Performance Report on Adaptive Resolution Isosurface Algorithm

Ramakrishnan Kazhiyur-Mannar*
Dept of Computer Science and Engineering

Rephael Wenger†
Dept of Computer Science and Engineering

Abstract

Isosurfaces obtained from densely sampled datasets often contain a large number of simplices that do not provide any useful topological or geometric information. It might be beneficial if one could replace them without losing too much information with fewer, larger simplices.

An algorithm to identify regions in the dataset where there was no significant topological event happening was proposed by us in an earlier paper [R. Kazhiyur-Mannar et al. 2003]. The paper also described an algorithm to construct crack-free isosurface from an adaptive resolution dataset.

We report the performance statistics of our implementation of the algorithms.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Hierarchy and geometric transformations;

Keywords: adaptive resolution, isosurface, multiresolution, surface simplification, controlled topology simplification

1 Introduction

Given a continuous scalar field, i.e., a scalar function of $\mathbb{R}^d$, an *isosurface* is a set of points with identical scalar values. Lorensen and Cline [Lorensen and Cline 1987] gave a simple, efficient algorithm, called the Marching Cubes algorithm, for constructing a polyhedral approximation of an isosurface from a regular grid sampling of a scalar function in $\mathbb{R}^3$. The regular grid divides a volume into cubes whose vertices are the grid vertices.

The Marching Cubes algorithm reconstructs the isosurface within each cube and returns the union of these patches as the approximation of the isosurface. Various modifications were proposed to avoid cracking problems in the original algorithm [Montani et al. 1994; Nielson and Hamann 1991].

Subsequently, Bhaniramka, Wenger and Crawfis [Bhaniramka et al. 2000] gave a hypercube based Marching Cubes algorithm in higher dimensions. The higher dimensional algorithms reconstruct $(d-1)$-dimensional surface patches within simplices or hypercubes and the union of these surface patches forms the isosurface.

In order to capture the topology of the underlying isosurface accurately, and ensure a close approximation of its geometry, these algorithms require a dense sampling of the data. Consequently, the output surface mesh has a high simplex count. This increases the storage space, and rendering time it requires.

Many of these simplices represent relatively flat portions of the isosurface, and could be replaced by fewer large simplices. One way of reducing isosurface complexity is to generate flat portions of the isosurface from larger mesh elements, ignoring the grid vertices in the interior of those regions.

A grid with pairs of different sized elements adjacent to each other has some grid vertices belonging to the smaller element of a pair that is not a vertex of the larger element. Thus, it is not a complex. At the intersections, the isosurface develops cracks, as described in [Müller and Stark 1993].

In an earlier work [R. Kazhiyur-Mannar et al. 2003], we described certain conditions for adaptively subsampling a dataset in $\mathbb{R}^d$, which could provide guarantees regarding the topology of the output isosurface. We also described a new algorithm to handle the cracking problem. In this work, we report the performance of our implementation of the algorithms.

In Section 2, we describe the strategies we proposed in [R. Kazhiyur-Mannar et al. 2003]. In Section 4, we provide analysis of some results we have obtained from our implementation, including an analysis of the parameters which could affect the performance of the algorithm.

2 Algorithm

We will now provide a brief overview of our adaptive sampling strategy, and the strategy we adopt to solve the cracking problem. We refer the reader to [R. Kazhiyur-Mannar et al. 2003] for detailed description of our algorithm and references to previous work.

Our adaptive subsampling approach is similar to that described in [He et al. 1996]. A grid vertex is labeled '+' if its scalar value is greater than the isovalue and '-', otherwise. A region where all '+' vertices precede the '-' vertices or vice versa in an axis-parallel direction is *monotonic* in that direction.

A rectangular region in the labeled grid has the *monotonicity property* if it is either zero dimensional or it is monotonic in any axis-parallel direction, and both the facets of the region orthogonal to that direction have the monotonicity property. We stated, along with an outline of the proof, that the isosurface patch inside a rectangular region that has the monotonicity property is a $(d-1)$-disk.

In [Müller and Stark 1993], it was proved that if subsets of a $2D$ isosurface, each of which is a 1-disk, are piece-wise replaced by other 1-disks, the global topology of the isosurface is preserved, provided there are no self-intersections. In [R. Kazhiyur-Mannar et al. 2003], we argued that this generalizes to higher dimensions too.

