Edge Groups: An Approach to Understanding the Mesh Quality of Marching Methods

Carlos A. Dietrich Carlos E. Scheidegger João L. D. Comba Luciana P. Nedel Cláudio T. Silva

Abstract— Marching Cubes is the most popular isosurface extraction algorithm due to its simplicity, efficiency and robustness. It has been widely studied, improved, and extended. While a lot of early work was concerned with efficiency and correctness issues, lately there is a push to improve the quality of Marching Cubes meshes so that they can be used for computational experiments. In this work we present a new classification of MC cases that we call Edge Groups, which helps elucidate the issues that impact the triangle quality of the meshes that the method generates. This formulation allows a more systematic way to bound the triangle quality, and is general enough to extend to other polyhedral cell shapes used in other polygonization algorithms. Using this analysis, we also discuss ways to improve the quality of the resulting triangle mesh, including some that require only minor modifications of the original algorithm.

Index Terms—Isosurface extraction, Marching Cubes.

1 Introduction

Marching Cubes (MC) [20] became the most popular algorithm for isosurface extraction due to a powerful combination of simplicity, efficiency and robustness of implementation. There has been much work on improving the original proposal, from understanding the underlying mathematics to improving its performance and extending it to more general settings. With ever-increasing computing power, simulations are now complex enough that mesh generation is an intermediate step in many numerical codes. Because of this, there is a pressing need for algorithms that are fast *and generate good meshes*. Since good meshes make numerical processing faster and more accurate [34], improving the quality of the meshes generated by Marching Cubes has become an important problem. Although much of the published work has involved post-processing and remeshing, there is also interest in modifying the core Marching Cubes routine as well. Our work falls in this category.

In this paper we introduce the concept of *Edge Groups*, which sheds light on the triangle quality generated by MC and other marching methods. The underlying motivation behind our formulation is very simple. MC approximates a scalar field inside each cell by a collection of triangles, with each triangle being formed by intersection points located across cell edges. Considering that the quality of the generated mesh can be found by looking individually at each triangle, we focus on the different ways edges can be combined to form a triangle. Each of these different configurations is a different edge group. Since there is a finite number of edges in a cubic cell, there are only finitely many ways three edges contribute to make a triangle. Even though infinitely many triangles can be formed for each configuration, intersection points are forced to lie on edges, so we can bound the triangle quality for each configuration. We call each of these edge configurations an *Edge Group*.

Using this framework, we revisit marching methods to identify the edge groups in each case. For instance, we show that in MC there are only 8 edge groups (Figure 1), and that each group has a very distinct behavior in regard to their triangle quality bounds. The same idea can be applied to marching methods that use tetrahedral cells.

- *Carlos A. Dietrich, João L. D. Comba and Luciana P. Nedel are with Instituto de Informática, Universidade Federal do Rio Grande do Sul, Brasil. E-mail: {cadietrich, comba, nedel}@inf.ufrgs.br.*
- *Carlos E. Scheidegger and Cláudio T. Silva are with the Scientific Computing and Imaging Institute, University of Utah. E-mail: {cscheid, csilva}@sci.utah.edu.*

Manuscript received 31 March 2008; accepted 1 August 2008; posted online 19 October 2008; mailed on 13 October 2008.
For information on obtaining reprints of this article, please send e-mailto:tvcg@computer.org.

While the identification of edge groups is by itself interesting, we use them to derive ways to improve the triangle quality generated by these algorithms. By evaluating the histograms of triangle quality associated to each edge group of MC, we observe that one case in particular generates a large number of bad triangles. We also show that this particular edge group is responsible for the majority of the worst triangles in MC meshes. From this observation, we changed the connectivity of the triangulation in some entries in the MC table, replacing a triangle defined by this edge group by another edge group. This improves the triangle quality of the mesh by an order of magnitude in most cases we tested.

We also apply a similar analysis to revisit Macet, a method that improves triangle quality by warping the locations of the MC grid before generating the mesh [13]. Edge groups help explain why that method performs well.

In addition, edge groups naturally suggest a modification of the technique that further improves triangle quality in almost all datasets tested. In summary, our main contributions are as follows:

- A new interpretation of marching methods case tables called *edge groups* that helps explain and bound the triangle quality generated by Marching Cubes and its variants.
- A modified MC table based on the edge group formulation that improves the triangle quality of the original MC.
- An improved warping technique that generates triangle meshes with competitive quality to the state of the art while retaining the attractive performance and simplicity of MC.

The paper is organized as follows. In Section 2 we discuss related work in the area. We present the edge groups in Section 3, with a discussion on how it applies to cubic cells. This formulation sets the ground for discussing two ways to improve triangle quality in Section 4. Results of these approaches are listed in Section 5. In Section 6 we discuss our results and evaluate the edge groups on tetrahedral cells. Our conclusions are presented in Section 7. The appendix gives a formal proof on the existence of eight edge groups for MC.

2 Related Work

Several researchers have focused on understanding and describing combinatorial aspects of MC and related techniques. The original MC paper by Lorensen and Cline [20] enumerates the 15 cases by inspection of the possible 256 cases, without a formal proof. Nielson [28] demonstrates the existence of 23 cases by rotation only, despite mirroring and complementary operations, which can produce erroneous results. Banks et al. [2] present a formal proof based on group theory that also generalizes to higher dimensions and other polytopes. Other related papers include the follow-up to this work [3] that uses Pólya

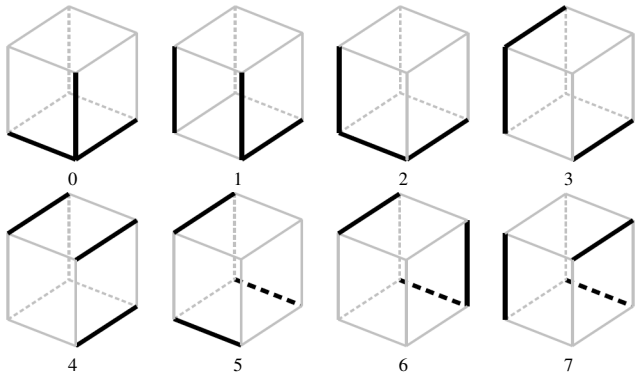


Fig. 1. The eight edge groups observed in MC. All triangles generated by MC come from one of these groups (which include symmetric or rotationally equivalent configurations).

theory, and counting-cases for four-dimensional cases [4]. While they count cases that occur based on the sign of the vertices of the cell, we classify triangles inside each case by looking at the relative connectivity of the edges that generate the vertices. A survey of MC was recently presented by Newman and Yi [24].

