Consiglio Nazionale delle Ricerche

A boundary representation for extracting sharp surfaces from regularly-gridded 3d objects

R. E. Loke

IIT TR-19/2005

Technical report

Settembre 2005

Istituto di Informatica e Telematica

A boundary representation for extracting sharp surfaces from regularly-gridded 3d objects

Robert Edward Loke

***Abstract*— Geometry extraction from volume data is important in many applications. On a regular 3d grid, current approaches do not consistently preserve object details such as sharp corners and edges of 26-connected objects. We describe a boundary representation in which we geometrically constrain the connectivity, so that such details can be maintained. Application of our model for object surfacing compares favorable to current surfacing methods.**

***Index Terms*— Curve, surface, solid, and object representations**

I. Introduction

In surface rendering, object boundaries are visualized by first extracting a geometric model of isosurfaces or segmented object regions in a volume and then by rendering the model. Surface rendering and modeling have innumerable applications, such as finite element analysis and computational fluid dynamics, in different application areas. In CAD for e.g. gaming, designed objects must be efficiently represented in order to reduce the enormous amount of triangles which is required to visualize entire scenes. Surface rendering is nowadays also often used for volume visualization. One advantage of surface rendering is that, independent from the existence of hardware accelerators, e.g. Pfister et al. (1999), it is fast if compared to volumetric visualization techniques such as the one by Levoy (1988). Furthermore, it enables a highly interactive analysis by "flying" through and around the data in a region of interest. Software libraries such as OpenGL (Open Graphics Library) or VRML (Virtual Reality Modeling Language) provide interfaces for surface visualization.

Current challenges in surface rendering are to design algorithms which

- are general enough and valid/satisfactory for both 6- and 26-adjacency relations of objects,
- detect and preserve object features like sharp corners and edges, irrespective of the object orientation, and
- can be applied to manifold as well as to non-manifold objects.

A traditional concern in algorithm design is the efficiency of surface representation without deformations and aliasing effects.

In order to obtain the geometric model of boundaries in a volume, surfaces can be built by approximation techniques or by surface construction. Volume boundaries can be approximated using, for example, splines [3], [4], wavelets [5], [6], digital geometry [7], [8], parametric surfaces including nonuniform rational b-spline—NURBS—patches [9]–[11], implicit surfaces [12]–[15], and other [16], [17]. Implicit surfaces are connected, closed and can easily be optimized, allowing a fast rendering. However, the main drawbacks of approximation techniques are that, depending on the number of points, their computation is slow and that their shape is difficult to control, especially around characteristic features like e.g. sharp object edges. Because we will only consider surface construction in the remaining of this paper, we want to note that, after surface construction, approximation surface models could still be built starting from polygon models [12], [15]. Current approaches to surface construction can be categorized into isosurfacing and boundary triangulation. Isosurfacing methods such as Marching Cubes and Discretized Marching Cubes (DMC) were originally designed in order to visualize isosurfaces in volume data. These methods are of great practical use because they work for arbitrary data. For example, even for isolated points and lines a surface is built. However, the original versions did not have a mathematical background; only later this background was provided (see next section). Boundary triangulation methods fit the geometry through the boundary of an object using sets of possible object/background configurations. Known boundary representations are Morgenthaler surfaces and simplicity surfaces. These methods traditionally have a strong mathematical basis but are not always of direct practical value, i.e. they are not applicable to arbitrary datasets because configurations can occur in the data which do not yield any surface output.

Current methods in both surface construction approaches do not preserve important shape details such as sharp edges and corners. We propose a practical algorithm which exploits an intuitive boundary representation in order to triangulate object boundaries. In a new discrete boundary model we restrict the connectivity between the boundary points such that object details are maintained. The model is related to the simplicity surfaces of Couprie and Bertrand. Contrary to other methods we do not work directly on the object itself but on its 6-border. In combination with three filters for concave corners this allows us to preserve sharp object details without introducing unwanted aliasing. The algorithm can be applied to arbitrary data in order to extract surfaces.

The structure of the paper is as follows. First we introduce the necessary terminology and elaborate on related research in Section II. In Section III we give an operational definition for object boundaries with concavity and describe the discrete boundary model with its triangulation. In Section IV we conclude and give some directions for future work.

The author is with CNR, IIT, Via Giuseppe Moruzzi 1, 56124 Pisa, Italy. Email: robert.loke@iit.cnr.it

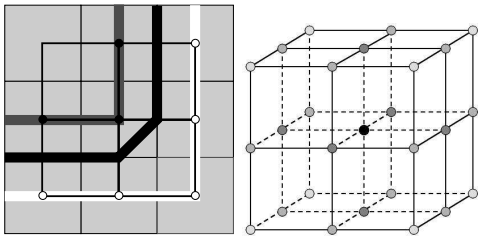

Fig. 1. Left: 2D illustration of a block of 3×3×3 voxels (gray squares): Shown are four cuberilles between nine points (the coordinates of each point are equal to the center of its corresponding voxel) and an object (black dots) within a background (white dots). The fat black line denotes a standard MC isosurface obtained with the DMC implementation. With discrete surface methods, one obtains two surfaces: one for the boundary of the object (fat gray line) and one for the boundary of the background (fat white line). The fi gure shows the surface contours when our method is applied; i.e. sharp at the convex edge and sharp at the concave edge. Right: Adjacency relations and the eight cuberilles which are uniquely defi ned in a 3×3×3 neighborhood. The labels denote 6-, 18- and 26-adjacent points. The point in the center of the neighborhood is in common in all cuberilles.

II. Related research and terminology

There are two main surface construction approaches which convert a volumetric representation into a surface representation: an isosurface generation with a MC type of algorithm [18] that finds a surface after thresholding and detection of intersected cells, or a discrete surface generation, for instance for a binary volumetric representation that results from a segmentation and a connected component labeling [19]. With the discrete surface generation, the vertices of the surface elements are positioned at the voxel centers, while for the isosurface generation, the vertices are on the edges between the voxel centers, at positions where the isosurface intersects the cell between the voxels. Figure 1 clarifies this difference between MC and the discrete surface representation. Cuberilles refer to the cells of the discrete grid between the points which are defined by the voxel centers. Eight unique cuberilles can be referred to in a 3×3×3 voxel neighborhood. The figure also clarifies the notion of n-adjacency. Each point has a 3×3×3 neighborhood with points which are 26-adjacent to the central point from which 6 points have a Manhattan distance of one, 12 points a distance of two, and 8 points a distance of three steps in orthogonal directions. Two points are n-adjacent if they are n-neighbors. The 6-neighborhood (respectively, 18-, 26-neighborhood) of a point at (x, y, z) is comprised by those points for which $|x - a| + |y - b| + |z - c| = 1$ (2, 3), with (a, b, c) arbitrary coordinates. In this paper, we define two points to be n-connected ($n = 6, 18, 26$) if there exists a path between the points such that all subsequent points on the path are maximally n-adjacent one to another. Thus, 6-connected points are also 18-connected and 26-connected, but 18-connected ones not 6, and 26-connected ones not 18 nor 6.

