A Voxel-based Approach to Approximate Collision Handling

John Dingliana and Carol O'Sullivan

August 4, 2004

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). Traditionally, when the collision detection stage is interrupted in such systems, an approximation of contacts is required to deliver a suitable collision response. In our system we use a BVH tree 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 efficiency and plausibility of the resulting animation.

1 Introduction

Fully accurate physics remains an unrealistic goal to strive for in interactive animation. Even the most detailed models of real world physics used in off-line animation are based on assumptions and empirical approximations of real-world laws and physical constants [3]. Furthermore, although many techniques exist for highly accurate physical simulation, most of them still cannot guarantee a target framerate for arbitrarily complex simulation scenarios. 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 tradeoff accuracy for processing time whenever required.

Collision detection, contact modelling and collision response are fundamental components in physical simulators. Unfortunately however, these process are computationally intensive and are a source of bottlenecks in the interactive simulation pipeline and inviting targets for optimisation. In this article we will present a new approach to adaptive collision handling, which uses probabilistic analysis in the attempt to optimise 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 Related Work

Due to the computational cost of dynamic simulation, it is prudent, wherever possible, to optimise computations and to make strategic simplifications in different parts of the system in order to achieve interactive frame-rates, whilst minimising any detrimental effects on the realism of the behaviour of the interactive virtual environment.

Simulation Level of Detail (SLOD) approaches address this issue by making direct changes to the procedures that govern the motion of characters or objects in the scene [20][2][5]. Similar approaches exist in the literature for Collision Handling and we will discusss these briefly in this section.

2.1 Bounding Volume Collision Handling

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, which are much simpler than the models used in the rendering and visualisation of the objects.

One class of modelling data structure that is particularly useful for use as a proxy model is the bounding volume hierarchy (BVH). A bounding volume hierarchy is the union of several geometrically simpler volumes (nodes), constructed in order to represent a more complex object at different levels of detail. In collision detection, we usually ensure that the BVH is a conservative over estimate of the object that is being represented. Commonly used BVH's include Sphere Trees [13], Axis-Aligned Bounding Boxes (AABB-Tree) [25], Oriented Bounding Boxes (OBB-Tree) [10], K-dops [15] and Shell Trees [16].

Hubbard was one of the first to propose achieving time-critical SLOD by changing the level of detail of the model used in Collision Detection [13]. In a time-critical collision detection system, the BVH traversal for potentially colliding objects can be interrupted before it has been determined for certain whether or not a collision has occured. 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. The collision response presented in Hubbard's system, however, was limited to simply swapping the velocities of objects due to the absence of an approximate contact model which makes approximate response possible.

Dingliana and O'Sullivan [6] extended this to a fully time-critical collision handling mechanism with contact modelling and response, and O'Sullivan and Dingliana [21] explored the plausibility tradeoffs resulting from using different LODs in different visual regions of the scene. However, the contact model presented is defined for each colliding node pair, meaning that for the full collision response, multiple contacts need to be reduced by heuristic averaging or by an expensive computational process, such as by solving a large Linear Complimentarity Problem for the system [8]. In this article we will extend this method further by improving the quality of the contact approximation without requiring the resolution of a prohibitively large simulataneous contact problem.

In a more recent study, Otaduy and Lin introduced the notion of Contact Levels of Detail (CLODs) [24] and presented an approach for accelerating collision queries and computing contacts, which can also be used for time-critical computations. This is particularly relevant as their approach incorporates useful run-time error metrics to select appropriate levels of contact refinement and they provide a detailed investigation of the consequences of the speed and resolution tradeoffs for collisions in dynamic simulation. Although our approach differs in that it does not deliver an actual contact manifold but an approximated data set of force or impulse vectors, we take the liberty of referring to our approach as a Contact Level of Detail method as the final Collision Response will be determined by our approximated contact model. What we specifically hope to provide, in this article, is a useful voxel-based alternative for a CLOD implementation, targetted specifically at a time-critical collision handling solution.