Our adaptive subsampling strategy is to rectangular regions that satisfy the monotonicity property, and replace them with a d-dimensional hypercube. The isosurface patch that is obtained from this hypercube, as described in [Bhaniramka et al. 2000], would be a $(d-1)$-disk, and hence, a topology preserving replacement for the

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

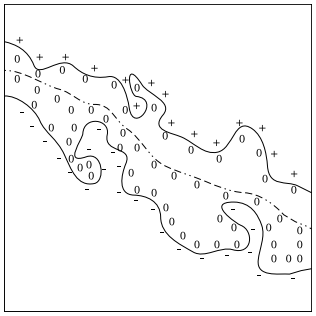

Figure 1: Tolerance Band: In the modified version, instead of considering the true isosurface, we consider the interval volume between two isosurfaces around the true isosurface. If a disk can be fit into this volume, (as shown as the dotted line) the region is monotonic.

intersection of the region with the full-resolution isosurface, also an $(d-1)$-disk.

The detection of regions could proceed in either both top-down and bottom-up fashion. The relative advantage of using top-down approach is that we would merge bigger regions first, reducing the candidates to be considered later. However, if the isosurface is very complex, this could be a disadvantage, since we would encounter many failures in the first iterations. In bottom-up approach, we could also consider caching the monotonicity information from lower levels and computing the monotonicity of a region from the sub-regions that encompass it.

To fix cracks, we adopted a novel approach of hexahedral mesh collapses, where every vertex of the hexahedral mesh which is not corner of *every* hypercube it is incident on (called a *hanging vertex*) collapsed to a corner of a hypercube of which it is not a corner. This results in a *conforming* grid, which is a complex. Though the elements of this grid are no more hypercubes, we could consider them as hexahedral elements, with some edges having zero length, and use the algorithm of [Bhaniramka et al. 2000] to extract the conforming isosurface.

The crack-fixing method, however could alter topology due to multiple collapses if the corner to which a hanging vertex collapses itself is a hanging vertex. To preclude this possibility, we introduced a preprocessing stage, called *step-down* in which each block with a neighbor smaller than itself is subdivided in that direction in which the neighbor is smaller. This makes all hanging vertices stable, but could create new hanging vertices. However, these new vertices would *always* collapse to vertices that are stable, precluding multiple collapses.

Controlled topology simplification may be required if dataset has topological features that are insignificant. Standard image processing methods, such as salt-and-pepper noise removal could be extended to higher dimensions for this purpose. However, there could be features that are geometrically large, but created because of insignificant variation in the height field of the underlying dataset.

For such cases, we proposed a *tolerance band* scheme where the monotonicity condition was altered so that if a volume interval of a user-specified thickness (in terms of height value) around the isosurface contains an $(d-1)$-disk, the region could be merged.

The error introduced by the merger of voxels in a region invariably increases with the size of the region, and it bound by it. To reduce geometric distortion, we limit the maximum size of the region considered for merger. This, however, invariably leads to a loss in simplification achieved.

3 Datasets

We report results of our adaptive resolution algorithm on 3D and 4D data using different levels of adaptive subsampling, defined by the maximum block size that is allowed, and tolerance band widths. The performance measurements were done on a machine with Intel®Xeon™2.8 GHz processor and 2 GB RAM. All times shown are wall times, and include overheads such as disk I/O and caching.

We used medical data sets from University of Iowa's radiology data set collection, and a online collection of data sets by Dr. Stefan Roettger of the University of Erlangen. For 4D data, we used 33 time steps of the Vortex and Jetstream data sets.

The dimensions of data sets used in this section and the size of the full resolution isosurfaces are listed in Table 2.

In our analysis of the performance of our algorithm, we measure the simplification achieved in terms of the number of simplices in the adaptive resolution mesh, and the total time taken by the algorithm. We measure wall times and not processor times. Thus, they include the overheads including time taken for swapping memory and disk I/Os.

4 Results

4.1 Cost of Individual Stages

To analyze the effect of each stage on the performance, we consider their individual performance in terms of their execution time and the effect on the simplification of the surface.

The simplification cannot be measured directly after each stage. However, we could get an idea of the progression of the simplification from the number of blocks that are left at the end of each stage.

Figure 3 shows a break-up of time spent in each stage. We note that, understandably, subsampling is the costliest of the stages. For our sample data, subsampling stage accounts for between 71% to 82% of the total running time. The percentage is higher for datasets that did *not* yield good simplification. This is primarily because the stages that follow run slower. Thus, the overall running time for these datasets is correspondingly high.

