Feature Sensitive Isosurface Extraction from Gradient Data

Arindam Bhattacharya *
The Ohio State University

Rephael Wenger†
The Ohio State University

Abstract

Previously published algorithms to construct isosurfaces with sharp edges and corners require "Hermite" data, the exact intersection points of grid edges and the isosurface and the exact gradients at those intersection points. We are interested in constructing isosurfaces with sharp edge and corners from regular grid of scalar data without any information about intersection points or gradients at those intersection points. We decompose the problem into two parts: 1) Compute gradients at grid vertices from scalar data; 2) Compute an isosurface with sharp edges and vertices from a regular grid of scalar and gradient values. We focus on the second problem, computing an isosurface from scalar and gradient data. We also describe a method for visualizing and evaluating the accuracy of a reconstructiong of sharp features.

Index Terms: I.3.5 [Computational Methodologies]: Computer Graphics—Computational Geometry and Object Modeling; G.3 [Mathematics of Computing]: Probability and Statistics

1 Introduction

Algorithms to construct isosurfaces with sharp edges and corners are described in [1, 2, 3, 4, 5] and various other papers. These algorithms all rely upon "Hermite" data, the exact intersection points of grid edges and the isosurface and the exact gradients at those intersection points. While some of the papers suggest constructing Hermite data from a regular grid of scalar values by computing gradients at grid vertices and applying linear interpolation, none of the papers present experimental results of such a procedure. As we show, a simple approach based on linear interpolation will not work.

We separate the problem of computing isosurfaces with sharp features from scalar data into two parts: 1) Compute gradients at grid vertices from scalar data; 2) Compute an isosurface with sharp edges and vertices from a regular grid of scalar and gradient values. Both parts are challenging.

In this extended abstract, we address the second problem of computing an isosurface with sharp features from scalar and gradient data. We show that computing Hermite data from gradient data requires a gradient based calculation, not interpolation, and that Hermite data can be bypassed completely by directly computing isosurface vertices from selected gradients. We also show that modeling sharp isosurface features sometimes requires placing isosurface vertices outside their related cube, using the Linf metric to place vertices on sharp edges, and placing multiple isosurface vertices in some cubes.

2 Isosurface construction

Our basic approach is similar to the dual contouring algorithm as outlined in [2]. The isosurface is composed of a set of quadrilaterals corresponding to edges intersected by the isosurface. Each isosurface vertex corresponds to a grid cube. The intersection points of the cube edges and the isosurface determine a set of tangent planes. Using quadric error measures, the isosurface vertex is located as the point which is "close" to each of these tangent planes.

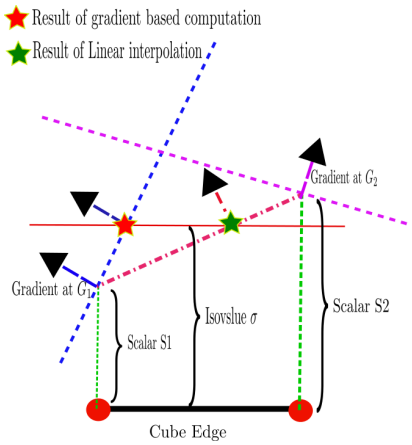


Figure 1: Linear interpolation vs Gradient based calculation

2.1 Gradient based Computation

Kobbelt et. al. [3] propose using interpolation for computing gradients on grid edges. However, as shown in Figure 1, linear interpolation fails miserably when the gradient field is not smooth. The correct intersection point is totally different from the one predicted by linear interpolation. The gradient at the intersection point is a copy of the gradient at one endpoint, not a combination of the two gradients.

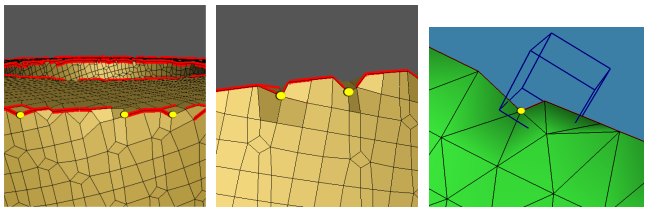
Using the gradients, we can compute the correct intersection point of the isosurface and the grid edge. We applied the dual contouring algorithm of [2] using intersection points computed using linear interpolation and using intersection points based on gradient calculations. Figure 2a compares isosurfaces computed from the two intersection calculations and shows the problems with linear interpolation.

