A fast direct search algorithm for contact detection of convex polygonal or polyhedral particles

Fei Zheng[a,b,c], Yu-Yong Jiao[d,*], Michael Gardner[c], Nicholas Sitar[c]

[a] State Key Laboratory of Geomechanics and Geotechnical Engineering, Institute of Rock and Soil Mechanics, Chinese Academy of Sciences, Xiaohongshan, Wuchang District, Wuhan 430071, PR China [b] University of Chinese Academy of Sciences, 19 A Yuquan Rd, Shijingshan District, Beijing 100049, PR China [c] Department of Civil and Environmental Engineering, University of California, Berkeley, CA 94720-1710, United States [d] Faculty of Engineering, China University of Geosciences, Wuhan 430074, PR China

* Corresponding author. E-mail address: yyjiao@cug.edu.cn (Y.-Y. Jiao).

Abstract

A fast direct search (FDS) algorithm is presented to increase the efficiency of contact detection for convex polygonal and polyhedral particles. All contact types are detected using only a small subset of these contact types: vertex-to-edge for polygons while vertex-to-face and edge-to-edge for polyhedra. First, an initial contact list is generated. Then in subsequent steps the contact list is updated by checking only local boundaries of the blocks and their separation. An exclusion algorithm is applied to avoid unnecessary examination for particles that are near but not-in-contact. The benchmark tests show that the FDS produces significant speed-up in various cases.

Keywords: Contact detection, Convex particles, Valid entrance, Initial detection, Updated detection

1. Introduction

Contact analysis is a necessary part of any computational method dealing with interaction between independent particles, such as in the discrete element method (DEM) and discontinuous deformation analysis (DDA). Contact detection comprises a significant portion of the computational effort in these types of analyses and therefore an accurate and efficient contact detection algorithm is essential. The aim of the contact detection phase is to provide possible contact types, contact points, contact normal directions and contact modes for all potential contact elements. Focusing on contacts between two particles, the particle shape becomes an important factor affecting computation accuracy and efficiency. In rock engineering, rock masses typically consist of densely packed polyhedral blocks and contact detection between polyhedral blocks is more complicated than between spheres, which are common in other applications. Moreover, the efficiency of contact detection between convex polyhedra is much higher than that for concave polyhedra, owing to the utilization of convex properties in the detection algorithm. This paper focuses on improving the efficiency and accuracy of contact detection for convex polyhedra.

From a geometrical perspective, convex polygons and convex polyhedrons have distinct properties which can be used in geometry representation and contact detection. For the geometric representation, two main methods are used. First, the point set of a twodimensional (2-D) convex polygon or a three-dimensional (3-D) convex polyhedron can be algebraically represented by the intersection of several inequalities. The linear programming method [1] uses these expressions as a basis. Second, a boundaryrepresentative method for polygons or polyhedra can be used. Boundaries of a polygon consist of vertices and edges while boundaries of a polyhedron consist of vertices, edges and faces.

Several convex polyhedron properties can be used in contact detection. First, a virtual infinite, rigid plane, named the common plane [2], can be found to separate two polygons or two polyhedra. This common plane can be considered as a reference plane to quickly compute the contact point and contact normal direction. Second, if a point is inside a polyhedron (polygon), this point must be inside all the half-spaces that form the polyhedron (polygon). As a result, if a point is outside any half-space forming the polyhedron (polygon), it is outside the polyhedron (polygon). This property is quite useful in being able to exclude vertices that are not in contact. Third, all 2-D planar angles, 3-D solid angles and dihedrals forming boundaries of convex polyhedrons are convex. Fourth, contacts of two convex polygons fall into the following types: vertex-to-vertex (v-v), vertex-to-edge (v-e) and parallel edge-toedge (parallel e-e), whereas contacts of two convex polyhedra consist of: vertex-to-vertex (v-v), vertex-to-edge (v-e), vertex-to-face (v-f), crossing edge-to-edge (crossing e-e), parallel edge-to-edge (parallel e-e), edge-to-face (e-f) and face-to-face (f-f).

Numerous contact detection algorithms have been presented to improve the efficiency of detecting contacts between convex polyhedra. One approach is the common plane algorithm [2]. The common plane simplifies the contact detection process by testing the relationship of the common plane and each block instead of the relationship of two blocks. This approach was further optimized to produce the fast common plane (FCP) algorithm [3] and the shortest link method (SLM) [4]. Another genre is referred to as the direct search or improved direct search algorithms [8–10]. Other methods include the approaching face method [5], an improved approaching face algorithm [6], vertex-to-face searching algorithm [7] and multi-shell cover method [11]. Boon et al. [1] generalize contact between convex polyhedral particles as a convex optimization problem and used a linear program method to solve it.

Most recently, Shi [12] proved that contact between two convex polyhedra can be simplified to contact of a reference point and an entrance convex polyhedron. The boundary of the entrance polyhedron consists of contact covers (v-v, v-e, v-f and crossing e-e), and the position relationship of the reference point and contact covers determines the contact point, contact normal direction and contact mode. In this paper, we build on the theoretical work of Shi [12] and present the fast direct search (FDS) algorithm to improve the efficiency and accuracy of the contact detection. The FDS algorithm is applied in DDA to do discontinuous computation, two basic types (v-v and v-e) are used in 2-D case and four basic contact types (v-v, v-e, v-f and crossing e-e) are used in 3-D case for contact force computation.

2. The fast direct search algorithm (FDS)

The contact detection procedure consists of two search phases: neighbor search and delicate search. The neighbor search algorithm is based on the cell mapping method [2], but can be optimized using other neighbor search algorithms - such as NBS [13], DESS [14] or CGrid [15] - depending on the application. The fast direct search (FDS) algorithm is specifically concerned with the delicate search during which contact points, normals and modes are established.

FDS aims to obtain contact geometry information for convex polygonal or polyhedral particles more efficiently. In order to obtain higher efficiency, the FDS algorithm is executed in two phases. First, initial contact detection is executed and then updated contact detection is initiated when necessary. The initial contact detection phase forms a list of contact types for a new block geometry configuration, while the updated contact detection phase updates the contact information in this list starting from the contact detection results in the previous iteration.

Herein the basic concepts used in FDS are first introduced and then a detailed discussion of the contact detection process, data structures used and associated computational cost is presented.

2.1. Basic concepts used in contact detection

2.1.1. Valid entrance concept

Valid entrance is defined as a physical no-overlap status of potential contact pairs, similar to the concept of first entrance developed by Shi [12,16]. Valid entrance means that two potential contact objects should not have an overlapping area if their potential contact points are superimposed. In this way, a valid entrance check is used as a criterion in judging basic contact pairs. No overlap between two contact objects, from another perspective, is the same as the concept of a continuous face that separates these two objects. Here for convex objects, a plane separating the two objects can be found if they do not overlap.

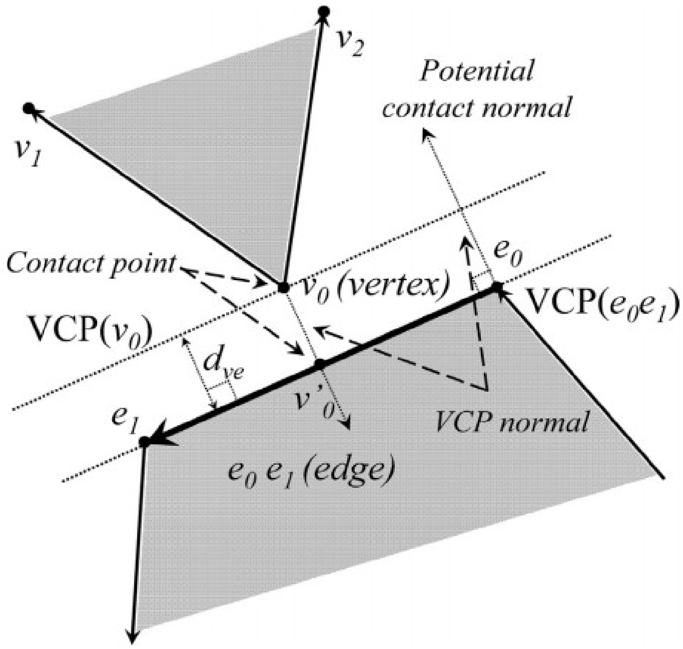

Fig. 1. Entrance of vertex-to-edge and its VCPs.

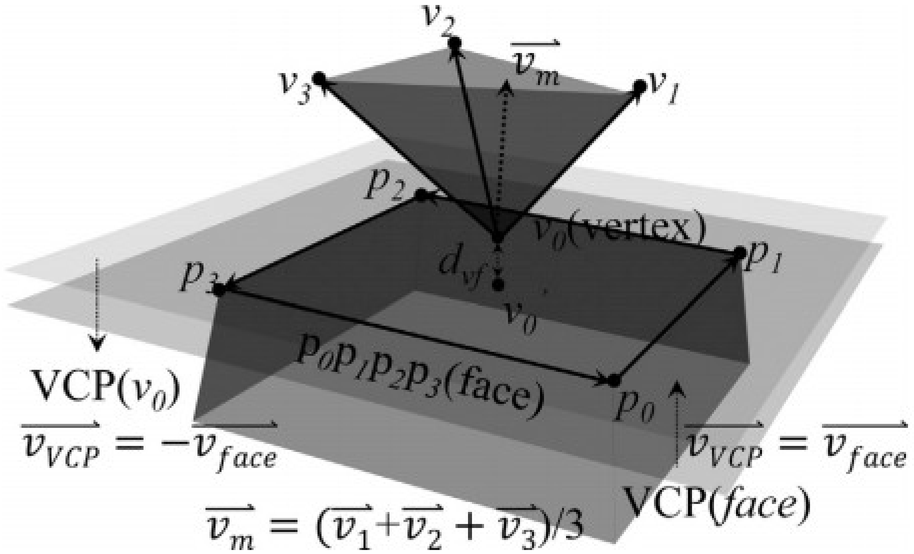
v_2
$\overrightarrow{v_m}$
v_3
v_1
p_2
v_0(vertex)
p_1
p_3
d_{vf}
v_0
$p_0p_1p_2p_3$(face)
VCP(v_0)
$\overrightarrow{v_{VCP}} = -\overrightarrow{v_{face}}$
p_0
$\overrightarrow{v_{VCP}} = \overrightarrow{v_{face}}$
VCP($face$)
$\overrightarrow{v_m} = (\overrightarrow{v_1}+\overrightarrow{v_2}+\overrightarrow{v_3})/3$

Fig. 2. Entrance of vertex-to-face and its VCPs.