Figure 4 shows a comparison of the number of blocks that are left after each stage. Obviously, we want less number of blocks. However, while subsampling reduces the number of blocks, balancing and step-down *increase* that number and grid conforming does not affect it.

If the dataset is subsampled more-or-less regularly, we would find less number of blocks split for balancing. We find that for all our datasets, the ratio between the number of blocks left after step-down to those left after subsampling is between 2 and 3. Even for very smooth datasets, such as geometric datasets, this holds true.

4.2 Effect of User Specified Parameters

Besides the topology and geometry of the isosurface itself, there are three user defined parameters that could affect the performance

Dataset	Dimensions	IsoValue	Size (# simplices)		Time
			Full Resolution	Decimated	(sec)
Foot	256x256x256	75	1,979,452	428,522 (21%)	1860
Lobster	301x324x56	25	464,800	158,342 (34%)	258
Sheep Heart	352x352x256	50	4,012,826	476,619 (11%)	3660
Visible Male (Head)	128x256x256	60	1,067,744	122,114 (11%)	543
Visible Male (Head)	128x256x256	70	1,841,056	881,532 (47%)	1110
CT-Chest	384x384x240	100	4,016,014	1,203,719 (29%)	1968
CTA-Brain	512x512x120	100	3,360,896	806,600 (24%)	858
Daisy Pollen	192x180x168	60	515,620	178,711 (34%)	468
Engine	256x256x256	25	1,214,992	74,495 (6%)	216
Frog	256x226x44	50	426,072	152,743 (35%)	198
Baby	256x256x98	75	737,270	81,776 (11%)	126
Jetstream (4D)	104x128x128x33	0.0001	***	17,733,459	300 min
Vis Male (Interval Volume)	128x256x256x17	[56,73]	***	3,035,579	132 min

Figure 2: Sizes of the simplified isosurface when the decimation is performed with maximum block size set to 4. Note that these do not use tolerance bands, and hence, are topology preserving.

Dataset	IsoValue	Partitioning (sec)	Subsampling (sec)	Balancing (sec)	Step-down (sec)	Conforming (sec)	Surface Extraction (sec)	Total Time (sec)
VW Head	1100	57	417	36	13	15	20	558
Bucky Balls	50	0	1	0	0	0	18	19
Foot	75	12	175	260	99	230	5,213	5,987
Lobster	25	3	24	12	4	22	694	759
Sheep Heart	50	24	135	79	32	103	2,518	2,887
Visible Male (Head)	60	3	31	21	6	15	406	482
Visible Male (Head)	70	4	42	82	25	76	1,971	2,200
CT-Chest	100	16	146	171	56	150	3,553	4,092
Tooth	250	5	44	42	14	39	949	1,093
Daisy Pollen	60	3	23	18	8	18	436	506

Figure 3: Comparing Running Times of Individual Stages

of the algorithm. In this section, we describe three important parameters that might affect the algorithm's performance, in terms of simplification achieved, and running time.

4.2.1 Maximum Block Size

Surface simplication commonly entails a loss of geometry. In our algorithm, the allowable geometric distortion could be controlled easily by limiting the maximum size of the regions considered for merger.

This parameter is especially important if we use top-down approach, because, absent this parameter, the whole dataset would have to be considered for merger, and checking the monotonicity of the whole dataset might be very costly.

For a smaller value of the maximum block size, the algorithm spends less time in the subsampling phase. The resultant blocks also tend to be more uniform. Therefore, less number of blocks would require changes during balancing, step-down and conforming. However, the increase in the number of blocks would offset these advantages. The extent of simplification obtained may drastically reduce for very small values.

Performance of the algorithm for various values of the maximum block size allowed is Figure 5. When the maximum block size is set at 2, we get twice as many simplices as that for a maximum block size of 4. In contrast, the difference between the simplex counts for block sizes of 8 and 16 are hardly different. In most cases, the running time for a maximum block size of 16 is *more* than that for a block size of 8. This is because of the time the algorithm spends in trying to find very big monotonous blocks, which rarely exist.

For most real datasets, we found that the simplification was optimal for a maximum block size of 4. This could be different for very smooth (uniformly varying) datasets, where the simplification increases with the maximum block size allowed.

