Eurographics Symposium on Geometry Processing (2003)
L. Kobbelt, P. Schröder, H. Hoppe (Editors)

CLODs: Dual Hierarchies for Multiresolution Collision Detection

Miguel A. Otaduy and Ming C. Lin

Department of Computer Science, University of North Carolina, Chapel Hill, U.S.A

Abstract

*We present "*contact levels of detail*" (CLOD), a novel concept for multiresolution collision detection. Given a polyhedral model, our algorithm automatically builds a "dual hierarchy", both a multiresolution representation of the original model and its bounding volume hierarchy for accelerating collision queries. We have proposed various error metrics, including object-space errors, velocity dependent gap, screen-space errors and their combinations. At runtime, our algorithm uses these error metrics to select the appropriate levels of detail* independently *at each potential contact location. Compared to the existing exact collision detection algorithms, we observe significant performance improvement using CLODs on some benchmarks, with little degradation in the visual rendering of simulations.*

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Hierarchy and Geometric Transformations

1. Introduction

Collision detection is an important problem in physically-based modeling, computer animation, virtual environments, electronic prototyping, robotics and many other engineering applications. Collision detection is often one of the computational bottlenecks in achieving interactive simulation for complex environments. The running time of any collision query algorithm depends on both the input size (the combinatorial complexity of the models) and the output size (the number of contact points). Despite the huge body of literature, the existing techniques cannot offer the desired performance for many real-time applications, such as haptic rendering of large structures, real-time interaction with massive CAD models, and interactive dynamic simulations of complex objects.

In this paper, we investigate the use of multiresolution techniques for collision detection. Model simplification has been an active research area for the past decade. Many efficient and effective simplification algorithms have been proposed to accelerate the visual rendering of highly complex models in real time. Applications of mesh simplification algorithms to the problem of collision detection can potentially accelerate collision queries. However, to date only relatively simple algorithms have been proposed for convex polytopes. A simple approach would be to generate a series of simplified representations, also known as levels-of-detail (LOD), and use them directly for collision detection. But, collision queries require auxiliary data structures, such as bounding volume hierarchies (BVH) or spatial partitioning, to achieve good runtime performance. These data structures often take a substantial amount of memory, while the LODs themselves can require a large amount of storage as well. To represent the auxiliary data structures for each static LOD can quickly increase the memory requirement to the point where the memory access time may dominate the overall runtime performance. On the other hand, constructing BVHs for on-the-fly simplification can be time consuming.

Main Contribution: We present the notion of "*contact levels of detail*" (CLODs) that is based on a novel data structure and algorithms for multiresolution collision detection. The data structure is a "dual hierarchy" that serves both as a BVH for accelerating collision queries and a multiresolution representation of the original model for computing contact information. We have also proposed several error metrics, including object-space errors, velocity dependent gap, screen-space errors and their combinations. At runtime, the algorithm uses these error metrics to select adaptive CLOD at each potential contact location *independently*. In addition, CLODs offer the capability to perform time-critical collision queries for real-time applications. We have tested our approach on a diverse set of benchmarks with challenging contact scenarios. We observed noticeable performance improvement with little degradation in the visual rendering of simulation results on some of the benchmarks.

Organization: Section 2 briefly surveys previous work. We present the general data structure and associated proximity query algorithm for contact levels-of-detail in section 3. Sections 4 and 5 describe a specific implementation of hierarchy construction using convex hulls and its use for runtime contact queries respectively. We discuss the implementation is-

sues and demonstrate the effectiveness of our approach in section 6.

2. Related Work

This research is built upon a large collection of knowledge in mesh simplification and collision detection. We briefly survey related work in this section.

2.1. Model Simplification

Many techniques for mesh decimation and model simplification have been proposed for the last few years. We refer readers to an excellent book on this subject[18]. There has been some growing interest in perception-based simplification for interactive rendering[17]. However, these techniques are based on different criteria than those for collision detection. Although we precompute the level of detail (LOD) hierarchy offline, the way we select the appropriate LOD on the fly is "contact dependent" at each potential contact location across the object surfaces. It shares a more common theme with view-dependent simplification[18], which uses higher resolution representations on the silhouette of the object and much coarser approximations on the rest of the object that is not as noticeable to the viewpoint.

El-Sana and Varshney[4] proposed to construct a continuous, multiresolution hierarchy of the model during preprocessing. At run time, a high-detail representation is used in the region of contact, and a coarser representation farther away. The proposed approach only applies to haptic rendering using a point probe to explore a 3D model. It does not extend naturally to collision queries between two interacting 3D objects, since multiple disjoint contacts can occur simultaneously at widely varying locations without much spatial coherence. The latter problem is the focus of our paper.

2.2. Collision Detection

Data structures based on hierarchical representations have been extensively used in the design of efficient algorithms for collision detection between geometric models (see survey[12, 14]). Examples of typical bounding volumes include spheres and axis-aligned bounding boxes, due to their simplicity in performing overlap tests between two such volumes. Other hierarchies include octrees, k-d trees, k-DOPs, OBBTrees, trees based on S-bounds, R-trees and their variants, etc.[12, 14]. Other spatial representations are BSP's and their extensions to multi-space partitions, space-time bounds or four-dimensional tests (see a brief survey[22]), and many more.

