Computer-Aided Design 40 (2008) 1055–1066

Contents lists available at ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad

Efficient algorithm to detect collision between deformable B-spline surfaces for virtual sculpting

Harish Pungotra[a], George K. Knopf[a,*], Roberto Canas[b]

[a] Department of Mechanical and Materials Engineering, Faculty of Engineering, The University of Western Ontario, 1151 Richmond St. N, London, Ontario, Canada, N6A 5B9
[b] National Research Council (NRC), London, Ontario, Canada

ARTICLE INFO

Article history:
Received 23 June 2008
Accepted 26 September 2008

Keywords:
Collision detection
Haptics
Virtual sculpting
Deformable objects
B-spline surface
Concept design
Virtual reality

ABSTRACT

A structured computational framework to efficiently detect collision between deformable freeform shapes in a VR environment is proposed in this paper. The deformable shape is represented as a B-spline surface and no assumption is made with regard to the degree of the surface, extent of deformation or virtual material properties. The proposed technique calculates and stores transformation matrices and their inverse during preprocessing, which are then used to discretize the B-spline surfaces. It exploits the fact that the transformation matrices for calculating discrete points on the B-spline are independent of the position of control points and therefore can be pre-calculated. The intensity of the points is dynamically increased at lower levels of detail as per accuracy requirements, and finally the regions which are likely to undergo collision are tessellated using these points. Spheres are used to determine lower levels of detail which makes this algorithm highly suitable for multiple contact collision detection. The algorithm efficiently calculates tangents and surface normals at these points. The surface normals give inside/outside property to the triangulated region and tangents provide the necessary information to model tangential forces such as frictional forces. The proposed algorithm is especially suitable for sculpting during concept design and its validation before exchanging information with existing CAD softwares for detailed design. A comparison based on the worst case scenario is presented to prove the efficiency of the present algorithm.

1. Introduction

Recent advancements in high-speed, multi-core computer hardware and virtual reality (VR) technology provide opportunities to link the more fluid processes of creative conceptual design with the rigidly defined tasks of product detailing and engineering analysis. The need for a viable VR-based conceptual design tool comes from several case studies described by Sener et al. [1], Cheshire et al. [2], and Ye et al. [3,4]. One key conclusion derived is, that the human–computer interface and related software tools for interacting with the virtual models must be intuitive to the user, provide sensory feedback during design, and mimic the natural way the consumer would interact with the product concepts that are being created.

Shape modification of a virtual object can be simulated using either geometric- or physics-based algorithms [5]. Geometric techniques only adjust the vertices of the underlying mesh model in response to external forces. The reaction force is typically determined using Hooke's law where the depth of haptic tool penetration is calculated based on the current and home positions of the node that is nearest to the contact point. The concept was originally suggested by Sederberg [6] and was further extended by Basdogan [5,7] for applications in medical simulation. Hoffmann [8] used the technique for simulation of physical systems in robotics. Physics-based models, on the other hand, are able to both estimate the direction and magnitude of nodal movement based on realistic material properties and the external forces introduced to the model through the haptic tool. Fig. 1 shows the basic architecture of a typical physics-based haptic system.

The haptic device works as an interface between the real and the virtual world. As the user interacts with the virtual object, the relative positions of the virtual tool and the object are calculated. A collision detection algorithm then provides the contact information of the virtual tool with the object. The force exerted by the user through haptic device and position of interaction with the object are used by a physics-based model to calculate the deformation of the object and reactive forces to be sent back to the user.

Many types of surfaces have been used to represent the model in the virtual environment. There are three important types of surfaces that can be used to represent a virtual model: implicit surface, tessellated surface, and parametric surface. An implicit

* Corresponding author. Tel.: +1 519 661 2128; fax: +1 519 661 3757.
E-mail address: gknopf@eng.uwo.ca (G.K. Knopf).

0010-4485/$ – see front matter Crown Copyright © 2008 Published by Elsevier Ltd. All rights reserved.
doi:10.1016/j.cad.2008.09.006

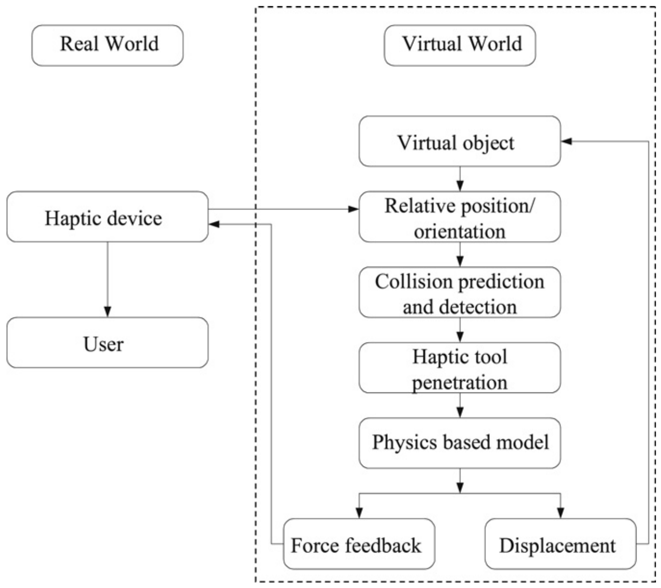

Fig. 1. Basic architecture of a physics-based haptic system.

surface consists of points in three dimensional space that satisfy a specific requirement, represented mathematically by a function f whose argument is a point p such that if $f(p) = 0$, then the point 'p' is on the surface [9]. However the intersection test for implicit surfaces of degree higher than 3 is computationally very intensive.

