Title

Marching Diamonds for Unstructured Meshes

Permalink

Authors

Anderson, John C.
Bennett, Janine
Joy, Ken

Publication Date

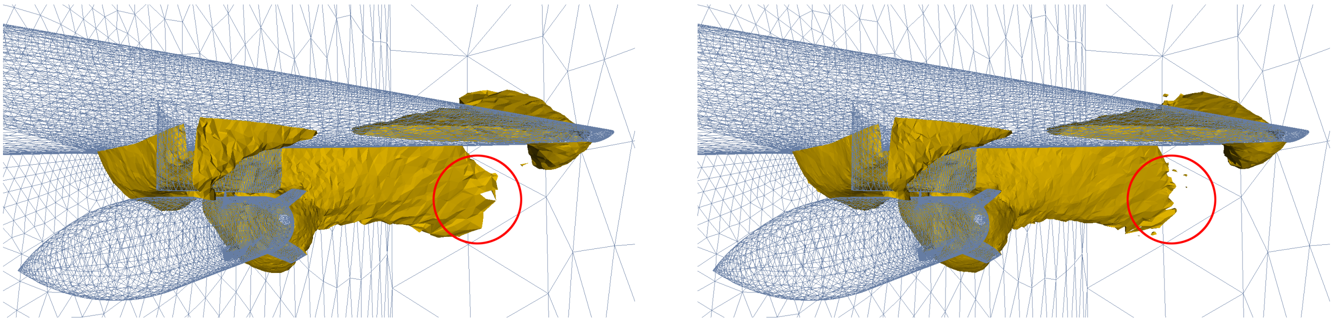
2005

Peer reviewed

Marching Diamonds for Unstructured Meshes

John C. Anderson[*] Janine C. Bennett[†] Kenneth I. Joy[‡]

Institute for Data Analysis and Visualization
Computer Science Department
University of California, Davis



(a) Marching Tetrahedra

(b) Marching Diamonds

Figure 1: Isosurface of a flow passing an aircraft wing with an attachment generated by (a) Marching Tetrahedra, and (b) Marching Diamonds. The surface extracted by Marching Diamonds is smoother, and in the circled area more accurately represents the dataset.

Abstract

We present a higher-order approach to the extraction of isosurfaces from unstructured meshes. Existing methods use linear interpolation along each mesh edge to find isosurface intersections. In contrast, our method determines intersections by performing barycentric interpolation over diamonds formed by the tetrahedra incident to each edge. Our method produces smoother, more accurate isosurfaces. Additionally, interpolating over diamonds, rather than linearly interpolating edge endpoints, enables us to identify up to two isosurface intersections per edge. This paper details how our new technique extracts isopoints, and presents a simple connection strategy for forming a triangle mesh isosurface.

CR Categories: I.3.6 [Computer Graphics]: Methodology and Techniques;

Keywords: isosurface extraction, interpolation, unstructured mesh

1 Introduction

The seminal work on isosurface extraction from scalar fields is Marching Cubes (MC) by Lorenson and Cline [9]. There are two major limitations of the MC algorithm: the triangulation of isopoints within a cube can be ambiguous, and a hexahedral input mesh is required. The asymptotic decider method of Nielson and Hamann [14] uses bilinear interpolation to resolve triangulation ambiguities across faces, however, it does not address ambiguities that exist when triangulating cell interiors. Internal ambiguities have been addressed [12, 3, 4, 8, 13], with the algorithm by Nielson [13] presenting the most thorough classification of isosurfaces generated via trilinear interpolation.

Dual contouring methods [7, 16] generate isosurfaces that are dual to surfaces created by the MC method. These methods have several advantages over the traditional MC method such as capturing sharp features and generating crack-free isosurfaces on adaptive input domains.

Marching Tetrahedra (MT) [15, 19] is an approach to isosurface extraction that addresses both flaws of the MC technique: there are no triangulation ambiguities, and it allows for isosurface extraction over a more general set of input domains (any mesh can be decomposed into tetrahedra). A drawback of the MT algorithm is that the resulting isosurfaces are dependent on the tetrahedral decomposition of the input domain since linear interpolation is used along each edge.

Zhou et al. [24] address the decomposition issue for tetrahedral meshes that derive from an initial hexahedral domain. Bilinear interpolation is used for edges inserted during the tetrahedrization process, and trilinear interpolation is applied within hexahedral cell interiors. While their approach removes the dependence of the resulting isosurface on the decomposition chosen, it does not work on arbitrary unstructured meshes.

Marching Diamonds (MD) is a new approach for extracting high-quality isosurfaces from unstructured mesh domains, while avoiding the limitations of existing methods. Rather than focusing on edges, our algorithm works on diamonds, which are formed by the set of tetrahedra surrounding an edge. Points within a diamond are represented using a barycentric coordinate system, and intersections of the isosurface with the diamond's defining edge are found by solving for the roots of a cubic polynomial. Because isopoints are associated with mesh edges, a simple connection strategy is sufficient to form a triangle mesh isosurface.

Section 2 describes our algorithm in two dimensions. Section 3 describes MD for three-dimensional unstructured meshes; it outlines the barycentric coordinate representation of points within a di-

[*] e-mail:anderson@cs.ucdavis.edu

[†] e-mail:bennettj@cs.ucdavis.edu

[‡] e-mail:kijoy@ucdavis.edu

amond (derived in the Appendix), and describes a connection strategy for forming a triangle mesh isosurface. We discuss implementation issues in Section 4, and Section 5 compares this algorithm to MT on example datasets.

