Simplicial Subdivisions and Sampling Artifacts

Hamish Carr
Dept. of Computer Science
Univ. of British Columbia

Torsten Möller
Dept. of Computer Science
Simon Fraser University

Jack Snoeyink
Dept. of Computer Science
Univ. of North Carolina - Chapel Hill

Abstract

We review several schemes for dividing cubical cells into simplices (tetrahedra) in 3-D for interpolating from sampled data to $\mathbb{R}^3$ or for computing isosurfaces by barycentric interpolation. We present test data that reveal the geometric artifacts that these subdivision schemes generate, and discuss how these artifacts relate to the filter kernels that correspond to the subdivision schemes.

1 Introduction

In scientific applications, data is frequently generated or sampled on a regular rectilinear grid in three dimensions. The sampled data values are extended to the entire space by some type of interpolation. The interpolated function is then visualized, often with isosurfaces, the three-dimensional analogue to contour lines on a topographic map. Isosurfaces may be displayed directly [13], or constructed as a geometric object for the purpose of segmentation [3]: the extraction of significant boundaries - for example, the surface of a bone in a medical data set.

Typically, isosurfaces are generated using a trilinear interpolation function over each unit cube [8, 13], although a tricubic interpolation function has been suggested [26]. However, correctly approximating even a trilinear interpolation function is relatively complex and costly [8], so many researchers have instead opted to work with tetrahedral approximations. To approximate the trilinear interpolation function, each cubical cell is divided into 5, 6, or more tetrahedra. The function is then interpolated using barycentric interpolation over each tetrahedron.

This approach has several advantages. Cubes may be subdivided implicitly, and generated at run-time as needed, or generated explicitly and stored. Each tetrahedron has 2 cases (after symmetries) for generating isosurfaces, compared with 15 for Marching Cubes [13], or as many as 38 for the topologically correct version [8]. Some techniques require tetrahedra [7, 14, 23, 22, 24] in order to function at all. Finally, barycentric interpolation on tetrahedra is monotonic. All critical points must be at vertices of the mesh: this makes the task of locating critical points much simpler.

These advantages do not come without a price. The number of cells in the mesh increases by a factor of 5 or more, and the number of polygons (usually triangles) generated for a single isosurface typically increases by a factor of 2 or more, compared with Marching Cubes [13]. Additionally, the barycentric interpolation does not match the assumed trilinear interpolation function, causing geometric artifacts in the surfaces constructed, some of which are distinctly visible when rendered.

In Sec. 2, we review the existing literature on simplicial subdivisions of regular meshes in 3-D. In Sec. 3, we present a set of desiderata to be considered when analysing any particular subdivision. Then, in Sec. 4, we apply these desiderata to various 3-D subdivisions. In Sec. 5, we discuss the relationship between the artifacts seen, and the sampling rate of the image being considered. Finally, we present our conclusions in Sec. 6.

2 Previous Work

Several authors have discussed the advantages of simplicial subdivision of cubical cells [1, 4, 5, 20]: the analyses in these papers have focussed on the number of triangles generated [20], whether the surface generated is topologically consistent [1, 20], and whether the surface is topologically correct [20].

The simplicial subdivision most often used is, understandably, the 5-fold minimal subdivision (Sec. 4.2), in which each cube is subdivided into 5 tetrahedra, and no additional vertices are required [1, 4, 5, 14, 17, 19, 20, 22]. Also reported [1, 2, 10, 12, 17, 20] is the Freudenthal subdivision (Sec. 4.3), in which each cube is subdivided into 6 tetrahedra, arranged around a major diagonal of the cube. In our own work [6, 7], we have experimented with the body-centred cubic lattice (Sec. 4.4), which generates an average of 12 congruent tetrahedra per cell. Bloomenthal [4] uses a 12-fold subdivision (Sec. 4.5), in which each cube is divided into 12 tetrahedra, using the original cube vertices and the centroid of the cell. Albertelli & Crawfis [1] extended this subdivision to 14, 16, 18, 20, 22 & 24 tetrahedra, by adding face centroids to the list of vertices. We reported a set of abstract desiderata for simplicial subdivisions, and, based on these desiderata, chose the 24-fold subdivision (Sec. 4.6), also used in [1, 2, 11, 25]. A 48-fold subdivision (Sec. 4.7), equivalent to a 6-fold subdivision on a finer grid is also reported in [12, 27].

Instead of tetrahedral subdivisions, many researchers have worked directly with the cubes. The principal such technique is "Marching Cubes" [13], which, unfortunately, is topologically inconsistent. One solution [8, 18] is to add more cases, based on the saddle points of the trilinear interpolation function, but this can lead to a large number of possible cases.

3 Desiderata For Simplicial Subdivision

In order to analyse subdivisions effectively, we must first consider what goals we wish to achieve in subdividing cubical cells. As there are several potentially inconsistent goals, it will be impossible to say that one subdivision is best under all circumstances, so we simply state a set of desiderata

Presented at IEEE Visualization 2001
October 21 - October 26, 2001
San Diego Paradise Point Resort, San Diego

for subdivision, based on our previous work [6, 7]. Each desideratum will be assigned a name, for ease of reference.

We assume that the data has been sampled on a regular grid in three dimensions. Although much of the analysis below applies equally to curvilinear and irregular grids, we have not considered these grids in detail to date.

We further assume that the original interpolation function $f(p)$ is trilinearly interpolated over the cube, unless stated otherwise. For each subdivision, we replace the original mesh with a tetrahedral mesh, and perform barycentric interpolation over each tetrahedron. This implicitly substitutes a new interpolated function $F(p)$ for the original $f(p)$. This definition means that $F(p)$ is a piecewise-linear interpolation function.

i) Contained: the interpolated value $F(p)$ at a given p should not depend on any vertices that $f(p)$ does not depend on. For the trilinear interpolation function f, this is equivalent to requiring that $F(p)$ depend only on the vertices of the cube in which p is contained.

ii) Symmetric: the subdivision should be symmetric under rotations and reflections of the cube.

iii) Parsimonious: the subdivision should not magnify the dataset - i.e. no data points should be added.

iv) Minimal: the subdivision should use as few simplices as possible.

v) Implicit: if possible, the subdivision should be implicit, for processing efficiency. One of the principal advantages of a regular mesh is that adjacency information need not be stored explicitly: instead, vertex (x, y, z) is considered to be adjacent to $(x + 1, y, z)$, and so on. Where possible, it is advantageous to avoid explicit storage of adjacency in the subdivided mesh.

vi) Continuous: $F(p)$ should be continuous between adjacent cells. This prevents cracks or holes forming during isosurface generation. Fig. 3(d) illustrates cracks formed from a discontinuous subdivision.

vii) Correct: the subdivision should minimize the difference between $F(p)$ and $f(p)$. In this paper, we consider the topological differences in Sec. 4, the effects of the subdivisions on the filter kernel in Sec. 5, and some numerical results in Sec. 4.8. We note however, that "correctness" is intrinsically difficult to quantify.