Isosurface extraction methods are usually categorized by the general approach they use in computing the isosurface [29]. *Spatial decomposition* methods (introduced by [16]) subdivide the domain of the function f into smaller parts, often called *cells*, and generate local approximations to the isosurface inside each cell [1, 8, 17, 20, 27, 35]. *Surface tracking* methods place seed sampling points on the isosurface and trigger region growing-like algorithms from the seeds, iteratively searching for optimal positions for new sampling points. These algorithms trace their roots to work of Wyvill et al. [38] and have been recently extended [33] to generate higher quality triangulations.

The *divide-and-conquer* nature of spatial decomposition methods often leads to robust and efficient methods, like MC [20], while the optimal placing of each sample performed in the surface tracking methods results in higher-quality meshes. In the past, efficiency and quality have been treated as orthogonal features, but a recent trend is to consider both aspects when designing an algorithm to improve mesh quality [13, 18]. Gibson [14] proposes a method based on MC that places sampling points at the center of each *active cell* (a cell crossed by the isosurface), and connects them to sampling points in adjacent cells. This procedure resembles the Cuberille method [15] and generates a dual of the MC mesh. A related technique is dual-contouring [17] which has been shown to generate higher quality meshes than MC. Bruin et al. [11] extended Gibson's technique to couple a gradient descent iteration to the mesh post-processing step, which reduces the distance between the mesh and the *real* isosurface. Dual MC techniques are also studied by Nielson [27] who proposes a method that leads to a polygon mesh surface which is the mathematical dual of a modified form of the MC surface called the "MC Patch" surface.

Tzeng [36] proposes a post-processing step in which small edges are collapsed to eliminate poorly shaped triangles. Labelle and Shewchuk [18] propose warping of the sampling grid to eliminate poorly-shaped tetrahedra *before* their construction. By carefully chosing the amount and direction of warping, together with the lattice in which the surface is extracted, they are able to prove lower bounds on tetrahedra quality. Raman and Wenger [32] propose an extended MC lookup table and a snapping technique that modifies scalar values at grid vertices to improve triangle quality. The Macet algorithm [13] transforms active edges to places that improve the quality of the output mesh. Common to the solutions described above is the idea that moving the sampling grid points [13, 18, 36] or the generated cut points [11, 14, 32] results in a significant improvement of triangle quality. Most of these approaches, however, are guided by intuition. The framework of edge groups introduced in this work provides evidence as to the reason that these methods work.

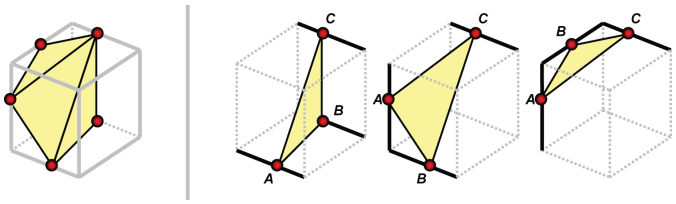


Fig. 2. The quality of each triangle generated by Marching Cubes depends on the combination of edges that generate the triangle.

3 Edge Groups

In this section we introduce edge groups to understand and analyze the triangle quality generated by MC. The connectivity of cut points of a given cell in MC is determined by the sign of the function f at cell vertices, and is used to define a piecewise triangular surface within each cell. We call the point in which the isosurface crosses the cell edges a *cut point*, and the associated edge an *active edge*. In this work we are interested in the quality of each triangle. We chose as quality metric the radii-ratio, which computes the ratio of incircle to circumcircle, normalized to lie between zero and one; an equilateral triangle has maximum quality one [30].

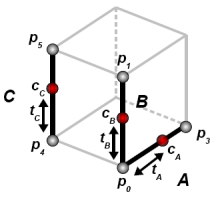
The quality of triangles generated inside a cubic cell is hard to analyze. Each cell generates a set of triangles (up to five in MC), and the quality of each triangle is dictated by the position of its vertices, some of which are shared among adjacent triangles. Our analysis is based on the fact that *the triangles MC generates are not arbitrary: their endpoints are necessarily on edges of a fixed-size cubic cell*. This led us to look at the different ways that three edges can define a triangle in MC. In Figure 2 we show a case that produces three triangles. The shape of each triangle is directly related to the *edge configuration* of each triangle: the three active edges that define a triangle in a MC case.

We represent the position of each cut point c_i along the active edge by a single parameter, $t_i \in [0,1]$. The quality of the triangle set for a given MC case can be given by a function $q : \Re^n \to \Re$, which takes n parameters (positions of cut points) to calculate the mesh quality, where n varies from three to twelve (the number of active edges in a cubic cell, which changes from case to case). As previously observed, the behavior of this high-dimensional parameter space can be quite complicated [18].

3.1 Edge Groups for Cubic Cells

Because of the many symmetries of the cubic cell, different edge configurations can define the same triangle. Such symmetries are easy to consider in the cube where all edges have same size and right angles between adjacent edges. We combine symmetric configurations into a group by partitioning the configurations under the equivalence classes: two edge configurations are equivalent if their representation can be transformed into each other by the symmetries of the cubic cell.

An *edge group* represents a set of equivalent edge configurations that can be formed. The quality of an edge group is defined by the quality of the worst triangle generated in this group. For example, if the edge group is formed by edges A, B and C, then the cut points along the edges are defined as:



$$\begin{aligned} c_A &= p_0 + t_A(p_3 - p_0) \\ c_B &= p_0 + t_B(p_1 - p_0) \\ c_C &= p_4 + t_C(p_5 - p_4) \end{aligned} \qquad (1)$$

