Topological Map Based Algorithms for 3D Image Segmentation

Guillaume Damiand, Patrick Resch

Topological Map Based Algorithms for 3D Image Segmentation

Guillaume Damiand and Patrick Resch

LIRMM, 161 rue Ada, 34392 Montpellier Cedex 5, France
{damiand,resch}@lirmm.fr

Abstract. One of the most commonly used approach to segment a 2D image is the *split* and *merge* approach. In this paper, we are defining these two operations in 3D within the topological maps framework. This mathematic model of regions segmented image representation allows us to define these algorithms in a local and generic way. Moreover, we are defining a new operation, the *corefining*, which allows to treat big images. They are cut into small units, treated separately, then the result of each of them are combined to reconstruct the final representation. These three operations let us view efficient 3D segmentation algorithms, which is a difficult problem due to the size of data to treat.

1 Introduction

The region segmentation is a difficult problem that was studied in many different works in 2 dimensions. It consists in making a partition of an image into connected sets of pixels verifying an homogeneity criterion, and that we are calling regions. The main approaches for the region segmentation are the *split and merge* methods.

The *top-down* approach [16, 13] consists in taking big regions and cutting into smaller and smaller regions. The *bottom-up* approach [6, 11] is the opposite approach, which begins with many small regions that are merged into bigger and bigger regions. At last the *mixed* approach [12, 17] consists in mixing the two previous ones, possibly making again the process until for example, result stabilization.

These approaches require a "good" model of images representation. Many works in 2 dimensions [10, 7, 8, 3] have shown that topological maps are a model that allows to introduce these segmentation algorithms in an efficient way. Moreover, thanks to that model we can also define many processing algorithms allowing to access or to modify the result of this segmentation.

Recently, the topological maps have been extended in dimension 3 [5, 2]. Indeed, more and more domains need to work in dimension 3, as medical imagery, geology, or industry. In order to carry out segmentation algorithms as in dimension 2, we need to define basic operations on topological maps. This problem turns out to be more difficult than in dimension 2, because first now we have an additional dimension, that sets down new problems, but also because it is much

more difficult to represent and to make visual objects in 3 dimensions. These visualization problems act as a brake to the comprehension and the development of new algorithms. Moreover, there are complexity constraints, in memory space as well as in execution time, that are much more important than in dimension 2. Indeed, the treated data quantity is much more important in dimension 3. It requires us to define very efficient algorithms to use them on big images.

In this paper, we present the two algorithms of *merge* and *split* on the 3D topological maps. These two algorithms are the basic operations for the regions segmentation. The use of the topological maps let us define these algorithms in a generic way, because they work on any configuration, and in a local way because they treat the map element by element, only looking the direct neighborhood of the current element. These two properties make these algorithms more simple to understand but more efficient in complexity too. We also present the algorithm of *corefining* that allows us to treat big 3D images in parallel. We are going to cut this image into several small units, we are going to segment each unit in parallel, then we will reconstruct all the image with this units, using the corefining operation. Because of a lack of space and to not lose ourselves into technical details we are just presenting the principle of the algorithms and the main ideas of these three operations. For more details about these operations, we can report to [18, 19].

We are first presenting Section 2 the combinatorial maps then the topological maps that are combinatorial maps verifying specific properties. Then we are presenting our three algorithms : merge Section 3, split Section 4 and corefining Section 5. We are describing their principle and the different cases we have met. At last, we are concluding and presenting some perspectives Section 6.

2 Topological Maps Recall

Topological maps allow to represent the nD regions segmented images. They encode at the same time the topology and the geometry of images. Topological maps are combinatorial maps with particular properties. So we are beginning by recalling the notion of combinatorial map. This is just a short reminder; a more detailed description can be found in [5, 1].

