yeung [7] and Ahn [21].

3. Previous contact detection algorithms in 3D DDA and DEM

3.1. Neighbor search

Among the most classical algorithms available for the search of neighbor blocks are no binary search (NBS) algorithm [10] and double-ended spatial sorting (DESS) [11]. A review of neighbor search methods is available in [9].

In 3D NBS, each block is approximated with a sphere shell. The diameter of the sphere is obtained from the size of blocks in the system. Thus the NBS contact detection algorithm builds a system consisting sphere shells occupying finite space, shown in Fig. 2.

The task of NBS is to find all shell couples that are close to each other in a sense that the distance between their center points is less than or equal to the sum of their radii, in other words that they overlap or touch.

However, if there are significant differences in the sizes of blocks or some blocks are relatively lathy in shape, the CPU time for neighbor searching would increase notably. This is due to the

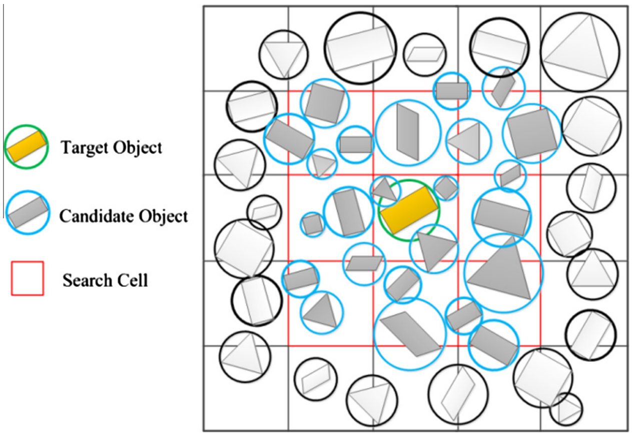


Fig. 2. No binary search (NBS) in 2D (reproduced from [10]).

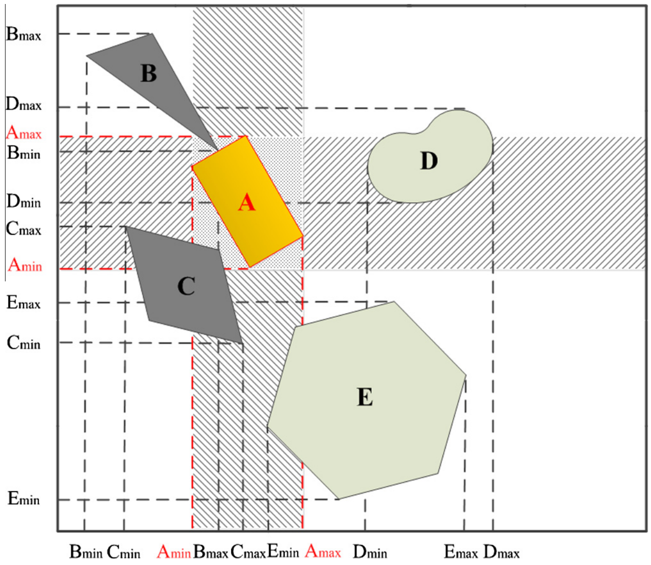
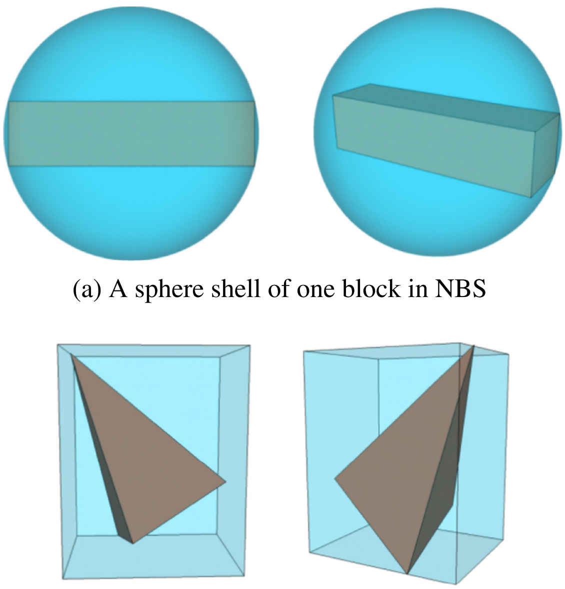


Fig. 3. Double-ended spatial sorting (DESS) in 2D (reproduced from [11]).

reason that large quantity of empty space resulting in unnecessary search takes up too much time, as shown in example of Fig. 4(a).

The double-ended spatial sorting (DESS) algorithm shown in Fig. 3 was developed from simple spatial sorting [11]. Six axis-extremities of each block are calculated by searching vertex coordinates in three dimensional cases. The values of the extremities are the real valued projections of that bound on the given axis. In other words, they are the lower or upper bounds of each block as projected on the axes. Mathematically, two blocks have the potential to contact each other only when their bounding volumes overlap.

DESS has its disadvantages under some specific circumstances where edges or boundary surfaces of searching boundaries are parallel to coordinate axes in 2D cases or coordinate planes in 3D. Thus if the edges or boundary surfaces has angles of 45° approximately with coordinate axes or planes, there would be invalid searching processes due to relatively large blank space of search boundaries as shown in Fig. 3(b).



(a) A sphere shell of one block in NBS

(b) A cubic shell of one block in DESS

Fig. 4. Search shell of a block in 3D.

3.2. Geometric resolution

In the second stage, called geometric resolution, pairs of contacting objects obtained from the first stage are examined in more detail to find the contact points or area and calculate the contact forces.

In rock mass engineering where discontinuous numerical methods are applied principally, concave objects are common for most engineering problems. Though CP algorithm performs sufficient accuracy and high efficiency in 3D discrete simulation, it still cannot be applied to concave block systems directly. Cundall [14] pointed that convex blocks can be decomposed into two or more convex blocks: one is termed a “master block”; the others are “slave blocks”. But this requires block division and concave–convex inspection for all blocks. Thus, geometric algorithms are utilized expediently in rock mass engineering and other simulations including concave or other complex objects.

In the neighbor search process, possible contact relations between blocks are ascertained. Then, contact patterns of these blocks need to be identified. Contact pattern is identified by judging relations between geometric elements (vertices, edges and faces). Ahn (2010) introduced the process of distance check in DDA contact definition. Firstly, the distance between two geometric elements of a contacting blocks pair is calculated and checked. The distance between two geometric elements is calculated by 3D vector projection analysis. Then quantity and types of contacts checked for distance between two blocks are recorded.

Liu and Lemos (2001) point out that contact patterns include seven types, namely vertex-to-vertex, vertex-to-edge, vertex-to-face, intersecting edge-to-edge, parallel edge-to-edge, edge-to-face and face-to-face. These types are defined by detected geometric contact relations and block structures (Ahn, 2011). The quantity and types of distance judgment contacts are used to identify contact type as shown in Table 1.

There are three types of contact elements in contact patterns identification, edges and faces. Take a hexahedron for example. It includes eight vertexes, twelve edges and six faces, totally 26 sub-units. Thus in a process to ascertain the contact pattern between two hexahedra, there are 676 possible combinations. A comparison of contact combination with the new algorithm is introduced in Section 3.2.

Jiang [6] pointed out that all types of 3D contacts can be converted to the type of vertex to face contact. After the contact type is identified, this original contact could be further simplified into several point-to-plane contacts. Then the contact springs and frictional force are added between a vertex of the penetrating block and a face of the penetrated block. The vertex named penetrating point passes through the face called entrance plane in the contacting stage.

Table 1
Relations between contact type and distance check (after Ahn, 2011).

Structural contact type	Quantity of geometric contacts	Possible combinations of geometric contact
Vertex-to-vertex	1	[VV] only
Vertex-to-edge	1	[VE] or [EV]
Vertex-to-face	1	[VF] or [FV]
Intersecting edge-to-edge	1	[EE] only
Parallel edge-to-edge	2	Two of any [VV], [VE], or [EV]
Edge-to-face	2	Any combination is possible.
Face-to-face	3 Or more	Any combination is possible.

