Adaptively Adjusting Marching Cubes Output to Fit A Trilinear Reconstruction Filter

Fabio Allamandri, Paolo Cignoni, Claudio Montani, and Roberto Scopigno

Istituto di Elaborazione dell'Informazione Consiglio Nazionale delle Ricerche,
Via S. Maria, 46 - 56126 Pisa ITALY
Email: {cignoni,montani}@iei.pi.cnr.it , r.scopigno@cnuce.cnr.it

Abstract. The paper focuses on the improvement of the quality of iso-surfaces fitted on volume datasets with respect to standard MC solutions. The new solution presented improves the precision in the reconstruction process using an approach based on mesh refinement and driven by the evaluation of the trilinear reconstruction filter. The iso-surface reconstruction process is adaptive, to ensure that the complexity of the fitted mesh will not become excessive. The proposed approach has been tested on many datasets; we discuss the precision of the obtained meshs and report data on fitted meshes complexity and processing times.

1 Introduction

The Marching Cubes (MC) algorithm [11] is nowadays the most diffuse technique for the extraction of iso-surfaces out of volume datasets. The reasons for the MC success include its simple logical structure, implying a nearly straightforward implementation, and its computational efficiency. MC has been incorporated in many commercial and public domain visualization systems. Many papers appeared on enhancements, optimization, extensions and applications of this technique [23, 22, 16, 15, 2, 1, 10]. One of the few limitations of MC is the linearity of the reconstruction kernel used. MC adopts a local approach, i.e. each cell is tested for a possible iso-surface patch independently from the others. Each patch is computed by adopting a table-driven approach, and is defined by the position of vertices located on cell edges. The iso-surface patch returned is therefore a linear approximation (planar faces), whose vertices are located on cell edges (this ensures iso-surface C^0 continuity between cells) and are computed using linear interpolation. When a very high resolution dataset is used, the simplicity of the reconstruction filter is not easily perceptible, unless we perform substantial zooming into the mesh. But if the latter case holds, or if dataset resolution is low, the adoption of a more sophisticated interpolation filter might be required to improve smoothness of the fitted surface.

In this paper we focus on volume data applications based on the visualization of iso-surfaces. We look for methods which produce a "surface-based" output (i.e. ray casting solutions are considered not appropriate), to allow hardware-assisted interactive visualization and data distribution/rendering in web environments.

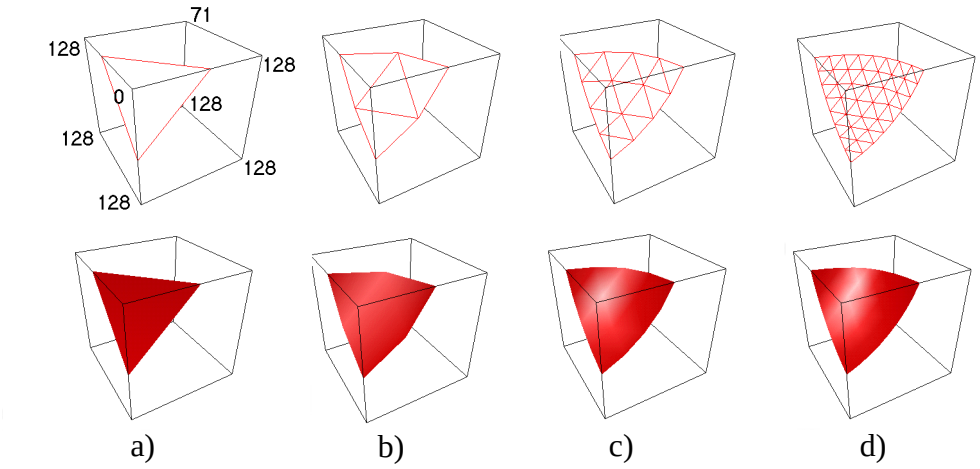

Fig. 1. Enhancing reconstruction precision: different patches are extracted from the same cell, using increased precision and deeper refinement.

Therefore, we present a solution which improves the precision in the reconstruction process, with respect to standard MC solution, using mesh refinement and the trilinear reconstruction filter (see Figure 1). The iso-surface reconstruction process is adaptive, to ensure that the complexity of the fitted mesh will not become excessive (thus reducing or preventing interactive visualization).