4 3-D Subdivisions

In this section, we give a brief taxonomy of subdivisions, and test each subdivision against the desiderata stated above. The results are summarized in Fig. 1. We also illustrate the visual artifacts for each subdivision in Fig. 3, using a small synthetic dataset, which we discovered to be particularly effective for this purpose. The principal feature of this dataset is a zigzag arrangement of peaks aligned with the sampling grid. Nine Gaussian distributions were placed in a volume of $13 \times 13 \times 13$, with different scales, and summed. The formula is provided in Fig. 2.

4.1 Taxonomy of Subdivisions

Tetrahedral subdivisions can be constructed either by restricting the subdivision to the vertices already in the grid, or by adding additional vertices to the grid. Additional vertices can be inserted along an edge of the cell, in a face of the cell, or in the body of the cell. Although it is logically possible to construct subdivisions with face (or edge) vertices added, but the body vertex omitted, we are unaware of any cases where this has been done. The reason for this is probably that when a vertex is added to an edge or a face, symmetry is lost unless similar vertices are added to the other edges or faces. This results in many additional simplices. In practice, additional vertices are added to the body first, then to faces, then to edges.

(i)	(x_i, y_i, z_i)	(σ_i)
1	$(3,3,3)$	0.6
2	$(4,4,3)$	0.6
3	$(5,3,3)$	0.6
4	$(6,4,3)$	0.6
5	$(7,3,3)$	0.6
6	$(8,4,3)$	0.6
7	$(9,3,3)$	0.6
8	$(8,8,5)$	2.4
9	$(9.5,3,9)$	1.5

$$f(x,y,z) = \sum_{i=1}^{9} e^{-\frac{d_i^2}{2\sigma_i^2}}$$

$$d_i^2 = (x-x_i)^2+(y-y_i)^2+(z-z_i)^2$$

Figure 2: The Test Dataset

This gives rise to the following taxonomy of subdivisions: we list the principal subdivisions that have been suggested to date:

1. Subdivisions with no additional vertices:
   (a) Minimal (5-fold) - Sec. 4.2
   (b) Freudenthal (6-fold) - Sec. 4.3
2. Subdivisions with body vertex added:
   (a) Body Centered Cubic (BCC) - Sec. 4.4
   (b) Face-Divided (12-fold) - Sec. 4.5
3. Subdivisions with body and face vertices added:
   (a) Face-Centered (24-fold) - Sec. 4.6
   (b) 14-, 16-, 18-, 20- and 22-fold: see Sec. 4.6
4. Subdivisions with body, face and edge vertices added:
   (a) Edge-Divided (48-fold) - Sec. 4.7

4.2 The 5-fold Minimal Subdivision

The minimal possible subdivision (see Fig. 6(a)) is perhaps the most popular simplicial subdivision of the cube [1, 4, 5, 14, 17, 19, 20, 22]. It is obtained by cutting off four vertices of the same parity from the cube. Each of these vertices is removed by cutting the cube along the plane defined by the vertex' neighbours in the cube. In this way, each cut reduces the number of vertices remaining by 1. Four such cuts reduce the remaining set to 4 vertices, which define a tetrahedron in the center of the cube. This subdivision is minimal, because any given tetrahedron must share at least one face with another tetrahedron in the cube. This face must have three vertices in common with the other tetrahedron, so we can remove at most one vertex from the cube for each tetrahedron cut from the cube. With 8 vertices, it takes at least 4 cuts to reduce the number of vertices to 4, so 5 tetrahedra must be the minimal subdivision. Also note that, when a vertex is removed, its neighbours each gain two edges and lose one, becoming degree four. These vertices cannot then

Subdivision	Contained	Symmetric	Parsimonious	Minimal	Implicit	Continuous	Correct
Minimal (5)	yes	no	yes	yes	possible	possible	no
Freudenthal (6)	yes	no	yes	almost	yes	possible	no
BCC (12 average)	no	yes	no	no	yes	yes	no
Face-Divided (12)	yes	no	no	no	possible	possible	no
Face-Centered (24)	yes	yes	no	no	yes	yes	no
Edge-Divided (48)	yes	yes	no	no	yes	yes	no

Figure 1: Table of Subdivisions. "Possible" implies a desideratum that can be satisfied by a suitable choice of added vertices or edges.

be "cut off" in a single tetrahedron, forcing us to remove vertices by parity. As a result, there are only two ways of reducing a cube to five tetrahedra.

Because the four vertices cut off are treated differently from the four remaining, this subdivision is not symmetric. Worse, the central tetrahedron is a different shape from the other four. Finally, observe that opposite faces of the cube are divided by diagonals in different directions. As noted in [11, 17, 20, 22], the two possible 5-fold subdivisions must alternate throughout the grid for $F(p)$ to be continuous. This is possible for regular grids, for which $F(p)$ will depend on the parity of the cell. For a curvilinear grid, it is possible to have (for example) a torus composed of an odd number of hexahedral cells, in which case the alternation is not possible, and continuity cannot be guaranteed.

For Fig. 3(d), the same subdivision was used in every cell: the discontinuity results in cracks in the isosurface. Fig. 3(e) uses the first case for odd-parity cells, and the second case for even-parity cells. Fig. 3(f) uses the first case for even-parity, and the second case for odd-parity cells. These parity rules lead to different results. The peaks in the zigzag are connected for even parity, and disconnected for odd parity. This is because the face diagonal is added between the peaks in Fig. 3(e), connecting them. In Fig. 3(f), a face diagonal between two low-valued vertices separates the peaks, disconnecting them.

However, this subdivision is contained, parsimonious, minimal, implicit, and continuous (when parity is used).

Finally, note that the small surfaces are blocky, and even the larger surfaces are not smooth.