Beyond point-to-plane, edge-to-edge contact is also a basic contact type. Yeung and Jiang [7] presented an edge-to-edge contact model with CP algorithm to be used in 3D DDA, including the detection of edge-to-edge contact type, judging of the first entrance faces, and the criteria for inter-penetration. Wu [22] discussed another edge-to-edge contact algorithm using geometric and vectors analysis to transfer the edge-to-edge contacts to the vertex-to-face ones.

4. A new 3D multi-shell contact detection algorithm

Contact detection in 3D cases is more complicated than that in 2D cases for more dimensions and degrees of freedom. As mentioned in Section 1, contact detection is a costly process in multi-block simulation in 3D DDA and DEM. The high computing cost of contact detection limits the development and engineering application of 3D DDA. Consequently, contact detection algorithm with high efficiency and sufficient accuracy is an important part for 3D discontinuous numerical simulation.

4.1. Neighbor search

In rock engineering, rock joints occupy much less volume compared to the intact rock blocks. Rock blocks are usually closely spaced and contact neighboring blocks in different patterns. In previous neighbor searching algorithm as NBS and DESS, there is a high probability that contact detection shell of a block with elongated or other irregular shape is calculated to intersect the shell of another non-contacted block physically. As shown in Fig. 5(a) and (d), a lathy block using NBS algorithm only has two blocks contacted actually while its contact detection shell intersects other four shells. In DESS algorithm described in Fig. 5(b) and (e), a lathy block with a 45-degree slant to x–y axis plane also has two blocks contacted actually, but its contact detection shell intersects additional eight shells.

As a result of complexity and expensive costs in 3D contact pattern identification, smaller contact detection space will reduce the quantity of candidate blocks detected in neighbor search. Figs. 5(c) and (f) show the contact search shells of the multi-cube cover systems. The shells based on a block form a group of cubes covering all the vertices, edges and faces of this block. Each face of the rectangle is parallel to the coordinate planes which is similar to that in DESS.

The shells cover for a convex block could be generated in the following steps:

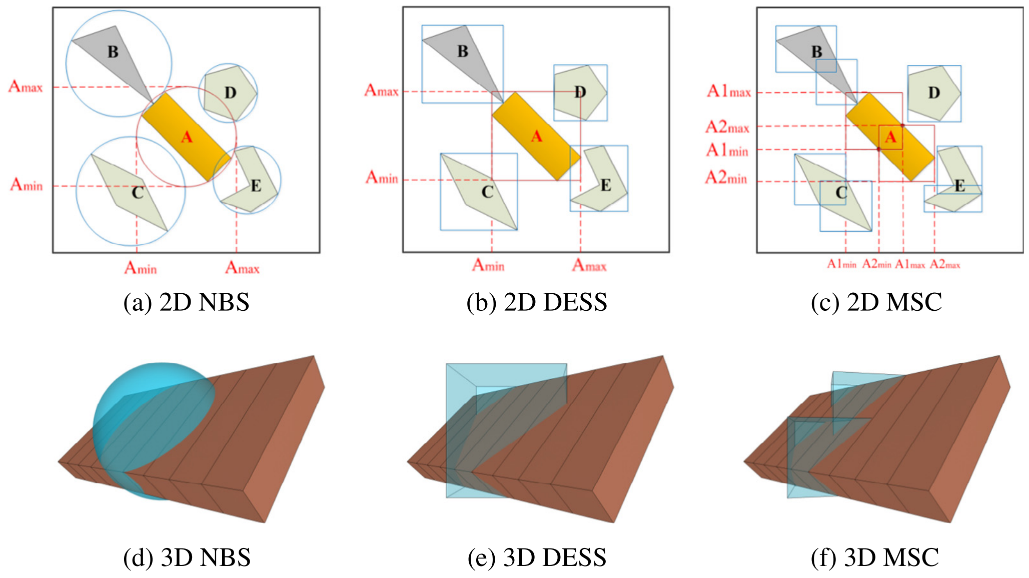
(1) Divide the possible space of the blocky system into a series of subspace and store the maximum and minimum x, y, z coordinates values of each subspace.
(2) Divide each long edge (length $> \lambda$, λ is a user-defined critical length) into N segments respectively.
(3) Divide all the vertices and diversion points of this block into groups and store the number of block which the group belongs to.
(4) Store the maximum and minimum x, y, z coordinates values of each group.
(5) Store the number of subspace the group located in.

Only the data of steps (4) and (5) need to be updated by time steps, the other steps are performed before the first time step. N in step (2) is a positive integer greater than or equal to 2 and its influence on computational efficiency is discussed in Section 5.

For a concave block shown in Fig. 6(a), the group is divided according to the division of convex sub-blocks included. Fracture modeling using numerical methods is well-developed following major advances in the past decade or two such as the extended finite element method (XFEM) and meshless methods [28–35]. Cracked objects are concave in most cases and can be divided into sub-blocks according to the cracks in these objects (as shown in Fig. 6(b)).

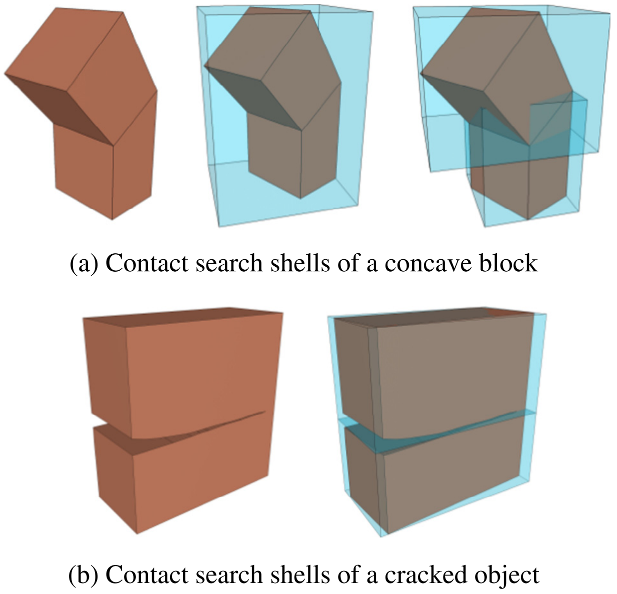
Among the five 2D blocks as shown in Fig. 5(a)–(c), only blocks A and B are physically contacted. When the NBS is applied, the sphere contact detection shells enlarged the actual block volume with nugatory empty space. A false detection occurs when two blocks do not contact each other physically, but their contact shells overlap. DESS has the same problem when the edges of the blocks are not parallel to the coordinate planes. Thus, there are five contact overlaps detected by search shells with NBS and DESS. However, MSC algorithm reduces the total volume of contact shells compared with NBS and DESS. MSC algorithm could lessen false detections in some cases and only one shell overlap is detected in Fig. 5(c).

Three dimensions currently include more false detections than those in two dimensions in similar models. Take models with 9 blocks and 8 actual block overlaps in Fig. 5(d)–(f) for example.



(a) 2D NBS
(b) 2D DESS
(c) 2D MSC
(d) 3D NBS
(e) 3D DESS
(f) 3D MSC

Fig. 5. Contact search shells in NBS, DESS and MSC.


(a) Contact search shells of a concave block

(b) Contact search shells of a cracked object

Fig. 6. Comparison of contact search shells for concave objects.

There are respectively 26 shell overlaps in NBS and DESS while 15 contact block pairs are detected in MSC.

Therefore, the contact search space volume is reduced by using this system, especially for elongated blocks. Furthermore, all the geometric elements could be divided into groups to reduce quantity of distance searching loops in contact pattern identification and details are given in Section 4.2.

4.2. Contact pattern identification

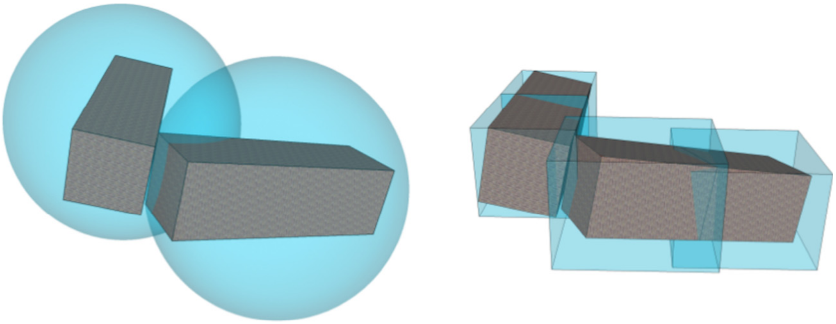
There are three types of contact sub-units in contact patterns identification, vertexes, edges and faces. Take the hexahedron for example. It includes eight vertexes, twelve edges and six faces, totally 26 sub-units.