In the last two decades a lot of research has been dedicated to improving the MC data isosurfacing algorithm. Topology improved [20], [21] and efficiency enhanced—in terms of a reduced triangle count—versions [22]–[24] are all based on locally triangulating cuberilles [25]. Other versions decompose the cuberilles into voxels [26] or tetrahedra [27], use boxes instead of cuberilles [28], use polyhedra or polygonal volume primitives instead of triangles [29], [30], use rules instead of a lookup table for cuberille configurations [31], use heterogeneous grids to guarantee topologically coherent surfaces [32], or optimize the search of relevant cuberilles [33], [34]. The DMC algorithm [22] is a hybrid between isosurfaces and discrete surface models. The cell/edge intersections are not found on isovalue positions but at mid-edge positions. This simplifies the surface topology, reduces the number of degenerated triangles and simplifies merging small triangles in larger surface patches (over several cuberilles).

Discrete surface models were introduced in the early eighties [35], [36] before the MC algorithms became popular. They define surfaces through the boundary of an object by considering various boundary voxel configurations which separate the object interior from the background (object exterior). The discrete surface representation does not interpolate between the voxel coordinates of the object boundary and those of the boundary of the background (or those of the boundary of the object interior). This simplifies—similar as the DMC method—object triangulations, avoids degenerated triangles and helps merging triangles into larger patches. A drawback of not interpolating may be a decreased imaging resolution. However, we expect that this will be overcome by future improvements in scanning devices and technology, which will lead to higher data resolutions, and hence will reduce the error. Furthermore, when volumes will get larger in the future, there will be a growing need of faster rendering algorithms and algorithms which minimize the triangle counts of the renditions.

Quite some mathematical ingenuity has been invested in the study of boundary voxel configurations to find out under which conditions a discrete surface has similar topological properties as a regular manifold surface model, i.e. correctly separates the interior from the exterior. This is relevant for thinning and skeletonization [37]–[39], ray casting volumetric objects [40]–[42], and surface construction [43], [44]. The study of these properties is known as digital topology [35], [45] and has resulted in several boundary definitions by Morgenthaler and Rosenfeld [46], Malgouyres [47], Kovalevsky [48] and Couprie and Bertrand [49]. The latter introduced the notion of simplicity surface and showed that this surface has some nice properties: (1) any Morgenthaler closed 26-surface [46] is a simplicity surface; (2) any strong 26-surface [47], [50] is a simplicity surface; and (3) any simplicity surface satisfies the Jordan property, i.e. its complement has two connected components (proof yet to be published).

The definition of a simplicity surface is based on the notion of “simple” points. Simple points are boundary points that are topological redundant and as such *not* part of a simplicity surface. Figure 2 shows a 3×3×3 object with one empty central voxel (the interior point). All points on the corners and edges of the 3×3×3 cell structure are simple points, because removing these points will not change the topology. The resulting 26-surface (i.e. an octahedron) will be the minimal enclosure/cover for the interior (Figure 2b). This is the only “valid” boundary in a mathematical sense. It will be clear that if we want the resulting surface to be the cover of the

complete 3×3×3 cell structure (what in most practical cases would be the desired result) then we should exclude the 26-connected surfaces in those cases where 6- and 18-adjacencies are sufficient (as in our model). However, if we have an oblique or curved surface, then we would like to use the diagonal shortcuts of the 26-adjacency in order to avoid the staircasing of the 6-surface representation. Couprie and Bertrand give several operational definitions for simple points and they prove that a simplicity surface can be built out of only 8 different 2×2×2 voxel configurations [49]; see also Section III-C. On the basis of these configurations it could be easy to define a triangulation method for discrete surfaces. However, the definition of simplicity surface is not of much value from a practical point of view, because in arbitrary data still undefined configurations may occur and because we do not always want to remove the simple points.

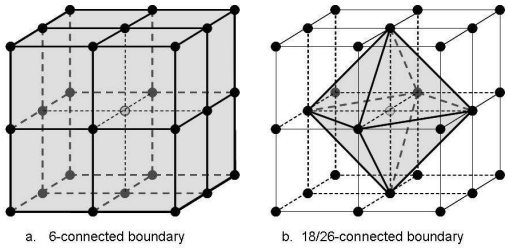

Fig. 2. A 6- and 18-connected boundary. Black/white dots denote whether the voxels which correspond to the points belong to the object/background.

A practical discrete surface generation method which is applicable to arbitrary data and which has a clear mathematical basis like other boundary triangulation methods was proposed by Kenmochi et al. [51] (below we refer to this method as the Kenmochi method). They define the boundary of a discrete solid as the boundary of the set of connected tetrahedra that constitutes the volumetric object. To construct the boundary, the object is first decomposed into a set of tetrahedra, and after removing the "double" surfaces shared by neighboring tetrahedra, the "single" outside faces constitute the overall boundary. They also presented a construction method that directly generates the composite boundary and deals with degenerated cases as dangling edges and folded surfaces. Kenmochi et al. give a slice-by-slice and cell-by-cell construction method that directly generates a correct surface using 14 triangulation patterns for 2×2×2 voxel configurations. The Kenmochi method differs from the discrete surface models which have been described before in that the voxels belong either to the object or to the background (object exterior), and not to the object boundary, to the object interior or to the background. Thus, this method is not a boundary approach but an object approach.

The Kenmochi method is a sound and useful method. However, we can make some interesting observations: the method only works for "fat" objects, i.e. objects which can be decomposed into tetrahedra; although the Kenmochi method maintains sharp corners in case of convex configurations, it generates oblique faces in concave situations due to the fact that the method generates tetrahedra in corners. For instance, Fig. 3 shows alternative triangulations of Kenmochi pattern P7a (the name of the configuration is taken from [51]) which for oblique surfaces is not desired, but which in some cases, such as sharp concave corners, better preserves the object detail.

