3D Topological Map Extraction From an Oriented Boundary Graph

Fabien Baldacci, Achille Braquelaire, Guillaume Damiand

3D Topological Map Extraction from Oriented Boundary Graph

Fabien Baldacci[1], Achille Braquelaire[1], and Guillaume Damiand[2]

[1] Université Bordeaux 1, CNRS, Labri, UMR5800, F-33405, Talence Cedex, France
{baldacci,braquelaire}@labri.fr
[2] Université de Lyon, CNRS,
Université Lyon 1, LIRIS, UMR5205, F-69622, Villeurbanne Cedex, France
guillaume.damiand@liris.cnrs.fr

Abstract. The extraction of a 3D topological map from an Oriented Boundary Graph can be needed to refine a 3D Split and Merge segmentation using topological information such as the Euler characteristic of regions. A presegmentation could thus be efficiently obtained using a light structuring model, before proceeding to the extraction of more information on some regions of interest. In this paper, we present the topological map extraction algorithm which allows to reconstruct locally a set of regions from the Oriented Boundary Graph. A comparison of the two models is also presented.
Keywords. 3D split and merge, image segmentation, topological structuring.

1 Introduction

The Segmentation process consists in defining a partition of the image into homogeneous regions. Split and merge methods [HP74] are widely used in segmentation. It consists in alternatively splitting a region, and merging adjacent ones according to some criteria, in order to define a partition of the image. To be efficient, it needs a topological structuring of the partition in order to retrieve some information such as: the region containing a given voxel, the list of regions adjacent to a given one, the list of surfaces defining the boundary of a region, etc [BBDJ08].

Several models have been proposed to represent the partition of an image. A popular model is the Region Adjacency Graph [Ros74], which is not sufficient for most of 3D segmentation algorithms due to the lack of information encoded. A more sophisticated model is the topological map model [Dam08] that uses combinatorial maps in order to encode the topology of the partition and an intervoxel matrix [BL70] for the geometry. It encodes all information required to design split and merge segmentation algorithm including high topological features allowing to retrieve Euler characteristic and Betti numbers of a region [DD02]. Since high level topological features are not necessary for basic split and merge segmentation algorithms, an other model have been proposed [BBDJ08]. This model uses a multigraph called Oriented Boundary Graph (OBG) to encode

the topology associated with the same geometrical level than in the topological map model.

This second model is both more efficient (table 1) and less space consuming. The space consumption difference cannot be exactly computed because the memory optimized implementation of the *OBG* is still under development, but unoptimized version show that it will be at least two to four times less space consuming depending on the number of regions and surfaces of the partition. The space consumption can be a critical constraint with large image or with algorithms needing a highly oversegmented partition during the segmentation process.

Image size (nb of voxels)	Image complexity (nb of regions)	Topological map model construction time (s)	OBG model construction time (s)
256x256x256	34390	35.8	3.2
324x320x253	103358	69.15	5.26
512x512x475	279618	277.88	23.15
512x512x475	518253	301.27	24.37
512x512x475	1121911	317.01	27.11

Table 1. Construction time comparison between the two models.

The *OBG* model is more efficient than the topological map one, it can be efficiently parallelized [BD08] and is sufficient for split and merge segmentation that does not use topological characteristics of regions as criteria. But this missing information could in some cases be necessary and that is the reason why we have studied the extraction of the topological map of some regions of interest from the *OBG*. It could also be useful for the topological map model to use a more efficient model for a presegmentation step and extract the topological map from the simplified partition using an algorithm avoiding to traverse all the voxels. Thus this work is suitable both for the OBG model in order to have an on-demand high topological features extraction, and for the topological map model in order to be efficiently extracted from a presegmented image, for which the equivalent presegmentation using topological map is too much time or space consuming.

This paper is organized as follow. In section 2, we describe and briefly compare the two topological models. Then in section 3 we describe the topological map extraction algorithm from the *OBG*. We conclude in section 4.