In the process of distance judging in geometric resolution, the distances between two geometric sub-units of a pair of contacting blocks are calculated and checked. In previous algorithms as NBS and DESS, every geometric sub-unit of the two blocks should be considered and involved in a full range searching loop for distance judging. As shown in Fig. 7(a), two contacting blocks using single searching shell each have 26 geometric sub-units. Full range search among the geometric sub-units leads to numerous distance judging loops as described in Fig. 8(a).

In the algorithm presented in this paper, a block is divided into several parts and each part is covered by a single shell. Meanwhile, geometric sub-units of the block are divided into several groups according to the detecting shells. Taking Figs. 7 and 8(b) for example, a hexahedron is covered by two cubic shells and they also divide 26 sub-units into two groups. Each of group A and B includes 4 vertexes, 8 edges and 5 faces as shown in Fig. 8(b). A geometric sub-unit could belong to different shells of one block at the same time. Loops involve the common geometric sub-units only perform once to avoid unnecessary repetition. When shell A of block 1 inserts shell A of block 2, there is 289 possible combinations between the sub-units of the two shells while 338 possible combinations between the two blocks. A full range loop is not necessary since only vertex to face and edge to edge searches are done in distance calculation. Quantities of vertex to face and edge to edge contacts calculated by projector distance between neighboring blocks are utilized to judge real contact sub-units and type. The details are given in Section 4.3.

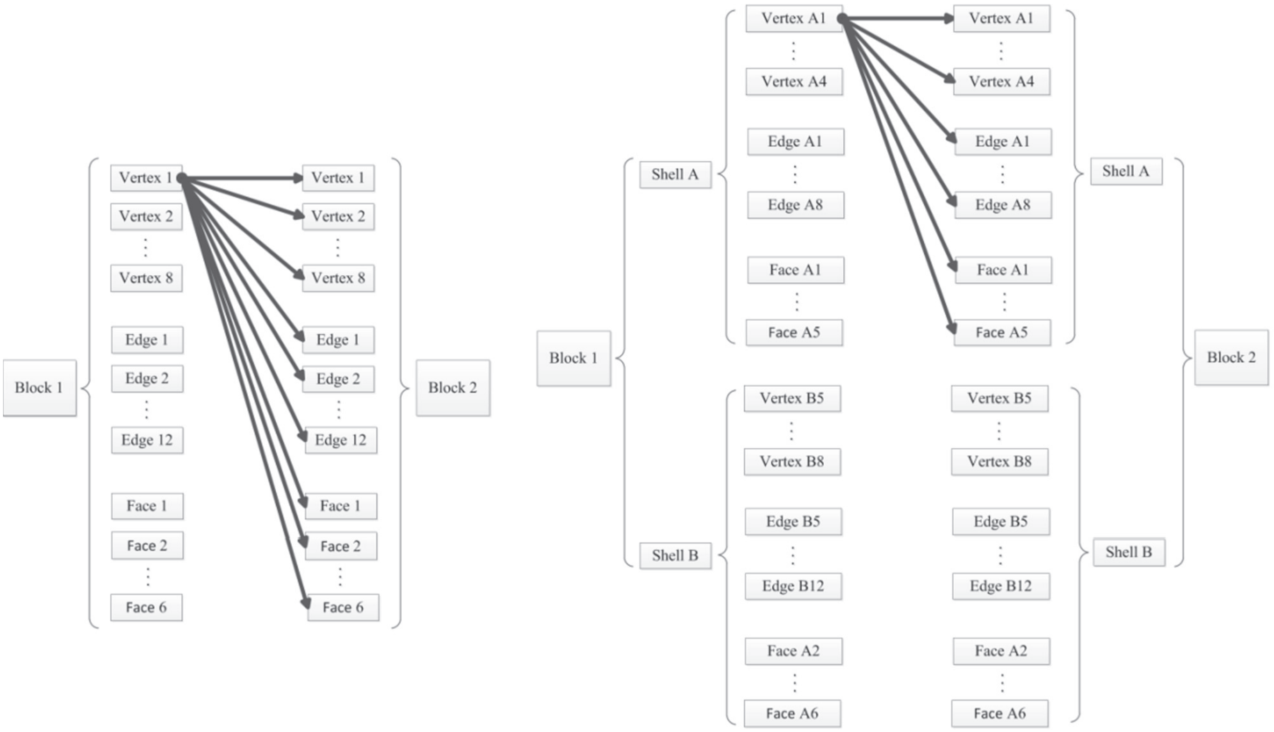
Though a shell could insert several shells of another block at the same time, the quantity of distance calculation loops in contact pattern identification of MSC is reduced notably than that of NBS or DESS. Neighbor searching process requires lower computational cost than distance judging loops in contact pattern identification process. Loops in neighbor searching only include relatively simple calculation and judgment processes. But loops in contact pattern identification are more complicated and consist of type selection, projection transform and distance judgment. In MSC algorithm, loops of neighbor searching increase while loops of distance judging in contact pattern identification decrease relative to DESS and NBS. Thus, the total computing cost for contact detection is reduced in this algorithm.

A geometric sub-unit could be involved in different groups at the same time. When a shell named Shell A inserts more than one shell at the same time, only nearest one of these shells is considered. For convex blocks, the contacting geometric element is one of the common elements or elements belong to the closest shell. If geometric element e8 is a common element of shell A and B, the two shells insert Shell C of another block at the same step. Loop between e8 and the geometric elements belonging to Shell C would only perform once to avoid unnecessary repetition. Because when a shell is not in the middle of a block, it only intersects with one or two shells of another block. This converts a full range search for geometric sub-units into a matching process with limited range (see Table 2).


(a) Two contacting blocks with single searching shells

(b) Two contacting blocks with multi-shell systems

Fig. 7. Two contacting blocks in 3D.



(a) Schematic diagram of NBS

(b) Schematic diagram of MSC

Fig. 8. Search loops of geometric elements between two contacting blocks.

Table 2
Quantity of geometric elements of the three shells.

Shells	Quantity of sub-units	Vertexes	Edges	Faces
Shell A	17	4	8	5
Shell B	8	0	4	4
Shell C	17	4	8	5

Though a shell could insert several shells of another block at the same time, quantity of distance judging loops in contact pattern identification of MSC is reduced notably than that in NBS or DESS. Neighbor searching process needs lower computational cost than distance judging loops in contact pattern identification process. Loops in neighbor searching include relatively simple calculation and judgment processes. However, loops in contact pattern identification are more complicated and consist of type selection, projection transform and distance judgment. In MSC algorithm, loops of neighbor searching increase while loops of distance judging in contact pattern identification decrease relative to DESS and NBS. Thus, the total computing cost for contact detection is reduced in this algorithm.

4.3. Contact state computation

After contact types and position are detected, the penalty method is utilized to calculate the potential energy generated by the contact springs. In this stage, vertex to face contact and edge to edge contact are different in generating contact matrixes. Jiang [6] and Wu [15] developed two algorithms to calculate vertex to face contact. Yeung [7] and Wu [22] gave the methods for edge to edge contact. Different geometric equations are given in those algorithms. In this work, a series of unified formulas for vertex to face contact and edge to edge contact are presented to give an efficient and terse process.

