Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2017) 25: 2501 – 2512
© TÜBİTAK
doi:10.3906/elk-1604-328

A marching algorithm for isosurface extraction from face-centered cubic lattices

Yusuf SAHİLLİOĞLU*
Department of Computer Engineering, Middle East Technical University, Ankara, Turkey

Received: 25.04.2016 • **Accepted/Published Online:** 20.09.2016 • **Final Version:** 29.05.2017

Abstract: This work provides a novel method that extracts isosurfaces from face-centered cubic (FCC) lattices. It has been theoretically shown that sampling volumetric data on an FCC lattice tiled with rhombic dodecahedra is more efficient than sampling them on a Cartesian lattice tiled with cubes, in that the FCC lattice can represent the same data set as a Cartesian lattice with the same accuracy, yet with approximately 23% fewer samples. This fact, coupled with the good properties of rhombic dodecahedra, encouraged us to develop this related isosurface extraction technique. Thanks to the sparser sampling required by the FCC lattices, the de facto standard isosurface extraction algorithm, namely marching cubes, is accelerated significantly, as demonstrated. This reduced sampling rate also leads to a decrement in the number of triangles of the extracted models when compared to the marching cubes result. Finally, the topological consistency problem of the original marching cubes algorithm is also resolved. We show the potential of our algorithm with an indirect volume-rendering application.

Key words: Indirect volume-rendering, isosurface extraction, non-Cartesian lattice, face-centered cubic lattice, marching

1. Introduction

Implicit representation of data, either in the form of mathematical functions, e.g., a sphere, or thresholded real-world data sets, e.g., a medical image, is converted to a polygon-based representation with the isosurface extraction operation. Isosurfaces that represent points of equal values (e.g., distance, intensity) within a volume of space can be utilized in many applications, such as shape deformation, constructive solid geometry, and indirect volume rendering, the latter being our interest in this paper.

The visualization of volumetric data sets is typically based on the representation of the underlying point lattice. Although marching cubes [1] has provided a simple yet successful solution to this rendering problem by using a Cartesian lattice, it has been shown that results can be enhanced drastically in terms of quality, efficiency, and robustness by using non-Cartesian lattices such as the body-centered cubic (BCC) and face-centered cubic (FCC) lattices. Next-generation volumetric data sampling schemes [2] have drawn attention to the usage of such lattices, which in a sense rendered the Cartesian cubic lattice used by the marching cubes algorithm obsolete.

The optimality of the BCC lattice was proved in [2] by reducing the problem of sampling to a well-known mathematical problem of 3D sphere packing. The BCC has almost one-third (29.3%) lower sampling density compared to the Cartesian lattice while maintaining the quality of the extracted isosurface. The FCC lattice is

*Correspondence: ys@ceng.metu.edu.tr

the second best choice for this purpose, as it achieves the same accuracy as a Cartesian lattice with 23% fewer samples [3].

In this paper, we will introduce an isosurface extraction technique to render volumetric data sampled on the FCC lattices. The main reason for preferring the FCC to the BCC is that the FCC achieves a similar efficiency as the BCC while tiling the space with rhombic dodecahedra, which are simpler to process than the truncated octahedra used in BCC tiling. We discuss the previous work in Section 3 and address the problem in Section 4. We present our results in Section 5, followed by a conclusion in Section 6.

2. Contributions

Our contributions can be summarized as follows:

- Tiling the space with rhombic dodecahedra;
- Finding the indexing of the decomposition of parallelepipeds into six tetrahedra that leads to manifold extractions;
- Developing a comparative framework with the well-known Cartesian lattices and marching cubes algorithm to clearly (visually and quantitatively) demonstrate the advantage of our fast non-Cartesian method.

2.1. Related work

A preliminary isosurface extraction technique, called contour tracing, connects the adjacent representative contours of the object with triangles [4]. This idea, which dates back to 1975, is now mostly ruled out due to several problems, including the existence of many contours in each slice and difficulty in tracing slices that exhibit high variances in between.

A better extraction idea is that presented in [5], which represents the volume by subdividing it into small cubes, called cuberilles. These cuberilles are centered on a single sample and are rendered only if the isovalue in their center is above the given threshold, which in turn renders only the boundary cubes that separate the inside cubes from the outside ones. This idea of cube-rendering causes an undesired jaggy structure in the output, as demonstrated in Figure 1a.

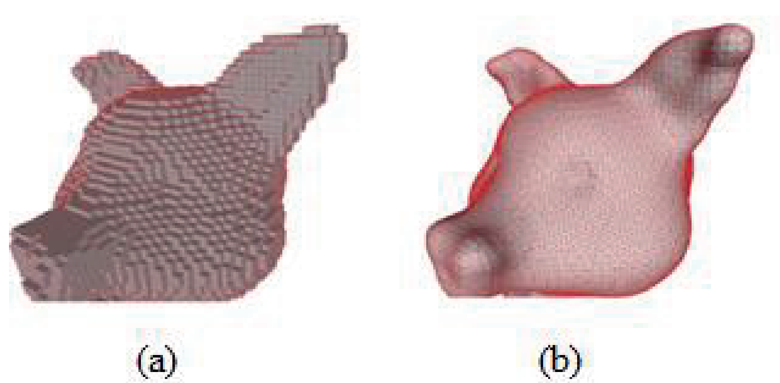

Figure 1. Isosurfaces extracted by (a) cuberille method and (b) marching cubes method.