A 2-D vertex-to-edge, v-e, entrance is valid if both edge vectors of the vertex angle point out of the half-space formed by the edge. As shown in Fig. 1, vertex v_0 and edge e_0e_1 is a valid entrance pair because both edge vector v_0v_1 and v_0v_2 point out of half-space surrounded by edge e_0e_1. In 2-D vertex-to-vertex, $v(i)$-$v(j)$, entrance checking, all edge $e(i)$ and $e(j)$ joint to vertex $v(i)$ and $v(j)$ are considered. $V(i)$-$v(j)$ entrance checking is subdivided into $v(i)$-$e(j)$ and $e(i)$-$v(j)$ entrance checking. If any $v(i)$-$e(j)$ or $e(i)$-$v(j)$ entrance is valid, *v*(*i*)-*v*(*j*) entrance is valid. Then half-space outer normals of all valid *v*(*i*)-*e*(*j*) or *e*(*i*)-*v*(*j*) entrances are stored in a list of potential contact normals.

In 3-D, vertex-to-face, v-f, or crossing edge-to-edge, e-e, entrance is valid if an infinite plane can be found that separates the vertex angle and the half-space surrounded by the face, or separates the two dihedrals connecting the two edges. For v-f entrance, the face can serve as the separating plane, and two steps are used in checking validity of v-f entrance. First, the average value of edge vectors that join the vertex angle is obtained and used to roughly judge if the vertex angle overlaps the half-space formed by the face. If the average vector points into the half-space, overlap take place and the v-f entrance is not valid; otherwise, the second step is conducted. All edge vectors that joint the vertex angle are used to check overlap towards the half-space. If all vectors point out of the half-space, this v-f entrance is valid and the face normal is a valid entrance plane normal; otherwise, the v-f entrance is invalid. For a crossing e-e entrance, the normal of a potential separating plane is obtained by the cross product of vectors along the two edges. Then two edge vectors along the edge dihedral faces are obtained for each edge. These vectors are used to judge if the two edges are separated by the contact plane, as shown in Fig. 3. If they are, then this e-e pair is a valid entrance and the potential contact plane normal is a valid entrance plane normal; otherwise, it is invalid. In 3-D *v*(*i*)-*v*(*j*) or *v*(*i*)-*e*(*j*) entrance checking, all edges and faces *e*(*i*), *e*(*j*), *f*(*i*) and *f*(*j*), that joint the vertex angles, and the edge dihedrals are considered. *V*(*i*)-*v*(*j*) or *v*(*i*)-*e*(*j*) entrance checking is subdivided into *v*(*i*)-*f*(*j*), *f*(*i*)-*v*(*j*) and *e*(*i*)-*e*(*j*) entrance checking. If any of these subset entrance is valid, *v*(*i*)-*v*(*j*) entrance is valid. Then half-space outside normals of all valid *v*(*i*)-*f*(*j*), *f*(*i*)-*v*(*j*) and *e*(*i*)-*e*(*j*) entrance are stored in a list of potential contact normals.

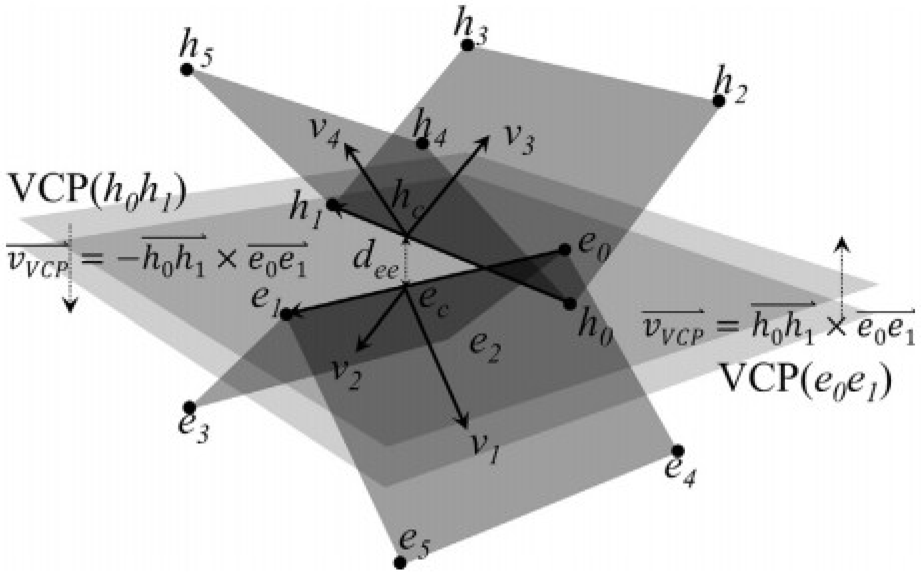
h_5
h_3
h_2
v_4
h_4
v_3
$\mathrm{VCP}(h_0h_1)$
h_1
h_c
$\overrightarrow{v_{VCP}} = -\overrightarrow{h_0h_1} \times \overrightarrow{e_0e_1}$
d_{ee}
e_0
e_1
e_c
h_0
$\overrightarrow{v_{VCP}} = \overrightarrow{h_0h_1} \times \overrightarrow{e_0e_1}$
e_2
v_2
$\mathrm{VCP}(e_0e_1)$
e_3
v_1
e_4
e_5

Fig. 3. Entrance of crossing edge-to-edge and its VCPs in 3D polyhedra.

2.1.2. Contact patterns

Contacts of 2-D polygons include v-v, v-e and parallel e-e types; however, all these contact types can be described using only the ve contact type: parallel e-e contacts can be regarded as a combination of v-v and v-e contacts. So, for contact detection of 2-D polygons, only v-e pairs are directly examined. Then according to the number of valid v-e contact pairs, the final contact type (v-v and v-e) can be deduced.

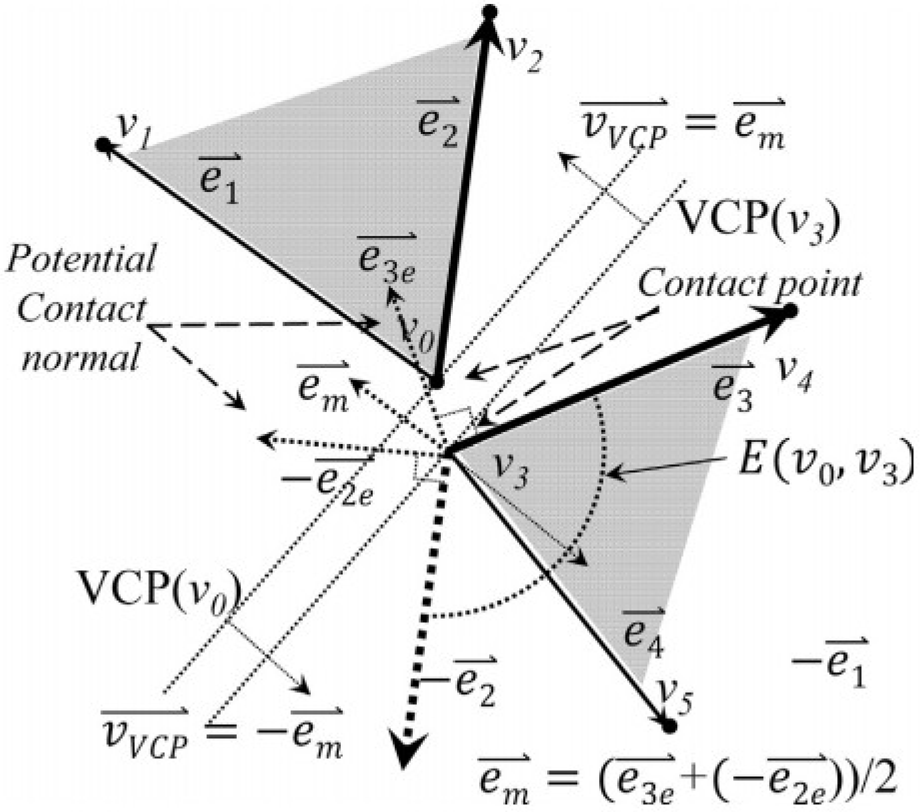

Fig. 4. Entrance of vertex-to-vertex and its VCPs in 2D polygons.

In 3-D, possible contact types are v-v, v-e, v-f, parallel e-e, crossing e-e, e-f and f-f. Parallel e-e contacts can be regarded as a combination of v-v and v-e contacts. E-f and f-f types can be regarded as combinations of four basic contact types (v-v, v-e, v-f and crossing e-e types). As noted by Cundall [2], the simplest contact detection routine for convex polyhedra is to search for v-f and e-e contact. Also, as already noted, the contact between two convex polyhedra can be simplified into a contact of a reference point and an entrance polyhedron [12], while boundary surfaces of the entrance polyhedron consist of contact covers formed by v-f and crossing e-e contacts. The spatial relationship between the reference point and boundary of the entrance polyhedron will define the contact geometry information. Among the four basic types, v-v and v-e contacts can be specially recognized in the v-f and crossing e-e detection process. Recognizing this, the FDS is able determine the basic contact types (v-v, v-e, v-f and crossing e-e) while only needing to directly detect v-f and crossing e-e contact pairs.