O'Sullivan and Dingliana[20] investigated LOD techniques for collision simulations and studied various factors affecting collision perception, including separation, eccentricity, causality, distractors, and accuracy of simulation results. Their work is based on the model of human visual perception and validated by psychophysical experiments. The feasibility of using these factors for scheduling interruptible collision detection among large numbers of visually homogeneous objects is also demonstrated. Instead of addressing the scheduling of multiple collision events among many objects, we focus primarily on the problem of contact queries between two highly complex objects.

Recently techniques based on GPU acceleration have also been proposed for collision queries[9, 16]. However, these approaches are not necessarily faster for rigid bodies where precomputation can be effectively carried out off-line, since the readback from framebuffer and depth buffer cannot be done fast enough.

The concept of time-critical collision detection was first introduced by Hubbard[11] using sphere-trees. Collision queries are performed as far down the sphere-trees as time allows, without traversing the entire hierarchy. The contact information was derived from two colliding bounding spheres and could deviate arbitrarily from the actual contact normals and contact locations. This idea can be applied to any type of bounding volume hierarchies (BVH). An error metric is often desirable for interactive applications to formally and rigorously quantify the amount of error introduced. However, no tight error bounds have been provided using such approaches. This can be problematic, especially when contact normals and contact points are required to compute a plausible collision response.

Some techniques limited to convex polytopes have been proposed to exploit hierarchical representations and motion coherence for fast distance computation [2, 7]. The progressive hull representation [23] also presents some favorable characteristics for collision detection purposes, but to the best of our knowledge no data structure for collision detection has been defined using this representation. In addition, these techniques suffer from a surface enlargement problem due to meeting containment conditions, that can result in notable surface deviations when being used for multiresolution collision detection. Our implementation of CLODs creates a BVH of convex hulls, taking advantage of the work by Ehmann and Lin[3]. Our work differs from theirs in the sense that we create a new multiresolution representation of the objects to be used in multiresolution collision detection.

The basic idea of CLODs has been successfully applied to haptic rendering[21]. Here we generalize CLODs by defining implementation-independent data structures and algorithms, present more diverse error metrics, and apply CLODs to rigid body dynamic simulation.

3. Contact Levels of Detail

In this section, we present our approach for constructing hierarchies of CLODs. First we analyze the advantage of using CLODs in hierarchical collision detection, then we describe the associated data structures, as well as the process for creating them, and finally we explain how CLODs are used in runtime collision queries.

3.1. Hierarchical Collision Detection

Bounding volume hierarchies (BVHs) are commonly used for collision detection between general geometric objects. To perform intersection tests, two models are queried by recursively traversing their BVHs in tandem. Each recursive step tests whether a pair of bounding volumes a and b, one from each hierarchy, overlap. If a and b do not overlap, the recursion branch is terminated. Otherwise, if they do overlap, the algorithm is applied recursively to their children. If a and b are both leaf nodes, the primitives within them are tested directly.

The test between the two BVHs can be described by the bounding volume test tree (BVTT)[13], a tree structure that holds in each node the result of the query between two BVs. When temporal coherence is present, collision tests can be accelerated by *generalized front tracking* (GFT)[3]. GFT caches the front of the BVTT, where the result of the queries switches from "true" to "false", for collision query in the next time step. The overall cost of a collision test is proportional to the number of nodes in the front of the BVTT.

When large areas of the two objects are in a close proximity, a larger portion of the BVTT front is close to the leaves, and it consists of a larger number of nodes. The size of the front also depends on the resolution at which the objects are modeled; higher resolution implies a BVTT with a greater depth. We can draw the conclusion that the cost of a collision query depends on two key factors:

- The size of the contact area
- The resolution of the models

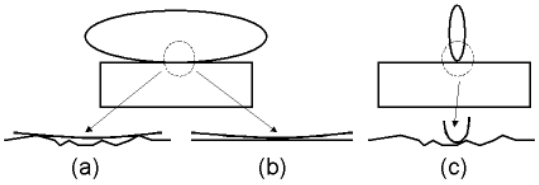

Figure 1: *a) Large contact area in high resolution; b) Large contact area in low resolution; c) Small contact area in high resolution*

The contact between two real objects typically occurs along a certain contact area. With polygonalized models, this may result in multiple contact points. The finer the resolution of the objects, the larger the number of contact points, as seen in Fig. 1. However, employing a larger resolution may have little effect on the forces computed between the objects, because these forces are computed as a sum of contact forces arising from a net of contact points. We can argue that intuitively a larger contact area allows the objects to be described at a coarser resolution. This is also supported by studies on tactual perception[21]. In this paper we exploit this hypothesis to create multiresolution representations of the objects, and use them at each contact location independently for selecting the appropriate resolution of each approximation. Using CLODs we can achieve nearly constant cost for collision queries by exploiting, among other factors, the relation between contact area and the resolution of local contact features.

3.2. Data Structure

In order to perform efficient multiresolution collision detection, we need to achieve two main objectives:

1. Create *accurate multiresolution representations*.
2. Embed the multiresolution representations in *effective bounding volume hierarchies*.