4.2.2 Tolerance Width

In topologically complex, or especially noisy datasets, the effect of tolerance could be drastic. Tolerance improves both running time and the simplification achieved. However, the topology of the isosurface is not necessarily preserved when a tolerance is specified. A large value for tolerance might lead to elimination of features and significant components of the isosurface.

Table 6 shows the improvemnt in performance of our algorithm when tolerance bands are used. Note the difference in the improvement seen in the case of the two isosurfaces of the Visible Male dataset. The surface corresponding to isovalue 60 represents the skin, which is smooth, and does not comprise of many small topological features. The surface corresponding to isovalue 70, which represents the bone has many non-persistent internal structures. Hence, even a narrow band yields tremendous benefit in the case of the bone surface, while in the case of the skin surface, it has comparitively small effect.

Dataset	IsoValue	Post-Subsampling	Post-Balancing	Post-Step-down
VW Head	1100	199,903	232,033	471,669
Bucky Balls	50	1,872	3,052	7,183
Foot	75	496,917	656,999	1,189,315
Lobster	25	59,993	94,513	199,125
Sheep Heart	50	205,801	333,118	689,079
Visible Male (Head)	60	57,857	84,699	163,311
Visible Male (Head)	70	268,178	416,050	863,704
CT-Chest	100	394,088	635,122	1,365,759
Daisy Pollen	60	84,765	106,188	170,908
Engine	25	60,212	79,160	130,441
Tooth	250	181,405	241,119	415,987

Figure 4: Number of Blocks left after each stage

Dataset	IsoValue	Final Simplex Count				Time (s)			
		m=2	4	8	16	m=2	4	8	16
VW Head	1100	821,623	429,151	388,293	386,493	921	570	537	548
Bucky Balls	50	4,452	6,055	7,384	7,736	19	15	19	9
Engine	25	173,087	74,532	92,528	139,338	656	216	222	292
Lobster	25	152,691	158,592	178,497	187,835	753	634	759	853
Sheep Heart	50	—	476,619	537,803	609,922	—	3,655	2,887	3,348
Visible Male (Head)	60	186,716	122,114	130,147	147,465	1,907	544	482	695
Visible Male (Head)	70	—	—	881,532	906,215	—	—	2,200	2,355
CT-Chest	100	1,337,323	1,203,719	1,278,761	1,350,177	6,109	3,930	4,092	4,597
Tooth	250	325,821	310,257	326,074	345,441	3,032	1,194	1,093	1,126
Daisy Pollen	60	181,503	178,711	180,392	183,339	677	472	506	533
Jet Stream (4D)	0.001	—-	17,733,459	—	—	—	300 min		

Figure 5: Final Simplex Count for Maximum Block Size

4.2.3 Minimum Component Size

Salt-and-pepper noise in the dataset is removed by specifying the minimum size of a component of the isosurface. A union-find operation is used to detect components in the dataset, and all components that do not contain the minimum number of grid vertices specified are artificially eliminated by flipping the signs of the vertices in them.

Eliminating small components may increase or decrease the running time, and may increase the simplification obtained. If the dataset is noisy with the isosurface having small floating components, the performance could improve (sometimes drastically). If, however, the dataset does not have too much noise, the cleanup operation would be wasteful, reduce the performance in terms of running time.

References

BHANIRAMKA, P., WENGER, R., AND CRAWFIS, R. 2000. Isosurfacing in higher dimensions. In *Proceedings of the Conference on Visualization '00*, IEEE Computer Society Press, 267–273.

HE, T., HONG, L., VARSHNEY, A., AND WANG, S. W. 1996. Controlled topology simplification. *IEEE Transactions on Visualization and Computer Graphics 2*, 2, 171–184.

LORENSEN, W., AND CLINE, H. 1987. Marching cubes: A high resolution 3d surface construction algorithm. In *Proceedings of the 14th annual conference on Computer Graphics and Interactive Techniques*, ACM Press, 163–169.

R. KAZHIYUR-MANNAR, WENGER, R., CRAWFIS, R., AND DEY, T. 2003. Adaptive resolution isosurface construction in three and four dimensions. Tech. Rep. OSU-CISRC-7/03–TR38, Department of Computer and Information Sciences, The Ohio State University.

MONTANI, C., SCATENI, R., AND SCOPIGNO, R. 1994. A modified look-up table for implicit disambiguation of marching cubes. *The Visual Computer 10*, 6 (December), 353–355.

