Pergamon

PII: S0097-8493(97)00085-X

Comput. & Graphics, Vol. 22, No. 1, pp. 83–90, 1998

Printed in Great Britain
0097-8493/98 $19.00 + 0.00

Technical Section

A NEW TETRAHEDRAL TESSELATION SCHEME FOR ISOSURFACE GENERATION

S. L. CHAN† and E. O. PURISIMA

National Research Council of Canada, Biotechnology Research Institute, 6100 Royalmount Avenue, Montréal, Québec, H4P 2R2, Canada

Abstract—The marching cubes algorithm is widely used to generate isosurfaces from a 3D scalar field. A major problem associated with it is the possibility of mismatch between adjacent surface elements, leading to holes on the surface. In this work, we propose using a tetrahedral tesselation of space which would eliminate this problem. Comparing with existing remedies of the marching cubes method which subdivide each cube into tetrahedra, the current tesselation is more regular and symmetric. © 1998 Elsevier Science Ltd. All rights reserved

Key words: isosurface generation, marching cubes, tetrahedral tessellation.

1. INTRODUCTION

The marching cubes algorithm [1] is a popular method for generating contour surfaces for visualizing a data field in 3D space. In the marching cubes algorithm, a grid with regular cubic elements is set up over the region of interest. Functional values are sampled at the grid points, either by calculation or through a scanning instrument. Grid points with values above a pre-defined cutoff are switched on while those with values below the cutoff are switched off. The algorithm then goes through the cubes where not all vertices assume the same on–off state. An element of the contour surface is generated in such cubes. After processing all relevant cubes, the whole contour surface can be constructed by assembling all these elements together.

Since each cube has eight vertices, there are $2^8 = 256$ possible patterns for the on–off states of its vertices. Fortunately, inversion symmetry and rotation symmetry together can reduce the number of topologically different patterns to 15 [1]. (Actually the number can be further reduced to 14 if reflection symmetry is taken into account too.) For each of these elementary patterns, the topology of intersection of the contour surface with the cube can be predetermined and triangulated and stored in an array. For cubes that do not have all vertices at the same state, an element of the contour surface can be constructed by recalling the appropriate array element.

To improve the quality of the surface, the positions of the intersections of the surface element on the cube are determined via interpolation of the data. Furthermore, unit normals can also be calculated at the grid points and subsequently interpolated onto the surface elements for gouraud shading, giving a better feel of the surface.

A major problem facing the marching cubes algorithm is that ambiguity may arise when cubes of certain patterns are adjacent to each other, leading to holes in the resulting contour surface [2]. Figure 1 illustrates such a situation. To describe it in simple terms, when opposite vertices of the square between a pair of cubes assume the same state (while the other pair of opposite vertices assume the other state), one may assume either that the on-vertices are connected topologically within the square section, or that the off-vertices are connected. Much effort has been put in to rectify this problem [3–11].

It has been observed that the pattern mismatch problem mentioned above can be overcome by the use of tetrahedral instead of cubic cells. Moreover, the number of different possible topological patterns per cell element is greatly reduced upon the use of tetrahedral cells.

One can obtain a tetrahedral tesselation of space by sub-dividing the cube in the marching cubes algorithm [9–11]. However, the resulting tesselation is not very regular (there are more than one type of tetrahedra with considerable difference in volume; the ratio of the lengths between the different types of edges is big). In the following, we shall derive a tetrahedral tesselation of space that is relatively regular and symmetric. Moreover, we shall show that using the present algorithm, the sampling efficiency is improved upon the conventional marching cubes algorithm.

2. CONSIDERATION

Here we shall look for a good way of tesselating space using tetrahedra. We shall use it as the basic

†Author for correspondence. Dr S. L. Chan, Tripos Inc., 1699 S Hanley Road, St Louis, Mo 63144, U.S.A.