An even better approximation is the marching cubes algorithm [1], which overcomes the staircase problem of cuberilles by using linear interpolation along the edges of the cubes to be marched, as shown in Figure 1b. This edge interpolation is possible due to the placement of the samples to the cube vertices, which effectively forms the Cartesian cubic lattice. This algorithm, in fact, is so promising that it is regarded as the de facto standard geometry-based isosurface extraction technique, although it has ambiguous pathological cases that cause topological inconsistencies.

An analogous algorithm to [1] is marching tetrahedra [6], which produces for each tetrahedron a facet approximation to the isosurface, instead of a cube. Although this choice clears the ambiguities of [1], it suffers from the excessive number of triangles produced. In this paper, we will make use of marching tetrahedra as an auxiliary procedure to our global isosurface extraction algorithm, hence not suffering from the topological inconsistencies inherent to the marching cubes algorithm.

There are also algorithms that utilize lattices other than the Cartesian cubic lattice. These are motivated by the analytical advantages of such non-Cartesian lattices [2]. There also exists sufficient evidence from nature that promotes the usage of these alternative non-Cartesian lattices. A honeycomb is arguably the most obvious example of a non-Cartesian lattice in nature [7]. Honey bees instinctively construct hexagonal cells that form the non-Cartesian hexagonal lattice representation of their honeycomb. An intuitive explanation of this choice is that the hexagon covers the largest surface area for its perimeter length when tiling the 2D plane [8]. Hence, bees use the minimal amount of wax to cover the space in their honeycomb construction. In other words, they effectively optimize the volume of the honey cells. This intuition extends well to 3D and explains geometrically why non-Cartesian lattices need an estimated 30% fewer samples (wax) than the Cartesian cubic lattice in order to represent (cover) the same volume.

Among non-Cartesian works, [9] and [10] developed marching algorithms for BCC lattices. Since the mesh that arises from BCC lattices involves a large number of cells, their work focused on reducing the number of cells by clustering tetrahedra into either octahedra or hexahedra. In [11] and [12], on the other hand, FCC lattices were used to leverage the sampling advantage of a non-Cartesian lattice. However, these works have the problem of topological ambiguity stemming from triangulating octahedral cells. The authors of [13] alleviated such topological problems by dividing each cell into a number of tetrahedra, an approach we also employ in our algorithm.

Applications dealing with the isosurfaces extracted from volumetric data arise mostly in the medicine and geometry-processing domains. For the former, one may visualize and further assess a medical condition using the intensity values that are specific to the organs [14–16]. For the latter, we see shape deformation [17,18] and reconstruction [19–22] applications running on signed distance fields, as well as sampling of functions [8,23] and Boolean operations in constructive solid geometry [24,25]. We also see various other fields, such as chemistry [26], crystallography [27], condensed matter physics [28], and solid state physics [29], utilizing isosurfaces defined on different grid structures in order to study the structure of atoms and molecules.

3. Algorithm

The block diagram of our overall isosurface extraction algorithm is given in Figure 2. The continuous scalar field f that implicitly represents the volumetric data of interest is discretized in the tiling rhombic dodecahedra cells of the FCC lattice. Our problem is to extract the isosurface, which is defined to be the zero set of

$$f : R^3 \rightarrow R, i.e., \{\mathbf{x} in R^3 | f(\mathbf{x}) = 0\}.$$

The FCC lattice enters the picture to fulfill the efficiency requirement, given that a significantly coarser sampling density than the Cartesian sampling density would provide the desired result, as depicted in Figure 3. Based

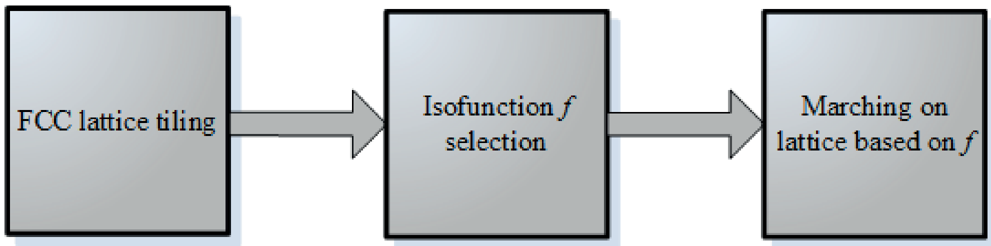

Figure 2. Overall isosurface extraction scheme.

on a similar argument, it is also clear that one can infer more information from the FCC lattice than from the Cartesian lattice when both lattices are equipped with the same number of points.

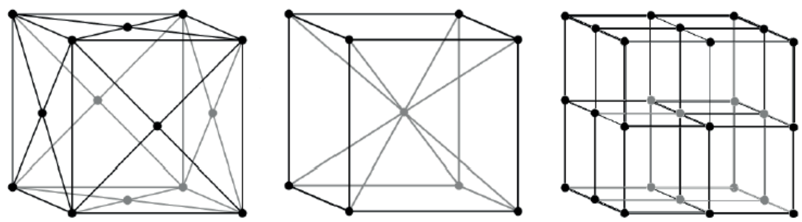

Figure 3. The nesting structure of the non-Cartesian FCC lattice (left), and the non-Cartesian BCC lattice (middle) for the original data given in the Cartesian lattice (right). Notice the reduced sampling densities of the non-Cartesian lattices. For the same volume, the FCC and BCC lattices have half and quarter sampling density, respectively, compared to the Cartesian lattice.

