5XO —

LA-UR- 00-1065

Los Alamos
NATIONAL LABORATORY
research note
Applied Physics Division
X-6: Transport Methods Group

To/MS: Distribution
From/MS: Todd J. Urbatsch/X-6, D409
Thomas M. Evans/X-6, D409
Phone/FAX: (505)667-3513,665-3677/665-5538

Symbol: X-6:00-??? (U)
Date: February 25, 2000

Subject: Monte Carlo Particle Transport on a General Hexahedral Mesh

Executive Summary

We are interested in tracking Monte Carlo particles on a general, hexahedral mesh whose cell faces are bilinear surfaces. We derive the solution for intersecting a ray with a parameterized bilinear surface comprised of four (non-coplanar) nodes. The matrix is nonlinear in the two parameters of the bilinear surface, but can nevertheless be solved analytically. We discuss the algorithm for tracking in a general hexahedral mesh and some ways to speed up the process. We also discuss sampling a position on a bilinear face and in a general hexahedral volume.

1. Introduction

We are looking to extend the mesh tracking capabilities of MILAGRO [1], a 3-D Implicit Monte Carlo (IMC) code for thermal radiative transfer, beyond its current orthogonal, structured mesh capability to a node based, unstructured, hexahedral mesh. There are many ways for us to track on this mesh given that data are stored on the nodes. Since the transport in MILAGRO is based on cell-centered quantities, one option is to track on node-centered cells constructed from pieces of all the original cells about the node. Unfortunately, although each node-centered cell would have constant data, each node-centered cell could be an ugly, non-convex, 24-sided cell. We could also track on a sub-mesh—on what amounts to the corners of the original mesh cells—which would be defined by the cell-centers and face-centers of the original cells. These corner cells will still be hexahedral like the original cells and there will be eight times as many of them, but we can assume constant data within the cell. A more advanced possibility is to perform the source and tallies using the eight basis functions within each cell. We opt to start with the most straightforward approach: to track on the original hexahedral mesh, where the data are homogenized over the cell.

Tracking on a hexahedral mesh requires sampling positions on a bilinear face, sampling positions in a trilinear volume, and calculating the distance from a particle to a bilinear surface along the particle's path. We present the equations for the bilinear surface and solve for a ray's intersection distance and surface point. The resulting matrix equation is nonlinear in the two bilinear parameters, but can be solved analytically. We conclude by discussing how to sample positions in a general hexahedral mesh.

2. Bilinear Surface

A bilinear surface is defined by four non-coplanar points

$$P_1 = (x_1, y_1, z_1) \ , \quad (1a)$$

$$P_2 = (x_2, y_2, z_2) \ , \quad (1b)$$

$$P_3 = (x_3, y_3, z_3) \ , \quad (1c)$$

$$P_4 = (x_4, y_4, z_4) \ , \quad (1d)$$

and the following "serendipity" parametric representation,

$$x = [1-u, 1+u] \begin{bmatrix} x_1 & x_4 \\ x_2 & x_3 \end{bmatrix} \begin{bmatrix} 1-v \\ 1+v \end{bmatrix} \ , \quad (2a)$$

$$y = [1-u, 1+u] \begin{bmatrix} y_1 & y_4 \\ y_2 & y_3 \end{bmatrix} \begin{bmatrix} 1-v \\ 1+v \end{bmatrix} \ , \quad (2b)$$

$$z = [1-u, 1+u] \begin{bmatrix} z_1 & z_4 \\ z_2 & z_3 \end{bmatrix} \begin{bmatrix} 1-v \\ 1+v \end{bmatrix} \ , \quad (2c)$$

where x, y, and z are the Cartesian coordinates on the bilinear surface and $u \in [-1, 1]$ and $v \in [-1, 1]$ are the independent parameters [2,3]. The parametric representation can be thought of as interpolating in u along the lines from P_1 to P_2 and P_4 to P_3, as shown in Fig. 1, then in v between those two lines.

