Joint Virtual Reality Conference of EGVE - EuroVR (2013)
B. Mohler, B. Raffin, H. Saito, and O. Staadt (Editors)

Free-form implicit haptic rendering

Konstantinos Moustakas

Electrical and Computer Engineering Department, University of Patras, Greece

Abstract

This paper presents a framework for haptic rendering in virtual environments based on distance maps over implicit support plane mappings. Initially, a rigid 3D object is modelled using support plane mappings so as to efficiently perform collision detection. Then, and after the collision queries are resolved, the surface of the 3D object can be directly reconstructed in constant time, using the equations of the support planes and the discrete distance map that encodes the distance of the object surface from the support plane. As a result analytical formulae can be extracted that provide the force feedback only as a function of the 3D object spatial transformation and position of the haptic probe. Experimental evaluation and computational complexity analysis demonstrates that the proposed approach can reduce significantly the computational cost when compared to existing collision detection and haptic rendering methods

Categories and Subject Descriptors (according to ACM CCS): I.3.4 [Computer Graphics]: Graphics utilities—Virtual Device Interfaces, H.5.2 [Information Interfaces and Presentation]: User interfaces—Haptic I/O

1. Introduction

Human perception combines information of various sensors, including visual, aural, haptic, olfactory, etc., in order to perceive the environment. Virtual reality applications aim to immerse the user into a virtual environment by providing artificial input to its interaction sensors (i.e., eyes, ears, hands, etc.). The visual and aural inputs are the most important factors in human-computer interaction (HCI). However, 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 audiovisual HCI by offering to the user an alternative way of interaction with the virtual environment [BC03]. However, haptic interaction involves complex and computationally intensive processes, like collision detection or distance calculation [5], that place significant barriers in the generation of accurate and high fidelity force feedback.

1.1. Related work

Seen from a computational perspective, haptic rendering can be decomposed in two different but heavily interrelated processes, namely collision detection and haptic rendering. Initially, collisions have to be identified and localized and then the resulting force feedback has to be estimated so as to accurately render the force that will be fed back to the user using specific assumptions on the physical model involved.

Concerning collision detection, bounding Volumes and the respective Hierarchies (BVH) have dominated the field of collision detection due to the simple and fast pairwise culling they can achieve. Even though they have been initially used for pairwise collision detection of complex rigid models [VMTS10], there are numerous other methods for extending the hierarchies to other types of applications, such as deformable model simulations [TCYM08], continuous collision detection [ZRLK07] and ray-tracing [WBS07]. Most of these methods take advantage of some kind of BVH at an initial culling step before performing more sophisticated algorithms for further collision pruning and contact determination.

The intersection tests between BVs are based on the Separating Axis Theorem for convex objects [GLM96], [CS08]. The theorem states that for a pair of disjoint convex objects there exists an axis, such that the projections of the objects on this axis do not overlap. Intersection tests for BVs exploit this theorem by testing the existence of a Separating Axis in a set of candidate axes. The basic difference among different types of BVs is the number of axes needed to be tested. There is a trade-off between the bounding efficiency and the computational cost of the intersection test. BVs with

DOI: 10.2312/EGVE.JVRC13.073-076

low efficiency, such as spheres, can be tested very fast for intersection while more efficient bounding volumes, such as the OBBs, require much more computation.

Concerning haptic rendering research can be divided into three main categories [LO08]: Machine Haptics, Human Haptics and Computer Haptics [SB97]. Machine Haptics is related to the design of haptic devices and interfaces, while Human Haptics is devoted to the study of the human perceptual abilities related to the sense of touch. Computer Haptics, or alternatively haptic rendering, studies the artificial generation and rendering of haptic stimuli for the human user. It should be mentioned that the proposed framework takes into account recent research on human haptics, while it provides mathematical tools targeting mainly the area of computer haptics.