Once the FCC lattice is constructed, we apply a marching algorithm, as described in the following subsection. Marching, which is the idea of processing each tiling polyhedron one by one (hence the act of marching through the cells), combines simplicity with high speed, stemming from the usage of look-up tables.

3.1. FCC tiling and marching

The tiling polyhedron of the FCC lattice is a rhombic dodecahedron [30], which we obtain by placing six cubes on the faces of a seventh central cube and then joining the centers of the outer cubes with the vertices of the central cube, as illustrated in Figure 4. Values of f within rhombic dodecahedra are obtained with trilinear interpolation. Since the trilinear approximation of f is actually linear along rhombic dodecahedra edges, the isosurface corresponding to the isovalue of zero can be interpolated based on linear interpolation when fed into a marching algorithm.

For easier visualization, we provide in Figure 5 (left) the 2D FCC tiling obtained by a mapping from a rhombic dodecahedron (3D) to a hexagon (2D). We visualize in Figure 5 (right) that each rhombic dodecahedron cell in gap-free FCC tiling is decomposed into 4 parallelepipeds. Marching through those parallelepipeds would yield the extraction of the isosurface from the 3D FCC lattice, which is our ultimate aim.

Marching directly through parallelepipeds, however, can result in undesired t-junctions and holes due to the topological ambiguity problems inherent in the marching cubes algorithm. Recall that a parallelepiped is simply a hexahedron with six parallelogram faces, which, in turn, is homeomorphic to a cube that the marching cubes algorithm can process. To achieve robustness against such topology problems, we march through the obtained tetrahedra by further decomposing each parallelepiped, as depicted in Figure 6, which creates 6 tetrahedra per parallelepiped.

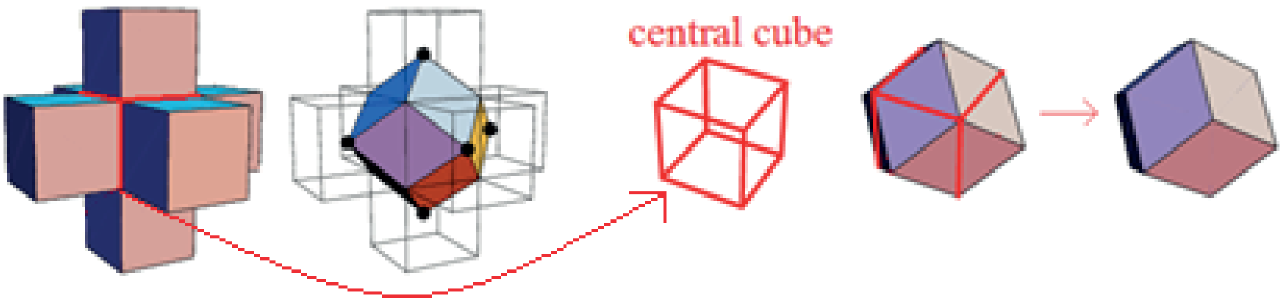

Figure 4. Creation of a rhombic dodecahedron from six outer and one central cubes [30].

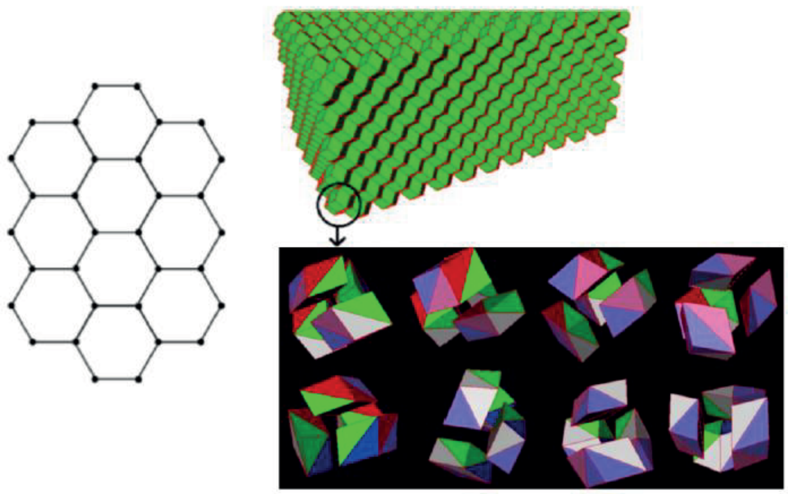

Figure 5. 2D FCC tiling with hexagonal cells (left) increased to 3D FCC tiling by using rhombic dodecahedron cells (right). Each rhombic dodecahedron in 3D tiling decomposes into 4 parallelepipeds (right-bottom).