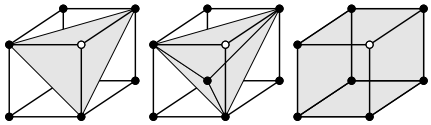

Fig. 3. Three different triangulation patterns which are possible for Kenmochi configuration P7a. The one to the left is applied in the Kenmochi method. Black/white dots denote whether the voxels which correspond to the points belong to the object/background.

III. Discrete boundary representation

A. Definitions

A volumetric representation consists of a three-dimensional grid of voxel positions, where each voxel stores one or multiple values. We limit the discussion here to regular 3d grids that constitute a discrete space with voxels that are either black (the object) or white (the object exterior, i.e. background). Let us assume that a connected component labeling has been run with 26-connectivity for the object and 6-connectivity for the background. From discrete topology [52], [53] we know that the 6-border of a 26-connected object is again 26-connected. Let us define the object boundary as the set of all points (recall from Section II that a point uniquely corresponds to a voxel and that the coordinates of a point are equal to the voxel center) which belong to the object and which have a 6-adjacent neighbor which belongs to the background.

In order to add border points for concave edges and corners which are not included in the object boundary we have defined a detection technique. This technique could already be applied "on" the discrete (voxel) grid, before applying the boundary triangulation. For binary objects such a preprocessing could be quickly applied and, in visualization pipelines, it could be combined with or be integrated in the 3d segmentation or connected component labeling.

We apply three masks in order to detect concavities; see Fig. 4. They operate in the 5×5×5 neighborhood of a point P. In a first pass, the mask at the top is applied in each of the three perpendicular xy, xz and yz planes of P on the discrete grid, in each plane four times to cover all edges. In a second pass, the two masks at the bottom are applied in each of the eight corners of the neighborhood of P. P is included in the boundary if all conditions which are imposed by one of the masks hold. Thus, we first extend the object border with all points at concave edges and then with points at all concave corners. These two passes are necessary because concave edges which are included in the boundary in the first pass may be needed to determine concave corners in the second pass. Points are only included in the boundary if all conditions in one of the masks applies, i.e. if the background forms a convex edge or a convex corner and the object correspondingly forms a concave edge or a concave corner. Concave edges and corners are only included when they are completely adjacent to their convex counterparts in the background. Such a strict

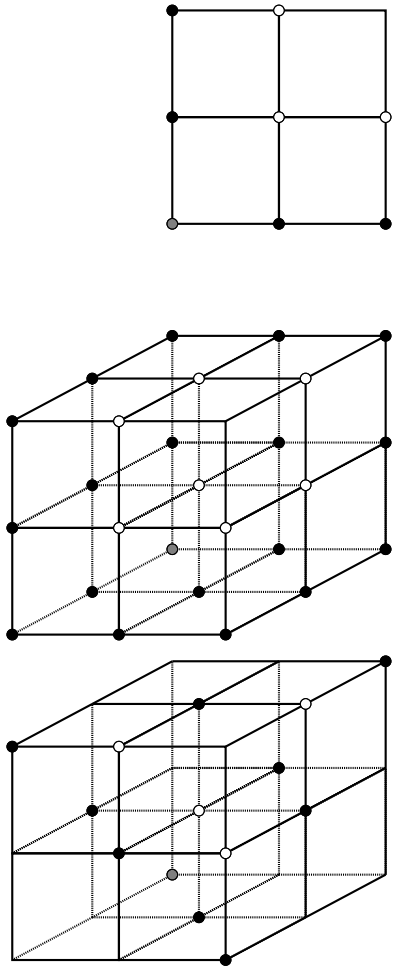

Fig. 4. Three detectors for determining the concave borders of a discrete object. The depicted masks are slided over the object on the 3d grid: in a first pass the mask at the top in order to determine all concave edges; in a second pass the two masks at the bottom in order to determine all concave corners. Any gray point which satisfies one of the depicted mask settings is added to the 6-border of an object. Black (gray) dots denote that the corresponding voxels belong to the boundary (object); white dots denote the background (object exterior and interior). Positions on cuberille corners without dots can be ignored.

definition is needed in order to avoid any unwanted aliasing. Figure 13 shows an example of the processing when we apply the mask at the top to a regular 2D image. The boundary is extended at concavities without introducing any unwanted aliasing.

In the following, boundary refers to the set of all defined boundary points and background to the set of all points which is contained by the regular 3d grid minus the boundary. Thus, the background includes both the object interior and the object exterior.

B. Boundary model

We propose a method to construct surfaces of discrete objects by building surface patches locally at the object boundary, using a matching of the boundary/background in each of the eight unique cuberilles in the $3\times3\times3$ neighborhood of each boundary point. This allows to reduce the number of required triangles because in such a larger neighborhood the patches obtained in each cuberille can be linked and

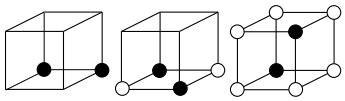

Fig. 5. Restricting connectivity between 6-, 18- and 26-adjacent points (denoted as black dots). Adjacent points are only directly connected if there are no *connectivity shortcuts* (black) on the position of the white dots. Positions on cuberille corners without dots do not affect the adjacency relationship.

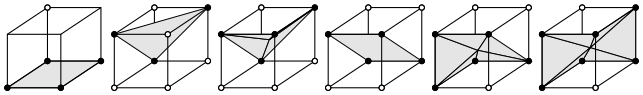

Fig. 6. The six basic object boundaries P1-P6 and their surface mappings. Black and white dots denote that the points belong to the boundary or not. In the configuration to the left, positions on cuberille corners without dots do not affect the boundary connectivity. At least one point at these positions must be from the background.

optimized at the neighborhood center. In order to be able to match the boundary in the cuberilles and to link the patches at the central boundary point in the neighborhood we have developed a new model to define the surface topology. In this model adjacent points are only directly connected if there is no lower-adjacency path possible (Fig. 5). This results in nine topologically different configurations to let us differentiate between all possible object/background topologies (Figs. 6, 7, 8 and 9).