As shown in Fig. 9(a), $P_2P_3P_4P_5P_6$ is a face of a block and P_1 is a possible invasion vertex of a neighboring block. The normal distance between P_1 and the face $P_2P_3P_4P_5P_6$, d_n, is given by

$$d_n = \left|\overrightarrow{P_2P_1}\right| \cos\left\langle \overrightarrow{P_2P_3}\times\overrightarrow{P_2P_4}, \overrightarrow{P_2P_1}\right\rangle = \frac{\left(\overrightarrow{P_2P_3}\times\overrightarrow{P_2P_4}\right)\cdot\overrightarrow{P_2P_1}}{\left|\overrightarrow{P_2P_3}\times\overrightarrow{P_2P_4}\right|} \qquad (10)$$

In Fig. 9(b), P_1P_2 and P_3P_4 are two inserting edges belong to different blocks respectively and P_5P_6 is the perpendicular line of these edges. The normal distance between the two edges, d_n, can be expressed as

$$d_n = \frac{\left(\overrightarrow{P_1P_2}\times\overrightarrow{P_3P_4}\right)\cdot\overrightarrow{P_1P_3}}{\left|\overrightarrow{P_1P_2}\times\overrightarrow{P_3P_4}\right|} \qquad (11)$$

d_n is defined as follows

$$d_n = -\frac{\varDelta}{S_m} \qquad (12)$$

where

$$\varDelta = \begin{vmatrix} 1 & x_1+u_1 & y_1+v_1 & z_1+w_1 \\ 1 & x_2+u_2 & y_2+v_2 & z_2+w_2 \\ 1 & x_3+u_3 & y_3+v_3 & z_3+w_3 \\ 1 & x_4+u_4 & y_4+v_4 & z_4+w_4 \end{vmatrix}$$

$$S_m = \left|\overrightarrow{P_2P_3}\times\overrightarrow{P_2P_4}\right| \text{ or } \left|\overrightarrow{P_1P_2}\times\overrightarrow{P_3P_4}\right|$$

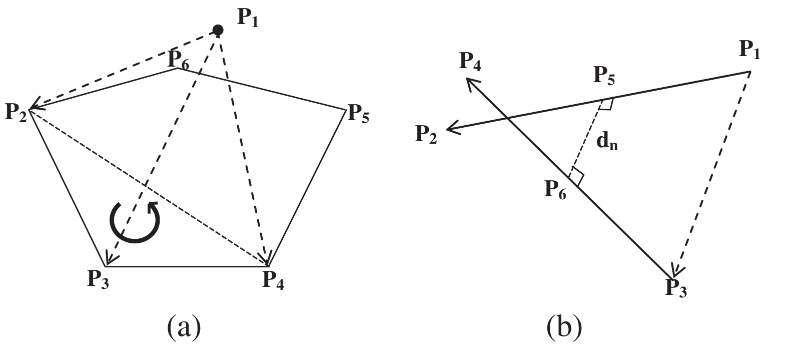


Fig. 9. Two basic contact models: (a) vertex to face contact and (b) edge to edge contact (not parallel).

$$\varDelta \approx \begin{vmatrix} 1 & x_1 & y_1 & z_1 \\ 1 & x_2 & y_2 & z_2 \\ 1 & x_3 & y_3 & z_3 \\ 1 & x_4 & y_4 & z_4 \end{vmatrix} S + \begin{vmatrix} 1 & u_1 & y_1 & z_1 \\ 1 & u_2 & y_2 & z_2 \\ 1 & u_3 & y_3 & z_3 \\ 1 & u_4 & y_4 & z_4 \end{vmatrix} + \begin{vmatrix} 1 & x_1 & v_1 & z_1 \\ 1 & x_2 & v_2 & z_2 \\ 1 & x_3 & v_3 & z_3 \\ 1 & x_4 & v_4 & z_4 \end{vmatrix} + \begin{vmatrix} 1 & x_1 & y_1 & w_1 \\ 1 & x_2 & y_2 & w_2 \\ 1 & x_3 & y_3 & w_3 \\ 1 & x_4 & y_4 & w_4 \end{vmatrix} \tag{13}$$

Then, set

$$Volu = \begin{vmatrix} 1 & x_1 & y_1 & z_1 \\ 1 & x_2 & y_2 & z_2 \\ 1 & x_3 & y_3 & z_3 \\ 1 & x_4 & y_4 & z_4 \end{vmatrix} \tag{14}$$

If the displacements of each block in a time step are small, the last term of D involving higher order terms of displacement increment can be ignored. Consequently, $\varDelta$ can be re-written as

$$\begin{aligned} \varDelta \approx\ & Volu + [(z_4 - z_1)(y_2 - y_3) + (z_2 - z_1)(y_3 - y_4) \\ & +(z_3 - z_1)(y_4 - y_2)]u_1 \\ & +[(z_4 - z_1)(x_3 - x_2) + (z_2 - z_1)(x_4 - x_3) + (z_3 - z_1)(x_2 - x_4)]v_1 \\ & +[(y_3 - y_1)(x_4 - x_2) + (y_4 - y_1)(x_2 - x_3) + (y_2 - y_1)(x_3 - x_4)]w_1 \\ & +[(y_3 - y_1)(z_4 - z_1) - (z_3 - z_1)(y_4 - y_1)]u_2 \\ & +[(z_3 - z_1)(x_4 - x_1) - (x_3 - x_1)(z_4 - z_1)]v_2 \\ & +[(x_3 - x_1)(y_4 - y_1) - (x_4 - x_1)(y_3 - y_1)]w_2 \\ & +[(y_4 - y_1)(z_2 - z_1) - (z_4 - z_1)(y_2 - y_1)]u_3 \\ & +[(z_4 - z_1)(x_2 - x_1) - (x_4 - x_1)(z_2 - z_1)]v_3 \\ & +[(x_4 - x_1)(y_2 - y_1) - (x_2 - x_1)(y_4 - y_1)]w_3 \\ & +[(y_2 - y_1)(z_3 - z_1) - (z_2 - z_1)(y_3 - y_1)]u_4 \\ & +[(z_2 - z_1)(x_3 - x_1) - (x_2 - x_1)(z_3 - z_1)]v_4 \\ & +[(x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1)]w_4 \end{aligned} \tag{15}$$

For vertex to face contact shown in Fig. 9(a), according to Eq. (2), the displacements of points P_1 at Block i and P_2, P_3, and P_4 at Block j can be expressed in the following form

$$\begin{pmatrix} u_1 \\ v_1 \\ w_1 \end{pmatrix} = [T_i(x_1, y_1, z_1)][D_i]$$

$$\begin{pmatrix} u_2 \\ v_2 \\ w_2 \end{pmatrix} = [T_j(x_2, y_2, z_2)][D_j]$$

$$\begin{pmatrix} u_3 \\ v_3 \\ w_3 \end{pmatrix} = [T_j(x_3, y_3, z_3)][D_j]$$

$$\begin{pmatrix} u_4 \\ v_4 \\ w_4 \end{pmatrix} = [T_j(x_4, y_4, z_4)][D_j]$$

In contrast, for edge to edge contact in Fig. 9(b), the displacements of points P_1, P_2 at Block i and P_3, P_4 at Block j can be expressed as follows

$$\begin{pmatrix} u_1 \\ v_1 \\ w_1 \end{pmatrix} = [T_i(x_1, y_1, z_1)][D_i]$$

$$\begin{pmatrix} u_2 \\ v_2 \\ w_2 \end{pmatrix} = [T_i(x_2, y_2, z_2)][D_i]$$

$$\begin{pmatrix} u_3 \\ v_3 \\ w_3 \end{pmatrix} = [T_j(x_3, y_3, z_3)][D_j]$$

$$\begin{pmatrix} u_4 \\ v_4 \\ w_4 \end{pmatrix} = [T_j(x_4, y_4, z_4)][D_j]$$

Therefore, the normal distance, d_n, can be expressed as:

$$d_n = \frac{S_0}{l} + \{E_i\}\{D_i\} + \{E_j\}\{D_j\} \tag{16}$$

where

$$\{E_i\} = (\, e_{i1} \quad e_{i2} \quad e_{i3} \quad e_{i4} \quad e_{i5} \quad e_{i6} \quad e_{i7} \quad e_{i8} \quad e_{i9} \quad e_{i10} \quad e_{i11} \quad e_{i12} \,)$$

$$\{E_j\} = (\, e_{j1} \quad e_{j2} \quad e_{j3} \quad e_{j4} \quad e_{j5} \quad e_{j6} \quad e_{j7} \quad e_{j8} \quad e_{j9} \quad e_{j10} \quad e_{j11} \quad e_{j12} \,)$$