The simplest haptic rendering approaches focus on the interaction with the virtual environment using a single point [MTS07]. Many approaches have been proposed so far both for polygonal, non-polygonal models, or even for the artificial generation of surface effects like stiffness, texture or friction [MTS07], [LD07]. The assumption, however, of a single interaction point limits the realism of haptic interaction since it is contradictory to the rendering of more complex effects like torque. On contrary, multipoint, or object based haptic rendering approaches use a particular virtual object to interact with the environment and therefore, besides the position of the object, its orientation becomes critical for the rendering of torques [LD07]. Apart from techniques for polygonal and non-polygonal models [LD07], voxel based approaches for haptic rendering [PPTH01] including volumetric haptic rendering schemes [PCY08] have lately emerged. Additionally, research has also tackled with partial success the problem of haptic rendering of dynamic systems like deformable models and fluids [BJ09].

1.2. Motivation and contribution

In general, with the exception of some approaches related to haptic rendering of distance or force fields [MNK*07], [BH07], one of the biggest bottlenecks of current schemes is that haptic rendering depends on the fast and accurate resolution of collision queries. The proposed approach aims to widen this bottleneck by providing a free-form implicit haptic rendering scheme based on support plane mappings. In particular, a 3D object is initially modelled using the associated support plane mappings [VMTS10]. Then the distance of the object's surface from the support plane is mapped at discrete samples on the plane and stored at a preprocessing step. During run-time and after collision queries are resolved, estimation of the force feedback can be analytically estimated. This results in constant time haptic rendering based only on the 3D transformation of the associated object and the position of the haptic probe.

2. Haptic rendering using support plane mappings

Support planes are a well studied subject of computational geometry and have been employed in algorithms for the separation of convex objects [DK85, CW96, vdB03]. From a geometrical perspective, a support plane E of a 3D convex object O is a plane such that O lies entirely on H_E^-. Support planes have become useful in previous algorithms based on the concept of support mappings. A support mapping is a function that maps a vector $\mathbf{v}$ to the vertex of $vert(O)$ that is "most" parallel to $\mathbf{v}$ [vdB03, Eri05]. As a direct consequence, a support plane can be defined as the plane that passes through $s_O(\mathbf{v})$ and is parallel to $\mathbf{v}$. In the following we will adopt the formulation used bu Vogiannou et.al. in [VMTS10].

2.1. Haptic support plane maps

After collision is detected, the force feedback provided to the user through the haptic device has to be calculated. In the present framework, force feedback is obtained directly from the model adopted for collision detection, thus handling collision detection and haptic rendering in an integrated way, as described in the sequel.

Let the parametric form of the support plane equation $S_{SP}(\eta,\omega)$ be:

$$\mathbf{S_{SP}}(\eta,\omega) = \begin{bmatrix} x_0+\eta u_1+\omega v_1 \\ y_0+\eta u_2+\omega v_2 \\ z_0+\eta u_3+\omega v_3 \end{bmatrix}, \forall \eta,\omega \in \Re \qquad (1)$$

where $\mathbf{u}$ and $\mathbf{v}$ constitute an orthonormal basis of the support plane and (x_0,y_0,z_0) its origin.

Assuming now a dense discretization of the η,ω space, we can define a discrete distance map of the support plane SP and the underlying manifold mesh surface S_{mesh}, by calculating the distance of each point of SP from S_{mesh}:

$$D_{SP}(\eta,\omega) = ICD(S_{SP},S_{mesh}) \qquad (2)$$

where ICD calculates the distance of every point sample (η,ω) of the support plane SP, alongside the normal direction at point (η,ω), from the mesh S_{mesh} and assigns the corresponding values to the distance map $D_{SP}(\eta,\omega)$. The distance map is used in the sequel to analytically estimate the force feedback.

2.2. Haptic rendering using SPMs

Referring to Figure 1, let point $\mathbf{H}_p$ be the position of the haptic probe and S_{mesh} represent the local surface of the object.