Instead of computing the location of the isosurface vertex from edge intersection points, we can compute the location of the isosurface vertex directly from gradients at the cube vertices. A grid vertex v_i at point p_i with scalar value s_i and gradient g_i determines a scalar field $f_i(x) = (x - p_i) \cdot g_i + s_i$. The level set $f_i^{-1}(\sigma)$ is a plane in $\mathbb{R}^3$ which approximates $f^{-1}(\sigma)$ near p_i.Using quadric error measures, construct the planes $f_i^{-1}(\sigma)$ for the cube vertices (or some subset of the cube vertices) and locate the isosurface vertex at a point which is close to each of these planes $f_i^{-1}(\sigma)$.

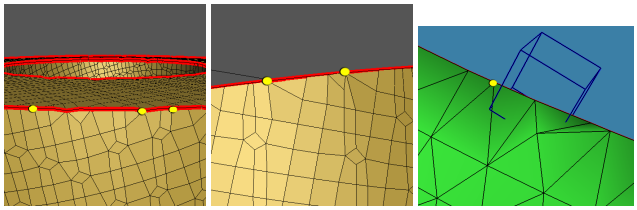
2.2 Clamping to the cube

Algorithms in [1, 2, 3] restrict an isosurface vertex to its corresponding cube **c**. When the point which minimizes the distance to the tangent planes lies outside cube **c**, the point coordinates are clamped to lie on the boundary of **c**. We found that this restriction leads to 'ridges' along edges (Figure 2b). In these cases it is necessary to place the isosurface vertex outside **c** (Figure 2e) to model sharp edges. The isosurface vertex is generated by **c** but lies in some

*e-mail: bhattaca@cse.ohio-state.edu
†e-mail:wenger@cse.ohio-state.edu



(a) Linear interpolation (b) Clamped to cube (c) No L_∞



(d) Grad based estimation (e) Vert allowed outside (f) Using L_∞

Figure 2: Comparisons.

other grid cube $\mathbf{c}'$. We allow the point to lie in $\mathbf{c}'$ only if $\mathbf{c}'$ does not itself generate any isosurface vertices, i.e., the span of $\mathbf{c}'$ does not contain the isovalue.

2.3 L_2 vs L_∞ norm

Let $\mathbf{c}$ be a grid cube with a corresponding isosurface vertex $v_\mathbf{c}$. When the selected gradients of $\mathbf{c}$ point in exactly two distinct directions (up to small perturbations), the planes $f_i^{-1}(\sigma)$ determine a line $\mathbf{l}$. This line $\mathbf{l}$ locally approximates a sharp edge of the level set $f^{-1}(\sigma)$, Let p be the point on $\mathbf{l}$ which is closest to the center of $\mathbf{c}$ under the L_2 norm. We usually locate the isosurface vertex $v_\mathbf{c}$ for $\mathbf{c}$ at point p. However, p may lie in some cube $\mathbf{c}' \neq \mathbf{c}$. If cube $\mathbf{c}'$ generates its own isosurface vertex, then p is not an appropriate location for $v_\mathbf{c}$. In this case, we compute the closest point $\tilde{p}$ on $\mathbf{l}$ to the center of $\mathbf{c}$ under the L_∞ metric. While p is usually in a cube $\mathbf{c}'$ sharing a face with cube $\mathbf{c}$, point $\tilde{p}$ is in a cube $\tilde{\mathbf{c}}$ which shares only an edge with $\mathbf{c}$. We attempt to locate $v_\mathbf{c}$ at p'. As shown in Figure(2c, 2f) using $\tilde{p}$ as an alternative location avoids ridges in the sharp edge.

