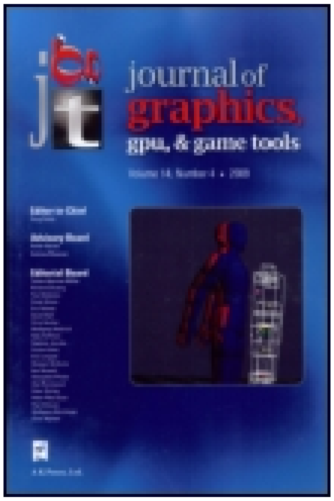

Journal of Graphics, GPU, and Game Tools
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/ujgt20

A Voxel-Based Approach to Approximate Collision Handling

John Dingliana [a] & Carol O'Sullivan [a]
[a] Trinity College Dublin
Published online: 30 Jan 2011.

Vol. 10, No. 4: 33–48

A Voxel-Based Approach to Approximate Collision Handling

John Dingliana and Carol O'Sullivan
Trinity College Dublin

Abstract. We present an approach for approximate collision handling which uses probabilistic information for improving the calculated collision response between objects in real-time physically based animation. The system uses an interruptible collision detection mechanism, which checks for intersections between successive levels of a bounding volume hierarchy (BVH). Interruptible collision detection approaches allow us to guarantee target frame rates and are often called on to make an approximation of contact points in order to deliver a suitable collision response within a given time schedule. In our system we use a BVH based on voxels and use the voxel occupancy information as well as the bounding volume interpenetration depth to improve the approximation of contact points when the collision handling stage is interrupted. We also present an evaluation of the plausibility of animations resulting from our approach.

1. Introduction

Fully accurate physics remains an unrealistic goal to strive for in interactive animation. Even the most detailed models of physics used in offline animation are based on assumptions and empirical approximations of real-world laws and physical constants [Chatterjee and Ruina 98]. Furthermore, although many techniques exist for highly accurate physical simulation, most of them still cannot guarantee a target frame rate for arbitrarily complex simulation scenarios.

1086-7651/05 $0.50 per page

Therefore, interactive animation, with very tight time constraints invariably requires that compromises be made in order to deliver physical simulations at real-time rates. A constructive goal therefore is to work within the limits of each target system, to achieve the optimal level of physical plausibility in the resulting simulation. Such a goal is made possible by the use of adaptive techniques which trade off accuracy for processing time whenever required.

In this article we will present a new approach to adaptive collision handling, which uses probabilistic analysis in the attempt to optimize the plausibility of adaptive simulations. The system uses a bounding volume hierarchy (BVH) which stores density information for each bounding volume node. When collision detection is interrupted and an approximated collision response is required based on the BVH intersection data alone, this density value is used to perturb the resulting approximated contact information.

2. Background

In dynamic simulation, it is prudent, wherever possible, to optimize computations and to make strategic simplifications in different parts of the system in order to achieve interactive frame rates. Fortunately a certain degree of simplification, particularly in dynamics, goes unnoticed by most human viewers and this makes it possible in some cases to tweak an animation to reach some desired final state without adversely affecting the believability of the animation to the viewer [Chenney and Forsyth 00, O'Sullivan and Dingliana 01]. In a similar way, Simulation Level of Detail (SLOD) approaches, less concerned with any desired final state, exploit uncertainty in order to make simplifications to the dynamic simulation in order to guarantee a target frame rate, whilst preserving the ongoing plausibility of the animated scene as much as possible [O'Brien et al. 02, Carlson and Hodgins 97].

Specifically due to the computational expense of the collision detection process, it is often not feasible to use arbitrarily complex models in real-time simulation. In fact, it is frequently the case that we can achieve adequate results by using *proxy models* for the purposes of dynamic simulation [Chenney et al. 01], which are much simpler than the models used in the rendering and visualization of the objects.

One class of modeling data structure that is particularly useful as a proxy model is the bounding volume hierarchy, which is a union of several geometrically simpler volumes (nodes), constructed in order to represent a more complex object at different levels of detail. In BVH collision detection, we usually ensure that the volume nodes are conservative overestimates of the object that is being represented and each level of the hierarchy represents a level of resolution of the physical representation of the object. Then, starting at the coarsest level (or root) of each object in the scene, we check to see if any collisions have occurred between bounding nodes and proceed to check the finer levels of detail (i.e., the child nodes) for intersection only if their parents have collided. This BVH *traversal* is used in order to cull expensive intersection computations and localize the points on the actual object that need to be tested for intersection. Commonly used BVHs include sphere trees [Hubbard 96], axis-aligned bounding boxes (AABB-tree) [van den Bergen 97], oriented bounding boxes (OBB-tree) [Gottschalk et al. 96], k-dops [Klosowski et al. 98], and shell trees [Krishnan et al. 97].

