Graphical Models 97 (2018) 1–16

Contents lists available at ScienceDirect

Graphical Models

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

Efficient non-incremental constructive solid geometry evaluation for triangular meshes

Bin Sheng*,[a], Ping Li*,[b], Hongbo Fu[c], Lizhuang Ma[a], Enhua Wu[d,e]

[a] Department of Computer Science and Engineering, Shanghai Jiao Tong University, China
[b] Faculty of Information Technology, Macau University of Science and Technology, Macau
[c] School of Creative Media, City University of Hong Kong, Hong Kong
[d] Faculty of Science and Technology, University of Macau, Macau
[e] State Key Laboratory of CS, ISCAS & University of CAS, China

ARTICLE INFO

Keywords:
Boolean operations
Plane-based geometry
CSG evaluation
Hybrid representation

ABSTRACT

We propose an efficient non-incremental approach to evaluate the boundary of constructive solid geometry (CSG) in this paper. In existing CSG evaluation methods, the face membership classification is a bottleneck in executive efficiency. To increase the executive speed, we take advantages of local coherence of space labels to accelerate the classification process. We designed a two-level grouping scheme to group faces that share specific space labels to reduce redundant computation. To further enhance the performance of our approach in the non-incremental evaluation, we optimize our model generation which can produce the results in one-shot without performing a step-by-step evaluation of the Boolean operations. The robustness of our approach is strengthened by the plane-based geometry embedded in the intersection computation. Various experiments in comparison with state-of-the-art techniques have shown that our approach outperforms previous methods in boundary evaluation of both trivial and complicated CSG with massive faces while maintaining high robustness.

1. Introduction

Constructive Solid Geometry (CSG) has long been a popular modeling tool for Computer-Aided Design and Computer-Aided Manufacturing (CAD/CAM). It constructs complex models by combining primitives using a series of regularized Boolean operations [1]: union, intersection, and difference. A CSG can be represented by a binary tree, called the CSG tree. The leaves of the CSG tree represent the primitives while the internal nodes represent the Boolean operations. Another widely-used method for representing CSG is polygonal mesh representation through boundary evaluation. Most boundary evaluation methods mainly contain two phases: intersection computation and face membership classification. For most of boundary evaluation methods, robustness and efficiency are two major issues. During the last few decades, many techniques have been developed to pursue robust boundary evaluation. However, in terms of efficiency, there is much space for improvement.

One of the keys in deciding the efficiency of the evaluation is the face classification. It is based on space labels of faces. The number of space label of a face equals to the number of primitives. For large CSG with massive faces and primitives, computing these space labels is extremely time-consuming. A common idea for acceleration is to take advantages of the local coherence of space labels. If a face is inside (or outside) a specific primitive, its neighboring faces are likely to be inside (or outside) the primitive. Determining whether two adjacent faces share the same space labels is relatively simple. Through grouping the faces that share the same labels and reusing these labels, unnecessary repetitive computation can be largely reduced.

Taking advantages of the local coherence of space labels, previous studies have developed localized schemes [2–4] based on different grouping units such as voxels and octree cells. These grouping units are essentially cubes. With these cubes, the space division data structures constructed during the intersection computation is able to be recycled. However, using the cube as a basic grouping unit has disadvantages in handling arbitrary shapes. Under localized schemes, connected faces that shared the same space label are grouped together. The face group, which is essentially a union of connected faces, can have arbitrary shapes. The cube-based grouping scheme can only provide a rough approximation for the most shape of the face groups. Fig. 1(a) and (b) presents a 2D illustration. The red cubes (represented in red grids in Fig. 1(a)) contain the intersection of two primitives. Faces (represented as edges in the figure) in these two cubes are left ungrouped since they have different space label. To compensate for the inaccuracy, extra time-consuming computation is introduced to classify these ungrouped

---

* Corresponding authors.
E-mail addresses: shengbin@sjtu.edu.cn (B. Sheng), pli@must.edu.mo (P. Li).

https://doi.org/10.1016/j.gmod.2018.03.001
Received 30 January 2018; Accepted 5 March 2018

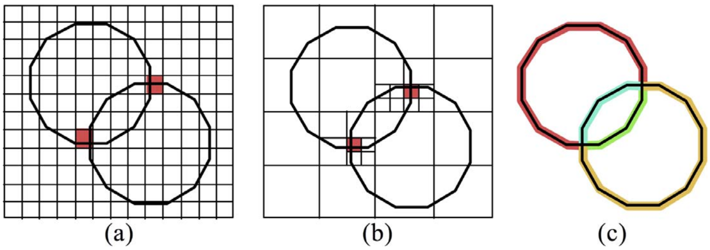

Fig. 1. 2D illustration of different grouping schemes. Face grouping in 3D can be represented with edge grouping in 2D. Because the boundaries of the polygons are represented as edges in 2D. Faces are grouped using (a) voxels, (b) octree cells, and (c) faces as the grouping units. Different groups in (c) are marked by different colors.

faces. To avoid additional computation, we proposed a face-based localized scheme which is able to handle arbitrary shape (Fig. 1(c)).