MÜLLER, H., AND STARK, M. 1993. Adaptive generation of surfaces in volume data. *The Visual Computer 9*, 4, 182–199.

NIELSON, G., AND HAMANN, B. 1991. The asymptotic decider: Rosolving the ambiguity in marching cubes. In *IEEE Visualization 1991*, IEEE Computer Society Press, IEEE, 83–91.

Dataset	IsoValue	Simplex Count Without Band	Time Without Band	Simplex Counts With Bands		Times (s) With Bands	
				A	B	A	B
Bucky Balls	50	7,384	19	6,695 (10)	6,430 (15)	17	16
Lobster	25	178,497	759	129,461 (10)	146,283 (5)	565	619
Visible Male (Head)	60	130,147	482	103,808 (5)	98,612 (7)	429	413
Visible Male (Head)	70	881,532	2,200	162,243 (5)	156,328 (10)	429	413
CT-Chest	100	1,278,761	4,092	302,020 (10)	84,576 (20)	1,080	679
Engine	25	92,528	448	60,590 (10)	70,100 (5)	313	351
Daisy Pollen	60	180,392	506	152,475 (8)	124,430 (16)	436	376
Baby	75	737,270	302,020	81,776(5)	73,115(10)	220	240
Frog	50	426,072	152,743	144,959 (10)	138,629 (15)	205	224

Figure 6: Effect of Tolerance Band width.

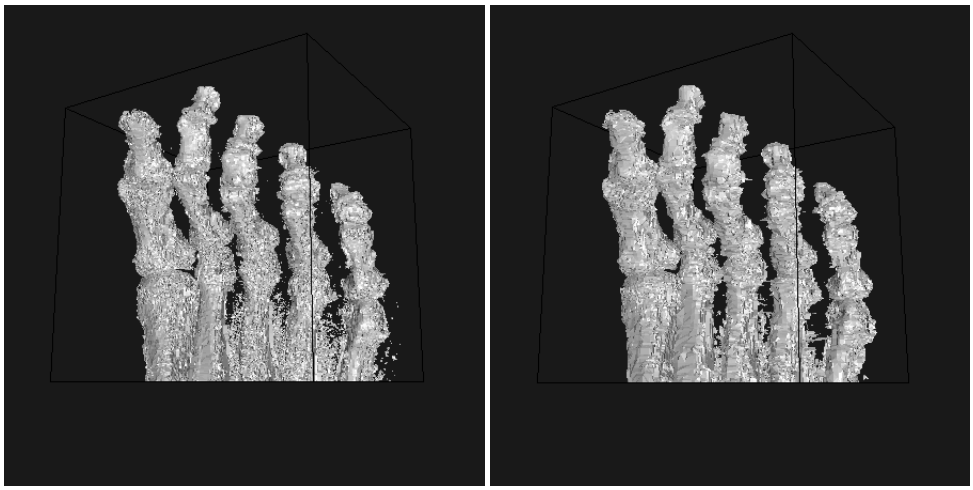

Figure 7: Foot dataset without tolerance band (1,261,993 triangles) and with tolerance band of width 15 (580,647 traingles).

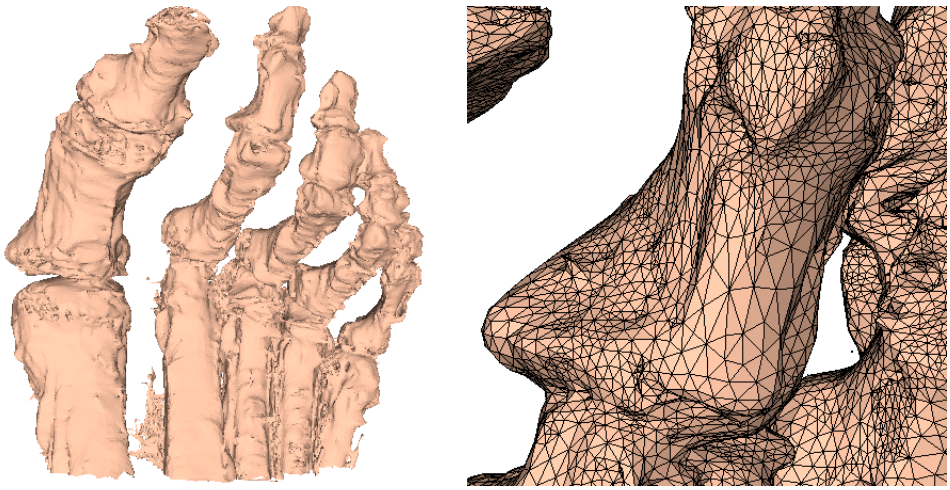

Figure 8: Foot isosurface, decimated with maximum block size set to 4. A closer look at the surface shows the adaptive resolution of the surface.

