1994

Locally Resolvable B-reps

Carlos Gonzalez-Ochoa

George Vanecek

Report Number:
94-076

LOCALLY RESOLVABLE B-REPS

Carlos Gonzalez-Ochoa
George Vanecek, Jr.

CSD-TR-94-076
November 1994

Locally Resolvable B-reps[1]

Carlos Gonzalez-Ochoa[2] and *George Vanecek Jr.*[3]

Department of Computer Sciences
Purdue University
West Lafayette, IN 47907-1398

Abstract

A typical B-rep describes a solid in its full detail. Operations that locally analyze the B-rep usually process the entire B-rep due to the lack of spatial ordering of its topological entities. For very complex objects, this global processing is time consuming and yet unnecessary. One approach in avoiding the global processing is to use several B-reps of different levels of detail. However, this does not always help. For instance, when objects in close proximity need to be analyzed for contact, the need for a local increase in resolution forces a global increase in detail. Our motivation is to unify the different levels of detail and provide the means to locally control them.

We introduce a *locally resolvable b-rep* (LRB-rep) as a non-manifold B-rep data structure for representing objects. The LRB-rep locally hides the solid's details with several layers of faces. A subset of the faces provides a global approximation of this solid, which we call the *wrapper*. The wrapper starts with the faces of the LRB-rep's outer layer and controlled by some application, may change its shape. Local face refinement reveals more detail and eventually resolves the wrapper to the original object with the highest level of detail.

The LRB-rep is motivated by the need to avoid checking all the faces while detecting collisions between two complex objects in close proximity. Instead of checking the complex boundaries of both solids, the wrapper of one solid can be checked against the other solid. The wrapper faces that interpenetrate are refined and recursively checked. Faces that do not interpenetrate are discarded from further checking and refinement.

In this paper we introduce the LRB-rep, the wrapper and its operations, and suggest their use in a collision detection application.

[1] This work has been support by ONR Grant N00014--94--1--0576.

[2] http://www.cs.purdue.edu/people/gonzalez, gonzalez@cs.purdue.edu

[3] http://www.cs.purdue.edu/people/vanecek, vanecek@cs.purdue.edu

1. Introduction

Classical boundary representations, such as the extended winged-edge data structure and its many variations, represent an object in all its detail [10]. The representations enumerate all the vertices, edges, and faces along with the topological adjacency information needed for the reconstruction of the boundary. When objects are not being modified and they are used in applications that treat them as rigid bodies, as for example, in rendering or when moved and collided with others, these classical boundary representations begin to lose their appeal. Although it is the boundary information that is needed in these applications, it is difficult to access spatially. Consequently, for very large and complex objects, auxiliary structures such as an octree, an MSP tree [6], a sphere tree, some grid structure, or some hierarchical bounding-volume structure needs to be added to find the entities in sublinear time. Such a hybrid system quickly grows in software complexity and become difficult to understand and maintain. Furthermore, it still retains a look-up complexity that is a function of the overall boundary complexity.

What is needed is a boundary-based representation that inherently provides a spatial access to local detail and that has a retrieval access complexity that is independent of the overall complexity. The later requirement means that if it takes n operations to resolve a face to its highest level of detail, it should ideally still take n operations after the other side of the object is partitioned into a million components. This is what is meant by local complexity being independent of global detail.

Representing an object at different levels of detail is a common technique used in visualization systems (for example, walk throughs [3]). The object is reconstructed at lower levels of detail based on the distance to the observer. This is based on the observation that much of the detail becomes invisible at a certain distance and therefore need not be rendered explicitly. Here however, it is an all or nothing proposition since close up means that the entire object is represented with all its detail.

If instead of rendering the objects we want to, for instance, check two objects in close proximity for contact, the above method would force the highest levels of detail. Localizing the search to only the faces in the local vicinity of the contact may be possible if the levels of detail are merged in a way that allows incremental testing of only some faces at each level.

One approach was suggested by de Floriani with the Face-adjacency Multigraph [1], which is a hierarchical representation using B-reps. Protrusions and depressions are treated as separate solids that are related by their contact faces. The relationship is recorded in a face-adjacency hypergraph data structure. This representation is fundamentally feature based and does not easily generalize to one that has an efficient locally resolvable nature.

