Usage of 2D Region Similarity For Surface Reconstruction From Planar Samples

Prof. Dr. Ing. Oscar E. Ruiz S., Prof. Dr. Math. Carlos A. Cadavid,
Student Assistants Miguel Granados, Sebastián Peña, Eliana Vásquez
CAD / CAM / CAE Laboratory
EAFIT University, Medellín, COLOMBIA

Abstract

In surface reconstruction from planar slices it is necessary to build surfaces between corresponding 2D regions in consecutive levels. The problem has been traditionally attacked with (i) direct reconstruction based on local geometric proximity between the regions, and (ii) classification of topological events between the slices, which control the evolution of the cross cuts. These approaches have been separately applied with mixed success. In the case (i), the results may be surfaces with over-stretched or unnatural branches, resulting from a local contour proximity which does not correspond to global similarity between regions. In (ii), the consequences from topological events upon the actual surface realization have not been drawn. In this paper an integration of (i) and (ii) is presented, which uses a criteria of similarity between composed 2D regions in consecutive slices to: (a) decide if a surface should actually relate those regions, (b) identify the topological transitions between levels and (c) construct the local surface for the related regions. The method implemented hinders over-stretched and unnatural branches, therefore rendering a surface which adjusts to geometrically-sound topological events. This is a good alternative when the surface reconstructed needs to be topologically faithful (for example in flow simulation) in addition to represent the a rough geometrical space (for example in radiation planning).

Glossary

Π_i, Π_{i+1}	Consecutive cross section planes sampling an object surface. Also apply to the (polygonal) cross sections contained in these planes.
A, B,...	Jordan Curve on plane Π_i, parallel to the xy-plane.
1, 2,...	Jordan Curve on plane Π_{i+1} parallel to the xy-plane.
S_i, S_{i+1}	Sets of Jordan Curves on planes Π_i and Π_{i+1} respectively. S_i={ A, B,...}, S_{i+1}={1, 2,...}.
Area(B)	Area enclosed by contour B, signed according to CCW / CW sense of B with respect to the Z vector.
$\subset$(C_i,C_j)	Containment relation. Contour C_i is contained in contour C_j, and no contour C_k exists such that contour C_i is contained in contour C_k and contour C_k is contained in contour C_j. In this case, it is said that " C_i is a hole inside C_j ". Contours C_i and C_j do not intersect. *Area*(C_i) and *Area*(C_j) have opposite signs.
F_k	Forest Graph whose nodes represent contours on plane Π_k, and edges e of the graph F_k mirror the containment relation $\subset(\ )$. Thus, an edge between contours C_i and C_j, $e(j,i)$, exists iff $\subset(C_i,C_j)$.
$T_{k,i}$	Tree. The *k-th* connected sub-graph of F_i. A tree is either the null tree, or a root node whose descendants are trees: **(**root $child_1$ $child_2$...$child_n$**)**, where

each child may be itself a tree. Examples of trees are: (A), (A B (C D E F)), (1 (2 (3 4 5 (6) 7) 8) 9), ( ). The trees A and (A) are equivalent.

P, Q — A planar polygon with holes. It is represented by a tree with 1 or 2 levels. Unless explicitly stated, it holds that *Area(* P *)* > 0, which means that P is a solid region with holes, traversed in CCW direction according to the plane normal vector.

mg_k — *k-th* Mapping Group "$G_{k,i}$={P_1, P_2,..,.P_L } vs. $G_{k,i+1}$={ Q_1, Q_2,..,.Q_M }", formed by two sets of polygons, P_r and Q_s, from levels *i* and *i+1* respectively. $G_{k,i}$ and $G_{k,i+1}$ conform 2D regions which are similar in shape and therefore assembled in the *k-th* mapping group. The set of all mapping groups between levels *i* and *i+1* is named $MG_{i,i+1}$.

DT_i, DT_{i+1} — DT_i is the 2D Constrained Delone Triangulation of the contour vertices on plane Π_i, such that DT_i contains all edges of contours on Π_i (by inserting additional points on the original edges), such that those edges participate in Delone triangles whose circumcenters lie *inside* the polygonal regions in Π_i. Similarly for DT_{i+1}.

VD_i, VD_{i+1} — 2D Voronoi Diagrams on planes Π_i and Π_{i+1}, for DT_i and DT_{i+1} respectively.

$TT_{k,i,i+1}$ — the *k-th* tetrahedra built by using vertices, edges or faces of DT_i and DT_{i+1}, respectively.

DT, VD — Delone Triangulation and Voronoi Diagram for the contour vertex sets on planes Π_i and Π_{i+1}, considered together.

M — A Piecewise Linear continuous 2-manifold whose cross sections with the planes Π_i are the contour sets S_i.

$M_{i,i+1}$ — The restriction of M to the inter-planar space between Π_i and Π_{i+1}. $M_{i,i+1}$ is a 2-manifold (in general unconnected), whose borders are S_i and S_{i+1}. Building $M_{i,i+1}$ is the goal of the proposed work, since $M = \cup_i M_{i,i+1}$.

covering — A *covering* of a set A is a set of subsets of A whose union is A.

1. INTRODUCTION.

In spite of considerable advance in surface reconstruction from point samples, the subject continues being an open problem, far away from being satisfactorily solved. The goal is to calculate implicit surfaces (2-manifolds in R^3), either smooth or Piecewise Linear (PL), with or without border, connected or unconnected, which interpolate or fit a point data set. The problems may be traced to the fundamental assumption by all algorithms, that the Shannon principle of digital sampling is respected ([Shannon.49]). Its equivalent, the Nyquist principle applied to shape sampling, establishes that the surface recovered is distorted by samples whose spatial sampling interval is larger than half of the smallest detail to be captured. This is a fundamental fact, which can only be overcome in a correct way by collecting massive series of data. Since this is not desirable, the known methods propose heuristics which take advantage of data pattern (slice sampling, grid sampling, etc.) and use this underlying information to overcome the inherently insufficient data set. However, in all cases, limitations remain as a consequence of the mentioned trade off. Methods dealing with cross section data may be geometrical or topological. The geometrical ones concentrate in the building of the surface from metric considerations ([Barequet et al.96], [Bernardini et al.99], [Boissonat et al.93], [Oliva et al.96]), using in many cases Voronoi-Delone (V-D) related algorithms, or minimization techniques. The

topological contributions consider the evolution of the surface to be built as a history of topological events (additions of 0-, 1- and 2-handles) along the sampling axis ([Fomenko et al.97], [Shinagawa et al.91], [Morse.34]). In the cases of grid data (range images) the basic steps of segmentation, meshing and mesh registration have been extensively studied and implemented ([Turk et al.94], [Curless et al.96], [Neugebauer.97]). Research is open to improve or deal with the low statistical quality of the data in the periphery of the images, which affects the whole process.

Methods for unstructured point samples take basically two approaches: (i) Calculation of smooth analytical implicit functions which minimize some energy or potential function ([Turk et al.02], [Carr et al.97], [Morse et al.01], [Bookstein.89], [Bloomenthal.88,97], [Hoppe et al.93]). Once the implicit function is available, one of the most popular algorithms is the Maching Cubes one ([Lorensen et al.87]). (ii) Calculation of PL implicit surfaces using interpolation on the local point set, by using a numerical parameter ([Edelsbrunner.94], [Amenta et al.01]). Approach (i) is extremely elegant, but it is mostly applicable in small data sets, since the model fitting implies matrix operations that become unpractical as the point set becomes large. Approach (ii) computes directly or indirectly the locii of hulls involving the given point set, calculated by using spherical subspaces instead of flat ones. Edelsbrunner et al. compute directly the alpha – shape, while Amenta et al. first compute the 3D Medial Axis Transform (3D-MAT) of the point set, and then infers the goal surface. Both cases are sensitive to irregular sample intervals, as are all known methods.