For vertex to face contact,

$$\begin{aligned}
e_{ir} = {} & [(z_4-z_1)(y_2-y_3)+(z_2-z_1)(y_3-y_4)+(z_3-z_1)(y_4-y_2)]t_{1r}(x_1,y_1,z_1)/S_m \\
& +[(z_4-z_1)(x_3-x_2)+(z_2-z_1)(x_4-x_3)+(z_3-z_1)(x_2-x_4)]t_{2r}(x_1,y_1,z_1)/S_m \\
& +[(y_3-y_1)(x_4-x_2)+(y_4-y_1)(x_2-x_3)+(y_2-y_1)(x_3-x_4)]t_{3r}(x_1,y_1,z_1)/S_m \\
e_{jr} = {} & [(y_3-y_1)(z_4-z_1)-(z_3-z_1)(y_4-y_1)]t_{1r}(x_2,y_2,z_2)/S_m \\
& +[(z_3-z_1)(x_4-x_1)-(x_3-x_1)(z_4-z_1)]t_{2r}(x_2,y_2,z_2)/S_m \\
& +[(x_3-x_1)(y_4-y_1)-(x_4-x_1)(y_3-y_1)]t_{3r}(x_2,y_2,z_2)/S_m \\
& +[(y_4-y_1)(z_2-z_1)-(z_4-z_1)(y_2-y_1)]t_{1r}(x_3,y_3,z_3)/S_m \\
& +[(z_4-z_1)(x_2-x_1)-(x_4-x_1)(z_2-z_1)]t_{2r}(x_3,y_3,z_3)/S_m \\
& +[(x_4-x_1)(y_2-y_1)-(x_2-x_1)(y_4-y_1)]t_{3r}(x_3,y_3,z_3)/S_m \\
& +[(y_2-y_1)(z_3-z_1)-(z_2-z_1)(y_3-y_1)]t_{1r}(x_4,y_4,z_4)/S_m \\
& +[(z_2-z_1)(x_3-x_1)-(x_2-x_1)(z_3-z_1)]t_{2r}(x_4,y_4,z_4)/S_m \\
& +[(x_2-x_1)(y_3-y_1)-(x_3-x_1)(y_2-y_1)]t_{3r}(x_4,y_4,z_4)/S_m
\end{aligned}$$

For edge to edge contacts,

$$\begin{aligned}
e_{ir} = {} & [(z_4-z_1)(y_2-y_3)+(z_2-z_1)(y_3-y_4)+(z_3-z_1)(y_4-y_2)]t_{1r}(x_1,y_1,z_1)/S_m \\
& +[(z_4-z_1)(x_3-x_2)+(z_2-z_1)(x_4-x_3)+(z_3-z_1)(x_2-x_4)]t_{2r}(x_1,y_1,z_1)/S_m \\
& +[(y_3-y_1)(x_4-x_2)+(y_4-y_1)(x_2-x_3)+(y_2-y_1)(x_3-x_4)]t_{3r}(x_1,y_1,z_1)/S_m \\
& +[(y_3-y_1)(z_4-z_1)-(z_3-z_1)(y_4-y_1)]t_{1r}(x_2,y_2,z_2)/S_m \\
& +[(z_3-z_1)(x_4-x_1)-(x_3-x_1)(z_4-z_1)]t_{2r}(x_2,y_2,z_2)/S_m \\
& +[(x_3-x_1)(y_4-y_1)-(x_4-x_1)(y_3-y_1)]t_{3r}(x_2,y_2,z_2)/S_m \\
e_{jr} = {} & [(y_4-y_1)(z_2-z_1)-(z_4-z_1)(y_2-y_1)]t_{1r}(x_3,y_3,z_3)/S_m \\
& +[(z_4-z_1)(x_2-x_1)-(x_4-x_1)(z_2-z_1)]t_{2r}(x_3,y_3,z_3)/S_m \\
& +[(x_4-x_1)(y_2-y_1)-(x_2-x_1)(y_4-y_1)]t_{3r}(x_3,y_3,z_3)/S_m \\
& +[(y_2-y_1)(z_3-z_1)-(z_2-z_1)(y_3-y_1)]t_{1r}(x_4,y_4,z_4)/S_m \\
& +[(z_2-z_1)(x_3-x_1)-(x_2-x_1)(z_3-z_1)]t_{2r}(x_4,y_4,z_4)/S_m \\
& +[(x_2-x_1)(y_3-y_1)-(x_3-x_1)(y_2-y_1)]t_{3r}(x_4,y_4,z_4)/S_m
\end{aligned}$$

This potential energy contribution from the normal contact spring is calculated as follows

$$\Pi_n = \frac{1}{2}k_n(d_n)^2. \tag{17}$$

By minimizing the potential energy of normal contact, the following matrices can then be added to the sub-matrices $[K_{ii}]$, $[K_{ij}]$, $[K_{ji}]$ and $[K_{jj}]$ in the global stiffness matrix of Eq. (4)

$$k_n\{E_i\}\{E_i\} \to [K_{ii}] \tag{18}$$

$$k_n\{E_i\}\{E_j\} \to [K_{ij}] \tag{19}$$

$$k_n\{E_j\}\{E_i\} \to [K_{ji}] \tag{20}$$

$$k_n\{E_j\}\{E_j\} \to [K_{jj}]. \tag{21}$$

The vectors $\{F_i\}$ and $\{F_j\}$ are calculated as follows, which are then added to the global force vector

$$-k_n\frac{Volu}{S_m}\{E_i\} \to \{F_i\} \tag{22}$$

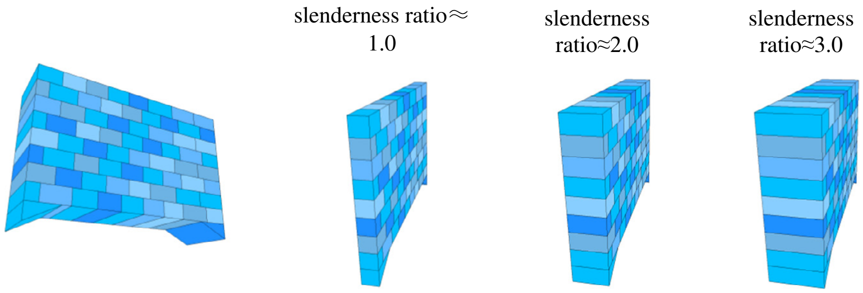
$$-k_n\frac{Volu}{S_m}\{E_j\} \to \{F_j\}. \tag{23}$$

5. Test examples

Three 3D examples are presented in the following sections to compare the efficiencies of single sphere shell, DESS and the MSC algorithm. The first example considers slenderness ratio as a design variable. The angles between main axes of blocks and coordinate planes are set in the second test. The third example is a comprehensive test to assess the performance of the three algorithms in a system containing over 200 blocks.

5.1. Hexahedrons drop example

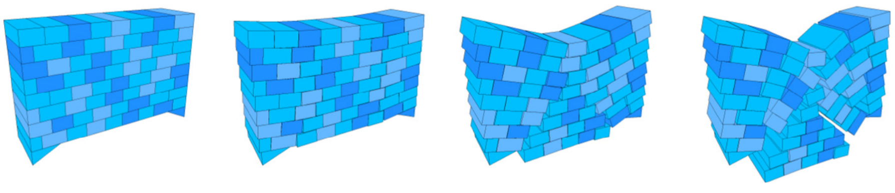
The first example involves a system of 70 blocks as shown in Fig. 10(a). The density, Young's modulus and Poisson's ratio values for each block are 2.6×10^3 kg/m^3, 4 MPa and 0.25, respectively. The maximum time step is 0.001 s and the maximum allowable displacement at each step is 0.01 m. Each of the two pentahedra at the bottom is fixed by four nodes while the hexahedra are free. Three cases are different in long axis length of these blocks and the lengths are 20 m, 40 m and 60 m respectively as shown in