In a *time-critical* collision detection system [Hubbard 96], the BVH traversal for potentially colliding objects can be interrupted before it has been determined for certain whether or not a collision has occurred. In order to compute physically based collision response in such a case, the potentially colliding objects need to be treated as if they were colliding and an *approximate response* needs to be calculated [Dingliana and O'Sullivan 00]. Frequently, this method of collision detection leads to large numbers of individual contact points being detected between two or more colliding objects, meaning that for the full collision response, multiple contacts need to be reduced by averaging normal directions or through an expensive computational process, such as by solving a complex Linear Complementarity Problem for the system [Giang et al. 03]. In this article, our primary goal will be to improve the quality of the contact approximation without requiring the resolution of a prohibitively large simultaneous contact problem.

3. Voxel-Based Contact Levels of Detail

In this section we present our approach for delivering refinable contact levels of detail using a voxel-based approach. It should be noted that, although the discussion centers around an octree-related BVH data structure used in our specific implementation, the approach applies equally well to a broad range of BVH schemes with very little modification to the implementation of the algorithm, provided we can enforce the constraint that nonoverlapping sets of bounding volume nodes are used for each level in the hierarchy.

3.1. The Volume Representation

The bounding volume hierarchy used in this article is one made up of uniformly sized regular cubes, which we refer to henceforth as a *VoxTree*. The individual volume nodes at each level of detail (i.e., at each level in the hierarchy) are all of uniform size and all cubes in the tree share the same orientation (see Figure 1), although when it comes to object-object intersection tests, nodes on the BVH of individual objects will not be axis-aligned with

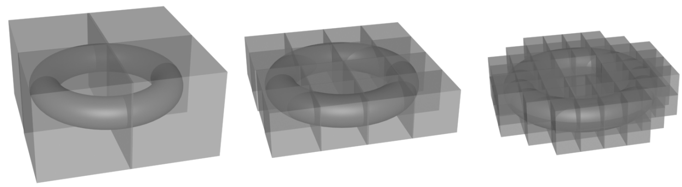

Figure 1. Example VoxTrees for a simple geometric shape.

each other, as they would be in an AABB-tree collision scheme. An octree would be a specific instance of a VoxTree, which only differs in that there is no limit on the branching factor in the hierarchy. Having uniform sized nodes in each level limits the efficiency of the tree to a certain degree but is a key requirement in probabilistic collision response as we will see later. On the other hand, the hierarchy is fully compatible with an interruptible collision detection system as described in [Hubbard 96].

At the finest level of the BVH the data structure is basically a voxel array and our system can accept a volume representation as the primary definition of the object. Alternatively if the original model is polygonal, a built-in voxelizer extracts the volume representation and calculates voxel density values. Note that it is also possible to use the polygonal representation itself for a final pass of highly accurate polygon intersection tests if time remains in the frame for collision processing. In either case, we group nodes recursively up the tree beginning at the voxel representation and at each stage we store the bounding node's *occupancy* (or density) value, which is a value between 0 and 1 indicating the fraction of the node that is occupied by the underlying object.

3.2. Contact Modeling and Collision Response

In physically based animation, the collision response mechanism is required to calculate a change of state for the colliding objects based on laws of dynamics and the input from the collision detection phase. We model all of our colliding entities as perfectly rigid; deformations during collision are infinitesimal and change of state is calculated based on instantaneous impulses [Witkin et al. 01, Mirtich 96].

The primary calculation involves determining the scalar j that represents the magnitude of the impulse along the collision normal $\hat{\mathbf{n}}$ at a position $\mathbf{p}$. In a rigid body dynamic simulation, this two-vector couple $< \hat{\mathbf{n}}, \mathbf{p} >$ encapsulates the two main variables that the contact modeling mechanism has to determine and pass on to the collision response mechanism. We will refer to this two-vector couple henceforth as a *contact primitive*. Note that $\hat{\mathbf{n}}$ is the direction of the impulse vector in a frictionless collision and is an approximation of the common normal to the two surfaces in contact.