Let also S_{SP} represent the distance of point $\mathbf{H}_p$ from the support plane, which corresponds to point $\mathbf{P}_M$ on the SP. If collision is detected, the absolute value of the force fed onto

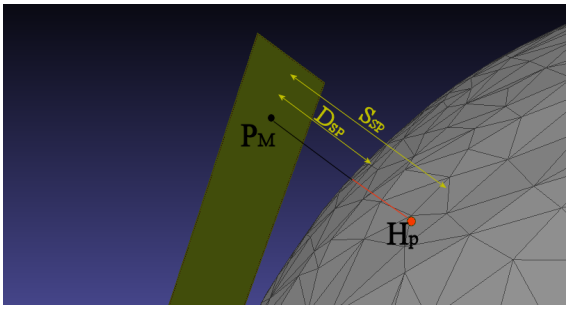

Figure 1: *Distance calculation using distance maps over support planes*

the haptic device is obtained using a spring model as illustrated in Figure 1. In particular:

$$\|\mathbf{F}\| = k \cdot |S_{SP} - D_{SP}(\mathbf{P_M})| \tag{3}$$

where k is the spring constant. $D_{SP}(\mathbf{P_M})$ is the distance of point $\mathbf{P_M}$ from the mesh and is stored in the distance map of the support plane. Notice that the term $|S_{SP} - D_{SP}(\mathbf{P_M})|$ is an approximation of the actual distance of $\mathbf{H}_p$ from the mesh that becomes more accurate if the support plane surface approximates well the mesh.

The direction of the force should in general be perpendicular to the local area, where collision is detected. An obvious solution to the evaluation of the direction of this force would be to detect the surface element (i.e. triangle), where the collision occurred and to provide the feedback perpendicularly to it. This approach is not only computationally intensive, but also results in non-realistic non-continuous forces at the surface element boundaries. In the present framework the analytical approximation of the mesh surface is used utilizing the already obtained SP approximation and the distance map. Based on this approximation the normal to the object's surface can be approximated rapidly with high accuracy. In particular, if $D_{SP}(\eta,\omega)$ is the scalar function of the distance map on the support plane, as previously described, the surface S_{mesh} of the modelled object can be approximated by equation (4) (Figure 1):

$$\mathbf{S}_{mesh}(\eta,\omega) = S_{SP}(\eta,\omega) - D_{SP}(\eta,\omega)\,\mathbf{n}_{SP} \tag{4}$$

where S_{SP} is the surface of the support plane, D_{SP} the associated distance map and $\mathbf{n}_{SP}$ its normal vector that can be easily evaluated through $\mathbf{n}_{SP} = \mathbf{u} \times \mathbf{v}$.

Now the calculation of the force feedback demands the evaluation of the normal vector $\mathbf{n}_{SP}$ on the object's surface. that is obtained through equation (5). In the following the brackets (η,ω) will be omitted for the sake of simplicity.

$$\mathbf{n}_S = \frac{\partial \mathbf{S}_{mesh}}{\partial \eta} \times \frac{\partial \mathbf{S}_{mesh}}{\partial \omega} \tag{5}$$

where

$$\frac{\partial \mathbf{S}_{mesh}}{\partial \eta} = \frac{\partial \mathbf{S}_{SP}}{\partial \eta} - \frac{\partial D_{SP}}{\partial \eta}\mathbf{n}_{SP} - D_{SP}\frac{\partial \mathbf{n}_{SP}}{\partial \eta} \tag{6}$$

Since $\mathbf{n}_{SP}$ is constant over the SP equation (6) becomes:

$$\frac{\partial \mathbf{S}_{mesh}}{\partial \eta} = \mathbf{u} - \frac{\partial D_{SP}}{\partial \eta}\mathbf{n}_{SP} \tag{7}$$

A similar formula can be extracted for $\frac{\partial \mathbf{S}_{mesh}}{\partial \omega}$:

$$\frac{\partial \mathbf{S}_{mesh}}{\partial \omega} = \mathbf{v} - \frac{\partial D_{SP}}{\partial \omega}\mathbf{n}_{SP} \tag{8}$$

All above terms can be computed analytically, except from $\frac{\partial D_{SP}}{\partial \eta}$ and $\frac{\partial D_{SP}}{\partial \omega}$ that are computed numerically.

