Computers and Geotechnics 98 (2018) 205–216

Contents lists available at ScienceDirect

Computers and Geotechnics

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

Research Paper

A fast method for fracture intersection detection in discrete fracture networks

Shaoqun Dong[a,b], Lianbo Zeng[a,b,*], Peter Dowd[c], Chaoshui Xu[c], Han Cao[d]

[a] College of Geoscience, China University of Petroleum, Beijing 102249, China
[b] State Key Laboratory of Petroleum Resources and Prospecting, China University of Petroleum, Beijing 102249, China
[c] School of Civil, Environmental and Mining Engineering, University of Adelaide, Adelaide 5005, Australia
[d] Institute of Remote Sensing and Digital Earth, Chinese Academy of Sciences, Beijing 100094, China

ARTICLE INFO

Keywords:
Fracture intersection detection
Filtering
Refining
Bounding box
Sweeping line
Discrete fracture network

ABSTRACT

The detection of fracture intersections is an important topic in discrete fracture network modelling for assessments such as connectivity analysis and subsequent fluid flow evaluations. However, the standard method for such detection is very time-consuming especially for large fracture networks as the detection time often increases exponentially with the number of fractures in the network. In this paper, we introduce the bounding box and sweeping line (BBSL) method as a new fast algorithm to solve the problem. BBSL comprises two consecutive steps: filtering and refining. In the filtering step, an axis-aligned minimum bounding box (AABB) and an improved sweeping line method (SLR – sweeping line for rectangles in 2D or SLC – sweeping line for cuboids in 3D) are introduced to filter out pairs of fractures that have no possibility of intersection. The proposed refining in BBSL consists of coarse refining and fine refining. Coarse refining combines the inner and outer products of vectors to filter out non-intersecting pairs of fractures. Fine refining is then used to further assess fracture intersections and to determine the intersection coordinates. To demonstrate the application of the proposed method a series of comparison experiments were conducted using 2D and 3D discrete fracture networks with different fracture densities. For filtering, the results show that the proposed method is significantly more efficient than the commonly used methods such as brute force (BF) and sweeping and pruning (SAP). For refining, the proposed method significantly outperforms the commonly used refining method.

1. Introduction

Discrete fracture network (DFN) modelling is widely used to model fractures in rock masses [1–4] for flow analysis as it is simple to implement and allows for a better integration of geological data into flow models [5]. The common DFN models include the Baecher model [6–8] and the GEOFRAC (Veneziano's) model [3,9]. The Baecher model is based on marked point processes [2] while the GEOFRAC model is based on Poisson line (2D) or Poisson plane (3D) processes [3]. The Baecher model is more widely used due to its simplicity and has many modified versions, such as the enhanced Baecher model [10,11], the Nearest Neighbour model [12], the War Zone model [12], fractal model [13,14] (the Levy-Lee clustering model [15], parent and daughter model [10], binary fractal fracture network model [16,17]), the random polygon model [2,4]. DFN models in general only deal with spatial distributions of fractures, but fracture mechanics can also be integrated [18–22]. A detailed discussion of the integration is, however, beyond the scope of this paper.

In the DFN approach, fracture connectivity analysis plays an important role in assessing the flow behaviour [9,11,23,24] of the fracture network and in understanding the mechanical properties of rock masses (such as deformation and strength) [25,26]. The analysis of intersections between fractures within the network is the key to assessing properly the fracture connectivity in a DFN model [9,27].

Fractures in 2D DFN models are commonly represented by line segments and in 3D DFN models are represented by planar polygons, circles or ellipses [2–4,11]. In this paper, line segments and planar polygons are used respectively for 2D and 3D DFN models and therefore fracture intersection detection becomes the detection of intersections between lines in 2D and planar polygons in 3D. In video games, animation, physical simulations and robotics, a wide range of collision detection (CD) algorithms have been developed to detect the intersection of two or more objects [28,29]. Detection is usually performed in two phases: a broad phase and a narrow phase [30,31]. Broad phase collision detection, also termed filtering, provides an efficient way to remove pairs of objects that are obviously will not collide, which is

* Corresponding author at: College of Geoscience, China University of Petroleum, Beijing 102249, China.
E-mail address: lbzeng@sina.com (L. Zeng).

https://doi.org/10.1016/j.compgeo.2018.02.005
Received 4 August 2017; Received in revised form 21 December 2017; Accepted 5 February 2018

usually a computationally low-cost operation [32]. To improve the efficiency, objects in the broad phase of CD are usually encapsulated into simple shapes as collision detection among simple shapes is much quicker. Narrow phase collision detection, also termed refining, analyses more carefully the pairs of objects that are not culled in the first step filtering. This step employs more refined, but slower, algorithms to determine with certainty whether objects intersect each other and if so to calculate the intersection between them.

Recent research has focused on filtering and refining to improve the efficiency of fracture intersection detection [3,4,9,24,26,25,33,34]. Studies on refining often focus on fast and accurate assessments of fracture intersections. Einstein et al. [9] proposed an algorithm for determining intersections between two fractures, implemented in GEOFRAC, a stochastic fracture pattern-modelling program. The method is a systematic workflow to calculate the intersection between pairs of fractures. Alghalandis [27] formulated a framework for fracture intersection analysis and implemented it in ADFNE [4], an open-source toolbox of fracture modelling, which is also used in this research.