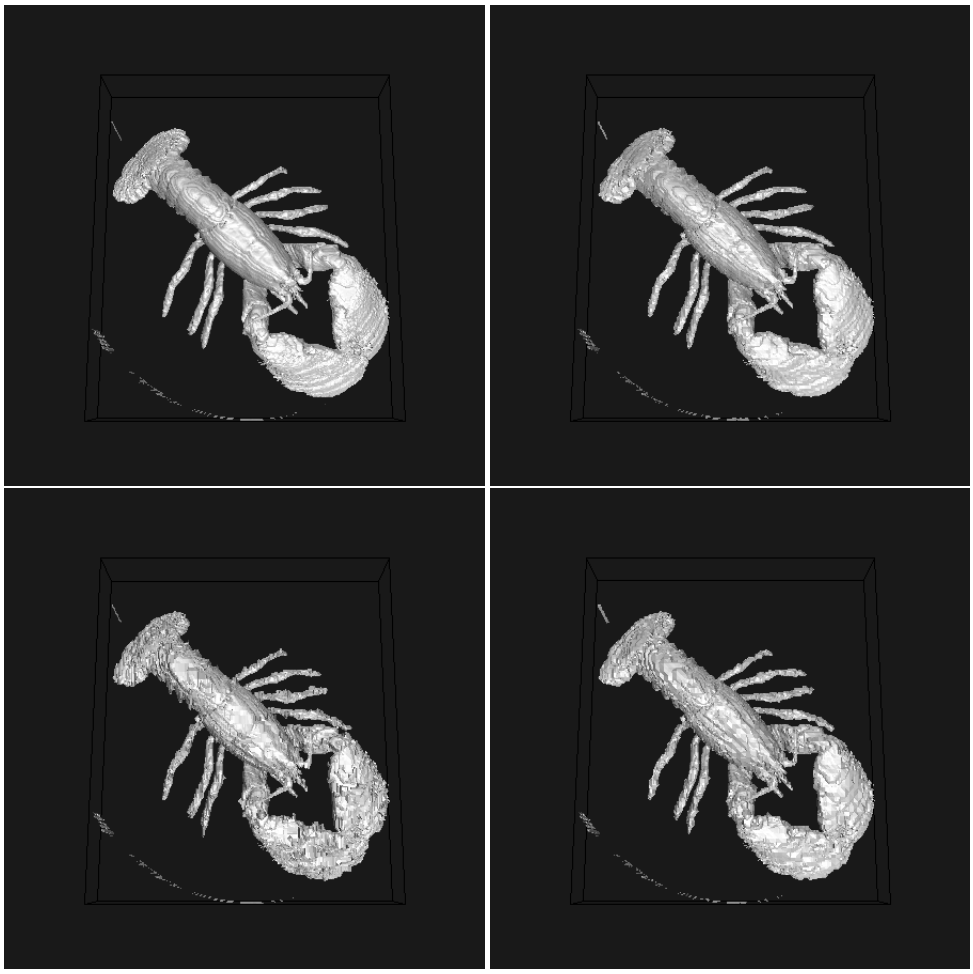

Figure 9: Lobster at different resolution levels. Clockwise from top-left: Full Resolution (232,692 triangles), adaptive resolution with maximum block sizes set at 2(152,691 triangles), 4(158,592 triangles), and 8 (178,497 triangles).