2 Previous work

The excessive simplicity of the reconstruction filter used by MC has been pointed out firstly by Fruehauf [5]. He compared images rendered using the MC output meshes to adopting a ray casting approach (which generally uses a tri-linear reconstruction filter) and showed how much they differ. An advantage of ray casting is to allow the adoption of whichever reconstruction filter; many different interpolation filters have been proposed [12–14] to evaluate/interpolate more precisely both field values and gradients. Unfortunately, ray casting produces images of the isosurface we are interested in (a view-dependent process), rather than extracting explicitly the iso-surface. For many applications, producing an image is not enough. The explicit reconstruction of surface geometries may be needed, for example, in virtual simulation environments. Moreover, a shortcoming of the ray casting approach is the non-interactive rendering time. For these reasons, the precision of the fitted iso-surfaces cannot be improved in many applications by simply adopting a ray casting solution together with a more sophisticated reconstruction filter.

The technique proposed in this paper adopts a regular mesh refinement approach. The idea of improving the quality of a mesh by applying [recursively] a

sequence of local refinements is not new, and it has been proposed: to construct adaptive piecewise linear representations of implicit surfaces [6, 21]; to reconstruct adaptively the surface of three-dimensional objects from multiple range images [17]; to extract a surface out of sampled scalar/vectorial 3D datasets starting from an initial surface seed and then applying an iterative surface inflation process [20]; and to refine a surface under a strict surface curvature approximation constraint [9].

The extraction of smooth iso-surfaces has also been recently performed using triangular rational quadratic Bézier patches [7].

3 MC with a trilinear reconstruction filter

The goal is to support a non-linear reconstruction filter in a surface fitting context. The proposed solution has been designed as an extension to the classical MC approach and follows the following list of requirements:

- surface fitting is performed with a *local* approach;
- given a generic reconstruction filter, the simplicial surface mesh produced must approximate the ideal iso-surface defined by the given reconstruction kernel at a user-selected approximation;
- C^0 continuity has to be ensured.

The idea is therefore to enhance the MC algorithm by giving the possibility to refine adaptively each surface patch until the requested precision is fulfilled (Figure 1). The overall pipeline is as follows (with V the volume dataset, q the iso-surface threshold, K the reconstruction filter, ε the given approximation precision, and $maxRec$ the maximum level of recursion which may be produced).

PreciseMC$(V, q, K, \varepsilon, maxRec)$:
FOR EACH cell $c_{i,j,k} \in V$ DO
fit an iso-surface patch S on $c_{i,j,k}$ (using standard MC);
FOR EACH face $f \in S$ DO
TryToRefine$(f, V, q, K, \varepsilon, maxRec, 1)$;

TryToRefine $(f, V, q, K, \varepsilon, maxRec, lev)$:
FOR EACH sampling_point p_i on f DO
evaluate the approximation ε_i of f in p_i with respect to filter K;
IF $\varepsilon_i > \varepsilon$ THEN $Split\_points := Split\_points + p_i$;
IF $Split\_points = \{\}$
THEN output(f)
ELSE refine f in $\{f_j\}$ (using $Split\_points$);
FOR EACH f_j DO
IF $lev \leq maxRec$
THEN TryToRefine$(f_j, V, q, K, \varepsilon, maxRec, lev + 1)$
ELSE output(f_j);

Mesh refinement is therefore adaptive, because we subdivide only those faces which do not approximate sufficiently the ideal iso-surface. The recursive refinement is halted either if a simplicial approximation which satisfies the given

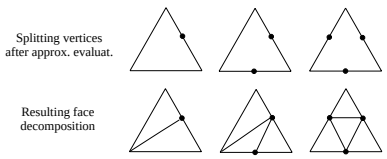

Fig. 2. Face refinement based on edge midpoints evaluation (rule A).

precision is found, or a maximum recursive level is reached. The user may therefore drive the fitting process by selecting two parameters: the approximation ε and the maximum number of refinement steps $maxRec$.

3.1 Evaluation of approximation and refinement rules

The precision of each face can be evaluated at least in two different manners. A first possibility is to measure a *field-based difference* (i.e. given a simplicial S mesh which approximates the reconstruction filter K, compute the difference between the given threshold value q and the value of the field in the points of S). A second approach is to measure a *geometric difference* between the current iso-surface S and the ideal iso-surface S_K (e.g. evaluate the Hausdorff distance between S and S_K). Both these evaluations may be performed in a precise or in an approximate manner.