Our method is based on face abstraction, and cellular decomposition of space. Several adjacent faces can be abstracted to a smaller set of new faces which together with the original set enclose a region of space outside and adjacent to the solid. That is, (many)

faces at a higher resolution detail are abstracted to (fewer) faces at lower resolution detail. Successive face abstractions can yield a simple but larger volume such as a block. The process of abstracting faces is similar to filling in the depressions and cavities with material until a smooth simple object results. This idea has been studied in the area of feature recognition originally by M. Henderson.

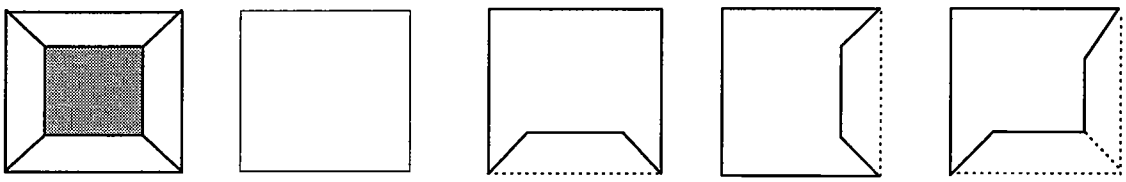

Figure 1 *An example of a 2D LRB-Rep (a) along with four different shapes of the wrapper. (b) shows the initial wrapper. (c), (d), and (e) show the wrapper after three different face refinements.*

We propose a new boundary-based representation called the Locally Resolvable B-rep (LRB-rep). This is a typical polyhedral and nonmanifold B-rep which contains faces, edges, and vertices added in the face-abstraction process in addition to the original entities. Based on the application, a subset of the faces, edges, and vertices is conditionally selected to approximate the interior solid at various levels of detail. We refer to this subset as the wrapper. The exterior most faces of the LRB-rep compose the initial wrapper. The face refinements modify the shape of the wrapper by removing and adding faces to the wrapper. The limit of all the face refinements is the original B-rep with the greatest level of detail. As an example, Figure 1 shows a 2D LRB-rep and three possible wrapper shapes, with the dotted lines indicating where each wrapper was refined.

In this paper we introduce the LRB-rep, the wrapper and its operations, and motivate their use in collision detection applications.

2. Locally Resolvable B-reps

An LRB-rep extends a standard B-rep by allowing additional faces on the exterior of the solid, and requiring a single *face-collector* for each closed region of space. In this section we describe the restrictions placed on the faces and the collectors, and the properties needed for a consistent LRB-rep.

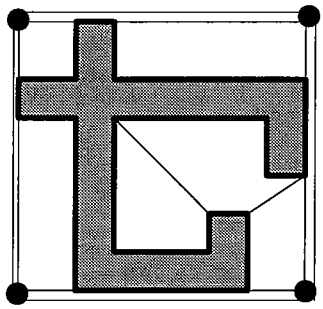

Figure 2 *An example of a 2D LRB-rep. The innermost edges of the polygon are darkened. Four of the vertices have been darkened to illustrate the overlapping adjacent edges. In total there are 22 vertices, 33 edges, and 12 collectors. Of these, 18 edges and vertices belong to the original polygon.*

In effect, the LRB-rep adds a topological node called a *face collector* (or just a *collector*) to the solid-face-edge-vertex hierarchy of a B-rep. This node adds a face-region topological-adjacency information typical in Voronoi structures [9]. The regions can be of any shape and possibly have no volume as happens for regions between overlapping faces. Each LR-Brep has at least two collectors, called the *inner* and the *outer* collectors. The inner collector represents the interior of the object and thus collects all the original faces of the object. The outer collector represents the exterior and collects the faces of the outer-most wraper.

From any exterior face of the LRB-rep, a number of face-region and region-face transitions resolves the exterior face (at a low level of detail) to an interior face (at a high level of detail). The number of transitions needed to resolve to the interior face depends on the number of faces at the highest level of detail in the local proximity and partly on how the LRB-rep was constructed. There is no unique (or canonical) LRB-rep for an arbitrarily complex object as the face-abstraction process of adding new faces to the outside is arbitrary. As an example of an LRB-rep, Figure 2 shows a 2D example.