2 Marching Diamonds In Two Dimensions

Consider a scalar or signed distance field over a triangulated set of scattered data points in the plane. For each edge e of the triangulation, we form a diamond $\mathscr{D}_e$ consisting of the two triangles sharing that edge, see Figures 2(a) and 2(b). The field over the diamond can be approximated as a bilinear function of the diamond's four vertices, d_i ($i = 0, \ldots, 3$). If $\mathscr{D}_e$ is convex, the isocontour representing a constant value over the field is a hyperbola. Rather than approximating the intersection of the isocontour with e by linear interpolation, we calculate the intersection of the hyperbola with e, which requires the solution of a quadratic polynomial [14]. Consequently, we may find up to two intersections along e.

To simplify the calculations for finding isopoints, the diamond $\mathscr{D}_e$ is mapped to a reference diamond $\mathscr{R}$, see Figure 2(c). The vertices of $\mathscr{R}$, r_i ($i = 0, \ldots, 3$), have coordinates $(1,1)$, $(-1,1)$, $(0,0)$, and $(0,2)$, respectively. r_2 and r_3 are endpoints of the diagonal in $\mathscr{R}$ corresponding to the edge e. Using barycentric interpolation in $\mathscr{R}$, which is equivalent to bilinear interpolation because $\mathscr{R}$ is a square, we calculate the coordinates of intersections of the hyperbolic isocontour with the reference diamond's diagonal. These coordinates are then used as weights on the vertices of $\mathscr{D}_e$ to calculate points on the isocontour. Having mapped back to $\mathscr{D}_e$, the isopoints may not lie on e, but will lie on the hyperbola that represents the isocontour through $\mathscr{D}_e$. If the endpoints of e are on opposite sides of the isocontour, one intersection will be found. Otherwise, we may find zero or two intersections. When two intersections are found, the region between the intersections will be on the opposite side of the isocontour from both endpoints of e.

This method also allows us to consider meshes with non-convex diamonds. Since we calculate intersections in the convex polytope $\mathscr{R}$, the calculated barycentric coordinates can be used to find isopoints of $\mathscr{D}_e$, even though $\mathscr{D}_e$ may be non-convex.

In order to connect isopoints into isocontour segments on a triangle-by-triangle basis, two cases must be considered. (1) If two edges in a triangle each have a single intersection, and the third edge has no intersections, the two isopoints are connected. (2) If an edge e within a triangle has two intersections, we split the diamond $\mathscr{D}_e$ before continuing with isocontour generation.

Splitting is a multi-step process. First, we insert a new vertex v generated midway between the two intersections. The original vertices of $\mathscr{D}_e$ and v are retriangulated to form four new triangles. Finally, before isocontour generation continues, the four diamonds defined by the newly inserted edges (v, d_i) $(i = 0, \ldots, 3)$ are checked for isocontour intersections. Note that the original two intersections found along e are no longer used because e is no longer in the mesh; instead, equivalent single intersections are found in the diamonds $\mathscr{D}_{(v,d_2)}$ and $\mathscr{D}_{(v,d_3)}$. The result of splitting the diamond $\mathscr{D}_e$ from Figure 2(a) is shown in 2(d).

After the connection process has completed, each triangle in the original unstructured mesh will either contain no isocontour segments, or will have been split as necessary to exhibit one of the 9 configurations shown in Figure 3.

We find that MD gives very good results in two dimensions. Figure 4 shows the difference in quality between MD and linear interpolation of isopoints identified for the function $f(x,y) = \operatorname{sinc}(x) * \operatorname{sinc}(y)$.

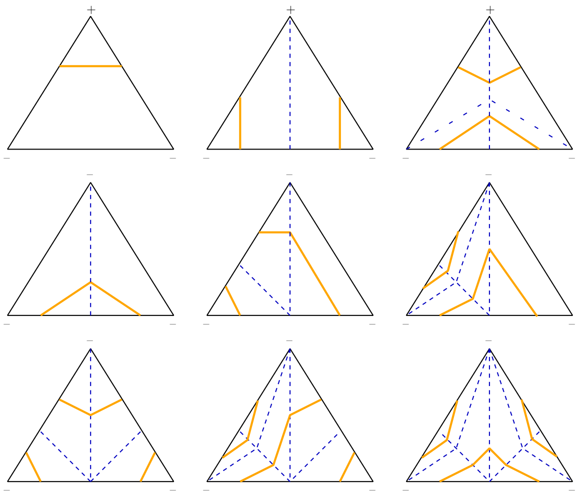
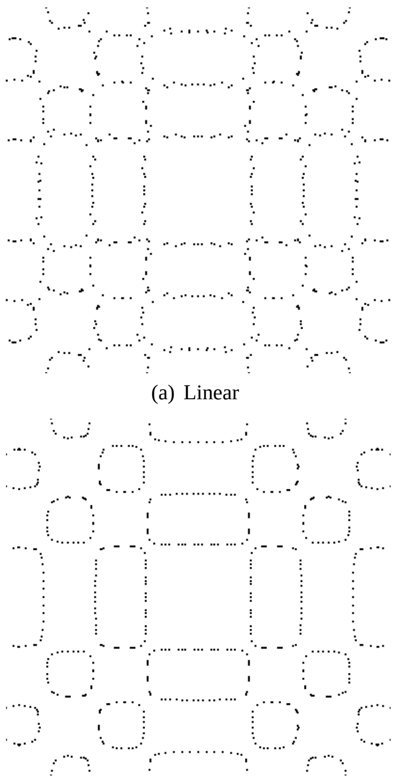