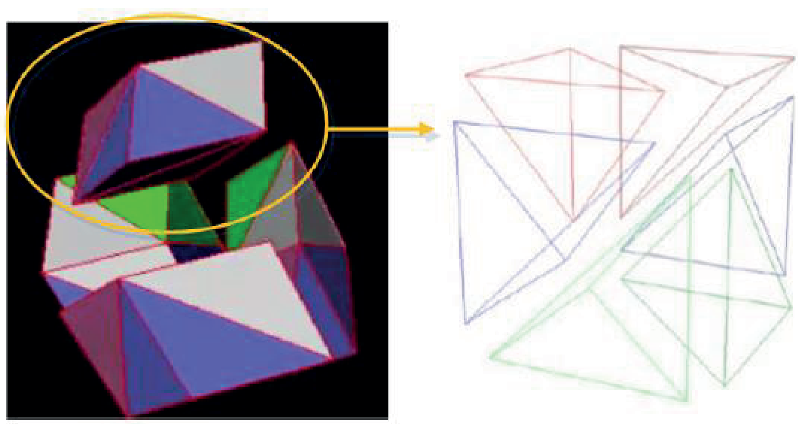

Figure 6. Decomposition of a parallelepiped into six tetrahedra used by our algorithm.

3.2. Tetrahedra alignment

Tetrahedra within a rhombic dodecahedron, shown in Figure 7 (left), as well as across any given two rhombic dodecahedra, shown in Figure 7 (right), must be aligned in a consistent way such that the corresponding edges of adjacent tetrahedra fall completely onto each other. A misalignment would create t-junctions and holes in the resulting isosurface. Decomposition, given in Figure 6 (right), achieves the desired alignment. Please see also the accompanying video for this alignment (http://user.ceng.metu.edu.tr/∼ys/vid.mp4).

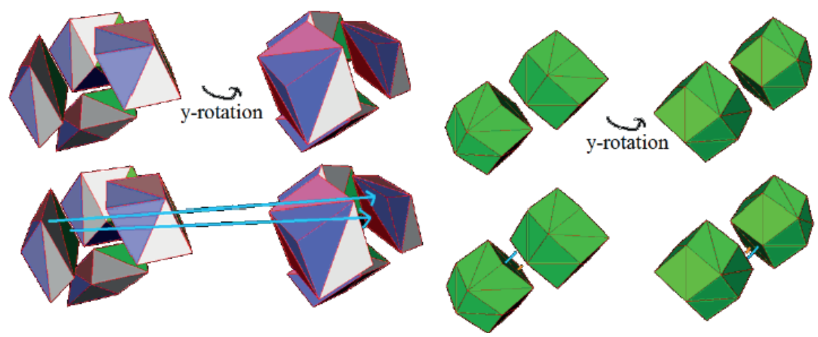

Figure 7. Alignment of tetrahedra within a rhombic dodecahedron (left) and across two dodecahedra (right). Corresponding edges of adjacent tetrahedra align well (some correspondences highlighted with cyan and orange lines).

4. Results and discussion

We have conducted several tests that compare the accuracy and efficiency of the isosurfaces extracted by our method and the original marching cubes algorithm. An isosurface in this section is the zero set of the following function $f\colon \mathrm{R}^3 \rightarrow \mathrm{R}$ that we adapt from [22]:

$$f(\mathbf{x}) = \min\{G[Proj_I^n(\mathbf{x})] - 0.5\},$$

where Proj is the projection of the point $\mathbf{x}$ to the nth binary silhouette image I^n (0 for outside, 1 for inside), and the interpolator function G is directly copied from [22]. Namely, function G, taking values in [0,1], is the bilinear interpolation of the subpixelic projection of (x', y') of the point $\mathbf{x}$. Thus, the isolevel function $f(\mathbf{x})$ takes values in [–0.5, 0.5], and the zero crossing of this function reveals the isosurface. The value $f(\mathbf{x})$ is provided by the image of the silhouette that is farthest away from the point, or, in other words, where G assumes its minimum value. We have used silhouette images of two real-world objects from 72 different views, each obtained by rotating the subject 5 degrees around the vertical axis, as shown in Figure 8.

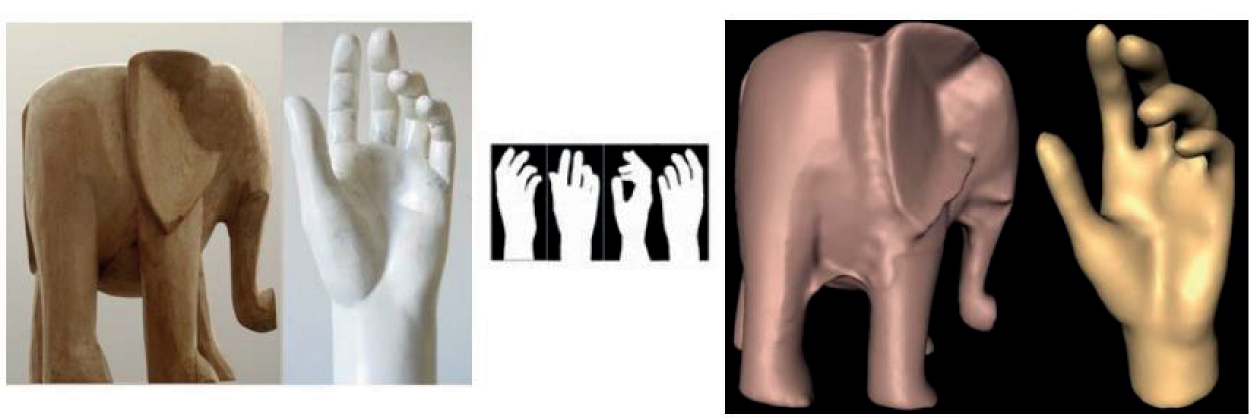

Figure 8. Left pair: Original images of the elephant and hand objects and some silhouette images from the hand (middle). Right pair: Corresponding ground-truth digital surfaces obtained from [22] for evaluation purposes.

