Enhancing Haptic Rendering through Predictive Collision Detection

Athanasios Vogiannou[1, 2], Konstantinos Moustakas[2], Dimitrios Tzovaras[2], and Michael G. Strintzis[1, 2]

[1] Electrical & Computer Engineering Department, Aristotle University of Thessaloniki, 54006 Thessaloniki, Greece
[2] Informatics and Telematics Institute, Centre for Research and Technology Hellas, P.O. Box 361, 57001, Thermi-Thessaloniki, Greece
{tvog,moustak,tzovaras,michael}@iti.gr

Abstract. This paper presents an efficient collision detection method for interactive haptic simulations of virtual environments that consist of both static and moving objects. The proposed method is based on a novel algorithm for predicting the time of proximity between a pair objects and the appropriate employment of the calculated prediction in a complex virtual scene with multiple objects. The user is able to interact with the virtual objects and receive real-time haptic feedback using the PHANToM Desktop haptic device, while the visual results are shown in the screen display. Experimental results demonstrate the efficiency and the reliability of the presented approach compared to state-of-the-art spatial subdivisions methods, especially for haptic rendering, where collision detection and response is a procedure of critical importance.

Keywords: collision detection and prediction, haptic interaction.

1 Introduction

The goal of Virtual Reality is the immersion of the user into a virtual environment by providing artificial input to its interaction sensors, for example the eyes, the ears and the hands. While the visual inputs are the most important factors in human-computer interaction, virtual reality applications will remain far from being realistic without providing to the user the sense of touch. The use of haptics augments the standard visual interface by offering to the user an alternative way of interaction with the components of the virtual environment. Haptic rendering provides a realistic human-computer interface for the manipulation of virtual objects, improving this way the level of immersion of the user into the virtual world. However, haptic interaction requires the integration of efficient modules for the dynamic generation of contact and friction information.

Collision detection is a major component of dynamic simulations responsible for enhancing interaction and realistic behavior between the virtual objects. Extended research has been performed in the area since the first applications of virtual reality emerged and excellent surveys of prior work can be found in [1], [2], [3] and [4].

The majority of the previous work in collision detection emphasizes on queries between a pair of objects. The dominant approach in this field is Bounding Volume

R. Shumaker (Ed.): Virtual and Mixed Reality, LNCS 5622, pp. 394–402, 2009.

Hierarchies (BVH). The hierarchy consists of primitive objects like spheres [5], AABBs [6], OOBs [7] or discrete orientation polytopes [8]. The main characteristic of bounding volumes is the existence of efficient methods to test for intersection between them. If an intersection is detected in the root of the BVH tree, the algorithm proceeds by checking the volumes in the children of the root and so on, until the leaf nodes are reached and the accurate points of collision are found.

The previous methods are widely used for testing pairs of objects. However, as virtual environments become larger, and include more complex objects, problems arise by the high computational cost of testing each pair. In such scenes it is inefficient to employ the brute force approach of testing all possible pairs of objects because most of the time, the objects are far from each other. Many approaches have been introduced to overcome this difficulty by reducing the pairwise collision tests in large complicated 3D scenes. The most known are the Sweep and Prune technique [9, 10] and Space Partitioning methods [11, 12, 13, 14]. The Sweep and Prune technique can be used to maintain the objects of the scene in some sort of spatial ordering and therefore narrow down testing pairs to objects in adjacent positions of the sorted data structure. Space Partitioning methods divide space into regions and reduce testing in objects of the same region. The most notable of them is the Binary Space Partioning Hierarchy Tree [12], [13].

In this paper we present a new collision detection method for virtual environments containing a significant number of static and moving objects. The proposed method is based on a novel algorithm for predicting the time of proximity between a pair objects. Prediction is particularly useful in interactive haptic applications since with the appropriate object organization, the computational cost of collision detection can be dramatically reduced. In the rest of the text, the general aspects of the proposed framework are presented in section 2 while the implementation details of the approximate collision prediction algorithm are given in section 3. Further details about the interactive haptic rendering are presented in section 4 and the experimental results demonstrating the efficiency of the presented approach are given in section 5.

2 Collision Detection Framework

Broad/narrow phase methods have been widely used for collision detection between a large number of objects [16, 11]. In such systems, the broad phase is responsible for quickly eliminating tests between objects that are impossible to collide due to their distant positions [12, 13, 16], while the narrow phase performs collision tests in pairs of objects using accurate algorithms like in [6], [7] and [8]. From an abstract point of view, the two phases can be seen as ``black boxes''. The objects of the scene are given as input to the broad phase. The produced output contains all the pairs of objects to be further checked for collision, i.e. the broad phase cannot decide that there is no collision for them. These pairs are the input of the narrow phase which returns the accurate collisions for every object in the scene.