Substituting now equations (4), (6), (7), (8) in equation (5) the normal direction $\mathbf{n}_S$ can be obtained.

Since, the direction of the normal along the surface of the modelled object is obtained using equation (5), the resulting force feedback is calculated through:

$$\mathbf{F}_h = k\,|S_{SP} - D_{SP}(\mathbf{P}_M)|\,\frac{\mathbf{n}_S}{\|\mathbf{n}_S\|} \tag{9}$$

Moreover, it should be emphasized that several effects like friction, haptic texturing or haptic icons can be very easily integrated in equation (9) thus leading to a very compact closed form computation of haptic rendering.

3. Computational complexity

An experimental analysis of the proposed support plane mapping based haptic rendering approach, in terms of timings for simulation benchmarks would not be fair for the state-of-the-art approaches, since it would encode the superiority of SPM based collision detection [VMTS10] and would not directly highlight the proposed haptic rendering approach. In the following an analysis of the computational complexity of the proposed scheme in comparison to the typical state-of-the-art mesh-based haptic rendering scheme is discussed.

After collision is reported, a typical force feedback calculation scheme would need to identify the colliding triangle of the involved 3D object in $O(n)$ time, where n is the number of triangles, or in $O(logn)$ time if bounding volume hierarchies are used. Then the force can be calculated in constant $O(1)$ time. In order to avoid force discontinuities, for example force shading, and if there is no adjacency information then the local neighbourhood of the colliding triangle can be found again in $O(n)$ time, where n is the number of triangles, or in $O(logn)$ time if bounding volume hierarchies are

used. Finally, the mesh-based haptic rendering scheme has no additional memory requirements per se.

On the other concerning the proposed free-form implicit haptic rendering scheme, after a collision is detected, the resulting force feedback can be calculated in constant time $O(1)$ using equation (9). In order to avoid depth discontinuities the distance map can be smoothed, in an image processing sense, in a preprocessing phase. Even if this step is performed during run-time it would take $O(k)$ time, where k is the local smoothing region or the filtering kernel window. On the other hand the proposed scheme has $O(m \cdot s)$ memory requirements, where m is the number of support planes and s the number of samples per support plane. Taking now into account that the more support planes are used the smaller their size and the less samples are necessary for a specific sampling density we can safely assume that the memory requirements are linear to the total number of samples that depends on the sampling density used.

The following table summarizes the computational complexity analysis of the proposed free-form haptic rendering scheme, when compared to the mesh-based approach.

Process	Mesh-based	Free-form
Force	$O(n)$ or $O(logn)$	$O(1)$
Smoothing	$O(n)$ or $O(logn)$	$O(1)$
Memory	-	$O(m \cdot s)$

Table 1: *Computational complexity comparison*

4. Conclusions-Discussion

The proposed approach introduces an implicit free-form haptic rendering scheme of rigid bodies based on distance maps over support plane mappings and therefore exploits the superiority and bounding efficiency of SPMs for collision detection and extends it for direct closed-form haptic rendering. The proposed approach is seen to be highly efficient when compared to the state-of-the-art mesh-based haptic rendering at a cost, however, of increased memory requirements. Moreover, as presented, it cannot deal with deformations. However, if the deformations are analytically defined then a solution should be possible. This topic remains a research work direction. Finally, vectorial distance maps, instead of scalar as presented in the proposed scheme, can be also incorporated so as to efficiently deal with complex concavities or cases where the object's manifold is not topologically equivalent to a sphere.