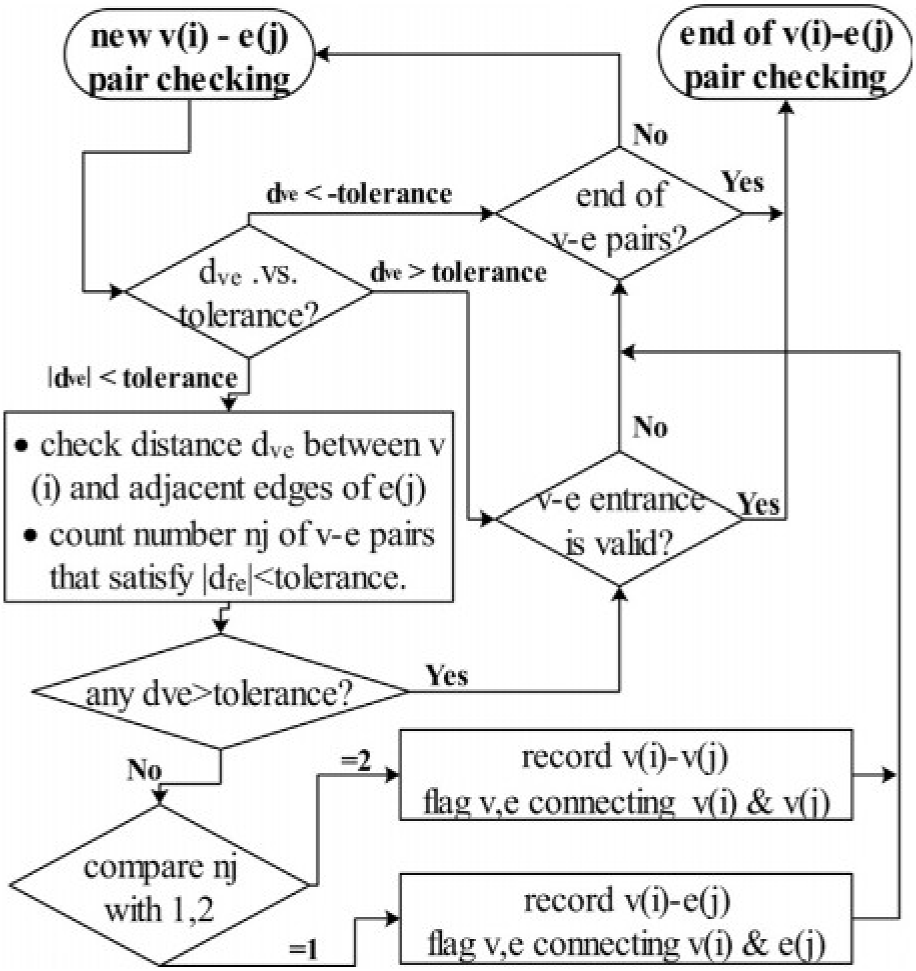
new v(i) - e(j) pair checking
end of v(i)-e(j) pair checking
No
Yes
end of v-e pairs?
dve < -tolerance
dve .vs. tolerance?
dve > tolerance
|dve| < tolerance
• check distance dve between v (i) and adjacent edges of e(j)
• count number nj of v-e pairs that satisfy |dfe|<tolerance.
No
v-e entrance is valid?
Yes
any dve>tolerance?
Yes
No
=2
record v(i)-v(j)
flag v,e connecting v(i) & v(j)
compare nj with 1,2
=1
record v(i)-e(j)
flag v,e connecting v(i) & e(j)

Fig. 5. Initial v-e checking procedure for convex polygons.

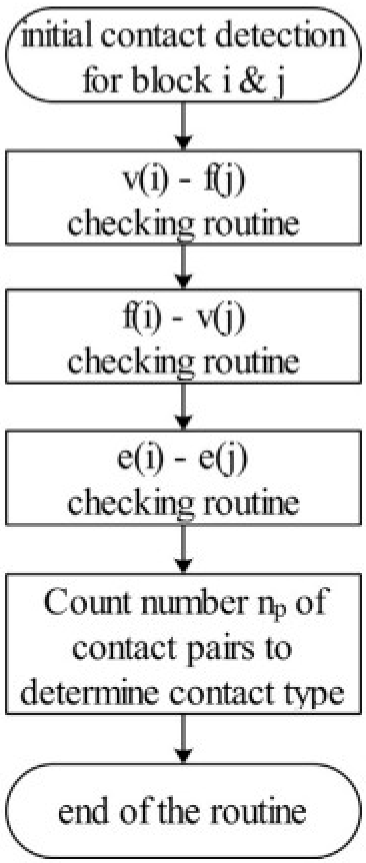
initial contact detection for block i & j
v(i) - f(j) checking routine
f(i) - v(j) checking routine
e(i) - e(j) checking routine
Count number n_p of contact pairs to determine contact type
end of the routine

Fig. 6. Initial contact checking procedure for convex polyhedra.

2.1.3. Virtual Contact Plane (VCP)

Each contact defines a pair of virtual common planes, defined by a point and a normal. The two VCPs pass through respective contact points defined as follows: (1) the points that give the minimum distance between the two objects before they contact; (2) the points that give the shortest path to separate the penetrating objects when they contact. For example, the contact points of a v-f or v-e contact pair are the vertex and its orthogonal projection point on the face or the edge. The VCP normal of a contact pair is calculated using the valid entrance plane normals obtained during the valid entrance check. The normal points out of the polygon or polyhedron and, consequently, each VCP pair has opposite normal vectors, as shown in Figs. 1–3.

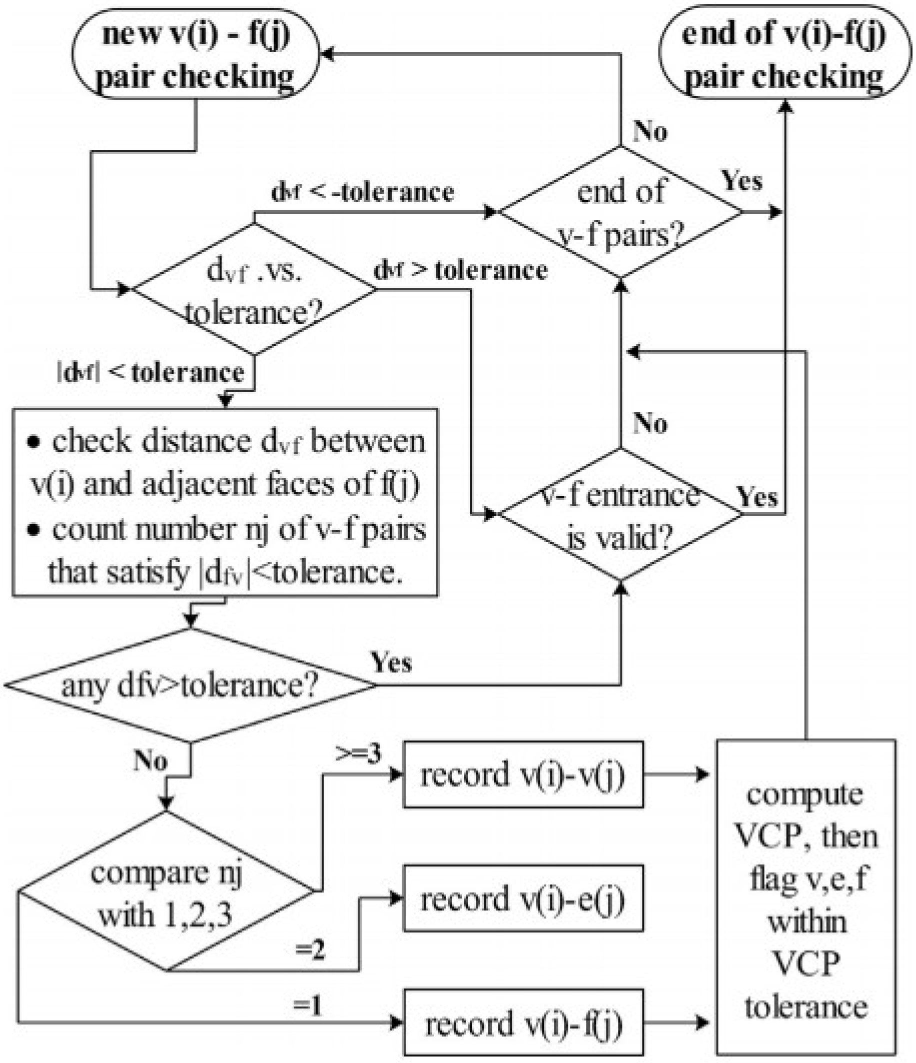
new v(i) - f(j) pair checking
end of v(i)-f(j) pair checking
No
end of v-f pairs?
Yes
dvf < -tolerance
dvf .vs. tolerance?
dvf > tolerance
|dvf| < tolerance
• check distance dvf between v(i) and adjacent faces of f(j)
• count number nj of v-f pairs that satisfy |dfv|<tolerance.
No
v-f entrance is valid?
Yes
any dfv>tolerance?
Yes
No
>=3
record v(i)-v(j)
compare nj with 1,2,3
record v(i)-e(j)
=2
=1
record v(i)-f(j)
compute VCP, then flag v,e,f within VCP tolerance

Fig. 7. Initial v-f checking procedure for convex polyhedra.

For a 2-D v-e contact pair, the VCPs are two lines passing the vertex and its orthogonal projection point onto the edge, respectively and they are parallel to the edge, as shown in Fig. 1. For a 2-D v-v contact pair, the VCPs pass through the two vertices. A list of potential VCP normals has been established in v-v valid entrance check. The average of the vectors in potential VCP normal list is selected as the VCP normal, as shown in Fig. 4. Similarly, for a 3- D contact pair including faces (v-f, e-f and f-f), the VCP normals are parallel to the outer normal of their unique valid entrance faces and point out of the blocks, as shown in Fig. 2. For 3-D crossing e-e contact pairs, VCP normals are parallel to the cross product of the two edge vectors, as shown in Fig. 3. The determination of the VCP normal for a v-v or v-e contact pairs is executed in two steps. First, a list of potential VCP normals is established in 3-D v-v or v-e entrance checking (Section 2.1.1). Second, the average of the vectors in the VCP normal list is obtained and serves as the

VCP normal for this contact pair. It should be mentioned that if the list of the potential VCP normals is empty, then the v-v or v-e pair is not a valid entrance pair.