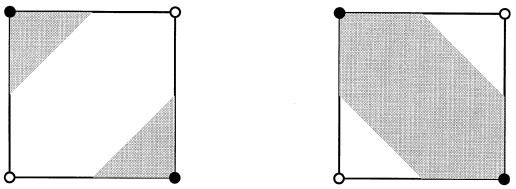

Fig. 1. Possibility of mismatch between adjacent cubes on their mutual interface. On-vertices are represented by filled circles and off-ones by open circles. Shaded areas represent volume inside the contour surface. If a cube with the pattern on the left lies next to one with the pattern on the right, the contour surface will not join up on their mutual interface.

building block for the marching cubes algorithm. We understand that the regular tetrahedron won't work because it cannot be used to tesselate space without leading to voids in the packing. Our aim is to look for a tesselation that is simple, highly symmetric and as regular as possible.

A lattice is defined as an infinite array of points in space with identical environments, including orientation. It offers a natural starting point for constructing a repeating tesselation of space. There are altogether only 14 types of lattices in 3D space, called the Bravais lattices [12]. These 14 lattice types can be classified into seven systems which are characterized by different symmetry elements [12]. Of these, the cubic system is the one displaying the highest symmetry. We shall focus our attention on the 3 types of lattices in the cubic system: the cubic primitive, the body-centred cubic, and the face-centred cubic lattices (Fig. 2).

Consider the following algorithm: join all pairs of neighboring lattice points within a certain cutoff distance by line segments, and look for any tetrahedra thus defined, *i.e.* any quadruplets of points which are all connected to each other. If there is none, join all second nearest neighbors and look again. We shall see how fast one can end up with a tetrahedral tesselation.

For the primitive lattice, after the first neighbors are connected, cubes are formed. In the body-centred lattice, upon connection of the first neighbors (the dotted lines in Fig. 2(b)), no tetrahedron is defined. In the face-centred lattice, upon joining the first neighbors (*e.g.*, the dotted lines in Fig. 2(c). For clarity purposes, only features on the three faces facing the reader are shown.), some regular tetrahedra are formed (*e.g.* PQR in Fig. 2(c)) but they do not fill up all the space.

Upon the joining of second neighbors, 'crosses' appear on the faces of the cubes in the primitive lattice (dotted lines in Fig. 2(a). For clarity purposes, only features on the three faces facing the reader are shown.). By cutting the cube up along *some* of these lines, each cube can be decomposed into five tetrahedra (Fig. 3): four identical ones plus a fifth (a regular tetrahedron) at the centre that is twice as

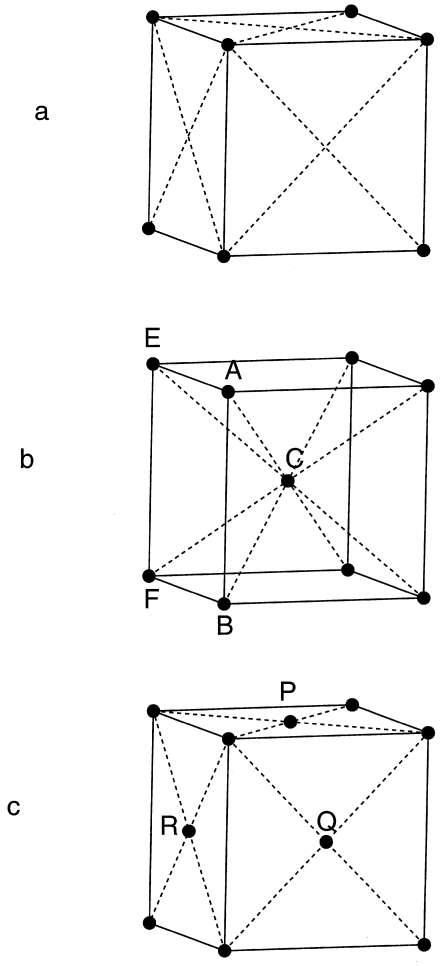

Fig. 2. The three cubic lattices: (a) the cubic primitive, (b) the body-centered cubic, (c) the face-centered cubic. For clarity purposes, features on backward facing faces (which are identical to the forward facing faces) are not shown in (a) and (c). A cubic unit cell of each lattice is drawn. When each cube in an infinite cubic grid is decorated as that in the unit cell, the whole (infinite) lattice is obtained. Note that for each lattice, the relative distribution of neighboring lattice points is identical for each lattice point.

big as any of the other four. This way of breaking each cube down into several tetrahedra has in fact appeared in some previous attempts to overcome the pattern-mismatch problem in the marching cubes method [9–11].