4.3 The Freudenthal (6-fold) Subdivision

This subdivision is actually older than the minimal subdivision in popularity, having been introduced for simplicial pivoting [10, 12, 2], and has been used extensively for rendering [1, 17, 20]. It is obtained by picking a major diagonal of the cell. The vertices on this diagonal are connected to all the other vertices, either by existing edges of the cube, or by diagonals across faces of the cube. This results in 6 tetrahedra packed around the major diagonal (see Fig. 6(b)). Note that, although the tetrahedra are all the same size and shape, 3 of them are mirror images of the others (i.e. the tetrahedra are not isomorphic without reflection). Since there are four major diagonals in the cell, there are four different Freudenthal subdivisions, each of which has a different directional bias.

This subdivision is contained, minimal, implicit, and only slightly less parsimonious than the minimal 5-fold subdivision. Note that the face diagonals that are added are the projections of the major diagonal onto the faces of the cube. Provided that these face diagonals are consistent between adjacent cubes, this subdivision will be continuous: this is achieved most simply by using the same diagonal in each cube. However, it is possible to achieve this in other ways: the 48-fold subdivision in Sec. 4.7 is a good example.

Due to the major diagonal, this subdivision is not symmetric. If we look at Fig. 3(g), we see a distinct axial bias in the direction of the major diagonal. If we choose a different major diagonal, as in Fig. 3(h), we get a different result. Also note the grooves visible on the silhouette of the larger surfaces. Clearly, neither symmetry nor correctness is satisfied.

It is possible to subdivide a cube into six tetrahedra in other ways [1]. It is not possible to do so without introducing a major diagonal, in which case similar artifacts to those in Figs. 3(g) and 3(h) are to be expected. In addition, these subdivisions are less symmetric than the Freudenthal subdivision, which at least has the advantage that all the tetrahedra are isomorphic to each other by reflection or rotation. Maintaining consistency between cells also becomes dependent on tracking the parity of the cell. Since these subdivisions fail more of the desiderata that the Freudenthal subdivision, we will not consider them further.

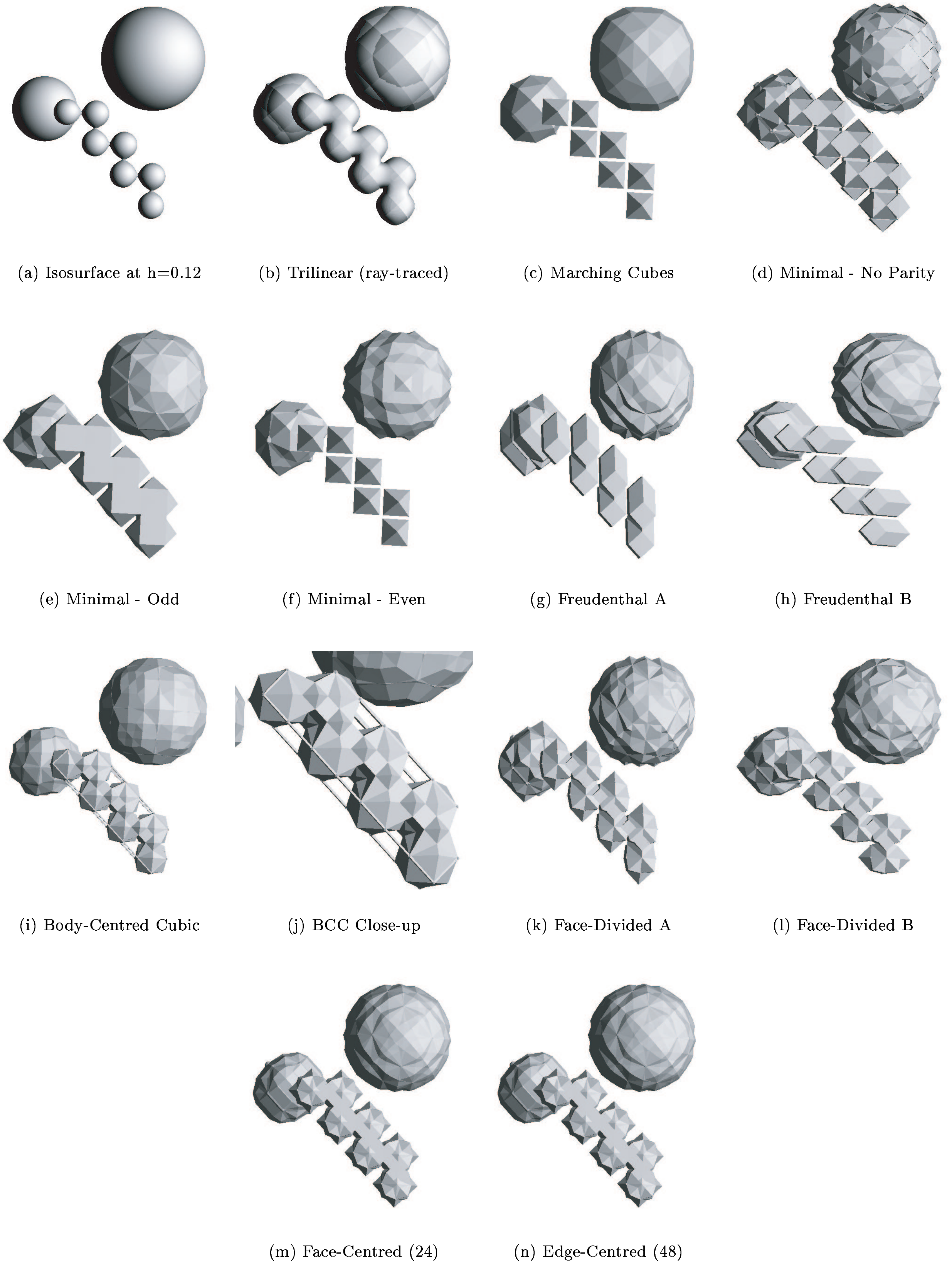
4.4 The Body Centered Cubic Subdivision

This subdivision,suggested to us by Herbert Edelsbrunner, is based on a body-centered cubic lattice (BCC mesh). An additional vertex is added to each cell. These additional vertices are connected to the new vertices in all adjacent cells, and to the vertices of the cell to which they belong, generating 24 tetrahedra. Since each tetrahedron is shared with an adjacent cell, an average of 12 tetrahedra per cell are generated, at the cost of a single additional vertex per cell. In Fig. 6(e), we illustrate this subdivision: unlike the other subdivisions, it is not possible to show a single cube, divided into tetrahedra. Instead we show all 24 tetrahedra that intersect a given cube.