In the perspective of the present article, it is preferred to exploit known information of the point sample (i.e. the planarity of the cross sections). The works described in [Bajaj et al.96], [Barequet et al.96], [Bernardini et al.99], [Boissonat et al.93] and [Oliva et al.96] act on the total set of contours $\boldsymbol{S_i}$ and $\boldsymbol{S_{i+1}}$. All of them are good surfacing algorithms if the data set fed to them is a qualified one. Our approach is to pre-process and split $\boldsymbol{S_i}$ and $\boldsymbol{S_{i+1}}$ into subsets, such that the algorithms mentioned above is only fed with a sequence of well screened sub-problems. This ensures that over-stretched or unnatural branches are not even tried. In order to proof the concept, we use a domestic implementation on the work of Boissonat & Geiger ([Boissonat et al.93], [Geiger.93]), which is called here the ***BG( )*** algorithm. The $\boldsymbol{M_{i,i+1}}$ surface is the union of the results of a of sequence calls to the ***BG( )*** algorithm, each call being responsible for a well screened sub-problem.

2. BACKGROUND.

The literature survey will concentrate on the algorithms which exploit the planarity of the point samples to build $\boldsymbol{M_{i,i+1}}$ for every two consecutive levels. Algorithms working with general point samples will not be discussed here.

2.1. Topological Evolution of the Cross Sections.

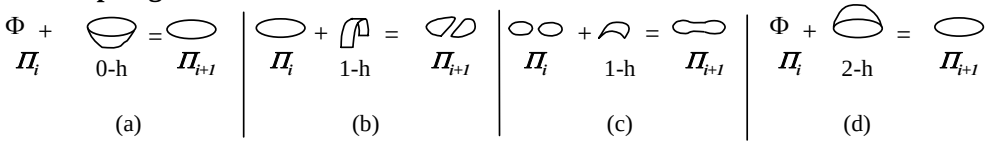


Figure 1. Morse transitions and their effects on contour population (Φ: no contour in level). (a) Creation: add 0-handle. (b) Splitting: add 1-handle, (c) Merging: add 1-handle, (d) Annihilation: add 2-handle.

The population and relations among contours between slices Π_i and Π_{i+1} evolve as a result of the changes in the cross sections of $\boldsymbol{M}$. Based on [Morse.34], other works such as [Fomenko et al.97] and [Shinagawa et al.91] present a set of conceptual operators (addition of 0-, 1- or 2-handles, Figure 1), which cause the evolution of cross sections of $\boldsymbol{M}$. The record of these events is known as Reeb graph ([Shinagawa et al.91b]), and encodes the surface in what respects to *topology*. The *geometry* (position and shape) of the manifold requires what is called a homotopy model to reconstruct the surface. In [Shinagawa et al.91b] the authors construct the Reeb Graph from geometrical considerations such as similarity of *single* contours, therefore completing one part of the geometrical aspect of [Shinagawa et al.91]. However, none of the three contributions actually presents results in constructing the surface. The 2D similarity of *composed* contours is not discussed, and therefore the complications of having topological changes such as addition of 1-handles involving internal holes are not addressed. In [Ruiz et al.02.a]. a preliminar work of surface realization using 0-, 1- and 2-handles is discussed. In [Ruiz et al.02.b] a heuristic algorithm, linear in the number of contours was presented for identifying similar 2D composed shapes. In the present work, rules are given to post-process the sets of 2D similar regions to detect occurrence of 0-, 1- and 2-handles, and to use the similar 2D composed regions to drive the construction of the $\boldsymbol{M_{i,i+1}}$ manifold.

2.2. Geometrical Evolution of the Cross Sections.

Algorithms of the V-D (Voronoi-Delone) variety build surfaces based on local geometric proximity criteria. Here, the most important ones are discussed.

[Barequet et al.96] classify portions of polygonal regions in Π_i and Π_{i+1} into matching and non-matching ones. The matching ones are threaded completing closed paths with edges in Π_i, Π_{i+1} and the space between the planes. Regions spanned by these closed paths are triangulated. The remaining, non-similar parts of contours are also threaded into non-planar closed paths called “clefts”. Clefts have holes, similar to a planar polygonal region. Those holes are integrated to the external edge path by a bridging pattern, therefore leaving clefts without holes, which are triangulated by minimizing the summation of the triangle areas. This approach was not chosen in our work to complement the pre-processing proposed here, because the authors indicate the possibility of selfintersections and the minimization considerably increases the complexity of the algorithm.

The approach by [Boissonat et al.93] (the $\boldsymbol{BG(\ )}$ algorithm) constructs $\boldsymbol{M_{i,i+1}}$ by using contour sets $\boldsymbol{S_i}$ and $\boldsymbol{S_{i+1}}$ to build tetrahedra $\boldsymbol{TT_{k,i,i+1}}$, k=1, 2, ..., whose faces, edges and vertices are inside the polygonal regions implicit in $\boldsymbol{S_i}$, and $\boldsymbol{S_{i+1}}$. The tetrahedra are built by either (a) joining a triangle of $\boldsymbol{DT_i}$ with a vertex of $\boldsymbol{DT_{i+1}}$ (or vice versa) or (b) joining two edges of $\boldsymbol{DT_i}$, and $\boldsymbol{DT_{i+1}}$, if their dual edges in $\boldsymbol{VD_i}$ and $\boldsymbol{VD_{i+1}}$ intersect when projected on the xy-plane. In either case, the tetrahedra are then merged, disappearing their connecting faces, leaving only external faces (triangles), which form $\boldsymbol{M_{i,i+1}}$. The $\boldsymbol{BG(\ )}$ algorithm has

the following characteristics: (a) it is indifferent to distance $d(\Pi_i, \Pi_{i+1})$ between the planes. (b) Tetrahedra are built based on the projections of $\boldsymbol{DT_i}$ and $\boldsymbol{DT_{i+1}}$ on the xy-plane, and *not* on the $\boldsymbol{DT}$ and $\boldsymbol{VD}$. Therefore, these tetrahedra are not the restriction of $\boldsymbol{DT}$ to the mid-space between Π_i and Π_{i+1}. (c) Because of (b), some resulting tetrahedra may be very slanted, since they join very distant (not even frontally faced) regions of polygons on Π_i and Π_{i+1}. (d) The surface built is not guaranteed to be a 2-manifold.

The algorithm presented here (***contour-map-BG( )***) recognizes and uses the advantages of the Boissonat & Geiger approach, while applying it to restricted subsets of the original problem to offset its limitations. A sequence of subsets of the original problem is built to avoid the characteristic (c) mentioned above, which results in counter – intuitive branches being built. In ***BG( )*** the criteria to build tetrahedra between contours of $\boldsymbol{S_i}$ and $\boldsymbol{S_{i+1}}$ is of local geometric proximity between projections of the $\boldsymbol{VD_i}$, $\boldsymbol{VD_{i+1}}$, $\boldsymbol{DT_i}$ and $\boldsymbol{DT_{i+1}}$ on the xy-plane. The actual inter-planar distance may be large, and therefore the tetrahedra so built are not Delone ones, but approximations of them, dictated by rule (b) above.