In the body-centred lattice, lattice points separated by an edge of a cube are second neighbors (solid lines in Fig. 2(b)). The second neighbors of the lattice point at the centre of a cube (C in Fig. 2(b) and Fig. 4) are lattice points at the centre of the six neighboring cubes (*e.g.* D in Fig. 4). Therefore after second neighbors are connected, one ends up with a nice tesselation of tetrahedra (Fig. 4). They do not overlap, and they fill up all the space without leaving any void around. Compared with the tetrahedra created in the primi-

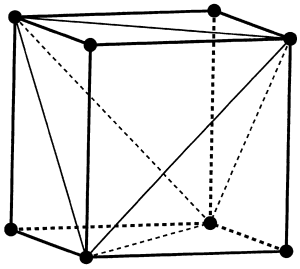

Fig. 3. A cube can be broken down into five tetrahedra when cut along the lines shown.

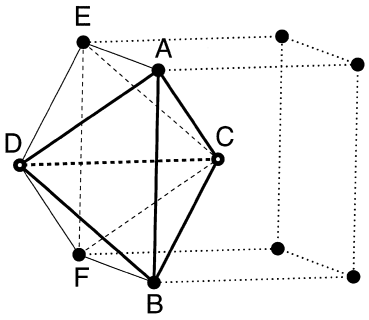

Fig. 4. A tesselation of space, derived from the body-centered cubic lattice, using only one type of tetrahedron (e.g. ABCD, BFCD, FECD, EACD). Compare with Fig. 2(b) for a better interpretation. D is the lattice point at the center of the cube to the left of face ABFE. All interfaces between adjacent tetrahedra are identical. There are two types of edges: AB and CD are of the longer type; AC, AD, BC, BD are of the shorter type.

tive lattice (previous paragraph), the current tesselation is more regular: there is only one type of tetrahedral shape; and the ratio of lengths of the longer edges to the shorter edges is smaller.

In the case of the face-centred lattice, many tetrahedra are formed upon connection of the second neighbors (*e.g.* solid lines in Fig. 2(c)). However, the ratio of lengths of the longer edges to the shorter edges is bigger than that in the body-centred lattice, suggesting that the tetrahedra are less regular. Table 1 gives these length ratios as well as a quantitative summary of the first and second neighbors.

We thus conclude that a good way of tesselating space using tetrahedra can be derived from the body-centred cubic lattice and we shall use it as the basic building block for the marching cubes algorithm.

3. METHOD

Based on the results of the previous section, we derived a tetrahedral tesselation of space from the body-centred cubic lattice in the manner illustrated in Fig. 4. Note that all tetrahedra obtained this way have the same shape although their orientations differ.

There are two types of edges on this tesselation of tetrahedra. The short edges join a lattice point with one of its eight nearest neighbors. Examples are AC, AD, BC and BD in Fig. 4. The long edges join a lattice point with one of its six second-nearest neighbors. Examples are AB and CD in Fig. 4. There are two long edges and four short edges on each tetrahedron. There are four tetrahedra surrounding each long edge and six tetrahedra surrounding each short edge.

Now let's consider the different patterns of the vertices' on–off states of each tetrahedron. When all four vertices are at the same state, no surface element needs to be constructed. When three vertices assume a state different from the fourth one, a surface can be constructed parallel to the triangle defined by the three vertices. Since all triangles are identical on the tetrahedron, there is only one pattern. Finally, when two vertices assume a state different from the other two, there can be two (not significantly different) possibilities: (a) identical vertices are joined by a long edge, or (b) identical vertices are joined by a short edge. In either case, a quadrilateral surface element can be constructed bisecting the tetrahedron. Hence we see that there are altogether only four possible patterns for each tetrahedron, as opposed to 15 in the marching cubes setup.