In the broad phase of the proposed framework, collision prediction is employed in order to efficiently reduce the tested object pairs. The time value of possible collision for pairs of objects is estimated in continuous time, using a conservative prediction of the trajectories of the objects. The purpose of the prediction is to approximately

determine the time when the two objects will be in close proximity. Further collision tests are not executed in every frame but only in the time intervals where the objects are close to each other. Proximity is explicitly defined using the bounding spheres of each object. When the spheres overlap, the objects are considered to be close to each other. Otherwise the objects are considered to be far enough to be excluded from the narrow phase tests.

The narrow phase follows tests each pair for collision using an external algorithm. By external we mean that it is not a structural part of the framework. If no collision is found for a testing pair, the objects of this pair remain in the narrow, until the respective bounding spheres stop overlapping. This feature differentiates the proposed broad/narrow phase approach from the usual one, as both phases function independently on the objects and not with the classical sequential method. Figure 1 displays the proposed approach for the broad/narrow phase.

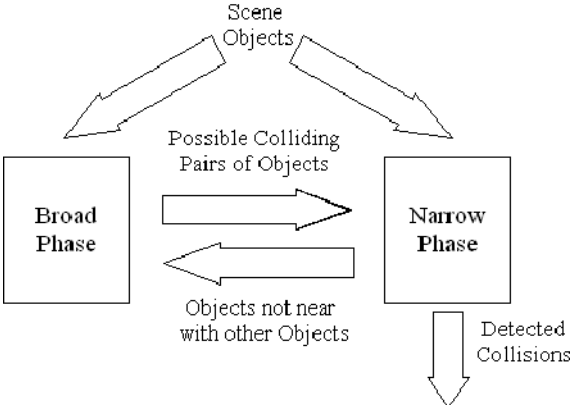

Fig. 1. The flow control diagram of the objects in the proposed framework

3 Approximate Collision Prediction

Prediction is performed utilizing the knowledge of the object's motion and a simple geometric representation of each object that consists of bounding spheres. Spheres are preferred to other bounding volumes, because a simple distance check is enough to determine if a point lies in the interior or the exterior of the sphere, while being also rotation invariant. Furthermore, the registration of the Phantom haptic device is quite straightforward using a sphere at the probe position.

In general, the objects are separated in Interactive Objects (IO) and Environment Objects (EO). IOs are considered to be controlled by the user while EOs are objects that "belong" entirely to the virtual environment. The substantial difference between them is that the motion of IOs cannot be defined accurately, while for the EOs it is well known. In common VR applications with haptic rendering there is only one IO in the scene, such as the Phantom probe. Both types of objects are considered to

translate along a second degree curve for a specific time interval. Although this assumption may seem restrictive, the actual motion of the object is not restricted, since the prediction algorithm needs only a second degree estimation of the real path, even if it is not accurate. The Least Squares Curve Fitting method [15] can be used to calculate an approximation based on points from the real path.

We further extend this approach and instead of using just a specified path, a set of points around this path is considered to represent all the possible positions that the object could lie at a specified time. These points belong to the interior of an area defined by an absolute deviation from the predicted path. The deviation extends to all directions, so that the possible positions of the object form a spherical volume. The center of this sphere moves along the estimated curve while the radius depends on the deviation. A 2D projection of the resulting path is shown in Figure 2 where **s**(t) is the estimated path and R(t) the deviation.

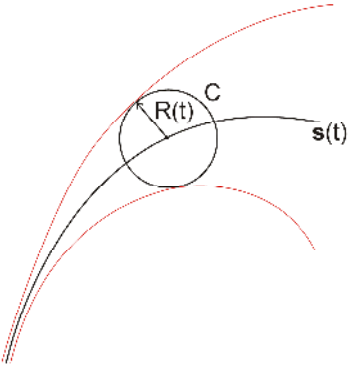

Fig. 2. Estimated path with deviation. The deviation R(t) extends the predicted path **s**(t) in all directions. All points inside C indicate a possible position of the object at time t.

R(t) is considered a simple second degree polynomial such as $R(t) = a + bt + ct^2$. Deviation is also considered positive and monotonically increasing because as time passes, is more probable that the object will diverge from the path. Hence, the deviation model should "tolerate" this change in order to have a valid estimated path for an acceptable time interval.

3.1 Estimation of Proximity

The bounding sphere of an object combined with the estimated path and the deviation, result in a moving bounding volume which increases with time, as shown in Figure 3, where S is the bounding sphere of the object and S´(t) is the extended moving bounding volume. The following analysis involves two objects with arbitrary deviation in their movement. It is then trivial to extend to the specific cases (EO-EO, IO-IO, EO-IO).