The algorithm proposed (and implemented) pre-processes the $\boldsymbol{S_i}$ and $\boldsymbol{S_{i+1}}$ contour sets identifying subsets of them which represent composed 2D similar regions, facing each other. Those regions are considered as corresponding to each other under a global shape criterion. These matching regions are entered to the ***BG( )*** algorithm, which is therefore fed only with reasonable contour matches, avoiding counter – intuitive branches or equivalently, highly slanted tetrahedra. Notice that this avoidance is desirable when the surface reconstruction must be faithful both in geometry and topology (for example when the cavity and conduit topology are crucial), as opposed to applications in which only an approximation of the mass enclosed in the surface is required (for example, radiation planning).

3. METHODOLOGY.

A *covering* of the original $\boldsymbol{S_i}$ and $\boldsymbol{S_{i+1}}$ sets is seek, in such a way that subsets of contours with similar 2D shapes are identified. Such sets, which represent a likely shape evolution across cutting planes are then screened for redundancies, and the cases of 0-, 1- and 2-handle additions are classified. The surface algorithms of the V-D variety are then fed with the pre-processed sub-problems, and $\boldsymbol{M_{i,i+1}}$ is built. The methodology followed in this approach is:

(a) **Contour orientation and inclusion calculation.** The polygon sets are pre-processed to ensure a correct sense of their Jordan curves and consistent area signs, inclusion of holes in polygons, etc. (see Glossary). This stage takes the $\boldsymbol{S_i}$ and $\boldsymbol{S_{i+1}}$ sets of Jordan Curves on planes Π_i and Π_{i+1}, and builds the corresponding trees $\boldsymbol{T_{k,i}}$ and $\boldsymbol{T_{m,i+1}}$ (therefore identifying polygons with holes) and forests $\boldsymbol{F_i}$ and $\boldsymbol{F_{i+1}}$.

(b) **Calculation of 2D-similar composed shapes or mapping groups.** Sets of polygons with holes of level Π_i are matched against 2D similar ones on level Π_{i+1}, forming mapping groups. Coverings of the node sets of $\boldsymbol{F_i}$ and $\boldsymbol{F_{i+1}}$ are given by the so obtained mapping groups "$\boldsymbol{G_{k,i}}$ vs. $\boldsymbol{G_{k,i+1}}$" k = 1, 2,... . When a contour A of level i cannot be mapped to any other in level $i+1$, the mapping group "{A**}** vs. Φ" is formed. Regions with negative area *Area( )* are also matched (holes are matched to holes), recalling that holes may have holes inside (which are of course, *solid* regions).

(c) **Post-processing of mapping groups.** From step (b), a contour may appear in several mapping groups. The mapping groups are then purged of repetitions by factoring out sub-shapes. In addition, mapping groups which represent impossible topologies are eliminated.

(d) **Skin construction.** The remaining mapping groups are used for sequential calls to $BG(G_{k,i}, G_{k,i+1})$ k=1,2,...etc. These calls with coverings of the original problem avoid over-stretched faces and branches.

(e) **Topological test.** A Boundary Representation (B-Rep) is built using the triangular facets produced in the calls to the ***BG( )*** algorithm. The B-Rep structure enforces the characteristics of Piecewise Linear 2-manifolds with border, while making explicit the neighborhood relations, normal vector uniformity and borders. Complete, "watertight" 2-manifolds are only a particular case of bordered ones.

Since several of the outlined steps are well known algorithms, only (b) and (c) will be discussed in detail.

3.1. Calculation of 2D-similar composed shapes or mapping groups.

F_i and F_{i+1} are graphs whose nodes are polygons with holes of levels Π_i and Π_{i+1}, respectively. In this stage, from sets of polygons, ***nodes(F_i)*** and ***nodes(F_{i+1})***, mapping groups $G_{k,i}$ vs. $G_{k,i+1}$ k=1,2,...etc. are formed. For each k, $G_{k,i} \subseteq$ ***nodes(F_i)*** and similarly $G_{k,i+1} \subseteq$ ***nodes(F_{i+1})***. The 2D composed region represented by the union of polygons in $G_{k,i}$ is similar to the corresponding 2D composed region represented by the union of polygons in $G_{k,i+1}$. There are N mapping groups between levels i and $i+1$, and so,

$$MG_{i,i+1} = \{ G_{1,i} \text{ vs. } G_{1,i+1},\ G_{2,i} \text{ vs. } G_{2,i+1},\ \ldots,\ G_{N,i} \text{ vs. } G_{N,i+1} \}$$

the $G_{k,i}$ satisfy the following relations:

$$\bigcup_{k=1}^{N} G_{k,i} = nodes(F_i), \qquad \bigcup_{k=1}^{N} G_{k,i+1} = nodes(F_{i+1})$$

$$\text{in general, } G_{k,i} \bigcap G_{j,i} \neq \Phi, \text{ for some } j \neq k \text{ and } G_{k,i+1} \bigcap G_{j,i+1} \neq \Phi \text{ for some } j \neq k$$

Equation 1. Set Relations among Mapping Groups.

Since every polygon must appear in at least one $G_{i,k}$, Φ is used to express mapping groups in which a polygon A of one level is not related to any other in the opposite level ("{A**}** vs. Φ").

Mapping groups of holes must also be determined. Holes are polygonal regions with negative area *Area( )*, which have internal polygons with positive area. Therefore, the holes of holes are *solid* regions.

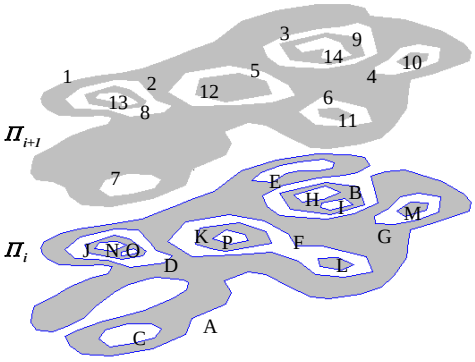
The combinatorial algorithm of testing all possible subsets of ***nodes(F_i)*** and ***nodes(F_{i+1})*** against each other to determine the mapping groups is, of course, too expensive. A linear (in the number of contours) approximation to the solution has been implemented. The algorithm appears in Table 1.

Table 1. Mapping - Group Algorithm

```
function mapping_groups( S1, S2: set of polygons ): set of mapping groups
1  result = {}
2  while (S1 is not empty) or ( S2 is not empty)
3     if ( S1 )
4        seed_polygon = extract_next( S1 );
5        group_n = {};
6        group_m = { seed_polygon };
```

```
7     else
8        seed_polygon = extract_next( S2 );
9        group_n = { seed_polygon };
10       group_m = {};
11    end_if
12    queue = { seed_polygon };
13
14    while ( queue )
15       polygon p = extract_first( queue );
16       for each polygon q in level opposite to p do
17          if (min( Area( p∩q )/Area(p), Area( p ∩ q )/Area(q) ) > threshold )
18             add q to queue
19             add q to group_m (or group_n)
20             extract q from S1 (or S2);
21          end_if
22       end_for
23    end_while
24    result = result + { (group_m vs. group_n) }
25 end_while
```

The strategy used to find an approximation to the mapping groups is similar to the strategy used in calculating the partition of an equivalence relation: all elements *q* related to an element *p*, already in the set, are included in the set (line 16), with *p* eliminated from the search space (line 15). Eventually, all elements take the role of *p*, and their related ones *q* are brought in. The queue of expandable elements grows, stabilizes, shrinks and is eventually exhausted, bringing the end of the iteration in line 13. A new *seed_polygon* is obtained to start the process again (lines 3-11). If no *seed_polygon* is available (i.e. both sets *S1* and *S2* are exhausted) the algorithm finishes. It should be noticed that the present algorithm uses heuristics in order to avoid the combinatorial problem, posed by the exhaustive trial of all sets of 1, 2, 3,...etc. polygons of one level against all equivalent sets of the opposite level. It is based on the hypothesis that two polygons, one in level *i*, and the other in level *i+1*, have a real relation as crosscuts of an object if the projection of one onto the other represents a significant portion (threshold) of its own area. Although this criterion is not a perfect one, the tests conducted showed a great deal of stability of matches and robust performance with respect to the threshold used, yielding very intuitive results. Figure 2 shows an example of results for the contour mapping algorithm.