2 Presentation of the Two Models

Let us present some usual notions about image and intervoxels elements. A voxel is a point of discrete space $\mathbb{Z}^3$ associated with a value which could be a color or a gray level. A three dimensional image is a finite set of voxels. In this work, combinatorial maps are used to represent voxel sets having the same labeled

value and that are 6-connected. We define region as a maximal isovalued set of 6-connected voxels.

To avoid particular process for the image border voxels, we consider an infinite region R_0 that surrounds the image. If a region R_j is completely surrounded by a region R_i we say that R_j is *included* in R_i.

2.1 Recalls on 3D Topological Maps

A 3D topological map is an extension of a combinatorial map used to represent a 3D image partition. Let us recall the notions on combinatorial maps, 3D images, intervoxel elements and topological maps that are used in this work.

A combinatorial map is a mathematical model describing the subdivision of space, based on planar maps. A combinatorial map encodes all the cells of the subdivision and all the incidence and adjacency relations between the different cells, and so describe the topology of this space.

The single basic elements used in the definition of combinatorial maps are called *darts*, and adjacency relations are defined onto darts. We call β_i the relation between two darts that describes an adjacency between two i-dimensional cells (see Fig. 1B for one example of combinatorial map and [Lie91] for more details on maps and comparison with other combinatorial models). Intuitively, with this model, the notion of cells is represented by a set of darts linked by specific β_i relations. For example, a face incident to a dart d is represented by the set of darts accessible using any combination of β_1 and β_3 relations. Moreover, given a dart d, which belongs to an i-cell c, we can find the i-cell adjacent to c along the $(i-1)$-cell which contains d by using $\beta_i(d)$. For example, given a dart d that belongs to a face f and a volume v, the volume adjacent to v along f is the 3-cell containing $\beta_3(d)$. Lastly, we call i-sew the operation which puts two darts in relation by β_i.

In the intervoxel framework [KKM90], an image is considered as a subdivision of a 3-dimensional space in a set of unit elements: voxels are the unit cubes, surfels the unit squares between two voxels, linels the unit segments between surfels-cells and pointels the points between linels (see example in Fig. 1C).

The *topological map* is a data structure used to represent the subdivision of an image into regions. It is composed of three parts:

- a minimal combinatorial map representing the topology of the image;
- an intervoxel matrix used to retrieve geometrical information associated to the combinatorial map. The intervoxel matrix is called the *embedding* of the combinatorial map;
- an inclusion tree of regions.

Fig. 1 presents an example of topological map. The 3D image, composed of three regions plus the infinite region R_0 (Fig. 1A), is represented by the topological map which is divided in three parts labeled B, C and D. The minimal combinatorial map extracted from this image is shown in Fig. 1B. The embedding of the map is represented in Fig. 1C, and the inclusion tree of regions in Fig. 1D.

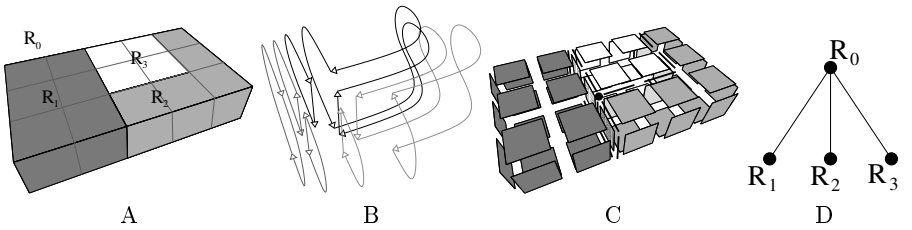

Fig. 1. The different parts of the topological map used to represent an image. (A) 3D image. (B) Minimal combinatorial map. (C) Intervoxel matrix (embedding). (D) Inclusion tree of regions.

The combinatorial map allows the representation of all the incidence and adjacency relations between cells of an object. In the topological map framework, we use the combinatorial map as a topological representation of the partition of an image into regions. Each face of the topological map is separating two adjacent regions and two adjacent faces do not separate the same two regions. With these rules, we ensure the minimality (in number of cells) of the topological map (see [Dam08] for more details on topological maps).

