Nuclear Instruments and Methods in Physics Research A 371 (1996) 434–438

NUCLEAR INSTRUMENTS & METHODS IN PHYSICS RESEARCH Section A

ELSEVIER

Particle tracking in sophisticated CAD models for simulation purposes

J. Sulkimo, J. Vuoskoski*

CERN, Geneva, Switzerland

Received 11 August 1995

Abstract

The transfer of physics detector models from computer aided design systems to physics simulation packages like GEANT suffers from certain limitations. In addition, GEANT is not able to perform particle tracking in CAD models. We describe an application which is able to perform particle tracking in boundary models constructed in CAD systems. The transfer file format used is the new international standard, STEP. The design and implementation of the application was carried out using object-oriented techniques. It will be integrated in the future object-oriented version of GEANT.

1. Introduction

A high energy physics detector [1] is a complex and expensive device. In order to design these detectors so that they are suitable for physics purposes, simulation programs like GEANT [2] are used. Since the construction of sub-detectors is often carried out in different geographical locations, multiple computer aided systems are utilized for designing, engineering and manufacturing these complex and sophisticated detectors. To ensure flexible and fast design processes from the physicist's and engineer's point of view, the communication between GEANT and computer aided design (CAD) systems should be seamless.

CAD systems and GEANT have different functional requirements and therefore different constraints. The purpose of CAD modellers is to be able to describe models in a precise and accurate way for engineering analysis and manufacturing purposes. The number of solids to be handled at a time is not critical, while the representation has to be as exact as possible. Consequently, CAD systems are not normally able to represent a large number of solids simultaneously.

GEANT, on the other hand, has to describe a large number of solids, say 10 million or more, corresponding to realistic modern detectors. The purpose of the GEANT modeller is, therefore, to be able to represent effectively a large number of solids corresponding to a complex detector in order to perform fast and efficient particle tracking. One can often accept approximative geometric representations, such as restriction to second order surfaces, while the number of solids to be represented is critical.

Transferring detector models from GEANT to CAD systems is well understood and reported in Refs. [3,4]. If the number of solids is large, it can cause some difficulties in the data transfer in this direction. However, transferring detector models from CAD systems into GEANT is problematic because GEANT is normally not capable of employing detector models constructed in CAD systems: one can define objects in a CAD system which do not have any representation in GEANT. The difficulties derive from the fundamental differences between the geometric modellers of both systems. Modern CAD systems normally use boundary representation (B-Rep) [5,6] to describe objects. This differs from the constructed solid geometry (CSG) [5,6] used in GEANT. Conversions from B-Rep to CSG can lead to ambiguities and thus are generally not possible [6,7], while conversions from CSG to B-REP require very large computer memories for realistic detectors.

Different approaches to solve this problem are discussed in Ref. [8]. The most comprehensive solution is to include geometric representations of CAD systems in the GEANT geometric modeller, thus exploiting both CAD modelling systems and GEANT tracking at the same time. This then allows the transfer of sophisticated models into GEANT without restriction. An extension to the GEANT geometric modeller in order to be able to operate with B-Rep models requires the implementation of effective tracking routines. Moreover, this approach can take advantage of the efficient GEANT volume searching routines.

This paper describes GEREP, an application for the proposed future object-oriented GEANT 4 toolkit [9]. GEREP is able to perform particle tracking in detector

* Corresponding author. Tel. +41 22 767 6074, fax +41 22 767 8666, e-mail jouko.vuoskoski@cern.ch.

SSDI 0168-9002(95)01238-9

models described using B-Rep. The models are read from STEP (STandard for the Exchange of Product model data) [10-12] files produced by CAD systems. We present an algorithm for particle tracking in order to compute effectively the closest trajectory intersection point. Test results are given. GEREP was designed using Object Modelling Technique (OMT) [13] and implemented using C++ [14]. A detailed description of GEREP is presented in Ref. [15].