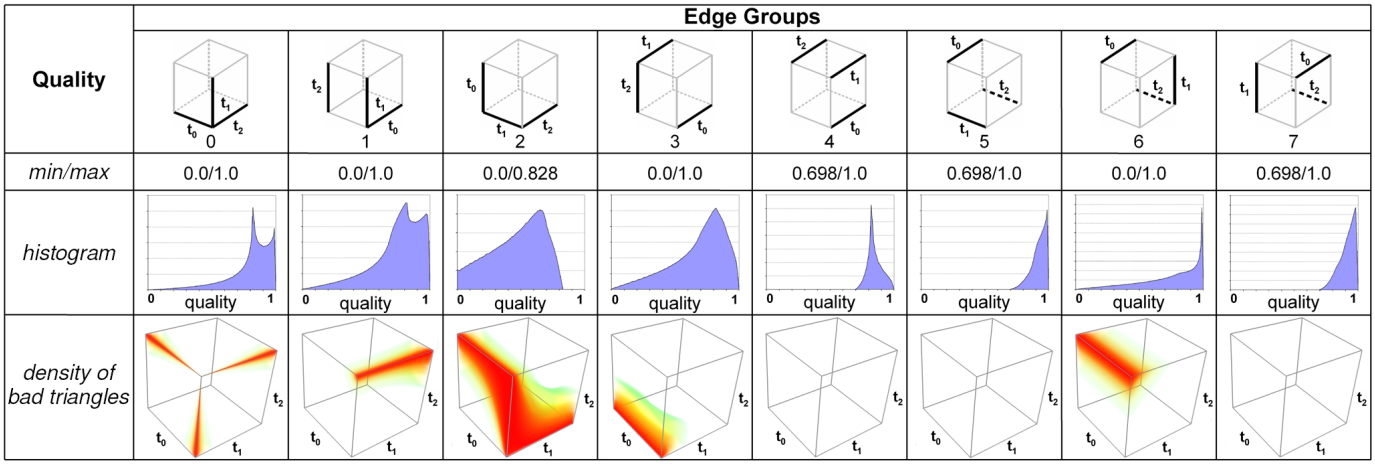


Table 1. Quality bounds, quality histograms and volume renderings of triangle quality for each edge group of the cubic cell. The quality is measured as the radii-ratio of the triangle. Even though the minimum quality of edge groups 0, 1, 2, 3, and 6 indicates that the cubic cell can produce degenerate triangles, the quality histograms show that most of the triangles generated by each edge group have good radii-ratios. Each point in the volume represents a triangle, with red triangles approaching degenerate ones. Volume renderings for edge groups 4, 5 and 7 appear empty because they generate no bad triangles.

By definition, t_A, t_B and t_C vary in the interval $[0,1]$. Each vector $\vec{v} = (i,j,k)$ in the parameter space Θ spanned by t_A, t_B and t_C corresponds to a triangle defined by vertices c_i, c_j and c_k, respectively (see inset). We associate a quality measurement to each vector $\vec{v}$ by generating the cut points c_i, c_j and c_k and computing the quality (radii-ratio) of the resulting triangle. Any vector $\vec{v}$ inside Θ can be mapped to cut points along the edges A, B and C, which guarantees a continuous mapping between the parameters i, j, and k and a quality value. This mapping allows searching for minimum and maximum values of q, which indicate the worst and best triangles of each edge group.

Any triangle generated by MC can be categorized into one of eight edge groups. Figure 1 shows the *complete* list of edge groups, since the triangles generated by *any* table (no matter how connections are performed inside each cell) will fall in one of these cases. Edge groups 5 and 6, for example, do not occur in the original MC table, since they only appear as components of the complement of case 6 (see Figure 4). Quality statistics for each edge group is obtained by numerically evaluating the function q along several domain points. Minimum and maximum values are validated with a non-linear optimization routine implemented in MATLAB. Results obtained for all edge groups are in Table 1. Five edge groups on the cubic cell (groups $0,1,2,3$ and 6) can produce degenerate triangles. However, quality histograms show that most groups generally create good triangles. In Section 4, we exploit this property to improve triangle quality. Table 1 also shows distinct behavior across edge groups. Edge groups 4, 5 and 7 do not produce degenerate triangles and have the same lower bound quality – which is resultant from three triangles (one per case) with the same geometry. Group 2, in particular, cannot generate equilateral triangles. Most importantly, it is the only group with non-zero density of degenerate triangles. This is clearly illustrated by the volume renderings in the third row of Table 1. All other groups produce degenerate triangles only on edges or corners of the parameter space, while edge group 2 seems to have a non-zero volume of the parameter space producing degenerate triangles. This means that simply removing edge group 2 from Marching Cubes cases will significantly reduce the probability of generating bad triangles. The relationship between the geometric configurations of edges and the resulting quality is discussed below.

4 Improving Triangle Quality based on Edge Groups

The insight provided by edge groups can be used to improve the quality of the extracted mesh. Some observations that motivate our approach include:

- Each edge group has a different probability to generate bad triangles, as illustrated by the histograms of Table 1.
- All edge groups generate well-shaped triangles. The combination of parameters ($\vec{v}$) which results in good triangles, however, differs significantly among edge groups.
- Most edge groups can generate bad triangles. This seems to be related to the number of shared vertices in each edge group, since the proximity of the isosurface to a shared vertex may result in cut points close to each other.

With these in mind, we describe the design of two methods to improve triangle quality. The first method tries to reduce the probability of generating a bad triangle by eliminating particular edge groups from MC cases, noting that most MC cases admit different triangulations.

The second method is based on reducing the number of shared vertices among edges, since they seem to be directly related to the overall quality of an edge group. This was observed in [13], where a method to transform active edges and increase the distance among cut points was proposed. Although their claim was supported solely by experimental results, there is no formal evidence that the method in fact improves triangle quality, or the quality bounds of the resulting mesh. In Section 4.2, we use edge groups to explain the effectiveness of the method. In addition, we present an improved version of the algorithm that increases significantly the quality of the extracted mesh. As we discuss, this improvement was largely informed by the analysis of edge groups.

4.1 Changing the MC Table

In MC, a triangle is exactly defined when only three cut points are found inside each cell. When more than three cut points occur (in a single connected component), there is more than one way to connect these points to form a triangulation. The decision on how to connect the cut points impacts which edge groups are associated to the triangles generated. By changing the connectivity of the triangulation stored in the original MC table, and *replacing* edge groups associated to the triangles of some table entries, we can try to remove from the table groups that have a higher probability of generating low-quality triangles, as the edge group 2 of Table 1. In Figure 3, we show a box plot of the occurrences of edge groups across isosurfaces. We also show the plot for the 1000 worst triangles of these isosurfaces. Notice that edge group 2 dominates the bottom end of the quality distribution, experimentally confirming our intuition.

These results suggest the removal of edge case 2 in each entry of the

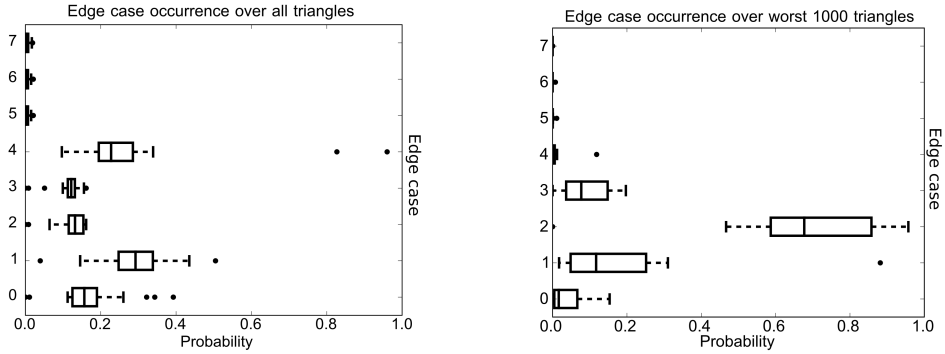


Fig. 3. Box plots show the edge group distribution for 30 different datasets collected by [7]. The left plot shows the overall distribution of edge cases for the datasets, and the right plot shows the distribution of the worst 1000 triangles. Notice that case 2 dominates the bad triangles.