Although the BCC lattice is highly symmetrical, a distinctly nasty artifact appears: "girders" form between adjacent peaks. To illustrate how this occurs, we have constructed a 2-D simplicial subdivision of a square cell that we call the "pseudo-BCC" subdivision: see Fig. 4(b). This subdivision is obtained by adding a vertex at the center of each cell. The central vertex is then connected to all four vertices of that square cell, and all 4 neighbouring central vertices. The edges between vertices of each square cell are suppressed. As with the 3-D BCC subdivision, girders form when we use this subdivision to interpolate function values.

To show how the girders are formed, consider the zero-valued vertices in the second and third rows of Fig. 4(a). Intuitively, we expect these vertices to be connected: that is, we expect that no contour exists that separates them. The bilinear interpolation function satisfies this intuitive expectation: a sample contour at the isovalue 0.20 is shown.

For the pseudo-BCC subdivision, the contour at isovalue 0.20 is shown in Fig. 4(c). Note that the lower row of cell



(a) Isosurface at h=0.12

(b) Trilinear (ray-traced)

(c) Marching Cubes

(d) Minimal - No Parity

(e) Minimal - Odd

(f) Minimal - Even

(g) Freudenthal A

(h) Freudenthal B

(i) Body-Centred Cubic

(j) BCC Close-up

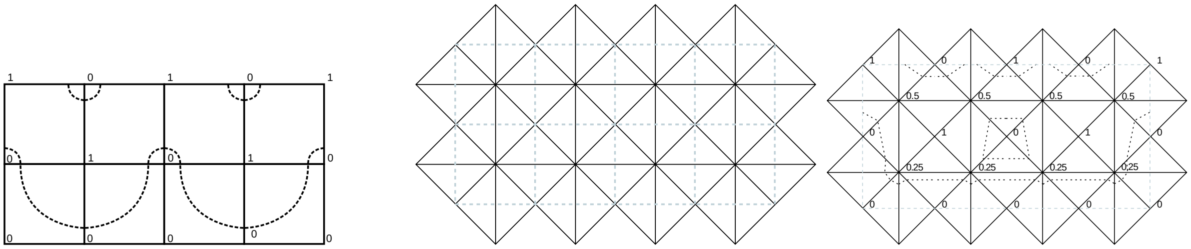
(k) Face-Divided A

(l) Face-Divided B

(m) Face-Centred (24)

(n) Edge-Centred (48)

Figure 3: The Artifacts (at $f = 0.12$)



(a) 0.20 Contour (Bilinear)

(b) The "pseudo-BCC" Subdivision)

(c) 0.20 Contour (BCC)

Figure 4: Simplified "Girders" in 2-D

centers all have the value 0.25, so they separate the zeros in the second and third rows of vertices: a sample contour at 0.20 is again shown. This contour "encloses" the edge between the cell centers. It is not difficult to see that the two parallel girders shown in Fig. 3(j) are caused by the contour enclosing the edge between cell centers in the same way.

The girder effect occurs principally because the pseudo-BCC subdivision is not contained: the interpolation function makes reference to values in adjacent cells. As a result, neither the pseudo-BCC subdivision in 2-D nor the BCC subdivision in 3-D is topologically correct. So, although the BCC subdivision is strongly symmetric, implicit and continuous, it is not parsimonious (it adds extra vertices), minimal, contained, or correct.

4.5 Face-Divided 12-fold Subdivision

A 12-tetrahedra subdivision can also be constructed from face-centered square pyramids [1, 4]. Each face of the cube is joined to the center of the cube, resulting in 6 square pyramids. Each of these is then divided into two tetrahedra by adding an arbitrary diagonal across the face (see Figs. 6(c) and 6(d)). Note that these tetrahedra are asymmetrical: not all vertices are treated equally. Since each face can be divided in two ways, there are 64 possible configurations, but if we wish to ensure continuity between cubes, we are constrained in our choices.

In practice, it is simplest to use the same subdivision in each cube, in which case opposing faces must have matching diagonals, and only two unique cases remain: in the first (Fig. 6(c)), a main diagonal is picked. As with the Freudenthal subdivision (Sec. 4.3), the vertices on this diagonal are each connected to all other vertices of the cube. As a result, this subdivision can be obtained by dividing each cell in the Freudenthal subdivision into two new cells, and the visible artifacts in Fig. 3(k) are similar to that case. Alternately, two vertices at opposite ends of a diagonal are cut off by dividing the adjacent faces (see Fig. 6(d)): again, the artifacts in Fig. 3(l) are similar to the Freudenthal subdivision.

Although contained, this subdivision is not symmetric, parsimonious, minimal, or correct. As with the 5- and 6- fold subdivisions, care is required to ensure that the subdivision is implicit and continuous.

4.6 Face-Centered 24-fold Subdivision

The face-divided subdivision can be further divided by adding vertices in the center of faces of the cube [1]. Adding only some of the face-centers gives subdivisions with 14, 16, 18, 20, 22 tetrahedra, respectively. Adding all the face-centers gives 24 tetrahedra per cube [1, 2, 6, 7, 11, 25, 27]. This subdivision (see Fig. 6(f)) can also be obtained by taking each of the 24 simplices in the BCC subdivision (Fig. 6(e)), and dividing it into two along the plane of the cube's face.

This subdivision is contained, symmetric, implicit, and continuous. It is not parsimonious, because it requires an average of 4 interpolated data points per voxel. Nor is it minimal, because it has nearly five times as many simplices as the minimal subdivision.

In practice, the principal visual artifact for this subdivision (see Fig. 3(m)), is that some spikiness is visible on finer details.

4.7 48-fold Subdivision

The 24-fold subdivision can be further subdivided by adding vertices on the cube's edges to obtain a 48-fold subdivision [12, 27] (see Fig. 6(g)). Interestingly, this subdivision can also be obtained by dividing the original cube into 8 sub-cubes, then applying the 6-fold subdivision to the sub-cubes, choosing the main diagonals that coincide with the main diagonals of the original cube.