Mapping Groups

SOLIDS

- {(A C D E F G)} vs. {(1 2 3 4 5 6 7)}
- {(B H I)} vs. {(9 14)}
- {(K P)} vs. {12}
- {M} vs. {10}
- {L} vs. {11}
- {(J N O)} vs. {(8 13)}

HOLES

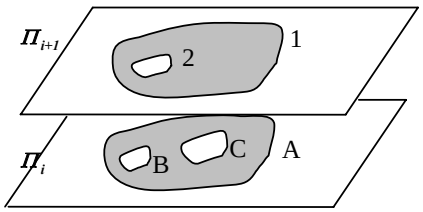
- {C} vs. {7}
- {(D J)} vs. {(2 8)}
- {E} vs. {Φ}
- {(F K L)} vs. {(5 12),(6 11)}
- {(G M)} vs. {(4 10)}
- {N,O} vs. {(13)}
- {P} vs. {Φ}
- {H,I} vs. {(14)}

Figure 2. An example of contour mapping algorithm results.

3.2. Post-processing of mapping groups.

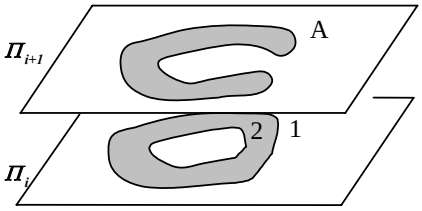
Equation 1 implies that a contour may appear in several mapping groups. The basic reason is that a contour that is a hole region within a solid region participates in a group which maps solid regions to solid regions, and will also appear in the groups which map hole regions to hole regions. This repetition leads to the fact that mapping groups, as output by

the mapping group algorithm cannot be fed to a surfacing module, since polygon holes would be given skin twice.



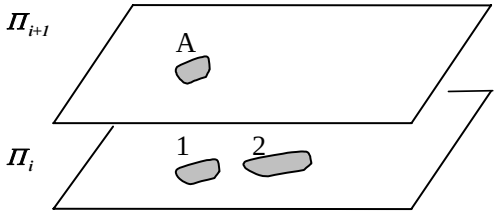
solid: {(A B C)}vs. {(1 2)}
holes: {B} vs. {2}
{C} vs. Φ

Figure 3. Contour death by 2- (or 0-) handle addition.



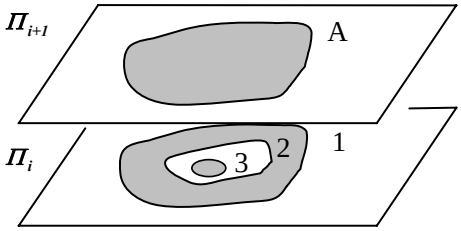
solid: {(1 2)} vs. {A}
holes: {2} vs. Φ

Figure 4. Contour death by 1-handle addition (donnut-croissant transition).



solid: {A} vs. {1}, Φ vs. {2}

Figure 5. Contour death by 2- (or 0-) handle addition.



solid: {(1 2) 3}vs. {A}, {3} vs. Φ
holes: {(2 3)} vs. Φ

Figure 6. Double contour death by 2- (or 0-) handle addition.

In addition, the issues of simultaneous death or birth of nested contours must be solved. A set of definitions follows along with screening rules (presented here without formal proof).

3.2.1. Definition. Level of a contour within a forest.
The level of a contour A in forest ***F*** is the depth of the node A in its corresponding tree in the forest ***F***. For example, if $\boldsymbol{F_i}$ = { (A (B (C (F)) (D (E G (H I)))),... }, then *depth*(A)=0, *depth*(C)=2, *depth*(H)=4 (see Figure 9 and Figure 10).

3.2.2. Definition. Level of a Mapping Group.
The level of a mapping group, *level*(***mg***), is the lowest number in the levels of the contours in ***mg***, dictated by the forests $\boldsymbol{F_i}$ and $\boldsymbol{F_{i+1}}$. For the example in Figure 9 and Figure 10, *level*({ I }vs.{ (7 9) }) is 4, while *level*({ (A B) } vs.{ (1 2))) is 0.

3.2.3. Definition. Ordering of Mapping Groups.
A set of mapping groups admits an ordering $\prec$, dictated by the level of each mapping group. The ordering used here will be descending; first are the leaves, last are the roots of the forests.

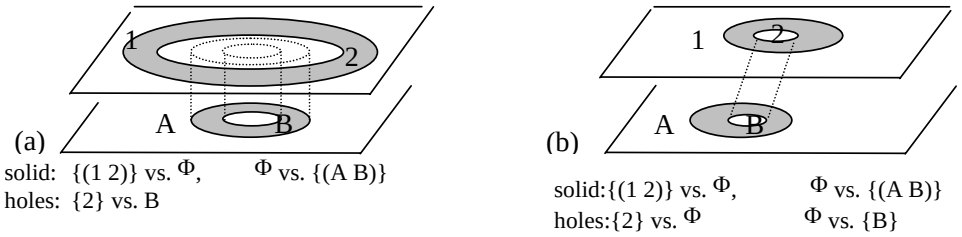


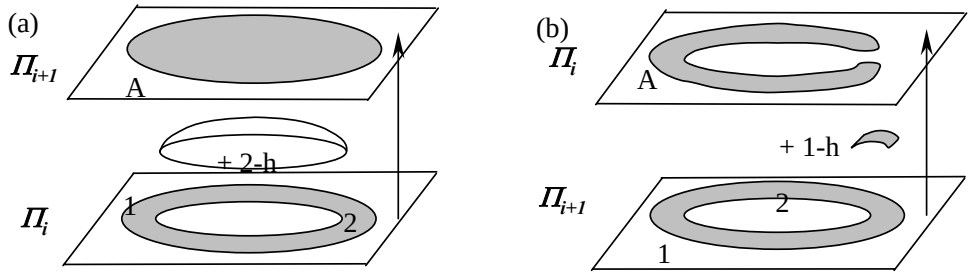
Figure 7. Topologically invalid hole mapping.

3.2.4. Screening of Hole Mappings.

Mapping Groups involving holes require one of these possible actions: (i) If ***mg***={B} vs. {K} (where B is a hole and K ≠ Φ), disregard the mapping group if it is a degenerate byproduct of the mapping group algorithm. (ii) If ***mg*** = {B} vs. Φ, tile hole region B if a 0- or 2-handle transition is identified. This case is detected because [*Area*(B∩Q)/*Area*(B)]≥(1-*threshold*) for some *solid* region Q in the level opposite to the level of B. (iii) If ***mg*** = {B} vs. Φ, and [*Area*(B∩Q)/*Area*(B)] ≤ *threshold* for every *solid* Q in the level opposite to the level of B (which is a hole), neither discard nor tile B, but process it within a 1-handle transition. B itself will not be triangulated, but it is essential to consider it *along* other contours. (iv) Otherwise apply ***BG*(*mg*)** directly.

(i) Topology Condition for Discarding Hole mappings. Holes whose containing solid contour does not map, should be discarded. Figure 7(a) shows that hole mapping group mg_2={ 2 } vs. { B } is a product of the mapping group algorithm. This happens even if the solid mapping group mg_1={(1 2)} vs. {(A B)} never appears due to the fact that polygons (1 2) and (A B) have no overlapping at all. The topology rule expressed here allows to reject mg_2 in this case. The rule has a more obvious ground shown in Figure 7(b): holes B and 2 cannot be mapped if their surrounding solids 1 and A themselves are not mapped, because a self – intersecting surface would be produced.