(a) a system containing 70 blocks

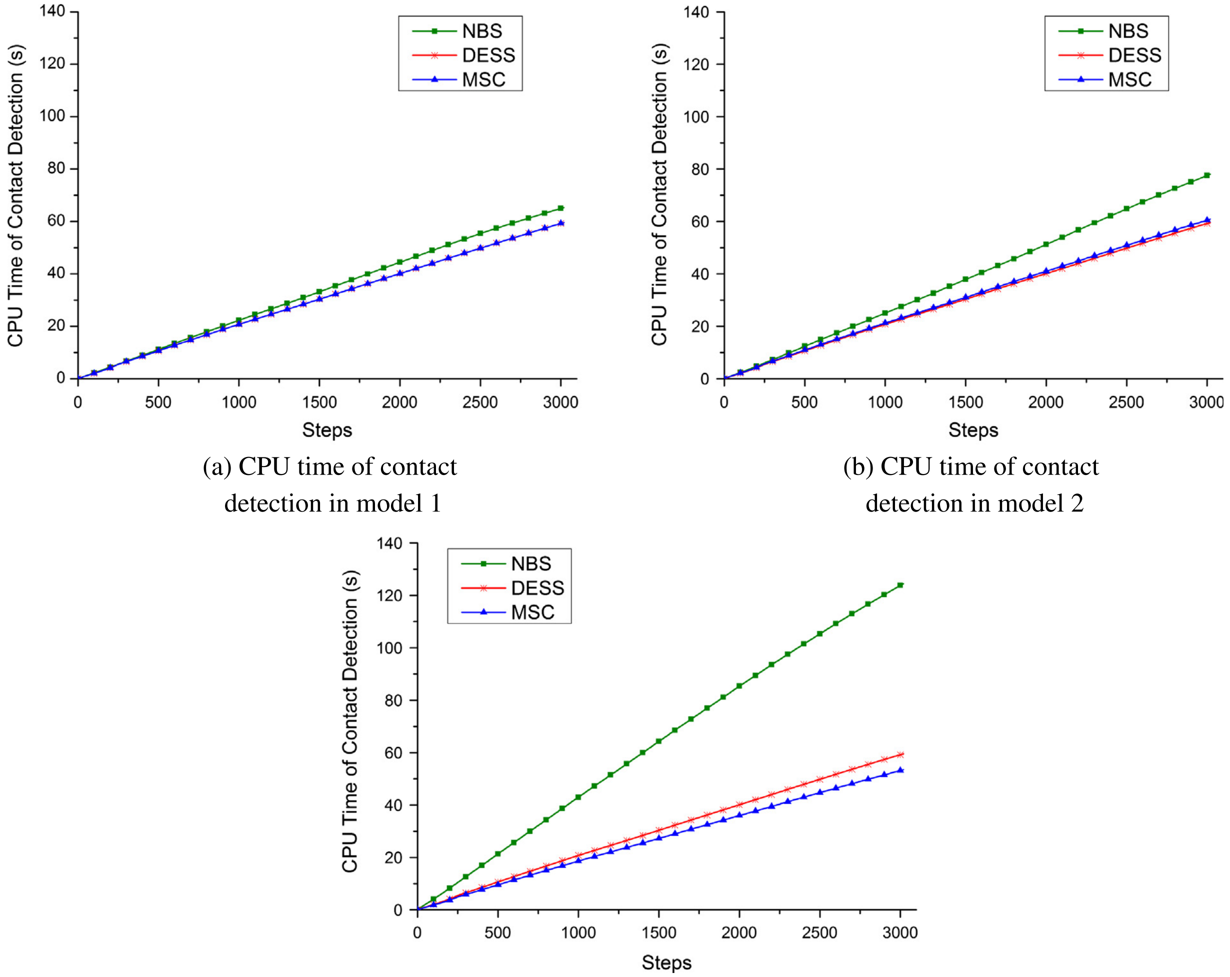
(b) Three models with different slenderness ratios

Fig. 10. Hexahedrons drop test.



(a) time step=0 (b) time step=1000 (c) time step=2000 (d) time step=3000

Fig. 11. DDA result of 60m case.



(a) CPU time of contact detection in model 1

(b) CPU time of contact detection in model 2

(c) CPU time of contact detection in model 3

Fig. 12. The contact detection time increments of the three cases.

Fig. 10(b). Accordingly, quantity of shells for each block described in Fig. 10(b) is 1–3 respectively.

The results of the test example are shown in Fig. 11. The CPU time increments of NBS, DESS and MSC with respect to calculation steps in the three models are compared respectively in Fig. 12.

NBS takes more computational costs than DESS and MSC with the increase of edge length. A searching shell in NBS expands as slenderness ratio increases, so it inserts more shells from different blocks concurrently. But only a number of these blocks contact the original block actually. Thus, contact detection time grows dramatically as slenderness ratio increasing in NBS. In all models, DESS algorithms preform almost the same efficiency. A rectangular searching shell grows only in one dimension not affecting quantity of bocks detected in DESS. Consequently contact detection frequency and time for DESS almost remains the same in three models.

All detection shells in three different types are almost in similar sizes close to those of blocks in model 1, but sphere shells are still in larger size than the other two types of shells. Thus, NBS in model 1 take more CPU time than DESS and MSC. In model 2, MSC algorithm spends more calculation time than DESS for MSC needs more search loops between blocks. When MSC only generates one or two shells for each block, it is the same with DESS in neighbor searching and contact pattern identification. Loops between geometric sub-units are less than those of NBS and DESS when three shells are generated for each block. This is due to one shell not in the middle of a block only inserts one or two shells of another block. This converts a full range search for geometric sub-units into a matching process with limited range. Thus the case using two shells perform less efficiency than the case using three shells.

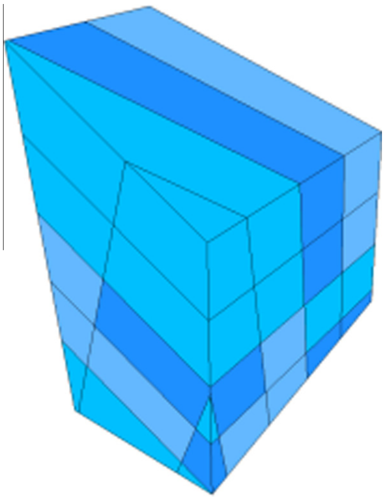


Fig. 13. Model of blocks slide test.

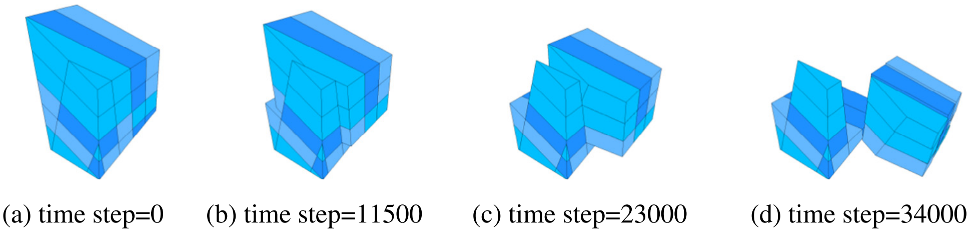


Fig. 14. DDA results of test 2.

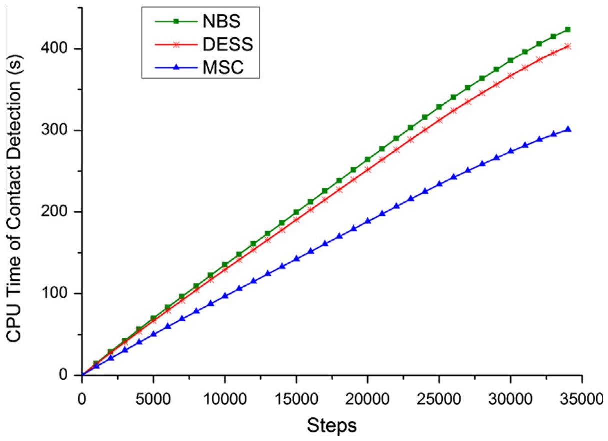


Fig. 15. The contact detection time increments of test 2.

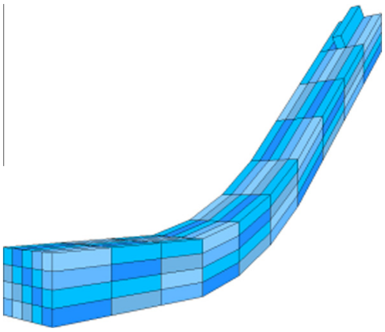


Fig. 16. Model of single block slide test.