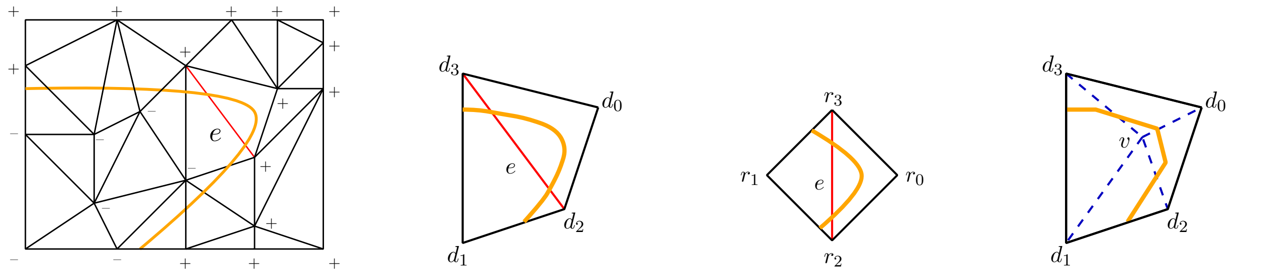
Figure 3: Original triangles in the mesh through which the isocontour passes will exhibit one of these 9 configurations. Isocontour segments (gold) are formed to connect isopoints. Diamonds with two intersections are split during isocontouring (inserted edges in dashed blue).



(a) Linear

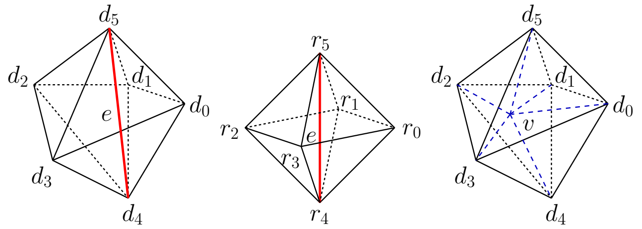
(b) Marching Diamonds

Figure 4: Isopoints of the function $f(x,y) = \operatorname{sinc}(x) * \operatorname{sinc}(y)$ identified using (a) linear interpolation, and (b) Marching Diamonds. Note the quality of the results achieved using MD.



(a) Isocontour (b) Diamond, $\mathscr{D}_e$ (c) Reference diamond, $\mathscr{R}$ (d) Approximate isocontour over split of diamond $\mathscr{D}_e$

Figure 2: We consider in (a) an isocontour of a signed distance field over a triangulated set of scattered data points in the plane. The isocontour twice intersects the diamond $\mathscr{D}_e$ defined by edge e (red) as shown in (b). In (c) $\mathscr{D}_e$ is mapped to a reference diamond $\mathscr{R}$ to solve for the barycentric coordinates of isocontour intersections. During the isocontour connection process shown in (d), $\mathscr{D}_e$ is split by inserting a new vertex v, and isocontour segments are formed over the region of the original diamond.



(a) Diamond, $\mathscr{D}_e$ (b) 4-Reference diamond, $\mathscr{R}_4$ (c) Split of diamond $\mathscr{D}_e$

Figure 5: In order to use Warren et al.'s barycentric coordinates, we map the diamond $\mathscr{D}_e$ in (a) to the 4-reference diamond $\mathscr{R}_4$ in (b). If two intersections are found along e, then $\mathscr{D}_e$ is split by inserting a new vertex v as shown in (c).

3 Marching Diamonds

Lifting this method to three dimensions requires additional effort, but our approach is similar. Here, a diamond $\mathscr{D}_e$ is formed by the set of tetrahedra incident along edge e. To approximate the scalar field over $\mathscr{D}_e$, we map it to a reference diamond. The barycentric coordinates of isosurface intersections with e in the reference diamond can be obtained by solving a cubic equation. These coordinates are then used as weights on vertices of $\mathscr{D}_e$ to obtain isopoints. Splitting is still used during the isosurface connection process.

Consider an edge e in a three-dimensional unstructured mesh. The *diamond* $\mathscr{D}_e$ is composed of the k tetrahedra incident along e, as shown in Figure 5(a). The vertices of $\mathscr{D}_e$ are denoted d_i ($i = 0,\ldots,k+1$), where d_i ($i = 0,\ldots,k-1$) are the *ring* vertices, and d_k and d_{k+1} are the endpoints of e. Let s_i ($i = 0,\ldots,k+1$) be the scalar values associated with the vertices of $\mathscr{D}_e$.

We define a convex *k-reference* diamond $\mathscr{R}_k$ with vertices given by:

$$r_i = \begin{cases} (\cos\mu_i, \sin\mu_i, 1) & \text{if } 0 \le i \le k-1 \\ (0,0,0) & \text{if } i = k \\ (0,0,2) & \text{if } i = k+1 \end{cases}$$

where $\mu_i = \frac{2\pi(i \bmod k)}{k}$, with associated scalar values s_i mapped from $\mathscr{D}_e$. Figure 5(b) illustrates a 4-reference diamond $\mathscr{R}_4$.