2.2 Probabilistic Collision Detection

Barzel *et al.* [1] discuss plausible simulation and suggest that a range of possible state changes might appear acceptably realistic to most human viewers. Chenney and Forsyth [4] show how we can use this knowledge to plausibly tweak an animation to reach some desired final state. Our goal will be to try to reach similar thresholds of acceptability in spite of the approximations forced upon us by an interruptible collision handler.

He and Kaufmann [12] use what they call *the surface crossing probability* for a colliding node in an Octree (or sphere-tree) to determine the likelihood that the collision with the volume node actually signifies an interpenetration of the part of the object represented by the colliding node while Klein and Zachmann [14] present a method for evaluating the probability of error resulting from adaptive collision detection. In both of these, the primary goal is to optimise collision detection and thus they do not discuss how to use the collision detection data in synthesising physically based collision response.

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. The general approach should also apply equally well to a broad range of similar BVH trees with very little modification to the implementation of the algorithm.

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, individual for different objects will not be axis-aligned as 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 in later sections. On the other hand, the hierarchy is fully compatible with an interruptible collision detection system as described by Hubbard.

At the finest level of the BVH the data structure is basically a voxel array and our system can accept a volume representaion as the primary definition of the object. Alternatively if the original model is polygonal, a built in voxeliser extracts the volume representation and calculates voxel density values although it is possible to use the polygonal representation as the finest level of detail. 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 value which is a representation of how much of the node is occupied by the underlying object.

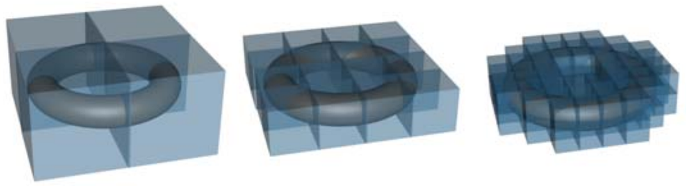

Figure 1: Example VoxTrees for a simple geometric shape

3.2 Contact Modelling 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. It is common practice to make some simplifying assumptions in order to provide a generically applicable system and, in this article, we will focus on a simple dynamics solver that deals with the problem of collision handling for rigid bodies. 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 as described by Baraff [26] and by Mirtich [19].

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 modelling 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*.

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 centre of two colliding nodes, while $\mathbf{p}$ is a point that divides this line in proportions equal to the sizes of the two nodes. Note that this is identical to the approach taken in [6] even though our approach uses cubes instead of spheres.

3.3 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 2). 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 modeller using a mathematical simultaneous contact solution. Thus, we need to reduce the number of contact primitives passed to the collision response module.

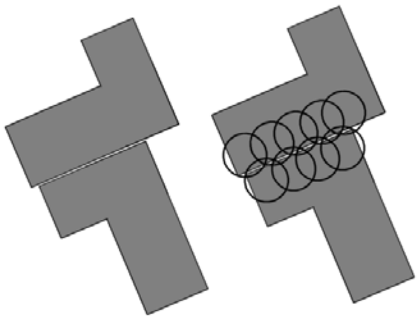

Figure 2: A single face-face collision may sometimes be detected as multiple BVH node collisions

A näive solution would be to interrupt the contact modelling phase when we know we have gathered as much data as the collision response module will be able to handle. However, as the bottleneck is in the response model and contact modelling can in practice deliver much more than collision response can handle, it would be desirable if we could somehow exploit the extra contact modelling data to generate a more accurate collision response without having to solve the expensive simulataneous 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 modelling methods, such as the ones we will discuss in following sections, can be used to perform this reduction and enable a more optimised collision response output.

3.4 Simple Averaged Normals

For non-concave 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 modelling as shown in Equations 1 and 2. We can assume that this is the only viable alternative to solving a prohibitively large LCP in approaches such as Dingliana and O'Sullivan's.

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.

$$\mathbf{p}_{ave} = \frac{\sum \mathbf{p}_j}{N} \tag{1}$$

$$\hat{\mathbf{n}}_{ave} = \frac{\sum \hat{\mathbf{n}}_j}{|\sum \hat{\mathbf{n}}_j|} \tag{2}$$

Simple averaging is adequate when the surfaces are non-concave 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[9] discusses a clever extension to this approach which works by examining a cloud of contact points at run time and finding a reduced set of contact primitives for a specific contact manifold.