We compare the visual quality of the output meshes for the Cartesian cubic lattice and the FCC lattice in Figures 9–12. Additionally, we quantify the mesh qualities in Tables 1 and 2 by measuring the compatibilities of the corresponding normals between the ground-truth surfaces, shown in Figure 8 (bottom), and the resulting extracted surfaces via the following function:

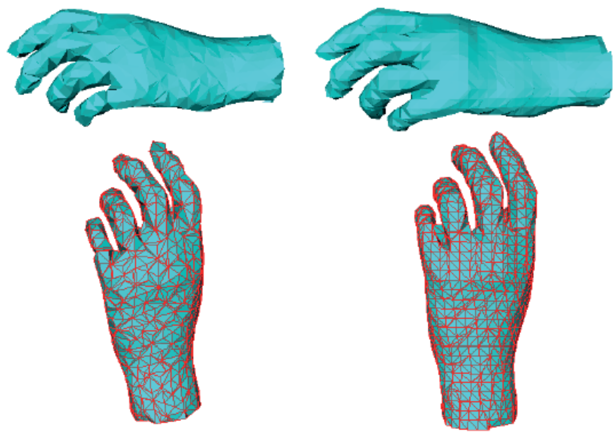

Figure 9. Isosurface extraction based on our method on the FCC lattice (left) and the marching cubes method on the Cartesian cubic lattice (right). Two different views are given in each column.

Table 1. Isosurface extraction information for the hand isosurface. MC stands for marching cubes. This table presents how the execution times (fourth column) change as we switch from the well-known Cartesian lattice (rows 2–3 and 5–6) to our FCC lattice (rows 1 and 4). Accuracy change is also reported in the sixth column. The first 3 rows are for the coarse lattice and the other 3 are for the denser lattice. The same layout applies to Table 2.

Lattice	Method	# Lattice points	Time (s)	# Triangles	Q
FCC	Ours	6912	0.09	2133	0.31
Cartesian	MC	9270	0.21	3249	0.33
Cartesian	MC	6656	0.08	2074	0.43
FCC	Ours	12,450	0.17	3847	0.26
Cartesian	MC	19,467	0.44	5872	0.27
Cartesian	MC	12,360	0.15	3713	0.30

Table 2. Isosurface extraction info for hand isosurface. MC stands for marching cubes.

Lattice	Method	# Lattice points	Time (s)	# Triangles	Q
FCC	Ours	5376	0.08	2596	0.32
Cartesian	MC	7344	0.19	3899	0.35
Cartesian	MC	5265	0.07	2570	0.39
FCC	Ours	10,210	0.16	4540	0.28
Cartesian	MC	15,422	0.43	6867	0.29
Cartesian	MC	10,107	0.15	4436	0.35

$$Q(\mathbf{n}, \mathbf{n}') = (\sum_{\mathbf{n}_i} q(\mathbf{n}_i \cdot \mathbf{n}'_j))/n,$$

where the unit normal set $\mathbf{n} = \{\mathbf{n}_1, .., \mathbf{n}_n\}$ defines the final orientation, and vertex j of the extracted surface

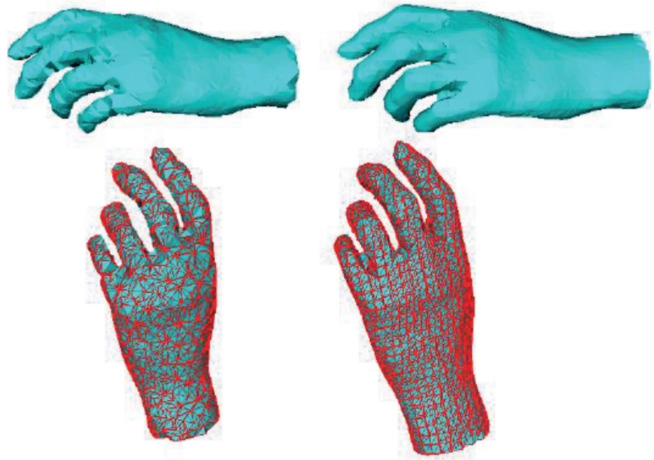

Figure 10. Isosurface extraction based on our method on the high-resolution FCC lattice (left) and based on the marching cubes method on the high-resolution Cartesian cubic lattice (right). Two different views are provided in each column.

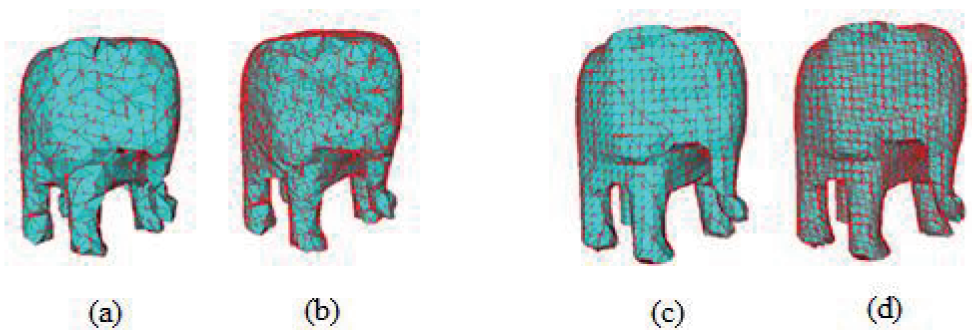

Figure 11. Isosurface extraction based on our method (a) and marching cubes method (c) on the low- (a, c) and high-resolution (b, d) FCC lattices and Cartesian cubic lattices.