We have noted that there have been previous efforts to rectify the problem of mismatch between adjacent cubes by breaking the cube, the basic

Table 1. The number of neighboring lattice points and their distances

	no. of points	Distance/l*	Examples in Fig. 2	Distance ratio†
Cubic primitive				
first neighbors	6	1	solid lines	1.414
second neighbors	12	$\sqrt{2}$	dotted lines	
Body-centred cubic				
first neighbors	8	$\sqrt{3}/2$	dotted lines	1.155
second neighbors	6	1	solid lines	
Face-centred cubic				
first neighbors	12	$1/\sqrt{2}$	dotted lines	1.414
second neighbors	6	1	solid lines	

*l is the linear dimension of the cubic unit cell as displayed in Fig. 2.
†ratio of distance to second neighbor to the distance to first neighbor.

building block in the marching cubes algorithm, down into several tetrahedra [9–11]. In these works, there is a considerable variation in the shape and in some cases even the volume of the tetrahedra involved. In contrast, there is only one tetrahedral shape involved in the current work. Moreover, there is also an element of asymmetry involved in breaking the cubes down to tetrahedra and some arbitrariness in the choice of the orientation of the breakdown. Although a method has been developed to overcome this arbitrariness [13], extra computational effort will be involved. In contrast, the current tesselation of tetrahedra is more regular and symmetric overall.

4. SAMPLING EFFICIENCY

Another advantage of using the current setup based on the body-centred lattice is that sampling of space is more efficient than in the conventional marching cubes method which is based on the primitive lattice. For the same density of lattice points where sampling is carried out, the maximum possible distance $d_{\max}$ from any point to the nearest lattice point is reduced. This maximum distance is that between a point at the centre of a cube and the nearest lattice points in the primitive lattice, and that between a point at the centre of a tetrahedron and the nearest lattice points in the body-centred lattice. If a is defined such that $\frac{1}{a^3}$ is the density of lattice points (*i.e.* a^3 is the effective volume associated with each lattice point), for the primitive lattice,

$$d_{\max} = \frac{\sqrt{3}}{2}a = 0.866a$$

while for the body-centred lattice,

$$d_{\max} = \frac{\sqrt{5}}{(2)^{\frac{5}{3}}}a = 0.704a.$$

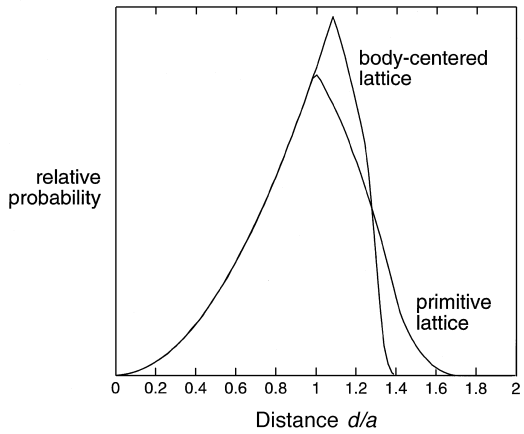

Fig. 5. The relative probability of finding a point at a distance d from the nearest lattice point for two types of lattices. a is defined such that $1/a^3$ is the density of lattice points.

Fig. 5 gives a plot of the relative probability of finding a point at a distance d from the nearest lattice point in the two lattices. The two curves are normalized to the same vertical scale so that the areas under the curves are the same. The curves coincide up to $d = a$. It can be seen from the curves that the expectation value of d is also smaller in the body-centred lattice than in the primitive lattice.

Figure 5 together with the results on $d_{\max}$ suggest that the current algorithm provides a better coverage of space for the same number of sampling scans or functional evaluations. For applications where sampling is time-consuming, this is an advantage of sampling using the body-centred lattice over the primitive lattice.

5. IMPLEMENTATION

Similar to the marching cubes algorithm, the intersections of the contour surface on the edges of the tetrahedra are interpolated to improve the quality of the surface. Here we would like to mention, as one of the reviewers pointed out, that data at the four vertices of a tetrahedron uniquely define a linear function which interpolates the data at the vertices. This linear function has isosurfaces which are guaranteed to be planar. For many graphical applications, such planar polygons can be used directly for rendering. Otherwise these quadrilateral elements can be readily decomposed into two triangles.