3.5 Collision Probability

Further optimisation 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 value called the *occupancy* (or density) of a volume node ρ_i, which is the percentage of the bounding node volume that is actually occupied by the physical object that it is approximating. We will find that there can be significant variance in the values of $\rho_1, \rho_2, \rho_3, ...$ even though they might be sibling nodes in a homogeneous tree (see Figure 3). This in itself is not unobvious as volume rendering techniques have used node density for rendering voxelised data structures, but in our approach we use the density in calculating the dynamic behaviour of the object.

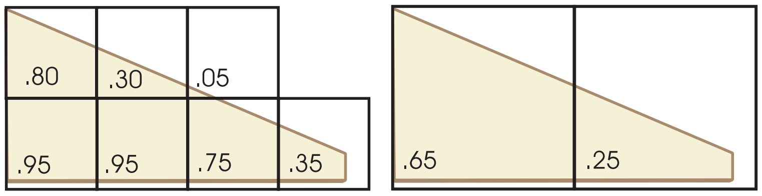

Figure 3: Bounding node occupancy

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. 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.

$$P_{colliding} = \kappa \times \rho_a \times \rho_b \tag{3}$$

κ is a measure of interpenetration and can be described as the probability that an intersection between nodes signifies an actual collision. Equation 3 is in fact simply a generalisation of the bounding volume collision detection described in 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 3.8, but first we should discuss how the value κ is calculated in practice.

3.6 Measuring Interpenetration

The actual value of the interpenetration probability, let us call it K, should 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 4, we take K to be the ratio of the volume of intersection to the volumes of the smaller of the two nodes (V_a and V_b). This is illustrated in Figure 4.

$$K = \frac{V_a \bigcap V_b}{min(V_a, V_b)} \tag{4}$$

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

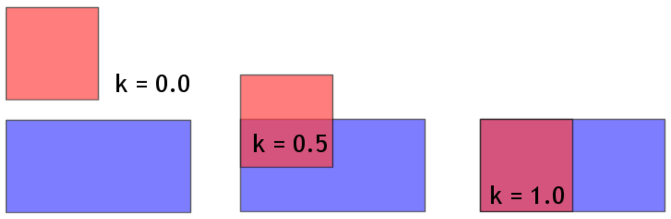

Figure 4: Interpenetration ratio

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:

$$k \propto \frac{(\frac{1}{2}d_K)^3}{(min(R_a, R_b))^3} \tag{5}$$

where (see Figure 5):

$$d_K = R_a + R_b - d \tag{6}$$

In practice, we can safely remove the cubic powers in Equation 5 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:

$$\kappa = \frac{d_K}{2(min(R_a, R_b))} \tag{7}$$

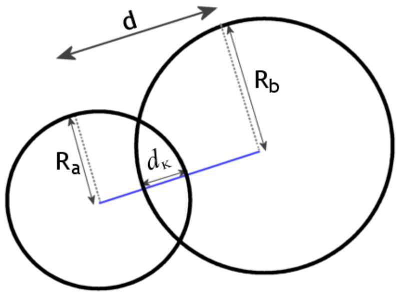

Figure 5: Calculation of diameter of interpenetration d_k

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.

3.7 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 [17].

$$\hat{\mathbf{n}}(x,y,z) = \frac{\nabla\boldsymbol{\rho}(x,y,z)}{|\nabla\boldsymbol{\rho}(x,y,z)|} \tag{8}$$

A similar approach can be taken for volumetric collisions. The occupancy metric 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, McNeely *et al* [18] present an approach which generates contact data for a 6DOF haptic device based on a voxel representation of an object surface and this is a very similar result to what we require for collision response in dynamic simulation. However, in their 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.

However, 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).