(ii) Null Hole Mappings in 0- and 2-handles. Figure 8 shows two situations in which mapping groups are exactly the same. However, in the first case (Figure 8(a)) the algorithm should "tile" contour 2, and can do so independently from the relation {1} vs. {A}. This case is one of addition of a 2-handle (the 0-handle case is identical). Surfacing calls would be ***BG(*** {1} **,** {A} ***)*** and ***BG(*** {2} **,** Φ ***)***.



Mapping groups Solid: {(1 2)} vs. A
Hole: {2} vs. Φ
BG( ) calls: ***BG(*** {1} , {A}) BG({2} , Φ***)***

Mapping groups Solid: {(1 2)} vs. A
Hole: {2} vs. Φ
BG( ) calls: ***BG(*** {(1 2)} , {A} ***)***

Figure 8. Different Treatment in Hole Mapping with Φ , according to handle addition nature.

(iii) Null Hole mappings in 1-handle. In contrast to (ii), in Figure 8(b) the mapping group {(1 2)} vs. {A} is indivisible, because a 1-handle is added (a donut - croissant transition). The algorithm presented allows to detect this case in the post-processing of mapping groups, by exploiting the geometric information obtained from the intersection area of the hole and its parent's match on the opposite level, and therefore to correctly process the case (ii) with a 2-handle and case (iii) with a 1-handle, by making the calls ***BG(***{(1 2)},{A}***)*** (see bottom of Figure 8).

3.2.5. Elimination of Redundant Surfacing Calls.

Because of intersections among mapping groups, a process of elimination of those intersections is required. Table 2 shows the corresponding algorithm. At the end of this process, the new set ***MG*** contains non-intersecting mapping groups ***mg***. For each one of them, its surface is built via a call ***BG(mg)***. The order "$\prec$" introduced in the mapping groups guarantees that the simplest ones are given a surface first. The algorithm in Table 2 ensures that they are extracted from more complex ones. Therefore it is guaranteed that a contour is not considered twice, and that no contour is left without participating in a mapping group.

Table 2. Algorithm for Depuration of Redundant Mapping Groups

```
function depurate_map_groups(MG)
1  {
2     result = {}
3     while ( MG )
4        mg = lowest( MG , ≺ )
5        MG = MG – {mg}
6        for each mgi in MG
7           mgi = mgi – ( mgi ∩ mg )
8        end_for
9        append(result ,mg )
10    end_while
11    return( result )
12 }
```

4. RESULTS

4.1. Example A

Figure 9 shows two typical contour sets for levels Π_i and Π_{i+1}. Correspondingly, Figure 10 displays the hierarchical tree and forest graph structures.

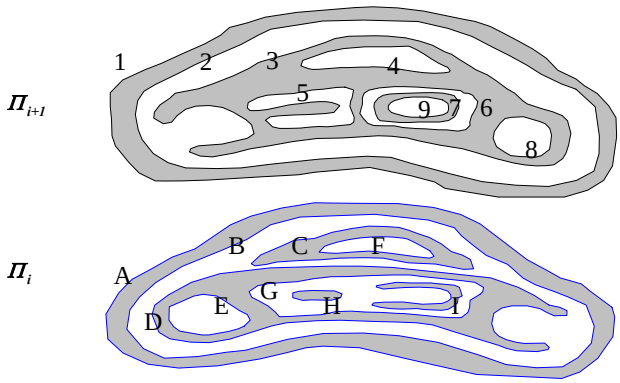


Figure 9. Polygon set in consecutive levels.

4.1.1. Contour Mappings

The following solid and hole regions were mapped according to the mapping group stage of the ***contour-map-BG( )*** algorithm:

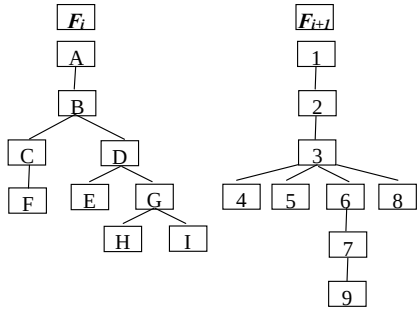
Solid Region map:

{(A B)}vs.{(1 2)) {(C F), (D E G), H}vs.{(3 4 5 6 8)} {I}vs.{(7 9)}

Hole map:

{(B C D)} vs.{(2 3)} {F}vs.{4} {E}vs.{Φ}

{Φ}vs.{8} {(G H I)}vs.{ 5 ,(6 7)} {Φ}vs.{9}



Contours: A, B, C, D, E, F, G, H, I, 1, 2**,** 3**,** 4**,** 5**,** 6**,** 7**,** 8**,** 9

Polygons: (A B), (C F), (D E G), H, I, (1 2)**,** (3 4 5 6 8), (7 9)

Tree Graphs: $\boldsymbol{T_{1,i}}$ = (A (B (C (F)) (D (E G (H I)))))**,** $\boldsymbol{T_{1,i+1}}$ = (1 (2 (3 (4 5 (6 (7 (9)) 8))))

Forests: $\boldsymbol{F_i}$ = { $\boldsymbol{T_{1,i}}$ }**,** $\boldsymbol{F_{i+1}}$ = { $\boldsymbol{T_{1,i+1}}$ }

Figure 10. Contour, polygon, tree and forest relations**.**

From the algorithm discussed above, the mapping groups including the empty (Φ) set are removed, since they represent an addition of 1-handles, which is handled through the ***BG( )*** algorithm without an explicit call to tiling algorithms. A sorting based on the $\prec$ order is applied on the list of remaining map groups, with the following results:

1.	{I}vs.{(7 9)}	2.	{F}vs.{4}	3.	{(G H I)}vs. {(5 (6 7))}
4.	{(C F), (D E G)}vs. {(3 4 5 6 8)}	5.	{(B C D)}vs. {(2 3)}	6.	{(A B)}vs. {(1 2))

4.1.2. Mapping group intersection elimination. Call to the ***BG( )*** algorithm

The algorithm in Table 2 is applied to eliminate the intersections between mapping groups. The depuration results are as follows:

1.	{I}vs.{(7 9)}	2.	{F}vs.{4}	3.	{(G H}vs.{5 6}
4.	{C (D E)}vs.{(3 8)}	5.	{B}vs.{2}	6.	{A}vs.{1}

By using these mapping groups, the calls to the ***BG( )*** algorithm are: ***BG(***{I},{7, 9}***)***, ***BG(***{F}, {4}***)***, ***BG(***{G,H},{5,6}***)***, ***BG(***{C,D,E},{3,8}***)***, ***BG(***{B},{2}***)***, ***BG(***{A},{1}***).*** Notice that the calls to ***BG( )*** do not admit hierarchies, but plain sets.

Figure 11 and Figure 12 show the result of direct application of the ***BG( )*** algorithm. Over – stretched branches may be observed in three places between contours B (level Π_i) and 3 (level Π_{i+1}), as a result of the sole application of geometric (Voronoi-Delone) closeness to join contours of opposite levels. In contrast, the approach proposed here, ***contour-map-BG( )***, (see Figure 13 and Figure 14), of forming separate working spaces (mapping groups) for the ***BG( )*** algorithm, avoids unnatural branches as the ones described, because (A B) polygon is mapped to (1 2), and not to (3 4 5 6 8), therefore making a branch from B to 3 impossible. In the proposed strategy, the 2D similarity criteria acts in a global manner, thus avoiding that unrelated contours share the same working space at a given time. Next, the ***BG( )*** algorithm is applied to the sub-problems, which only contain similar 2D composed shapes.