Similar to other approaches based on tetrahedral volume elements [9–11], the current approach suffers from a drawback of having a high triangle count [14]. Nevertheless, for the simplest implementation of topologically consistent surfaces, tetrahedron-based methods have their attractiveness [14]. In algorithms where each cube is broken down into five or six tetrahedra [9–11], each sampling point is associated with five or six tetrahedra. In the current approach, there are 12 tetrahedra associated with each cube. Since there are two sampling points per cube, the number of tetrahedra per sampling point is six. Hence the triangle count of the current method is similar to that of previous approaches that breaks down the cube into tetrahedra. To confirm this, we compared our results from those obtained from Bloomenthal's marching cubes program [15] with tetrahedral decomposition. The testing functions used were the torus and the three-pole-blend function that came with Bloomenthal's program. For identical sampling point densities, 6868 triangles were generated with Bloomenthal's program while 6512 triangles were generated with our program. For the three-pole-blend function, Bloomenthal's program generated 10136 triangles while our program generated 9828 triangles.

Figure 6(a) gives a view of the triangulated surface created this way for the isoprobability surface of the d_{z^2} electron orbital of a hydrogen atom.

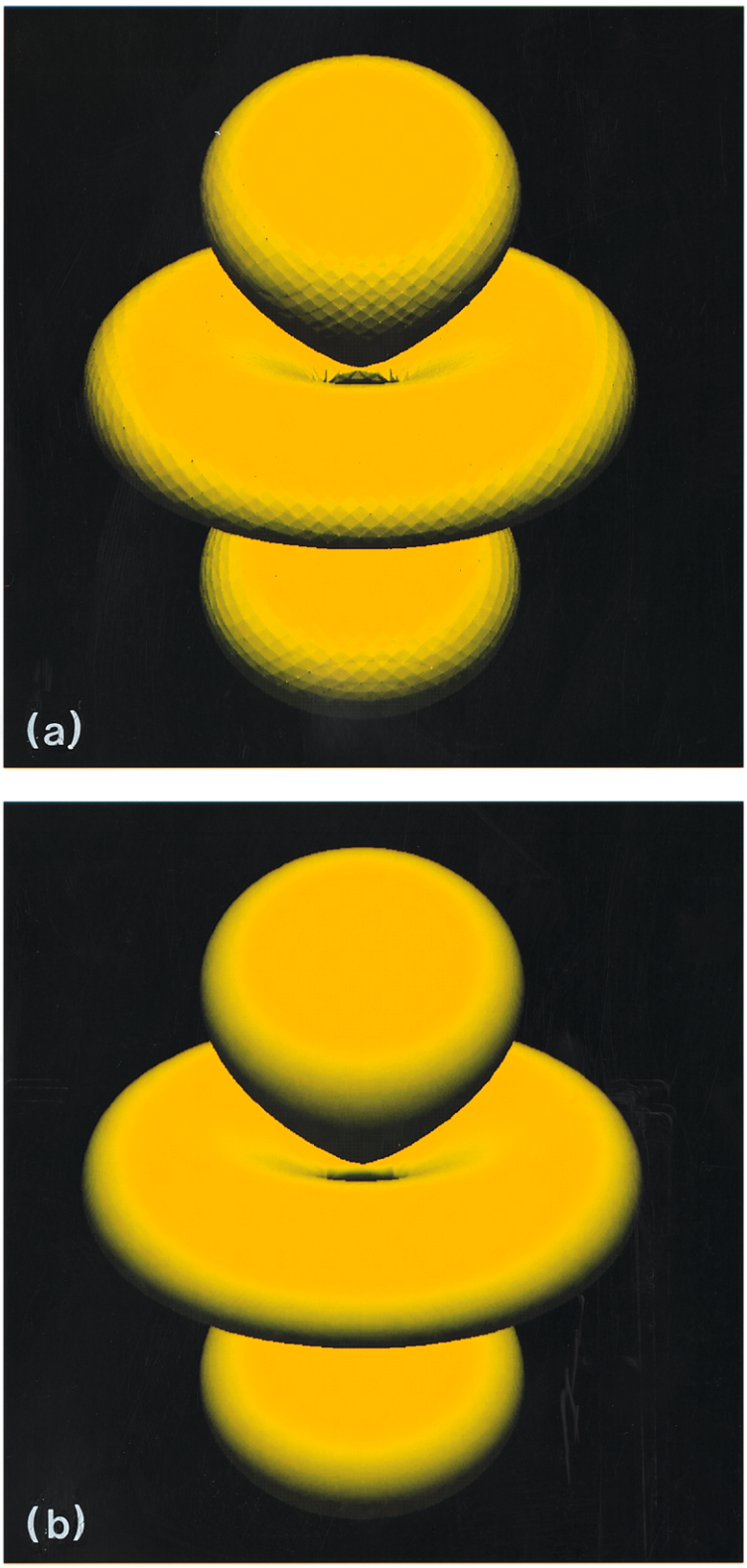

Fig. 6. A probability contour for the d_{z^2} electron orbital of a hydrogen atom. (a) the contour surface before gouraud shading, (b) the surface after gouraud shading.