Multiresolution representations are often created by performing mesh decimation on the given polyhedral models. The difficulty arises when trying to embed these representations in BVHs. If we consider each LOD of the object as one whole model, each LOD would require a distinct BVH for collision detection. This would result in a very inefficient collision query, because the front of the BVTT would have to be updated for the BVH of each LOD. Instead, we introduce a procedure to create one unique dual hierarchy that serves as both a multiresolution representation and a BVH.

Assuming that the input model is described as a triangle mesh M_0, the data structure for CLODs is composed of:

- A sequence of LODs $\{M_0, M_1, ..., M_{n-1}\}$, where M_{i+1} is obtained by applying simplification operations and removing high resolution geometric detail from M_i.
- For each mesh M_i, a partition of the set of triangles of M_i into disjoint clusters $\{c_{i,0}, c_{i,1}, ..., c_{i,m}\}$.
- For each cluster $c_{i,j}$, a bounding volume $C_{i,j}$.
- A tree T formed by all the BVs of clusters, where BVs of clusters in M_i are children of BVs of clusters in M_{i+1}, and all the BVs except the ones corresponding to M_0 have at least one child.
- For every BV, $C_{i,j}$, the maximum directed Hausdorff distance $h(C_{i,j})$ from its descendent BVs.

The tree T of BVs, together with the Hausdorff distances, serves as the BVH for culling purposes in collision detection. Directed Hausdorff distances are necessary because in our definition of CLODs the set of BVs associated with one particular LOD may not bound the surface of previous LODs. The Hausdorff distances are used to perform conservative collision tests, as later explained in Sec. 5.1.

An additional constraint is added to the data structure, such that the coarsest LOD, M_{i-1} is partitioned into one single cluster $c_{i-1,0}$. Therefore, the root of the BVH will be the BV of the coarsest LOD. Descending to the next level of the hierarchy will yield the children BVs, whose union encloses the next LOD. At the end of the hierarchy, the leaf BVs will enclose the original surface M_0.

The process of creating the CLODs, depicted in Fig. 2, starts by grouping the triangles of the original surface into clusters. The size and properties of these clusters depend on the type of BV that is used for the BVH, and will be such that the collision query performance between two BVs is optimized. The next step is to compute the BV of each cluster.

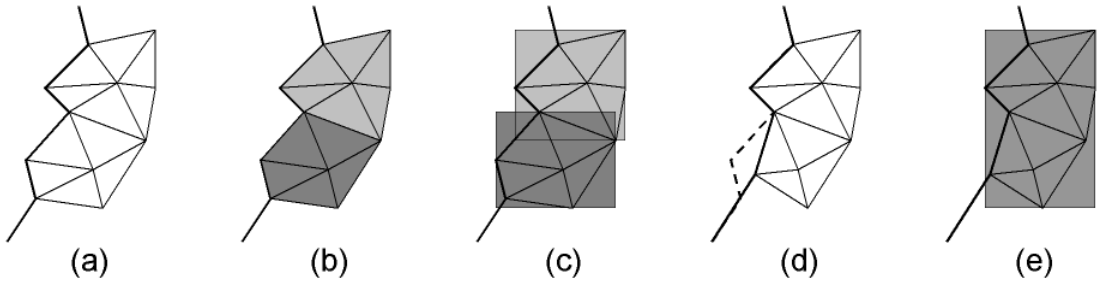

Figure 2: *Constructing Dual Hierarchies: (a) Initial surface; (b) Clusters of triangles; (c) BVs for each cluster (in this case, AABBs); (d) Mesh simplification; (e) BV of the union of clusters after some conditions are met.*

After this initialization, we start a mesh decimation process with a bottom-up construction of the BVH. This is achieved by merging clusters and computing the BV of their union.

The atomic simplification operations need to satisfy the following:

- **Constraints imposed by the BVH:** The containment of surface geometry inside the BVs has to be preserved after each simplification operation. This may impose topological and/or geometric constraints.
- **Design requirements to achieve better efficiency:** The union of clusters is possible when certain conditions are met. The BVH will be more effective in collision pruning, if these conditions are taken into account when designing the atomic simplification operations.

In Sec. 4, we present a specific implementation of CLODs that uses convex hulls as the BVs.

3.3. Runtime Queries with CLOD

Using CLODs, multiresolution collision detection can be implemented by slightly modifying the typical collision detection procedures using BVHs. For each node x of the BVTT, we perform a collision query. If the query returns "false", we do not need to descend to the children. If the query result is "true", then we perform a test for selective refinement. This test can embed various perceptual error metrics, and it determines if the resolution of x is fine enough to describe the contact information at each query location. If the refinement test returns "true", then we can directly compute contact information for x, otherwise we descend to its children in the BVTT. Descending to the children involves descending to the children BVs, as well as refining the surface representation. This approach handles the selective refinement at each query location independently in a very efficient way.

CLODs can also be used to perform time-critical collision detection[11]. We need to store the nodes of the BVTT front, and assign priorities to each node based on the refinement test.

In Sec. 5 we describe how we implement the collision queries using CLODs based on convex hulls, as well as various error metrics that we have designed.

4. Dual Hierarchies of Convex Hulls

In this section we describe a particular implementation of CLODs using BVHs of convex hulls. We first address the reasons for choosing convex hulls as the BVs, and then describe the details of constructing dual hierarchies.