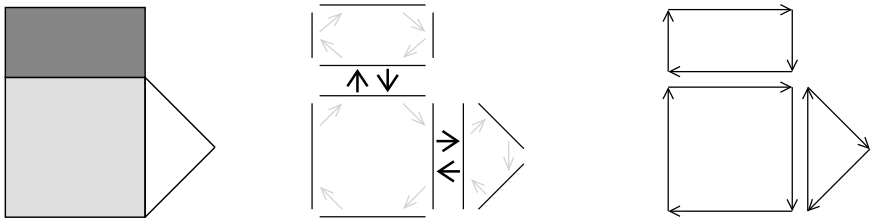
2.1 Combinatorial Maps

Combinatorial maps are a mathematical model of representation of space subdivisions in any dimension. They were introduced in the sixties by [9], at first as a planar graph representation model, and extended by [14] in dimension n to represent orientable or not-orientable quasi-manifold. Combinatorial maps encode space subdivisions and all the incidency relations. They are made of abstract elements, called *darts*, on which are defined application, called β_i. We are giving here the combinatorial map definition in n dimensions, that we can find for example in [15].

Definition 1 (combinatorial maps). *Let $n \geq 0$. A n combinatorial map, (or n-map) is an $(n+1)$-uplet $M = (B, \beta_1, \ldots, \beta_n)$ where :*

1. *B is a finite set of darts;*
2. *β_1 is a* permutation *on B;*
3. *$\forall i$, $2 \leq i \leq n$, β_i is an* involution *on B;*
4. *$\forall i$, $1 \leq i \leq n-2$, $\forall j$, $i+2 \leq j \leq n$, $\beta_i \circ \beta_j$ is an* involution.

In this definition, there is an application β_i for each space dimension which puts in relation two i-dimensional cells. When two darts are linked with β_i, we say that they are *β_i-sewed*. Each space cell is implicitly represented by a set of darts. We can see figure 1.*a* an example of an image and figure 1.*b* the corresponding combinatorial map. Each dart is represented by a segment, the β_1 relation by light grey arrows and the β_2 relation by dark grey arrows. β_1 put in relation a dart and the next dart of the same face. For example, the light grey face of the image is represented by four β_1-sewed darts in the map. The adjacency between this face and the dark grey face is represented by two darts β_2-sewed together. We are using the simplified representation (figure 1.*c*) that does not represent explicitly the applications, because it is more understandable.

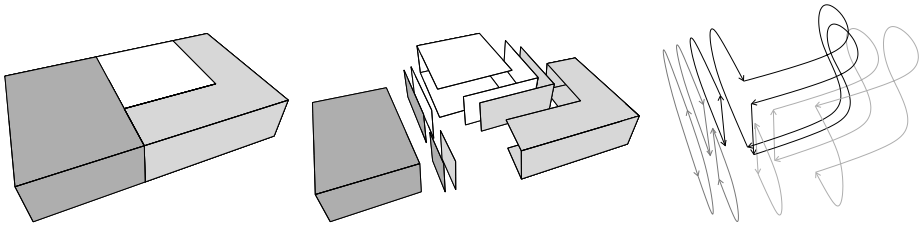


a. A 2D object. b. The corresponding map. c. Simplified representation.

Fig. 1. An object and the corresponding combinatorial map represented by two different way.

2.2 The Topological Maps

In the combinatorial maps framework, several representations of a single object exist. We want a unique characterization of objects, for example to make easier isomorphisms. That is the main goal of the topological maps. They are mathematical model of 3D segmented images representation, which encode all incidency and adjacency relations. They represent interpixel elements composing the edge of boundary faces of an 3D image. Moreover, they are minimal, stable for rigid transformations, and they characterize the image's objects with their topology. We remind that a boundary face is a surface between two neighbouring

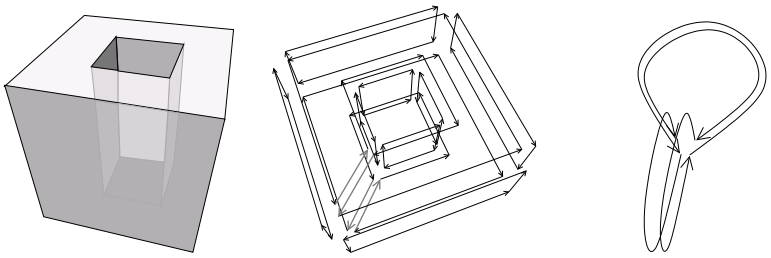


a. A 3D image b. Its boundary faces. c. The topological map.