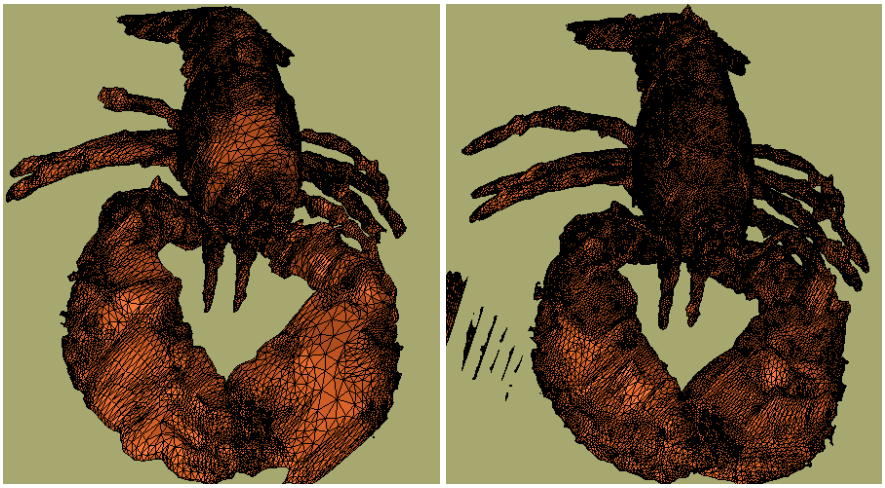

Figure 10: Lobster isosurfaces at adaptive resolutions with maximum block sizes of 8 and 4. Notice the variation in the isosurface resolution in the relatively flat regions such as the pincer, compared to the smaller features like antennae.