4.1. Selection of the BVs

Overlap tests between convex hulls can be executed rapidly in expected constant time with motion coherence[7]. Furthermore, convex hulls provide at least equally good, if not superior, fitting to the underlying geometry as OBBs[6] or k-DOPs[12].

Most importantly the fitting property is related to the performance of the query to obtain contact information of a certain CLOD. As explained in Sec. 3.3, when the refinement test determines that a CLOD does not need to be refined, we must get contact information at that level. That implies getting contact information from the triangles of that specific CLOD. If the BVs at that level are one of AABB, OBB or k-DOPs, the efficiency of getting contact information using triangles is related to the number of triangles in each cluster. However, with convex hulls, if we ensure that the clusters are themselves convex surface patches, the contact information at the triangle level is obtained for free when performing the query between BVs[3].

The initial clusters will be defined as the surface patches of a convex surface decomposition[1, 3]. We follow the definition of convex surface patches by Ehmann and Lin [3], which imposes two types of convexity constraints on the process of creating dual hierarchies:

- **Local constraints**: the interior edges of convex patches have to remain convex after simplification operations are applied.
- **Global constraints**: the enclosing convex hulls cannot protrude the surface of the object.

However, it is important to note that convex hulls pose some limitations on the types of models that can be handled. In our implementation, the convex hulls are formed by convex surface decomposition, which requires the input models to be described as 2-manifold oriented surfaces.

4.2. Construction of the Dual Hierarchy

As already mentioned, the process is initialized by performing a convex surface decomposition of the object and computing the convex hulls of the patches. This is followed by a simplification loop represented schematically in Fig. 3.

We apply atomic simplification operations taking into account convexity constraints. After each operation, we test if the union of every pair of neighboring convex patches is a valid convex patch. If so, we merge the patches, construct the convex hull of the union, and set parent children relationships between the BVs. We have chosen to generate a new LOD every time that the number of convex patches is halved.

Due to the convexity constraints, we might reach a point where no more simplification operations are possible. We complete the construction of the hierarchy by unconstrained pairwise merging of patches[3]. The levels of the hierarchy created in this manner cannot be used as CLODs in collision queries, but are necessary to complete the BVH.

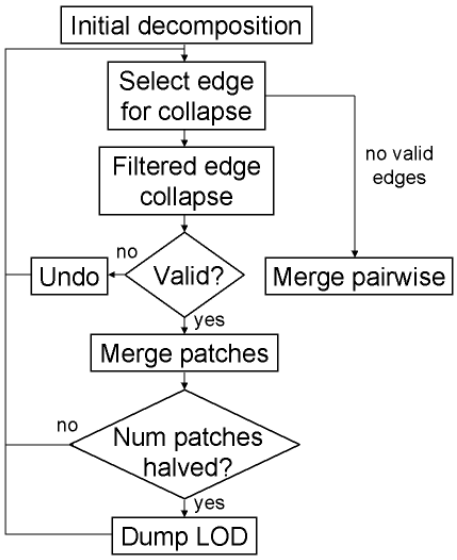

Figure 3: *Generation of CLODs*

There are various aspects of the simplification process central to our construction of CLODs that need to be defined:

- The type of simplification operation
- The assignment of priorities for simplification
- The local retriangulation after the simplification

We select edge collapse as the atomic simplification operation. There are many options for prioritizing edges and selecting the position of the resulting vertices: minimization of energy functions[10], optimization approaches[15], quadric error metrics for measuring surface deviation[5], and more. However, none of these approaches meets the convexity constraints or takes into account the factors that maximize the efficiency of CLODs. Another possibility is to employ the progressive hulls representation[23], which presents the interesting property of containment for collision detection purposes. However, in this representation successive LODs are not simply multiresolution representations of the initial mesh, because they also undergo an enlargement process which can result in noticeable visual artifacts. Our local simplification operations are inspired by multiresolution analysis and signal processing of meshes.

4.2.1. Multiresolution Analysis of Meshes

An LOD M_j of a mesh M_0 at resolution r_j is defined as an approximation of M_0 that stores all the surface detail at resolutions lower than r_j. Following this definition, we have decided to prioritize the edges to be collapsed based on their resolution. We use the same definition of edge resolution as in [21], which is inspired by the definition of resolution for functions in the 1D setting. By the same definition of LODs, the collapse of an edge at resolution r_j involves removing the geometric detail at resolutions higher than r_j. We implement this by incorporating a filtering step in the placement of vertices resulting from edge collapse operations.

We have compared this approach to other techniques for placing the vertices resulting from edge collapses, and it has proved to accelerate the process of merging convex patches. This is probably because the filtering step tends to remove concavities at high resolution, while other techniques fail to remove these concavities in favor of preserving other properties. Note that a more efficient merging of convex patches has beneficial consequences in the performance of CLODs at runtime.

4.2.2. Filtered Edge Collapse

We employ a local simplification operation **filtered edge collapse** [21] subject to local and global convexity constraints. This operation is composed of the following steps:

1. A topological edge collapse.
2. An initialization process that sets the position of the resulting vertex using quadric error metrics [5].
3. Unconstrained filtering of the position of the vertex using Guskov's minimization of second order divided differences[8].
4. Solving an optimization problem to minimize the distance of the vertex to its unconstrained position while taking into account the local convexity constraints.
5. A bisection search between the initial position of the vertex and the one that meets the local constraints, to find a location where the global convexity constraints are also met.