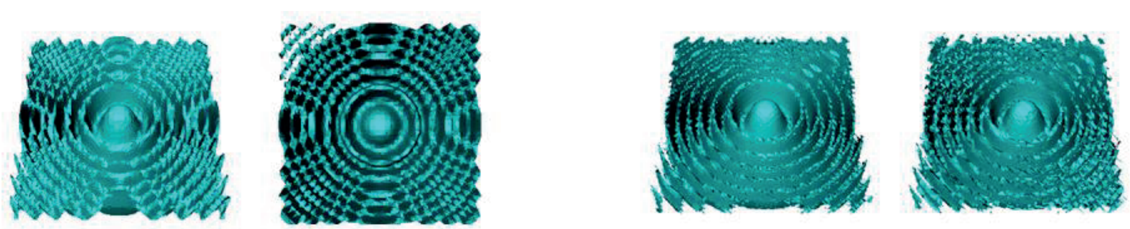

Figure 12. ML isosurface extracted from the FCC lattice with our marching method (left) and from the Cartesian lattice with the marching cubes method (right). Two different views are provided for each isosurface output.

with the unit normal $\mathbf{n}'_j$ is in correspondence with the closest vertex i of the ground-truth surface, whose unit normal is $\mathbf{n}_i$, and

$$q(x) = 1 - xifx > 0, 1ifx \leq 0$$

is used to penalize incompatible normals. Other metrics that quantify the performance of the isosurface extractions are the resolutions of the lattices and output meshes as well as the execution times reported in Tables 1–3.

In Figure 9, we show the visual quality of the isosurface extracted from the marching cubes algorithm

Table 3. Isosurface extraction info for the ML isosurface. MC stands for marching cubes.

Lattice	Method	# Lattice points	Time (s)	# Triangles
FCC	Our	65,536	0.81	21,829
Cartesian	MC	68,921	0.89	19,993

for data points sampled on the Cartesian cubic lattice in comparison to our corresponding quality from the marching algorithm for data points sampled on the FCC lattice. The first noticeable feature is that whereas the Cartesian lattice uses 9270 points (in the form of a $36 \times 15 \times 18$ grid structure), the FCC lattice employs only 6912 points (in $24 \times 12 \times 24$ layout) to achieve a visual quality on a par with the Cartesian framework. We can even favor the output from the FCC lattice, considering that it captures the gap between the fourth and fifth fingers, whereas the Cartesian output does not. We also note that thanks to employing the lower-resolution lattice, extraction with our marching algorithm is faster than that of the marching cubes algorithm. Another advantage of this sparse lattice is the lower number of triangles produced in the final representation. We finally decrease the resolution of the Cartesian lattice to be close to the FCC lattice by using $32 \times 13 \times 16 = 6656$ points, and we observe similar execution times and final number of triangles at the expense of degraded mesh quality (please see Table 1).

Figure 10 provides a similar comparison with increased lattice resolutions to see the behavior of the execution times and the final number of triangles. We upsample the Cartesian lattice to 19,467 points and the corresponding FCC lattice has 12,450 points. A linear increase in execution time reveals the scalability of our approach, as shown in Table 1. Note that our competitor, namely the marching cubes method on the Cartesian cubic lattice, still fails to capture the gap between the fourth and fifth fingers despite the increased resolution. The final number of triangles generated with the marching cubes method on the Cartesian lattice is about 1.5 times higher than our result (5872 vs. 3847), which is yet another advantage of our method.

As far as the execution times are concerned, we observe in Table 1 that extractions from the FCC lattice via our marching algorithm are faster than the marching cubes method running on the Cartesian cubic lattice.

Figure 11 and the accompanying Table 2 demonstrate the extraction of a different isosurface with the same comparison settings as the hand isosurface. It is again observed that in significantly less time the FCC lattice produces an isosurface that is on par with the Cartesian result in terms of visual quality. For the isosurfaces in Figure 11 (top), the FCC lattice is composed of $16 \times 14 \times 24 = 5376$ sampling points, whereas the Cartesian lattice has a higher resolution of $24 \times 18 \times 17 = 7344$ points. For the isosurfaces in Figure 11 (bottom), the FCC and Cartesian lattices have 10,210 and 15,422 points, respectively. Thanks to the reduced sampling rate of our FCC lattices, the average number of triangles needed to represent the elephant isosurface is about 1.5 times less with our method compared to the marching cubes results.

We also perform further quantitative evaluation of our extraction in comparison to the marching cubes extraction, based on the popular ad hoc tool Metro [31]. To this end, we compare the geometric Euclidean distance from our resulting surface and the marching cubes surface to the ground-truth surface based on the closest matching points as 0.29 and 0.33, respectively, where the maximum average distance is normalized to 1. The same ordered pair of numbers is (0.28, 0.32) for the elephant model, which again favors our resulting extraction in both metrics.