The intervoxel matrix is the embedding of the combinatorial map. Each cell of the map is associated with intervoxel elements representing geometrical information of the cell.

The inclusion tree of regions represents the inclusion relations. Each region in the topological map is associated to a node in the inclusion tree. The nodes are linked together by the inclusion relation previously defined.

2.2 Oriented Boundary Graph Model

The second model is composed of a multigraph called Oriented Boundary Graph [BBDJ08]. Each node of the graph corresponds to a region of the partition. Each surface of the segmented image corresponds to an oriented edge of the graph. Surfaces and edges are linked by associating an oriented surfel to each edge. Each edge is also linked to a representative linel for each border of its corresponding surface; this is necessary to retrieve the surface adjacency relation (which is used to compute the inclusion relation). Geometrical position of the region relating to its boundary surfaces is retrieved according to the orientation of the embedding surfel and the position of the node (beginning or end of the oriented edge). This graph is sufficient to encode the multiple region adjacency relation, the surface adjacency relation and thus the inclusion relation could efficiently be computed. The geometrical level encoded by an intervoxel matrix is the same as the one used with topological map, and links are defined in order to go both from the geometrical level to the topological one and reciprocally. An example is shown on Fig. 2.

This model, contrary to the precedent one, can be directly extracted from the description of the image partition without treatments on the resulting sur-

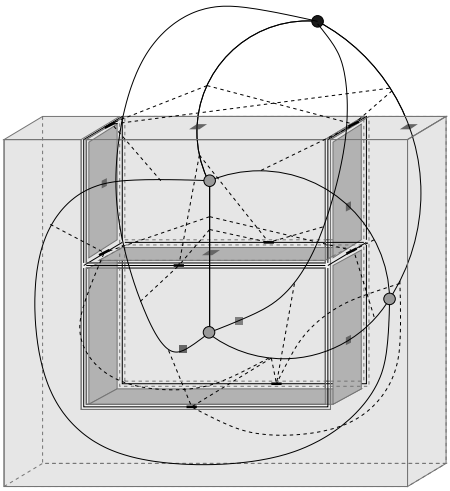

Fig. 2. Example of image partition with the corresponding representation using the Oriented Boundary Graph model.

faces. It only needs a strong labelling of the voxels and linels that can be locally computed (by looking at the neighbors of the considered element). For each new label, the corresponding topological element have to be created, and the topological and geometrical elements to be linked themselves. This model requires less processing than the topological map one to be maintained and is so more efficient by avoiding to have each surface homeomorphic to a topological disc. Furthermore, this split algorithm requires only local treatment and could be efficiently computed in parallel [BD08]. Information encoded by the *OBG* model are sufficient to design basic split and merge segmentation algorithms. But it could be necessary for some segmentation algorithms to use some high topological features such as Euler characteristic of some regions, either as a segmentation criteria or as a constraint on regions in a merge step. In order to design efficiently those segmentation algorithms it is necessary to build the topological map of a set of selected regions.

2.3 Comparison of Both Models

Let us recall advantages and drawbacks of each model. The Topological Map model encodes the whole topology of the partition, from the regions and surfaces adjacencies to the Euler characteristic and Betti number of regions. Computing this map consumes an important memory space and requires a time consuming extraction algorithm. The *OBG* is an enhanced multiple adjacency graph with an intervoxel matrix associated. It is intended to be simpler than the topological map but also less expressive. It has an efficient extraction algorithm and uses low memory space. But some high topological features such as the characteristics of regions are not encoded. Given a description of the image partition with a matrix

of labels, the *OBG* extraction algorithm have a $O(v + s + l)$ complexity with v the number of voxels, s the number of surfels and l the number of linels, because each operation is proceeded once by element and takes $O(1)$. The topological map extraction algorithm has the same theoretical complexity than the the *OBG* extraction algorithm, $O(v+s+l)$, for the same reason: each cell of the intervoxel subdvision is processed exactly once. However, the number of operations achieved for each cell is more important, which explains the difference in extraction times.