The following properties hold for the faces of an LRB-rep:

- Each face belongs to two collectors.
- No two faces can intersect transversely except at their edges.
- Faces my overlap.
- Each face is adjacent to at least one other face at each edge.
- The faces of the original object all belong to the inner collector.

The following properties hold for the collectors of an LRB-rep:

- Each collector is associated with one and only one region of space and collects all the faces that border the region.

- Each region has exactly one collector.

The following operations are defined on the LRB-rep:

- `newLRB-rep: B-rep → LRB-Rep`
  Create a new LRB-rep given a B-rep. Create two collectors each associated with all the faces of the B-rep; one is the inner and the outer collector.
- `innerCollector: LRB-rep → Collector`
  Return the inner collector. This collector contains all the faces of the original B-rep.
- `outerCollector: LRB-rep → Collector`
  Return the outer collector. This collector contains the faces for the initial wrapper.
- `abstract: LRB-rep × Face → LRB-rep`
  Adds a new face, determines what region the face is added to, and updates the collectors. This face cannot partially overlap other faces. More details on this operation are given below.
- `subdivide: LRB-rep × Face × {Face} × Collector → LRB-rep`
  Subdivide a face by a set of faces that completely cover the face on the side indicated by the collector. More details on this operation are given below.
- `merge: LRB-rep × Collector × Collector → LRB-rep × Collector`
  Discards the common faces to both collectors and associates the resulting faces into a single collector.
- `associate: LRB-rep × Collector × Face → LRB-rep`
  Bind the given face with the collector.
- `deassociate: LRB-rep × Collector × Face → LRB-rep`
  Remove the given face from the collector.
- `newCollector: LRB-rep → Collector`
  Create a new collector in the LRB-rep with no associated faces.
- `sharedFaces: LRB-rep × Collector × Collector →{Face}`
  Return the set of shared faces, given that the regions of the two collectors are adjacent, otherwise returns the empty set.
- `faceCollectors: LRB-rep × Face → Collector × Collector`
  Return the two collectors associated with the face.
- `facesOfCollector: LRB-rep × Collector → { Face }`
  Return the set of faces associated with the collector.
- `ifFaceOfCollector:LRB-rep × Collector × Face → Boolean`
  Return true if the face is associated with the collector.

Not any collection of faces and collectors is well defined. We say that an LRB-rep is consistent if all its regions and faces are consistent. A face is *consistent* if both of the following properties are satisfied:

1. All the edges of a face are adjacent to one or more other faces.
2. Each face is associated with two distinct collectors.
3. No two faces intersect transversely.

A region is consistent if it is associated with a collector, and the collector with the region.

We now elaborate on two of the above operations: face abstraction and face subdivision.

In the process of abstraction, new faces are added to the LRB-rep and existing collectors are modified. This is performed by the operation *abstract*. Here, two cases can occur depending on whether the region to which the face is being added splits into two regions:

1. The region to which the face is added does not split. Here the face is associated on both sides with the same collector, leaving the face temporarily in an inconsistent state.
2. The region splits. Here a new collector is created, the faces of the original collector that lie on one side of the inserted face are deassociated with the original collector and associated with the new collector. Finally, the inserted face is associated with both collectors.

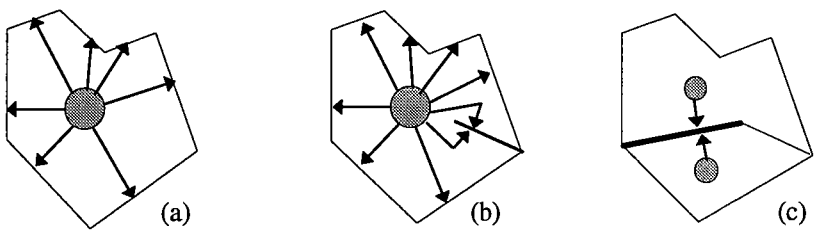

Figure 3. *The possible results of adding faces, (b) and (c), by the operation abstraction to an original region (a).*

As an example of the abstraction operation, Figure 3 shows these cases.