We finally provide comparisons for the extractions based on synthetic data, namely analytical functions. The first function is the Marschner–Lobb (ML) isolevel function introduced in [32]:

$$f(\mathbf{x}) = [(1 - \sin(\pi z/2)) + \mu(1 + \Omega(sqrt(x^2 + y^2)))]/2(1 + \mu)$$

where

$$\Omega(r) = \cos(2\pi\alpha\cos(\pi r/2)),$$

where x, y, z are coordinates of the point $\mathbf{x}$, $\mu = 0.25$, and $\alpha = 6$, as suggested by [32]. When executed on the $41 \times 41 \times 41$ Cartesian and $32 \times 32 \times 64$ FCC lattices, i.e. roughly the same resolutions with approximately 67K points, marching cubes and our marching method produce the images in Figure 12. The isosurface extracted from the non-Cartesian FCC lattice by our marching method is better than that of the former method, i.e. marching cubes on the Cartesian lattice. This verifies our other positive claim about FCC lattices, mentioned in the beginning of Section 4, that we infer more information from the FCC lattice than from the Cartesian lattice when both lattices are equipped with the same number of points. We also note that the execution times of both methods are similar, as are the lattice resolutions (Table 3). We further support this claim by noting that when the cubic lattices in Figures 9 and 11 are downsampled to 6.9K and 5.3K points, respectively, to match the number of samples used by the corresponding FCC lattices, we see losses in visual quality.

We also note in Table 3 that the number of triangles in the resulting ML surfaces favors the marching cubes method, as we are using lattices of almost equal density. In such a case, it is natural to expect more triangles from our method, as the marching tetrahedra algorithm is known to produce more triangles than the marching cubes algorithm. Finally, it was recently shown that the ML function is not convenient for the comparison of cubic, BCC, and FCC lattices, as it does not have a spherical frequency spectrum [3]. We nevertheless want to provide extractions based on this function, as it demonstrates the advantages of our method, such as smoothness, and its disadvantages, such as excessive triangles.

The other synthetic analytical functions we employ are the spherical with linear falloff:

$$f(\mathbf{x}) = sqrt(x^2 + y^2 + z^2),$$

and the following simple max function:

$$f(\mathbf{x}) = \max(x, y, z).$$

Using spherical falloff, we have the chance to compare our resulting isosurface with the ground-truth spherical isosurface that we obtain by starting with a crude tetrahedron approximation and repeatedly bisecting the facets while simultaneously moving them to the surface of the sphere. We observe in Figure 13 that our isosurface mesh is as smooth as the ground-truth isosurface obtained at the subdivision level with almost the same number of points as our mesh. Figure 13 also shows the smoothness of the three planes, resulting from the processing of the simple max function.

Figure 13. Ground-truth sphere surface obtained by subdividing an initial tetrahedron (left pair) that has the same smoothness as our resulting sphere isosurface extracted from the FCC lattice (middle pair). Another smooth surface from our method matches the ground-truth planes (rightmost).

All execution times are obtained on a laptop with 2 GB of RAM and a 2-GHz processor. The system can easily be sped up further once the marching procedure, which processes each tiling cell independently, is spread on multiple cores. We finally note that the source code, executable, and video for the method that we present in this paper are publicly available.

5. Conclusion

We have introduced a novel isosurface extraction scheme based on the efficient FCC lattice and marching techniques. Having tiled up the volumetric space with the rhombic dodecahedra cells, we decompose each tiling cell into tetrahedra so as to apply the marching tetrahedra algorithm. Tests on the implicit data, based on different isovalue functions, revealed the quantitative and qualitative performance as well as the time efficiency of our FCC-based results in comparison with the conventional Cartesian-based results. We point out the importance of the tetrahedra alignment, whose absence might otherwise create topological inconsistencies in the resulting isosurface manifold.

After careful decomposition of the rhombic dodecahedron cells of the FCC lattice into tetrahedra, our method consists of the marching tetrahedra algorithm, which may produce an excessive number of triangles while handling the topological ambiguity problems. As future work, we consider developing an edge collapse procedure based on a convenient error metric to reduce the triangle count. Another solution may be the incorporation of the marching cubes method into our framework, as it produces fewer triangles than the marching tetrahedra. We should, however, be careful in designing this hybrid framework, as the marching cubes method may lead to topological problems on some configurations, especially when coupled with the marching tetrahedra algorithm.

Acknowledgement

This work has been supported by TÜBİTAK under the project EEEAG-115E471.

References

[1] Lorensen WE, Cline HE. Marching cubes: a high resolution 3D surface construction algorithm. In: SIGGRAPH 1987 Conference on Computer Graphics and Interactive Techniques; 27–31 July 1987; Anaheim, CA, USA. New York, NY, USA: ACM, pp. 163-169.

[2] Theussl T, Moller T, Groller M. Optimal regular volume sampling. In: IEEE 2001 Conference on Visualization; 21–26 October 2001; San Diego, CA, USA. New York, NY, USA: IEEE. pp. 91-98.

[3] Vad V, Csebfalvi B, Rautek P, Groller E. Towards an unbiased comparison of CC, BCC, and FCC lattices in terms of prealiasing. Comp Graph Forum 2014; 33: 81-90.

[4] Keppel E. Approximating complex surfaces by triangulation of contour lines. IBM J Res Dev 1975; 19: 2-11.