We use Warren et al.'s [23] method to define a barycentric coordinate system over the reference diamond. Using barycentric coordinates, a point p in $\mathscr{R}_k$ can be written as

$$p = \alpha_0 r_0 + \alpha_1 r_1 + \cdots + \alpha_{k+1} r_{k+1}, \tag{1}$$

and the scalar field at p can be approximated by

$$s = \alpha_0 s_0 + \alpha_1 s_1 + \cdots + \alpha_{k+1} s_{k+1}. \tag{2}$$

Our approach limits the use of barycentric coordinates to the representation of intersections of the isosurface with e. This simplifies our mathematics greatly. A point $(0,0,z)$ on e in $\mathscr{R}_k$, with $z \in [0,2]$, can be represented using the coordinates:

$$\alpha_i = \begin{cases} \frac{4|C|z(2-z)^2}{E} & \text{if } 0 \le i \le k-1 \\ \frac{D(2-z)^3}{E} & \text{if } i = k \\ \frac{Dz^3}{E} & \text{if } i = k+1 \end{cases} \tag{3}$$

where $E = 4k|C|z(2-z)^2 + D\left[(2-z)^3 + z^3\right]$, and C and D are constants dependent on k. The derivation of these coordinates is presented in the Appendix. Note that for a point on e, the barycentric coordinates for the ring vertices of $\mathscr{R}_k$ are equal.

Let s be the desired isovalue. Substituting the barycentric coordinates from Equation 3 into 2 we have:

$$s = \frac{4|C|z(2-z)^2}{E}\sum_{i=0}^{k-1} s_i + \frac{D(2-z)^3}{E} s_k + \frac{Dz^3}{E} s_{k+1} \tag{4}$$

By solving for the roots of this equation, we get values of z that, when used in Equation 3, provide barycentric coordinates in $\mathscr{R}_k$ of isosurface intersections. To obtain the isopoints of $\mathscr{D}_e$, we use the barycentric coordinates α_i as weights on the vertices of $\mathscr{D}_e$. The isopoints of $\mathscr{D}_e$ are thus $\sum_{i=0}^{i+1} \alpha_i d_i$ (note the similarity to Equation 1). Having mapped back to $\mathscr{D}_e$, the isopoints will not necessarily lie on e, but will lie on the isosurface.

As written, Equation 4 is cubic with up to three real roots. However, by rewriting the equation, it is clear that at most two of the roots correspond to actual points on e in $\mathscr{R}_k$. Consider the following formulation of Equation 4 as a Bézier curve defined by:

$$f(z) = L(2-z)^3 + \frac{M}{3} 3z(2-z)^2 + Nz^3$$

where, $L = D(s_k - s)$, $M = 4|C|\left(\sum_{i=0}^{k-1} s_i - ks\right)$, $N = D(s_{k+1} - s)$, and $z \in [0,2]$. This curve has control points $c_0 = 8L, c_1 = \frac{M}{3}, c_2 = 0$, and $c_3 = 8N$. If $f(z)$ has three roots in the interval $[0,2]$, then by the variation diminishing property the control polygon of $f(z)$ must cross the z axis three times (see Farin [5]). This is impossible since one of the interior control points is zero. Therefore we conclude that $f(z)$ has at most two roots in the interval $[0,2]$, and that the isosurface intersects e in $\mathscr{R}_k$ (and therefore e in $\mathscr{D}_e$) at most twice.

Our triangulation strategy in three dimensions is very similar to that explained in Section 2 for forming two-dimensional isocontours. In order to connect isopoints into isosurface triangles on a tetrahedron-by-tetrahedron basis, two cases must be considered. (1) If either of the cases handled by MT arise [15, 19], we directly triangulate the isopoints with one or two triangles. (2) If some edge e within a tetrahedron has two intersections, we split the diamond $\mathscr{D}_e$ before continuing with isosurface generation.

Splitting is a multi-step process. First, we insert a new vertex v generated midway between the two intersections. Next, we add the edges (v,d_i) $(i=0,\ldots,k+1)$, and the $2k$ tetrahedra: $(v,d_{k+1},d_i,d_{(i+1) \bmod k})$ and $(v,d_k,d_{(i+1) \bmod k},d_i)$ $(i=0,\ldots,k-1)$. Finally, before isosurface generation continues, the $k+2$ diamonds defined by the newly inserted edges (v,d_i) $(i=0,\ldots,k+1)$ are checked for isosurface intersections. Note that the original two intersections found along e are no longer used because e is no longer in the mesh; instead, equivalent single intersections are found in the diamonds $\mathscr{D}_{(v,d_k)}$ and $\mathscr{D}_{(v,d_{k+1})}$. Figure 5(c) illustrates the split of the diamond shown in 5(a).

4 Implementation Issues

The use of this technique is straightforward, with few caveats:

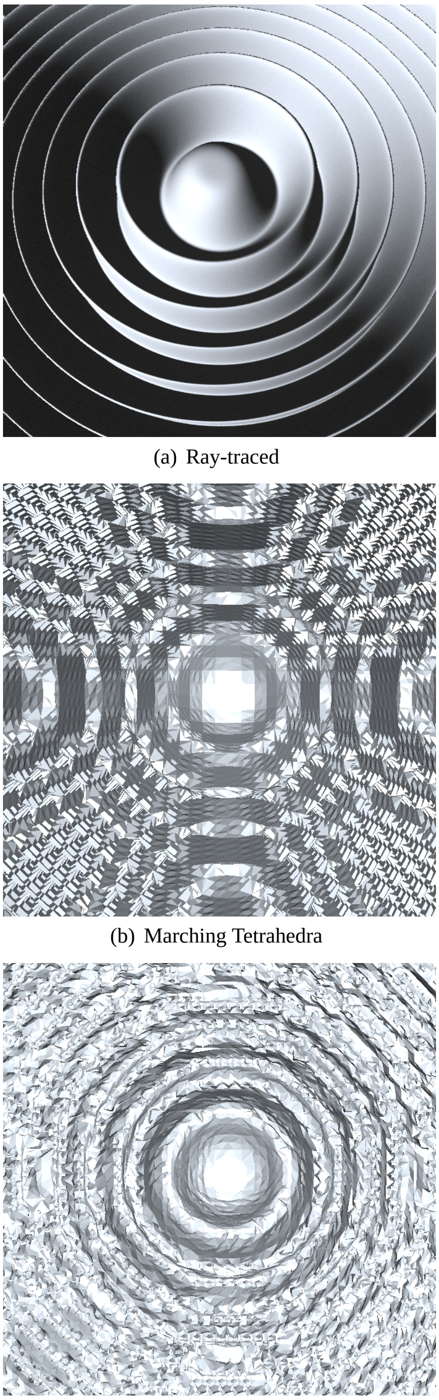
- We utilize the algorithm of Schwarze [17], which will locate up to three real roots of Equation 4. Only two, however, will lie in the range $z \in [0,2]$.
- We cannot define diamonds on boundary edges of the mesh. For these edges, we use linear interpolation.
- This algorithm was designed for meshes that have few non-convex diamonds. For non-convex diamonds, results may be less accurate because intersection calculations are performed within a convex reference diamond.
- Numerically it is possible (although rare) to encounter situations in which splitting proceeds indefinitely. Let l be the splitting level of a diamond $\mathscr{D}_e$, with $l=0$ for diamonds formed by edges of the original mesh. When a diamond is split, the newly introduced diamonds are assigned a splitting level of $l+1$. In our implementation, when two intersections are found in a diamond with $l \geq 2$, they are discarded and the diamond is not split. We have found that this does not negatively affect MD's visible output.