MC table. Obviously, removing this edge group will cause the edge groups of the other triangles in that particular entry to change. Even though the edge groups of adjacent triangles may be replaced by an edge group with a higher probability of generating bad triangles, edge case 2 is responsible for the majority of the worst cases. Exchanging edge group 2 out for a better one in expense of exchanging a group with good quality for a slightly worse one has a positive impact on the overall quality of the entry in the case table. This simple modification results in a significant improvement in the mesh quality, reported in Section 5. An example of removing edge case 2 is illustrated in Figure 4 for three MC cases.

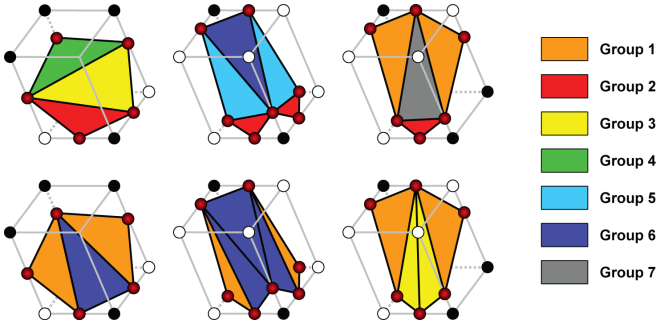


Fig. 4. Replacing cases in MC. The top row shows the original triangulation in 3 MC cases (case 5, *complement* of case 6 and case 11, respectively [20]), while the bottom row shows the modified connectivity. The reconnection of the cut points removes the edge group 2 from these cells, reducing the probability of generating low quality triangles.

4.2 Improving *Macet*

Macet [13] is a technique that improves triangle quality by modifying the inner computation of MC. It is based on the observation that, by moving the location of grid points in MC, they separate cut points that are too close to each other. If done carefully, this can improve triangle quality. They observed that MC generates better triangles when

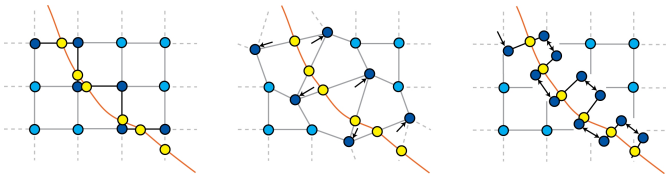


Fig. 5. Intuition behind Macet: small changes to grid vertices positions (left) may improve triangle quality. Moving vertices along the gradient (center) or along tangential paths (right) improve triangle quality.

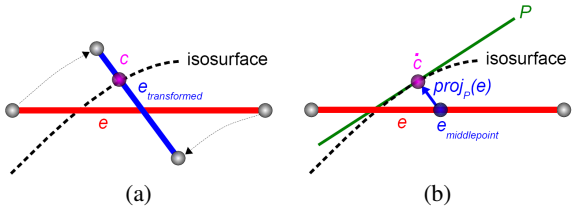
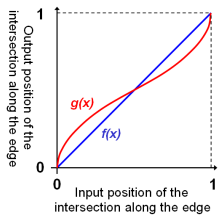


Fig. 6. Edge transformations are redefined as a projection of the edge midpoint onto the plane P tagent to the isosurface. This allows the use of edge group analysis to measure the quality bounds of the transformed edges.

active edges are locally perpendicular to the isosurface, and proposed two ways to move edges defined between grid points (see Figure 5). Similar conclusion is obtained by looking where edge groups generate bad triangles. Tables 1 and 5 show that most edge groups can produce degenerate triangles. This happens because an edge group generates badly-shaped triangles when they have two edges incident to a common grid point, which leads to cut points very close to each other.

Although effective, Macet lacks theoretical guarantees on the improvement of the triangle quality. Edge group analysis helps validate the intuition behind Macet. By applying the edge transformations discussed in Macet over edge groups, we can measure the quality of the resulting triangles. We redefine Macet as a projection $proj_P(e)$, which corresponds to the projection of the midpoint of the edge e onto the plane P tangent to the isosurface. The projection results in the closest point $\dot{c}$ in P to the midpoint of e, as illustrated in Figure 6. The projection $proj_P(e)$ is equivalent to the transformation of the edge e when P is a good approximation to the isosurface around e.



The main conclusion derived from the analysis of the quality histograms of edge groups is the need to move cut points away from the corners of shared vertices. This creates a tradeoff between triangle quality and isosurface fidelity: by moving the intersection points from their actual position we are potentially increasing its distance to the isosurface. On the left side of Figure 7 we illustrate the suggested movement for cut points in edge groups 0, 1 (first row), 2 and 3 (second row). Observe that in edge group 2 we have one edge that has two suggested movements, but in opposite directions. Such opposite movements become even more common when we consider the several edge groups that a given edge may belong (right side of Figure 7). In this figure, we show a cut vertex that belongs to 7 different triangles and consequently different edge groups. The opposing movements suggest that cut points are to be moved towards the center of each edge. (A similar movement, but for a different goal, was proposed in the Discretized Marching Cubes

Dataset	Original MC Table				Modified MC Table			
	(original)	+ Macet	+ Displ.	+ Macet + Displ.	(original)	+ Macet	+ Displ.	+ Macet + Displ.
Silicium	0.00157 (1.14462)	0.19913 (1.75566)	0.03329 (1.14462)	0.26031 (1.75566)	0.01792 (1.14462)	0.33889 (1.75566)	0.09806 (1.14462)	0.34459 (1.75566)
Engine	0.00042 (0.84591)	0.23566 (1.62972)	0.01778 (0.84591)	0.29275 (1.62972)	0.0008 (0.84591)	0.24611 (1.62972)	0.02443 (0.84591)	0.29275 (1.62972)
Bonsai	0.00013 (0.54739)	0.1029 (1.09751)	0.01017 (0.54739)	0.16867 (1.09751)	0.00152 (0.54739)	0.1029 (1.09751)	0.03138 (0.54739)	0.19497 (1.09751)
Lobster	0.00088 (0.42495)	0.11659 (0.83487)	0.026 (0.42495)	0.22984 (0.83487)	0.00153 (0.42495)	0.18981 (0.83487)	0.03416 (0.42495)	0.24716 (0.83487)

Table 2. Results comparing the worst triangle quality in four datasets using the original and our modified MC table. For each table we list in the first line, the worst triangle obtained using MC, Macet, MC and the Displacement edge transformation, and Macet with the Displacement edge transformation. The second line reports the maximum difference in absolute isovalue between the mesh and the actual isosurface. Observe the improvement in triangle quality between columns with red text (change in the MC table), and in the blue columns (improvements in the Macet algorithm.

algorithm [22].) Forcing intersection points to be placed at the middle of the edges is unnecessary in some situations. Therefore, we apply a symmetric non-linear displacement using a modulation function. On the x axis we describe the original position of the intersection (normalized between 0 and 1, from start to end of the edge), and on the y axis the normalized position of the displaced cut point. We chose the popular *gain* function described in [31] to modulate the intersection value. This allows us to apply non-linear displacements, and specially to apply a greater displacement to places they are more needed: closer to the endpoints of the edge. The current gain function we used has parameter value equal to 0.3 which we found experimentally to perform well. We evaluated our displacement modulation function using both MC and Macet, and report results in next section.