We evaluate an approximate *geometric difference* by computing the distance between each face of S and the ideal iso-surface on a discrete number of sampling points. There are many different criteria to select the set of sampling points. A possible choice may be to select the midpoints of the edges (quaternary subdivision). For each of these points p_i, we evaluate the distance between p_i and a corresponding point p_i' on the ideal iso-surface S_K. If this distance is greater than the selected error threshold ε, we classify point p_i as a *splitting point*. Then we refine the current face by inserting the splitting points (the new local triangulation is simply determined by an ad hoc table, see Figure 2). We adopt therefore an heuristic refinement approach, to allow refinement of only a subset of edges. In this case, four different configurations are possible (the three ones represented in Figure 2 plus the one with no splitting vertices). Let us call this refinement criterion Rule A.

Other rules are also possible. One variation of Rule A is to evaluate four splitting points, adding the baricenter to the three edge midpoints (see Figure 3). Let us call it Rule B. Rule B has a disadvantage: because we evaluate all the four split candidates at once, we may decide to split on the central point also when its insertion does not really improve mesh approximation. For example, look at the case when the distance between the central point p_c and the ideal iso-surface S_K is greater than ε, and thus, according to Rule B, we use p_c to split face f. But, if we consider the refinement obtained by using only the other three split points, then in many cases the actual difference between the two refined meshes could

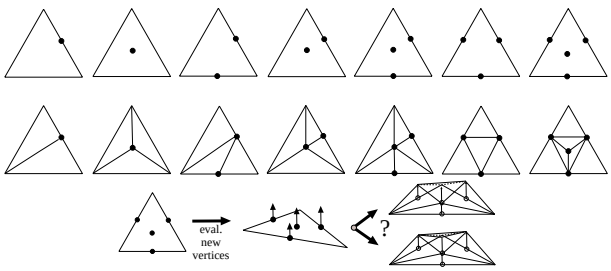

Fig. 3. Face refinement based on edge midpoint and center point evaluation (rule B); but center point splitting is not always necessary and increases substantially the resulting mesh size.

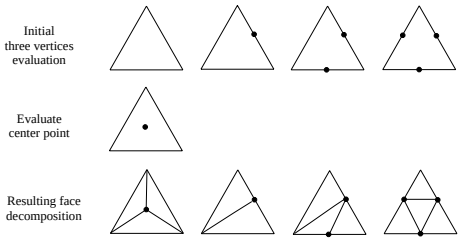

Fig. 4. Two phases evaluation, under Rule A1.

be much smaller than ε (see Figure 3). In that case we create three new faces that are not really needed to obtain the required approximation. To prevent an excessive increase in the number of faces due to the above reason, we introduce two alternative criteria based on four sampling points. The first one, called Rule A1, extends criterion A by sampling the central point in a second step, only when none of the three edge midpoints is classified as a splitting one (see Figure 4). The second one, Rule A2, always evaluates a fourth sampling point in a second phase. In this case, the initial location of this candidate point is not on the plane of the face to be split, but it depends directly on the current splitting points locations (after relocation on the ideal surface S_K, see Figure 5).

The three different criteria result into different meshes; see the evaluation of the results reported in Section 5.

3.2 Splitting point displacement

In the previous discussion we have not specified how do we find the point p' on the ideal iso-surface S_K which corresponds to the potential splitting point p we are evaluating. A solution is to start a sampling process on the line which originates from the current point p and is parallel to the field gradient in p itself. As far as we sample points on this line, we compute the field value and the current

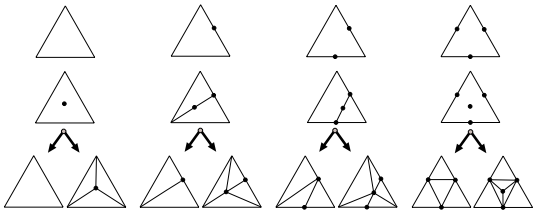

Fig. 5. Two phases evaluation, under Rule A2.