The configurations have been determined by considering all topologically different boundary connectivities in a cuberille, using 6-, 18- and 26-connectivity for objects, and 6-connectivity for the background. In our model, two boundary points can be connected when they are: (A) 6-adjacent, (B) 18-adjacent *and* do not have a 6-adjacent boundary neighbor in common, or (C) 26-adjacent *and* do not have a 6- nor an 18-adjacent boundary neighbor in common (see Fig. 5). When they *do* have a 6- or an 18-adjacent neighbor in common (in case B or C), they should not be connected directly—only indirectly via another neighbor, i.e. a *connectivity shortcut*. The advantage of strictly defining the boundary connectivity in this way is that sharp edges and corners in the boundary can be preserved and correctly modeled.

Based on these definitions, we can build 6-, 18- and 26-connected skeletons or build surface patches for 6- and 18-connected closed curves. Here, we concentrate on the surface mapping of object boundaries. In surface mapping, only 6- and 18-connectivity make sense, because (according to our definition of connectivity) 26-connected boundaries can not yield any surfaces, only skeletons. Hence, a point at the object boundary is at least 6- and at most 18-connected to each other point of the boundary in its 26-neighborhood. Figure 6 shows the six basic boundary configurations. According to our definition of connectivity, the boundaries are 6-connected (P1), 18-connected (P2), 6- and 18-connected (P3, P4 and P5) and, again, 6-connected (P6). Thus, 26-adjacent points are always 6- and/or 18-connected via connectivity shortcuts, e.g. in P4 the 26-adjacent boundary points are 18-connected. Configurations P3, P5 and P6 we call connectivity shortcut configurations, because they 6-connect 18-adjacent boundary points in configurations P2, P4 and P5. Figures 7, 8 and 9

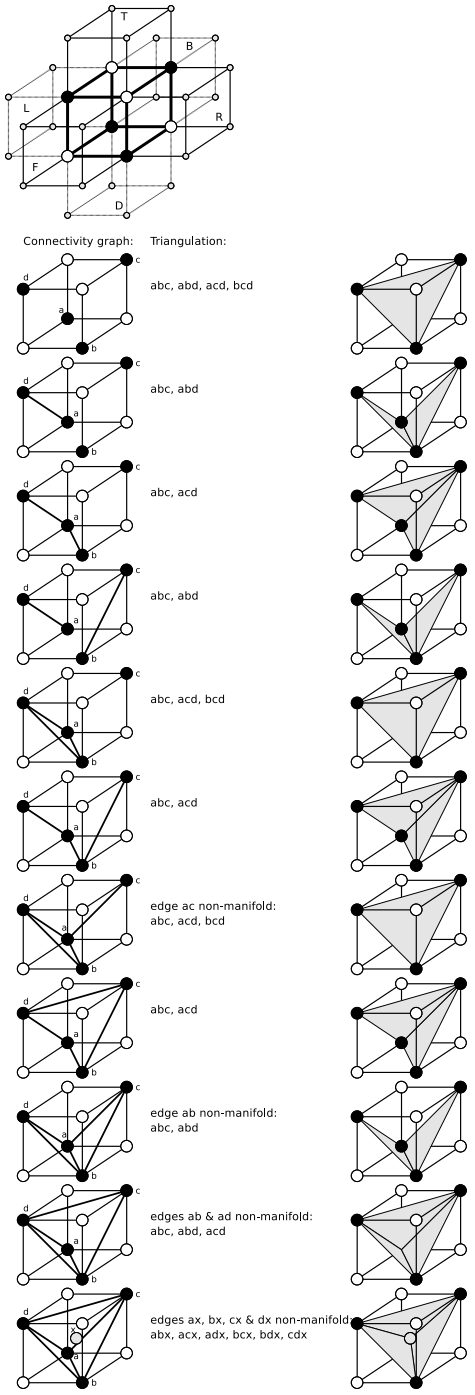

Fig. 7. The 7th object boundary P7 and its surface mappings. The triangulation of the cuberille in the center (see top figure) depends on the connectivity graph which is derived from the six adjacent cuberilles. If an adjacent cuberille is empty (denoted by four small white dots) the corresponding connectivity path can be erased from the graph (see text). If an adjacent cuberille is not empty (i.e. the four small dots would not all be white) the corresponding connectivity path must be maintained in the graph. Once the connectivity graph has been determined the triangulation pattern is fixed.

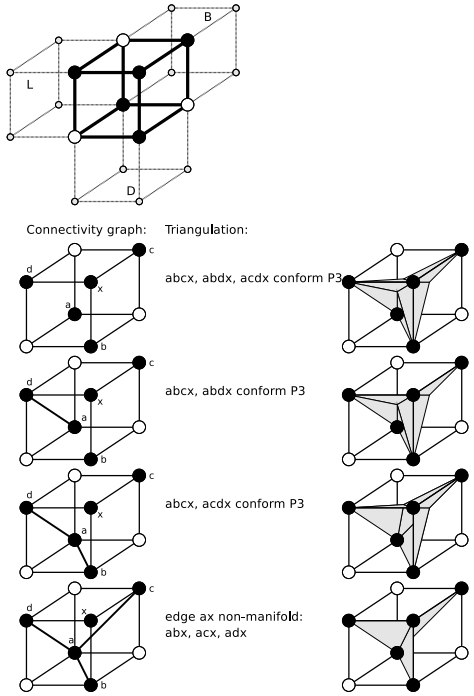

Fig. 8. The 8th object boundary P8 and its surface mappings.

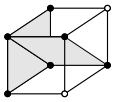

Fig. 9. The 9th object boundary P9 and its surface mapping.

show the three compound boundary configurations which can be derived from Fig. 6 by setting, in P2, P3 and P5, those points which do not affect the boundary connectivity from white to black.

Figures 6, 7, 8 and 9 also show the patches which are used in the surface mapping. For P1, P2 and P4, the smallest possible patches have been used. For the connectivity shortcut configurations P3, P5 and P6, the definition of surface patches is less trivial. Here, we simply use patches with one additional vertex in the patch center, the vertex coordinates being equal to the mean of all boundary points in the cuberille. This is the most trivial way to model the boundary shortcuts. Another, more advanced approach, which can be used to further improve the surface modeling, is described in Section IV. For P7 and P8, the boundary triangulation is not uniquely defined in the cuberille. The surfaces can consist of two, three or four patches and the position of the patches can vary. We found that the surface mapping depends on the six adjacent cuberilles (R, L, T, D, F and B) for P7 and on three adjacent cuberilles (L, D and B) for P8 (see Section III-C for a theoretical explanation). Therefore, we have developed an algorithm which exploits the connectivity of the boundary in the cuberille with the boundary in the adjacent cuberilles. In the algorithm we represent the connectivity between the points inside the cuberille with a graph. This graph we call the *connectivity graph* G. Let G of P7 be $\{ab, ac, ad, bc, bd, cd\}$ and G of P8 be $\{ab, ac, ad\}$; see again Figs. 7 and 8. If an adjacent cuberille is empty (i.e. all four additional points belong to the background) we