Fig. 2. A 3D image and the corresponding topological map.

regions. We can see on figure 2.*a* an example of a 3D image, and on figure 2.*b* its boundary faces. The construction of a topological map is progressive : at the beginning all image's voxels are encoded with a combinatorial map, then we simplify this map with successive mergings. The corresponding topological map of our example is shown on figure 2.*c*. We can see that each face of the map encodes a boundary face of the image.

Inclusions of volumes are represented by a *inclusion tree* where each node corresponds to a region and its son nodes correspond to included regions. The root is R_0, the infinite region which rounds up the image. When there is a hole on a face, the connexion between exterior and interior borders of this face is represented with a *virtual edge*. This is a one degree edge (that is adjacent twice to the same face). Such an edge should have been removed during the construction of the topological map, but have been retained to conserve the map connected, and each face homeomorphic to a topological disk. These edges are also useful to represent closed faces (without border), as we can see for the torus example shown figure 3.



a. A torus. b. Intermediary step. c. The topological map.

Fig. 3. The topological map of a torus.

This torus is represented with a single closed boundary face. An intermediate step of the topological map's construction is shown on figure 3.*b*, where we can

see two virtual edges (in grey) which keep the upper and lower faces connected. The topological map shown figure 3.c is only composed of virtual edges. We obtain the classic minimal representation of the torus composed of one face, one vertex and two edges.

Combinatorial maps encode only the topological part of our model. To encode the geometry of the corresponding image, we are using a geometrical model, which links a geometrical face to each topological face of the map . We are calling *embedding* this geometrical model. This distinction between topology and embedding allows the differentiation of treatments and sometimes enables a hierarchization of these treatments. Indeed, some operations only work on the topological model, others only on the geometrical model, and some on both. We can see figure 4.a the topological map of two adjacent objects and figure 4.a the embedding of these objects. Each dart of the topological is linked with the border

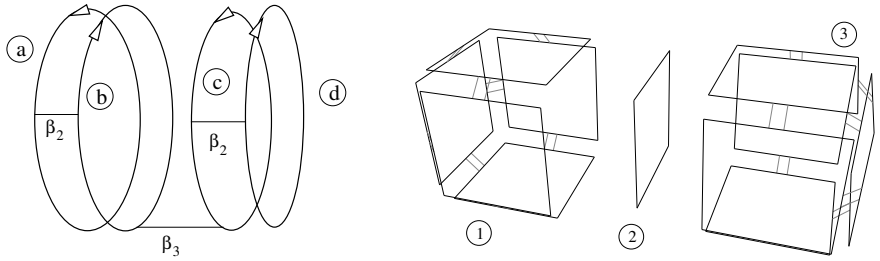


Fig. 4. Embedding example of a topological map.

of an embedding surface. For our example, the dart named a of the topological map is linked with the surface named 1, the two darts named b and c are linked with the same surface 2 because they belong to the same topological face, and the dart named d is linked with the surface 3.

3 Merge

The merging operation consists, starting from two adjacent regions R_1 and R_2, to gather them into a single region R', union of the two first regions. Algorithm 1 make this operation on a topological map. It is local because it treats independently each dart of R_1. There are two different cases depending if the current dart belongs to a virtual edge or not.