Advantages and drawbacks of each model could be summarized as follows:

- OBG: enhanced multiple region adjacency graph with an intervoxel matrix embedding
  - advantages: simple, efficient extraction algorithm, low memory space consumption
  - drawbacks: does not represent topological characteristics of regions
- Topological map: combinatorial map describing the subdivision of an image into sets of vertices, edges, faces and volumes
  - advantages: represent topological characteristics of region, represent all the cells (vertices, edges, faces and volumes)
  - drawbacks: high memory space and time consumption required for the extraction algorithm

Converging to an optimal model that have the efficiency of an OBG and the expression of a Topological Map is not possible. That is the reason why we need a conversion algorithm allowing to take advantages of the two models, by not using the same model during the whole segmentation process.

3 The Conversion Algorithm

The principle of the algorithm is to start with an OBG G embedded in an intervoxel matrix I, and a set of connected regions S, and to compute the local 3D topological map M representing S, but taking into account neighbors regions not included in S.

The extraction of the topological map is achieved by building the map of each surface in S, and linking them together using corresponding β_i in order to represent S. Surfaces and real edges already exists in the OBG, thus only fictive edges have to be computed and the β_i relations need to be fixed in order to have the 3D topological map corresponding to S.

The algorithm is divided into two subtasks: the main task reconstructs the map corresponding to a set of regions. This task uses a second task which reconstruct one face of the region.

3.1 Region Reconstruction

Algorithm 1 is the main part which reconstructs the part of topological map representing a given set of regions.

To reconstruct a given region R, we run through edges of the OBG. Indeed, each edge corresponds to a surface of R. Now, two cases have to be considered depending if the surface is closed or not.

Algorithm 1: Region set reconstruction

```
Input: G an OBG
       S a set of connected regions
Output: The topological map representing S
foreach edge e adjacent to a region of S do
    if ∃ at least a linel associated to e then
        l ← first linel of e;
        b ← build the face associated to l;
        foreach linel l' associated to e (except l) do
            b' ← build the face associated to l';
            insert a fictive edge between b and b';
    else b ← NULL;
1   if g(e) ≠ 0 then
        create 2 × g(e) edges, loop sewed on themselves;
        insert them around b;
```

1. if the surface is closed, there is no linel associated to the surface in the OBG (because linels represent border of surfaces). In such a case, we need to construct a map composed of $2 \times g(e)$ edges, 1 vertex and 1 face (with $g(e)$ the genus of the current surface).
2. if the surface is open, let us denote by k the number of boundaries, each one being a list of consecutive linels forming a loop. Each boundary is reconstruct by using Algorithm 2. Moreover, each new face is linked with previous map by adding a fictive edge. This ensures to obtain a connected map. Then, we may need to add some edges in order to obtain a surface with the "correct topology". In order to do so, as in the previous case, we construct a map composed of $2 \times g(e)$ edges, 1 vertex and 1 face, but in this case we link this new map with the map already build and corresponding to boundaries in order both to obtain a connected map and having the correct topology.

Adding fictive edges to the existing edges in the OBG allows to retrieve the two properties of the topological map that are missing in the OBG. Indeed, fictive edges (i) link the different boundaries of a surface and (ii) conserve a valid Euler characteristic for each surface.

Before applying Algorithm 1, we need to compute the Euler characteristic of each surface since this information is not present in the OBG but it is needed during the map reconstruction. For that, we compute for each edge e of G, $\#v$ $\#e$ and $\#f$ (respectively the number of vertices, edges and faces of the surface associated to e).

The Euler characteristic of the face associated to e is denoted by $\chi(e) = \#v - \#e + \#f$. The genus associated to this surface is denoted by $g(e)$ and computed with the Euler formula formula: $g(e) = \frac{2-\chi(e)}{2}$. The Euler characteristic of the surface of a region r is the sum of the Euler characteristic of all its faces (the fact that vertices and edges incident to two faces are counted twice is not a

problem for the Euler characteristic since it uses the difference between these two numbers).