Another barrier for pursuing high efficiency is the incremental algorithm adopted in previous methods [2,5,6]. Previous methods are designed to evaluate one Boolean operation at a time. For a large CSG tree with more than two primitives, it has to be decomposed into a series of Boolean operations which are evaluated separately. These incremental algorithms are highly inefficient and inevitably generate unneeded massive intermediate results. Incremental algorithms have been used in design application for a considerable long period. In practical design, constructing CSG models by progressively adding primitives is very common. The intermediate results can be used for fast preview. However, with the appearance of GPU-based approximate evaluation algorithms [3,4] and CSG visualization algorithms [7,8], the intermediate results generated by the incremental methods are no more suitable for preview computation. Thus, the non-incremental algorithm is a better choice for final mesh generation.

In this paper, we propose a robust approach to perform CSG boundary evaluation with triangular mesh primitives. To overcome the drawbacks of the cube-based localized scheme, our approach uses a special two-level face-based localized scheme and applies a flood-filling algorithm to group faces. To avoid unnecessary computation and pursue high efficiency, an optimized non-incremental evaluation of CSG is applied instead of traditional incremental algorithms. The robustness and exactness of our approach are strengthened by applying plane-based representation in the intersection computation. In general, our approach has the following contributions:

1.1. *Face classification using two-level grouping*

A two-level grouping scheme is designed to reuse space labels. The input faces are firstly grouped according to the primitive they belong to in the first level. Then, groups in the first level are further divided according to the intersection as shown in Fig. 2(d). A flood-filling algorithm is applied to enable efficient grouping and label propagation among adjacent faces. This scheme makes a balance between the benefit of face label sharing and the cost of grouping faces for the best performance in the face classification.

1.2. *Efficient non-incremental evaluation*

The non-incremental evaluation we used in our approach contains a set of techniques, including face-nested Binary Space Partitioning (BSP) and multi-level CSG tree trimming. These techniques are able to process the complex conditions of the intersection and face classification efficiently.

1.3. *Plane-based triangle intersection test*

To avoid the introduction of errors during intersection computation, we combine a triangle-triangle intersection method with a plane-based representation. With P-reps, our triangle intersection test is free from constructing new points.

Multiple experiments have confirmed that our approach has advantages in efficiency and robustness when compared to the state-of-the-art techniques [2,5,9–11]. Our approach is able to quickly and robustly perform CSG evaluations not only for trivial CSG, e.g. single Boolean operations, but also for large CSGs with hundreds of primitives.

The remaining parts of our paper are organized as follow. The next section gives a literature review of the issues in CSG evaluation. In Section 3, we give a brief introduction to the CSG evaluation including terminology and definitions. Section 4 provide an overview of our approach. In Section 5 and Section 6, we provide detail descriptions of the core of our approach: the plane-based intersection computation and the face classification framework. Experimental results and comparison with previous methods are presented in Section 7. Finally we conclude our paper with a short summary and an outlook to future research in Section 8.

2. Related work

As mentioned in [12], “nonrobustness refers to qualitative or catastrophic failures in geometric algorithms arising from numerical errors.” In other words, geometric robustness does not equal to precise numeric. Small numeric errors may be negligible in some scientific computation, but may sometimes cause topological deficiency or other catastrophic failures in geometry. Pursuing robustness of CSG evaluation has been a challenging problem since its inception in 1980s [13,14]. The non-robustness is inherited from the Boolean operations on solids in the process of building blocks of CSG. Previous research attempted to solve such issues using arbitrary precision arithmetic [9,15–18] and exact interval computation [19–21]. These methods achieve robustness at the cost of massive memory and have no limitation in computational time. Thus, they may be impractical for evaluation of CSG with massive faces. For example, the state-of-the-art robust Boolean operation [18] (implemented with arbitrary precision arithmetic in the Computational Geometry Algorithms Library (CGAL) [22]) is 20 times slower than its non-robust version. To minimize the cost of efficiency and guarantee robustness simultaneously, introduction of plane-based representation in the evaluation is a practical choice. Sugihara and Iri [23] introduced a plane-based representation of polyhedra. They pointed out that Boolean operations are fundamentally robust using plane equation as the primary geometry representation. According to their theory, the evaluation of Boolean operations can be performed without the operation of “constructions” [24], thereby avoiding the introduction of any numerical errors. Bernstein and Fussell [6] further noticed that a Binary Space Partitioning (BSP) merging for Boolean operations [25,26] is actually a plane-based technique. Therefore, they combined the two conceptions, plane-based geometry and BSP merging, to develop an unconditionally robust method for Boolean operations of polyhedra. Equiped with Shewchuk’s adaptive geometry predicate [27], the speed of Bernstein and Fussell’s method increases dramatically and but still twice slower than previous non-robust methods. Wang et al. [28] design an efficient algorithm to extract the manifold surface that approximates the boundary of a solid represented by a BSP tree.