The filtered edge collapse operation is described in further detail in [21].

In Fig. 5 (see color plate) we show an example of a dual hierarchy of CLODs. The process is applied to the model of a wrinkled torus. Fig. 5-b shows a detailed view of the convex decomposition of the original surface. In the next CLODs, the filtering smoothens the concave areas, and the convex

patches grow forming triangle strips on the hyperbolic regions. Likewise, the simplification operations coarsen the triangulation. Also notice that initially the simplification operations are concentrated in the region where the wrinkles merge and is also the area with the finest detail.

5. Runtime Queries

In this section we discuss our implementation of collision queries using CLODs of convex hulls, present various error metrics for selecting the appropriate CLODs, and demonstrate the application to rigid body dynamic simulation.

5.1. Collision Queries between CLODs of Convex Hulls

A typical collision query between two BVs a and b determines if a and b are closer than a distance δ. The collision detection algorithms based on BVHs must ensure that if a leaf node x of the BVTT returns "true" to the collision query, then all its ancestors must return "true" too. This is usually achieved by containing the surface of each object inside the union of the BVs at every level of its BVH. In our implementation of CLODs this containment property does not hold, but we can ensure the correctness of the collision detection by modifying the collision distance δ_{ab} between two convex hulls a and b:

$$\delta_{ab} = \delta + h(a) + h(b) \tag{1}$$

where $h(a)$ and $h(b)$ are maximum directed Hausdorff distances from the descendant BVs of a and b to a and b respectively.

5.2. Error Metrics

We have designed a functional ϕ to evaluate the resulting error, when we stop a collision query at a node ab of the BVTT.

$$\phi_a = \frac{s_a}{r_a^2} \tag{2}$$

where s_a is the surface deviation of the convex hull a with respect to the original surface and r_a is its resolution. Both values are precomputed. The functional ϕ can be regarded as a measure of the volume of the fictitious features that are filtered out when using a as the CLOD.

As introduced in Sec. 3.1, larger contact areas allow the models to be described at coarser resolution. We take into account this observation by averaging the functional ϕ over an estimated contact area D. Thus, we compute a weighted surface deviation s^* as:

$$s^*_{ab} = \frac{max(\phi_a, \phi_b)}{D} \tag{3}$$

s^* can been considered as the surface deviation errors weighted by a constant that depends on both the contact area and the resolutions of local surface features. The contact area D is estimated based on precomputed per-vertex support areas (see [21] for more details).

The node of the BVTT ab is refined if s^*_{ab} is larger than a threshold value, s_0. This threshold can be determined based upon different perceptual metrics.

Size Dependant Metric

s_0 will be determined as a fixed percentage of the radii of the objects involved in the contact queries.

Velocity Dependant Metric

Set s_0 as a value proportional to the relative velocity of the colliding objects at the contact location. This is based on the observation that the gap between the objects is less noticeable as the objects move faster.

View Dependant Metric

Determine s_0 based on screen-space errors. Given N pixels of error, a distance l from the camera to the contact location, a distance n to the near plane of the view frustum, a size f of the frustum in world coordinates, and a size i of the image plane in pixels:

$$s_0 = \frac{N \cdot l \cdot f}{n \cdot i} \tag{4}$$

Selective refinement using CLODs can be implemented combining any of these error metrics. It can also support other types of perceptual error metrics[20].

Constant Frame-rate: Another important feature is the fact that CLODs can be used for time-critical collision detection. The factor $\frac{s^*}{s_0}$, computed at every potential contact location, can be used to prioritize the refinement. To achieve a guaranteed framerate for real-time applications, the algorithm will perform as many collision queries as possible, within a fixed time interval. The query event queue will be prioritized based on $\frac{s^*}{s_0}$.

5.3. Application to Rigid Body Dynamic Simulation

In rigid body dynamic simulations, many of the factors involved in the selection of CLODs, such as the velocity of the objects, the contact area and the distance to the camera, will vary dynamically. This results in transitions in the refinement tests and thus switching between selected CLODs. Rigid body dynamic simulations often require the execution of time-stepping algorithms to search for the time instants prior to collision events. With CLODs the result of the collision queries may vary depending on the selected resolution, therefore special treatment is necessary so that switching CLODs do not generate inconsistencies or deadlock situations in the time-stepping algorithms. (The analogy in visual rendering is the “popping” effects often seen during LOD switching.)

Given a node x_i of the BVTT, with collision query “false” at times t_i and t_{i+1} of the simulation, and a node x_j, child of x_i, with collision query “true” at both time instants, if the refinement test of x_i switches from “false” to “true” at $t \in [t_i, t_{i+1}]$, the time stepping method will encounter an inconsistency. It will try to search for a non-existent collision event in the interval $[t_i, t_{i+1}]$. We solve this problem by estimating a collision time t_c interpolating the separation distance of the node x_i at t_i and the penetration depth of the node x_j at t_{i+1}. We apply the contact response and restart the numerical integration from t_c with x_i as the active CLOD.