can erase the corresponding path from the graph, because then that path does not necessarily has to make part of the surface. For example (see Figs. 7 and 8), if cuberille B is empty we can erase ac from G. For P7 (P8), this can be done for each of the six (three) adjacent cuberilles. After inspecting all cuberilles, G contains only those paths that should be connected in the surface. Figures 7 and 8 show all topologically different surfaces which result from this analysis (totally, there exist 2^6 different connectivity graphs for P7 and 2^3 for P8 but these can be reduced to the depicted 11 graphs for P7 and the depicted 4 graphs for P8). Note that for the cuberille which corresponds to the second, fourth and eighth connectivity graphs two different triangulations are possible. For example, the second can be triangulated with abc and abd or with abc and acd. In order to improve the surface modeling in these ambigue situations the surface fit could be improved by considering additional context information (see Section IV). However, this is not needed to guarantee that we generate Euclidean surfaces at all closed edges in these cases. Also note that not all surfaces are manifold, because edges occur where more than two surface patches meet. For P9, the surface is non-manifold, because it has one edge where three surface patches meet. See Section III-C for a discussion on non-manifold surfaces. In practice, we can for each cuberille already code the "emptyness" with regard to its adjacent cuberilles on the discrete voxel grid in a preprocess such that the triangulation only depends on the boundary in and the byte stored for each cuberille.

C. Topological interpretation and validation

Configurations P1-P8 generate manifold surfaces for *closed* object boundaries and non-manifold surfaces for object boundaries which are not closed such as the endings of a discrete plane. P7 and P8 also generate manifold surfaces for closed objects and P7-P9 generate non-manifold surfaces for complex object boundaries which can be interpreted as multiple boundary crossings. Although the latter surfaces are non-manifold they have the property that they are closed. For P7, the subconfigurations are: 1 for manifold of closed object, 5 and 8 for manifold of closed object boundary, 2, 3, 4 and 6 for non-manifold surfaces of boundary endings, and 7, 9, 10 and 11 for non-manifold surfaces of boundary crossings.

Recently, as already mentioned in the introduction, Couprie and Bertrand (1998) have studied the manifold properties of 26-surfaces. They prove that whether a point is part of a 26-boundary only depends on its $3\times3\times3$ neighborhood, and that eight possible configurations (Figure 10) are sufficient to validate that these points form a so-called simplicity surface and that such a surface in the mathematical sense will be coherent, i.e. closed, oriented and without gaps. Note that the eight closed curve is only defined when the adjacent cuberille in the back is empty. This configuration may correspond to subconfiguration 1, 2, 3, 4, 5, 6, 8, or 10 in Fig. 7. It has two possible triangulations. For P7 (P8) and its triangulation, we also inspect the neighboring cuberilles, but we consider more surface mappings. Interestingly, while they derived their configurations following a theoretical approach, we obtained very similar configurations (Figs. 6 and 7) empirically through looking for efficient surface construction methods for $3\times3\times3$ neighborhoods.

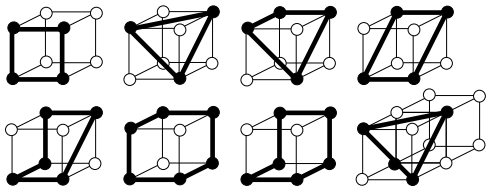

Fig. 10. Simple closed curves for the configurations which make up simplicity surfaces adapted from Couprie and Bertrand (1998).

Although the boundaries which can be obtained with our model seem to be related to simplicity surfaces, there are also differences. For instance, our configurations P8 and P9 and the sharp corner which can be composed from P1 or the sharp corner which can be composed from P7 do not occur in simplicity surfaces. However, whereas in our boundary model we preserve object details such as sharp corners and edges by enforcing the surface through these points, in simplicity surfaces these details are first removed from the boundary. Interestingly, we can infer different triangulation patterns from single closed curves. For example, we can triangulate the second closed curve in Fig. 10 with just one triangle or with three triangles in order to form a sharp corner (the sharp corner which can be composed from P7). Also, the third closed curve can be interpreted as P3 or as P8 and the sixth closed curve can be interpreted as P6 but also as the sharp corner which can be composed from P1. Thus, although we can not give a pure mathematical validation of the additional configurations which occur in our model, we can indirectly validate them by relating them to simplicity surfaces.

The boundary configurations which yield non-manifold surfaces in our model (such as P9 and those in P7 and P8) can not be related to simplicity surfaces. However, the modeling of non-manifold surfaces is important in many applications. In analyses of datasets on regular grids arbitrary configurations may occur where non-manifold modeling may be very useful and important. Also, methods exist for the smoothing, editing and decimating of non-manifold models, see e.g. [54], [55]. A formal definition which proves that our model is coherent for non-manifolds is an important issue for future research.

Importantly, we can show that our model can be applied to arbitrary datasets, because almost all possible boundary configurations are mapped to surfaces. The only configurations which are not mapped to surfaces are depicted in Fig. 11. The configurations to the right are not necessarily mapped by our model, because the mapping depends on the boundary configurations in the adjacent cuberilles. If the adjacent configurations make part of our model (i.e. P3, P5, P6, P8 or P9) then all boundary points in the configurations to the right in Fig. 11 are indirectly connected. If the adjacent configurations make again part of those listed in Fig. 11 then the boundary points can be interpreted as surface endings and can be triangulated with surface patches which are not conform our boundary representation.

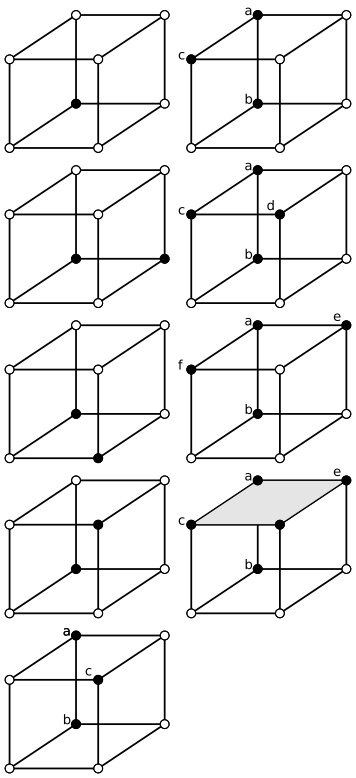

Fig. 11. Configurations which do not yield (complete) surface mappings. Under strict conditions, abc, acd, abe, abf and aef may be interpreted as surface endings and can be triangulated.