This subdivision is contained, symmetric, implicit, and continuous, but neither parsimonious nor minimal, as it requires 8 times the original number of vertices, and nearly 10 times the minimal number of simplices. Again, in practice, the visual artifacts are relatively minor.

4.8 Numerical Results

In order to compare the different subdivision schemes quantitatively, we performed several computational experiments on the test function, defined in Fig. 2.

First, we took this data set in a 13 × 13 × 13 lattice, and evaluated the maximum absolute error and maximum relative error within each lattice cube using MATLAB's constrained minimization function `fmincon`. We did this for each type of cubical subdivision, plus trilinear interpolation. For the 5-fold and 6-fold, which are non-symmetric, we used subdivisions at two orientations. For the 12-fold, BCC, and 24-fold, which need additional data points and are thus not parsimonious, we tested subdivisions that interpolated this new data from the lattice vertices, and those that obtained the true data by function evaluation. We also tried various sizes for the edges of the lattice cubes, using powers of 2 from 2^1 down to 2^{-5}. The sizes 1, 1/2, and 1/4 are the most important.

After studying these results, we decided to focus on absolute error, since cubes with large errors produce the most noticeable artifacts. We therefore selected a number of cubes with absolute errors greater than 10^{-2}, and ran experiments to compute maxima for different subdivisions and different scales.

Finally, from these experiments, we chose 40 points that had realized the maximum errors in their respective cubes. We tracked these points using different subdivision schemes and different lattice cube sizes. We plotted the absolute error of the results on logarithmic scales, using logarithms base 2.

We show some sample graphs in Fig. 7. In these there are some common features:

1. A smaller lattice reduces the error roughly in proportion to the length of the cube side. (Unfortunately, it blows up the size proportionally to $1/side^3$.)

2. For test points near a maximum, we can observe a large increase in accuracy with the first steps of refinement, then a tapering off toward the end.

3. The maximum error for trilinear interpolation is similar to the simpler, simplex-based interpolation schemes.

4. It was not uncommon for the asymmetric subdivisions (5-fold, and 6-fold) to have both the best and the worst errors.

5. Using the real midpoints sampled from the function generally reduces the error of the 12-fold, 24-fold, and BCC subdivisions. The reduction is comparable to that achieved by halving the cube size, when the midpoints are interpolated.

5 Sampling Issues

Usually, no *a priori* information is available for the (continuous) underlying function that is sampled. It is normally assumed that the function is band-limited. Since the function is given on a regular rectangular grid, the aliased spectra in the frequncy domain are replicated on a regular grid as well. This means that the frequency support of the function is the rectangular region $[-\pi/T_x, \pi/T_x] \times [-\pi/T_y, \pi/T_y] \times [-\pi/T_z, \pi/T_z]$, where T_x, T_y, and T_z are the sampling distance in x, y, and z direction [9]. This frequency π/T is also known as the *Nyquist* limit.

In order to reconstruct a continuous function from these sampled values, we would have to multiply with a box function in the frequency domain. This is equivalent to convolution with the Sinc function ($sinc(x) = (\sin \pi x)/\pi x$)in the spatial domain. Hence the 3D Sinc function, of which we show the 2D equivalent in Fig. 5(a), is generally considered to be the ideal reconstruction kernel. [21]

However, Sinc interpolation is expensive to compute, since it is an IIR filter (infinite impulse response), and requires the entire sampled dataset to be processed to compute a single interpolated value. As a result, many approximations to the Sinc interpolation have been suggested [15, 21].

In practice, the standard interpolation filter in volume graphics is trilinear filtering. The cost to apply a $m \times m \times m$ filter at a single location is $O(m^3)$. For trilinear interpolation, $m = 2$, and the cost is acceptable. For higher-order (i.e. larger) filters, m increases, and the cost of computation rapidly becomes unacceptable. Trilinear filtering is also attractive from a topological point of view. Not only does it satisfy the intuitive expectation of connectedness that we describe in Sec. 4.4, but all maxima and minima occur at grid vertices, and a closed form exists for computing the remaining saddle points [16].

We note that our test dataset consists of features that are near the Nyquist sampling limit, so it is not surprising that all of the simplicial subdivisions in Sec. 4 perform poorly.

Some sense of why this happens can be obtained by looking at the Fourier transforms of the interpolation kernels corresponding to each of the subdivisions. However, since these interpolation kernels are themselves three-dimensional, we illustrate the problems by looking at two-dimensional kernels instead.

In two dimensions, we can obtain a simplicial subdivision by cutting each square cell along both diagonals, as in Fig. 5(e). Broadly speaking, this subdivision is akin to the 24-fold 3-D subdivision described in Sec. 4.6, since the center of the cell is connected to all the vertices in a symmetric fashion. In choosing this subdivision, we implicitly choose the barycentric interpolation function over the simplices: the corresponding interpolation filter is shown in Fig. 5(f). This filter is a poor match for the ideal filter, but makes a reasonable approximation of the bilinear filter. It does, however, have more pronounced sidelobes than the bilinear filter, and, in particular, diagonal sidelobes, as shown in Fig. 5(g). As an indication of how closely this filter approximates the bilinear filter, we show the difference between the two filters in Fig. 5(h): the maximum difference between the two is approximately 0.11.

In comparison, we show the pseudo-BCC subdivision in Fig. 5(i), which we last saw in Sec. 4.4. The corresponding interpolation filter shown in Fig. 5(j), the Fourier transform in Fig. 5(k), and the difference between this filter and the bilinear filter in Fig. 5(l). Note the strong diagonal ridges in the Fourier transform and the sidelobes in the cardinal directions. In particular, the filter is non-zero at the center of the aliased spectra along the major axes (at $\pm 4\pi$). This is an undesirable feature, which will cause strong aliasing artifacts. Furthermore, the maximum difference from the bilinear filter is 0.30.