5 Results

Marching Diamonds in two dimensions, as discussed in Section 2, generates superior results compared to linear interpolation. In three dimensions, our algorithm continues to extracts high-quality isosurfaces. The reader is encouraged to refer to Table 1 for information about the isosurfaces discussed below.

Consider the analytic function defined by Marschner and Lobb [10]:

$$\rho(x,y,z) = \frac{\left(1-\sin(\pi z/2)+\alpha\left(1+\rho_r\left(\sqrt{x^2+y^2}\right)\right)\right)}{2(1+\alpha)}$$



(a) Ray-traced

(b) Marching Tetrahedra

(c) Marching Diamonds

Figure 6: Three isosurfaces of the Marschner and Lobb function with $s=0.5$ generated by (a) ray-tracing, (b) MT, and (c) MD. Note the increased accuracy of the MD reconstruction of this function compared to MT.

Table 1: Isosurfacing with Marching Tetrahedra and Marching Diamonds

	Marschner and Lobb ($s = 0.5$)		Aerodynamics ($s = 0.7$)		Aerodynamics ($s = 0.8$)	
	MT	MD	MT	MD	MT	MD
Isosurface Triangles	57,986	117,406	25,983	32,309	39,570	46,592
Split Diamonds	—	10,255	—	893	—	990
Tetrahedra in Mesh	355,914	457,328	567,862	576,714	567,862	577,652
Non-convex Diamonds	—	0	—	23,345	—	23,347

where,

$$\rho_r(r) = \cos(2\pi f_M \cos(\frac{\pi r}{2}))$$
$$f_M = 6$$
$$\alpha = 0.25.$$

We evaluate this function over a 40x40x40 rectilinear grid with domain $-1 \leq x, y, z \leq 1$. This grid is then tetrahedrized using a six tetrahedra per cell decomposition. We look at isosurfaces with $s = 0.5$. Figure 6(a) shows a ray-traced image of the Marschner and Lobb function. Figures 6(b) and 6(c) show the isosurfaces extracted by MT and MD, respectively. The surface reconstructed by MD is considerably more accurate than that produced by MT. Table 1 provides triangle counts associated with both surfaces. The higher triangle count for the surface extracted by MD is due the large number of diamonds for which two intersections were found.

To further demonstrate MD, we consider a dataset of an aerodynamics flow simulation run upon an unstructured mesh of an aircraft wing. This mesh contains over half a million tetrahedra. Figures 1 and 7 illustrate isosurfaces generated from this dataset with $s = 0.7$ and $s = 0.8$, respectively.

Since MD can identify up to two intersections per edge, it is often able to extract complex isosurfaces that are mis-handled by MT. An example of this can be seen in the surfaces of Figure 1. Here, the MD-produced surface in 1(b) correctly separates isosurface components in the circled area from the nearby, larger surface. The components are incorrectly connected by MT in 1(a).

Another benefit of the MD algorithm is that it extracts smooth isosurfaces. Figure 7 demonstrates this aspect of our higher-order approximation; we see that the surface extracted by MD in 7(b) is considerably smoother than that extracted by MT in 7(a). For $s = 0.8$, we derive from Table 1 that the MD surface contains only 17.7% more triangles than the MT surface.

6 Conclusions

Marching Diamonds is a new approach for extracting high-quality isosurfaces from unstructured mesh domains, while avoiding the limitations of existing methods. Rather than focusing on edges, our algorithm works on diamonds. Using a novel barycentric coordinate representation we are able to identify up to two isosurface intersections along each diamond's defining edge. We have found that MD produces smoother, more accurate isosurfaces on analytic and real-world datasets. The primary computational burden introduced by this algorithm is the calculation of the roots of a cubic polynomial. In the future, we intend to quantify the effects of non-convex diamonds on our algorithm, and to work on improving the speed of MD.

Acknowledgments

This work was supported by the National Science Foundation under contracts ACR 9982251 and ACR 0222909, and by Lawrence Livermore National Laboratory under contract B523818. We thank Hiroshi Akiba for producing the two-dimensional results in Figure 4, Aaron Lefohn for providing the ray-traced image in Figure 6, and Greg Nielson for providing input on an initial version of this work. The members of the Visualization and Graphics Group of the Institute for Data Analysis and Visualization (IDAV) at UC Davis were extremely helpful.