6. Implementation and Performance

In this section we describe the implementation issues and the benchmarks used to analyze the performance and effectiveness of CLODs.

6.1. Benchmark Models

Fig. 6 in color plate shows the set of benchmark models. They are:

- A wrinkled torus falling along a spiral peg
- A spoon sliding inside a cup
- A soup of numbers settling in a bowl

In Table 1 we present the statistics of the dual hierarchies created to represent the benchmark models. Note that in all cases the models are simplified to a coarsest CLOD with a complexity in the order of 1000 triangles and 100 convex pieces. This limitation is imposed by the topology of the objects and the localized influence of the simplification operations. The statistics of the *numbers* represent an average of all the models from *number* 0 to *number* 9.

Models	Cup	Spoon	Spiral	Torus	Bowl	Numbers
Tris_{orig}	64000	86016	32160	32000	12288	1080
BVs_{orig}	15490	16125	6448	5919	4336	161
Tris_{simp}	1532	1074	716	644	634	306
BVs_{simp}	241	61	100	92	63	37
CLODs	7	9	7	7	8	3
LODs	14	14	13	13	13	9

Table 1: ***Models and Associated Hierarchies.*** The number of triangles (Tris_{orig}) and convex patches (BVs_{orig}) of the initial mesh of the models; the number of triangles (Tris_{simp}) and convex patches (BVs_{simp}) of the coarsest valid CLOD; and CLODs and total number of LODs in the hierarchies.

6.2. Runtime Performance

The rigid body simulations captured in the snapshots in Fig. 6 on the color plate were computed using impulsive methods[19]. These simulations were performed on a Pentium-4 2.4Ghz processor PC with 2.0GB of memory, a NVidia GeForce-FX graphics card and Windows2000 OS.

The timing profile of Fig. 4 shows query times and the number of nodes in the BVTT front for a simulation of the spoon sliding inside the cup. We have compared results obtained with an exact collision detection algorithm[3] against the results of our CLOD technique using the size dependant error metric. In particular, we have used values of s_0 of 3.5%, 0.35%, and 0.035% of the radius of the cup. As the timings show, CLODs with $s_0 = 3.5\%$ perform at least as good as the exact algorithm for most of the simulation duration. During several time intervals, we observe a performance gain of almost one order of magnitude. In the other two benchmarks, we have not observed as significant performance gains using CLODs. Thus, their timing profiles are omitted here, given the page limit.

We have also evaluated CLODs with velocity and view dependant error metrics. In Fig. 7 (see color plate) coarse CLODs are selected when the spoon hits the bottom of the cup, and fine resolution CLODs are selected when the spoon slides along the side of the cup. In the first case the polygon count of the representations is roughly 10 times larger than in the second case.

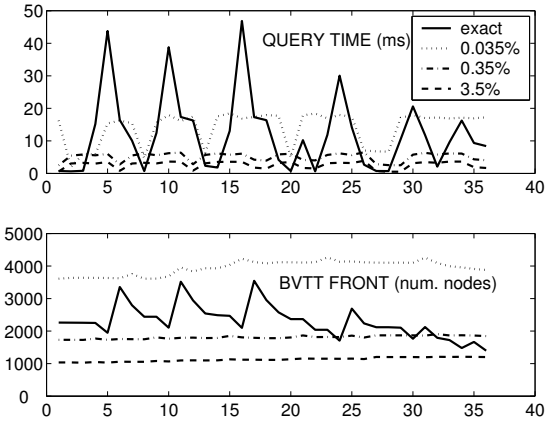

Figure 4: *Comparison of query time and size of the BVTT front for exact full res. collision detection and CLODs*

6.3. Limitations and Analysis

We have chosen a diverse set of benchmarks with varying model complexity and observed the performance of CLODs on different simulation scenarios. As a result of the modification to the collision tolerance introduced in Sec. 5.1, the multiresolution collision detection algorithm using CLODs cannot prune as efficiently as the exact algorithm[3], when the objects are separated by a distance notably larger than δ. However, when they come close to contact, the temporal coherence in the front of the BVTT is much higher with CLODs than with the exact algorithm. This can be verified in Fig. 4, where the number of nodes does not vary much with CLODs. This also explains the spikes present in the timings for the exact algorithm. These spikes take place at the instants when the objects are about to interpenetrate. Even if the number of nodes in the front of the BVTT is smaller for the exact algorithm than for CLODs with $s_0 = 0.035\%$, the coherence is lower, because suddenly many nodes in the BVTT come closer than δ, thus yielding longer query times.

We believe that the insignificant performance gain in the simulation of the torus falling along the spiral is due to similar reasons. Because of the high traveling speed and the impacts of the falling torus, the simulated objects are rarely in close proximity or have low spatial coherence. As for the simulation of the numbers settling in a bowl, the polygonal complexity of the models of the numbers is not high enough to benefit from CLODs.

To summarize, we can conclude that CLODs perform remarkably better than exact collision detection algorithms (up to one order of magnitude) for situations of sliding or resting contacts between complex models. These are in fact some of the most challenging contact scenarios, because large areas of the objects are in parallel close proximity[6].