A contact primitive needs to be generated for each individual contact detected between objects in the simulation and in our VoxTree, an approximation of $\hat{\mathbf{n}}$ can be obtained by taking the vector along a line drawn through the center of two colliding nodes, while $\mathbf{p}$ is a point that divides this line in proportions equal to the radius of the two nodes. Note that this is similar to the approach taken in [Dingliana and O'Sullivan 00] to compute the contact primitive for a pair of colliding spheres. Even though our approach uses cubes, we still base the contact primitive on the centers and the sizes of the two colliding nodes.

3.3. Collision Probability

Further optimization is made possible when we consider the fact that volume nodes are inexact representations of the underlying object. Not only do the nodes approximate the underlying object but, very often, they do so inconsistently. In other words, we should consider a volume node's occupancy value, which we call ρ_i. We will find that there can be significant variance in the values of $\rho_1, \rho_2, \rho_3, \ldots$ even though they might be sibling nodes in a homogeneous tree (see Figure 2). This in itself is not unobvious as volume rendering techniques have used node density for rendering voxelized data structures, but in our approach we use the density values in calculating the dynamic behavior of the object.

We highlight the concept that, when an intersection is detected between volume nodes, this only signifies a *possible collision* between those two objects at the locations represented by those nodes. In related previous work, *the surface crossing probability* for a colliding node in an octree has been used to determine the likelihood that the collision with a volume node actually signifies an interpenetration of the part of the object represented by that node [He and Kaufmann 97]. However the primary goal of using probabilistic measures, to date, has been to optimize the collision detection phase and not for

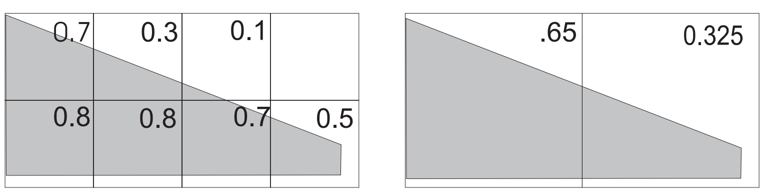

Figure 2. Bounding node occupancy.

synthesizing physically based collision response. In our approach we actually use probabilistic values to refine the approximation of collision response in interruptible collision handling.

Strictly speaking, the probability that a volume intersection actually signifies a collision between the bounded objects is proportional to the occupancies of the two intersecting volume nodes (ρ_a and ρ_b, respectively) and to the degree of interpenetration between the two nodes, κ. Thus,

$$P_{colliding} = \kappa \times \rho_a \times \rho_b. \tag{1}$$

κ here is a value between 0 and 1 which signifies the degree of interpenetration that has occurred between two nodes as a fraction of the maximum interpenetration possible. Equation (1) is in fact simply a generalization of previous BVH collision detection approaches, which can be said to always assume $\rho = 1$ and $\kappa = 1$. We will see precisely how the probability value $P_{colliding}$ is used in Section 4.2, but first we should discuss how the value κ is calculated in practice.

3.4. Measuring Interpenetration

An actual measure of interpenetration, let us call it k, should ideally be obtained by calculating the volume of the interpenetration region (we will call this the *volume of intersection*) between the two volume nodes and comparing this with the volumes of the nodes themselves. As shown in Equation (2), we take k to be the ratio of the volume of intersection to the volume of the smaller of the two interpenetrating nodes (V_a and V_b). This is illustrated in Figure 3. Thus,

$$k = \frac{V_a \bigcap V_b}{min(V_a, V_b)}. \tag{2}$$

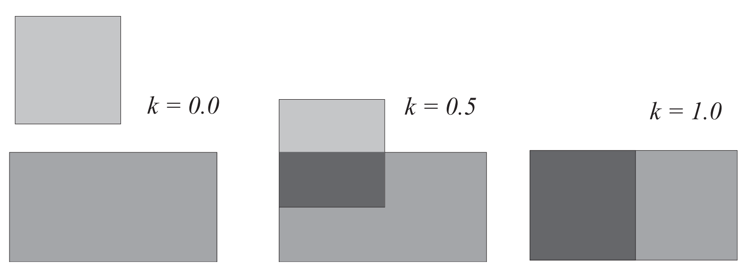

Figure 3. Interpenetration ratio.

However, calculating the volume of intersection is an expensive process, particularly when we consider that the intersecting bounding volume nodes might be spheres, cubes, or other more complicated volumes.