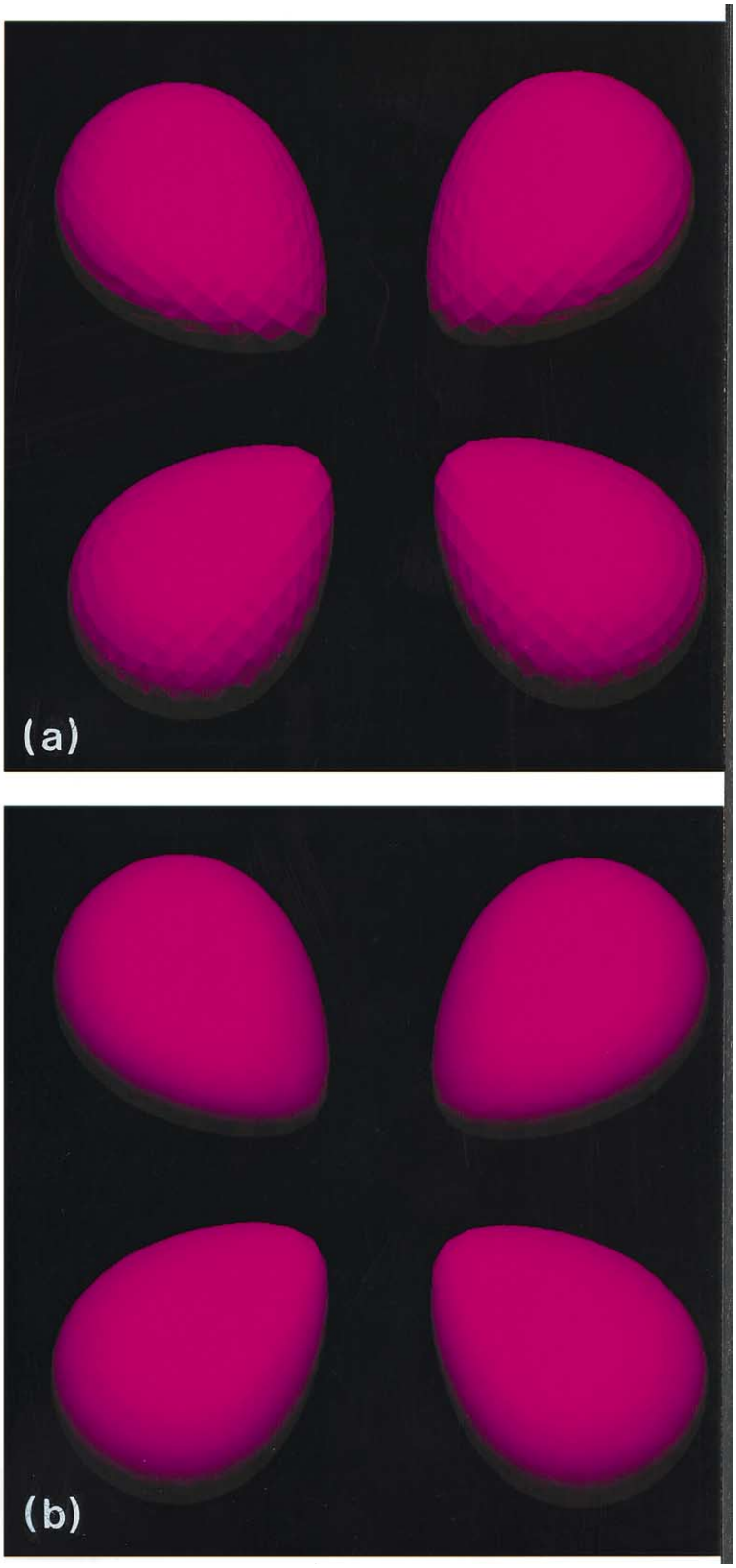

Fig. 7. A probability contour for the $d_{x^2-y^2}$ electron orbital of a hydrogen atom. (a) the contour surface before gouraud shading , (b) the surface after gouraud shading.

Figure 6(b) gives the same surface smoothened by gouraud shading. The method of calculating the unit normals is given in the appendix. Figure 7a and 7b give the corresponding plots for the $d_{x^2-y^2}$ electron orbital of a hydrogen atom.

In some scanning instruments, sampling may be constrained to a cubic primitive lattice grid like that used in the conventional marching cubes method. Fortunately points in the body-centred cubic lattice can be considered as a subset of the points in the cubic primitive lattice. Therefore if the maximum resolution of the scanning instrument is not reached, one can obtain the data on a body-centred lattice simply by sampling only at the appropriate points (*e.g.*, at points of a cubic grid where the x, y, z indices are either all even or all odd). However, if the highest resolution is required, one will need to fill in all data points on a body-centred lattice by interpolation. This will unfortunately slow down the procedure.

6. CONCLUSION

A major problem facing the original marching cubes algorithm for isosurface generation is the pattern mismatch problem [2]. It has been noted that the use of tetrahedral instead of cubic cell elements can eliminate this problem. Moreover, the number of different possible topological patterns for each cell element is also greatly reduced upon the use of tetrahedral cells.

In this work we have introduced a new tetrahedral tesselation of space for the marching cubes algorithm. Compared with existing methods that subdivide the cube into tetrahedra, the current tesselation is more regular and symmetric since space is directly divided into tetrahedral elements instead of going through the intermediate step of the cube. There is only one tetrahedral shape involved in the current algorithm, and the ratio of the longer edges to the shorter edges is relatively small. Moreover, it has been shown that sampling efficiency is improved compared to using the standard cubic grid.

However, for applications that involve a scanning instrument, our use of a body-centred cubic lattice may not be readily compatible. One has to either sample at a subset of points of the simple cubic grid, or interpolate to obtain the value at the centre of each cubic element, which will lead to a performance penalty.

Acknowledgements—This is National Research Council of Canada publication no. 39982.