gradient using the reconstruction filter K. Sampling terminates as soon as we reach the searched value q (i.e. point p'), and we compute the Euclidean distance between p and p'.
Another manner to find point p' can be to analytically compute the nearest intersection between the gradient half-line and the local section of ideal isosurface S_K. This is surely possible in the case of a tri-linear reconstruction filter. But we preferred to adopt the previous solution, based on ray sampling, to be more general: given a reconstruction filter K, we only need to know how to compute K in a generic point p.
The robustness of geometrical computations is obviously a fundamental issue. All of the splitting points are shared between pairs of incident faces. To prevent the occurrence of different values in the replicated evaluation of a candidate splitting point (and potential topological inconsistencies), we must avoid redundant evaluations. All of the evaluated splitting point coordinates (plus accessory information, such as the local geometrical approximation and the gradients computed on such points) are therefore stored in a *hash table*, to prevent redundant evaluations.

4 Management of topologic anomalies

The proposed approach produces an adjustable-precision approximation of the ideal iso-surface by simply refining the standard MC linear patch. This implies that, given a cell and its configuration, we need the initial mesh patch to be topologically correct, otherwise the refinement process can produce erroneous results. Potentially ambiguous configurations of the MC look up table have been identified [4]. These are the configurations where pairs of vertices on a face which are connected by the diagonals have the same classification (both ON or OFF). In general, this problem can be managed with two different approaches. We can adopt a modified MC lookup table [15], which avoids cracks but does not ensure that the surface produced is topologically correct. Or we may directly extract topologically correct geometries, at the expenses of some overhead [18, 16]. The solution proposed by Natarajan [16] chooses for each cell the correct configuration by evaluating the value q' that correspond to a *saddle point* of the local interpolation function.

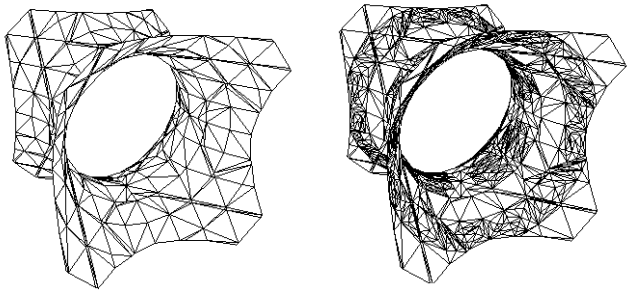

Fig. 6. Two meshes extracted from the "tube" dataset, using standard MC on the left and the enhanced precision method on the right (max recursion level = 3).

To ensure topological correctness of the initial patches we used the Natarajan's solution , and the overhead introduced resulted small if compared to the processing time spent for the evaluation of the splitting point approximation. But this solution has been defined for the tri-linear reconstruction filter, and how to extend it to the case of a generic filter is not straightforward.

5 Experimental results

The current prototypal implementation of the approach presented, PreciseMC, has been coded in C++. Its GUI has been designed to be used in a distributed client/server environment, using the Java language.
In the current implementation we only provide a trilinear reconstruction filter. All the times reported have been obtained on a PentiumPro 200Mhz (64 MB RAM). The adoption of an hash table to store splitting points and vertices data (coordinates, approximation error, interpolated normals) has proved highly effective both to improve robustness and to reduce computation times by avoiding redundant computations (in average, times are halved).

PreciseMC has been evaluated on many datasets. We report here results on some iso-surfaces fitted on an 11x14x14 section of the SOD dataset[1], and on two synthetic datasets: "tube" (resolution 9x9x9)[2], and "F" (resolution 30x30x4). Two isosurfaces fitted on the "tube" dataset are shown in Figure 6 (MC on the left, PreciseMC on the right). Figure 7 shows two meshes extracted from the SOD dataset with threshold 50. The one on the left was fitted using standard MC, and is composed of 654 faces. The mesh on the right was fitted with PreciseMC, using at most five levels of recursion. It is composed of 14,244 faces. Note the difference in the section where the iso-surface bifurcates: the mesh fitted with PreciseMC is much more smoother and thinner.

[1] SOD is a regular rectilinear dataset (electron density map of an enzyme), produced by D. McRee, Scripps Clinic, La Jolla (CA).
[2] A sample dataset defined and used in [7].

Table 1. Time and complexity of the iso-surfaces fitted on the the "tube" and "F" datasets using the three different splitting points evaluation rules (with 10 the max. number of recursive subdivisions).