7. Summary and Future Work

We have introduced the concept of "contact levels of detail" (CLOD) for multiresolution collision queries. CLODs are based on a dual hierarchy that serves as both a multiresolution representation of the input model and its bounding volume hierarchy for collision queries. The runtime algorithm dynamically selects adaptive CLODs at each potential contact independently using various error metrics. This approach considerably speeds up the overall performance of collision queries in complex environments with challenging contact scenarios.

This research may be extended in several directions:

- A further investigation of the problem of switching CLODs and its implications on the contact response models of various simulation methods;
- Optimization of the simplification process to obtain more effective hierarchies, perhaps incorporating an operation with a more global support than the current ones;
- The implementation of CLODs with other types of BVHs.

Acknowledgments

This research is supported in part by a fellowship of the Government of the Basque Country, National Science Foundation, Office of Naval Research, U.S. Army Research Office, and Intel Corporation. We would like to thank Stephen Ehmann and Young Kim for their help on integrating SWIFT++ and DEEP with our collision detection. We are also grateful to Dinesh Manocha, Jack Snoeyink, Mark Foskey, and the anonymous reviewers for their feedback on the earlier drafts of this paper.

References

1. B. Chazelle, D. Dobkin, N. Shouraboura and A. Tal Strategies for polyhedral surface decomposition: An experimental study. *Computational Geometry: Theory and Applications 7*, 327–342, 1997.
2. S. Ehmann and M. C. Lin. Accelerated proximity queries between convex polyhedra using multi-level voronoi marching. *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2000.
3. S. Ehmann and M. C. Lin. Accurate and fast proximity queries between polyhedra using convex surface decomposition. *Proc. of Eurographics*, 20(3), 2001.
4. J. El-Sana and A. Varshney. Continuously-adaptive haptic rendering. *Virtual Environments 2000*, pp. 135–144, 2000.
5. M. Garland and P. S. Heckbert. Surface simplification using quadric error metrics. In *Proc. of ACM SIGGRAPH*, pp. 209–216, 1997.
6. S. Gottschalk, M. Lin, and D. Manocha. OBB-Tree: A hierarchical structure for rapid interference detection. *Proc. of ACM SIGGRAPH*, pp. 171–180, 1996.
7. L. Guibas, D. Hsu, and L. Zhang. *H-Walk*: Hierarchical distance computation for moving convex bodies. *Proc. of ACM Symposium on Computational Geometry*, 1999.
8. I. Guskov, W. Sweldens, and P. Schroder. Multiresolution signal processing for meshes. *Proc. of ACM SIGGRAPH*, pp. 325 – 334, 1999.
9. K. Hoff, A. Zaferakis, M. Lin, and D. Manocha. Fast and simple geometric proximity queries using graphics hardware. *Proc. of ACM Symposium on Interactive 3D Graphics*, 2001.
10. H. Hoppe. Progressive meshes. *Proc. of ACM SIGGRAPH*, pp. 99–108, 1996.
11. P. Hubbard. *Collision Detection for Interactive Graphics Applications*. PhD thesis, Brown University, 1994.
12. J. Klosowski, M. Held, J.S.B. Mitchell, H. Sowizral, and K. Zikan. Efficient collision detection using bounding volume hierarchies of k-dops. *IEEE Trans. on Visualization and Computer Graphics*, 4(1): 21–37, 1998.
13. E. Larsen, S. Gottschalk, M. Lin, and D. Manocha. Distance queries with rectangular swept sphere volumes. *Proc. of IEEE Int. Conference on Robotics and Automation*, 2000.
14. M. Lin and S. Gottschalk. Collision detection between geometric models: A survey. *Proc. of IMA Conference on Mathematics of Surfaces*, 1998.
15. P. Lindstrom and G. Turk. Fast and memory efficient polygonal simplification. *Proc. of IEEE Visualization*, pp. 279–286, 1998.
16. J. C. Lombardo, M.-P. Cani, and F. Neyret. Real-time collision detection for virtual surgery. *Proc. of Computer Animation*, 1999.
17. D. Luebke and B. Hallen. Perceptually driven simplification for interactive rendering. *Rendering Techniques; Springer-Verlag*, 2001.
18. D. Luebke, M. Reddy, J. Cohen, A. Varshney, B. Watson, and R. Huebner. *Level of Detail for 3D Graphics*. Morgan-Kaufmann, 2002.
19. B. Mirtich and J. Canny. Impulse-based simulation of rigid bodies. In *Proc. of ACM Interactive 3D Graphics*, 1995.
20. C. O'Sullivan and C. Dingliana. Collisions and perception. *ACM Trans. on Graphics*, 20(3): pp. 151–168, 2001.
21. M. A. Otaduy and M. C. Lin. Sensation preserving simplification for haptic rendering. *Proc. of ACM SIGGRAPH*, 2003. (To appear).
22. S. Redon, A. Kheddar, and S. Coquillart. Fast continuous collision detection between rigid bodies. *Proc. of Eurographics*, 2002.
23. P. V. Sander, X. Gu, S. J. Gortler, H. Hoppe and J. Snyder. Silhouette clipping. *Proc. of ACM SIGGRAPH*, 2000.