Ideally, we would like a measure that relates more to a distance. If we can assume that most BVH nodes will be regular solids, whose volumes are roughly proportional to the cube of their radii, and extend this assumption also to the volume of intersection, which will have an effective radius $\frac{1}{2}d_K$ (the "diameter" of interpenetration), then we have

$$k \propto \frac{(\frac{1}{2}d_K)^3}{(min(R_a, R_b))^3}, \tag{3}$$

where (see Figure 4):

$$d_K = R_a + R_b - d. \tag{4}$$

In practice, we can safely remove the cubic powers in Equation (3) without any serious consequences, since all that we require is a scalar value that will reasonably allow us to quantitatively compare two different levels of interpenetration, and that converges correctly to the limits of 0 and 1 for "no interpenetration" and "full interpenetration," respectively. We then get a measure for interpenetration κ, that is considerably quicker to compute:

$$\kappa = \frac{d_K}{2(min(R_a, R_b))}. \tag{5}$$

The assumption that a spherical approximation is representative of different BVH node types may seem rather bold. However, it is adequately close for volume nodes of regular dimensions, such as the regular cubes in the VoxTree. Furthermore, the result converges to the accurate result as we traverse further down the BVH and deal with increasingly smaller-sized volume nodes.

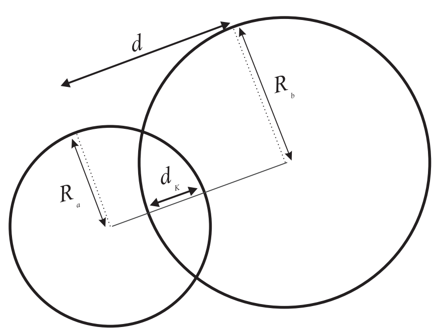

Figure 4. Calculation of diameter of interpenetration d_k.

3.5. Normals from Density Gradients

In volume graphics, it is a common approach to derive normals from voxels by calculating the density gradient for different voxel samples [Levoy 88]:

$$\hat{\mathbf{n}}(x, y, z) = \frac{\nabla \boldsymbol{\rho}(x, y, z)}{|\nabla \boldsymbol{\rho}(x, y, z)|}. \tag{6}$$

A similar approach can be taken for volumetric collisions. The occupancy value for each volumetric node is equivalent to the density of the volume node and a similar approach to that used in volumetric rendering can be used to calculate an approximate normal $\hat{\mathbf{n}}(x, y, z)$ at any point, based on the occupancy gradient.

In the haptics literature, it has been proposed that a voxel representation of an object surface can be used to generate contact data for a 6DOF haptic device [McNeely et al. 99] and this is a very similar result to what we require for collision response in dynamic simulation. However, in such an approach, the contact model basically represents the collision between a single point (representing the position of the top of the haptic controller) and the object being tested and is not directly mappable to a collision response resulting from a larger, more complex, contact manifold between two simulation proxies.

Nevertheless, the concept of using a density metric in normal calculations is a sound one. We simply need to take into account that, for collisions, it is important not only to consider the individual objects but also the objects that they are colliding with. Also, it is important that we consider the spatial properties of both of the individual colliding objects at the time of collision (i.e., their positions and orientations).

4. Compressing Contact Data

In hierarchical collision detection, contact primitives tend to be more numerous at the finer levels of detail than in accurate collision detection. For instance, if we take the example of two planes in contact, this may be dealt with as a single contact primitive by an accurate collision detection algorithm, i.e., a face-face collision (see Figure 5), but may result in multiple contact primitives in a BVH approach. When the branching factor of the BVH is significant, it becomes prohibitively expensive to calculate response on the large numbers of primitives that are output by the contact modeler using a mathematical simultaneous contact solution. Thus, we need to reduce the number of contact primitives passed to the collision response module.

A naive solution would be to interrupt the contact modeling phase when we know we have gathered as much data as the collision response module will

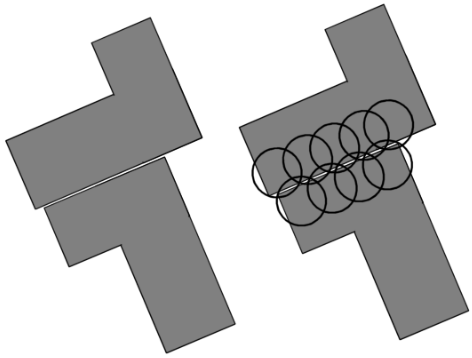

Figure 5. A single face-face collision may sometimes be detected as multiple BVH node collisions.

be able to handle. However, as the bottleneck, in this case, is in the response model and contact modeling can in practice deliver much more than collision response can handle, it would be desirable if we could somehow exploit the extra contact modeling data to generate a more accurate collision response without having to solve the expensive simultaneous contact response.