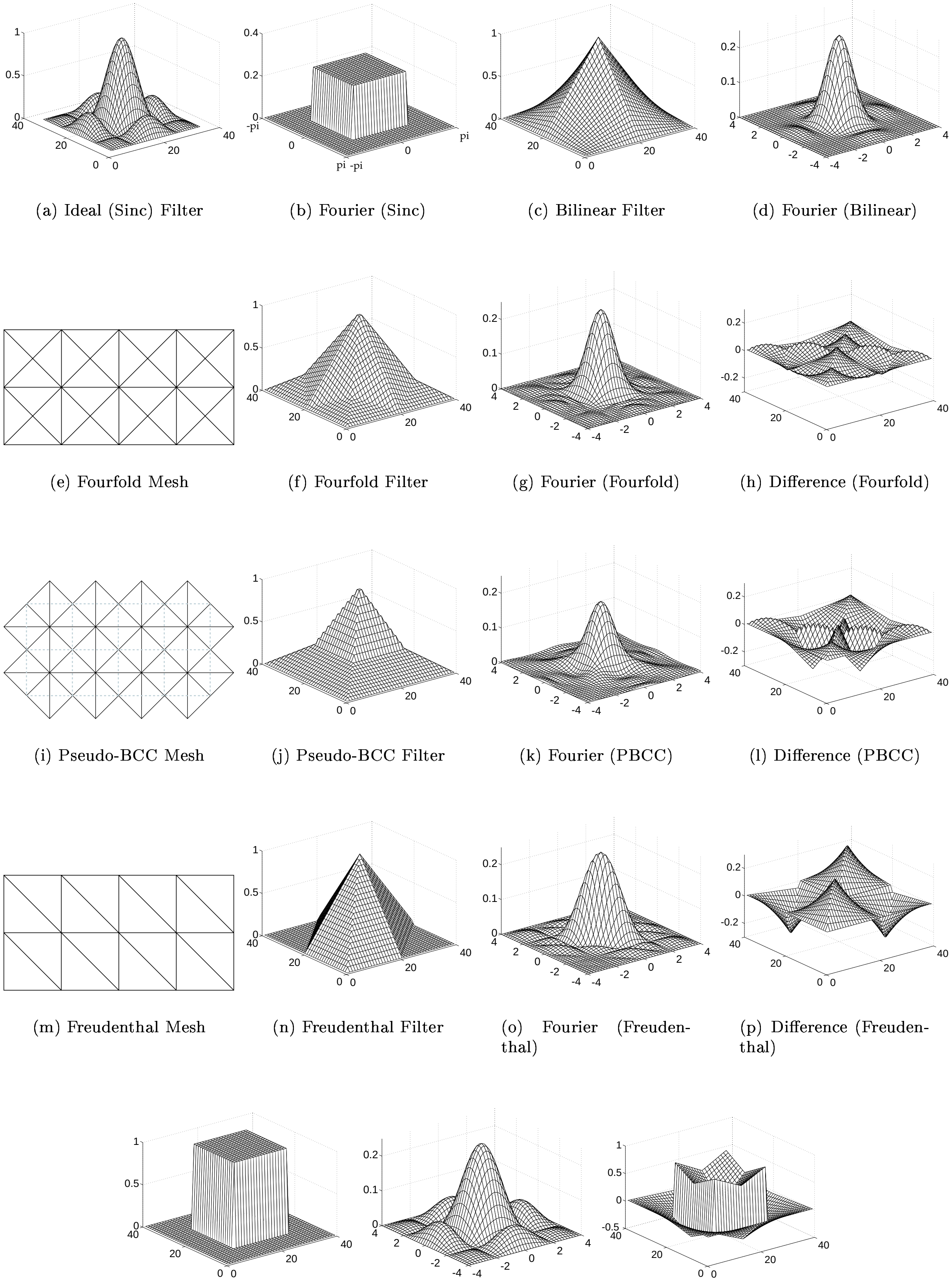
We next consider the Freudenthal subdivision in Fig. 5(m), which corresponds to the 3-DFreudenthal subdivision described in Sec. 4.3. Both the filter in Fig. 5(n) and the Fourier transform in Fig. 5(o) demonstrate the strong axial bias of this filter, with multiple sidelobes extending along the diagonal. This bias results in a maximum difference from the bilinear filter of 0.25.

Since these cases correspond roughly to the major subdivisions we described in Sec. 4, similar sidelobes in 3-D Fourier space can be expected, reinforcing the concerns raised about these subdivision schemes.

Finally, for comparison, we show the nearest neighbour filter in Fig. 5(q), and its Fourier transform in Fig. 5(r). We show the difference between the nearest neighbour and bilinear filters in Fig. 5(s). The maximum difference is relatively large at 0.7, so it was necessary to reduce the vertical scale.

6 Conclusion

We have reviewed the various simplicial subdivisions proposed for 3-D applications, and examined the geometric ar-



(a) Ideal (Sinc) Filter

(b) Fourier (Sinc)

(c) Bilinear Filter

(d) Fourier (Bilinear)

(e) Fourfold Mesh

(f) Fourfold Filter

(g) Fourier (Fourfold)

(h) Difference (Fourfold)

(i) Pseudo-BCC Mesh

(j) Pseudo-BCC Filter

(k) Fourier (PBCC)

(l) Difference (PBCC)

(m) Freudenthal Mesh

(n) Freudenthal Filter

(o) Fourier (Freudenthal)

(p) Difference (Freudenthal)

(q) Nearest Neighbour

(r) Fourier (NN)

(s) Difference(NN)

Figure 5: Filter Kernels for 2-D Simplicial Subdivisions

tifacts that result from each (see Table 1). We have also considered the sampling artifacts in the Fourier domain for each of these subdivisions. None of the subdivisions is entirely satisfactory: some care is called for when selecting one for use. Unsurprisingly, there appears to be a trade-off between size (as measured by parsimony and minimality) and accuracy (as measured by symmetry, consistence and correctness).

If there is good reason to assume that the sampling is significantly better than the Nyquist limit, we expect that these artifacts will be relatively minor: any of the subdivisions may then be used, for example, the 5-fold or 6-fold subdivisions.

If, however, the features of the dataset are unknown, or if the sampling rate is barely sufficient, we suggest using the 24-fold subdivision. This requires interpolation (or sampling) of 4 additional vertices per cube, on average, increasing the number of vertices to be processed by a factor of 5, and the number of cells by a factor of 24 (although the cells are simpler).

There appears to be no good reason to use a 48-fold subdivision, except for hierarchical applications such as the one in Zhou, Chen & Kaufman [27]. Instead of interpolating additional vertices for this subdivision, additional samples should be taken (which is, in effect, what Zhou, Chen & Kaufman do). But this raises an interesting point: if we are going to increase the input size by a factor of 5 or more, how much do we gain from the simplicity of the tetrahedron's interpolation function?