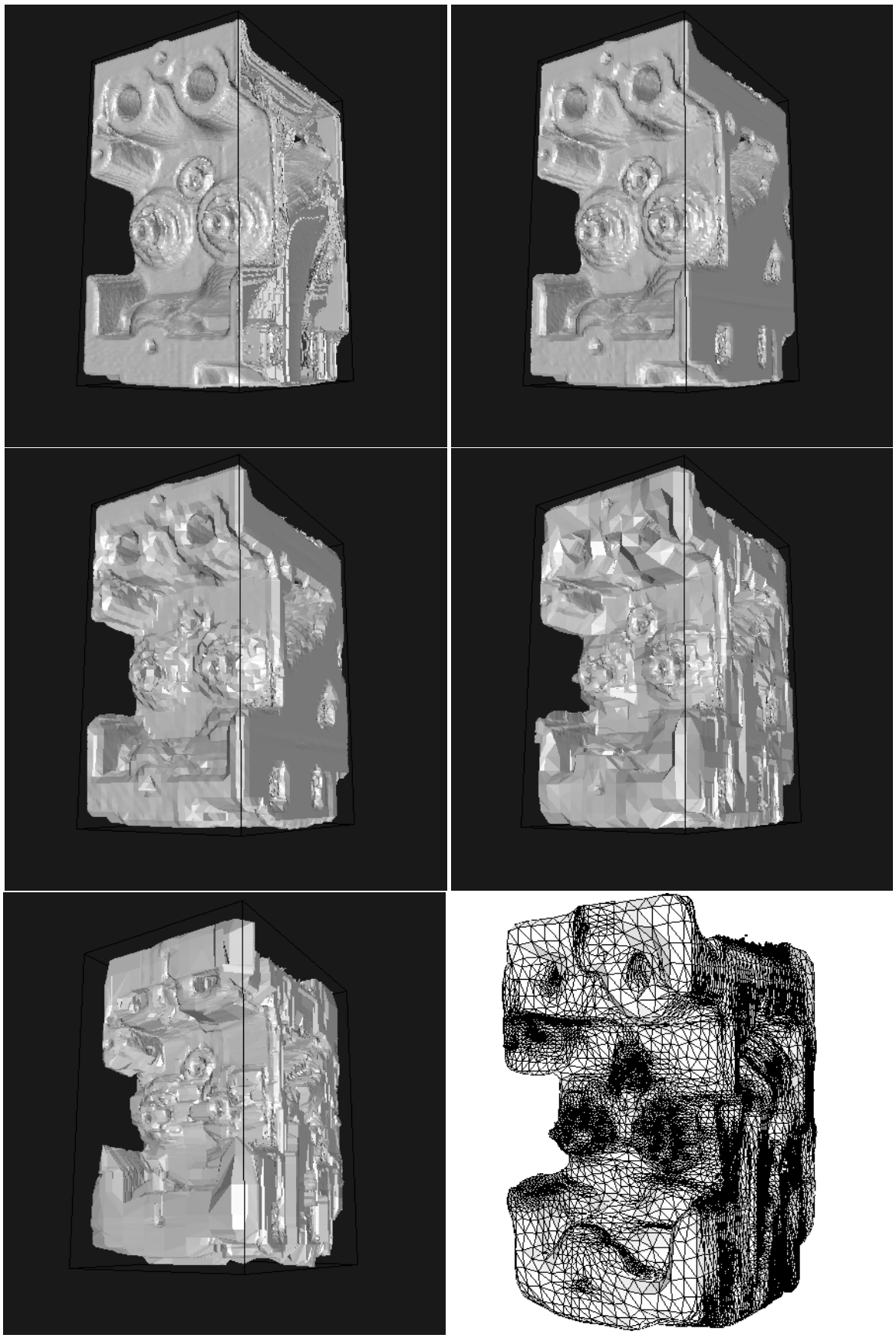

Figure 11: Engine at different resolution levels: Full Resolution (1,226,632 triangles), and at maximum block sizes 2 (173,087 triangles), 4 (74,532 triangles), 8 (92,528 triangles), and 16(139,338 triangles). The last image shows the variation in the resolution in the isosurface with maximum block size 8.

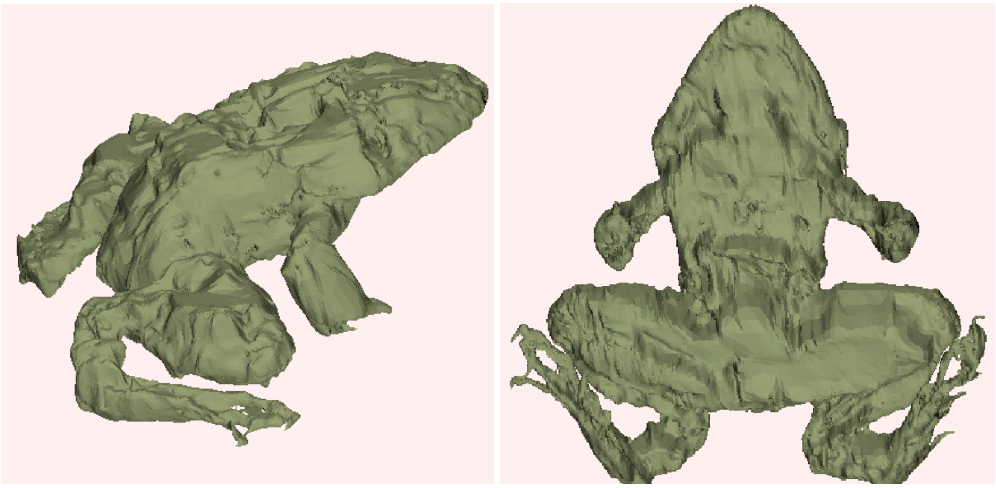

Figure 12: Isosurface from the Frog dataset. The resolution at the toes is maintained at a higher level than in the body region. The use of tolerance has resulted in simplification of the internal structures of the surface.

Figure 13: Isosurfaces from the Visible Male dataset adaptive resolution isosurface (with maximum block size set at 4).

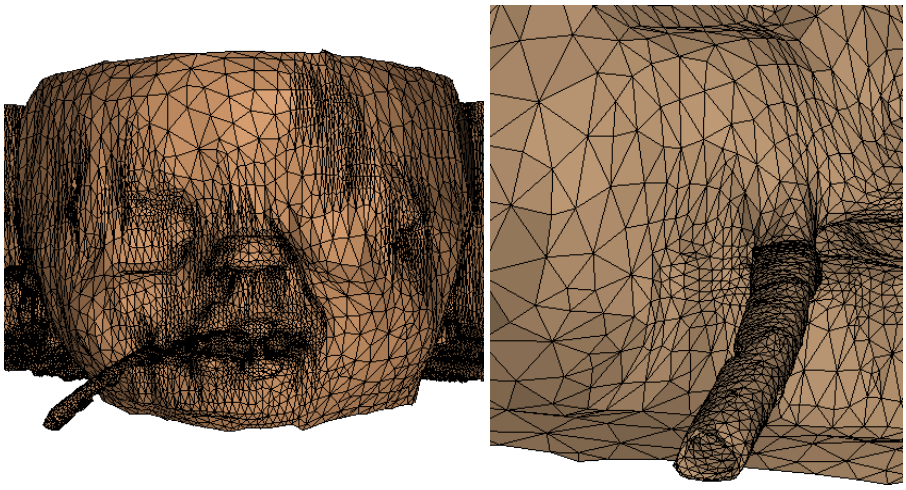

Figure 14: Adaptive resolution isosurface (with maximum block size set at 4) in the Baby dataset. The closer view in the right shows the variation in the resolution in the regions with small features such as the corner of the lips, compared to flatter regions such as the cheek.