So essentially the collision response process needs to be preceded by a *contact point reduction* phase which attempts to generate a reduced number of normals that, when passed to collision response, will achieve close to the the same result cheaply. Heuristic contact modeling methods, such as the ones we will discuss in following sections, can be used to perform this reduction and enable a more optimized collision response output.

4.1. Simple Averaged Normals

For nonconcave pairs of objects where many adjacent nodes of similar orientation are colliding (e.g., the nodes of two almost-parallel colliding faces), an effective solution is to simply average the normals returned by contact modeling as shown in Equations (7) and (8). We can assume that this is the only viable alternative to solving a prohibitively large LCP in previously documented collision handling approaches of this kind [Dingliana and O'Sullivan 00].

Note that $\hat{\mathbf{n}}_j$ and $\mathbf{p}_j$ here are collision directions and collision points, respectively, for individual node intersections between the BVH trees at the current levels of traversal. Thus,

$$\mathbf{p}_{ave} = \frac{\sum \mathbf{p}_j}{N}, \tag{7}$$

$$\hat{\mathbf{n}}_{ave} = \frac{\sum \hat{\mathbf{n}}_j}{|\sum \hat{\mathbf{n}}_j|}. \tag{8}$$

Simple averaging is adequate when the surfaces are nonconcave and the multiple points of contact are close together, such as when they are derived from collisions involving adjacent volume nodes on the same object. Giang [Giang 04] uses this kind of method and discusses a clever extension to this approach which works by examining a cloud of contact points at runtime and finding a reduced set of contact primitives for a specific contact manifold.

4.2. Weighting by Interpenetration and Density

In our system, we not only use the node occupancy but also the collision probability, P in Equation (1), as the factor for perturbing the collision normals obtained from direct normal approximation. We take into account how much individual nodes have interpenetrated, as well as the node occupancy, and use the collision probability to get a weighted average of a group of contact primitives:

$$\hat{\mathbf{n}}_A = \frac{\sum P_i \hat{\mathbf{n}}_i}{|\sum P_i \hat{\mathbf{n}}_i|}, \tag{9}$$

$$\mathbf{p}_A = \mathbf{p}_{ave} + \frac{\sum P_i(\mathbf{p}_i - \mathbf{p}_{ave})}{\sum P_i|\mathbf{p}_i - \mathbf{p}_{ave}|}. \tag{10}$$

4.3. Grouping

In all cases of averaging or weighted averaging, there is a marginally increased probability of returning erroneous results when the contact manifold is very concave or if there are disjoint groups of colliding nodes within the area that is being examined. Indeed, in most cases involving contacts between nonconvex objects, it would not be correct to apply the the weighted averages in Equations (9) and (10) to all detected contact primitives. Instead, we only apply the average to localized groups of contact primitives to get several reduced averages for a simpler simultaneous resolution. We therefore should ensure, if normals are to be reduced correctly with our technique, that we identify the groups of primitives for which it is safe to do so.

An effective solution is to pregroup BVH nodes in the tree at the preprocessing stage. When the tree is initially generated, we store information in the node data-structures about which polygonal primitive they should be associ-

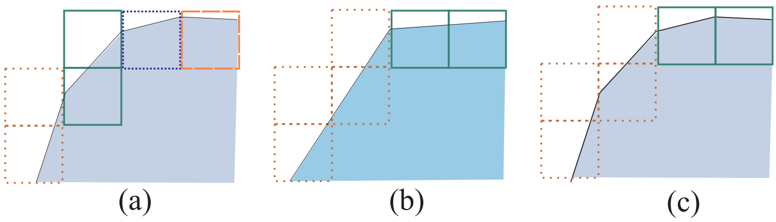

Figure 6. Assigning groupings based on a reduced mesh.

ated with. When we find, in the collision detection phase, that the collision involves nodes of the same group, this indicates that it is safe to average them.

If our input model is highly tessellated, we can pregroup nodes belonging to polygons with similar orientations or we can apply mesh reduction before the grouping phase to associate nodes belonging to the same "almost-planar" part of the object surface. Figure 6(a) shows grouping of surface nodes for a bounding hierarchy, color coded by group. Figure 6(b) shows groupings for a reduced mesh and 6(c) shows the reduced grouping mapped back on to the high-resolution mesh. If our original model is provided as volume data we determine regions where it is safe to group nodes by examining the surface normal calculated according to Equation (6).