"Tube" Dataset (9x9x9, threshold=130.5, MC times = 0.01 sec.)												
precision	**Rule A**				**Rule A1**				**Rule A2**			
(ε)	#faces	maxL	meanL	time	#faces	maxL	meanL	time	#faces	maxL .	meanL	time
1/1000	732	1	0.322	0.050	732	1	0.322	0.070	784	1	0.322	0.081
1/2000	1,256	2	0.782	0.090	1,256	2	0.782	0.130	1,486	2	0.782	0.171
1/4000	1,932	3	1.056	0.160	1,996	5	1.185	0.220	2,732	5	1.084	0.341
1/5000	2,308	4	1.258	0.190	2,420	7	1.484	0.270	3,380	7	1.250	0.420
1/10000	4,412	9	1.968	0.410	4,460	10	2.064	0.551	7,204	9	1.669	0.992
"F" Dataset (30x30x4, threshold=73.5 MC times = 0.01 sec.)												
precision	**Rule A**				**Rule A1**				**Rule A2**			
(ε)	#faces	maxL	meanL	time	#faces	maxL	meanL	time	#faces	maxL	meanL	time
1/100	2,708	7	0.801	0.210	2,726	7	0.805	0.300	3,914	4	0.784	0.471
1/200	4,556	9	1.188	0.411	4,638	10	1.201	0.570	7,726	10	1.205	1.041
1/400	8,350	9	1.558	0.881	8,814	9	1.579	1.142	16,742	10	1.548	2.864
1/800	14,238	10	2.035	1.803	14,450	10	2.040	2.273	32,350	10	2.016	6.830
1/1000	18,058	10	2.186	2.573	18,350	10	2.194	3.095	43,488	10	2.155	10.265

Table 2. Time and complexity of the iso-surfaces fitted on the "SOD" dataset with different settings for the approximation precision and the maximum number of recursive decompositions.

SOD dataset (11x14x14, threshold=50.5, MC times = 0.01 sec.)														
	precision = 1/100				**precision = 1/500**					**precision = 1/1000**				
maxL	1	2	3	4	1	2	3	4	5	1	2	3	4	5
meanL	0.492	0.674	0.748	0.772	0.843	1.516	1.948	2.138	2.212	0.902	1.689	2.292	2.552	2.843
no. faces	1,141	1,449	1,581	1,603	1,856	3,953	5,925	6,855	7,125	2,074	5,364	9,800	13,080	14,244
time	0.050	0.090	0.110	0.121	0.100	0.240	0.451	0.631	0.701	0.130	0.381	0.841	1.372	1.703

To evaluate the different results produced with the three refinement rules introduced in Section 3.1, we show in Table 1 the size (number of faces, $\#faces$) of the meshes produced by PreciseMC, under the same approximation precision ε. The table reports also the maximum ($maxL$) and mean ($meanL$) recursive depth required to reach the user-selected approximation ε. Precision is given using cell edge size units (e.g. ε=1/100 means precision not less than 1/100 of the cell edge). From the analysis of these results, we can say that Rule A1 has to be preferred, because it is fast, it is more precise than Rule A and it does not increase too much the size of the mesh. In particular, Rule A2 shows an excessive increase in the size of the mesh produced (nearly the double than the meshes produced with Rules A and A1).

In Table 2 the running times for the SOD dataset with different settings for the approximation precision and the maximum number of recursive decomposition are given.

We measured the actual difference between meshes extracted with MC and PreciseMC using the Metro tool [3]. Metro numerically compares two triangle meshes S_1 and S_2. It performs a surface sampling process on the first mesh, and for each elementary surface parcel it computes a point–to–surface distance with the other mesh. At the end of the sampling process, Metro switches the meshes and execute sampling again, to get a symmetric evaluation of the error.

Metro returns both *numerical* and *visual* evaluations of surface meshes "likeness". We have compared two meshes extracted from the SOD dataset (using the same threshold of Figure 7); the MC mesh is composed of 654 faces, and the PreciseMC one of 14,244. The Metro test gave a maximal distance between the two meshes of 0.39 units (i.e. cell edge length), and a mean distance of 0.12 units. A snapshot of the Metro output is shown in Figure 8; it is zoomed to view in detail the mesh section which describes the thin bifurcation.