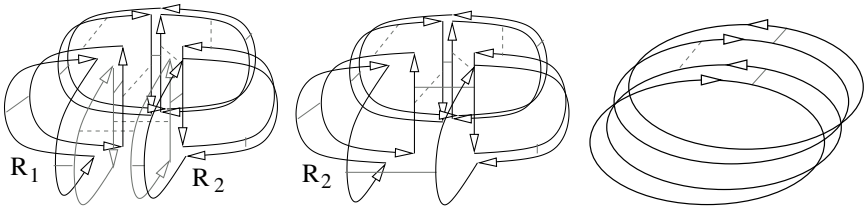
In the first case, we are testing if the deletion of this edge disconnects the map into several connected components. If it happends, we are modifying the inclusion tree this way. In the second case, we are just sewing the two faces adjacent to the currently treated edge together, and we are deleting this edge. This operation is made for all the darts of the boundary faces, so the two regions are merged and boundary faces are finally destroyed. At last, the topological map is simplified,

Algorithm 1 Merge 3D

Data: Two adjacent regions R_1 and R_2.
Result: The two regions are merged into R_2.
foreach *dart b of the region* R_1 **do**
  if *b belongs to the boundary between* R_1 *and* R_2 **then**
    if *b belongs to a virtual edge* **then**
      foreach *dart t of the orbit* $< \beta_1, \beta_2 >$ *of* $\beta_1(b)$ *and* $\beta_{13}(b)$ **do**
        if *all regions of* $\beta_3(t)$ *are included into* $R_1 \cup R_2$ **then**
          Daughter(R_2)=Daughter(R_2)+regions of the connected component of $\beta_3(t)$
      Destroy the virtual edge incident to b;
    else
      β_2-sew($\beta_2(b)$, $\beta_{32}(b)$);
      Destroy $\beta_3(b)$ and b;
Simplification of $R2$;

because operations could have made the map incoherent or not minimal. For that, first we are removing degree 2 vertex and edges. Then, as virtual edges have just a topological existence, they can be moved to reduce the edge number. These simplification step guarantee the minimality of the topological map. Moreover, during this step, we are using the Euler characteristic to keep invariable all topological characteristics of the map.

We can see figure 5 an example of the merging operation. Figure 5.*a* shows a topological map that represent three adjacent objects, before the merging of R_1 and R_2. Figure 5.*b* represents the map obtained after the merging. We have destroyed the face between R_1 and R_2 and β_2-sewn each other darts incident to this face. We can see on this figure that this map is not a topological map because



a. A topological map. b. After the merging. c. After the simplification.

Fig. 5. An example of *Merge* 3*D*

it is not the minimal representation. Indeed, we have two edges incident to only two faces. These edges can be removed without lost of topological information. This is done by the simplification step of our algorithm. After the removal of

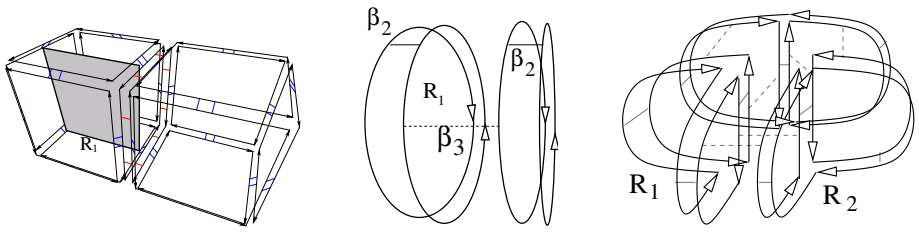
these edges, we obtain some vertices incident to only two different edges. They can be also removed without lost of information. Finally, we obtain the map shown figure 5.*c* that is the topological map representing two adjacent objects. We can note that this simplification step can be performed during the merging, but this leads to a less understandable algorithm, but also with a more efficient complexity. We chose to present here the more simple algorithm, even though its complexity is a little higher, to make easier its comprehension.

4 Split

This operation is the opposite of the merging one. It consists in spliting a region by a *separation face*. This face is composed of a list of vertices which represent the intersections of a plane and the edges of the map. Constraints have been fixed on this separation face to simplify the algorithm and to limit the number of different cases to treat. A simple solution to do complex splits is to combine several simple splits and merge operation to obtain the wanted result. The principle of the algorithm is first to insert a new boundary face into the embedding then the corresponding boundary face in the topological map which represents the region.

There are three different cases :

1. If the separation face of R_1 cuts the edge of an already existing boundary face, like the example shown figure 6.*a*. New vertices will be inserted into this boundary face then they have to be separated with inserted edges between these vertices. At last the new boundary face can be inserted in the middle of the region. It is sewed on this new edge. We can see on figure 6.*c* the resulting topological map of the operation of spliting.



a. Two adjacents cubes. b. The corresponding map. c. After the split.