We would argue that simplicial subdivisions should only be used where they are necessary for the correct operation of an algorithm, such as [7, 11, 14, 19, 22, 23]. In almost all other cases, the extended look-up table of Cignoni et al.[8] will generate fewer triangles, and require fewer interpolated points.

Our analysis of tetrahedral subdivisions took place in the context of regularly sampled grids. Although most of the analysis also applies to curvilinear grids, or irregular grids using hexahedra, additional work is possible in these realms: for example, although simpler, it would be useful to extend the analysis to prisms in irregular grids. It might also be profitable to construct a more rigorous taxonomy of all possible tetrahedral subdivisions, similar to that in [1].

7 Acknowledgments

This work has been supported by NSERC through a postgraduate fellowship and a research grant, by IRIS through a research grant, and by NSF grants 9988742 and 0076984. Thanks are also due to the anonymous reviewers for a number of suggestions that we have adopted.

References

[1] G. Albertelli and R. A. Crawfis. Efficient Subdivision of Finite-Element Datasets into Consistent Tetrahedra. *Vis 97*, 213–219.

[2] E. Allgower and S. Gnutzmann. Simplicial Pivoting for Mesh Generation of Implicitly Defined Surfaces. *Computer Aided Geometric Design*, 8:305–325, 1991.

[3] E. Artzy, G. Frieder, and G. T. Herman. The Theory, Design, Implementation and Evaluation of a Three-Dimensional Surface Detection Algorithm. *Computer Graphics and Image Processing*, 15:1–24, 1981.

[4] J. Bloomenthal. Polygonization of implicit surfaces. *Computer Aided Geometric Design*, pages 341–355, 1988.

[5] J. Bloomenthal. *An Implicit Surface Polygonizer*, *Graphics Gems IV*, 324–349. Academic Press, 1994.

[6] H. Carr. Efficient Generation of 3D Contour Trees. M.Sc. thesis, Univ. of British Columbia, Vancouver, BC, Canada, 2000.

[7] H. Carr, J. Snoeyink, and U. Axen. Computing Contour Trees in All Dimensions. *ACM-SIAM Symp. on Discrete Algorithms (SODA) 2000*, 918–926.

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

[9] D. Dudgeon and R. Mersereau. *Multidimensional Digital Signal Processing.* Prentice-Hall Inc., Englewood-Cliffs, NJ, 1984.

[10] H. Freudenthal. Simplizialzerlegungen von Beschränkter Flachheit. *Annals of Mathematics*, 43(3):580–582, 1942.

[11] M. P. Garrity. Raytracing Irregular Volume Data. *Computer Graphics*, 24(5):35–40, 1990.

[12] S. Lefschetz. *Introduction to Topology*, 140-141. Princeton University Press, Princeton, NJ, 1949.

[13] W. E. Lorenson and H. E. Cline. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. *Computer Graphics*, 21(4):163–169, 1987.

[14] N. L. Max, P. Hanrahan, and R. A. Crawfis. Area and Volume Coherence for Efficient Visualization of 3D Scalar Functions. *Computer Graphics*, 24(5):27–33, 1990.

[15] T. Möller, K. Mueller, Y. Kurzion, R. Machiraju, and R. Yagel. Design of accurate and smooth filters for function and derivative reconstruction. In *VolVis 98*, 143–151.

[16] B. Natarajan. On generating topologically consistent isosurfaces from uniform samples. *Visual Computer*, 11:52–62, 1994.

[17] G. M. Nielson and R. Franke. Computing the Separating Surface for Segmented Data. In *Vis 97*, 229–233.

[18] G. M. Nielson and B. Hamann. The Asymptotic Decider: Resolving the Ambiguity in Marching Cubes. *Vis 91*, 83–91.

[19] G. M. Nielson and J. Sung. Interval Volume Tetrahedrization. *Vis 97*, 221–228.

[20] P. Ning and J. Bloomenthal. An Evaluation of Implicit Surface Tilers. *IEEE Comp. Graphics and Appl.*, 13:33–41, 1993.

[21] A. Oppenheim and R. Schafer. *Discrete-Time Signal Processing.* Prentice-Hall Inc., Englewood-Cliffs, NJ, 1989.

[22] P. Shirley and A. Tuchman. A Polygonal Approximation to Direct Scalar Volume Rendering. *Computer Graphics*, 24(5):63–70, 1990.

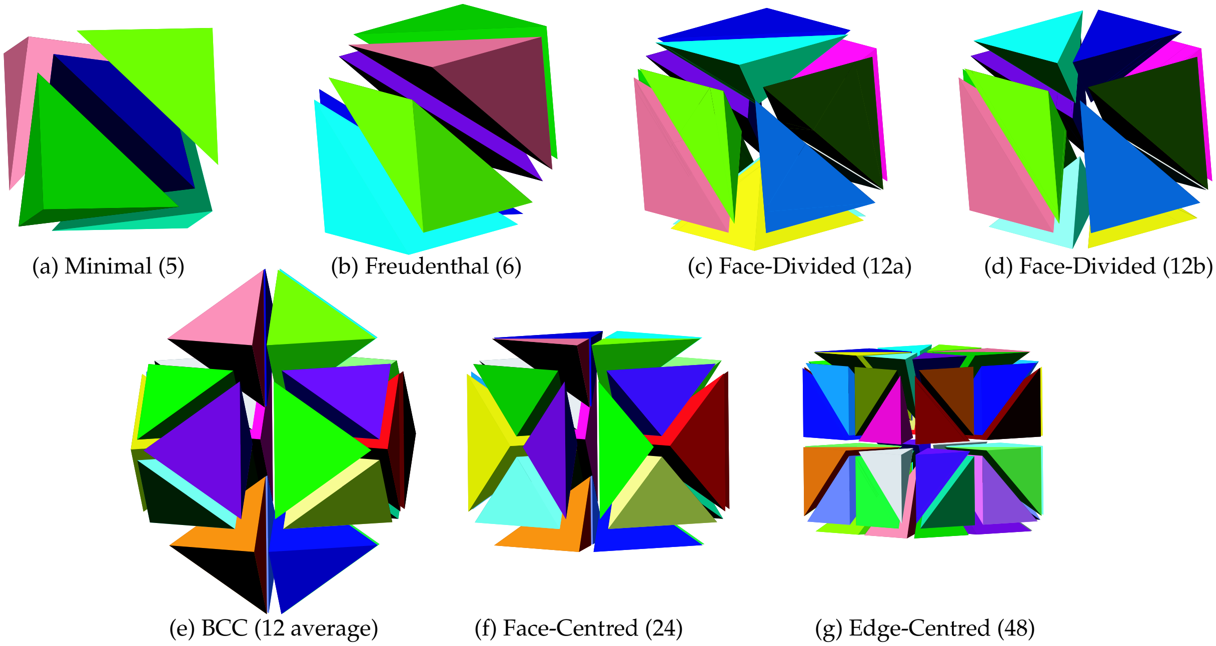
[23] C. Stein, B. Becker, and N. L. Max. Sorting and Hardware Assisted Rendering for Volume Visualization. *Vis 94*, 83–89, 1994.