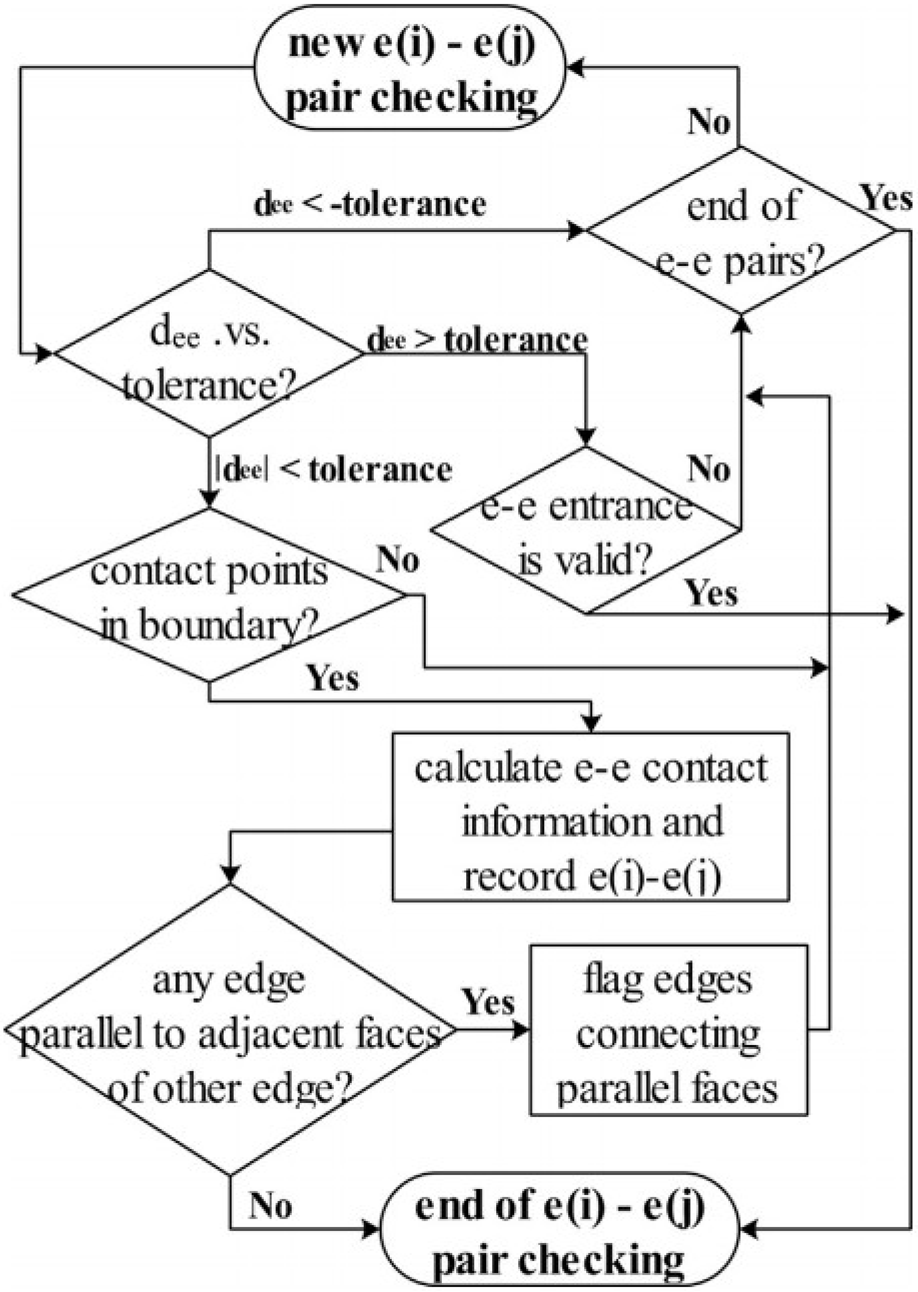
new e(i) - e(j) pair checking
No
end of e-e pairs?
Yes
d_{ee} < -tolerance
d_{ee} .vs. tolerance?
d_{ee} > tolerance
No
e-e entrance is valid?
Yes
$|d_{ee}|$ < tolerance
contact points in boundary?
No
Yes
calculate e-e contact information and record e(i)-e(j)
any edge parallel to adjacent faces of other edge?
Yes
flag edges connecting parallel faces
No
end of e(i) - e(j) pair checking

Fig. 8. Initial e-e checking procedure for convex polyhedra.

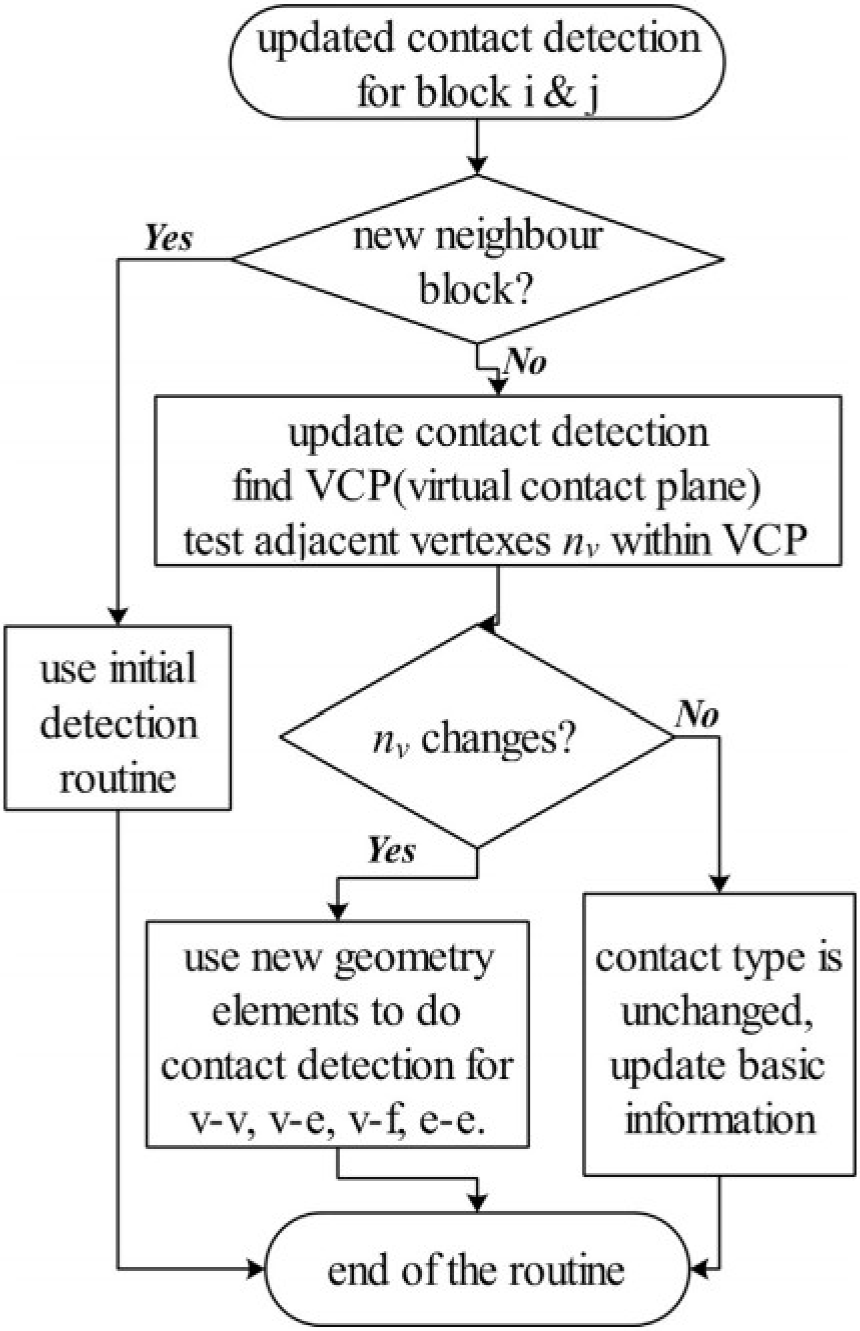
updated contact detection for block i & j
new neighbour block?
Yes
No
update contact detection
find VCP(virtual contact plane)
test adjacent vertexes n_v within VCP
use initial detection routine
n_v changes?
No
Yes
use new geometry elements to do contact detection for v-v, v-e, v-f, e-e.
contact type is unchanged, update basic information
end of the routine

Fig. 9. Updated contact checking procedure for convex polyhedra.

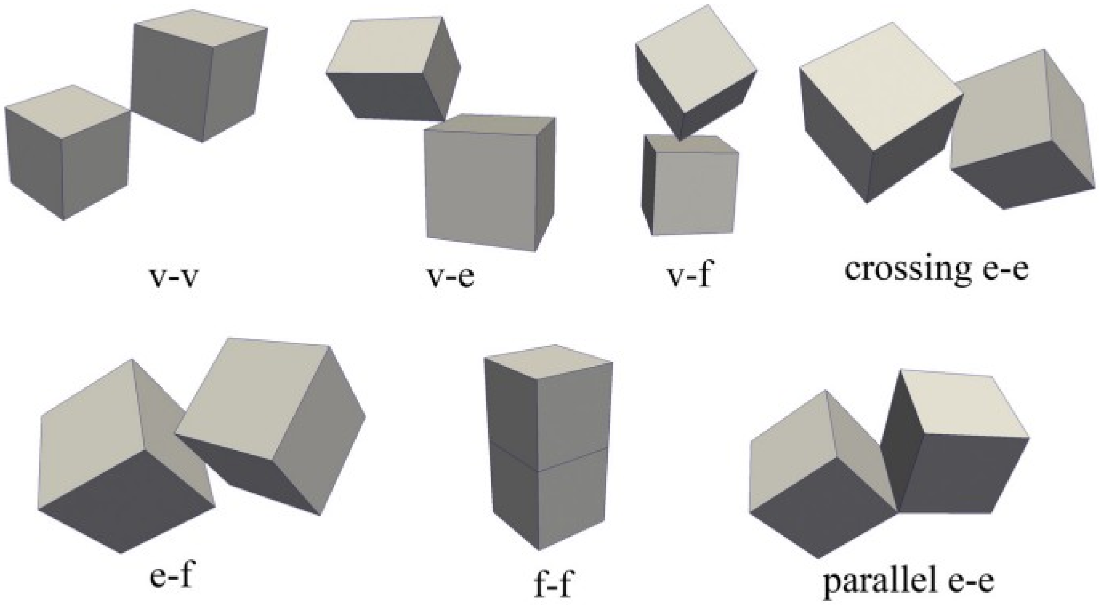

Fig. 10. Different contact types of two hexahedra.