Fig. 6. Case 1 of *Split* 3*D*

2. If the separation face does not cut any edge of already existing boundary faces (figure 7.*a*). A new face is inserted which is composed by four half-faces. These half-faces are copying of cutted region's faces. Two of this half-faces are linked with an virtual edge because they represent the same surface as it is shown on figure 7.*b*.

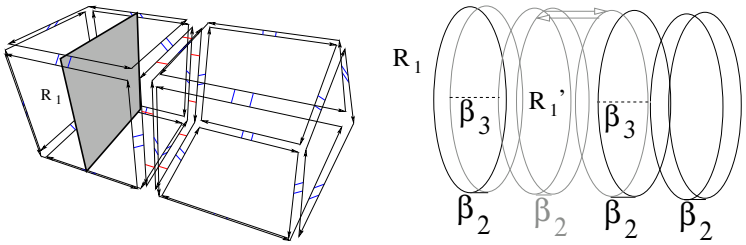
Algorithm 2 Split 3D

Data: A list *List* of vertex representing the separation face
A region R.

Result: R is splited in two regions along the separation face.

foreach *dart b of R* **do**
  foreach *dart t of the embedding of b* **do**
    foreach *vertex S of* List **do**
      InsertVertexPlongement(S,t);
      if *t is on an edge* **then**
        InsertVertexMap(S,b)

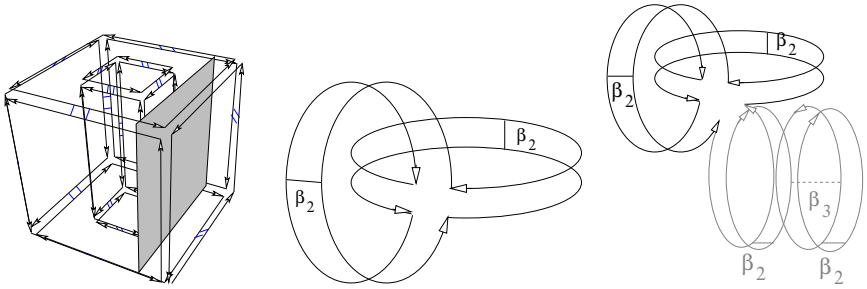
if *the separation face cuts an edge* **then**
  separate the concerned boundary face;
  Create 2 new boundary faces and β_2-sewed them to the previous separated boundary face;
else
  if *the separation face cuts a closed face* **then**
    Create 4 faces composed each by a single dart a, b, c and d;
    Insert a in the closed face;
    β_2-sew(a, b); β_2-sew(c, d); β_3-sew(b, c);
  else
    // The separation face cuts a not closed face F without cutting its edge;
    Create 4 faces F_a, F_b, F_c andF_d copyings of F;
    β_3-sew F_c and F_d and link F_a et F_b with a virtual edge;
    Insert the 4 faces between F and the face which was β_2-sewed to F;
  Simplify the map;



a. Two adjacents cubes. b. After the split.

Fig. 7. Case 2 of *Split* 3*D*

3. If the separation face cuts a closed face. A closed face is only represented with virtual edges in the topological map. There is an example on figure 8.*b* with its embedding in *a*. To obtain the minimal map, the new inserted boundary faces are only composed with one dart and one of this faces has to be directly β_1-sewed to the closed face. The result of the split for a torus is shown on figure 8.*c*. We can verify with the Euler formula that the characterization of the torus is preserved. There is now 1 vertex, 3 edges and 2 faces, which gives a genius of 1.



a. A torus and the separation face.

b. The corresponding map.

c. After the split.