IV. Discussion

Preserving object details such as sharp corners and edges is important in many applications. For example, in archaeology scans can be made of ancient parts of objects which are known to belong together, but it may not be known how they should be put together. With the scans represented on regular grids methods could be designed which automatically try to fit together all pieces. Although methods exist for recovering sharp features in geometry descriptions in a post-process [56] and for maintaining sharp features in geometry descriptions based on gradient computations [57], in general surface construction approaches do not consistently preserve object details directly in the surface mapping which results in a much simpler processing. On the contrary, current surface construction approaches often create unwanted aliasing at concavities. With DMC, an object can be rounded at concave corners on one regular grid and can be sharp for convex corners on another grid. With the Kenmochi method, the boundary of an object and the boundary of its background are typically not consistent either. Thus, fitting methods which employ sharp features in geometry descriptions cannot be applied. Obviously, this may hinder fitting together all pieces in the right order. With our method it is possible to consistently preserve both concave and convex sharp edges and corners of object boundaries directly in the surface mapping.

For future work we elaborate on the following extensions. Firstly, as already has been mentioned, we want to operate the boundary triangulation from the centers in $3\times3\times3$ neighborhoods such that output surface patches can be linked in order to optimize the triangle output. Secondly, the surface geometry could be improved by adaptively setting the surface patches which were used for the connectivity shortcut configurations. When the information in a neighborhood increases the additional vertices in these patches could be moved to those locations where the flatness of the surface is optimal. Figure 12 shows some possible patches in which the additional vertices were adaptively set. One way of implementing this mechanism is by moving the additional vertex/vertices in these patches towards the edge/edges.

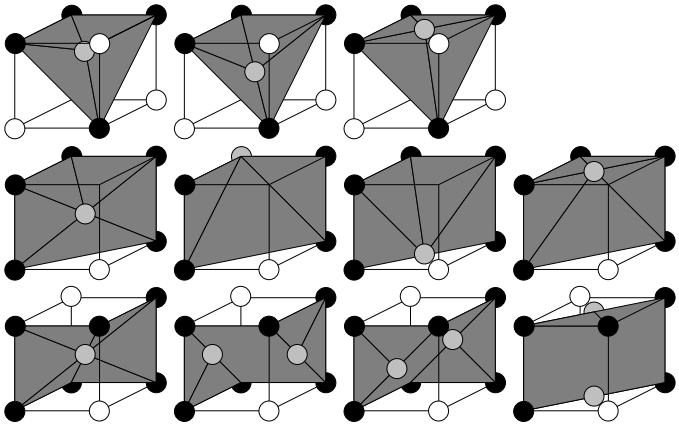

Fig. 12. The connectivity shortcuts (P3, P5 and P6 in Fig. 6) from other viewpoints (left), and different adaptations in which we change the position(s) of the additional vertex (vertices; gray). The best position(s) should be adaptively set, for example by using the boundary information which is available in the other cuberilles in the neighborhood.

References

[1] H. Pfister, J. Hardenbergh, J. Knittel, H. Lauer, and L. Seiler, "The VolumePro real-time ray-casting system," in *Proc. SIGGRAPH 99*, A. Rockwood, Ed., Los Angeles, CA, 1999, pp. 251–260.

[2] M. Levoy, "Display of surfaces from volume data," *IEEE Comput. Graph. Appl.*, vol. 8, no. 3, pp. 29–37, 1988.

[3] D. Salomon, *Computer Graphics and Geometric Modeling*. New York: Springer, 1999.

[4] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes, *Computer Graphics*. USA: Addison-Wesley, 1997.

[5] S. Valette and R. Prost, "Wavelet-based progressive compression scheme for triangle meshes: Wavemesh," *IEEE Trans. Visual. Comput. Graphics*, vol. 10, no. 2, pp. 123–129, 2004.

[6] M. Bertram, M. A. Duchaineau, B. Hamann, and K. I. Joy, "Generalized b-spline subdivision-surface wavelets for geometry compression," *IEEE Trans. Visual. Comput. Graphics*, vol. 10, no. 3, pp. 326–338, 2004.

[7] N. Amenta, M. Bern, and M. Kamvysselis, "A new Voronoi-based surface reconstruction algorithm," *Computer Graphics*, vol. 32, pp. 415–421, 1998.

[8] D. Attali and J. Lachaud, "Constructing iso-surfaces satisfying the Delaunay constraint," in *Proc. 10th Int. Conf. on Image Analysis and Processing*, Venice, Italy, 1999, pp. 382–387.

[9] L. A. Piegl and W. Tiller, "Reducing control points in surface interpolation," *IEEE Comput. Graph. Appl.*, vol. 20, no. 5, pp. 70–74, 2000.

[10] A. Raviv and G. Elber, "Interactive direct rendering of trivariate b-spline scalar functions," *IEEE Trans. Visual. Comput. Graphics*, vol. 7, no. 2, pp. 109–119, 2001.

[11] Z. Xie and G. E. Farin, "Image registration using hierarchical b-splines," *IEEE Trans. Visual. Comput. Graphics*, vol. 10, no. 1, pp. 85–94, 2004.

[12] H. Q. Dihn, G. Turk, and G. Slabaugh, "Reconstructing surfaces by volumetric regularization using radial basis functions," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 24, no. 10, pp. 1358–1371, 2002.

[13] A. P. Witkin and P. S. Heckbert, "Using particles to sample and control implicit surfaces," *Computer Graphics*, pp. 269–277, 1994.

[14] K. G. Suffern and R. J. Balsys, "Rendering the intersections of implicit surfaces," *IEEE Comput. Graph. Appl.*, vol. 23, no. 5, pp. 70–77, 2003.

[15] G. Yngve and G. Turk, "Robust creation of implicit surfaces from polygonal meshes," *IEEE Trans. Visual. Comput. Graphics*, vol. 8, no. 4, pp. 346–359, 2002.

[16] M. Jackowski, M. Satter, and A. Goshtasby, "Approximating digital 3d shapes by rational gaussian surfaces," *IEEE Trans. Visual. Comput. Graphics*, vol. 9, no. 1, pp. 56–69, 2003.