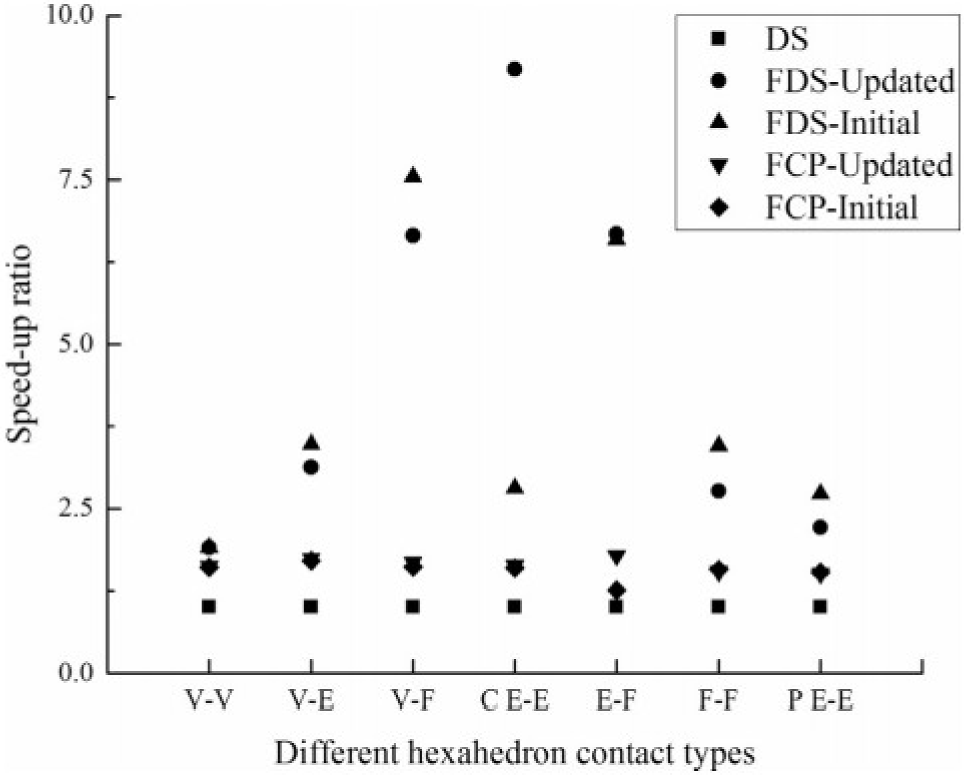

Fig. 11. Speed-up ratios of FDS and FCP for two hexahedra.

The VCP is applied in two aspects: (1) its normal can be used as the contact normal for its related contact pair; (2) vertices in its search region (defined in Section 2.1.4) are used to define the nearby vertices, edges or faces for neighbor block pairs in the updated contact detection phase.

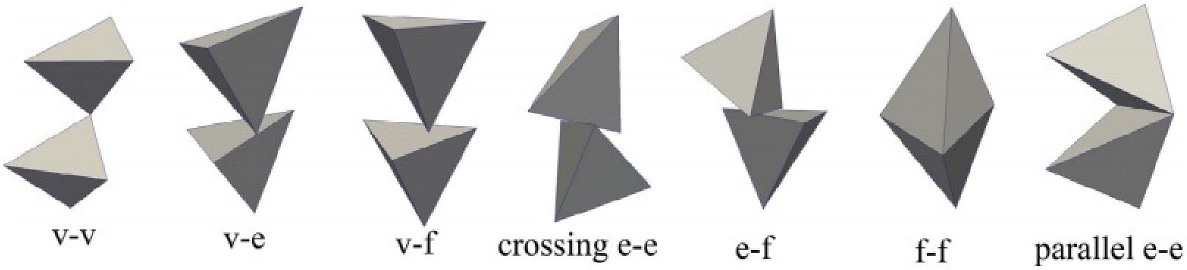

Fig. 12. Different contact types of two tetrahedra.

2.1.4. Exclusion algorithm and contact tolerance

During the neighbor search, some blocks may be regarded as possibly being in contact based on their bounding boxes or spheres. However, during the delicate search it is established that they are actually not in contact. To avoid unnecessary examination for these block pairs, an exclusion algorithm is applied in the FDS using contact tolerance criteria.

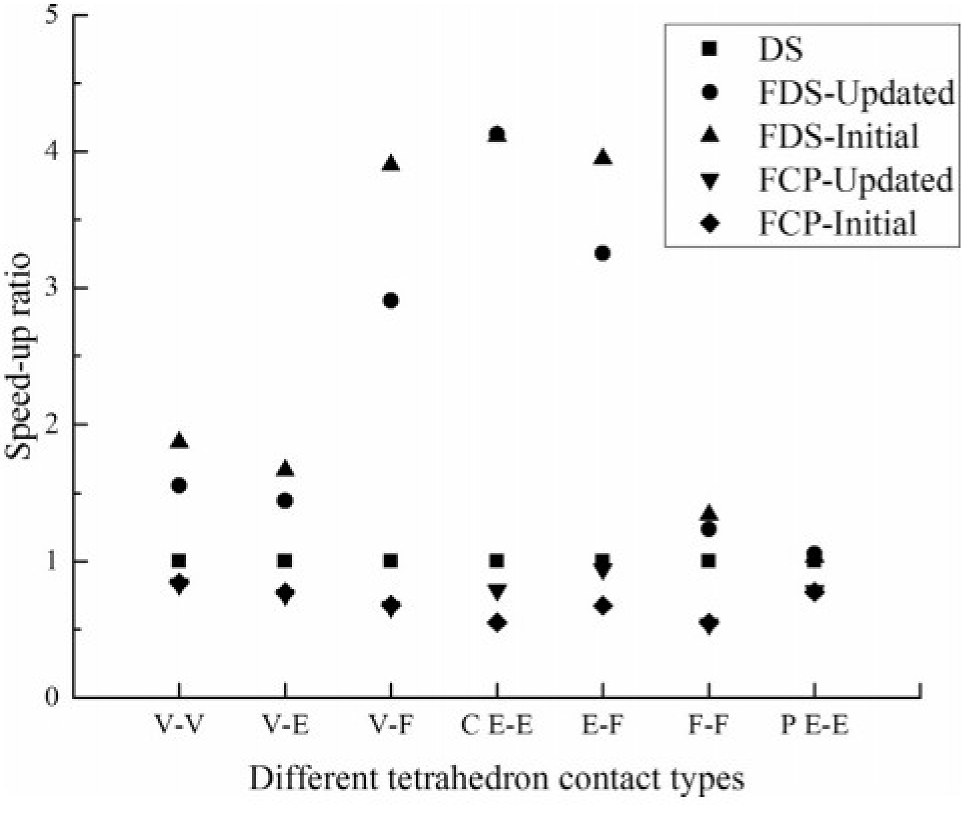

Fig. 13. Speed-up ratios of FDS and FCP for two tetrahedra.

Empirical tolerance criteria have to be defined for each contact type. In this work we define for a parallel e-e pair is one degree. The minimum distance tolerance d_{tol} to determine potential contact pairs is set as $d_{tol} = \max\{2 \cdot d_{max}, 0.05 \cdot d_{small}\}$, while d_{max} refers to the maximal block vertex displacement per time step; d_{small} refers to the smallest edge length of the block system. If the minimum distance of a checking pair is smaller than d_{tol}, this pair satisfies the distance judging criterion for potential contact pairs.

d_{tol} is also used as the orthogonal distance tolerance to determine if one vertex locates in search regions of a virtual contact plane (VCP) region. If the orthogonal distance d_{vp} between a vertex and a VCP is smaller than d_{tol}, the vertex is in VCP regions. So, the VCP searching region is the space formed by translating VCP from -d_{tol} to d_{tol} along its normal.

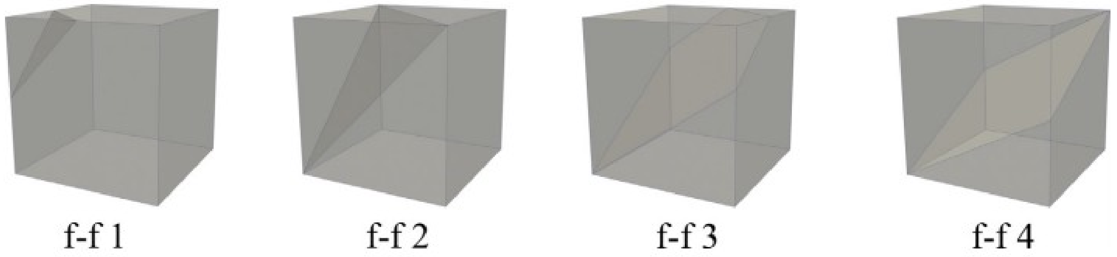

Fig. 14. Different f-f contact cases between two polyhedra generated by block cutting.

Thus, in the examination of v-e (2-D), v-f and crossing e-e (3-D) pairs, the orthogonal distance d_o of these pairs is obtained. If the orthogonal distance d_o of a checking pair is larger than d_{tol}, the pair may be too far apart to be in contact during the current time step. If their mutual entrance status is valid (no penetration) in addition to d_o being larger than d_{tol}, then the block pair is regarded as not-incontact during the current time step and the delicate search process for this block pair is skipped.

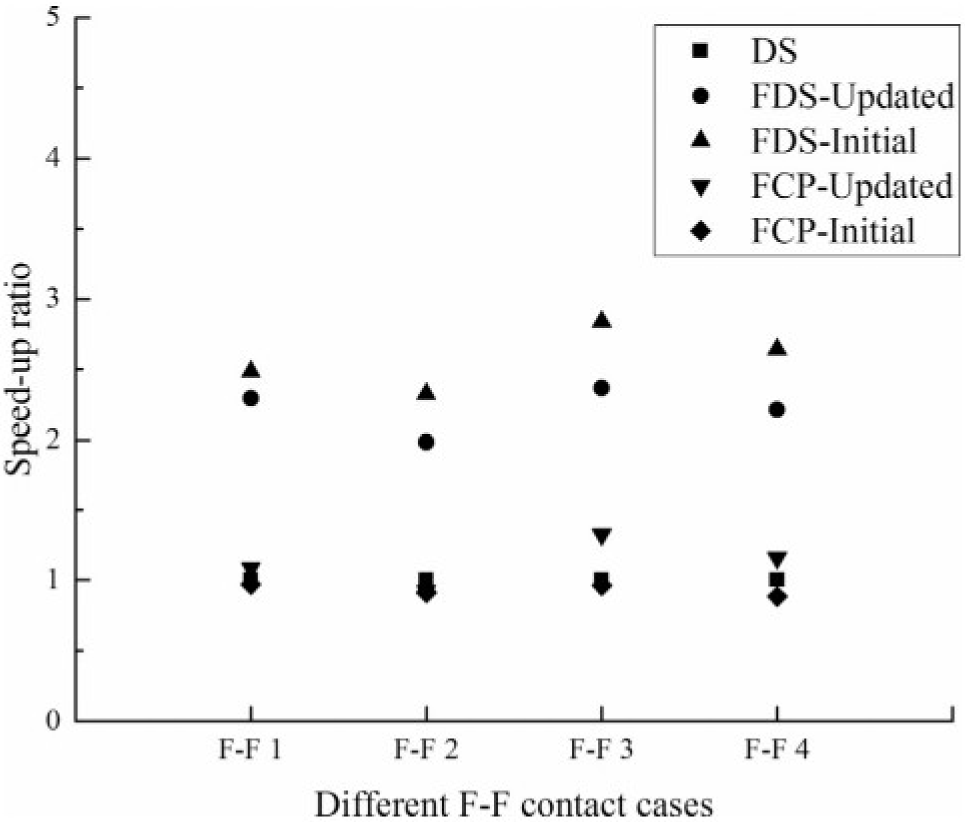

Fig. 15. Speed-up ratios of FDS and FCP for two polyhedra.