5.2. Blocks slide example

A test with 23 blocks was shown in Fig. 13. The density, Young's modulus and Poisson's ratio values for each block are 2.6 × 10^3 kg/m^3, 10 MPa and 0.25, respectively. The maximum time step is 0.001 s and the maximum allowable displacement at each step is 0.01 m. Eleven blocks at the bottom and a side are fixed. The other 9 block could slide down without initial velocity and friction. The angles between main axes of blocks and positive *x*, *y*, *z* coordinate axis are 50°, 135° and 120°, respectively. 3 blocks with one shell, 8 blocks with two shells and 12 blocks with three shells are applied in MSC.

Searching shells of NBS and DESS is larger than those of MSC in this test.

Fig. 14 describes results of contact detection at several time steps this test. The CPU time increment curves of NBS, DESS and MSC by steps are plotted in Fig. 15. Boundaries of contact searching shell in DESS are parallel to one of the coordinate planes, unnecessary detection space occurs when the principal axis of a block form an angle to coordinate planes. However, MSC algorithm reduces the total volume of contact shells compared with NBS and DESS. MSC algorithm could lessen false detections in this model. Thus, in this test, MSC performs better efficiency than the other two algorithms. After step 27,000, the sliding nine blocks are moving out of detection range gradually and time increment per step decreases.

5.3. Single block slide example

A test containing 201 blocks shown in Fig. 16 was analyzed to assess the efficiency of the new algorithm. 200 blocks were fixed to form an approximate curved surface and another block slide

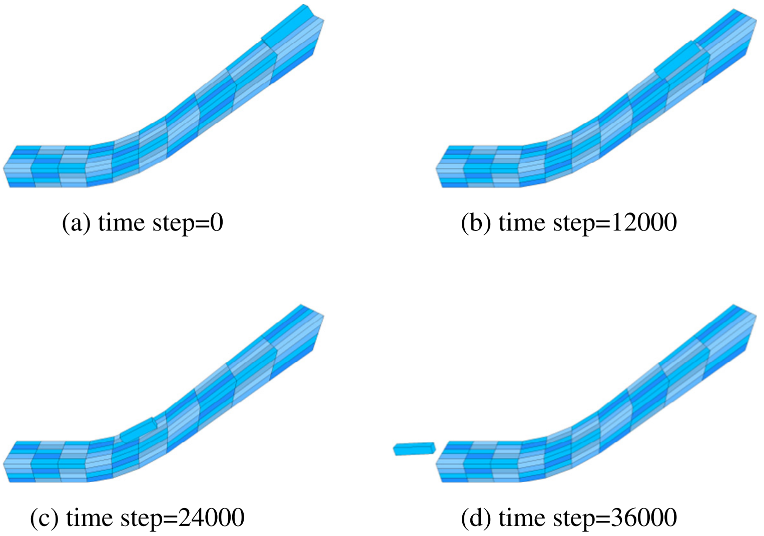


Fig. 17. DDA results of test 3.

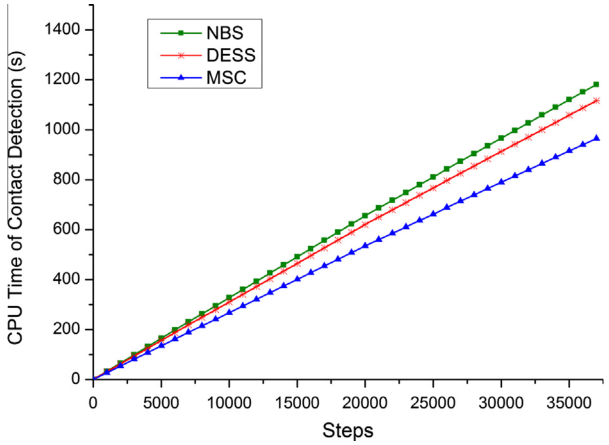


Fig. 18. The contact detection time increments of test three.

down on it. Each one of the fixed 200 blocks has a length of 10 m along X axis and its sectional area is 1×1 m. The density, Young's modulus and Poisson's ratio values for each block are 2.8×10^3 kg/m^3, 4 MPa and 0.25, respectively. The maximum time step is 0.001 s and the maximum allowable displacement at each step is 0.01 m. Contact searching processes between the 200 fixed blocks were skipped forcibly. However, contact detection could also occur between the fixed blocks. All blocks of this model are covered by three shells in MSC.

Fig. 17 shows the results of the 3D DDA calculation. The contact detection time increments of test 3 are shown in Fig. 18. Computation cost of MSC is lower than that of NBS or DESS. Total time increment for each step in MSC and NBS does not change greatly in the sliding process.

6. Conclusions

In this study, neighbor searching is coupled with contact pattern identification process in a smaller scale by dividing a block to several parts with a multi-shell system (MSC) in 3D DDA simulation. In the present MSC algorithm, a block is covered by the multi-shell system and each shell in the system is linked to only a part of the geometric sub-units (vertices, edges and faces). When a shell in a block penetrates another sphere of another block, contact pattern identification processes only occurs between the part of vertices, edges and faces which are linked to these spheres. This MSC algorithm comes with two advantages comparing to the previous methods. Firstly, the MSC method reduces detection shell volume of an elongated block, therefore detected blocks which are close but not contacted with this block practically decrease. Less candidate blocks means less candidate geometric sub-units for contact distance calculation and identification. Secondly, the loops between geometric sub-units of two contacting blocks are also reduced by the grouping which reduces the range of candidate geometric sub-units. Therefore, the quantity of contact mode identification loops in MSC algorithm is much less than those in the NBS and DESS. The results of 3D examples show that the MSC method is more efficient than the other two methods in most cases. In a multi-block system, the MSC algorithm shows improved computational efficiency for slant blocks. For block shape close to spheres, the MSC algorithm is the same as the NBS and has almost the same in terms of the computational cost with the DESS.

Acknowledgements

The authors gratefully acknowledge the supports from the Key Programme from Natural Science Foundation of China (41130751), National Basic Research Program of China (973:2011CB013800), Research Program for Development of Western China Communication (2011ZB04) and funding by SRF for ROCS, SEM, and Shanghai Chengguang Program (12CG20). The first author is supported by China Scholarship Council (CSC) during his study in UWA.