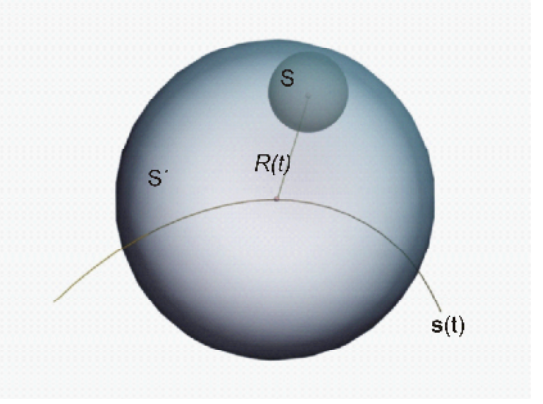

Fig. 3. Moving bounding volume. As the bounding sphere S of the object moves along the estimated path s(t), the deviation R(t) forms a new extended bounding volume S´(t).

Let $\mathbf{s}_1(t)$ be the path, $R_1(t)$ the deviation and r_1 the radius of the bounding sphere of the first object and $\mathbf{s}_2(t)$, $R_2(t)$ and r_2 for the second object respectively. The time of collision between the moving bounding spheres can be calculated by solving

$$| \mathbf{s}_1(t) - \mathbf{s}_2(t) | \leq R_1(t) + R_2(t) + r \ . \tag{1}$$

for min(t > 0), where $r = r_1 + r_2$. Squaring the previous equation ends up to a fourth degree polynomial equation for t, as $R_1(t)$, $R_2(t)$, $|\mathbf{s}_1(t)-\mathbf{s}_2(t)|$ are second degree polynomials. The complexity of solving (1) is prohibitive for real-time applications. To reduce the computational cost, a linear approximation of each path is used. Let t_{init} and t_{final} be the minimum and maximum time values of the interval to check for collision. Let $\mathbf{s}(t)$ denote any of $\mathbf{s}_1(t)$ or $\mathbf{s}_2(t)$ and R(t) any of $R_1(t)$ or $R_2(t)$. Then $\mathbf{s}(t)$ can be approximated in $[t_{init}, t_{final}]$ from the line segment given by

$$\mathbf{s}(m) = \mathbf{s}(t_{init}) + m\mathbf{u} \ . \tag{2}$$

for m in [0,1] and $\mathbf{u} = \mathbf{s}(t_{final}) - \mathbf{s}(t_{init})$. The same approximation can be employed for the deviation R(t), where $R(m) = R(t_{init})+mDR$ and $DR = R(t_{final}) - R(t_{init})$. Figure 4-(a) shows the maximum error value caused by the approximation. To deal with this error, the bounding volume extends further in order to enclose the disregarded area (Figure 4-(b)).

Let t_{error} be the time of the maximum approximation error and $\mathbf{v}_{error} = \mathbf{s}(t_{error}) - \mathbf{s}(t_{init})$ be the vector from the starting position of the estimation to the position of maximum error. It is trivial to prove that

$$t_{error} = t_{init} + (t_{final} - t_{init})/2 \ . \tag{3}$$

and the maximum error e_{max} equals to

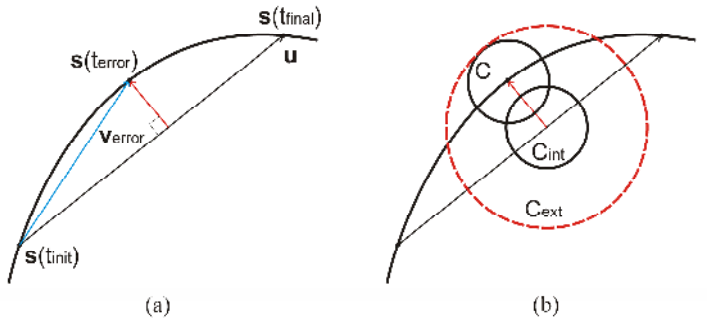

Fig. 4. Figure (a) displays the maximum absolute value of the approximation error which corresponds to the maximum distance between the linear approximation s(m) and the path s(t). The error is used to extend the bounding sphere C_{int}, moving along the linear interpolation, so that it includes the disregarded area of C, as displayed in (b).

$$\mathbf{e}_{max} = \left| \mathbf{v}_{error} \cdot \frac{\mathbf{u} \times (\mathbf{v}_{error} \times \mathbf{u})}{|\mathbf{u} \times (\mathbf{v}_{error} \times \mathbf{u})|} \right| . \tag{4}$$