STEP is a new and important international standard for product model data transfer between CAD systems, and it is becoming widely supported by industry. It separates the representation of product information from the implementation methods and provides a system-independent way of implementation. It includes an object-flavoured data specification language, EXPRESS [16,17], to describe the representation of the data. The use of a data specification language facilities development of applications. STEP specifies the implementation methods [12] used for data exchange, and also defines the geometric and topological representation [18].

The fundamentals of B-Rep are explained in Section 2 and in Section 3, an overview of GEREP is given. The particle tracking algorithm is presented in Section 4 and the implementation of GEREP in Section 5. The test results are shown in Section 6, and finally, in Section 7, we summarize.

2. Boundary representation

B-Rep models represent a solid indirectly by a representation of its bounding surface. A solid is represented as a volume contained in a set of faces together with topological information which defines the relationships between the faces. The boundary of a solid separates points inside from points outside of the solid. B-rep models can represent a wide class of objects but the data structure is complex, and can require a large space for a complex object.

Normally, a face is a bounded region of a planar, quadratic, toroidal, or sculptured surface. The bounded region of the surface that forms the face is represented by a closed curve that lies on the surface. A face can have several bounding curves to represent holes in a solid. The bounding curves of faces are represented by edges. The portion of the curve that forms the edge is represented by two vertices.

The description of faces varies. For example, a planar face can be represented in many different ways: using an analytical equation, a parametric equation, a normal vector and a point on the surface, etc. In addition, a closed 3D boundary which lies on the surface is needed to define the face. In conventional CAD systems, the use of faces is usually restricted to the quadrics like cones, cylinders, spheres, etc. Modern CAD systems use a variety of formats to represent sculptured faces. One of the most common is the B-Spline representation [19], a category of surfaces employing parametric polynomials.

The B-Spline surface is a collection of B-Spline curves, i.e. the tensor product of two curves defined by two parameters. The surface is defined as the set of points obtained by evaluating Eq. (1) for all parameter values of u and v between some $u_{\min}$ and $u_{\max}$, and $v_{\min}$ and $v_{\max}$:

$$S(u, v) = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \sum_{i=1}^{n} \sum_{j=1}^{m} N_i^k(u) N_j^l(v) P_{i,j} \tag{1}$$

where the k, l are the orders of the B-Spline surface in both directions, and $P_{i,j}$ is the array of $n \times m$ control points $(x_{i,j}, y_{i,j}, z_{i,j})$. Term $N_i^k(u)$ represents the polynomial B-spline basis functions of degree $k-1$ in u parameter direction, and $N_j^l(v)$ the basis functions of degree $l-1$ in v direction. A user creates a B-Spline surface by defining the order and drawing only the control points. The shape of the surface is modified by moving one or several control points.

3. Overview of GEREP

Fig. 1 shows the general layout of GEREP. It has two separate geometry class libraries, one for tracking purposes and another one for the STEP file input. It contains routines to perform the file input from STEP files, to access the geometry class libraries, to convert data from the STEP geometry class library to the tracking geometry class library, and to perform particle tracking in boundary models. GEREP computes the closest intersection point of a detector model and a given trajectory. The B-Rep solids included in a model are currently limited to B-Spline or planar faces.

GEREP reads a detector model from a STEP file written according to the STEP application protocol 203 (AP-203) [20] and loads it as object instances into the STEP geometry class library. After this, the model is converted to the tracking geometry class library, and the tracking performed. The geometrical information remains the same: no data are lost during the conversion. GEREP contains

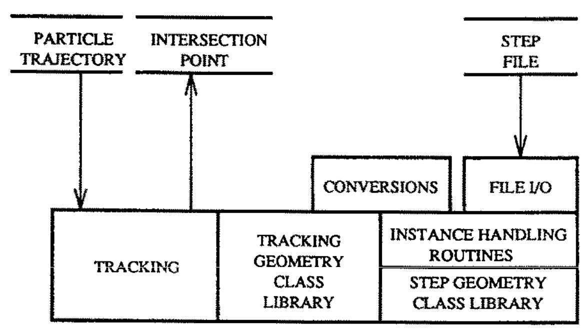

Fig. 1. The main components of GEREP.

also additional classes, like the bounding box class, which is needed at run-time and computed from the original information during the initialization phase.