References

[1] G.H. Shi, Discontinuous Deformation Analysis – A New Numerical Model for the Statics and Dynamics of Block System. Ph.D. Thesis, Dept. of Civil Engineering, University of California, Berkeley, 1988.
[2] G.H. Shi, R.E. Goodman, Generalization of two-dimensional discontinuous deformation analysis for forward modeling, Int. J. Numer. Anal. Methods Geomech. 13 (1989) 359–380.
[3] G.H. Shi, Three-dimensional discontinuous deformation analysis, in: N. Bicenic (Ed.), Proceedings of the Forth International Conference on Analysis of Discontinuous Deformation (ICADD-4), Glasgow, Scotland, June 6–8, 2001, pp. 1–21.
[4] G.H. Shi, Three-dimensional discontinuous deformation analysis, in: Ellsworth, et al. (Ed.), Proceedings of the 38th US Rock Mechanics Symposium, Washington, DC, July 7–10, 2001, pp. 1421–1428.
[5] G.H. Shi, Theory and examples of three dimensional discontinuous deformation analyses, in: S. Wang, B. Fu, Z. Li (Eds.), Frontiers of Rock Mechanics and Sustainable Development in the 21st Century, Balkema, Lisse, 2001, pp. 27–32.
[6] Q.H. Jiang, M.R. Yeung, A model of point-to-face contact for three-dimensional discontinuous deformation analysis, Rock Mech. Rock Eng. 37 (2004) 95–116.
[7] M.R. Yeung, Q.H. Jiang, N. Sun, A model of edge-to-edge contact for three-dimensional discontinuous deformation analysis, Comput. Geotech. 34 (2007) 175–186.
[8] L.F. Greengard, The Rapid Evaluation of Potential Fields in Particle Systems, ACM Distinguished Dissertation Series, MIT Press, 1988.
[9] J.R. Williams, R. O'Connor, A linear complexity intersection algorithm for discrete element simulation of arbitrary geometries, Eng. Comput. 12 (1995) 185–201.
[10] A. Munjiza, K.R.F. Andrews, NBS contact detection algorithm for bodies of similar size, Int. J. Numer. Methods Eng. 43 (1998) 131–149.
[11] E. Perkins, J.R. Williams, A fast contact detection algorithm insensitive to object sizes, Eng. Comput. 18 (2001) 48–61.
[12] H.H. Zhu, X.Y. Zhuang, Y.C. Cai, High rock slope stability analysis using the enriched meshless shepard and least squares method, Int. J. Comput. Methods 8 (2011) 209–228.
[13] J.H. Wu, C.H. Juang, H.M. Lin, Vertex-to-face contact searching algorithm for three-dimensional frictionless contact problems, Int. J. Numer. Methods Eng. 63 (2005) 876–897.
[14] P.A. Cundall, Formulation of a three-dimensional distinct element model – Part I: A scheme to detect and represent contacts in a system composed of many polyhedral blocks, Int. J. Rock Mech. Min. Sci. Geomech. 25 (1988) 107–116.
[15] J.H. Wu, Numerical Analysis of Discontinuous Rock Masses Using Discontinuous Deformation Analysis. PhD dissertation, School of Civil Engineering, Kyoto University, Japan, 2003.
[16] S.A.R. Beyabanaki, R.G. Mikola, K. Hatami, Three-dimensional discontinuous deformation analysis (3-D DDA) using a new contact resolution algorithm, Comput. Geotech. 35 (2008) 346–356.
[17] C.J. Ong, E.G. Gilbert, Fast versions of the Gilbert–Johnson–Keerthi distance algorithm: additional results and comparisons, IEEE Trans. Rob. Automat. 17 (2001) 531–539.
[18] B.V. Mirtich, Impulse Based Dynamic Simulation of Rigid Body Systems, PhD thesis, Department of Electrical Engineering and Computer Science, University of California, Berkeley, 1996.
[19] E. Nezami, Y.M.A. Hashash, D. Zhao, J. Ghaboussi, A fast contact detection algorithm for 3-D discrete element method, Comput. Geotech. 31 (2004) 575–587.
[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 Meth Geomech 30 (8) (2006) 783–801.
[21] T.Y. Ahn, J.J. Song, New contact-definition algorithm using inscribed spheres for 3D discontinuous deformation analysis, Int. J. Comput. Methods 08 (2011) 171.
[22] J.H. Wu, New edge-to-edge contact calculating algorithm in three-dimensional discrete numerical analysis, Adv. Eng. Softw. 39 (2008) 15–24.
[23] Q. Jiang, D. Feng, An study on an angle-face model of three dimensional discontinuous deformation analysis, Chinese J. Rock Mech. Eng. 19 (2000) 930–935 (in simplified Chinese).
[24] J. Liu, X. Kong, Three dimensional continuous and discontinuous deformation analysis, ACTA Mech. Sinica 34 (2002) 941–948 (in simplified Chinese).
[25] A.R. Keneti, A. Jafari, J.H. Wu, A new algorithm to identify contact patterns between convex blocks for three-dimensional discontinuous deformation analysis, Comput. Geotech. 35 (2008) 746–759.
[26] J.H. Wu, Numerical Analysis of Discontinuous Rock Masses Using Discontinuous Deformation Analysis. PhD dissertation, School of Civil Engineering, Kyoto University, Japan, 2003.
[27] S.A.R. Beyabanaki, R.G. Mikola, K. Hatami, Three-dimensional discontinuous deformation analysis (3-D DDA) using a new contact resolution algorithm, Comput. Geotech. 35 (2008) 346–356.

[28] X. Zhuang, C.E. Augarde, K.M. Mathisen, Fracture modeling using meshless methods and level sets in 3D: framework and modeling, Int. J. Numer. Methods Eng. 92 (2012) 969–998.

[29] X. Zhuang, C. Augarde, S. Bordas, Accurate fracture modelling using meshless methods, the visibility criterion and level sets: Formulation and 2D modelling, Int. J. Numer. Methods Eng. 86 (2011) 249–268.

[30] T. Rabczuk, P.M.A. Areias, A new approach for modelling slip lines in geological materials with cohesive models, Int. J. Numer. Anal. Methods Geomech. 30 (2006) 1159–1172.

[31] T. Rabczuk, G. Zi, S. Bordas, N.X. Hung, A simple and robust three-dimensional cracking-particle method without enrichment, Comput. Methods Appl. Mech. Eng. 199 (2010) 2437–2455.

[32] T. Belytschko, Y.Y. Lu, L. Gu, Element-free Galerkin methods, Int. J. Numer. Methods Eng. 37 (1994) 229–256.

[33] S. Ghorashi, S. Mohammadi, S. Sabbagh-Yazdi, Orthotropic enriched element free Galerkin method for fracture analysis of composites, Eng. Fract. Mech. 78 (2011) 1906–1927.

[34] X. Zhuang, H. Zhu, C. Augarde, An improved meshless Shepard and least square method possessing the delta property and requiring no singular weight function, Comput. Mech. 53 (2014) 343–357.

[35] X. Zhuang, Y. Cai, C. Augarde, A meshless sub-region radial point interpolation method for accurate calculation of crack tip fields, Theor. Appl. Fract. Mech. 69 (2014) 118–125.

[36] H. Nguyen-Vinh, I. Bakar, M.A. Msekh, J.-H. Song, J. Muthu, G. Zi, P. Le, S. Bordas, R. Simpson, S. Natararajan, T. Lahmer, T. Rabczuk, Extended finite element method for dynamic fracture of piezo-electric materials, Eng. Fract. Mech. 92 (2012) 19–31.

[37] L. Chen, T. Rabczuk, S. Bordas, G.R. Liu, K.Y. Zeng, P. Kerfriden, Extended finite element method with edge-based strain smoothing (Esm-XFEM) for linear elastic crack growth, Comput. Methods Appl. Mech. Eng. 209212 (4) (2012) 250–265.

[38] Y. Cai, H. Zhu, X. Zhuang, A continuous/discontinuous deformation analysis (CDDA) method based on deformable blocks for fracture modelling, Front. Struct. Civ. Eng. 7 (2014) 369–378.

[39] Y. Jia, Y. Zhang, G. Xu, X. Zhuang*, T. Rabczuk, Reproducing kernel triangular B-spline-based FEM for solving PDEs, Comput. Methods Appl. Mech. Eng. 267 (2013) 342–358.

[40] N. Nguyen-Thanh, H. Nguyen-Xuan, S. Bordas, T. Rabczuk, Isogeometric analysis using polynomial splines over hierarchical *T*-meshes for two-dimensional elastic solids, Comput. Methods Appl. Mech. Eng. 200 (21–22) (2011) 1892–1908.

[41] N. Nguyen-Thanh, J. Muthu, X. Zhuang, T. Rabczuk, An adaptive three-dimensional RHT-spline formulation in linear elasto-statics and elasto-dynamics, Comput. Mech. 53 (2) (2014) 369–385.

[42] C.H. Thai, A.J.M. Ferreira, S. Bordas, T. Rabczuk, H. Nguyen-Xuan, Isogeometric analysis of laminated composite and sandwich plates using a new inverse trigonometric shear deformation theory, Eur. J. Mech – A/Solids 43 (2014) 89–108 (2013).

[43] N. Valizadeh, S. Natarajan, O.A. Gonzalez-Estrada, T. Rabczuk, Bui Tinh Quoc, S.P.A. Bordas, NURBS-based finite element analysis of functionally graded plates: static bending, vibration, buckling and flutter, Compos. Struct. 99 (2013) 309–326.

[44] N. Nguyen-Thanh, J. Kiendl, H. Nguyen-Xuan, R. Wüchner, K.U. Bletzinger, Y. Bazilevs, T. Rabczuk, Rotation free isogeometric thin shell analysis using PHT-splines, Comput. Methods Appl. Mech. Eng. 200 (4748) (2011) 3410–3424.

[45] S. Ghorashi, N. Valizadeh, S. Mohammadi, Extended isogeometric analysis (XIGA) for simulation of stationary and propagating cracks, Int. J. Numer. Methods Eng. 89 (2012) 1069–1101.