References

[1] Ken Brodlie and Jason Wood. Recent advances in volume visualization. *Computer Graphics Forum*, 20(2):125–148, 2001.

[2] Shek Ling Chan and Enrico O. Purisima. A new tetrahedral tesselation scheme for isosurface generation. *Computers & Graphics*, 22(1):83–90, 1998.

[3] Evgeni V. Chernyaev. Marching cubes 33: construction of topologically correct isosurfaces. Technical Report CN/95-17, CERN, 1995. http://wwwinfo.cern.ch/asdoc/psdir/mc.ps.gz.

[4] P. Cignoni, F. Ganovelli, C. Montani, and R. Scopigno. Reconstruction of topologically correct and adaptive trilinear isosurfaces. *Computers & Graphics*, 24(3):399–418, 2000.

[5] Gerald Farin. *Curves and Surfaces for Computer-Aided Geometric Design: A Practical Guide*. Academic Press, New York, NY, USA, third edition, 1993.

[6] M. S. Floater, K. Hormann, and G. Kós. A general construction of barycentric coordinates over convex polygons. *Advances in Computational Mathematics*, 2004. To appear.

[7] Tao Ju, Frank Losasso, Scott Schaefer, and Joe Warren. Dual contouring of hermite data. *ACM Transactions on Graphics*, 21(3):339–346, 2002.

[8] Adriano Lopes and Ken Brodlie. Improving the robustness and accuracy of the marching cubes algorithm for isosurfacing. *IEEE Transactions on Visualization and Computer Graphics*, 9(1):16–29, 2003.

[9] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *ACM SIGGRAPH 1987*, pages 163–169, 1987.

[10] Stephen R. Marschner and Richard J. Lobb. An evaluation of reconstruction filters for volume rendering. In *IEEE Visualization 1994*, pages 100–107, 1994.

[11] Mark Meyer, Haeyoung Lee, Alan Barr, and Mathieu Desbrun. Generalized barycentric coordinates on irregular polygons. *Journal of Graphics Tools*, 7(1):13–22, 2002.

[12] B. K. Natarajan. On generating topologically consistent isosurfaces from uniform samples. *The Visual Computer*, 11(1):52–62, 1994.

[13] Gregory Nielson. On marching cubes. *IEEE Transactions on Visualization and Computer Graphics*, 9:283–297, 2003.

[14] Gregory M. Nielson and Bernd Hamann. The asymptotic decider: resolving the ambiguity in marching cubes. In *IEEE Visualization 1991*, pages 83–91, 1991.

[15] Paul Ning and Jules Bloomenthal. An evaluation of implicit surface tilers. *IEEE Computer Graphics and Applications*, 13(6):33–41, 1993.

[16] Scott Schaefer and Joe Warren. Dual marching cubes: Primal contouring of dual grids. In *Pacific Graphics 2004*, pages 70–76, 2004.

[17] Jochen Schwarze. Cubic and quartic roots. In *Graphics Gems*, pages 404–407. Academic Press Professional, Inc., San Diego, CA, USA, 1990.

[18] Holger Theisel. Exact isosurfaces for marching cubes. *Computer Graphics Forum*, 21(1):19–31, 2002.

[19] G.M. Treece, R.W. Prager, and A.H. Gee. Regularised marching tetrahedra: improved iso-surface extraction. *Computers & Graphics*, 23(4):583–598, 1999.

[20] Allen van Gelder and Jane Wilhelms. Topological considerations in isosurface generation. *ACM Transactions on Graphics*, 13(4):337–375, 1994.

[21] Joe Warren. Barycentric coordinates for convex polytopes. *Advances in Computational Mathematics*, 6:97–108, 1996.

[22] Joe Warren. On the uniqueness of barycentric coordinates. In *Contemporary Mathematics*, pages 93–99. AGGM02, 2003.

[23] Joe Warren, Scott Schaefer, Anil Hirani, and Mathieu Desbrun. Barycentric coordinates for convex sets. Technical report, Rice University, 2003. Submitted for publication.

[24] Yong Zhou, Weihai Chen, and Zesheng Tang. An elaborate ambiguity detection method for constructing isosurfaces within tetrahedral meshes. *Computers & Graphics*, 19(3):355–364, 1995.

APPENDIX

The vertices of the reference diamond $\mathscr{R}_k$ are given by:

$$r_i = \begin{cases} (\cos\mu_i, \sin\mu_i, 1) & \text{if } 0 \le i \le k-1 \\ (0,0,0) & \text{if } i = k \\ (0,0,2) & \text{if } i = k+1 \end{cases}$$

where $\mu_i = \frac{2\pi(i \bmod k)}{k}$, with associated scalar values s_i mapped from $\mathscr{D}_e$.

Warren et al.'s [23] formulation of barycentric coordinates requires identifying the planes that define the convex polytope. For $\mathscr{R}_k$ there are $2k$ defining planes. The normal vectors associated with these planes can be divided into two groups: the k lower normals $\overrightarrow{L_i}$ $(i = 0,\dots,k-1)$ of the planes intersecting r_k, and the k upper normals $\overrightarrow{U_i}$ $(i = 0,\dots,k-1)$ of the planes intersecting r_{k+1}.