4. Tracking algorithm

Several methods exist to compute intersection points of a B-Spline surface and a trajectory. We have chosen the clipping method explained in detail in Refs. [15,21]. In this method, the approach is to clip away iteratively regions of the B-Spline surface which are found not to intersect with the trajectory. The method offered us the possibility to develop a so called preliminary intersection point technique. This technique was then applied to the tracking algorithm.

The GEREP algorithm is based on the bounding box technique [22] and the preliminary intersection point technique. A bounding box is a box which has the faces parallel to the coordinate planes and contains the object. The bounding box technique is normally used for the fast initial determination of a possible intersection of an object and a trajectory, using the bounding box of the object instead of the object itself for the intersection computation. It is also used for the initial determination of the closest object. A preliminary intersection point is an approximated point where the intersection may occur. It is a middle point of a region which is defined using the clipping method. The preliminary intersection point technique is used for the initial determination of the closest intersection point when the bounding box of the closest object overlaps with bounding boxes of other objects.

The tracking algorithm is the following:

1) The bounding boxes for solids and for the faces of each solid are computed.

2) The trajectory is intersected first with the solid bounding boxes (i.e. the bounding box test) and the closest solid is then considered.

3) The bounding box test is then performed with each face bounding box of that solid. If the trajectory does not intersect any of the face bounding boxes, there is no intersection with this solid and the next closest one is taken under process. Otherwise, the closest face is considered. If there exists multiple intersections in one face, the intersections are isolated in individual patches by splitting [23] the face.

4) The preliminary intersection points are computed, and the closest of these points is checked to be closer than the next closest face bounding box of the solid being considered. If not, the next closest face is considered and the preliminary intersection points computed. This continues until the closest preliminary intersection point is closer than the next closest unprocessed face bounding box.

5) The point is checked to be closer than the next closest solid bounding box. If it is not, next closest solid is considered.

6) When the closest preliminary intersection point is closer than the next closest unprocessed bounding boxes of faces and solids, the final intersection point is computed by clipping the face until the parameter range is within the defined tolerance.

5. Implementation

GEREP was implemented to be very modular and easy to understand in order to allow in future other collaborators to participate in the development and implementation. At the stage, further code optimization to gain speed in tracking had to be given less importance.

The STEP file input and the STEP geometry class library were implemented using the NIST (US National Institute of Standards and Technology) STEP toolkit [24]. The object classes in the STEP geometry library were created by compiling the STEP AP-203 schema using Fed-X [25]. The implementation of all the classes was performed so that the data and the methods manipulating that data (e.g. particle tracking) are located in the same class.

Solids in the STEP geometry library are retrieved using the *InstanceManager* class of the NIST toolkit. They are decoded in a recursive way and the geometrical information and the topology converted into the tracking geometry library. Each of the classes in the tracking geometry library have the conversion routines as member functions.

6. Results

We demonstrate the efficiency of the tracking algorithm using four different test cases. In the first case, shown in Fig. 2, a trajectory is defined so that it intersects a 4th order B-Spline face twice. In the second case, shown in

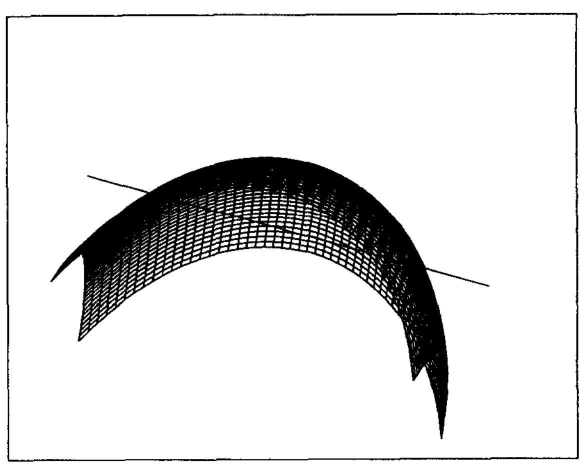

Fig. 2. Two intersections of a trajectory and a 4th order B-Spline face.