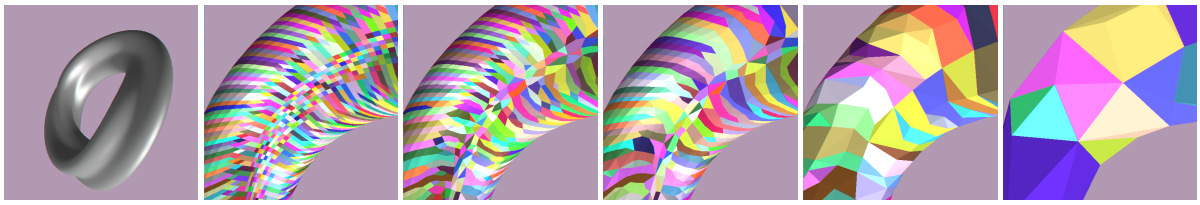

Figure 5: *From left to right, original mesh and detail view of $CLOD_0$, $CLOD_1$, $CLOD_2$, $CLOD_4$ and $CLOD_6$. Mesh simplification is combined with the construction of the BVH. Different colors correspond to different convex patches.*

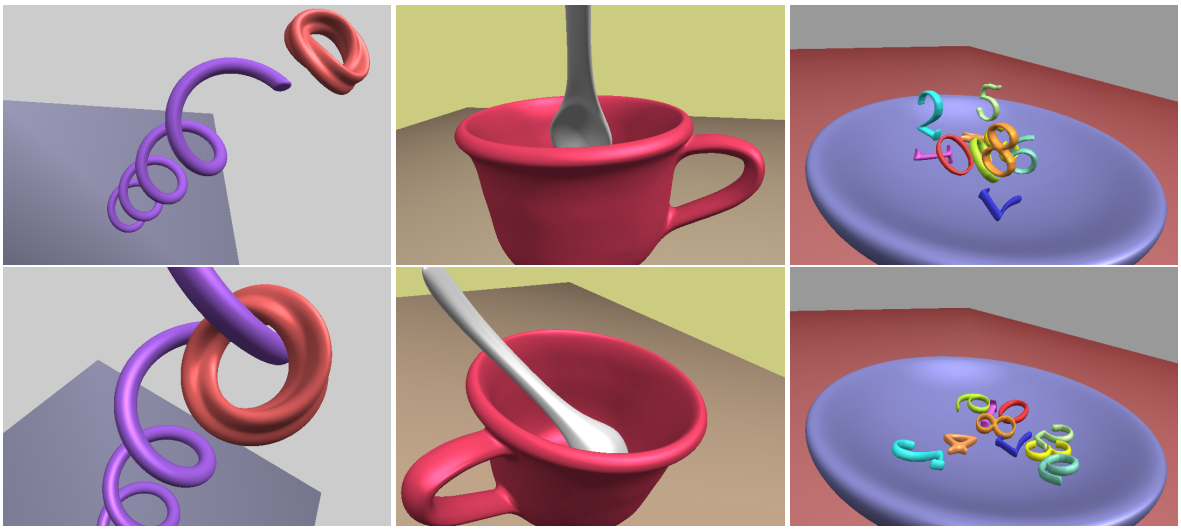

Figure 6: *Snapshots of our benchmark simulations. From left to right: torus falling along a spiral peg, a spoon in a cup, and a soup of numbers in a bowl.*

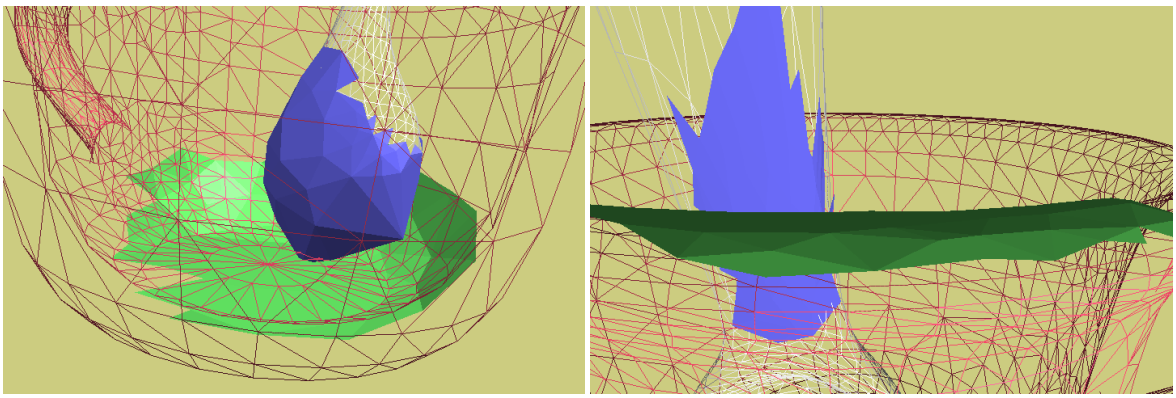

Figure 7: *Velocity dependent adaptive selection of CLODs used in detecting contact between a cup and a spoon. In blue and green, the vicinity of the contact locations shown at the resolution of the adaptively selected CLODs. In the right, coarse resolution is selected when the spoon falls inside the cup. In the left, fine resolution is selected when the spoon slides slowly along the side of the cup.*