The most widely used surface in geometric modeling is the tessellated surface because of the inherent simplicity of triangle–triangle collision detection and the advantages of hardware compatibility with tessellated models for rendering and graphical representations. However there are many disadvantages of using tessellated surfaces for representing deformable models. The most significant disadvantage of using a tessellated model is that the resolution of the surface cannot be dynamically changed during haptic interaction. Secondly, as the model deforms during interaction with the tool or other model, the quality of the triangles deteriorates. This makes collision detection computationally extensive and inefficient.

Over the past decade, B-spline representation has become the standard for CAD/CAM systems. Thus it is imperative that any haptic concept design module should utilize a B-spline surface to represent the virtual model in order to streamline the exchange of the information with existing CAD/CAM systems. A major obstacle in using the B-spline surface to represent a deformable model is the absence of an efficient algorithm to detect collision between two or more B-spline surfaces having a complex surface. For applications in concept design in a virtual reality environment using haptics, a collision detection algorithm must be capable of tackling complex surfaces, a large area of contact, multiple contacts and high deformation.

This paper proposes an efficient method for precise collision detection between two or more B-spline surfaces. Both the model and the tool are represented as B-spline surfaces and as a special case, the tool can also be represented as an implicit surface or a point-based tool. No constraint regarding the shape, degree, and number of control points of the B-spline surface representing the tool or model has been considered. Both the model and the tool can have complex shape, elastic or plastic properties, and multiple contacts. This makes this algorithm especially suitable for sculpting during concept generation and validation of the concept design. The paper compares the computations required for a tessellated surface and a cubic B-spline surface based on the proposed algorithm using the worst case scenario (Big $\boldsymbol{O}$ notation) to prove the efficiency of the proposed algorithm.

2. Related work

For applications in concept design, the number of objects is limited but the surfaces are complex. The deformation may be very high depending upon the material properties assigned to the objects. The model presented by Bordegoni [10] detects collision, computed as the intersection between the tessellated models and tools. The force response system is based on geometric technique. Knopf and Igwe [11] presented a novel, interactive virtual sculpting framework based upon a deformable mesh model generated by a self-organizing feature map (SOFM). Igwe [12] further developed a computational framework for economically representing deformable solid objects for conceptual design. A major drawback of this computational framework is its inability to detect collision in a haptic environment.

There are many collision detection algorithms proposed by various researchers. These can be subdivided into algorithms for rigid bodies and those for deformable bodies. Most of these algorithms fall in the former category and there are not many efficient collision detection algorithms for deformable bodies having a complex surface. In this section some of the important algorithms are briefly discussed.

Ho [13] proposed a '*Neighbourhood Watch*' algorithm that is capable of handling rigid bodies with both convex and concave surfaces. Various bounding geometries and spatial geometries help in performing a '*rejection test*' when two virtual objects are apart. Once the objects are in close proximity, spatial decomposition is used to perform a '*rejection test*' for decomposed/subdivided parts of the objects. Gregory [14] uses a pre-computed hybrid hierarchical representation, consisting of uniform grids and trees of tight-fitting Oriented Bounding Box Trees (OBB Trees). Ehmann [15] presented a unified approach to perform a set of proximity queries for general, rigid polyhedral objects and Bradshaw [16] presented work in Sphere-Tree construction and medial axis approximation using the Veronoï diagrams [17]. Hoffmann [8] approximates the object as the union of several cuboids enclosed by a single cuboid. The bounding geometry techniques work very well with rigid bodies but when applied to deformable bodies, the cost of updating these geometries, as the object deforms, slows down the collision detection response. If these bounding geometries are not updated in each frame to reduce the computational cost, the overlapping of bounding volumes make the '*rejection test*' inefficient. The computational cost of collision detection and the haptic rate achieved is not mentioned in these papers.

Thompson [18] uses a tracing algorithm, supporting the rendering of NURBS surface, which traces the closest point on a NURBS patch to the tool point. Dachille [19] uses a polyhedral representation which makes it easier to search for the nearest point on the surface, unlike the complicated NURBS surface intersection task proposed by Thompson [18]. In this case the nearest point on the surface does not need to be updated too frequently, so the system can compute the distance from the cursor to all the vertices of the polyhedral representation. In both of the techniques, the interaction with the virtual model is only at a point through a point-based tool. This limits the utility of the approaches, particularly in the area of concept design where the sculpting activity is required to be done by hands or other surface-based tools. Gao [20,21] uses a B-spline patch to represent the surface of a virtual model and an implicit surface to represent the virtual tool. Nodes are generated on the B-spline surface by discretization and are used to incorporate a mass spring system and to detect collision by inputting these nodes in the equation of the implicit surface of the tool. This technique cannot be used when two or more B-spline surfaces are present in the virtual environment and are interacting with each other. Only rigid and lower degree (up to 2) implicit