[24] M. van Kreveld, R. van Oostrum, C. L. Bajaj, V. Pascucci, and D. R. Schikore. Contour Trees and Small Seed Sets for Isosurface Traversal. *13th ACM Symp. on Comp. Geom.*, 212–220, 1997.

[25] C. Weigle and D. C. Banks. Complex-Valued Contour Meshing. *Vis 96*, 173–181.

[26] J. Wilhelms and A. van Gelder. Topological Considerations in Isosurface Generation. *Computer Graphics*, 24(5):79–86, 1990.

[27] Y. Zhou, B. Chen, and A. Kaufman. Multiresolution Tetrahedral Framework for Visualizing Regular Volume Data. *Vis 97*, 135–142.



(a) Minimal (5)

(b) Freudenthal (6)

(c) Face-Divided (12a)

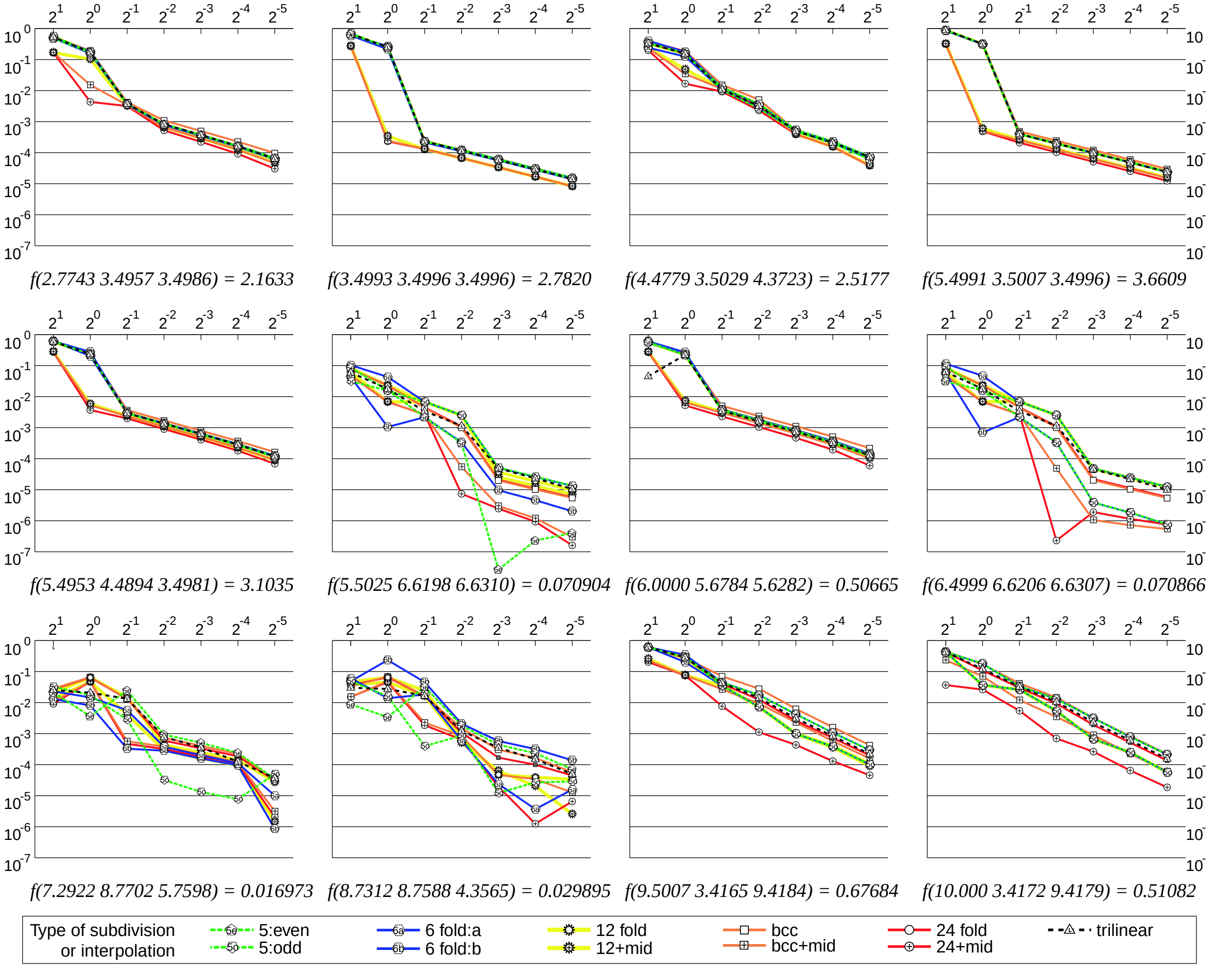
(d) Face-Divided (12b)

(e) BCC (12 average)

(f) Face-Centred (24)

(g) Edge-Centred (48)

Figure 6: The Subdivisions



f(2.7743 3.4957 3.4986) = 2.1633

f(3.4993 3.4996 3.4996) = 2.7820

f(4.4779 3.5029 4.3723) = 2.5177

f(5.4991 3.5007 3.4996) = 3.6609

f(5.4953 4.4894 3.4981) = 3.1035

f(5.5025 6.6198 6.6310) = 0.070904

f(6.0000 5.6784 5.6282) = 0.50665

f(6.4999 6.6206 6.6307) = 0.070866

f(7.2922 8.7702 5.7598) = 0.016973

f(8.7312 8.7588 4.3565) = 0.029895

f(9.5007 3.4165 9.4184) = 0.67684

f(10.000 3.4172 9.4179) = 0.51082

Figure 7: Absolute numerical error plotted against cube size at selected points for various subdivisions