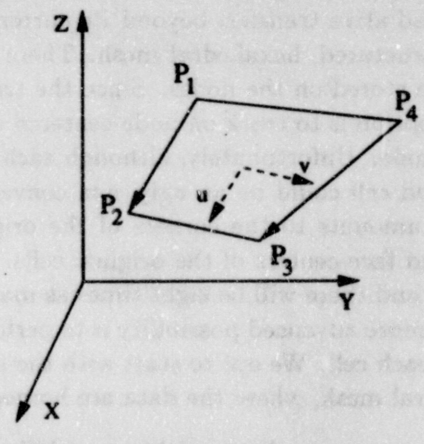

FIG. 1: A bilinear surface in 3-D with control points ordered as shown that can be represented parametrically in (u, v).

Expanding the parametric representation, we obtain a matrix equation in u, v, and uv

$$\mathbf{F}(u, v) = \frac{1}{4}[\mathbf{A} + \mathbf{B}u + \mathbf{C}v + \mathbf{E}uv] \ , \quad (3)$$

where **A**, **B**, **C**, and **E** are 3×1 matrices defined as follows:

$$\mathbf{A} = \begin{bmatrix} x_1 + x_2 + x_3 + x_4 \\ y_1 + y_2 + y_3 + y_4 \\ z_1 + z_2 + z_3 + z_4 \end{bmatrix} , \quad \mathbf{B} = \begin{bmatrix} -x_1 + x_2 + x_3 - x_4 \\ -y_1 + y_2 + y_3 - y_4 \\ -z_1 + z_2 + z_3 - z_4 \end{bmatrix} , \tag{4}$$

$$\mathbf{C} = \begin{bmatrix} -x_1 - x_2 + x_3 + x_4 \\ -y_1 - y_2 + y_3 + y_4 \\ -z_1 - z_2 + z_3 + z_4 \end{bmatrix} , \quad \mathbf{E} = \begin{bmatrix} x_1 - x_2 + x_3 - x_4 \\ y_1 - y_2 + y_3 - y_4 \\ z_1 - z_2 + z_3 - z_4 \end{bmatrix} . \tag{5}$$

3. Intersecting a Ray with a Bilinear Surface

The most computationally intensive task in tracking on a general hexahedral mesh is determining whether a particle in a cell intersects a surface of the cell and, if it does, the distance to the intersection. Let us consider a particle at position **R** traveling in direction **Ω**, where

$$\mathbf{R} = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} , \quad \mathbf{\Omega} = \begin{bmatrix} \Omega_x \\ \Omega_y \\ \Omega_z \end{bmatrix} . \tag{6}$$

The equation for the particle at a distance d along its trajectory is

$$\mathbf{P}(d) = \mathbf{R} + \mathbf{\Omega} d . \tag{7}$$

To find where the particle intersects the bilinear surface, we set their respective equations equal to each other:

$$\mathbf{P}(d) = \mathbf{F}(u, v) , \tag{8}$$

$$\mathbf{R} + \mathbf{\Omega} d = \frac{1}{4}[\mathbf{A} + \mathbf{B}u + \mathbf{C}v + \mathbf{E}uv] , \tag{9}$$

$$0 = (\mathbf{A} - 4\mathbf{R}) + \mathbf{B}u + \mathbf{C}v + \mathbf{E}uv - 4\mathbf{\Omega} d , \tag{10}$$

$$0 = (\mathbf{A} - 4\mathbf{R}) + \mathbf{B}u + \mathbf{C}v + \mathbf{E}uv + \mathbf{D}d , \tag{11}$$

giving us a matrix of three equations in the three unknowns (u, v, d), where

$$\mathbf{D} = -4\mathbf{\Omega} = -4 \begin{bmatrix} \Omega_x \\ \Omega_y \\ \Omega_z \end{bmatrix} . \tag{12}$$

Unfortunately, the nonlinear uv term means that we cannot solve this equation with a simple 3×3 matrix inversion.

The literature from the ray tracing industry (computer graphics, movie rendering, etc.) suggests either to perform a multidimensional Newton iteration or to represent the ray as two intersecting planes and perform a Newton iteration on the set of two equations formed by the intersection of the planes with the surface [2]. This seems reasonable except that the ray tracing industry generally represents objects with at least cubic surfaces or Bezier splines; the literature hardly ever mentions simple, bilinear surfaces. However, a writeup on the web mentioned using a bilinear surface to approximate a higher order surface in order to eliminate some costly surface intersection calculations [4]. The account mentioned in passing that bilinear surface intersections could be solved analytically. Obviously an analytic solution would be preferable to a Newton iteration because it would immediately produce both intersections. A Newton iteration could require extra work to check that the correct intersection was found.