The same error correction is not necessary for the approximation of R(t). As can be easily seen, R(t) is convex on the interval $[t_{init}; t_{final}]$. Consequently, the approximate deviation results to a larger moving bounding volume and no collision will be missed. Finally, equation (1) can be replaced by

$$|\mathbf{s}_1(m) - \mathbf{s}_2(m)| \leq R_1(m) + R_2(m) + r + e_1 + e_2 \ . \tag{5}$$

where e_1 and e_2 are the maximum approximation errors for the two objects. Let P(m) be the polynomial defined as

$$P(m) = |\mathbf{s}_1(m) - \mathbf{s}_2(m)|^2 - (R_1(m) + R_2(m) + r + e_1 + e_2)^2 \ . \tag{6}$$

Equation (5) holds when P(m) is negative or zero. Note that the valid approximation interval of m is [0,1] because outside this interval, m corresponds to time values outside $[t_{init}; t_{final}]$. That means that not every root of P(m) is a valid collision time. If m_s is a valid solution of P(m) according to the previous analysis, then the estimated time $t_{collision}$ of the next future collision can be approximated by

$$t_{collision} = t_{init} + m_s (t_{final} - t_{init}) \ . \tag{7}$$

If there is no possible collision, then we can be sure that there is no collision until t_{final}.

4 Interactive Haptic Rendering

Briefly, haptic rendering involves the process of detecting human-object collisions and generating the appropriate feedback depending on the texture and the "sense" of the

virtual objects [17], [18], [19], [20]. While all modules are important in human-computer interaction [18], in this paper we focus only on collision detection. In particular we are interested in virtual environments with a significant number of objects.

For the haptic interaction in the experiments, we employed the PHANToM (http://www.sensable.com). The PHANToM is a general purpose haptic device from SensAble Technologies providing highend force feedback while being very simple to use. It has six degrees of freedom for reading the position of the probe and three or six degrees of freedom concerning the feedback, depending on the model.

In the proposed collision detection method, the probe acts as the single IO in the environment while all the other virtual objects are considered as EOs. The user is able to interact with every EO in the scene by setting it in motion, if it is static, or changing its direction, if it is already moving. The challenging part of this simulation is to handle collisions and contact in real-time.

For the rest of the haptic rendering update, force feedback is calculated using a standard spring force model. The force applied to the user is perpendicular to the object surface depending on the spring constant that which represents the physical properties of the simulated object and the penetration depth of the probe into the object.

5 Experimental Results

Experimental tests were conducted to evaluate the performance of the proposed framework. The results elaborate on the performance gain in a real-time application and the comparison with other state-of-the-art approaches for broad phase collision detection. The algorithms of the proposed framework were developed in C++. Other two software packages were used in the tests: SOLID 3.5 (http://www.dtecta.com/) and ODE 0.9 (http://www.ode.org/). SOLID is a collision detection software package containing both broad and narrow phase algorithms, described in detail in [21]. ODE is a general rigid body simulation library with a collision detection engine which employs a Multi-resolution Uniform Grid or a Quad Tree [11] in the broad phase. Tests were performed using a Core2 6600 2,4GHz CPU PC with 2GB of RAM and GeForce 7600 GS Graphics Card.

Table 1. Average performance gain of the proposed approach over state-of-the-art methods

Number of Objects	Brute Force	Sweep & Prune	Quad Tree	Uniform Grid
20	97.9%	83.3%	80.4%	82.1%
40	96.3%	51.8%	51.4%	49.9%
60	97.8%	60.7%	62.6%	64.6%
80	97.9%	46.1%	54.8%	59.7%
100	97.4%	32.7%	38.7%	39.9%

During the experimental tests, the user was free to “touch” a number of static or arbitrary moving virtual objects using the PHANToM probe. The comparative results concerning the performance are displayed in Table 1. The first thing to notice is that

the brute force approach is an order of magnitude slower than the other methods, and the difference increases as more objects are added to the scene. This fact clearly suggests that the brute force approach is extremely aggregative for the application. Comparing the proposed method with the other methods, there is an almost constant improvement in the average performance. The update rate of the application varied from 80 Hz to 450 Hz. The improvement of the proposed method comprises a major advance in haptic interaction since it is very important to achieve very high update rates in order to produce realistic haptic feedback.