[17] S. Hahmann and G. P. Bonneau, "Polynomial surfaces interpolating arbitrary triangulations," *IEEE Trans. Visual. Comput. Graphics*, vol. 9, no. 1, pp. 99–109, 2003.

[18] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," *Computer Graphics*, vol. 21, no. 4, pp. 163–169, 1987.

[19] R. E. Loke and J. M. H. du Buf, "Interactively visualizing 18-connected object boundaries in huge data volumes," in *Discrete Geometry for Computer Imagery, LNCS 2886*, I. Nyström, G. Sanniti di Baja, and S. Svensson, Eds. Springer, Nov. 2003, pp. 544–553.

[20] A. van Gelder and J. Wilhelms, "Topological considerations in isosurface generation," *ACM Trans. Graphics*, vol. 13, no. 4, pp. 337–375, 1994.

[21] J. O. Lachaud and A. Montanvert, "Continuous analogs of digital boundaries: A topological approach to iso-surfaces," *Graphical Models and Image Processing*, vol. 62, pp. 129–164, 2000.

[22] C. Montani, R. Scateni, and R. Scopigno, "Discretized marching cubes," in *Proc. Visualization '94*, R. D. Bergeron and A. E. Kaufman, Eds., Washington D.C., USA, 1994, pp. 281–287.

[23] H. Müller and M. Stark, "Adaptive generation of surfaces in volume data," *The Visual Computer*, vol. 9, no. 1, pp. 182–198, 1993.

[24] R. B. Shu, C. Zhou, and M. S. Kankanhalli, "Adaptive marching cubes," *The Visual Computer*, vol. 11, no. 4, pp. 202–217, 1995.

[25] L. Chen, G. T. Herman, R. A. Reynolds, and J. K. Udupa, "Surface shading in the cuberille environment," *IEEE Comput. Graph. Appl.*, vol. 5, no. 12, pp. 33–43, 1985.

[26] C. Lin, D. Yang, and Y. Chung, "A marching voxels method for surface rendering of volume data," in *Proc. IEEE Computer Graphics International*, Hong Kong, 2001, pp. 306–313.

[27] A. Guéziec and R. Hummel, "The wrapper algorithm: Surface extraction and simplification," in *IEEE CVPR Workshop on Biomedical Image Analysis*, Seattle, WA, 1994, pp. 204–213.

[28] T. Poston, T. Wong, and P. Heng, "Multiresolution isosurface extraction with adaptive skeleton climbing," *Computer Graphics Forum*, vol. 17, no. 3, pp. 137–148, 1998.

[29] Y. Kenmochi and A. Imiya, "Deformation of discrete surfaces," in *Advances in Digital and Computational Geometry*, R. Klette, A. Rosenfeld, and F. Sloboda, Eds. Singapore: Springer, 1998, pp. 285–316.

[30] H. Yun and K. H. Park, "Surface modeling method by polygonal primitives for visualizing three-dimensional volume data," *The Visual Computer*, vol. 8, no. 1, pp. 246–259, 1992.

[31] D. Karron, "Novel surface rendering and object registration methods for three dimensional medical imaging: The "spiderweb" surface algorithm and the "pointers" technique for integrating multimodal images," Ph.D. dissertation, New York University, Jan. 1993.

[32] X. Daragon, M. Couprie, and G. Bertrand, "Marching Chains algorithm for Alexandroff-Khalimsky spaces," in *Proc. SPIE Vision Geometry XI*, vol. 4794, 2002, pp. 51–62.

[33] Y. Livnat, H. Shen, and C. R. Johnson, "A near optimal isosurface extraction algorithm using the span space," *IEEE Trans. Visual. Comput. Graphics*, vol. 2, no. 1, pp. 73–84, 1996.

[34] J. Wilhelms and A. V. Gelder, "Octrees for faster isosurface generation," *ACM Trans. Graphics*, vol. 11, no. 3, pp. 201–227, 1992.

[35] T. Y. Kong and A. Rosenfeld, "Digital topology: introduction and survey," *Comp. Vision, Graphics and Image Proc.*, vol. 48, pp. 357–393, 1989.

[36] T. Y. Kong and A. W. Roscoe, "Continuous analogs of axiomatized digital surfaces," *Comp. Vision, Graphics and Image Proc.*, vol. 29, pp. 60–86, 1985.

[37] R. Hall, T. Y. Kong, and A. Rosenfeld, "Shrinking binary images," in *Topological Algorithms for Digital Image Processing*, T. Y. Kong and A. Rosenfeld, Eds. Elsevier, 1996, pp. 31–98.

[38] C. Arcelli and G. Sanniti di Baja, "Skeletons of planar patterns," in *Topological Algorithms for Digital Image Processing*, T. Y. Kong and A. Rosenfeld, Eds. Elsevier, 1996, pp. 99–143.

[39] K. Palágyi and A. Kuba, "A 3D 6-subiteration thinning algorithm for extracting medial lines," *Pattern Recognition Letters*, vol. 19, pp. 613–627, 1998.

[40] D. Cohen-Or, A. E. Kaufman, and T. Y. Kong, "On the soundness of surface voxelizations," in *Topological Algorithms for Digital Image Processing*, T. Y. Kong and A. Rosenfeld, Eds. Elsevier, 1996, pp. 181–204.

[41] A. Kaufman, *Volume Visualization*. Los Alamitos (CA), USA: IEEE Computer Society Press Tutorial, 1991.

[42] A. Kodosh, D. Cohen-Or, and R. Yagel, "Tricubic interpolation of discrete surfaces for binary volumes," *IEEE Trans. Visual. Comput. Graphics*, vol. 9, no. 4, pp. 580–586, 2003.

[43] J. K. Udupa, "Connected, oriented, closed boundaries in digital spaces: Theory and algorithms," in *Topological Algorithms for Digital Image Processing*, T. Y. Kong and A. Rosenfeld, Eds. Elsevier, 1996, pp. 205–231.

[44] ——, "Multidimensional digital boundaries," *CVGIP: Graphical Models and Image Processing*, vol. 56, no. 4, pp. 311–323, 1994.

[45] T. Y. Kong and A. Rosenfeld, *Topological Algorithms for Digital Image Processing*. Elsevier, 1996.

[46] D. G. Morgenthaler and A. Rosenfeld, "Surfaces in three-dimensional digital images," *Information and Control*, vol. 51, pp. 227–247, 1981.