In our implementation, we used OpenGL selection in a preprocessing phase, to identify which triangles of a reduced mesh are bounded by each volume node. We were then able to simply assign safe groups based on which volume nodes shared the same triangles. Where a node bounds multiple triangles we can assign it to multiple safe groups. This approach was chosen for ease of implementation and overall is probably not the most efficient solution, so we do not document it in detail here. However, it was adequate for our purposes as this grouping needs only be performed at the preprocessing phase for rigid objects and does not affect runtime performance.

5. Results and Discussion

The evaluation of the plausibility of dynamic simulations is a relatively new concept. Most previous studies have attempted only to compare different dynamic simulation systems based on their efficiency. Recently, the value of having a measure of the perceptual plausibility of a simulation has become an important issue [O'Sullivan and Dingliana 01] and actual metrics have been proposed for quantifying the plausibility of a physically based animation [O'Sullivan et al. 03]. In this section we provide a brief evaluation of our system and a discussion of the overall applicability of the technique.

5.1. Simulation Levels of Detail Comparison

O'Sullivan et al. proposed certain metrics for the evaluation of physically based animations [O'Sullivan et al. 03] and, in a specific case study, they demonstrated how this could be used to evaluate the plausibility of different simulation levels of detail in a sphere-tree–based dynamic simulator.

To illustrate the effectiveness of the speed-accuracy trade-off in the system, we follow the method used in this case study and manually clamp the recursive refinement of the contact modeling process at different levels of the VoxTree for a large random sample of contact cases. We then compare the resulting approximate contact data with precomputed values calculated from an "exact" collision detection method, calculating an error value for resulting contact primitives. We also do a comparison of our approach with a standard approximate contact modeling mechanism which simply averages multiple contacts without taking node occupancy and intersection depth into account. Data is collected for 1,000 collisions and a resulting error value is computed for each SLOD. Spheres and blocks were chosen for this test as this made it easier to analytically compute an exact response result with which the approximate method could be compared.

Figure 7(a) shows the average error in the computed collision normal $\Delta\hat{\mathbf{n}}$ and contact point $\Delta\mathbf{p}$ for each Contact Level of Detail. The collision normal error $\Delta\hat{\mathbf{n}}$ is simply the difference in the angle in degrees from the exact solution, and $\Delta\mathbf{p}$ is the relative distance of the approximated contact point from the contact point calculated by the exact solution. Figure 7(b) shows a graph of the product of these two error values to illustrate the overall error in the contact model. As expected, the results show that the resulting contact model statistically converges towards the exact solution at increasing levels of refinement. We also see that our approach performs relatively better, particularly in the lower levels of refinement. This is significant since, for relatively complex scenes, time-critical collision handling is often forced to interrupt recursion at relatively low levels for a large percentage of simulation objects.

Figure 8 shows the average error in collision response for 100 collision frames in a simulation of randomly colliding objects. The values recorded are errors in the resultant angular velocity magnitude $\Delta\omega$, linear velocity magnitude Δv, and linear velocity directions $\Delta\hat{\mathbf{v}}$. As before, each time a collision was found the resultant response was calculated based on an exact model, a traditional simple averaging approach, and our weighted averaging solution. The error values are simply taken as the difference between the value returned by the exact model and each respective approximate method. For the linear velocity direction we took the difference in angle between the exact method and each of the approximate methods.

Figure 9 shows a visualization of the contact modeling data. The red lines represent the raw contact data per node and the green lines are the resulting reduced normals calculated by our contact modeling mechanism.

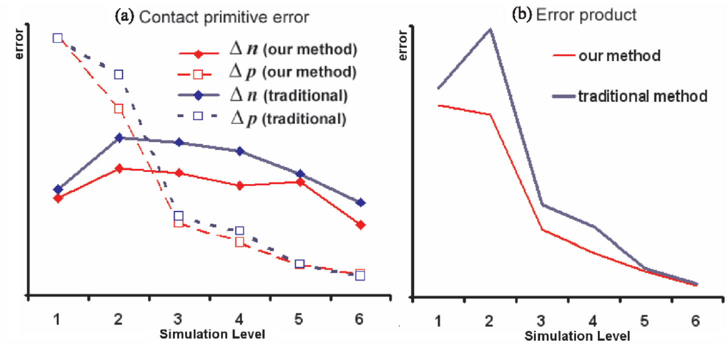

Figure 7. Contact modeling error.