We spoke with John Turner, a former colleague who now works for Bluesky Studios, and he confirmed that bilinear surfaces patched together are not smooth enough for the rendering industry because of the discontinuous normals at boundaries. Furthermore, he said that complicated surfaces are often tessellated into several simpler surfaces. He also verified that intersections with bilinear surfaces could be solved analytically; he was fairly certain that it reduces to a quadratic equation.

Jim Morel, X-6, provided us with crucial insight into the analytical solution. We can write Eq. 11 as an equation in the linear terms,

$$\mathbf{M}\begin{bmatrix} u \\ v \\ d \end{bmatrix} = -\mathbf{E}uv + (4\mathbf{R} - \mathbf{A}) \quad . \tag{13}$$

where

$$\mathbf{M} = \left[\begin{bmatrix} \mathbf{B} \end{bmatrix}\begin{bmatrix} \mathbf{C} \end{bmatrix}\begin{bmatrix} \mathbf{D} \end{bmatrix}\right] = \begin{bmatrix} B_1 & C_1 & D_1 \\ B_2 & C_2 & D_2 \\ B_3 & C_3 & D_3 \end{bmatrix} \quad . \tag{14}$$

Solving for the linear terms of the unknowns, we have

$$\begin{bmatrix} u \\ v \\ d \end{bmatrix} = -\mathbf{M}^{-1}\mathbf{E}uv + \mathbf{M}^{-1}(4\mathbf{R} - \mathbf{A}) \quad , \tag{15}$$

where each linear term is a function of uv and a constant.

If we make the following definitions,

$$Q_i = -\sum_{j=1}^{3} M_{ij}^{-1} E_j \quad , \tag{16}$$

$$S_i = \sum_{j=1}^{3} M_{ij}^{-1}(4R_j - A_j) \quad , \tag{17}$$

we can solve for u and v (on the LHS of the Eq. 15)

$$u = Q_1 uv + S_1 \quad , \quad v = Q_2 uv + S_2 \quad . \tag{18}$$

Then, if we multiply them together, we end up with an equation that is quadratic in (uv)

$$uv = Q_1 Q_2 (uv)^2 + (Q_1 S_2 + Q_2 S_1) uv + S_1 S_2 \quad , \tag{19}$$

$$0 = Q_1 Q_2 (uv)^2 + (Q_1 S_2 + Q_2 S_1 - 1) uv + S_1 S_2 \quad . \tag{20}$$

Thus, using the quadratic formulat, we have two solutions for (uv) that can be used in Eq. 15 to find the values of (u, v, d). The final desired intersection solution must have real values of $(uv) \in [-1, 1]$. If both values of (uv) are real and positive and $|uv| \leq 1$, then the final solution for this particular surface is the one with the smallest, positive value of d,

$$d = Q_3 uv + S_3 \quad . \tag{21}$$

4. Intersection Solution Details

At the beginning of each time step, we must calculate the inverse matrix M from Eq. 14 and store the nine values for each surface of our hexahedral mesh. We also need to calculate the three Q values from Eq. 16 and store them. We could calculate and store the three A values, but they are just sums of the four control point locations for each dimension, so they could be easily calculated on the fly.

For each potential surface intersection, we carry out the following calculations:

1. **Initialize $d_{min} = \infty$ and $u, v \notin [-1, 1]$**
2. **Calculate S_1 and S_2 from Eq. 17**
3. **Solve for $b = Q_1 S_2 + Q_2 S_1 - 1$**
4. **Solve for $4ac = 4(Q_1 Q_2)(S_1 S_2)$**
5. **If $b^2 > 4ac$, there are real roots, then**
   - **calculate $uv = (-b \pm \sqrt{b^2 - 4ac})/(2a)$ and, for each,**
     - **calculate S_3**
     - **calculate $d = Q_3 + S_3$**
     - **if $d < d_{min}$, then $d_{min} = d$, save index, and save uv**

Once we have the surface with the minimum intersection distance from all contending surfaces, we may calculate the corresponding u and v from Eqs. 18 and use them in Eqs. 2a through 2c with the appropriate coordinates to find the exact location of intersection.