3.8 Weighting by Interpenetration and Density

In our system, we not only use the node occupancy but also the collision probability, P in Equation 3, 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:

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

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

This method, after obtaining a weighted average over a relatively large numbers of contact primitives at simulation run-time, produces a reduced number of contact primitives whilst preserving the complexity of data produced by the contact modelling phase in the ensuing collision response. Furthermore, the whole system is still fully interruptible, meaning that we can guarantee real-time frame-rates without wasting the computation cycles that have already been performed by the contact modelling phase.

3.9 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 non-convex 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 localised groups of contact primitives to get several reduced averages for a simpler simultaneous resolution. We therefore must 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 pre-group BVH nodes in the tree at the pre-processing stage. When the tree is initially generated, we store information in the node data-structures about which polygonal primitive they should be associated 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 tesselated, 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, colour 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 eqn 8.

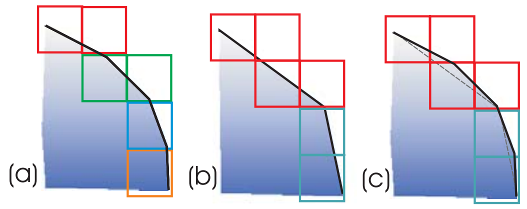

Figure 6: Assigning groupings based on a reduced mesh

4 Results and Evaluation

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, O'Sullivan and Dingliana [21] stressed the value of having a measure of the perceptual plausibility of a simulation and, in O'Sullivan *et al.* [22], an actual metric was proposed for quantifying the plausibility of a physically based animation. In a case study, they demonstrate how this can be used to evaluate the plausibility of different simulation levels of detail in a sphere-tree based dynamic simulator.

4.1 Simulation Levels of Detail Comparision

To illustrate the effectiveness of the speed-accuracy tradeoff in the system, we follow the example of O'Sullivan *et al*'s SLOD case-study. We manually clamp the recursive refinement of the Contact Modelling 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 comparision of our approach with a standard approximate contact modelling mechanism which simply averages multiple contacts without taking node occupancy and intersection depth into account. Data is collected for 1000 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 compute the fully correct response with which the approximate method could be compared.

Figure 7A shows the average error in the computed impulse direction $\Delta\hat{\mathbf{n}}$ and contact point $\Delta\mathbf{p}$ for each Contact Level of Detail. $\Delta\hat{\mathbf{n}}$ 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. 7B 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 as shown in O'Sullivan [23].

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

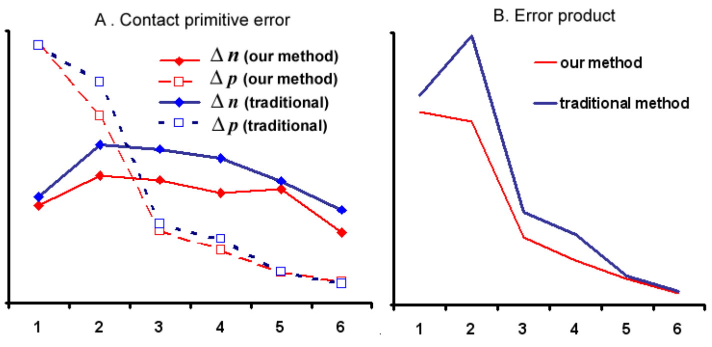

Figure 7: Contact Modelling Error

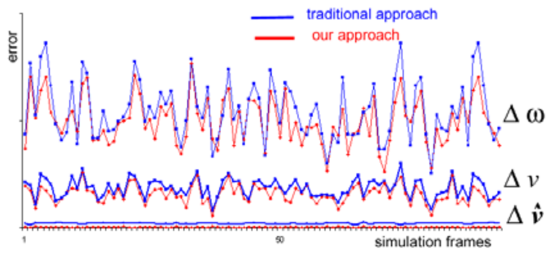

Figure 8: Contact Modelling Error

between the value returned by the exact model and each respective approximate method. For the linear velocity direction we took the differnece in angle between the exact method and each of the approximate methods.

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

5 Conclusions and Future Work

In this article we presented a new method for time-critical approximate collision response utilising node-occupancy information to generate a richer collision response output.