5.1 Performance Analysis

A concrete asymptotic analysis of the performance of the proposed method would conclude in a rather obscure result that could not give a clear picture of the algorithm. Therefore, it is preferred to describe the general characteristics of the proposed method. The overall performance highly depends on the frequency of collisions between the objects in the scene. Theoretically, the worst case is $O(n^2)$ and occurs when every object is near with all the other objects. However, the possibility of such a case is almost zero for large n in practical applications, as the usual case for the objects is to be near at discrete time intervals. The average performance can be roughly considered as $O(nc + k)$ where k is the number of pairs in the narrow phase and c is the number of objects that need estimation of future collision. The gain is higher for limited number of collisions in the scene, even for large number of objects, which can be relatively close to each other. It is also important to note that when no objects are near with each other, the performance is almost constant.

References

1. Jimenez, P., Thomas, F., Torras, C.: 3D Collision Detection: A Survey. Computer and Graphics 25, 269–285 (2001)
2. Lin, M.C., Gottschalk, S.: Collision Detection between Geometric Models: A Survey. In: IMA Conference on Mathematics of Surfaces (1998)
3. Hadap, S., Eberle, D., Volino, P., Lin, M.C., Redon, S., Ericson, C.: Collision Detection and Proximity Queries. In: ACM SIGGRAPH Course Notes (2004)
4. Teschner, M., Kimmerle, S., Zachmann, G., Heidelberger, B., Raghupathi, L., Fuhrmann, A., Cani, M.P., Faure, F., Magnetat-Thalmann, N., Strasser, W.: Collision Detection for Deformable Objects. In: Eurographics State-of-the-Art Report (EG-STAR), pp. 119–139 (2004)
5. Hubbard, P.: Approximating Polyhedra with Spheres for Time-Critical Collision Detection. ACM Transactions on Graphics 15, 179–210 (1996)
6. Bergen, G.v.d.: Efficient Collision Detection of Complex Deformable Models using AABB Trees. Journal of Graphics Tools 2, 1–14 (1997)
7. Gottschalk, S., Lin, M.C., Manocha, D.: OBBTree: A Hierarchical Structure for Rapid Interference Detection. Computer Graphics 30, 171–180 (1996)
8. Klosowski, J.T., Held, M., Mitchell, J.S.B., Sowizral, H., Zikan, K.: Efficient Collision Detection using Bounding Volume Hierarchies of k-DOPs. IEEE Transactions on Visualization and Computer Graphics 4, 21–36 (1998)

9. Cohen, J.D., Lin, M.C., Manocha, D., Ponagmi, M.: An Interactive and Exact Collision Detection System for Large-Scale Environments. In: Symposium on Interactive 3D Graphics, pp. 189–196 (1995)
10. Hudson, T., Lin, M.C., Cohen, J., Gottschalk, S., Manocha, D.: V-COLLIDE: Accelerated Collision Detection for VRML. In: VRML 1997: Second Symposium on the Virtual Reality Modeling Language (1997)
11. Ericson, C.: Real-time Collision Detection. Morgan Kaufmann, San Francisco (2005)
12. Luque, R., Comba, J., Freitas, C.: Broad-phase Collision Detection using Semi-Adjusting BSP-Trees. In: I3D 2005: Symposium on Interactive Graphics and Games, pp. 179–186 (2005)
13. Ar, S., Chazelle, B., Tal, A.: Self-customized BSP Trees for Collision Detection. Computational Geometry 15, 91–102 (2000)
14. Eitz, M., Lixu, G.: Hierarchical Spatial Hashing for Real-Time Collision Detection. In: IEEE International Conference on Shape Modeling and Applications, pp. 61–70 (2007)
15. Press, W., Teukolsky, S., Vetterling, T., Flannery, P.: Numerical Recipes in C: The Art of Scientific Computing. Cambridge University Press, Cambridge (1992)
16. Mirtich, B.: Efficient algorithms for two-phase collision detection. TR-97-23, Mitsubishi Electric Research Laboratory (1997)
17. Tzovaras, D., Nikolakis, G., Fergadis, G., Malasiotis, S., Stavrakis, M.: Design and implementation of haptic virtual environments for the training of visually impaired. IEEE Trans. Neural Syst. Rehabil. Eng., 266–278 (2004)
18. Otaduy, M.A., Lin, M.C.: Introduction to haptic rendering. In: ACM SIGGRAPH Courses (2005)
19. Basdogan, C., De, S., Kim, J., Muniyandi, M., Kim, H., Srinivasan, M.A.: Haptics in minimally invasive surgical simulation and training. IEEE Computer Graphics & Applications (2004)
20. Fritz, J.P.: Haptic rendering techniques for scientific visualization. Master thesis, University of Delaware, Newarty Delaware (1996)
21. Bergen, G.v.d.: Collision Detection in Interactive 3D Environments. Morgan Kaufmann, San Francisco (2003)