5. Reducing the Ray/Surface Intersection Work

Calculating the distances to each of the six sides of a hexahedron can be a considerable amount of work to do for each particle path. We discuss here two ways to reduce the workload.

One common way to eliminate some of the intersection work is for the particle to maintain a minimum-distance-to-boundary. If the particle is in the middle of a relatively thick cell, and it is going to collide without a chance of ever hitting a surface of the cell, the intersection calculations are not required.

Another common way to reduce work is to only perform the intersection calculation for surfaces toward which the particle is pointing. If the cost of eliminating surfaces from intersection consideration is less than the intersection calculations, then time savings can be realized.

Since the surface normals of a bilinear surface are not constant, both of these time-saving measures could be just as expensive as the intersection calculation. Thus, we are motivated to define an "inner box" that is made up of six simple planes and that is as large as possible while still residing fully inside the hexahedral cell. Then, the equations of these simple planes, or "inner planes," allow trivial determinations of minimum-distance-to-"inner plane," whether a particle is inside an "inner plane," and whether a particle is pointing toward an "inner plane."

We propose to construct the "inner planes" for each of a cell's six bilinear surfaces in the following way:

1. Calculate the outward normal, $\mathbf{n}$, at the surface center.
2. Define a plane at the surface center with normal $\mathbf{n}$.
3. Slide the plane inward along the negative direction of the outward normal until all four control points of the surface lie on or outside the plane.

Calculating the normal at the center of a bilinear surface is rather easy if we consider the parametric expression of the surface, Eq. 3. The center of the bilinear surface, $\mathbf{P}_{fc}$, resides at $u = 0$ and $v = 0$. The normal can be found from the cross product of two vectors lying on the surface at the surface center:

$$\left.\frac{\partial \mathbf{F}(u,v)}{\partial u}\right|_{v=0} = \mathbf{B} \ , \quad \left.\frac{\partial \mathbf{F}(u,v)}{\partial v}\right|_{u=0} = \mathbf{C} \ . \tag{22}$$

Thus, the unit normal is

$$\mathbf{n} = \frac{\mathbf{B}\times\mathbf{C}}{||\mathbf{B}\times\mathbf{C}||} = \frac{1}{||\mathbf{B}\times\mathbf{C}||}\left[(B_2C_3 - B_3C_2)\mathbf{i} + (B_3C_1 - B_1C_3)\mathbf{j} + (B_1C_2 - B_2C_1)\mathbf{k}\right] \ , \tag{23}$$

and the equation of the plane at the face center is

$$F(x,y,z) = \mathbf{n}\cdot(\mathbf{P} - \mathbf{P}_{fc}) = n_x x + n_y y + n_z z - (\mathbf{n}\cdot\mathbf{P}_{fc}) = 0 \ . \tag{24}$$

The normal at the center of the bilinear surface is the right-handed normal about the counter-clockwise ordering of the control points, so we do not know whether it is an outward or inward normal relative to a particular cell. A simple way to check is to see whether the cell center, $\mathbf{P}_{cc}$, is inside or outside of the plane at the surface center. If the cell center is outside the plane, i.e., $\mathbf{n}\cdot(\mathbf{P}_{cc} - \mathbf{P}_{fc}) > 0$, then the normal must be reversed, $\mathbf{n} = -\mathbf{n}$.

Now, to find the "inner plane" we substitute each of the four control points, $\mathbf{P}_i$, for $\mathbf{P}$ in the equation of the plane at the face center, Eq. 24, and find the control point index, m, with the minimum F. Then the equation of the "inner plane" is

$$F(x,y,z) = \mathbf{n}\cdot(\mathbf{P} - \mathbf{P}_m) = n_x x + n_y y + n_z z - (\mathbf{n}\cdot\mathbf{P}_m) = 0 \ . \tag{25}$$

These time-saving approaches require storing six indexed equations of a plane for each cell. During particle transport, if a particle is pointing away from an "inner plane" ($\mathbf{\Omega}\cdot\mathbf{n} < 0$), and the particle resides inside the "inner plane" ($F(x,y,z) < 0$), then the corresponding bilinear surface can be eliminated from the intersection calculation.