Although alternative BVH schemes exist, which may generate tighter bounding hierarchies for arbitrary objects, we maintain that there are several possible advantages to using our relatively simple voxel-based data structure. Most obvious of these is the ease of implementation, not just in the actual simulation stage but also when it comes to generating the representative BVH proxy model. Not only does it take significant effort to implement systems for building optimised BVH's such as SphereTrees and OBBTrees, but quite often the resulting hierarchies allow overlapping bounding nodes that cause problems for interruptible contact modelling approaches. This is because a single point on the object, since it may be bounded by multiple overlapping bounding nodes, can generate multiple contact primitives and generate erroneous results for collision response. On the other hand, many systems already exist, in the domains of volume-rendering and simulation, for generating not only the voxel-based representations of arbitrary objects, but also the node density values that we make use of in this article.

Furthermore, we suggest that the homogeneous, non-overlapping and orthogonal bounding voxels may present an ideal opportunity for hardware-based optimisation such as in CULLIDE [11]. Although we have not implemented this to date, further work is planned in this direction and also in investigating the feasibility of extending the approach for interactive deformable objects based on BucketTree [7].

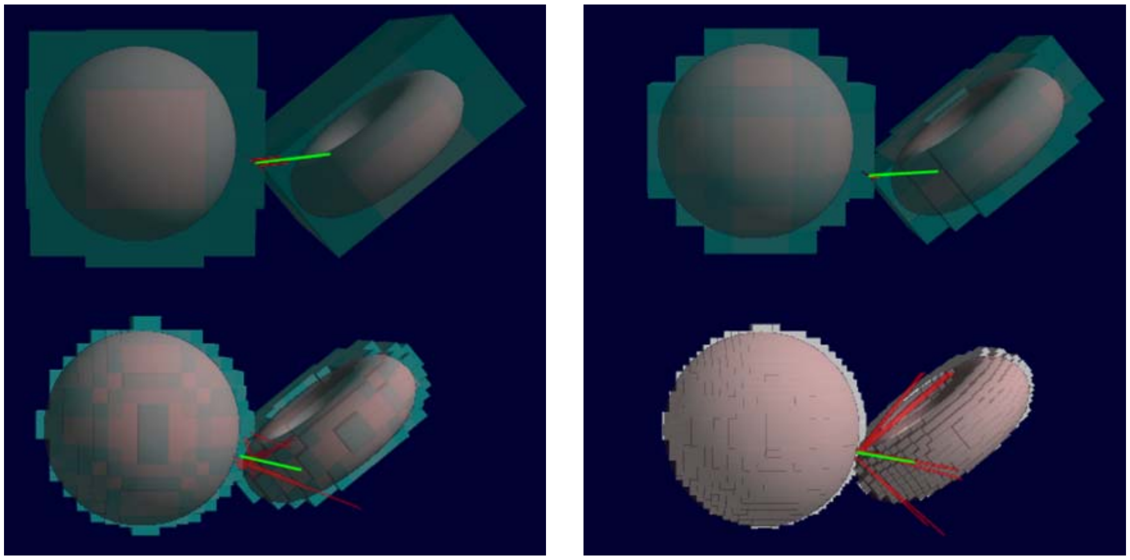

Figure 9: An illustration of resulting Contact Levels of Detail.

For now, we present the approach, not as a general replacement for existing BVH schemes that do indeed generate more efficient BVH structures in most cases, but as an easily implementable alternative in specific applications where a guaranteed frame-rate is a critical requirement and an interruptible collision handling mechanism is the ideal candidate.

References

[1] Ronen Barzel, John F. Hughes, and Daniel N. Wood. Plausible motion simulation for computer graphics animation. In *Computer Animation and Simulation '96*, pages 183–197, 1996.

[2] Deborah A. Carlson and Jessica K. Hodgins. Simulation levels of detail for real-time animation. In Wayne A. Davis, Marilyn Mantei, and R. Victor Klassen, editors, *Graphics Interface '97*, pages 1–8. Canadian Human-Computer Communications Society, 1997.

[3] Anindya Chatterjee and Andy Ruina. A new algebraic rigid body collision law based on impulse space considerations. *Journal of Applied Mechanics*, 65:939–951, December 1998.