5 Results

Table 2 contains the main results we obtained, including both the changes to the MC table and the modifications to the Macet algorithm. Additionally, Figure 8 visually compares enlarged sections of Silicium, Engine, Bonsai, and Lobster datasets obtained with the original MC table, Macet using the original table, and Macet using displacements and the new MC table proposed. We separate the discussion of these results in different sections to better present the impact of each modification.

5.1 Changing the MC Table

The impact that the simple change in the MC table can be verified by looking at the two red columns in Table 2. Observe that in three of the datasets the improvement of the quality of the worst triangle is at least of an order of magnitude. This modification improves the quality of the worst triangles at the expense of reducing the quality of triangles with good-aspect ratio, which represents a good trade-off. Since the new and old MC tables generated different triangulations, it is instructive to estimate how each triangulation is distant from the actual isosurface. We implemented a procedure that distributes a fixed number of sample locations over each triangle of the mesh, samples the volume at these locations and accumulate the maximum difference (negative and positive). These results are reported in Table 3. Results show that the difference is nearly the same using both tables.

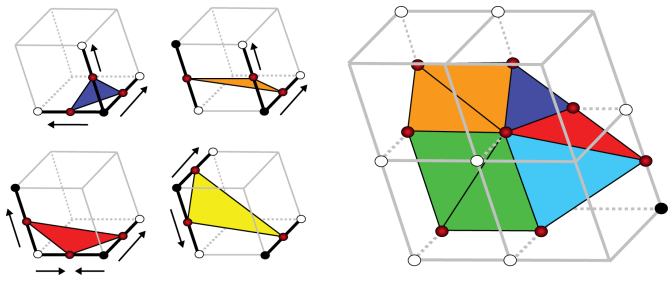


Fig. 7. Suggested movement of vertices for edge groups 0, 1, 2 and 3 (left). Since an edge is shared across several cells (right), a conflict occurs that suggests that a cut point should be moved to the center of an edge.

Dataset	Isolevel	MC	MC (new table)
Silicium	140.5	-14.508/30.783	-10.667/30.077
Engine	49.5	-24.611/33.988	-24.611/33.988
Bonsai	49.5	-31.737/23.050	-31.737/23.050
Lobster	30.5	-21.967/21.308	-21.967/21.345

Table 3. Minimum and maximum isovalue differences obtained using the old and new MC tables.

5.2 Improving *Macet*

Table 4 shows that the edge transformations indeed improve triangle quality. Macet imposes conditions that limit the application of edge transformations based on the neighborhood of the scalar field. Here such constraints are not being taking into consideration, since we are considering each edge case individually. The bounds presented assume that the function can be approximated by a linear function inside a cell, which might require an adaptive variation of MC. This edge group analysis shows that the minimum quality of the transformed edge groups is 0.272 in edge group 2, which corresponds to a triangle with minimal internal angle of 18°.

The results displayed in Table 2 allow for several considerations on the improved Macet algorithm. Here we include several combinations that span the alternatives introduced by this work. Results were separated by which MC table was used (original or modified). For each dataset, we list in the first row the triangle quality of the worst triangle for the original MC and Macet algorithms, the MC algorithm using only the new displacement edge transform, and the improved Macet algorithm using the new transform as well. The second row contains the maximum absolute distance to the isosurface. We highlight the improvement we obtain comparing the old Macet using the old MC table, with the Improved Macet using the new table (blue columns). The improvement on the quality of the worst triangle is substantial in the Bonsai, Lobster and Silicium datasets, being closer to twice as better. The Engine dataset shows also an improvement but not as substantial due to its noisy nature. The second line of results display the maximum distance between the mesh and the isosurface. Observe that, unlike our point raised when introducing the displacement modulation of cut points, it does not change this maximum distance. In other words, although locally the distance to the mesh may increase due to this displacement, the cut point involved in the maximum distance value is not affected by this operation.

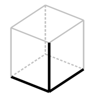
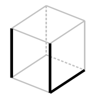
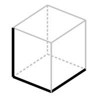
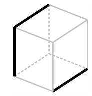
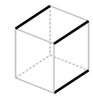
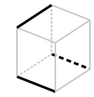
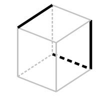
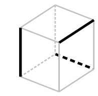
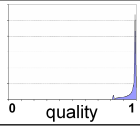
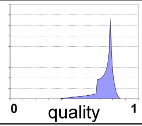
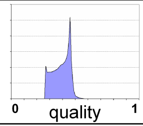
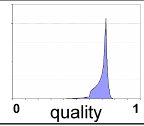
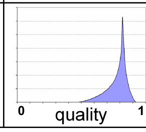
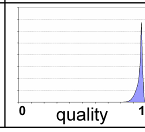
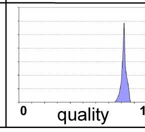
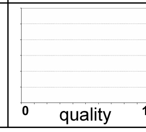
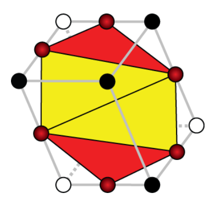
Quality	Edge Groups							
	 0	 1	 2	 3	 4	 5	 6	 7
min/max	0.828/1.0	0.395/0.865	0.272/0.598	0.417/0.791	0.464/0.93	0.788/0.989	0.756/0.881	0.978/1.0
histogram								

Table 4. Quality bounds and histograms for the transformed edge groups (of Macet) of the cubic cell. We do not show density of bad triangles because there are no degenerate triangles in these situations.

6 Discussion

The state of the art in quality isosurface mesh generation, to the best of our knowledge, are the works of Schreiner et al. [33] in advancing front triangulation, Dey and Levine [12] based on Delaunay refinements, and Meyer et al. [21], which combines techniques from computational geometry with dynamic particle systems to sample an isosurface appropriately and then reconstruct the triangulation. They report worst triangle quality ranging from 0.02 to 0.4 in the generated meshes, which is generally comparable to the results obtained with our improved Marching Cubes variant reported in Section 5. Note, however, that in every experiment we performed, our minimum triangle quality was above 0.19. More importantly, our method retains most of the performance advantages and simplicity of Marching Cubes: in particular, we could take advantage of acceleration structures such as octrees [37], interval trees [10] or span space methods [19]. On the other hand, our method is intrinsically based on Marching Cubes, and, as it is, incapable of generating meshes that adapt to the features in the surface such as curvature.