$\overrightarrow{L_i}$ is the normal of the plane defined by the vertices r_i, $r_{(i+1) \bmod k}$, and r_k:

$$\overrightarrow{L_i} = \left(\sin\mu_{i+1} - \sin\mu_i, \cos\mu_i - \cos\mu_{i+1}, -\sin\frac{2\pi}{k}\right)$$

$\overrightarrow{U_i}$ is the normal of the plane defined by the vertices r_i, $r_{(i+1)modk}$, and r_{k+1}:

$$\overrightarrow{U_i} = \left(\sin\mu_{i+1} - \sin\mu_i, \cos\mu_i - \cos\mu_{i+1}, \sin\frac{2\pi}{k}\right)$$

The normals $\overrightarrow{L_i}$ and $\overrightarrow{U_i}$ are similar because they are defined by two common vertices, so:

$$\begin{array}{rcl} \overrightarrow{L_i} & = & (\sigma_i, \tau_i, -\upsilon) \\ \overrightarrow{U_i} & = & (\sigma_i, \tau_i, \upsilon) \end{array} \qquad \begin{array}{rcl} \sigma_i & = & \sin\mu_{i+1} - \sin\mu_i \\ \tau_i & = & \cos\mu_i - \cos\mu_{i+1} \\ \upsilon & = & \sin\frac{2\pi}{k} \end{array}$$

We define the reference diamond $\mathscr{R}_k$ in terms the matrix inequality used to derive its barycentric coordinates:

$$\begin{bmatrix} \overrightarrow{L_0} \\ \vdots \\ \overrightarrow{L_{k-1}} \\ \overrightarrow{U_0} \\ \vdots \\ \overrightarrow{U_{k-1}} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \le \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 2\upsilon \\ \vdots \\ 2\upsilon \end{bmatrix}$$

To determine the barycentric coordinate $(\alpha_0, \alpha_1, \dots, \alpha_{k+1})$, corresponding to a point on e in $\mathscr{R}_k$, we begin by determining the weight functions ω_i $(i = 0,\dots,k+1)$ associated with the vertices r_i. In the following calculations, we assume that $x = y = 0$, as we are only interested in intersections of the isosurface along the edge e in $\mathscr{R}_k$. This simplifies calculations significantly. We also assume that plane indices are modulo k.

Weight Function for Ring Vertices

Warren et al.'s method derives barycentric weight functions for each vertex using the planes incident to that vertex. The vertex r_i $(i = 0,\dots,k-1)$ lies at the intersection of four planes with normals $\overrightarrow{L_{i-1}}$, $\overrightarrow{L_i}$, $\overrightarrow{U_{i-1}}$, and $\overrightarrow{U_i}$. Thus, the weight functions associated with the vertices r_i are given by:

$$\omega_i = \frac{\left| Det \begin{bmatrix} \overrightarrow{L_i} \\ \overrightarrow{L_{i-1}} \\ \overrightarrow{U_{i-1}} \end{bmatrix} \right| + \left| Det \begin{bmatrix} \overrightarrow{L_i} \\ \overrightarrow{L_{i-1}} \\ \overrightarrow{U_i} \end{bmatrix} \right|}{\upsilon^3 z^2 (2-z)}$$

We note that the two terms in the numerator simply calculate the volume of a parallelepiped spanned by three plane normals, and that the absolute value of each term is identical. Thus:

$$\begin{aligned} \omega_i &= \frac{2 \left| Det \begin{bmatrix} \overrightarrow{L_i} \\ \overrightarrow{L_{i-1}} \\ \overrightarrow{U_{i-1}} \end{bmatrix} \right|}{\upsilon^3 z^2 (2-z)} \\ &= \frac{4\,|\sigma_i \tau_{i-1} - \sigma_{i-1}\tau_i|}{\upsilon^2 z^2 (2-z)} \end{aligned}$$

Finally, by letting $C = \sigma_i \tau_{i-1} - \sigma_{i-1}\tau_i = -2\sin\frac{2\pi}{k} + \sin\frac{4\pi}{k}$ we are able to express ω_i as:

$$\omega_i = \frac{4\,|C|}{\upsilon^2 z^2 (2-z)}$$

Weight Functions for Vertices r_k and r_{k+1}

r_k is incident to k planes, with normals L_i $(i = 0,\dots,k-1)$. Given these normals, the weight function ω_k can be formulated as:

$$\omega_k = \frac{\left| Det \begin{bmatrix} \overrightarrow{L_0} \\ \overrightarrow{L_2} \\ \overrightarrow{L_1} \end{bmatrix} \right| + \left| Det \begin{bmatrix} \overrightarrow{L_1} \\ \overrightarrow{L_3} \\ \overrightarrow{L_2} \end{bmatrix} \right| + \cdots + \left| Det \begin{bmatrix} \overrightarrow{L_{k-3}} \\ \overrightarrow{L_{k-1}} \\ \overrightarrow{L_{k-2}} \end{bmatrix} \right|}{\upsilon^3 z^3}$$

Again we note that the $k-2$ terms in the numerator simply calculate the volume of a parallelepiped, and that the absolute value of each term is identical. Therefore:

$$\begin{aligned} \omega_k &= \frac{(k-2) \left| Det \begin{bmatrix} \overrightarrow{L_i} \\ \overrightarrow{L_{i+2}} \\ \overrightarrow{L_{i+1}} \end{bmatrix} \right|}{\upsilon^3 z^3} \\ &= \frac{(k-2)\,|-2C - (\sigma_i \tau_{i+2} - \sigma_{i+2}\tau_i)|}{\upsilon^2 z^3} \end{aligned}$$