[47] R. Malgouyres and G. Bertrand, "Complete local characterization of strong 26-surfaces: continuous analog for strong 26-surfaces," *Int. Journal on Pattern Recognition and Artificial Intelligence*, vol. 13, no. 4, pp. 465–484, 1999.

[48] V. A. Kovalevsky, "Finite topology as applied to image analysis," *Comp. Vision, Graphics and Image Proc.*, vol. 46, pp. 141–161, 1989.

[49] M. Couprie and G. Bertrand, "Simplicity surfaces: a new definition of surfaces in Z^3," in *Proc. SPIE Vision Geometry VII*, vol. 3454, 1998, pp. 40–51.

[50] G. Bertrand and R. Malgouyres, "Some topological properties of discrete surfaces," *Journal of Mathematical Imaging and Vision*, vol. 11, pp. 207–221, 1999.

[51] Y. Kenmochi, A. Imiya, and A. Ichikawa, "Boundary extraction of discrete objects," *Computer Vision and Image Understanding*, vol. 71, no. 3, pp. 281–293, 1998.

[52] T. Y. Kong and A. W. Roscoe, "A theory of binary digital pictures," *Comp. Vision, Graphics and Image Proc.*, vol. 32, pp. 221–243, 1985.

[53] T. Y. Kong, A. W. Roscoe, and A. Rosenfeld, "Concepts of digital topology," *Topology and its applications*, vol. 46, pp. 219–262, 1992.

[54] A. Hubeli and M. Gross, "Multiresolution methods for nonmanifold models," *IEEE Trans. Visual. Comput. Graphics*, vol. 7, no. 3, pp. 207–221, 2001.

[55] P. Cignoni, F. Ganovelli, C. Montani, P. Pingi, C. Rocchini, and R. Scopigno, "Virtual marching cubes: Simple and efficient topological simplification," CNR, Tech. Rep., 2000.

[56] M. Attene, B. Falcidieno, J. Rossignac, and M. Spagnuolo, "Edge-sharpener: Recovering sharp features in triangulations of non-adaptively re-meshed surfaces," in *Proc. Eurographics Symposium on Geometry Processing*, Aachen, Germany, 2003, pp. 62–71.

[57] L. P. Kobbelt, M. Botsch, U. Schwanecke, and H.-P. Seidel, "Feature sensitive surface extraction from volume data," in *Proc. 28th annual conference on Computer graphics and interactive techniques*, 2001, pp. 57–66.

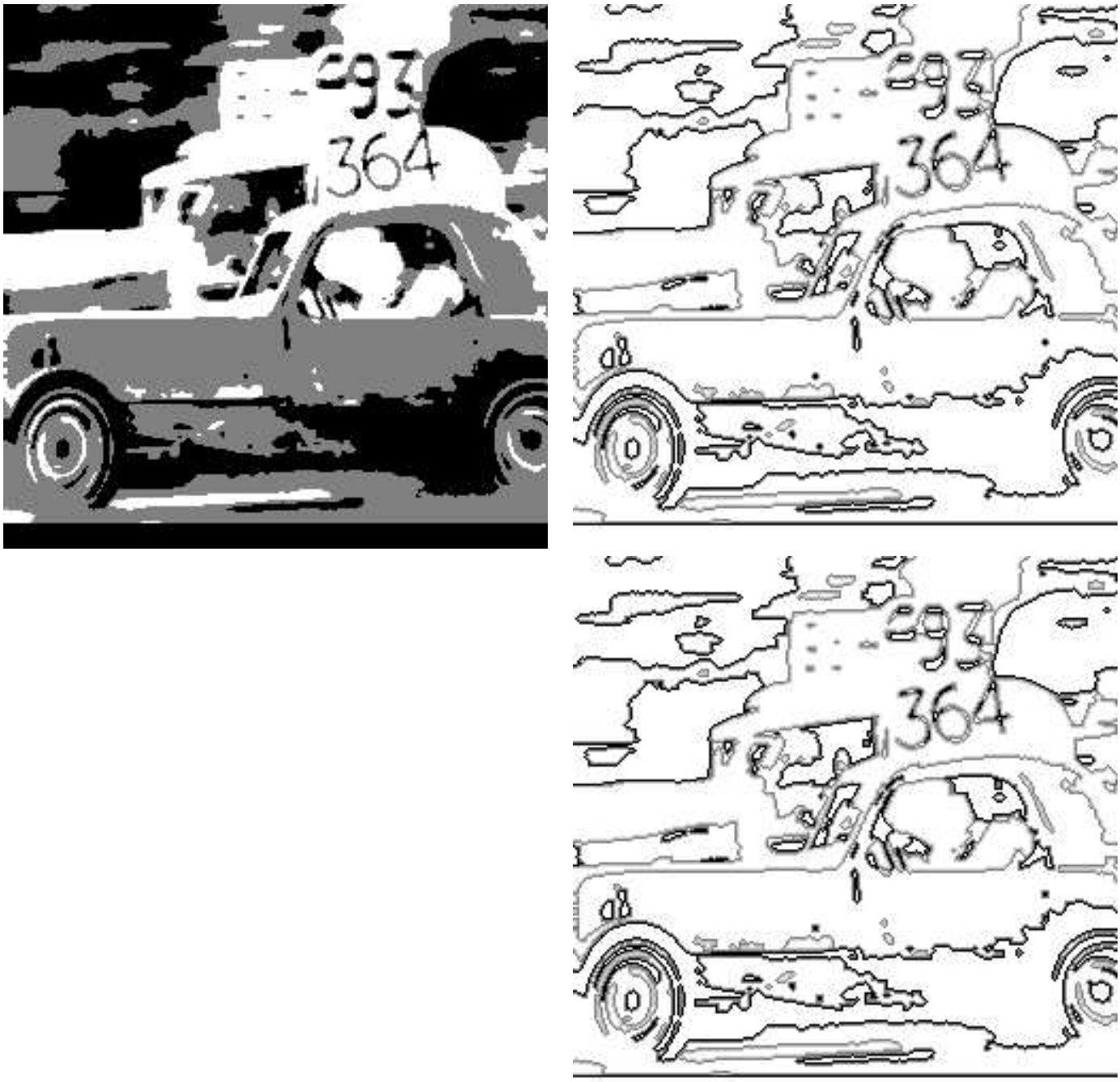

Fig. 13. Cars image: segmented regions, 4-borders of all regions and 4-borders extended with sharp corners.