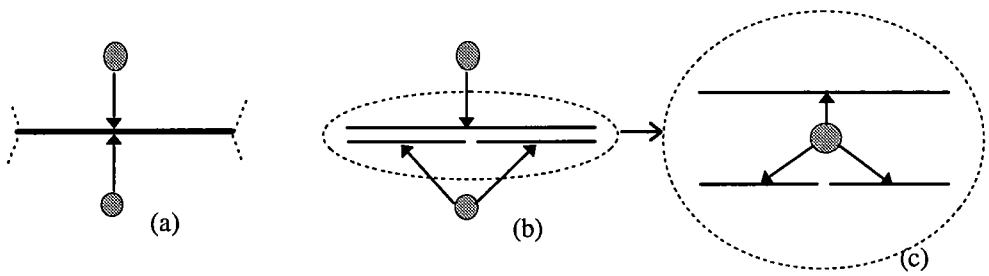

Figure 4. *A 2D example of a subdivision of a face (a) by two smaller faces (b). (c) shows the collector between the three faces.*

The *subdivide* operation covers a large face with two or more smaller faces that completely cover the larger face. Here the new region has no volume as show in Figure 4. A face can only be refined once (on the side facing the object).

Given a B-rep of an object, the LRB-rep can be constructed in one of two ways. Either face-abstractions can be applied to the original B-rep in a bottom-up approach, or face-refinements can be applied to a bounding volume in a top-down approach. The bottom-up approach is applicable for creating the LRB-rep from a given B-rep, while the top-down approach is applicable for creating random fractal objects of arbitrary complexity.

Depending on the application, some additional information can be associated with each LRB-rep face. This is to help speed up the refinement decision process. Two kinds of information possible are:

1. **Distance and direction to the solid**. Each side of a face that does not belong to the original object has an associated vector to the closest face of the original object. This information can prioritize the face refinement process.
2. **Face dimensions.** This consist of information such as face extent, and area. It can be used in rendering applications.

3. Wrappers

The LRB-rep consists of a lot of faces that by themselves are not ordered in any way. We define a *wrapper* to be a subset of the faces having a certain property and provide five useful operations. These are:

- `newWrapper: LRB-rep → Wrapper`
  Given an LRB-rep, construct a new wrapper with all the external faces of the LRB-rep.
- `refineFace: Wrapper × Face → Wrapper`
  Given that the face is not part of the original object, refine the face (along with possibly several others).
- `discardFace: Wrapper × Face → Wrapper`
  Remove the face from the wrapper.
- `isSolidFace: Wrapper × Face → Boolean`
  Return true if the face belongs to the original object.
- `wrapperFaces: Wrapper → { Face }`
  Return the set of faces currently in the wrapper.

The wrapper is what applications use to solve problems such as the two-body collision detection. With the `refineFace` and the `discardFace` operations, the initial wrapper can be refined where needed to locate a set of faces required by the application.

When considering which faces to refine in a wrapper, an application can refer to the additional information stored in the LRB-rep, that is, the distance to the solid and the face dimensions. An application can prioritize the faces and refine first the closest faces to the solid while leaving the faces farther away for later.

4. Empirical Results

We have implemented the LRB-rep data structure in C++ as part of the Proxima/Isaac projects. Although currently we cannot create the LRB-rep automatically for all objects, for certain objects that exhibit a fractal nature, algorithms can be formed that abstract the objects. Using the top-down method, we constructed several fractal objects of arbitrary complexity for which we can easily build the LRB-reps. As an example, Figure 5 shows a tetrahedral fractal refined to 34,474 faces, with an exterior wrapper of 24 faces. The total number of additional faces is 8,912, resulting in an LRB-rep of 43,396 faces and 4,463 collectors. For this object, a maximum of six face refinements are needed to resolve an exterior wrapper face to an inner boundary face. This means that given any face on the original object, that face can be reached from a face on the outer wrapper in only six face refinements.

In general, this number varies according to the local detail. Where there is a lot of detail the depth will be greater.