2.2. Initial contact detection

The initial contact detection considers the new geometry configuration. In 2-D, a thorough v-e search is executed in the initial detection of two convex polygons, as shown in Fig. 5. V(*i*)-e(*j*) check is first executed, in which all v(*i*)-v(*j*) and v(*i*)-e(*j*) types are established. Then e(*i*)-v(*j*) check follows and all e(*i*)-v(*j*) types are detected out.

In 3-D, v(*i*)-f(*j*), f(*i*)-v(*j*) and e(*i*)-e(*j*) contact pairs are checked one by one to find the unique contact type for the two convex polyhedra, as shown on the flow chart in Fig. 6. In the v(*i*)-f(*j*) check, all v(*i*)-v(*j*), v(*i*)-e(*j*) and v(*i*)-f(*j*) types are established. In subsequent f(*i*)-v(*j*) check, all e(*i*)-v(*j*) and f(*i*)-v(*j*) are contacts are detected and, finally, all e(*i*)-e(*j*) contacts are identified.

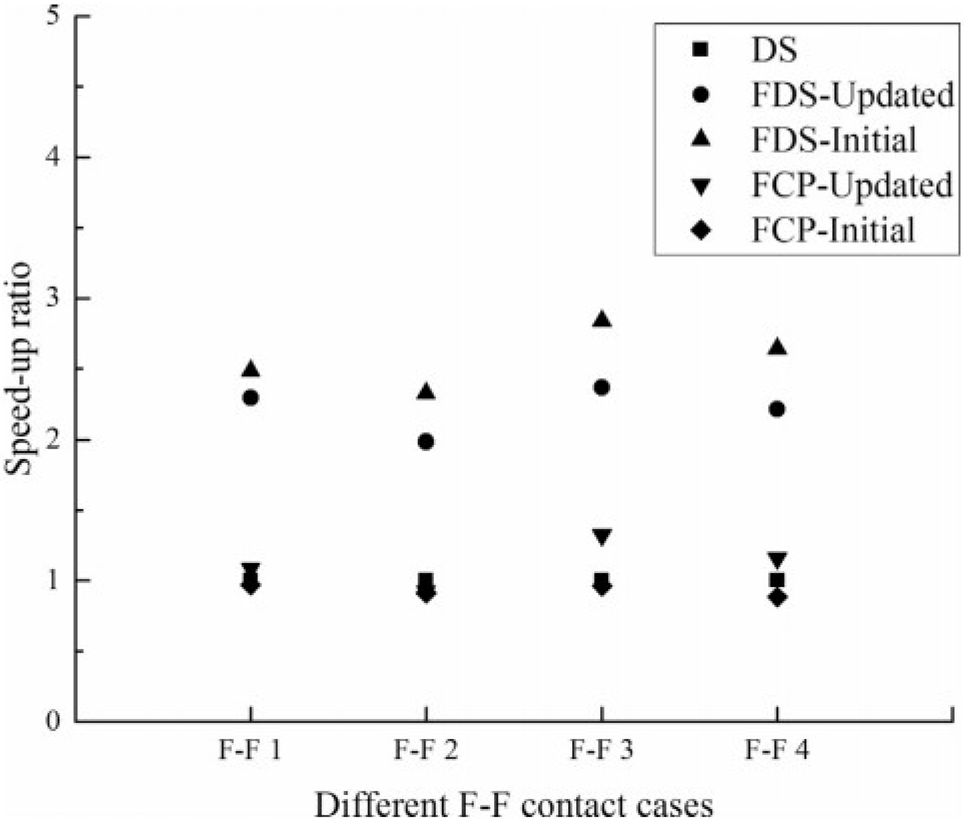

Fig. 15. Speed-up ratios of FDS and FCP for two polyhedra.

2.2.1. Vertex-to-face check

All 3-D v(*i*)-v(*j*), v(*i*)-e(*j*), v(*i*)-f(*j*), e(*i*)-v(*j*) and f(*i*)-v(*j*) contact pairs can be found in the v(*i*)-f(*j*) and f(*i*)-v(*j*) checking process. The v-f checking procedure is shown in Fig. 7.

First, the distance d_{vf} between the vertex coordinate and the infinite plane to which the face belongs is compared to the tolerance d_{tol}. Checking of this pair is continued only if d_{vf} is within the tolerance. If the v-f entrance is valid and d_{vf} is larger than the tolerance, then blocks *i* and *j* cannot contact within the current time step and all checking routines for this block pair are skipped.

If d_{vf} is within the tolerance, distance d_{vf} between v(*i*) and adjacent faces of f(j) is checked and the number, n_j, of v-f pairs that satisfy $d_{vf} < d_{tol}$ is counted. If distance d_{vf} of any v-f pair is larger than the tolerance, proceed to the next pair. Then n_j is compared with 1, 2 and 3. If $n_j > 3$, it is a v-v contact pair. If $n_j = 2$, it is a v-e contact pair. If $n_j = 1$, it is a v-f contact pair. Then the contact information is calculated and recorded including the VCPs. Only vertices, edges and faces within the VCP region are flagged and detected in following checking procedures.

2.2.2. Edge-to-edge check

The flow chart for the crossing e-e contact checking procedure is shown in Fig. 8. Orthogonal distance d_{ee} of the two edges is first compared with the contact distance tolerance. Valid entrance check is executed if $|d_{ee}| > d_{tol},$ to exclude cases where blocks are close enough but not in contact. If | $|d_{ee}| \leqslant d_{tol}$ and the nearest points are within the edge boundaries, the e-e contact information is calculated and recorded. An additional step follows the identification of valid crossing e-e pairs, to compute the VCPs and check if any edge is parallel to adjacent faces of the other edge. If it is, then it is a part of e-f or f-f contact pairs and edges connected with these parallel faces are flagged to be checked.

After all v-f and crossing e-e are found, the number n_j of contact pairs is counted. If n_j is 0, no contacts occur. If n_j is 1, the contact pair belongs to one and only one of v-v, v-e, v-f, crossing e-e types. If n_j is 2, the contact pairs may belong to parallel e-e or e-f contact types. If n_j greater than or equal to 3, the contact pairs belong to f-f contact types. The corresponding n_j related elements are then recorded on the contact list.

2.3. Updated contact detection

The contact types or the contact mode may change from one step to another. However, with the displacement limited by the tolerance, the contact detection results from the previous time step can be used to estimate where contacts may happen, thus reducing the number of contacts that need examinations.

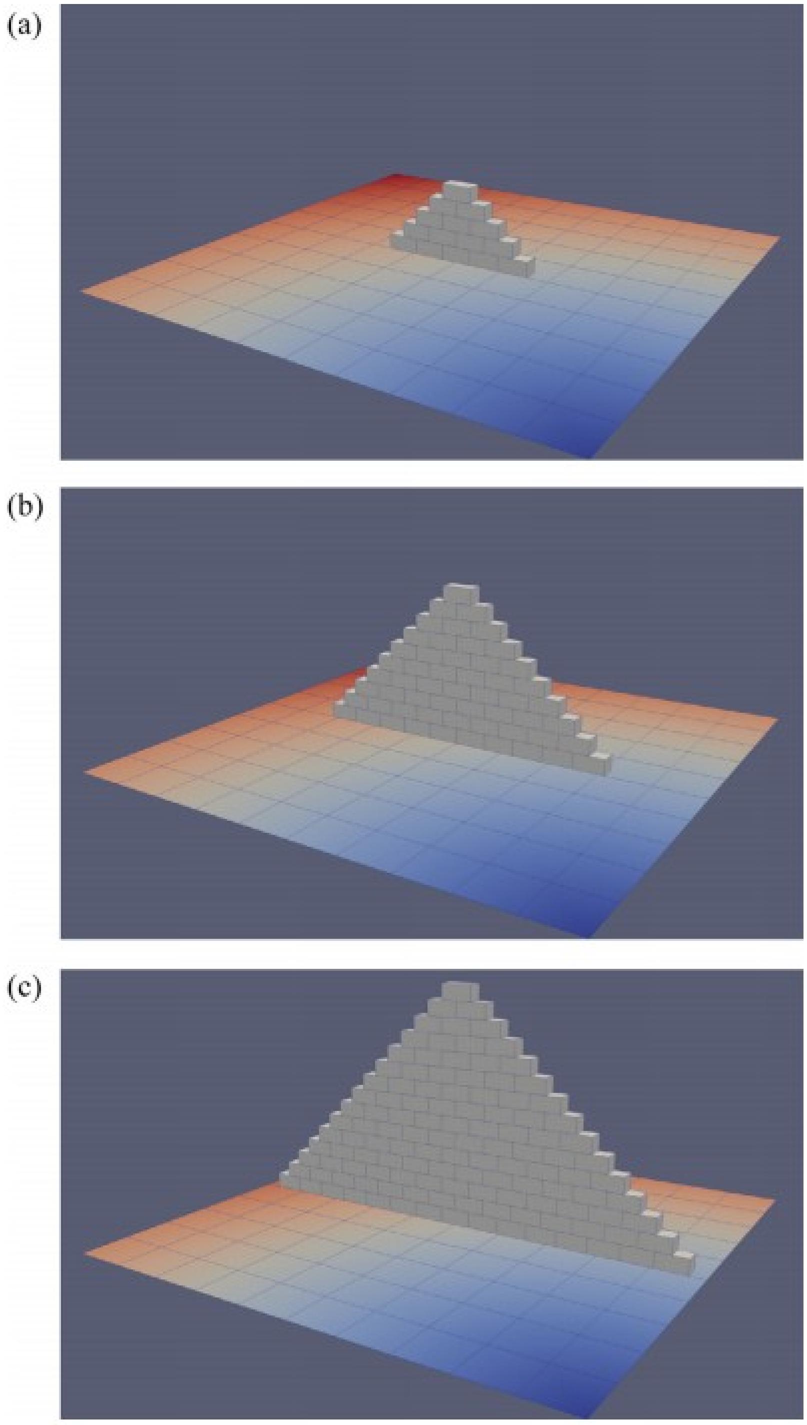

Fig. 16. Simulation of block wall models: (a) model 1 with 15 blocks; (b) model 2 with 55 blocks; (c) model 3 with 125 blocks.