In fracture intersection detection, if refining is employed directly without filtering, the computational cost will increase exponentially as the number of fractures increases because refining essentially uses an exhaustive method (or brute force method) to search through all possible pairs of fractures for intersection detection. For this reason, improving the effectiveness of filtering is a relatively efficient way to speed up the intersection detection and a bounding box is often employed for this purpose. Ivanova et al. [3] introduced the bounding sphere to filter out fracture pairs that obviously do not intersect and then used the algorithm proposed by Einstein et al. [9] to determine and calculate the fracture intersection. Liu et al. [33] proposed an approach for progressively identifying pairwise fracture intersections in 3D fracture networks based on spatial indexing and bounding boxes. Li et al. [34] proposed a method to further optimize the 'two stepwise approach' intersection analysis using oriented bounding boxes, which was shown to be efficient. Zhan et al. [26] employed the stepwise approach and bounding boxes to assess the fracture connectivity in the rock mass at the Songta hydropower station in Southwest China. This work improved fracture intersection detection over the refining only approach. In this paper, a fast and simple method, termed BBSL (bounding box and sweeping line), is introduced as a means of significantly improving the efficiency of fracture intersection detection.

The proposed filtering uses the axis-aligned minimum bounding box (AABB) and the improved sweeping line method to filter out pairs of fractures that do not obviously intersect each other. In 2D a sweeping line for a rectangle (SLR) is used and in 3D a sweeping line for a cuboid (SLC) is used. The sweeping line technique has been demonstrated to be efficient for collision detection in computer graphics and robotics and it is introduced in this paper to improve the efficiency of fracture intersection detection.

The proposed refining includes coarse refining and fine refining. Although filtering can remove a large number of fracture pairs that have no possibility of intersection with each other, there remains a significant number of retained fracture pairs that are not intersected. If only conventional refining is employed directly after filtering to assess the details of intersection in the retained list, the computing cost will be unnecessarily high. Therefore, a coarse refining step is introduced in our implementation before the fine refining (equivalent to conventional refining in this paper) is applied. As the proposed coarse refining is done by combining the inner and outer products of vectors for fast removal of unnecessary pairs of fractures, the overall computing cost is reduced. Fine refining uses the inner product only to further assess the pairs of fractures retained after coarse refining and evaluate their intersections, including the intersection coordinates. Finally, to demonstrate the application of the proposed method, a series of comparison experiments are conducted using 2D and 3D discrete fracture networks with different fracture densities.

2. The principle of bounding box and sweeping line (BBSL) method

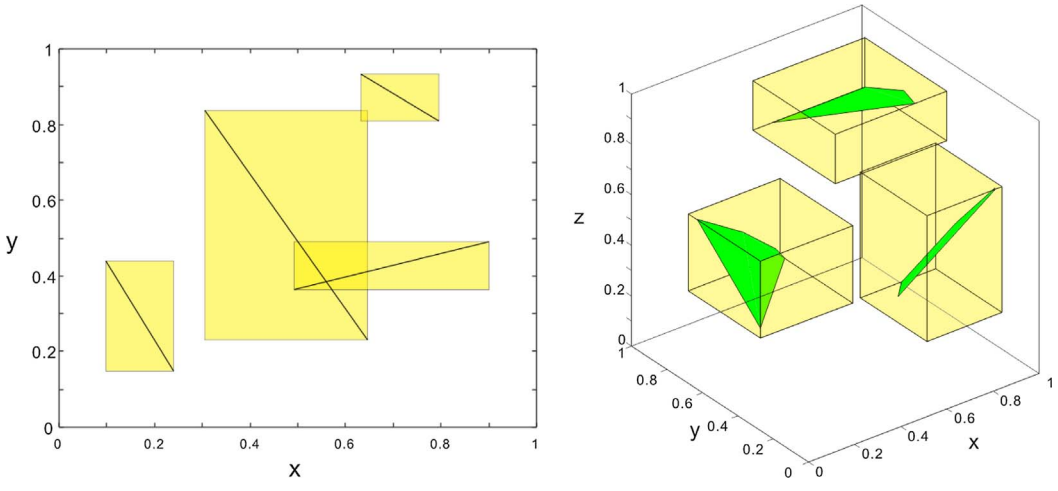
The proposed filtering uses object bounding boxes (BB) [35] and the sweeping line method (SL) to detect pairs of overlapping boxes which indicate possible fracture intersections. Details are presented in Section 2.1. Sections 2.2 and 2.3 discuss coarse and fine refining respectively. In Section 2.4, the systematic BBSL workflows for both 2D and 3D DFN models are summarized.

2.1. Filtering

2.1.1. Bounding box (BB)

In this application, a bounding box of a fracture is the minimum area (2D) or volume (3D) that completely contains the fracture. If two bounding boxes do not intersect, the two contained fractures cannot intersect.

For computational convenience, the common types of bounding area (volume) include bounding circle (sphere), bounding ellipse (ellipsoid), bounding rectangle (cylinder), axis-aligned bounding box (AABB) [36], oriented bounding box (OBB) [37]. AABB is the simplest and easiest bounding volume and is employed in our work. The edges of



(a) AABB in a 2D DFN model (b) AABB in a 3D DFN model

Fig. 1. AABB in DFN models.