REFERENCES

1. Lorensen, W. E. and Cline, H. E., Marching Cubes: A High Resolution 3D Surface Construction Algorithm. *Computer Graphics*, 1987, **21,** 163–169.
2. Dürst, M. J., Letters: Additional Reference to Marching Cubes. *Computer Graphics*, 1988, **22,** 72–73.
3. Röll, S., Haase, A. and von Kienlin, M., Fast Generation of Leakproof Surfaces from Well-Defined Objects by a Modified Marching Cubes Algorithm. *Computer Graphics Forum*, 1995, **14,** 127–138.
4. Xiang, Z., Shi, Y. and Xu, Y., Calculating the Electric Potential of Macromolecules: A Simple Method for Molecular Surface Triangulation. *Journal of Computational Chemistry*, 1995, **16,** 512–516.
5. Zhou, C., Shu, R. and Kankanhalli, M. S., Handling Small Features in Isosurface Generation Using Marching Cubes. *Computers and Graphics*, 1994, **18,** 845–848.
6. Matveyev, S. V., Approximation of Isosurface in the Marching Cube: Ambiguity Problem. In *Proceedings of Visualization '94*. IEEE, 1994, pp. 288–292.
7. Nielson, G. M. and Hamman, B., The Asymptotic Decider: Resolving the Ambiguity in Marching Cubes. In *Proceedings of Visualization '91*. IEEE, 1991, pp. 83–90.
8. Wilhelms, J. and van Gelder, A., Topological Considerations in Isosurface Generation Extended Abstract. *Computer Graphics*, 1990, **24,** 79–86.
9. Payne, B. A. and Toga, A. W., Surface Mapping Brain Function on 3D Models. *IEEE Computer Graphics & Applications*, 1990, **10,** 33–41.
10. Hall, M. and Warren, J., Adaptive Polygonization of Implicitly Defined Surfaces. *IEEE Computer Graphics & Applications*, 1990, **10,** 33–42.
11. Guéziec, A. and Hummel, R., Exploiting Triangulated Surface Extraction Using Tetrahedral Decomposition. *IEEE Transactions on Visualization and Computer Graphics*, 1995, **1,** 328–342.
12. A description of these lattices can be found in most text books in crystallography, *e.g.*, D. McKie, C. McKie, Essentials of Crystallography, Blackwell Scientific, Publications, Oxford, 1986.
13. Zhou, Y., Chen, W. and Tang, Z., An Elaborate Ambiguity Detection Method for Constructing Isosurfaces within Tetrahedral Meshes. *Computers and Graphics*, 1995, **19,** 355–364.
14. Ning, P. and Bloomenthal, J., An Evaluation of Implicit Surface Tilers. *IEEE Computer Graphics & Applications*, 1993, **13,** 33–41.
15. J. Bloomenthal, An Implicit Surface Polygonizer. In *Graphics Gems IV*, ed. P. S. Heckbert. Academic Press, 1994, pp. 324–350.

APPENDIX

calculating the unit normals

Let (i, j, k) denote a lattice point, $f(i, j, k)$ denote the functional value at this lattice point, and $\nabla f_x(i, j, k)$ denote the x component of the gradient vector at this lattice point.

The most commonly used approximation for deducing $\nabla f_x(i, j, k)$ in a primitive cubic lattice is setting

$$\nabla f_x(i, j, k) = \frac{f(i+1, j, k) - f(i-1, j, k)}{2a}$$

where a is the lattice repeating distance in the x, y and z directions.

This may not be the best approximation in the case of a body-centred lattice, because the lattice points at $(i + 1, j, k)$ etc. are not the ones nearest to the lattice point at (i, j, k). Instead, there are eight nearest neighboring lattice points located at $(i + 1/2, j + 1/2, k + 1/2)$ etc. Hence we take into account the functional values at these lattice points too. We set

$$\nabla f_x(i,j,k) = c\left(\frac{f(i+1,j,k) - f(i-1,j,k)}{2a}\right)$$
$$+\frac{1-c}{4}\left(\frac{f\left(i+\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2}\right) - f\left(i-\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2}\right)}{a}\right)$$
$$+\frac{1-c}{4}\left(\frac{f\left(i+\frac{1}{2},j+\frac{1}{2},k-\frac{1}{2}\right) - f\left(i-\frac{1}{2},j+\frac{1}{2},k-\frac{1}{2}\right)}{a}\right)$$
$$+\frac{1-c}{4}\left(\frac{f\left(i+\frac{1}{2},j-\frac{1}{2},k-\frac{1}{2}\right) - f\left(i-\frac{1}{2},j-\frac{1}{2},k-\frac{1}{2}\right)}{a}\right)$$
$$+\frac{1-c}{4}\left(\frac{f\left(i+\frac{1}{2},j-\frac{1}{2},k+\frac{1}{2}\right) - f\left(i-\frac{1}{2},j-\frac{1}{2},k+\frac{1}{2}\right)}{a}\right)$$

where c is a weight parameter between zero and one. Different values for c have been tried but there is little effect on the appearance of the surface in the cases we tested.