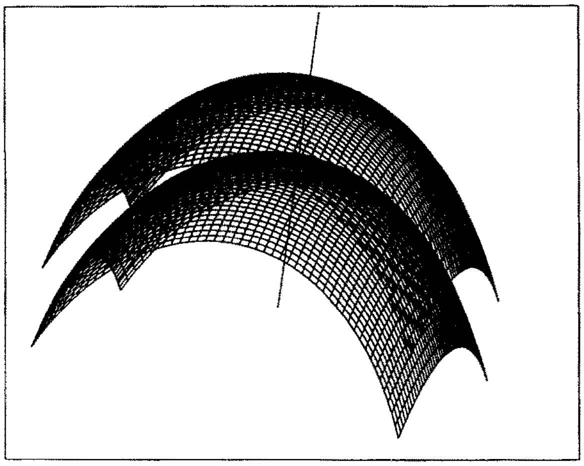

Fig. 3. Two intersections of a trajectory and two 4th order B-Spline faces.

Fig. 3, the trajectory is defined so that it intersects two 4th order faces, once each. The faces are arranged in such a way that their bounding boxes overlap with the preliminary intersection points so that the bounding box technique can not be used to determine the closest intersection point. The third and fourth cases are equivalent to the second case but the number of faces is 3 and 4, respectively. The GEREP version 1 computes all intersections. The version 2 uses the tracking algorithm and computes only the closest intersection. The results are shown in Table 1. The platform used in the tests was Sun SPARC2 with SUN4M running SUN-OS 4.1.3.

7. Conclusions

We have described an application, designed for the GEANT 4 toolkit, which is able to perform particle tracking in boundary models constructed in CAD systems. An algorithm for particle tracking was presented and test results shown.

The new particle tracking algorithm for complex B-Spline faces of B-rep models proved to be efficient enough for the target application. The particle tracking in B-Rep models in general is not as fast as in the current version of GEANT based on CSG, but the advantage is that very complex detector models can be accurately described, where needed. Moreover, when speed is not an issue, it is possible to perform particle tracking directly in models constructed in CAD systems. The computational cost of the intersection computations is caused by the mathematical complexity of B-Spline surfaces.

Table 1
Efficiency of the tracking algorithm

	Time [ms]			
	case 1	case 2	case 3	case 4
GEREP version 1	20	12	19	25
GEREP version 2	14	8	10	12
Number of intersections	2	2	3	4

At the moment, GEREP supports B-Rep models constructed using B-Spline and planar faces. The implementation for supporting quadric and toroidal faces is straightforward.

Code level optimization should further improve the execution speed. In our work, the NIST STEP toolkit was chosen since it is public domain and implemented in C++. The programming language used in this work, C++, is in many ways suitable for large applications, and it is also suitable for implementing STEP applications, as pointed out in Ref. [26].

GEREP will allow us to import sophisticated B-Rep detector models constructed in CAD systems into GEANT 4. The new STEP standard facilitates the data exchange, and GEREP will provide a general solution to transfer detector models from CAD systems into GEANT using STEP files. GEREP will be fully integrated in the GEANT 4 toolkit.

Acknowledgements

We are grateful to our colleagues at CERN for their help. In particular we would like to thank M. Ferran, S. Giani, N. Høimyr and A. Osborne for their help and advice. We owe our thanks to Samy Chemli at Matra Datavision for his help.