6 Conclusions

A new iso-surface fitting solution has been presented, PreciseMC. Given a trilinear reconstruction filter, it improves the precision of the reconstruction process, with respect to standard MC solutions, using an approach based on mesh refinement. The iso-surface reconstruction process is adaptive, to ensure that the complexity of the fitted mesh will not become excessive. Three different refinement rules have been evaluated. Surprisingly, Rule A1 gave the best results; it required low processing times and a reduced increase in the size of the extracted meshes. From a qualitative point of view, the results obtained with PreciseMC are much smoother, more regular and, in some cases, also thinner than those produced with standard MC. PreciseMC shows therefore great potential in medical applications, where it may be selectively adopted to improve the quality of those surfaces which correspond to very thin specimens, such as blood vessels or other internal small cavities. This may improve either the measures taken on the extracted mesh (e.g. to evaluate the occurrence of stenosis or aneurysms in the vessel [19]) or the quality of virtual navigation [8]. Further research is needed to try to extend this approach to other reconstruction filters.

References

1. C.L. Bajaj, V. Pascucci, and D.R. Schikore. Fast isocontouring for improved interactivity. In *1996 IEEE Volume Visualization Symp.*, pages 39–46.
2. P. Cignoni, P. Marino, C. Montani, E. Puppo, and R. Scopigno. Speeding up isosurface extraction using interval trees. *IEEE Trans. on Visualization and Comp. Graph.*, 3(2):158–170, 1997.
3. P. Cignoni, C. Rocchini, and R. Scopigno. Metro: measuring error on simplified surfaces. Technical Report B4-01-01-96, I.E.I. – C.N.R., Pisa, Italy, January 1996.
4. M. J. Dürst. Letters: Additional reference to "Marching Cubes. *ACM Computer Graphics*, 22(4):72–73, 1988.
5. T. Fruhauf. Raycasting opaque isosurfaces in nonregularly gridded CFD data. In P.Zanarini R. Scateni, J.J. van Wijk, editor, *Visualization in Scientific Computing*, pages 45–57. Springer, Wien, 1995.
6. Mark Hall and Joe Warren. Adaptive polygonalization of implicitly defined surfaces. *IEEE CG&A*, 10(11):33–42, 1990.
7. B. Hamann, I.J. Trotts, and G.E. Farin. On approximating contours of the piecewise trilinear interpolant using triangular rational-quadratic Bézier patches. *IEEE ToVCG*, 3(3):215–227, 1997.

8. L. Hong, S. Muraki, A. Kaufman, D. Bartz, and T. He. Virtual voyage: Interactive navigation in the human colon. In *SIGGRAPH 97 Conference Proceedings*, pages 27–34.
9. L. Kobbelt. Discrete fairing. In *Proceedings of the Seventh IMA Conference on the Mathematics of Surfaces*, pages 101–131, 1997.
10. Y. Livnat, H.V. Shen, and C.R. Johnson. A near optimal isosurface extraction algorithm for structured and unstructured grids. *IEEE Trans. on Vis. and Comp. Graph.*, 2(1):73–84, 1996.
11. W.E. Lorensen and H.E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *Computer Graphics (SIGGRAPH '87 Proceedings)*, 21(4):163–170, 1987.
12. S. Marschner and R. Lobb. An evaluation of reconstruction filters for volume rendering. In *IEEE Visualization '94*, pages 100–107, 1994.
13. T. Möller, R. Machiraju, K. Mueller, and R. Yagel. Classification and local error estimation of interpolation and derivative filters for volume rendering. In *Proceedings 1996 Symp. on Volume Visualization (Oct. 28-29)*, pages 71–78, 1996.
14. T. Möller, R. Machiraju, K. Mueller, and R. Yagel. A comparison of normal estimation schemes. In *IEEE Visualization '97*, 1997.
15. C. Montani, R. Scateni, and R. Scopigno. A modified look-up table for implicit disambiguation of Marching Cubes. *The Visual Computer*, 10(6):353–355, 1994.
16. B. K. Natarajan. On generating topologically consistent isosurfaces from uniform samples. *Visual Computer*, 11(1):52–62, 1994.
17. Peter J. Neugebauer and Konrad Klein. Adaptive triangulation of objects reconstructed from multiple range images. In *IEEE Visualization '97 - Late-Breaking Hot Topics Session*, October 1997.
18. G.M. Nielson and B. Hamann. The asymptotic decider: removing the ambiguity in marching cubes. In *Visualization '91*, pages 83–91, 1991.
19. A. Puig, D. Tost, and I. Navazo. Interactive cerebral blood vessels exploration system. In *IEEE Visualization '97*, October 1997.
20. L. A. Sadarjoen and F.H. Post. Deformable surface techniques for field visualization. *Computer Graphics Forum*, 16(3):109–116, 1997.
21. L. Velho. Simple and efficient polygonalization of implicit surfaces. *Journal of Graphics Tools*, 1(2):5–24, 1996.
22. J. Wilhelms and A. van Gelder. Topological considerations in isosurface generation. *ACM Computer Graphics*, 24(5):79–86, Nov 1990.
23. Jane Wilhelms and Allen van Gelder. Octrees for faster isosurface generation. *ACM ToG*, 11(3):201–227, July 1992.