Finally, by letting $D = (k-2)\,|-2C - (\sigma_i \tau_{i+2} - \sigma_{i+2}\tau_i)| = (k-2)\left|3\sin\frac{2\pi}{k} - 4\sin\frac{4\pi}{k} - \sin\frac{6\pi}{k}\right|$ we have:

$$\omega_k = \frac{D}{\upsilon^2 z^3}$$

Similarly, r_{k+1} is incident to k planes, with normals U_i ($i = 0,\ldots,k-1$). Given these normals:

$$\omega_{k+1} = \frac{\left|Det\begin{bmatrix}\overrightarrow{U_0}\\ \overrightarrow{U_1}\\ \overrightarrow{U_2}\end{bmatrix}\right| + \left|Det\begin{bmatrix}\overrightarrow{U_1}\\ \overrightarrow{U_2}\\ \overrightarrow{U_3}\end{bmatrix}\right| + \cdots + \left|Det\begin{bmatrix}\overrightarrow{U_{k-3}}\\ \overrightarrow{U_{k-2}}\\ \overrightarrow{U_{k-1}}\end{bmatrix}\right|}{\upsilon^3(2-z)^3}$$

$$= \frac{D}{\upsilon^2(2-z)^3}$$

Barycentric Coordinates

The final step in solving for the barycentric coordinates of points on e in $\mathscr{R}_k$ is to determine the sum of the weight functions $\sum_{i=0}^{k+1}\omega_i$ and to divide each of the weight functions ω_i by this sum.

$$\sum_{i=0}^{k+1}\omega_i = k*\left(\frac{4\,|C|}{\upsilon^2 z^2(2-z)}\right) + \frac{D}{\upsilon^2 z^3} + \frac{D}{\upsilon^2(2-z)^3}$$

$$= \frac{4k\,|C|\,z(2-z)^2 + D\left[(2-z)^3 + z^3\right]}{\upsilon^2 z^3(2-z)^3}$$

Thus, the barycentric coordinates for a point $(0,0,z)$ on e in $\mathscr{R}_k$, with $z \in [0,2]$, are:

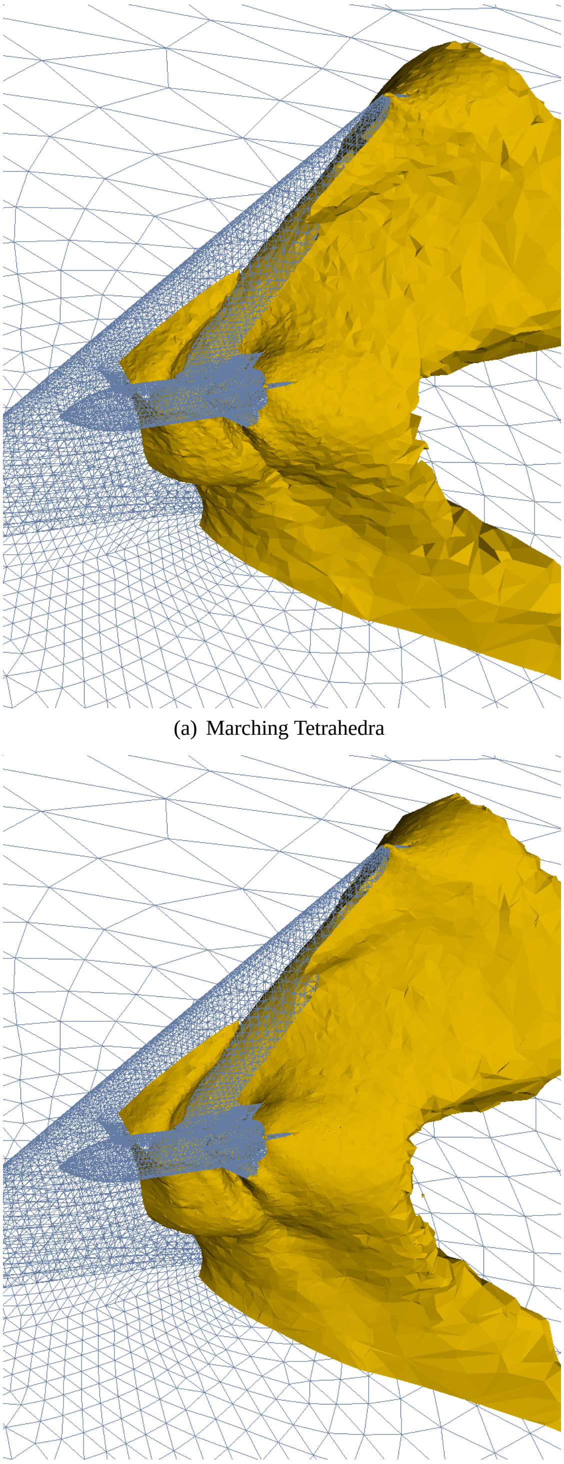
$$\alpha_i = \begin{cases} \frac{4|C|z(2-z)^2}{E} & \text{if } 0 \le i \le k-1 \\ \frac{D(2-z)^3}{E} & \text{if } i = k \\ \frac{Dz^3}{E} & \text{if } i = k+1 \end{cases} \tag{5}$$

where,

$$C = -2\sin\frac{2\pi}{k} + \sin\frac{4\pi}{k} \tag{6}$$

$$D = (k-2)\left|3\sin\frac{2\pi}{k} - 4\sin\frac{4\pi}{k} - \sin\frac{6\pi}{k}\right| \tag{7}$$

$$E = 4k\,|C|\,z(2-z)^2 + D\left[(2-z)^3 + z^3\right]. \tag{8}$$



(a) Marching Tetrahedra

(b) Marching Diamonds

Figure 7: Isosurface of a flow passing an aircraft wing with an attachment generated by (a) Marching Tetrahedra, and (b) Marching Diamonds. The MD surface is visually smoother than the MT surface, and contains only 17.7% more triangles.