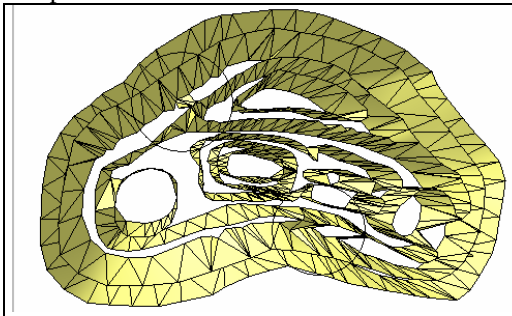


Figure 11. Front view. Result of direct use *BG( )* algorithm.

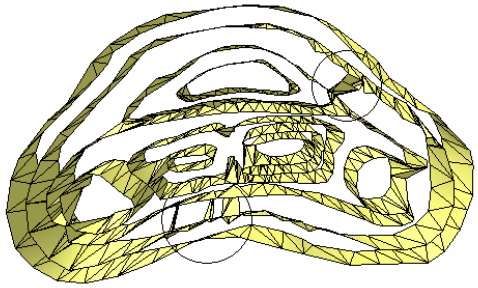


Figure 12. Back view. Result of direct use *BG( )* algorithm.

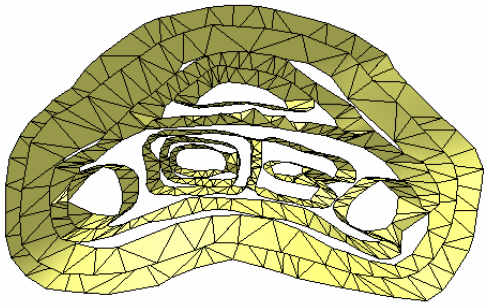


Figure 13. Front view. Result of *contour-map-BG( )* algorithm.

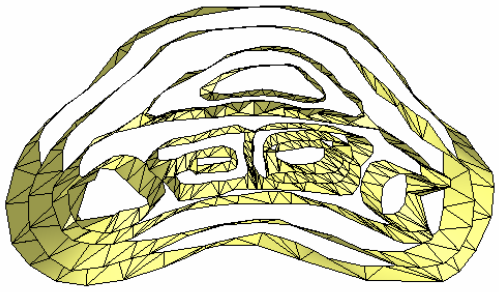


Figure 14. Back view. Result of *contour-map-BG( )* algorithm.

4.2. Example B

In Figure 15 it can be seen that tree (6 5) in level Π_{i+1} has no relation with tree (B D) in level Π_i. When the whole set of contours in Π_i and Π_{i+1} is fed to the ***BG( )*** algorithm

(Figure 16) (i) an over-stretched loft is produced between contours 6 and D, (ii) the algorithm ignores contours B and 5, (iii) contour C receives an unnatural branch from contour 2, and (iv) part of the polygonal region (1 2) is not completed. In contrast, the proposed ***contour-map-BG( )*** algorithm (Figure 17 and Figure 18) (i) recognizes the isolation between (6 5) and (B D), attempting no lofting between them, (ii) correctly tiles the (6 5) and (B D) polygonal regions, (iii) polygon (C E F) maps to (3 4) and not to (1 2), therefore avoiding the over-stretched branching and (iv) the relation between (3 4) and (1 2) is solved correctly via a 1-handle, as corresponds to "donut-croissant" transitions. Both algorithms, ***BG ( )*** and the proposed ***contour-map-BG( )*** correctly deal with contours E, F, and 4 by using 2-handles (for E and F) and 0-handle (for 4), respectively.

4.3. Example C

Figure 19 shows the contour data of another example. This is a frequent case in medical imaging, in which several topological changes take place between two consecutive slices of frames. Contour A closes the gap with C at one point, forming contour 1, while at the same time contour 1 is separated from contour 3 at the neck region.

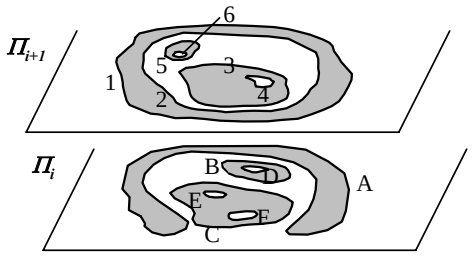


Figure 15. Example B. Contours in levels *i* and *i+1*.

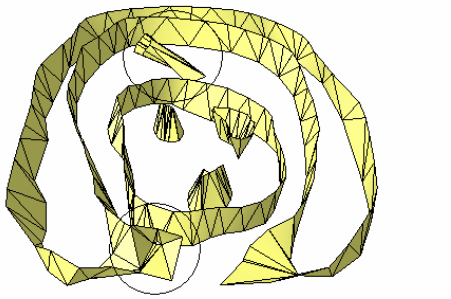


Figure 16. Example B. Result with direct *BG( )* algorithm.

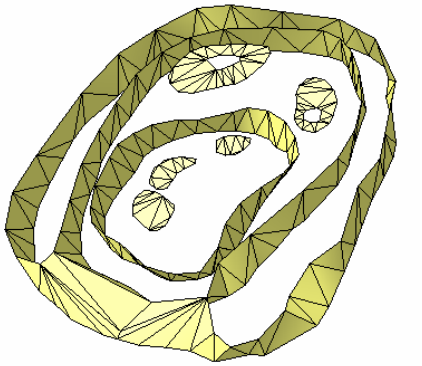


Figure 17. Example B. Upper view. Result of *contour-map-BG( )* algorithm.

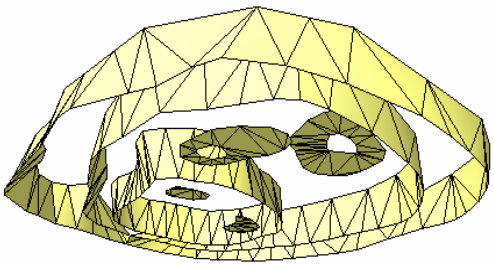


Figure 18. Example B. Lower view. Result of *contour-map-BG( )* algorithm.

The effect of directly applying the ***BG( )*** algorithm creates a branching from contour 2 (level Π_{i+1}) to B *and* A (level Π_i). See detail in Figure 20. In contrast, the ***contour-map-BG( )*** algorithm forms mapping groups {2} vs. {B} and {(1 2), 3} vs. {(A B), (C D)} among others. As a result, contours 2 and B are lofted. Any skin joining 2 to A is not even considered.

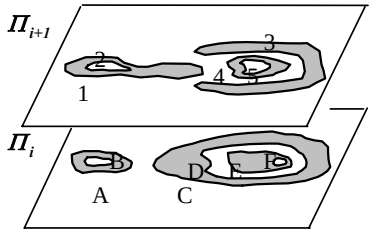


Figure 19. Example C. Contours in levels *i* and *i+1*.

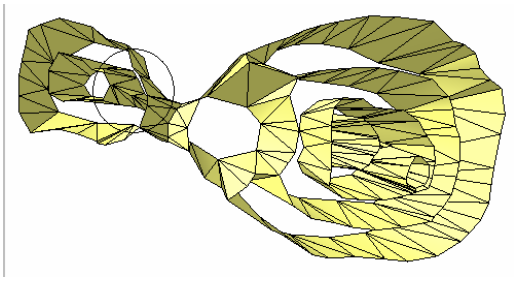


Figure 20. Example C. Direct application of *BG( )* algorithm for C data.