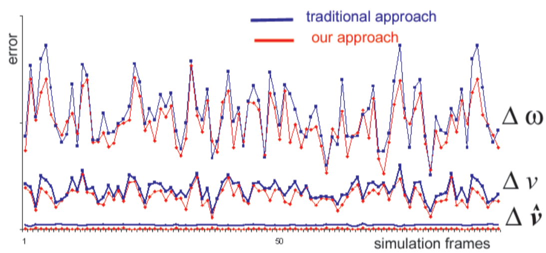

Figure 8. Contact modeling error.

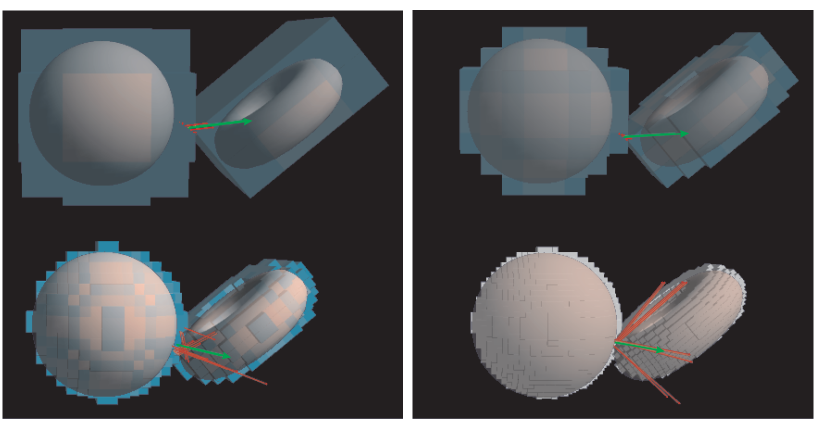

Figure 9. An illustration of resulting contact levels of detail. (Note that vector lines have been enhanced manually for clarity).

5.2. Discussion

Although many existing BVH schemes generate tighter bounding hierarchies than a VoxTree for arbitrary objects in most cases, the collision detection systems that they are based on generally require a final phase of polygon level collision and the BVH phase is only an optimization. In order to return approximate collision response using an interruptible collision detection system, the simple voxel-based data structure is usually more adequate. This is because a single point on the object in most traditional approaches may be bounded by multiple overlapping bounding nodes, and can thus generate multiple contact primitives leading to erroneous results for collision response. Moreover, we should restate that the contact modeling system we propose applies to more than just the VoxTree example we have used throughout our presentation. In fact the technique is applicable to any BVH structure made up of homogeneous sibling nodes at each level of resolution of the collision proxy. As stated, it is preferable that the nodes are also disjoint (i.e., nonoverlapping); however, we believe that the technique works reasonable well where overlaps are consistent throughout the object, e.g., in certain classes of sphere trees.

A voxel-based approach, however, has the advantage of ease of implementation as numerous implementations already exist for voxel and octree generation. BVH generators for some of the more advanced BVH strategies, on the other hand, often involve considerably more effort to implement. Finally, as our approach is specifically designed to optimize the accuracy of collision response in a time-critical system, it is best suited for specific applications where a guaranteed frame rate is a critical requirement and an interruptible collision-handling mechanism is the ideal candidate.

Acknowledgments. We would like to acknowledge the financial support of the Higher Education Authority of Ireland for making the work described in this article possible.

References

[Carlson and Hodgins 97] Deborah A. Carlson and Jessica K. Hodgins. "Simulation Levels of Detail for Real-time Animation." In *Graphics Interface '97*, edited by Wayne A. Davis, Marilyn Mantei, and R. Victor Klassen, pp. 1–8. Toronto, Ontario: Canadian Human-Computer Communications Society, 1997.

[Chatterjee and Ruina 98] Anindya Chatterjee and Andy Ruina. "A New Algebraic Rigid Body Collision Law Based on Impulse Space Considerations." *Journal of Applied Mechanics* 65 (1998), 939–951.

[Chenney and Forsyth 00] Stephen Chenney and David Forsyth. "Sampling Plausible Solutions to Multi-body Constraint Problems." In *SIGGRAPH '00: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 219–228. New York, NY: ACM Press, 2000.

[Chenney et al. 01] Stephen Chenney, Okan Arikan, and D. A. Forsyth. "Proxy Simulations For Efficient Dynamics." In *Eurographics 2001 Short Presentations*, pp. [65–74]. Aire-la-Ville, Switzerland: Eurographics Association, 2001.