6. Sampling on a General Hexahedral Mesh

Sampling Monte Carlo source particles requires the ability to sample positions on a bilinear face and in a general hexahedral cell.

Sampling uniformly on a bilinear surface only requires sampling u and v uniformly,

$$u = 2\xi_1 - 1 \ , \ v = \xi_2 - 1 \ , \tag{26}$$

where $\xi_i \in (0,1)$ are random numbers. Plugging u and v into Eqs. 2a–2c yields the location on the surface.

In order to sample on a bilinear surface according to specific values, S_i, at the four control points, we resort to the interpretation of the parametric interpolation, Fig. 1. Assuming that it is sufficient to separate the two independent parameters, (u,v), we consider each in turn. In the u dimension, for example, we have two equations for the two lines, $\overline{S_1S_2}$ and $\overline{S_4S_3}$,

$$\overline{S_1S_2} = \frac{1}{2}[(1-u)S_1 + (1+u)S_2] \ , \ u \in [-1,1] \ , \tag{27}$$

$$\overline{S_4S_3} = \frac{1}{2}[(1-u)S_4 + (1+u)S_3] \ , \ u \in [-1,1] \ . \tag{28}$$

Since we have two equations in one unknown, we average them to obtain

$$S(u) = \frac{1}{4}[(1-u)(S_1+S_4) + (1+u)(S_2+S_3)] \ , \ u \in [-1,1] \ . \tag{29}$$

(This averaging is equivalent to setting $v = 0$ in the parametric representation of the surface.) We can easily sample this linear function (with nonzero intercept) by breaking it into two functions, one with positive slope and one with negative slope, and each with zero intercept [5]. First, considering these two functions as triangles, we sample which of the triangles to further sample according to their relative areas. Then we sample u by

$$u = -1 + 2\sqrt{\xi_2} \text{ for the right triangle, or} \tag{30a}$$

$$u = 1 - 2\sqrt{\xi_2} \text{ for the left triangle .} \tag{30b}$$

Sampling v is done in an analogous manner.

For sampling in the cell volume, we extend the parametric representation from the bilinear surface to the trilinear volume. We represent the general hexahedral mesh cell with three independent parameters, (u,v,w), as shown in Fig. 2. We can think of this three-dimensional parametric representation as an interpolation in $w \in [-1,1]$ between two opposing bilinear faces, which are each interpolated in (u,v) according to Eqs. 2a–2c. The equation for a point, $\mathbf{P}(u,v,w)$, in the volume is,

$$P(u,v,w) = \frac{1}{8}\left[\mathbf{u}^T\mathbf{P}_{1234}\mathbf{v} \ , \ \mathbf{u}^T\mathbf{P}_{5678}\mathbf{v}\right] w \tag{31}$$

$$= \frac{1}{8}\left[[1-u, 1+u]\begin{bmatrix} P_1 & P_4 \\ P_2 & P_3 \end{bmatrix}\begin{bmatrix} 1-v \\ 1+v \end{bmatrix} , [1-u,1+u]\begin{bmatrix} P_5 & P_8 \\ P_6 & P_7 \end{bmatrix}\begin{bmatrix} 1-v \\ 1+v \end{bmatrix}\right]\begin{bmatrix} 1-w \\ 1+w \end{bmatrix} \tag{32}$$

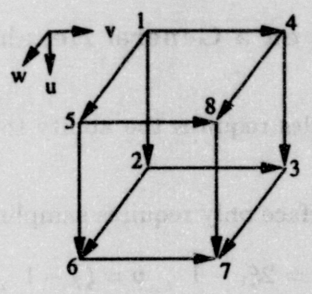

FIG. 2: A trilinear hexahedral mesh cell in 3-D, with control points ordered as shown, that can be represented parametrically in (u, v, w).

$$\begin{aligned} &= 1/8[(1-u)(1-v)(1-w)P_1 + (1+u)(1-v)(1-w)P_2 \\ &\quad +(1+u)(1+v)(1-w)P_3 + (1-u)(1+v)(1-w)P_4 \\ &\quad +(1-u)(1-v)(1+w)P_5 + (1+u)(1-v)(1+w)P_6 \\ &\quad +(1+u)(1+v)(1+w)P_7 + (1-u)(1+v)(1+w)P_8] \ , \end{aligned} \tag{33}$$

where P can be considered as any scalar or vector value, such as temperature, location, or velocity.