Financial support is acknowledged from the Institute of Particle Technology (HTI) in Espoo, Finland, the Institute for High Energy Physics (SEFT) in Helsinki, Finland, the Finnish Ministry of Trade and Education, and from CERN in Geneva, Switzerland.

References

[1] See for instance CMS: C.J. Seez, Nucl. Instr. and Meth. A 344 (1994) 1.
[2] R. Brun et al., GEANT Detector Description and Simulation Tool, CERN Program Library, W5013, 1994.
[3] H. Høimyr and J. Vuoskoski, Nucl. Instr. and Meth. A 357 (1995) 120.
[4] M. Dach and J. Vuoskoski, Nucl. Instr. and Meth. 368 (1996) 775.
[5] M.E. Mortenson, Geometric Modeling (Wiley, 1985).
[6] M. Mäntylä, An Introduction to Solid Modeling (Computer Science Press, 1988).
[7] E.G. Schlechtendahl (ed.), CAD Data Transfer for Solid Models, ESPRIT Research Reports, Project 322, CAD Interfaces, Vol. 3 (Springer, 1989).
[8] J. Ahvenainen, T. Oksakivi and J. Vuoskoski, Nucl. Instr. and Meth. A 361 (1995) 519.
[9] A. Dellacqua et al., GEANT 4: an Object-Oriented toolkit for simulation in HEP, CERN/DRDC/P58, 1994.

[10] ISO 10303, Industrial Automation Systems and Integration – Product Data Representation and Exchange – Part 1: Overview and Fundamental Principles, ISO TC 184/SC4, 1994.
[11] J. Owen, STEP: An Introduction (Information Geometers Ltd, 1994).
[12] ISO 10303, Industrial Automation Systems and Integration – Product Data Representation and Exchange – Part 21: Clear Text Encoding of the Exchanging Structure, ISO TC 184/SC4, 1994.
[13] J. Rumbaugh et al., Object-Oriented Modeling and Design (Prentice-Hall, 1991).
[14] K. Weiskamp and B. Flaming, The Complete C++ Primer (Academic Press, 1992).
[15] J. Sulkimo and J. Vuoskoski, GEREP, an Application for Particle Tracking in Boundary Models, Helsinki University of Technology, TKK-F-A743, 1995.
[16] ISO 10303, Industrial Automation Systems and Integration – Data Representation and Exchange – Part 11: Description Methods: The EXPRESS Language Reference Manual, ISO TC 184/SC4, 1994.
[17] D.A. Schenck and P.R. Wilson, Information Modeling: The EXPRESS way (Oxford University Press, 1994).
[18] ISO 10303, Industrial Automation Systems and Integration – Product Data Representation and Exchange – Part 42: Geometric and Topological Representation, ISO TC 184/SC4, 1994.
[19] R. Bartels, J. Beatty and B. Barsky, An introduction to SPLINES for use in Computer Graphics & Geometric Modeling (Morgan Kaufmann, 1987).
[20] ISO 10303, Industrial Automation Systems and Integration – Product Data Representation and Exchange – Application Protocol 203: Configuration Controlled Design, ISO TC 184/SC4, 1992.
[21] T. Nishita, T. Sederberg and M. Kakimoto, Computer Graphics 24(4) (1990) 337.
[22] A. Glassner, The theory and practice of ray tracing, Eurographics '91 (1991).
[23] T. Dokken, Computer Aided Geometric Design 2(1/3) (1985) 189.
[24] K.C. Morris, D. Sauder and S. Ressler, Validation Testing System: Reusable Software Component Design, National Institute of Standards and Technology, Gaithersburg, MD, NISTIR 4937, 1992.
[25] S.N. Clark and D. Libes, Fed-X: The NIST Express Translator, National Institute of Standards and Technology, Gaithersburg, MD, NISTIR 4822, 1992.
[26] T. Rando and L. McCabe, Computer Standards & Interfaces 16 (1994) 331.