3.2 Face reconstruction

The principle of the face reconstruction given in Algorithm 2 is to traverse the geometry (the linels) and to reconstruct darts and β_1 relations. Each created dart is linked with the associated triplet in the OBG. For each linel, if some neighbors triplets have already been treated, β_2 and β_3 are updated.

Algorithm 2: Face border reconstruction

Input: G an OBG
l_1 a linel belonging to the border of a face
Output: The part of the map corresponding to this border

$d_{prev} \leftarrow nil$; $d_{first} \leftarrow nil$;
foreach *linel* l *of the face border incident to* l_1 **do**
  if l *is incident to a pointel* p **then**
    Compute the surfel s from p, l and the current region;
    $d \leftarrow$ new dart associated to the triplet (p, l, s);
    if $d_{first} = nil$ **then** $d_{first} \leftarrow d$;
    else 1-sew(d_{prev}, d);
    $d' \leftarrow$ dart associated to the triplet (p, l_{prev}, s_2);
    if $d' \neq nil$ **then** 2-sew(d_{prev}, d');
    $d'' \leftarrow$ dart associated to the triplet (p, l_{prev}, s_3);
    if $d'' \neq nil$ **then** 3-sew(d_{prev}, d'');
    $d_{prev} \leftarrow d$;
1-sew(d_{prev}, d_{first});
$d' \leftarrow$ dart associated to the triplet (p, l_{prev}, s_2);
if $d' \neq nil$ **then** 2-sew(d_{prev}, d');
$d'' \leftarrow$ dart associated to the triplet (p, l_{prev}, s_3);
if $d'' \neq nil$ **then** 3-sew(d_{prev}, d'');
return d_{first};

During this computation, new created darts are associated to their triplets (that have to be oriented) in order to retrieve, when a new dart is created, incident darts to the same triplet, and thus update the β_2 and β_3 links.

This algorithm is local: when processing dart d associated to triplet (p, l, s), we search for darts already existing in the neighborhood of d and sew found darts with d. In Fig. 3, we explain how triplets (p, l_{prev}, s_2) and (p, l_{prev}, s_3) are computed from (p, l, s).

3.3 Complexity and Proof of Correctness of the Algorithm

Complexity. Algorithm 1 is in time $\mathcal{O}(n_l + g + n_s)$ with n_l the number of linels of reconstructed regions, g its genus, and n_s the number of surfels of reconstructed

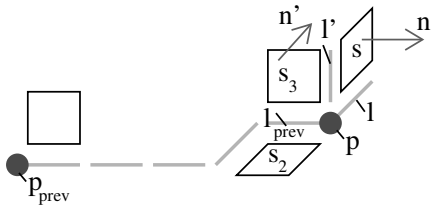

Fig. 3. How to compute triplets (p, l_{prev}, s_2) and (p, l_{prev}, s_3). (p, l, s) is the triplet associated to current dart d, and p_{prev} is the pointel incident to dart d_{prev}. We want to sew d_{prev} by β_2 and β_3 if corresponding darts are already created. s_3 is the first surfel found from s, by turning around linel l' in the direction of $\overrightarrow{n}$ (the normal of s, oriented towards the current region r). l_{prev} is the linel incident to p and s_3 (*i.e.* the previous linel of the current border). s_2 is the first surfel found from s_3, by turning around linel l_{prev} in the opposite direction of $\overrightarrow{n'}$ (the opposite of the normal of s_3. Indeed, the normal of s_3 is oriented towards the adjacent region of r, thus the opposite is oriented towards r).

regions. Indeed, Algorithm 2 passes through all linels of the process border. Each operation is atomic, and finding triplet (p, l_{prev}, s_2) and (p, l_{prev}, s_3) can be achieved in at most 4 operations (since there are at most 4 surfels around a linel). In Algorithm 1, we process successively and exactly once each border of the reconstructed region. This give the first part of the complexity $\mathcal{O}(n_l)$. The second part is due to the adding of $2 \times g$ edges which is done in linear time depending on g. The last part corresponds to the computation of g, which required each surfel to be considered once leading to a complexity in $\mathcal{O}(n_s)$. □

Proof of correctness. Firstly, Algorithm 1 build a combinatorial map where each dart is sew for β_1 and β_2. For β_1, this is directly due to Algorithm 2 which follows one cycle of closed linels. At the end of this step, we have created a closed list of darts which are β_1 sew. Moreover, since in Algorithm 1 we process each face of reconstructed region, and since the border of a region is closed, we are sure that given a face f, we process all the adjacent faces of f and thus each dart is β_2-sew.

