Nuclear Engineering and Technology 49 (2017) 1189–1198

Contents lists available at ScienceDirect

Nuclear Engineering and Technology

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

Original Article

Particle tracking acceleration via signed distance fields in direct-accelerated geometry Monte Carlo

Patrick C. Shriwise*, Andrew Davis, Lucas J. Jacobson, Paul P.H. Wilson

CNERG Research Group, University of Wisconsin - Madison, Madison, WI, USA

ARTICLE INFO

Article history:
Received 3 June 2017
Accepted 7 August 2017
Available online 26 August 2017

Keywords:
Monte Carlo
Radiation Transport
CAD
DAGMC

ABSTRACT

Computer-aided design (CAD)-based Monte Carlo radiation transport is of value to the nuclear engineering community for its ability to conduct transport on high-fidelity models of nuclear systems, but it is more computationally expensive than native geometry representations. This work describes the adaptation of a rendering data structure, the signed distance field, as a geometric query tool for accelerating CAD-based transport in the direct-accelerated geometry Monte Carlo toolkit. Demonstrations of its effectiveness are shown for several problems. The beginnings of a predictive model for the data structure's utilization based on various problem parameters is also introduced.

1. Introduction

The direct-accelerated geometry Monte Carlo (DAGMC) [1] toolkit provides the capability for robust radiation transport on CAD geometries. This allows nuclear analysis to be performed on the same models that are used in other engineering domains. The toolkit also has the capability to convert native Monte Carlo models to computer-aided design (CAD) for further development, modification, and analysis.

DAGMC tracks particles on discretized representations of analytic CAD surfaces as triangle meshes, such that all points on any given triangle are within a specified tolerance of the analytic representation. This mesh representation of the model is then stored in the mesh oriented database (MOAB) [2] in which geometric queries such as next surface intersection, point containment, and nearest to boundary are performed.

It has been shown that DAGMC can achieve robust transport equal to that of the native codes with which it communicates [3], but that it currently takes much longer [4]. This additional time occurs despite the acceleration techniques that DAGMC uses to avoid searching over the large number of triangles ($\sim 10^6$–10^7). Those acceleration techniques, implemented within MOAB, will be briefly discussed to motivate this work, which aims to further accelerate geometry queries within DAGMC.

To begin, this article gives a brief overview of the employed acceleration techniques in DAGMC for particle tracking. Next, the signed distance field data structure intended for rendering dynamic surfaces is introduced. The remaining content describes this data structure's adaptation for use in accelerated particle tracking for CAD-based Monte Carlo simulations. The application of this data structure and associated tracking methods are then demonstrated as a new method for particle tracking acceleration in DAGMC. Included demonstrations of this method are shown to greatly reduce run times in a few test problems for commonly encountered conditions during transport. Next, some studies involving the data structure's effectiveness for various problem parameters are investigated and discussed. Finally, conclusions, current limitations, and future extensions of this work are discussed.

2. Background

2.1. Acceleration techniques in DAGMC

Particle tracking in DAGMC relies on the capability to robustly perform geometric queries on the triangle surface meshes which represent the problem geometry. The most common form of geometric query in particle tracking is a next surface crossing query. This is called to determine if a particle within the current cell will reach its next event location or cross a surface of the cell, potentially entering a new medium with different physical characteristics, based on the particle's current position and trajectory. Geometry kernels in native Monte Carlo codes perform analytic calculations for the nearest intersection along the particle's trajectory from its current location. In DAGMC, a similar operation is performed, but on the triangle surfaces that compose the current cell (or volume).

* Corresponding author.
E-mail address: shriwise@wisc.edu (P.C. Shriwise).

http://dx.doi.org/10.1016/j.net.2017.08.008

The determination of the nearest triangle intersection with a trajectory from a starting position is a well-researched problem in the field of ray tracing. As such, a ray tracing data structure, the bounding volume hierarchy (BVH) [5], is applied within MOAB to accelerate these queries. More specifically, MOAB utilizes an oriented bounding box (OBB) BVH [6,7] to bound sets of triangles so that they may be rapidly excluded from the query in large groups via intersection checks with these OBBs.

This process begins by bounding all triangles with a single OBB. The triangles are then split into two sets based on which side of the splitting plane they are on. The splitting plane is chosen as the current OBBs median plane which divides the number of triangles into equal parts. OBBs are then generated for these subsets of triangles. Concurrently, the new OBBs are linked to the original OBB as children. This process recurs until OBBs are bounding small sets of triangles. A hierarchical structure is created along the way which can then be traversed to rapidly isolate sets of triangles in space. A two-dimensional (2D) visualization of this process using curve segments rather than triangles is provided in Fig. 1. Queries are satisfied using data structure by performing ray-box intersection checks with the root OBB. If an intersection with the root OBB is found, then the children of that OBB will also be checked for an intersection. If the OBB has no children, it is considered a leaf node and the triangles it bounds are then checked for an intersection with the ray. During traversal, each time an OBB is missed by the ray, all triangles bounded by that box are ignored. It is in this way that the search for an intersection is rapidly limited to a small number of triangles with a high chance of intersecting with the ray. This results in an algorithm with $O(log(N))$ complexity where N is the number of entities (triangles) in the data structure. MOAB's OBB tree can be used to satisfy point containment and closest intersection queries as well. Many variations of this data structure exist, and research is in progress to further accelerate particle tracking with these methods [4].