One limitation of the analysis we have performed is that it is not always possible to remove undesirable edge groups from cases in the MC table. In particular, there exist MC cases that must always have at least one triangle from edge group 2, such as the one shown in the inset. One possible way to circumvent this might be to dynamically decide which triangulation to use in the case of such convex polygons based on the barycentric coordinates of the cut points. This is an exciting avenue of research. More generally, our analysis of edge groups could be used to guide dynamically generated MC tables [5].

Edge groups can be identified for any spatial subdivision that define cut points along edges of its cells, such as methods based on MC, like Marching Tetrahedra [6,9] and Marching Octahedra [8]. As stated before, the analysis of edge groups provides quality bounds for the triangles generated in each cell, and thus for the quality of the polygonization method itself. For example, tetrahedral cells with different shapes are obtained in the Delaunay triangulation of a BCC (body centered cubic) grid and from the Coxeter-Freudenthal-Kuhn triangulation [23] (or simply the Kuhn triangulation) subdivision scheme. The analysis of edge groups in tetrahedral cells is more involved than in cubic cells because we have edges with different sizes, and with varying internal angles (as opposed to fixed-size edges and only right angle adjacencies among edges in the cubic cell). Our case analysis here followed an exhaustive procedure to identify all different edge groups. We found four different cases for the BCC cell, and nine for the Kuhn tetrahedron. The quality of the triangles generated for these cells depends on the shape of the tetrahedron that each scheme uses. Although this is an intuitive conclusion, the analysis of the tetrahedron edge groups shows quantitatively the effectiveness of each cell in generating well-shaped triangles (see Table 5). Observe that the BCC edge groups have superior quality if compared to the Kuhn counterparts. This is a direct consequence of the elongated shape of the tetrahedron resulting from Kuhn triangulation, which favors the generation of badly-shaped triangles. These preliminary results indicate that edge groups can help understand the triangle quality generated by MC-like algorithms in more general lattices, but further study is necessary to better understand the tradeoffs involved.

One important issue we also have not pursued here is the influence of the underlying interpolant in the generation of MC case tables and their edge groups, and, consequently, on the quality of the generated meshes. Nielson [26] provides one of the few MC tables in the literature that respect a particular interpolation kernel. Smoother kernels will generate different reconstructions, and so we should use different case tables. We believe a more careful study is necessary to fully understand the issues involved, and leave that for future work.

7 Conclusions

We introduced edge groups, which help explain the triangle quality of meshes generated by MC and other marching methods. The formulation shares the same combinatorial essence of MC case analysis, but takes a different angle by examining how edges are combined to form triangles. We identified that one edge group was responsible for most of the worst triangles generated by MC. With a simple reformulation of the entries to avoid that group, we produced a new MC table that generates better triangle meshes. To our knowledge there has been no discussion on a criteria to guide the way that the connectivity of the triangulation should be defined to improve the quality of the mesh. Nielson [25] proposes a connection table which leads to isosurfaces that are locally functions (each vertex star can be represented as a height field relative to one of the coordinate planes). There are also many works on extended tables to deal with ambiguity problems, but none of them have focused on the triangle quality problem. Our work leads to MC tables that produce better triangles than existing implementations.

Edge groups also serve as a convenient framework to study strategies to improve mesh quality. In this paper, we revisited the edge transformations proposed in Macet and obtained improved quality bounds that are competitive with the current state-of-the-art. The edge group formulation helps understand algorithms that compute intersections on fixed cell lattices. We expect improvements to be possible in other algorithms through similar analyses.

Acknowledgments. We would like to thank the anonymous reviewers for comments that helped us to substantially improve this paper. The work of Carlos Dietrich is supported by a CNPq scholarship. Carlos Scheidegger is supported by an IBM Ph.D. fellowship. João Comba is supported by CNPq grant 485853/2007-0. This research has also been funded the Department of Energy SciDAC (VACET and SDM centers), the National Science Foundation (grants CNS-0751152, CCF-0528201, OCE-0424602, CNS-0514485,

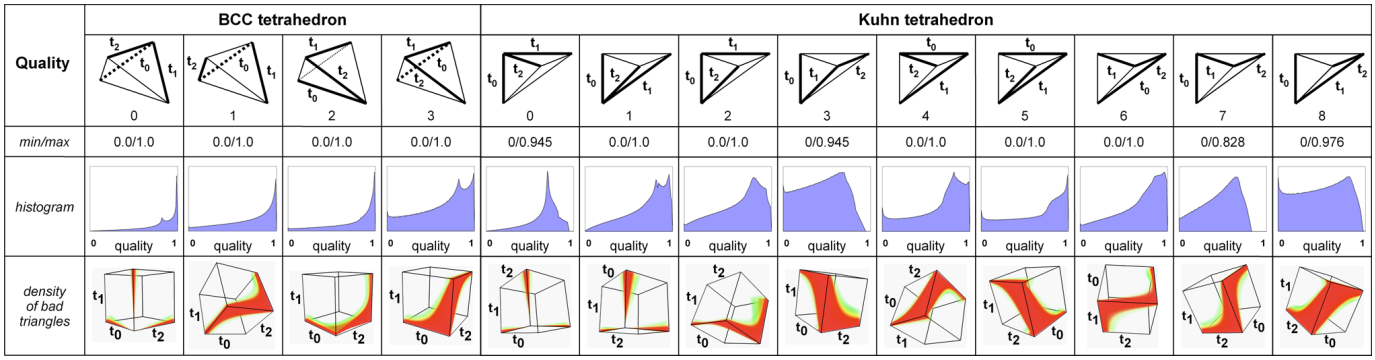


Table 5. Quality bounds, quality histograms and volume renderings of triangle quality for the edge groups of BCC (left) and Kuhn (right) tetrahedral cells. Histogram and density of bad triangles are created in the same way of Table 1.

IIS-0513692, CCF-0401498, OISE-0405402, CNS-0551724), and IBM Faculty Awards (2005, 2006, and 2007).

Reproducibility. The results from this paper are fully reproducible. The datasets are all publicly available, and the code used to generate the results is open-source. For details, please visit `http://www.sci.utah.edu/~cscheid/vis2008/edge_groups`.

REFERENCES

[1] L. Balmelli, C. J. Morris, G. Taubin, and F. Bernardini. Volume warping for adaptive isosurface extraction. In *IEEE Visualization 2002*, pages 467–474, 2002.

[2] D. C. Banks, S. A. Linton, and P. K. Stockmeyer. Counting cases in substitope algorithms. *IEEE Transactions on Visualization and Computer Graphics*, 10(4):371–384, 2004.