[4] S. Chenney and D. Forsyth. Sampling plausible solutions to multi-body constraint problems. In *Proceedings Siggraph 2000*, pages 219–228, 2000.

[5] Stephen Chenney, Okan Arikan, and D. A. Forsyth. Proxy simulations for efficient dynamics. In *Eurographics 2001 Short Presentations*, 2001.

[6] John Dingliana and Carol O'Sullivan. Graceful degradation of collision handling in physically based animation. *Computer Graphics Forum (Eurographics 2000 Proceedings)*, 19(3):239–247, 2000.

[7] F. Ganovelli, J. Dingliana, and C. O'Sullivan. Buckettree: Improving collision detection between deformable objects. Proceedings of the Spring Conference on Computer Graphics (SCCG) 1999.

[8] T. Giang, G. Bradshaw, and C. O'Sullivan. Complementarity based multiple point collision resolution. In *Proceedings Eurographics Ireland 2003*, 2003.

[9] Thanh Giang. *Time Adaptive Dynamic Simulation of Rigid Bodies*. PhD thesis, Computer Science Department, Trinity College Dublin, March 2004.

[10] S. Gottschalk, M.C. Lin, and D. Manocha. Obb-tree: A hierarchical structure for rapid interference detection. In *Proceedings SIGGRAPH '96*, pages 171–180, 1996.

[11] Naga K. Govindaraju, Stephane Redon, Ming C. Lin, and Dinesh Manocha. Cullide: interactive collision detection between complex models in large environments using graphics hardware. In *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware*, pages 25–32. Eurographics Association, 2003.

[12] Taosong He and A. Kaufmann. Collision detection for volumetric objects. In *Proceedings of the 8th IEEE Visualization '97 Conference*, October 1997.

[13] Philip M. Hubbard. Approximating polyhedra with spheres for time-critical collision detection. *ACM Transactions on Graphics*, 15(3):179–210, 1996.

[14] J. Klein and G. Zachmann. Time-critical collision detection using an average-case approach. In *Proceedings of ACM Symposium on Virtual Reality Software and Technology*, 2003.

[15] J.T. Klosowski, M. Held, J.S.B. Mitchell, H. Sowizral, and K. Zikan. Efficient collision detection using bounding volume hierarchies of k-dops. *IEEE transactions on Visualization and Computer Graphics*, 4(1):21–36, 1998.

[16] S. Krishnan, A. Pattekar, M. Lin, and D. Manocha. Spherical shells: A higher-order bounding volume for fast proximity queries, 1997.

[17] Marc Levoy. Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, 8(3):29–37, May 1988.

[18] 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*, pages 401–408. ACM Press/Addison-Wesley Publishing Co., 1999.

[19] Brian Mirtich. *Impulse-based Dynamic Simulation of Rigid Body Systems*. PhD thesis, University of California, Berkeley, 1996.

[20] D. O'Brien, S. Fisher, and M. Lin. Automatic simplification of particle system dynamics. In *Proceedings of Computer Animation 2001*, 2002.

[21] C. O'Sullivan and J. Dingliana. Collisions and perception. *ACM Transactions on Graphics*, 20(3), July 2001.

[22] C. O'Sullivan, J. Dingliana, T. Giang, and M. K. Kaiser. Evaluating the visual fidelity of physically based animations. *ACM Transactions on Graphics*, 22(3), July 2003. Proceedings of SIGGRAPH 2003.

[23] Carol O'Sullivan. *Perceptually-Adaptive Collision Detection for Real-time Computer Animation*. PhD thesis, University of Dublin, 1999.

[24] Miguel A. Otaduy and Ming C. Lin. Clods: dual hierarchies for multiresolution collision detection. In *Proceedings of the Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 94–101. Eurographics Association, 2003.

[25] Gino van den Bergen. Efficient collision detection of complex deformable models using aabb trees. *Journal of Graphics Tools*, 2(4):1–13, 1997.

[26] Andrew Witkin, David Baraff, and Michael Kass. Physically based modelling. In *Siggraph 2001 Course Notes 25.*, 2001.