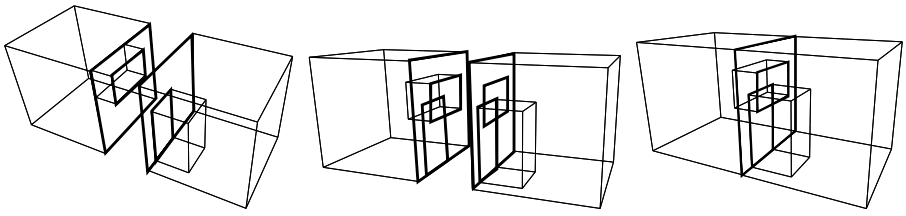
Fig. 8. Case 3 of *Split* 3*D*

5 Corefining

One method to segment a big 3D image consists in cutting it into several parts and segmenting each part in parallel. During the reconstruction of the image, we have to glue back the different segmented parts. The operation which performs this glue back is the corefining. To design the two faces of each volume which are going to be put in contact, we are passing two embedding darts to the algorithm. The faces which contain these darts gives the coordinates of the corresponding planes (as we works with interpixel).

In a first time, we build the new boundary faces. For that, we extract the embeddings of the two faces belonging to the two planes, because we consider that the initial image was cut in regular parts. We are building the map of these new border faces A and B. This construction can be different if the new boundary face cuts a closed face or the border of an existing boundary face, in a similar way that the three different cases explained for the split. We can see figure 9.*a* an example of this first step.

In a second step, we are inserting the intersection vertices of the two faces A and B in the embedding of the two maps of these two faces. These two maps



a. Step 1. b. Step 2. c. Final result.

Fig. 9. Corefining principle.

have now exactly the same vertices, topological as well as geometrical. We are β_3-unsewing and destroying faces belonging to R_0 which are previously β_3-sewn to A and B. Then, for each dart of A, we are looking for a similar dart in B (a dart which connects the same vertices). If such a dart does not exist, we are adding in B a copy of this dart. And we are exactly doing the same operation for each dart of B, with eventually adding some copies in A. Then we are obtaining two faces A and B totally similar, as we can see on figure 9.*b*. We just have to β_3-sew these two faces. For that, we are using the same coverage of the darts of A and B, and β_3-sew each couple of darts. We can see on figure 9.*c* the result of this final operation.

Algorithm 3 Corefining 3D

Data: Two maps M_1 and M_2 and two darts d_1 and d_2 of their embeddings.
Result: The maps M_1 and M_2 are β_3-sewn by the faces containig d_1 and d_2.
Step 1:
foreach *region R of M_1 (resp. M_2)* **do**
  Building the new boundary face A (resp. B) of R;
  Extracting this embeding and set it to A (resp. B);
Step 2:
Inserting intersection vertices of A and B into the faces A and B;
foreach *dart d of A* **do**
  if *it does not exist in B a dart similar to d* **then**
    Add a copy of d in B;
foreach *dart d of B* **do**
  if *it does not exist in A a dart similar to d* **then**
    Add a copy of d in A;
foreach *dart a and b of A and B joining the same vertices* **do**
  β_3-sew(a, b);

6 Conclusion

In this paper, we have presented the basic operations for the 3D segmentation : *merge* and *split* plus an interesting operation in a parallelization goal : *corefining*. These operations are defined on the 3D topological map, a mathematical model of 3D images representation. The two first operations were already presented in [4], but our approach differs from this solution by our aims to define local algorithms. The last operation is very interesting, because it allows us to segment big images by working in parallel onto different small parts of the image. Such a segmentation is very difficult to perform if we want to do in a direct way.

We have totally used all the topological map properties to obtain algorithms the most generic as possible. We are thus obtaining operations working on every possible case. Moreover, these algorithms operate in a local way, which simplifies a lot the different treatments. Indeed, we are treating each element without particular order, and we just have to look at the direct neighborhood of the current element. These different advantages allow us to obtain algorithms relatively simple and understandable, but also to keep a good complexity.