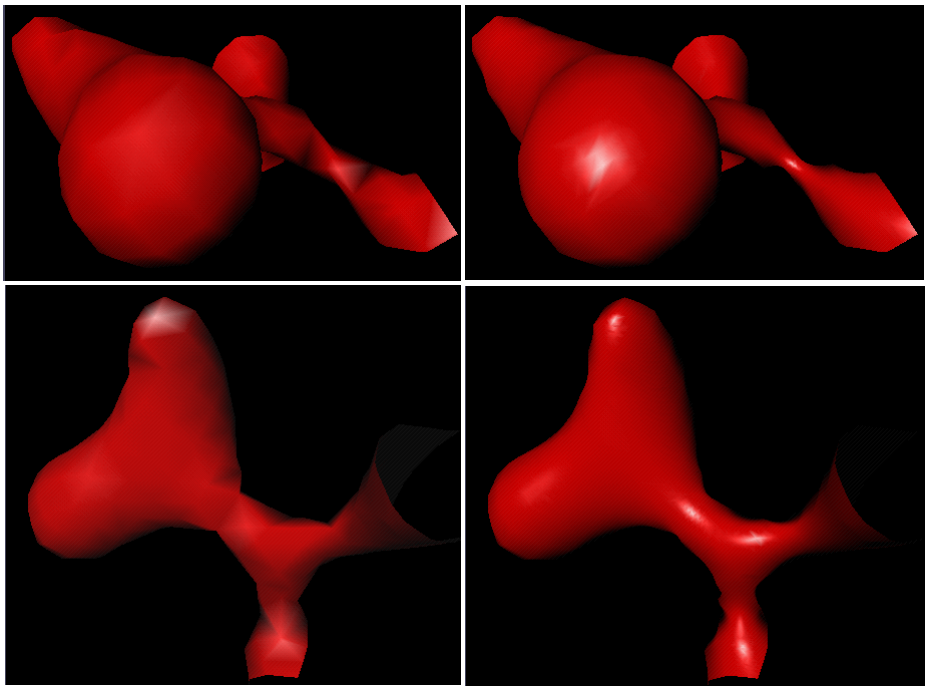

Fig. 7. Two meshes extracted from a section of the SOD dataset, using standard MC on the left and the enhanced precision method on the right (max recursion level = 4); above are two top views and a side view is used for the images below.

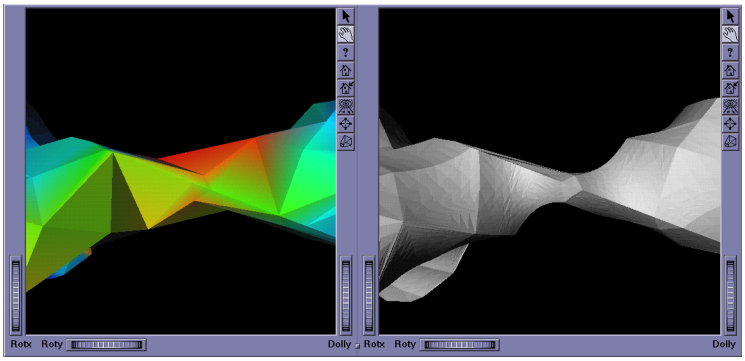

Fig. 8. Comparing iso-surfaces extracted (SOD dataset) with the Metro tool; a section of the MC mesh (left image) is colored according to the distance from the corresponding mesh section extracted with PreciseMC (right image).