Secondly, we need to prove that the Euler number of the reconstructed region is correct. We note g the genus of the region. Algorithm 2 computes only faces which are homeomorphic to topological disks (each face has one closed boundary). Thus, if we do not add fictive edges, we obtain a sphere, with $\chi = \#v - \#e + \#f = 2$. To this surface, we add $2 \times g$ edges (more precisely we add the sum of $2 \times g(e)$ for each edge e of the model, and this sum is equal to $2 \times g$ as explain above), without modifying the number of vertices nor the number of faces. Thus, the new Euler characteristic is $\chi' = \#v - (\#e + 2g) + \#f$ and so $\chi' = \chi - (2 \times g) = 2 - 2g$: we obtain the correct Euler characteristic of a surface of genus g. □

4 Conclusion

Split and merge segmentation in the 3D case could be a highly time consuming method without the use of a topological structuring. But an optimal structuring both in term of time and space consumption and in term of topological features representation could not be achieved. That is the reason why two models have been developed: the Topological map one which represent the whole topology of an image partition and the OBG model which is more efficient according to time and space consumption.

In this article we have developed an algorithm that allows to extract the Topological Map from the OBG. This operation allows to have an on-demand extraction of the Topological Map from some regions of the OBG, which allows to locally retrieve all the topological features of some regions of interest in the image partition.

The other utilization of this algorithm is to extract the Topological Map of the whole image partition but only a the step of the segmentation process where it is needed. The presegmentation will be done using the OBG in order to have a lower time consumption or in order to avoid a lack of memory.

In future works, we want to study the possibility to modify the topological map reconstructed, for example by an algorithm which take into account a topological criterion, and then to update locally the OBG model to reflect the image modifications.

References

[BBDJ08] F. Baldacci, A. Braquelaire, P. Desbarats, and Domenger J.P. 3d image topological structuring with an oriented boundary graph for split and merge segmentation. In *Proceedings of the 14th IAPR International Conference on Discrete Geometry for Computer Imagery*, pages 541–552, 2008.

[BD08] F. Baldacci and P Desbarats. Parallel 3d split and merge segmentation with oriented boundary graph. In *Proceedings of The 16-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision'2008*, pages 167–173, 2008.

[BL70] C. R. Brice and Fennema C. L. Scene analysis using regions. *Artif. Intell.*, 1(3):205–226, 1970.

[Dam08] G. Damiand. Topological model for 3d image representation: Definition and incremental extraction algorithm. *Computer Vision and Image Understanding*, 109(3):260–289, March 2008.

[DD02] P. Desbarats and J.-P. Domenger. Retrieving and using topological characteristics from 3D discrete images. In *Proceedings of the 7th Computer Vision Winter Workshop*, pages 130–139. PRIP-TR-72, 2002.

[HP74] S.L. Horowitz and T. Pavlidis. Picture segmentation by a directed split and merge procedure. In *ICPR74*, pages 424–433, 1974.

[KKM90] E. Khalimsky, R. Kopperman, and P.R. Meyer. Boundaries in digital planes. *Journal of Applied Mathematics and Stochastic Analysis*, 3(1):27–55, 1990.

[Lie91] P. Lienhardt. Topological models for boundary representation: a comparison with n-dimensional generalized maps. *Computer-Aided Design*, 23(1), 1991.

[Ros74] A. Rosenfeld. Adjacency in digital pictures. In *InfoControl*, volume 26, 1974.