Now, we have to implement these three operations in our computer software, to obtain finally some 3D segmentation algorithms. We are currently working to use these algorithms to perform segmentation refinement. Our approach consists in starting from a first segmentation, performed in a classical way on the voxel matrix with some algorithms used in the signal treatment research. Then, we are computing the topological map corresponding to this first segmentation. We can thus perform some treatments onto this map to refine this segmentation : automatic treatments, for example to remove small regions or particular configurations, or interactifve treatments made by an expert. These results are under development to obtain a software for cerebral tumor diagnostic. Moreover, we also work on some other operations, that could be combination of these three basic operations, but also some specific ones as the chamfering or boolean operations.

References

[1] Y. Bertrand, G. Damiand, and C. Fiorio. Topological encoding of 3d segmented images. In *Discrete Geometry for Computer Imagery*, number 1953 in Lecture Notes in Computer Science, pages 311–324, Uppsala, Sweden, december 2000.

[2] Y. Bertrand, G. Damiand, and C. Fiorio. Topological map: Minimal encoding of 3d segmented images. In *Workshop on Graph based representations*, pages 64–73, Ischia, Italy, may 2001. IAPR-TC15.

[3] J.-P. Braquelaire and L. Brun. Image segmentation with topological maps and inter-pixel representation. *Journal of Visual Communication and Image Representation*, 9(1):62–79, march 1998.

[4] J.-P. Braquelaire, P. Desbarats, and J.-P. Domenger. 3d split and merge with 3-maps. In *Workshop on Graph based representations*, pages 32–43, Ischia, Italy, may 2001. IAPR-TC15.

[5] J.-P. Braquelaire, P. Desbarats, J.-P. Domenger, and C.A. Wüthrich. A topological structuring for aggregates of 3d discrete objects. In *Workshop on Graph based representations*, pages 193–202, Austria, may 1999. IAPR-TC15.

[6] R. Brice and C.L. Fennema. Scene analysis using regions. *Artificial intelligence*, 1:205–226, 1970.

[7] L. Brun. *Segmentation d'images couleur à base Topologique.* Thèse de doctorat, Université Bordeaux I, décembre 1996.

[8] L. Brun and J.-P. Domenger. A new split and merge algorithm with topological maps and inter-pixel boundaries. In *The fifth International Conference in Central Europe on Computer Graphics and Visualization*, february 1997.

[9] R. Cori. Un code pour les graphes planaires et ses applications. In *Astérisque*, volume 27. Soc. Math. de France, Paris, France, 1975.

[10] J.P. Domenger. *Conception et implémentation du noyeau graphique d'un environnement* $2D1/2$ *d'édition d'images discrètes.* Thèse de doctorat, Université Bordeaux I, avril 1992.

[11] C. Fiorio and J. Gustedt. Two linear time union-find strategies for image processing. *Theoretical Computer Science*, 154:165–181, 1996.

[12] S.L. Horowitz and T. Pavlidis. Picture segmentation by a directed split-and-merge procedure. In *Proc. of the Second International Joint Conf. on Pattern Recognition*, pages 424–433, 1974.

[13] C.H. Lee. Recursive region splitting at hierarchical scope views. *Computer Vision, Graphics, and Image Processing*, 33:237–258, 1986.

[14] P. Lienhardt. Subdivision of n-dimensional spaces and n-dimensional generalized maps. In 5^{th} *Annual ACM Symposium on Computational Geometry*, pages 228–236, Saarbrücken, Germany, 1989.

[15] P. Lienhardt. Topological models for boundary representation: a comparison with n-dimensional generalized maps. *Computer Aided Design*, 23(1):59–82, 1991.

[16] R. Ohlander, K. Price, and D.R. Reddy. Picture segmentation using a recursive region splitting method. *Computer Graphics and Image Processing*, 8:313–333, 1978.

[17] M. Pietikainen, A. Rosenfeld, and I. Walter. Split and link algorithms for image segmentation. *Pattern Recognition*, 15(4):287–298, 1982.

[18] P. Resch. Algorithmes pour la manipulation des cartes topologiques en 2 et 3 dimensions. Mémoire de dea, Université Montpellier II, june 2001.

[19] P. Resch. Algorithmes pour la manipulation des cartes topologiques en 2 et 3 dimensions. Annexe technique, Université Montpellier II, june 2001.