References

[BC03] BURDEA G., COIFFET P.: *Virtual Reality Technology*. Wiley-IEEE Press, 2nd edition, 2003. 1

[BH07] BARLIT A., HARDERS M.: Gpu-based distance map calculation for vector field haptic rendering. In *Eurohaptics* (March 2007), pp. 589–590. 2

[BJ09] BARBIC J., JAMES D.: Six-dof haptic rendering of contact between geometrically complex reduced deformable models: Haptic demo. In *In Proc. of Eurohaptics* (March 2009), pp. 393–394. 2

[CS08] COMING D., STAADT O.: Velocity-aligned discrete oriented polytopes for dynamic collision detection. *IEEE TVCG 14*, 1 (2008), 1–12. 1

[CW96] CHUNG K., WANG W.: Quick Collision Detection of Polytopes in Virtual Environments. In *ACM Symposium on Virtual Reality Software and Technology 1996* (July 1996), pp. 1–4. 2

[DK85] DOBKIN D. P., KIRKPATRICK D. G.: A linear algorithm for determining the separation of convex polyhedra. *Journal of Algorithms 6*, 3 (1985), 381 – 392. 2

[Eri05] ERICSON C.: *Real-Time Collision Detection*. The Morgan Kaufmann Series in Interactive 3D Technology. Morgan Kaufmann, 2005. 2

[GLM96] GOTTSCHALK S., LIN M., MANOCHA D.: OBBTree: A Hierarchical Structure for Rapid Interference Detection. In *Computer Graphics, ACM SIGGRAPH* (1996), pp. 171–180. 1

[LD07] LAYCOCK S., DAY A.: A survey of haptic rendering techniques. *Computer Graphics Forum 26*, 1 (January 2007), 50–65. 2

[LO08] LIN M., OTADUY M.: Haptic rendering: Foundations, algorithms and applications. p. A.K.Peters publishing. 2

[MNK*07] MOUSTAKAS K., NIKOLAKIS G., KOSTOPOULOS K., TZOVARAS D., STRINTZIS M.: Haptic rendering of visual data for the visually impaired. *IEEE Multimedia 14*, 1 (January 2007), 62–72. 2

[MTS07] MOUSTAKAS K., TZOVARAS D., STRINTZIS M.: Sq-map: Efficient layered collision detection and haptic rendering. *IEEE Transactions on Visualization and Computer Graphics 13*, 1 (January 2007), 80–93. 2

[PCY08] PALMERIUS K., COOPER M., YNNERMAN A.: Haptic rendering of dynamic volumetric data. *IEEE Transactions on Visualization and Computer Graphics 14*, 2 (April 2008), 263–276. 2

[PPTH01] PETERSIK A., PFLESSER B., TIEDE U., HOHNE K.: Haptic rendering of volumetric anatomic models at sub-voxel resolution. In *In Proc. of Eurohaptics, Birmingham, UK* (2001), pp. 182–184. 2

[SB97] SRINIVASAN M., BASDOGAN C.: Haptics in virtual environments: Taxonomy, research status and challenges. *Computers and Graphics* (1997), 393–404. 2

[TCYM08] TANG M., CURTIS S., YOON S.-E., MANOCHA D.: Interactive continuous collision detection between deformable models using connectivity-based culling. In *SPM âĂŹ08: Proceedings of the 2008 ACM Symposium on Solid and Physical Modeling* (2008), pp. 25–36. 1

[vdB03] VAN DEN BERGEN G.: *Collision Detection in Interactive 3D Environments*. The Morgan Kaufmann Series in Interactive 3D Technology. Morgan Kaufmann, 2003. 2

[VMTS10] VOGIANNOU A., MOUSTAKAS K., TZOVARAS D., STRINTZIS M.: Enhancing bounding volumes using support plane mappings for collision detection. *Computer Graphics Forum 29*, 5 (2010), 1595–1604. 1, 2, 3

[WBS07] WALD I., BOULOS S., SHIRLEY P.: Ray tracing deformable scenes using dynamic bounding volume hierarchies. *ACM Transactions on Graphics 26*, 1 (2007). 1

[ZRLK07] ZHANG X., REDON S., LEE M., KIM Y.: Continuous collision detection for articulated models using taylor models and temporal culling. *ACM Trans. Graph 26*, 3 (2007). 1