[3] D. C. Banks and P. K. Stockmeyer. *Debruijn Counting for Visualization Algorithms*, chapter Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration. Springer-Verlag, 2008.

[4] P. Bhaniramka, R. Wenger, and R. Crawfis. Isosurfacing in higher dimensions. In *IEEE Visualization 2000*, pages 267–273, 2000.

[5] P. Bhaniramka, R. Wenger, and R. Crawfis. Isosurface construction in any dimension using convex hulls. *IEEE Transactions on Visualization and Computer Graphics*, 10(2):130–141, 2004.

[6] J. Bloomenthal. Polygonization of implicit surfaces. *Computer Aided Geometric Design*, 5(4):341–355, 1988.

[7] H. Carr, B. Duffy, and B. Denby. On histograms and isosurface statistics. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1259–1265, 2006.

[8] H. Carr, T. Theußl, and T. Müller. Isosurfaces on optimal regular samples. In *Symposium on Data Visualisation 2003*, pages 39–48, 2003.

[9] S. L. Chan and E. O. Purisima. A new tetrahedral tesselation scheme for isosurface generation. *Computers & Graphics*, 22(1):83–90, 1998.

[10] P. Cignoni, C. Montani, E. Puppo, and R. Scopigno. Optimal isosurface extraction from irregular volume data. In *VVS*, pages 31–38, 1996.

[11] P. W. de Bruin, F. Vos, F. H. Post, S. F. F. Gibson, and A. M. Vossepoel. Improving triangle mesh quality with surfacenets. In *MICCAI 2000*, pages 804–813, 2000.

[12] T. K. Dey and J. A. Levine. Delaunay meshing of isosurfaces. In *IEEE Shape Modeling and Applications 2007*, pages 241–250, 2007.

[13] C. A. Dietrich, C. Scheidegger, J. Schreiner, J. L. D. Comba, L. P. Nedel, and C. T. Silva. Edge transformations for improving mesh quality of marching cubes. *IEEE Transactions on Visualization and Computer Graphics*, 2008.

[14] S. F. F. Gibson. Constrained elastic surface nets: Generating smooth surfaces from binary segmented data. In *MICCAI 1998*, pages 888–898, 1998.

[15] G. T. Herman and H. K. Liu. Three-dimensional display of human organs from computed tomograms. *Computer Graphics and Images Processing*, 9(1):1–21, 1979.

[16] G. T. Herman and J. K. Udupa. Display of 3d digital images: Computational foundations and medical applications. *IEEE Computer Graphics and Applications*, 3:39–46, 1983.

[17] T. Ju, F. Losasso, S. Schaefer, and J. Warren. Dual contouring of hermite data. In *ACM SIGGRAPH 2002*, pages 339–346, 2002.

[18] F. Labelle and J. R. Shewchuk. Isosurface stuffing: fast tetrahedral meshes with good dihedral angles. *ACM Transactions on Graphics*, 26(3):57, 2007.

[19] Y. Livnat, H.-W. Shen, and C. Johnson. A near optimal isosurface extraction algorithm using the span space. *IEEE Transactions on Visualization and Computer Graphics*, 2(1):73–84, 1996.

[20] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *ACM SIGGRAPH 1987*, pages 163–169, 1987.

[21] M. Meyer, R. M. Kirby, and R. Whitaker. Topology, accuracy, and quality of isosurface meshes using dynamic particles. *IEEE Transactions on Visualization and Computer Graphics*, 12(5), 2007.

[22] C. Montani, R. Scateni, and R. Scopigno. Discretized marching cubes. In *IEEE Visualization 1994*, pages 281–287, 1994.

[23] D. W. Moore. Simplicial mesh generation with applications. Technical Report TR92-1322, Cornell University, Dept. of Computer Science, 1992.

[24] T. S. Newman and H. Yi. A survey of the marching cubes algorithm. *Computers & Graphics*, 30(5):854–879, 2006.

[25] G. M. Nielson. Mc*: Star functions for marching cubes. In *IEEE Visualization 2003*, 2003.

[26] G. M. Nielson. On marching cubes. *IEEE Transactions on Visualization and Computer Graphics*, 9(3), Sep 2003.

[27] G. M. Nielson. Dual marching cubes. In *IEEE Visualization 2004*, pages 489–496, 2004.

[28] G. M. Nielson, A. Huang, and S. Sylvester. Approximating normals for marching cubes applied to locally supported isosurfaces. In *IEEE Visualization 2002*, pages 459–466, 2002.

[29] S. Owen. A survey of unstructured mesh generation technology. In *7th International Meshing Roundtable*, pages 239–267, 1998.

[30] P. P. Pebay and T. J. Baker. A comparison of triangle quality measures. In *10th International Meshing Roundtable*, pages 327–340, 2001.

[31] K. Perlin and E. M. Hoffert. Hypertexture. *ACM SIGGRAPH 1989*, 23(3):253–262, 1989.

[32] S. Raman and R. Wenger. Quality isosurface mesh generation using an extended marching cubes lookup table. *Computer Graphics Forum*, 27(3):791–798, 2008.

[33] J. Schreiner, C. Scheidegger, and C. Silva. High quality extraction of isosurfaces from regular and irregular grids. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1205–1212, 2006.

[34] J. Shewchuk. What is a good linear finite element? interpolation, conditioning, anisotropy and quality measures. Preprint, 2002.

[35] G. M. Treece, R. W. Prager, and A. H. Gee. Regularised marching tetrahedra: Improved iso-surface extraction. *Computers & Graphics*, 23:583–598, 1999.

[36] L. Tzeng. Warping cubes: Better triangles from marching cubes. In *European Workshop on Computational Geometry*, 2004.

[37] J. Wilhelms and A. V. Gelder. Octrees for faster isosurface generation. *ACM Transactions on Graphics*, 11(3):201–227, 1992.

[38] B. Wyvill, C. McPheeters, , and G. Wyvill. Data structures for soft objects. *The Visual Computer*, 2(4):227–234, 1986.