Sampling uniformly in volume requires simply sampling u, v, w uniformly in [-1,1] and plugging into one of Eq. 33. To sample according to node values, we again assume that the parameters are separable. In each basis parameter, there exist four equations for lines from the four nodes at the low face (u(or v or w) $= -1$) to the four corresponding nodes at the high face (u(or v or w) $= 1$). We follow the same logic we used for sampling a bilinear surface. For each parameter u, v, w, we perform a face average at the low and high faces, and sample the linear function.

7. Summary

In order to track Monte Carlo particles on a general, 3-D, hexahedral mesh, we must know how to track to bilinear surfaces and how to sample positions on the bilinear faces and trilinear volumes. We have presented the parameterized equations for a bilinear surface and the equation of a particle intersection with a bilinear surface. The intersection equation is nonlinear in the bilinear parameters, but nevertheless has an analytical solution. We have presented the relevant equations, propose algorithms, and offer some ways to reduce the computational effort. We also discussed how to sample positions in a cell volume and on a cell face in a general hexahedral mesh.

8. Acknowledgments

We gratefully acknowledge the pertinent comments and insight of John Turner, Jim Morel, Rob Lowrie, and Grady Hughes.

References

[1] T. M. Evans and T. J. Urbatsch, "MILAGRO: A parallel Implicit Monte Carlo code for 3-d radiative transfer (U)," in *Proceedings of the Nuclear Explosives Code Development Conference*,

(Las Vegas, NV), Oct. 1998. LA-UR-98–4722.

[2] A. S. Glassner, ed., *An Introduction to Ray Tracing*. London: Academic Press, 1989.

[3] O. C. Zienkiewicz and R. L. Taylor, *The Finite Element Method*, vol. 1: Basic Formulation and Linear Problems. London: McGraw-Hill, fourth ed., 1994.

[4] B. Martin, "Ray tracing parametric surfaces." www.cs.utah.edu/~wmartin/1/talk/final.html.

[5] F. Brown and T. Sutton, "Monte carlo fundamentals," Laboratory Report KAPL-4823, kapl, February 1996. UC-32 (DOE/TIC-4500-R75).

Distribution:

Jim Morel, X-6, MS D409
Gordon Olson, X-6, MS D409
Grady Hughes, X-6, MS D409
Mike Gittings, X-2, MS B220
Robert Weaver, X-2, MS B220
Tony Scannapieco, X-3, MS D413
Art Forster, X-5, MS F663
Don Shirk, X-8, MS F663
Don Burton, X-9, MS F663
Eldon Linnebur, X-9, MS F663
Bill Chandler, X-9, MS F663
Forrest Brown, X-10, MS F661
Steven Nolen, X-10, MS F663
Mike Clover, X-11, MS F663
Gary Pfeufer, X-11, MS F663
Kim Simmons, X-11, MS F663
Bob Webster, X-11, MS F663
Todd Adams, X-6, MS D409
Ray Alcouffe, X-6, MS D409
Marv Alme, X-6, MS D409
Larry Auer, X-6, MS D409
Randy Baker, X-6, MS D409
Jon Dahl, X-6, MS D409
Tom Evans, X-6, MS D409
Chris Gesh, X-6, MS D409
Mark Gray, X-6, MS D409
Mike Hall, X-6, MS D409
Henry Lichtenstein, X-6, MS D409
John McGhee, X-6, MS D409
Dimitri Mihalas, X-6, MS D409
Shawn Pautz, X-6, MS D409
Randy Roberts, X-6, MS D409
Kelly Thompson, X-6, MS D409
Scott Turner, X-6, MS D409
Todd Urbatsch, X-6, MS D409
Todd Wareing, X-6, MS D409
Jim Warsa, X-6, MS D409
X-6 Files, MS D409
XDO Files, MS B218

TJU:tju