In 2-D cases, a contact point on a vertex may move to its adjacent edges and a contact point on an edge may move to its boundary vertex. In 3-D cases, a contact point on a face may move to a boundary edge or a vertex. A contact point on a vertex may move to its connected edges or faces and a contact point on an edge may move to an adjoining vertex or face. Based on the entrance block theory [12], contact covers close to the reference point need to be checked. Based on different contact types, different close objects (vertices, edges, faces) are considered, respectively.

The updated checking procedure is shown in Fig. 9. It begins by checking whether the neighbor block list changes. If new neighbor blocks are added to the list, the initial contact detection routine is activated. However, for neighbor blocks that existed in the previous time step, the updated contact detection routine is activated according to contact type. First, VCPs are established according to previous contact types, and then distances between the vertices connecting previous contact pairs and the VCP are checked. The vertices within the VCP tolerance d_{tol} are counted as a variable n_v to track changes of the nearest elements. If n_v does not change compared to the previous step, the contact type is retained and a routine to update contact information is followed. If n_v changes, the close elements change and an updated detection procedure is followed for new contacting elements. First, a new list of potential checking boundary vertices, edges and faces, is formed. Then v(*i*)-e(*j*) and e(*i*)-v(*j*) types are checked from the new generated boundary element lists for 2-D polygons, while v(*i*)-f(*j*), f(*i*)-v(*j*) and e(*i*)-e (*j*) types are checked from the new boundary element lists for 3D polyhedra. Basic contact types (v-v and v-e in 2-D; v-v, v-e, v-f and crossing e-e in 3-D) are obtained through the checking procedure, as well as their contact points, contact normal vectors and normal distance. Then the contact information is recorded in a contact list maintained during the entire calculation process.

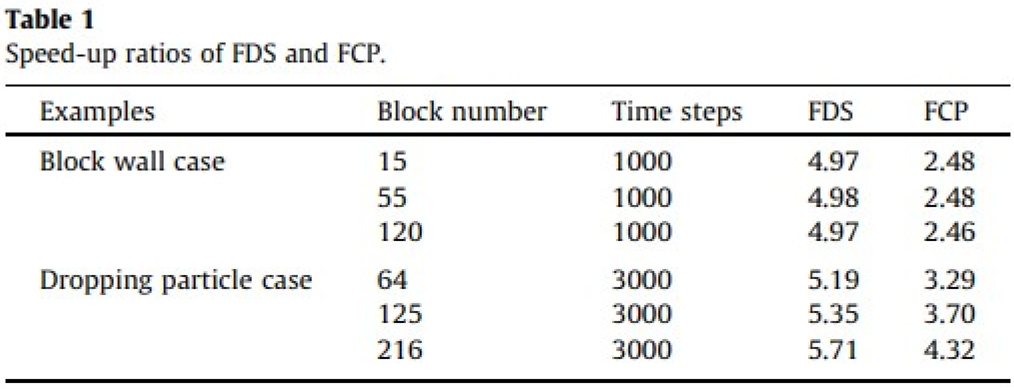

Table 1
Speed-up ratios of FDS and FCP.

Examples	Block number	Time steps	FDS	FCP
Block wall case	15	1000	4.97	2.48
	55	1000	4.98	2.48
	120	1000	4.97	2.46
Dropping particle case	64	3000	5.19	3.29
	125	3000	5.35	3.70
	216	3000	5.71	4.32

2.4. Data structures and computational cost

The traditional half-edge data structure [17] is used to represent the boundary of a polyhedron. Using this data structure, any geometry element can be obtained quickly. For example, any edges and faces connected to a vertex can be visited using a vertex object; adjacent vertices and faces connected to an edge can be accessed by using an edge object.

Assuming polyhedron *i* has n_{vi} vertices, n_{ei} edges and n_{fi} faces, the calculation cost of the fast direct search algorithm are as follows: In the initial checking subroutine, the checking cost for two convex polygons is $O(n_{vi} \cdot n_{ej} + n_{ei} \cdot n_{vj})$, and $O(n_{vi} \cdot n_{fj} + n_{fi} \cdot n_{vj} + n_{ei} \cdot n_{ej})$ for two convex polyhedra. During the actual detection computations, the total checking cost may be less because of some exclusion cases: (1) Once a contact pair is found, only adjacent geometry objects are flagged and checked in the subsequent checking procedure; (2) During detection cases of close but not-in-contact neighboring blocks, the exclusion algorithm terminates the unnecessary thorough checking; and (3) During the updated contact detection subroutine, only local boundary areas need to be detected and the computational cost is approximately equal to or less than that of two polygons in the un-optimized procedure.

3. Efficiency testing

3.1. Basic 3-D model

The direct search algorithm [8–10], which directly searches v-v, v-e, v-f and crossing e-e contact types, is used as the benchmark in the examples presented herein. A speed-up ratio *R* is defined as the ratio of the CPU run time for the direct search algorithm to that of the new fast direct search algorithm. Both the initial contact detection procedure and the updated contact detection procedure in the fast direct search algorithm and the fast common plane algorithm [3] are tested.

As shown in Fig. 10, case 1 examines different contact types (vv, v-e, v-f, crossing e-e, e-f, f-f and parallel e-e) of two hexahedra and the speed-up ratio is shown in Fig. 11. Case 2 considers the different contact types (v-v, v-e, v-f, crossing e-e, e-f, f-f and parallel e-e) of two tetrahedral, as shown in Fig. 12. The results are shown in Fig. 13. In Figs. 11 and 13 C E-E and P E-E refer to crossing e-e and parallel e-e. Case 3 considers F-F contacts of two blocks that are generated by cutting a hexahedron, as shown in Fig. 14. The results are shown in Fig. 15.

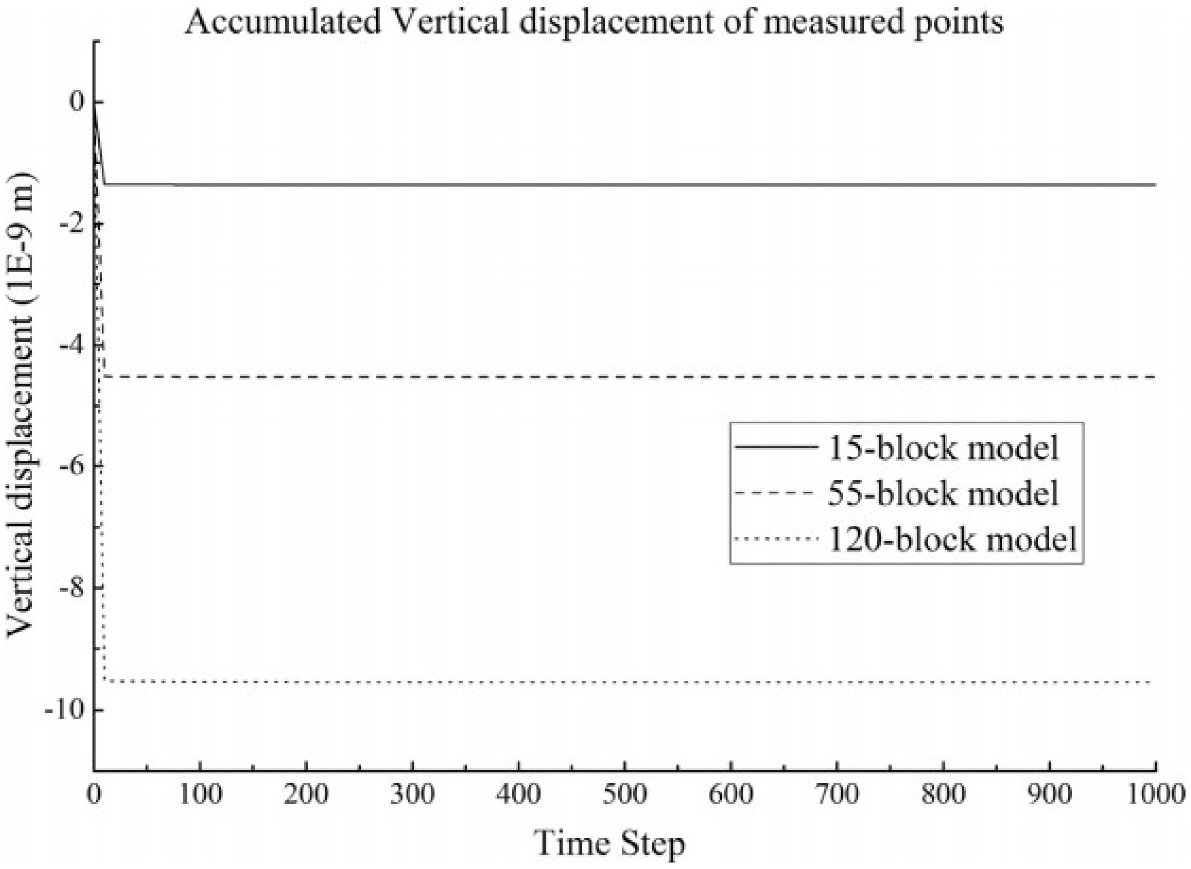

Fig. 17. Accumulated vertical displacement of measured points using FDS algorithm.

The results show that the speed-up produced by the fast direct search algorithm is significant, although it varies by contact type and shape of the objects in contact. The speed-up variation for the different contact types is related to the numbering sequence of particle topology and the number of adjacent vertices, edges and faces for the contact pair. The average speed-up ratio for different shapes of objects is different; on the order of 3.7 for hexahedrons and on the order of 2.2 for tetrahedral.

In basic model test, the FDS algorithm speed-up can be attributed to the following: (1) only v-f and e-e pairs are directly checked, thus reducing computational cost associated with v-v, v-e, e-f and f-f contacts; and (2) After the first contact pair is found, only its adjacent vertices, edges and faces are further examined, thus a thorough v-f and e-e pair checking is avoided.

In all basic test cases, the same contact types are obtained by using different contact detection algorithms which illustrates that the method is working correctly. For v-f, crossing e-e, v-v (belongs to e-f or f-f) and v-e (belongs to e-f or f-f) case, same contact normal is obtained using DS, FDS and FCP method, with a slightly different contact position, as different contact planes are used (DS and FDS use the face itself as the contact plane while FCP use the common plane as the contact plane). For v-v or v-e that not belong to ef or f-f types, the contact normals obtained by FDS and FCP are slightly different because the different definitions of common plane and virtual contact plane, also the contact position is slightly different owing to the utilization of different contact planes.