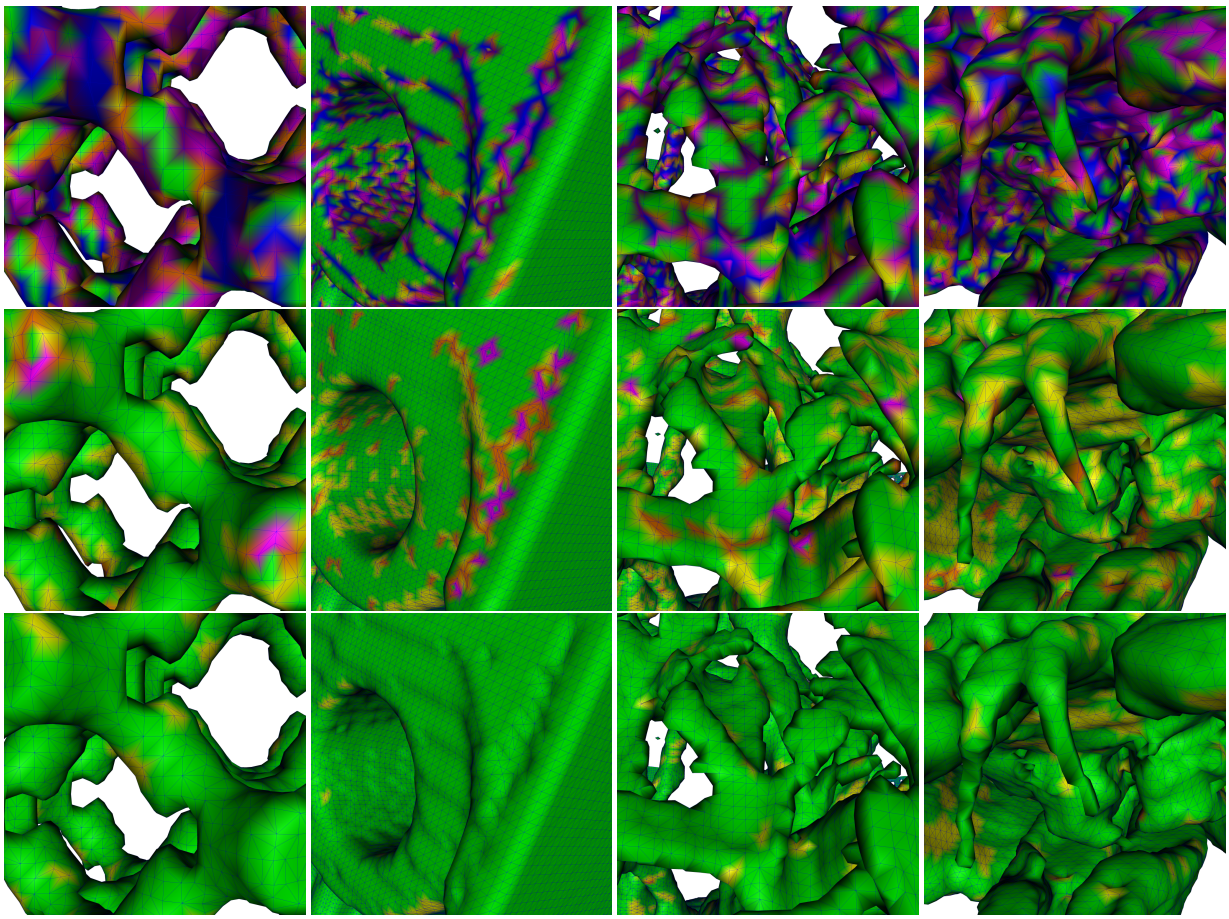


Fig. 8. From left to right, comparisons on enlarged sections of Silicium, Engine, Bonsai, Lobster. First row: MC using original table. Second row: Macet using original MC table. Third Row: Macet using displacements and the new MC table. Green triangles are good, purple triangles are bad. Observe that there is a clear improvement in the overall triangle quality in our proposal (bottom row images).

APPENDIX

The following lemma presents a proof on the number of different edge groups that can define a triangle in a cubic cell.

Lemma 1: There are exactly 8 **different edge groups in a cubic cell**.
Proof: The proof follows a simple enumeration argument. We look at edge configurations based on the number of shared vertices, which can be 0 (three independent edges), 1 (two edges connected, 1 independent) or 2 (three edges connected). Edge configurations are unchanged by rotations and reflections. We assume that $e_0 < e_1 < e2$. Consider now each case individually, using the value associated to each edge in the planar representation of the cell described in Figure 9.

- case 0 (0 intersections): Assume that $e_0 = 0$, which gives us a configuration $(0, e_1, e_2)$. Since e_1 and e_2 must not be connected, this rules out values 1, 2, 3 and 4. Consider the remaining values 5 through 11. Let $e_1 = 5$, which gives a configuration $(0, 5, e_2)$. Similarly, this rules out $e_2 \in \{6, 7\}$. The remaining configurations are all valid, $(0,5,8), (0,5,9), (0,5,10)$ and $(0,5,11)$, and are described in the first row of Figure 9. The same argument is used to create the remaining edge configurations in this case, and they are all displayed in Figure 9. Each configuration is associated to an edge group, which is displayed in a yellow background in the center of each edge configuration. Eleven configurations are identified, but due to symmetries they correspond to only three edge groups of Table 1: 4, 5 and 7.

- case 1 (1 intersection): Assume that $e_0 = 0$ and $e_1 = 1$, which gives us a start configuration $(0, 1, e_2)$. Since e_2 must be disconnected, edge values 2 through 6 are discarded. The remaining values are all valid, $(0,1,7), (0,1,8), (0,1,9), (0,1,10)$ and $(0,1,11)$, described in the case 1 box of Figure 9. Five configurations are enumerated, but due to symmetries they correspond to only three three edge groups of Table 1: 1, 3 and 6.

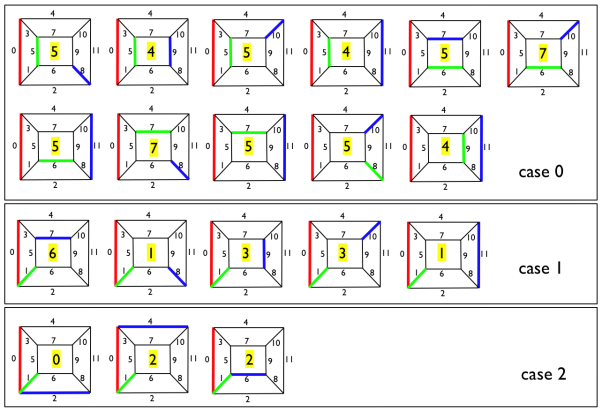


Fig. 9. Cases used to enumerate edge groups in a cubic cell.

- case 2 (2 intersections): Assume that $e_0 = 0$ and $e_1 = 1$, which gives us a start configuration $(0, 1, e_2)$. Since e_2 must be connected to either e_1 or e_2, edge values from 7 to 11 are discarded. Edge values 3 and 5 are discarded since they create an invalid configurations (planar edges). The remaining configurations are valid: $(0,1,2), (0,1,4)$ and $(0,1,6)$, described in the case 2 box of Figure 9. Three configurations are enumerated and reduce to symmetries to two edge groups of Table 1: 0 and 2,

In total we have 8 edge groups (3 in case 0, 3 in case 1 and 2 in case 2), as stated in the lemma.