[Dingliana and O'Sullivan 00] John Dingliana and Carol O'Sullivan. "Graceful Degradation of Collision Handling in Physically Based Animation." *Computer Graphics Forum (Eurographics 2000 Proceedings)* 19:3 (2000), 239–247.

[Giang 04] Thanh Giang. "Time Adaptive Dynamic Simulation of Rigid Bodies." Ph.D. thesis, Computer Science Department, Trinity College Dublin, 2004.

[Giang et al. 03] Thanh Giang, Gareth Bradshaw, and Carol O'Sullivan. "Complementarity Based Multiple Point Collision Resolution." In *Proceedings Eurographics Ireland 2003*, pp. [1–7]. Aire-la-Ville, Switzerland: Eurographics Association, 2003.

[Gottschalk et al. 96] Stefan Gottschalk, Ming C. Lin, and Dinesh Manocha. "OBB-Tree: A Hierarchical Structure for Rapid Interference Detection." In *SIGGRAPH '96: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, pp. 171–180. New York, NY: ACM Press, 1996.

[He and Kaufmann 97] Taosong He and A. Kaufmann. "Collision Detection for Volumetric Objects." In *VIS '97: Proceedings of the 8th Conference on Visualization '97*, pp. 27–ff. Los Alamitos, CA: IEEE, 1997.

[Hubbard 96] Philip M. Hubbard. "Approximating Polyhedra with Spheres for Time-Critical Collision Detection." *ACM Transactions on Graphics* 15:3 (1996), 179–210.

[Klosowski et al. 98] James T. Klosowski, Martin Held, Joseph S. B. Mitchell, Henry Sowizral, and Karel Zikan. "Efficient Collision Detection Using Bounding Volume Hierarchies of k-DOPs." *IEEE Transactions on Visualization and Computer Graphics* 4:1 (1998), 21–36.

[Krishnan et al. 97] Shankar Krishnan, Amol Pattekar, Ming Lin, and Dinesh Manocha. "Spherical shells: A higher-order bounding volume for fast proximity queries." Technical Report, Department of Computer Science, University of North Carolina, 1997.

[Levoy 88] Marc Levoy. "Display of Surfaces from Volume Data." *IEEE Computer Graphics and Applications* 8:3 (1988), 29–37.

[McNeely et al. 99] William A. McNeely, Kevin D. Puterbaugh, and James J. Troy. "Six Degree-of-Freedom Haptic Rendering Using Voxel Sampling." In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 401–408. New York, NY: ACM Press, 1999.

[Mirtich 96] Brian Mirtich. "Impulse-based Dynamic Simulation of Rigid Body Systems." Ph.D. thesis, University of California, Berkeley, 1996.

[O'Brien et al. 02] David O'Brien, Susan Fisher, and Ming Lin. "Automatic Simplification of Particle System Dynamics." In *Proceedings of Computer Animation 2001*, pp. [210–219]. Los Alamitos, CA: IEEE Press, 2002.

[O'Sullivan and Dingliana 01] Carol O'Sullivan and John Dingliana. "Collisions and Perception." *ACM Trans. Graph.* 20:3 (2001), 151–168.

[O'Sullivan et al. 03] Carol O'Sullivan, John Dingliana, Thanh Giang, and Mary K. Kaiser. "Evaluating the Visual Fidelity of Physically Based Animations." *ACM Transactions on Graphics* 22:3. Proceedings of SIGGRAPH 2003.

[van den Bergen 97] Gino van den Bergen. "Efficient Collision Detection of Complex Deformable Models using AABB Trees." *Journal of Graphics Tools* 2:4 (1997), 1–13.

[Witkin et al. 01] Andrew Witkin, David Baraff, and Michael Kass. *Physically Based Modelling, SIGGRAPH 2001 Course Notes 25*. New York: ACM Press, 2001.

Web Information:

http://www.acm.org/jgt/papers/DinglianaOSullivan05/

John Dingliana, Image Synthesis Group, Computer Science Department, Trinity College, Dublin - 2, Ireland (John.Dingliana@cs.tcd.ie)

Carol O'Sullivan, Image Synthesis Group, Computer Science Department, Trinity College, Dublin - 2, Ireland (Carol.OSullivan@cs.tcd.ie)

Received September 7, 2004; accepted in revised form April 12, 2005.