As a collateral effect of the incorrect union of 2 with A is that a medial axis appears in level Π_i joining contour A and B, which did not belong to the original contour set. Figure 21 and Figure 22 show that by forming the mapping groups before calling the ***BG( )*** algorithm precludes the over-stretched surface joining A, B and 2, and the edge path from A to B, incorrectly formed on plane Π_i when only the ***BG( )*** algorithm is applied.

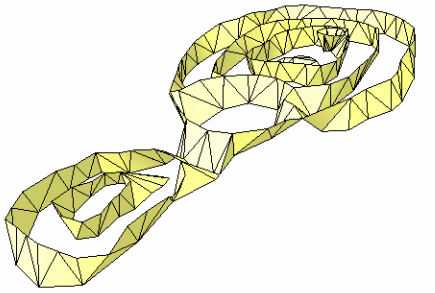


Figure 21. Example C. Front view result of *contour-map-BG( )*.

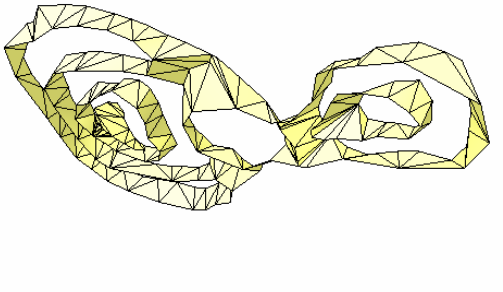


Figure 22. Example C. Back view result of *contour-map-BG( )*.

4.4. Example D

Figure 23 displays a skull data set, with a typical transition shown in Figure 24. The final surface has been included to facilitate the understanding of the point set, including void regions in the mouth and forehead neighborhoods. The surface has been synthesized using the mapping of similar 2D regions proposed here, along with the ***BG( )*** algorithm. The results appear in Figure 25 to Figure 28. Notice that as long as the voids in the sampling set do not (significantly) affect the topology or the geometry of the crosscuts, the final result is correct, with the only characteristic of having larger triangles. This is not a problem if the surface is topologically and geometrically correct, since there are many algorithms for relaxation and qualification of meshes. Since the data set has an interruption in the upper part of the skull, the algorithm produces a plateau, with the corresponding hole to the skull cavity, as in Figure 26. Figure 28 shows the surface between two levels of the Skull data set, which presents plenty of 0- , 1- and 2-handles. The surface is built by using mapping groups. As seen, the proposed algorithm correctly recognizes and handles these topological events.

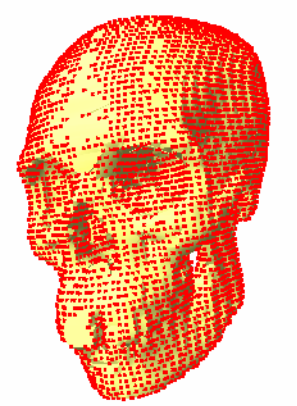


Figure 23. Skull data set. Point sample

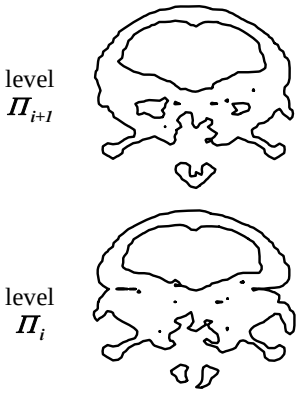


Figure 24. Consecutive levels of the skull data set.

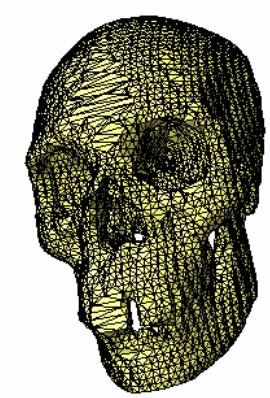


Figure 25. Result of the *contour-map-BG( )* algorithm.

5. CONCLUSIONS

An algorithm has been presented, which attacks the problem of surface reconstruction from slice samples using the point of view of the evolution of the crosscuts of the 2-manifold to be recovered. By using 2D shape similarity, inference on the topological events that take place between consecutive slices can be made. The match of 2D similar composed shapes also helps to steer the application of well known Voronoi-Delone (V-D)-based algorithms, which are very effective, but have the disadvantage of building over-stretched branches or unnatural bridges between far apart 2D regions of consecutive crosscuts.

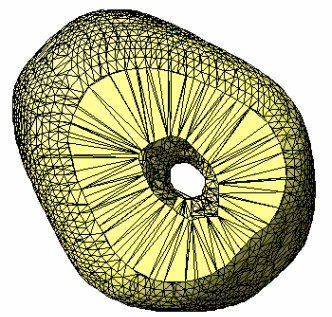


Figure 26. Result of the *contour-map-BG( )* algorithm. Upper view.

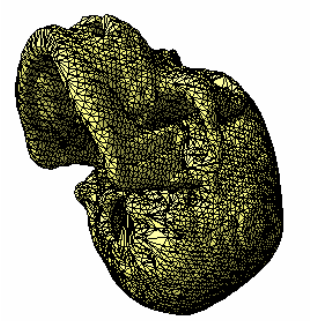


Figure 27. Result of the *contour-map-BG( )* algorithm. Lower view.

Figure 28. Result. *Contour_map_BG( )* algorithm for consecutive levels.

The algorithm presented here succeeds in avoiding such over-stretched or over-slanted surfaces, and therefore represents a step forward in ensuring geometrical *and* topological faithfulness between object and reconstructed model, while the V-D-based methods only ensure geometrical similarity. This advance comes at a price of algorithm speed, since only likely geometry and topology evolutions are permitted, and therefore additional data screening is required. For this reason, the algorithm presented is to be used when faithfulness to the actual topological evolution is a prime requirement. For approximate shape reconstruction, the V-D-based methods suffice.

Future work in the part of pre-processing corresponds to further screening of the mapping groups. At the present time, topological inconsistencies are filtered out after the mapping groups are calculated by reasoning on the parent – children relations in the inclusion trees. This screening may be partially avoided by introducing more strict conditions on the geometrical 2D similarity of the mapping groups. In both cases the result is correct, but the latter one depends less on post-processing rules and therefore it is a more desirable scenario.

REFERENCES

[Amenta et al.01] Nina Amenta, Sunghee Choi, Ravi Krishna Kolluri: The power crust, unions of balls, and the medial axis transform. Computational Geometry, Volume 19, Number 2-3, July 2001, pp. 127-153

[Bajaj et al.96] Bajaj, Ch., Coyle, E. Lin, K., Arbitrary Topology Shape Reconstruction from Planar Cross Sections, Graphical Models & Image Processing. Vol 58, No. 6, pp. 524-543, 1996.

[Barequet et al.96] Barequet, B. and Sharir, M., Piecewise-linear interpolation between polygonal slices. Comput. Vision Graph. Image Process, vol 63, 1996.

[Bernardini et al.99] F. Bernardini, Ch. Bajaj, J. Chen, D. Schikore. Automatic Reconstruction of 3D CAD Models from Digital Scans, International Journal of Computational Geometry and Applications, 9(4/5): 327-369, 1999.

[Bloomenthal.97] Bloomenthal, J. Editor. Introduction to Implicit Surfaces. (Morgan Kaufmann Publishers, 1997).

[Bloomenthal.88] Bloomenthal, J., Computer Aided Geometric Design Vol. 5, Issue 4, Nov. 1988. pp. 341 - 355

[Boissonat et al.93] D. Boissonat and B. Geiger, Three Dimensional Reconstruction of Complex Shapes based on Delaunay Triangulation. SPIE Proceedings Vol. 1905. Biomedical Image Processing and Biomedical Visualization, San Jose, CA, USA, 964-975, Feb 1993.