2.4 Multiple isosurface vertices

The simple dual contouring algorithm creates a single isosurface vertex per intersected cube. Restricting each cube to a single isosurface vertex, creates non-manifold regions in the isosurface near sharp features. As in [4], we use G. Nielson's dual contouring algorithm to allow multiple isosurface vertices for certain cube-isosurface intersection patterns. For certain "ambiguous" intersection patterns, there is a choice of where or whether to place multiple vertices. We resolve these ambiguities so that cubes containing sharp features have only one isosurface vertex while adjacent cubes may have more than one.

3 Algorithm

A (very) rough outline of our algorithm is as follows:

1. Resolve ambiguous cube-isosurface intersection patterns;
2. For each isosurface vertex $v_\mathbf{c}$ in cube $\mathbf{c}$:
   (a) Select a subset of $\mathbf{c}$'s vertices and gradients;
   (b) Position $v_\mathbf{c}$ based on the planes determined by the selected gradients;
   (c) Allow $v_\mathbf{c}$ to lie in some cube $\mathbf{c}' \neq \mathbf{c}$ as long as $\mathbf{c}'$ is empty;
   (d) Use both the L_2 and the L_∞ metric to position $v_\mathbf{c}$ along a sharp line;
3. Construct an isosurface quadrilateral for each grid edge intersected by the isosurface.

Vertex degree	a) Algorithm	b) Restricted	c) No L_∞	d) Single isov
1	2	73	7	20
2	784	542	775	782
3	2	302	15	24
>3	0	63	0	60
1, 3 or >3	4	438	797	104

Table 1: Vertex degrees of graph of sharp edges of an annulus isosurface with central axis in direction (1,1,1). a) Results from our algorithm. b) Isosurface vertices are restricted to their corresponding grid cubes. c) L_∞ norm is never used for positioning an isosurface vertex on a grid line. d) Each grid cube generates at most one isosurface vertex.

4 Measuring Sharp Features

To visualize sharp features in an isosurface, we compute the dihedral angle between adjacent isosurface polygons and report the common edge whenever the dihedral angle is less than a specified threshold (140°). We also report any edge incident on three or more isosurface polygons. Drawing the reported edges either alone or on the isosurface (Figure 2) allows us to quickly visualize and identify missing or incorrect sharp edges and vertices.

The set of isosurface edges with small dihedral angle can be viewed as a graph embedded in $\mathbb{R}^3$. To measure the representation of sharp features in an isosurface, we compute and output the number of vertices of degrees one, two, three or four and higher in this graph. (See Table 1 for an example.) Isosurfaces which have poor representations of sharp features tend to produce numerous graph vertices of degrees one, three or four and higher.

Table 1 shows the number of vertices with various degrees in the graph of sharp isosurface edges. For the given isosurface all vertices of this graph should have degree two. Comparison of the number of vertices with degrees other than two (4 vs. 438) shows that allowing vertices outside cube $\mathbf{c}$(Sect 2.2), using L_∞ (Sec 2.3)and multiple iso-vertices(Sec 2.4) provides far better results.

References

[1] C. Ho, F. Wu, B. Chen, and M. Ouhyoung. Cubical marching squares: Adaptive feature preserving surface extraction from volume data. *Computer Graphics Forum*, 24:2005, 2005.

[2] T. Ju, F. Losasso, S. Schaefer, and J. Warren. Dual contouring of hermite data. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '02, pages 339–346, New York, NY, USA, 2002. ACM.

[3] L. P. Kobbelt, M. Botsch, U. Schwanecke, and H.-P. Seidel. Feature sensitive surface extraction from volume data. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, pages 57–66, New York, NY, USA, 2001. ACM.

[4] S. Schaefer, T. Ju, and J. Warren. Manifold dual contouring. *IEEE Transactions on Visualization and Computer Graphics*, 13(3):610–619, May 2007.

[5] S. Schaefer and J. Warren. Dual marching cubes: Primal contouring of dual grids. In *Proceedings of the Computer Graphics and Applications, 12th Pacific Conference*, PG '04, pages 70–76, Washington, DC, USA, 2004. IEEE Computer Society.