[5] Herman GT, Lun HK. Three-dimensional display of human organs from computed tomograms. Comput Vision Graph 1979; 9: 1-21.

[6] Bourke P. Polygonising A Scalar Field Using Tetrahedrons. Norwich, UK: University of East Anglia, 1997.

[7] Strand R. Interpolation and sampling on a honeycomb lattice. In: IEEE 2010 International Conference on Pattern Recognition; 23–26 August 2010; İstanbul, Turkey. New York, NY, USA: IEEE. pp. 2222-2225.

[8] Middleton L, Sivaswamy J. Hexagonal Image Processing. New York, NY, USA: Springer, 2005.

[9] Treece GM, Prager RW, Gee AH. Regularised marching tetrahedra: improved iso-surface extraction. Comput Graph 1999; 23: 583-598.

[10] Carr H, Theussl T, Moller T. Isosurfaces on optimal regular samples. In: VISSYM 2003 Symposium on Data Visualization; 26–28 May 2003; Grenoble, France. Aire-la-Ville, Switzerland: Eurographics Association. pp. 39-48.

[11] Takahashi T, Yonekura T. Isosurface construction from a data set sampled on a face-centered-cubic lattice. Proc ICCVG 2002; 2: 754-763.

[12] Petkov K, Qiu F, Fan Z, Kaufman A, Mueller K. Efficient LBM visual simulation on face-centered cubic lattices. IEEE T Vis Comput Gr 2009; 15: 802-814.

[13] Strand R, Stelldinger P. Topology preserving marching cubes-like algorithms on the face-centered cubic grid. In: IEEE 2007 International Conference on Image Analysis and Processing; 10–14 September 2007; Modena, Italy. New York, NY, USA: IEEE. pp. 781-788.

[14] Wells P, Smith R, Suparta G B. Sampling the sinogram in computed tomography. Mater Eval 1997; 5: 772-783.

[15] Elaff I, El-Kemany A, Kholif M. Universal and stable medical image generation for tissue segmentation (the unistable method). Turk J Elec Eng & Comp Sci 2017; 25: 1070-1081.

[16] Aydoğan T, Bayılmış C. A new efficient block matching data hiding method based on scanning order selection in medical images. Turk J Elec Eng & Comp Sci 2017; 25: 461-473.

[17] Hang X, Paragios N, Metaxas D. Shape registration in implicit spaces using information theory and free form deformations. IEEE T PAMI 2006; 28: 1303-1318.

[18] Eyiyurekli M, Breen D. Interactive free-form level-set surface-editing operators. Comput Graph 2010; 34: 621-638.

[19] Hoppe H, DeRose T, Duchamp T, McDonald J, Stuetzle W. Surface reconstruction from unorganized points. In: SIGGRAPH 1992 Conference on Computer Graphics and Interactive Techniques; 26–31 July 1992; Chicago, IL, USA. New York, NY, USA: ACM. pp. 71-78.

[20] Frisken P, Rockwood J. Adaptively sampled distance fields: a general representation of shape for computer graphics. In: SIGGRAPH 2000 Conference on Computer Graphics and Interactive Techniques; 23–28 July 2000; New Orleans, LA, USA. New York, NY, USA: ACM. pp. 249-254.

[21] Kobbelt L, Botsch M, Schwanecke U, Seidel HP. Feature-sensitive surface extraction from volume data. In: SIGGRAPH 2001 Conference on Computer Graphics and Interactive Techniques; 12–17 August 2001; Los Angeles, CA, USA. New York, NY, USA: ACM. pp. 57-66.

[22] Sahillioğlu Y, Yemez Y. Coarse-to-fine surface reconstruction from silhouettes and range data using mesh deformation. Comput Vis Image Und 2010; 114: 334-348.

[23] Petersen DP, Middleton D. Sampling and reconstruction of wave-number-limited functions in N-dimensional Euclidean spaces. Info Control 1962; 5: 279-323.

[24] Bloomenthal J, Bajaj C. Introduction to Implicit Surfaces. New York, NY, USA: Kaufmann, 1997.

[25] Botsch M, Kobbelt L, Pauly M, Alliez P, Levy B. Polygon Mesh Processing. Boca Raton, FL, USA: CRC Press, 2010.

[26] Wells AF. Structural Inorganic Chemistry. 5th ed. Oxford, UK: Oxford University Press, 2012.

[27] Jackson AG. Handbook of Crystallography. New York, NY, USA: Springer, 2011.

[28] Marder MP. Condensed Matter Physics. 2nd ed. Hoboken, NJ, USA: Wiley, 2010.

[29] Ashcroft NW, Mermin ND. Solid State Physics. Upper Saddle River, NJ, USA: Prentice Hall, 1976.

[30] Wolfram Web Resources. Rhombic Dodecahedron. MathWorld, 2017. Available online at http://mathworld.wolfram.com/RhombicDodecahedron.html.

[31] Cignoni P, Montani C, Scopigno R. A comparison of mesh simplification algorithms. Comput Graph 1998; 22: 37-54.

[32] Marschner SR, Lobb RJ. An evaluation of reconstruction filters for volume rendering. In: IEEE 1994 Conference on Visualization; 21–28 October 1994; Washington, DC, USA. New York, NY, USA: IEEE. pp. 100-107.