[Bookstein.89] Bookstein, F., Principal Warps: Thin-Plate Splines and the Decomposition of Deformations. IEEE Transactions on Pattern Analysis and Machine Intelligence. Vol 11, No. 6, June 1989.

[Carr et al.97] Carr, J., Fright, W., Beatson, R., Surface Interpolation with Radial Basis Functions for Medical Imaging. IEEE Transactions on Medical Imaging, 16(1), pp.96-107, 1997.

[Curless et al.96] Curless, B., Levoy, M., A volumetric Method for Building Complex Models from Range Images. Proceedings SIGGRAPH-96, 1996, pp 303-312.

[Edelsbrunner.94] Edelsbrunner, H., Three Dimensional Alpha Shapes, ACM Transactions on Graphics, Vol 13, No 1, Jan, 1994, pp 43-72.

[Fomenko et al.97] Fomenko, A., Kunii, T. Topological Modeling for Visualization, (Tokio, Springer Verlag, 1997).

[Geiger.93] Geiger, B., Three-Dimensional Modeling of Human Organs and its Application to Diagnosis and Surgical Planning. PhD thesis, Ecole des Mines de Paris, 1993.

[Hoppe et al.93] Hoppe, H., DeRose, T., Duchamp, T., MacDonald, J., Stuetzle W., Mesh Optimization. Computer Graphics Proceedings, SIGGRAPH Annual Conference Series, 1993. pp 19-26.

[Lorensen et al.87] Lorensen, W., Cline, H., Marching Cubes: A High Resolution 3D Surface Construction Algorithm, ACM Computer Graphics, Vol. 21, No. 24, July, 1987, pp. 163-169.

[Morse.34] Morse, M., The calculus of variations in the large, American Mathematical Society, New York, 1934.

[Morse et al.01] Morse, B., Yoo, T. S., Rheingans, P., Chen, D. T., Subramanian, K.R., Interpolating implicit surfaces from scattered surface data using compactly supported radial basis functions. Proceedings of the Shape Modeling Conference, Genova, Italy, 89-98, May 2001.

[Neugebauer.97] Neugebauer P., Reconstruction of Real World Objects via Simultaneous Registration and Robust Combination of Multiple Range Images. International Journal of Shape Modeling, Vol. 3, No. 1&2, 1997, pp 71-90.

[Oliva et al.96] Oliva, J., Perrin, M., Coquillart, S., 3D Reconstruction of Complex Polyhedral Shapes from Contours using a Simplified Generalized Voronoi Diagram. Comp. Graphics Forum, 15 (3): C-397-C-408, 1996.

[Ruiz et al. 02a] Ruiz, O. Cadavid, C., Granados, M., Evaluation of 2D Shape Likeness for Surface Reconstruction. Journal Anales de Ingeniería Gráfica, ISSN: 1137-7704, No 15, 2002, pp.16-24.

[Ruiz et al. 02b] Ruiz, O. Karangelis, G., Boundary Representation of Anatomical Features. Computer Graphics Topics, No. 6 /2002. ISSN 0936-2770, pp 24-25.

[Shannon.49] Shannon, C., Communication in Presence of Noise. IRE 37, 10-21.

[Shinagawa et al.91] Shinagawa, Y., Kergosien, Y., Kunii, T., Surface coding based on Morse Theory. IEEE Computer Graphics & Applications 11 (5), 66-78.

[Shinagawa et al.91b] Shinagawa, Y., Kunii, T., Constructing a Reeb Graph Automatically from Cross Sections. IEEE Computer Graphics & Applications. 11(6), 44-51, Nov. 1991.

[Turk et al.94] Turk, G., Levoy, M., Zippered Polygon Meshed from Range Images. Proceedings SIGGRAPH-94, 1994, pp 311-318.

[Turk et al.02] Turk, G., O´Brien., Modeling with Implicit Surfaces that Interpolate. ACM Transactions on Graphics. Vol. 21, No. 4, October 2002, pp. 855-873.

ACKNOWLEDGEMENTS

The authors wish to thank the following people and institutions for the collaboration for this research: EAFIT University (projects DigitLAB, TOPO-2000, MedImagenes, FixGrid), Fraunhofer Institute Graphische Datenverarbeitung (Darsmstad, Germany), Divisions of Cognitive Computing and Medical Imaging (Drs. Sakas and Karangelis), Industrial Applications (Dr. Joachim Rix) and Geographic Information Systems (Dr. Uwe Jasnoch), as well as the Max-Plank Institute für Informatik (Saarbruecken, Germany) - Complexity Group (Drs. Kurt Melhorn and Lutz Kettner), University of Vigo, (Drs. Xoan Leiceaga Baltar and Fernando-Perez-Fontán) and the Colombian Council for Science and Technology - Colciencias.

VITAE

Associate Professor Oscar Ruiz was born in 1961 in Tunja, Colombia. He obtained B.Sc. degrees in Mechanical Eng. (1983) and Computer Science (1987) at Los Andes University, Bogotá, Colombia, a M.Sc. degree with emphasis in CAM (1991) and a Ph.D. with emphasis in CAD (1995) from the Mechanical & Industrial Eng. Dept. of University of Illinois at Urbana-Champaign, USA. Dr. Ruiz has held Visiting Researcher positions at Ford Motor Co. (Dearborn, USA. 1993 and 1995), Fraunhofer Inst. Graphische Datenverarbeitung (Darsmstad, Germany 1999 and 2001), University of Vigo (1999 and 2002). In 1996 Dr. Ruiz was appointed as Faculty of the Mechanical Eng. and Computer Science Depts. at EAFIT University, Medellin, Colombia, and has ever since the coordinator of the Laboratory for Interdisciplinary Research on CAD / CAM / CAE. Dr. Ruiz' interests are Computer Aided Geometric Design, Geometric Reasoning and Applied Computational Geometry.

Associate Professor C. Cadavid was born in 1965 in Medellín, Colombia, where he obtained a bachellor´s degree in Mathematics at the Universidad Nacional (1988). He obtained a Master´s degree from the University of Cincinnati (1990), and a Ph. D. Degree in Mathematics in 1998 from the University of Texas at Austin. He is currently a full time Faculty member at the Universidad EAFIT (Medellín). His areas of research are the topology of 4-dimensional manifolds and applications of topology to graphic computation.

Miguel Granados (Medellín, Colombia, 1981). He is a Computer Science student at EAFIT University, and research assistant in the CAD/CAM/CAE Laboratory EAFIT since 2000. In 2001 he was awarded a research internship at the Max-Plank Institute für Informatik (Saarbruecken, Germany), where he worked with the Complexity Group in generalization of B-Rep through Nef polyhedra. His main interest areas are computational geometry, computer vision and software engineering.

Eliana Vásquez (Medellín, Colombia, 1980) is a Computer Science student at EAFIT University, Medellin, COLOMBIA. In 2000 she joined the CAD / CAM / CAE Laboratory as research assistant. In 2000 and 2002 she was awarded a research internships in Fraunhofer Institute Graphische Datenverarbeitung (Darsmstad, Germany), Division of Cognitive Computing and Medical Imaging (A7). Her areas of interest are computer graphics, computational geometry, shape reconstruction applied to the medical field and computer vision.

Sebastian Peña (Manizales, Colombia, 1979) is a Mechanical Engineering student at EAFIT University, Medellín, Colombia. In May 2002 he joined CAD / CAM / CAE Laboratory at EAFIT. In 2003 he was awarded an internship for research on geometric characterization of aerospace structures at the Dept. of Telecommunications Engineering at University of Vigo, Spain. His main interest areas are robotic kinematics, mechanical design, computer aided geometric design for industrial applications, computational geometry and shape reconstruction.