3.2. 3-D model evaluation

The performance of the algorithm is evaluated using two test cases, described below. These test cases illustrate how the algorithm performs for densely and sparsely packed particle systems. Additionally, it shows the influence of the number of particles on the algorithm's performance. In both cases, DS, FDS and FCP were used as the contact detection algorithm in the DDA framework to test the detection efficiency, and the detection results of FDS was used to do the simulation. The average time of each delicate contact detection step is used to compute the speed-up ratio of different algorithms. In addition to the comparison of efficiency, the simulation results using FDS algorithm and DDA were also shown to verify the effectiveness of the FDS algorithm.

The first case is the stability calculation of a block wall. Three examples were tested, with 15, 55 and 120 blocks, as shown in Fig. 16. Each block is assumed rigid and the contact spring stiffness is assumed 1E10 N/m. Initial time step is set as 0.001 s and the contact detection times for the first 1000 time steps are presented for comparison, as shown in Table 1. During the computation, the vertical displacements of the top block centroids in the three examples are recorded, as shown in Fig. 17. In this case, blocks are densely packed with a large number of contact pairs. The results show that the speed-up ratio is modest, on the order of 5 for FDS and 2.5 for FCP, because the contact list seldom changes during the computation and the calculation results gradually converge.

The second case involves loosely distributed particles falling due to gravity. Three models were tested, with 64, 125 and 216 blocks, as shown in Fig. 18. All particles are assumed rigid tetrahedral blocks, with volumes around 1.7E4 m^3, and the contact spring stiffness is assumed as 1E5 N/m. The initial time step is 0.001 s and the contact detection times for the first 3000 time steps are presented for comparison, as shown in Table 1. In this case, there are many neighbor block pairs that may not be in contact with each other. The results show that the speed-up ratio is quite significant, by a factor between 5 and 6 for FDS and on the order of 3+ for FCP. This speed-up of FDS and FCP compared to DS is principally owing to the reduction of the cost of detecting contact pairs of not-in-contact neighboring blocks.

4. Conclusions and discussions

We present an improved contact detection algorithm for convex polyhedra, by using initial contact detection and updated contact detection in a serial analysis. This improved algorithm has several benefits. First, the contact detection is more efficient because the updated contact detection algorithm reuses the results from the prior time step. Second, thorough examination of contact pairs is avoided owing to the consideration of spatial relationships, properties of convex particles, and the assumption of small displacement in a time step. Third, the exclusion subroutine increases the efficiency of the detection algorithm because it avoids checking of contact pairs with the least possible overhead. The test results show that the fast direct search algorithm produces respectable speed-up in evaluating contact between tetrahedra and hexahedra. To further improve the efficiency of contact detection, this algorithm can be extended using parallel computation techniques.

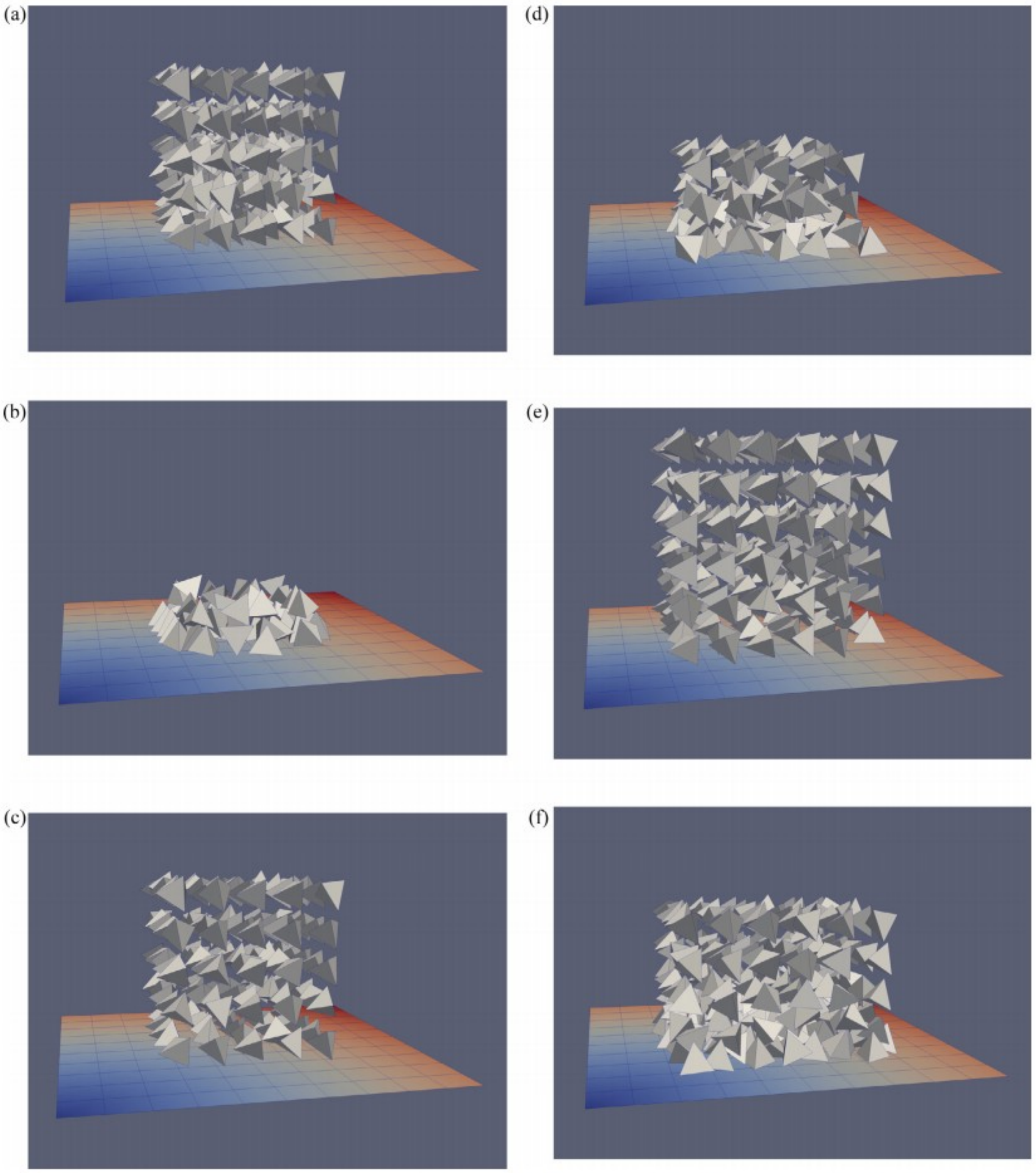

Fig. 18. Simulation of dropping particle models using FDS algorithm: (a) model 1 with 64 blocks; (b) model 1 after 3000 steps; (c) model 2 with 125 blocks; (d) model 2 after 3000 steps; (e) model 3 with 216 blocks; (f) model 3 after 3000 steps.

Acknowledgement

This study is supported by the National Basic Research Program of China (973 Program) (2014CB046904, 2014CB047101) and the China National Natural Science Foundation (51479191). The first author is supported by the China Scholarship Council during his study at UC Berkeley. The third and fourth authors are supported in part by the U.S. National Science Foundation (NSF) grant CMMI1363354 and the Edward G. Cahill and John R. Cahill Endowed Chair funds.

References

[1] Boon CW, Houlsby GT, Utili S. A new algorithm for contact detection between convex polygonal and polyhedral particles in the discrete element method. Comput Geotech 2012;44:73–82.

[2] Cundall PA. 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 Abstr 1988;25:107–16.

[3] Nezami EG, Hashash YMA, Zhao D, Ghaboussi J. A fast contact detection algorithm for 3-D discrete element method. Comput Geotech 2004;31:575–87.

[4] Nezami EG, Hashash YMA, Zhao D, Ghaboussi J. Shortest link method for contact detection in discrete element method. Int J Numer Anal Meth Geomech 2006;30:783–801.

[5] Liu XL, Lemos JV. Procedure for contact detection in discrete element analysis. Adv Eng Softw 2001;32:409–15.

[6] Keneti AR, Jafari A, Wu JH. A new algorithm to identify contact patterns between convex blocks for three-dimensional discontinuous deformation analysis. Comput Geotech 2008;35:746–59.

[7] Wu JH, Juang CH, Lin HM. Vertex-to-face contact searching algorithm for three-dimensional frictionless contact problems. Int J Numer Meth Eng 2005;63(6):876–97.

[8] Zhang H, Chen GQ, Zheng L, Han Z, Zhang YB, Wu YQ, et al. Detection of contacts between three-dimensional polyhedral blocks for discontinuous deformation analysis. Int J Rock Mech Min Sci 2015;78:57–73.

[9] Zheng F, Jiao YY, Zhang XL, Tan F, Wang L, Zhao Q. Object-oriented contact detection approach for three-dimensional discontinuous deformation analysis based on entrance block theory. Int J Geomech 2016:E4016009.

[10] Zhang H, Liu SG, Han Z, Zheng L, Zhang YB, Wu YQ, et al. A new algorithm to identify contact types between arbitrarily shaped polyhedral blocks for threedimensional discontinuous deformation analysis. Comput Geotech 2016;80:1–15.

[11] Wu W, Zhu HH, Zhuang XY, Ma GW, Cai YC. A multi-shell cover algorithm for contact detection in the three dimensional discontinuous deformation analysis. Theor Appl Fract Mech 2014;72:136–49.

[12] Shi GH. Contact Theory Sci China Tech Sci 2015;58:1–47.

[13] Munjiza A, Andrews KRF. NBS contact detection algorithm for bodies of similar size. Int J Numer Meth Eng 1998;43(1):131–49.

[14] Perkins E, Williams JR. A fast contact detection algorithm insensitive to object sizes. Eng Comput 2001;18:48–62.

[15] Williams JR, Perkins E, Cook B. A contact algorithm for partitioning N arbitrary sized objects. Eng Comput 2004;21:235–48.

[16] Shi GH. Discontinuous deformation analysis: a new numerical model for the statics and dynamics of block systems. Ph.D. Thesis. University of California, Berkeley; 1998.

[17] Muller DE, Preparata FP. Finding the intersection of two convex polyhedra. Theor Comput Sci 1978;7:217–36.