5. Discussion

This representation is motivated by our work in collision detection and virtual environments. In virtual environments, objects bump, slide, and maintain prolonged contact as the norm, not the exception. In general, most objects are in close proximity to other objects. When the objects are complex, contact analysis and collision detection must take time proportional to the local complexity of the contact and not the global complexity of the objects. For convex objects that remain apart, Lin and Canny's algorithm [6] provides a constant-time incremental algorithm that maintains their minimum distance. This algorithm, however, does not generalize well to nonconvex, complex objects and contact analysis. The algorithm does not provide contact region information once contact has been detected, and nonconvex objects need to be convexified. This results in an $O(n^2)$ complexity when n convex pieces result. For complex objects, n, can be prohibitively large. The development of the LRB-rep was geared specifically to address these shortcomings.

With the LRB-rep, the two body collision detection problem can be solved efficiently. This assumes only that the contact regions are localized to a small area on either object. The algorithm begins the collision test by considering the outer wrappers of both objects, properly mapped to the global frame of reference. The basic idea is this: if the wrappers intersect but can be separated (i.e., can be made not to intersect) by refining the intersecting faces, the objects do not collide. To start off, each face of one wrapper is tested for interpenetrating with the faces of the other wrapper. Although this is quadratic time, the number of faces tested is minuscule compared to the total number of faces in the actual object (for example, 24 vs. 30,000). If no faces of the initial wrappers intersect (assuming one is not contained in the other), the objects cannot be colliding. If any do, those faces are refined, and the new faces which resulted from each refinement are tested recursively.

We are currently working on several problems related to the LRB-reps; the creation of the LRB-rep using a bottom-up approach, the two-body and n-body collision detection problem, and the generalization of the LRB-rep to curved surfaces [5]. The two-body so-

lution uses the LRB-rep exclusively and we are close to finishing this. The *n*-body solution needs to combine the LRB-rep with an a spatial structure such as a BSP tree [11] to yield an incremental algorithm, therefore this is much harder problem than the two-body problem.

Acknowledgments

This work is a direct result of a group effort and we would like to acknowledge Jay Perry for generalizing Proxima's B-rep data structure to a nonmanifold representation, to N. Douglas Liun for helping in the discussions on the LRB-rep and for implementing a *n*-star object to LRB-rep construction algorithm. Haitao Jiang has been working on the *n*-body problem and has offered many helpful suggestions on the LRB-rep. We would also like to thank ONR and NIST for their continued support in our work on virtual environments.

References

[1] L. de Floriani. A variable resolution graph based model of three dimensional objects, *Advances in Engineering Software*, v. 10 n. 3, March 1988

[2] D. Waco, Y. Se Kim. Geometric Reasoning for Machining Features Using Convex Decomposition, *Proceedings of 2nd ACM Solid Modeling conference*, May 1993.

[3] T. A. Funkhouser and C. H. Sequin.Adaptive Display Algorithm for Interactive Frame Rates During *Visualization of Complex Virtual Environments. SIGGRAPH '93 Conference Proceedings*. 1993

[4] D. P. Dobkin and D. G. Kirkpatrick. Fast detection of polyhedral intersections, *Theoretical Computer Science*, vol 27, 1983.

[5] J. Peters. C^1 Surface Splines, *SIAM Journal of Numerical Analysis*. vol 1, No. 1, 1993.

[6] G. Vanecek Jr. Brep-index: A Multi-dimensional Spatial Partitioning Tree, *1st ACM/SIGGRAPH Symposium on Solid Modeling Foundations and CAD/CAM Applications*, June1991.

[7] W. J. Bouma and G. Vanecek, Jr. Modeling Contacts in a Physically Based Simulation. In *Proceedings of 2nd ACM Symosium on Solid Modeling and Applications*, pages 409-418, May 1993.

[8] M. C. Lin and J. F. Canny. A Fast Algorithm for Incremental Distance Calculation. *A Preliminary Version appeared In Proceedings of IEEE Robotics/Automation Conference 1991*, Sacramento, CA.

[9] A. Wallis and D. Lavender and A. Bowyer and J. Davenport. Computing Voronoi Diagrams of Geometric Models. *IMPA Workshop on Geometric Modeling*, 1991.

[10] C. H. Hoffmann. *Geometric and Solid Modeling: An Introduction*, Kaufman publ., 1989.

[11] B. Naylor. Binary Space Partitioning Trees as an Alternative Representation of Polytopes. *Computer-Aided Design*, 1990.