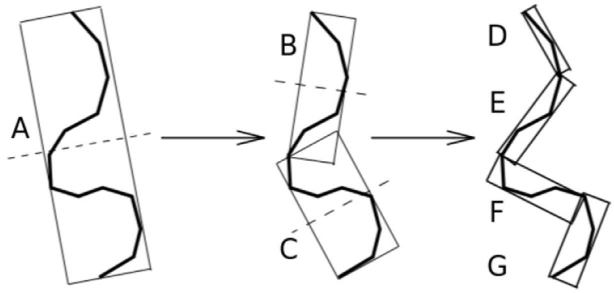

Fig. 1. Two-dimensional example of a bounding volume hierarchy using oriented bounding boxes on a discretized curve.

2.2. Signed distance fields and implicit surfaces

Signed distance fields are commonly derived from implicit surface functions, and variations on these functions are known as level-set functions. Both offer a rich and versatile representation of closed manifolds that can be used for modeling, simulation, and rendering. The constructive solid geometry (CSG) representations seen in native Monte Carlo codes are usually formed from Boolean combinations of predefined implicit surfaces at their core. While these predefined surfaces do not give the freedom of model creation and manipulation found in many CAD systems, important geometric information required for visualization and simulation can be readily recovered from these implicit surfaces which may be of value in CAD-based radiation transport simulations.

Implicit surface functions are multivariate functions defined over the R^3 domain as:

$$\Omega\left(R^3\right) \rightarrow R \tag{1}$$

where an isocontour of value, v, of the implicit surface can be described as

$$\Omega(\vec{x}) - v = 0 \tag{2}$$

for all points $\vec{x}$ satisfying that equation. For simplicity, the surface isocontour value is typically defined as 0.

By recognizing that the magnitude of $\Omega(\vec{x})$ is in fact a minimum interface distance function, one can construct a signed distance function, $SDV(\vec{x})$, using the isocontour representation and the magnitude of the function as seen in Eq. 3 [8].

$$SDV(\vec{x}) = |\Omega(\vec{x})| \tag{3}$$

Signed distance function generation from implicit surfaces is a particularly valuable property of implicit surfaces. A signed distance function, $SDV(\vec{x})$, meets the following requirements for any point $\vec{x}$:

- $SDV(\vec{x}) = 0$ for all $\vec{x}$ on the surface boundary,
- $SDV(\vec{x}) < 0$ for all $\vec{x}$ inside the surface boundary, and
- $SDV(\vec{x}) > 0$ for all $\vec{x}$ outside the surface boundary.

Implicit surfaces and level-set methods are easily extended to represent dynamic geometries by including a time dependence in the function, making them powerful tools for populating signed distance fields in simulation and rendering of fluids, smoke, fire, etc. In these applications the data structure is populated with signed distance values for a given time in the rendering. The signed distance field can then be used to determine point containment queries and trace rays at any time via a method in which the ray length is repeatedly clipped using signed distance values to approach a surface in a process called ray marching [9]. This work establishes the potential use of signed distance fields in static CAD-based Monte Carlo radiation transport, with possible future extension to dynamic geometries.

3. Signed distance field implementation in DAGMC

As an initial implementation, one signed distance field is generated for each volume in DAGMC with extents matching the axis-aligned bounding box of the volume. The signed distance field is represented as a uniform structured mesh with a signed distance value at each vertex in the mesh as indicated in Fig. 2. (Altering the sign convention when populating the data structure rather than incurring the additional computational cost of altering the sign of values for each operation is preferable.) It is mentioned above that signed distance fields are typically generated using an implicit, analytic representation, but a suitable data structure for populating the structured mesh with signed distance values is already in place in the form of DAGMC's bounding volume hierarchy. It is a more straightforward process to simply use DAGMC's current closest to location algorithm to generate signed distance values than to create an implicit surface approximation of the triangle mesh. Though the latter may be faster, performance improvements didn't seem to be impeded greatly by using this method to populate the data structure. This method also maintains a consistency between the intersections found by the ray tracing kernel and the signed distance field values.

DAGMC's closest to boundary algorithm returns, among other pieces of information, the nearest intersection location and the