

# Low-resolution Surface Mapping: a Topological and Geometrical Approach

Jean-Marie Favreau\*, Vincent Barra†

## Abstract

In this paper we tackle the problem of mapping a surface acquired from a real object onto a piece of the plane, taking into account the topological and geometrical properties of the surface, as well as the specificity of low-resolution acquisitions. We first describe the general background of surfaces unfolding and cutting. We then introduce the cutting process used to manage the topological constraint of the one-to-one mapping, with some speedup improvements. Specific geometrical constraints linked to the low resolution context are detailed, and some experimental results on real and synthetic data are finally presented.

## 1 Introduction

There are several ways to produce 3D acquisitions from a real object. They can mainly be separated into volumic acquisitions, *e.g.* in medical imaging, and point clouds generated from 3D scanners or cameras. Both of the data structures are usually turned into surfaces described by triangles, using isosurface reconstruction methods that ensure a topologically correct reconstructed surface [8], or a sign distance [3]. They can also be fitted with analytical surfaces such as RBF [1]. Due to the precision and resolution, the resulting virtual surface may not capture the full topological and/or geometrical details of the original surface, and are prone to artifacts that misrepresent the data. These errors are common for specific geometric configurations, producing non-existing junctions or handles, disjunctions or hole fillings.

To solve these topological artifacts, volumic approaches can be proposed, such as the use of skeletons [11]. This method thus can only manipulate volumic data, and cannot be influenced by geometrical properties like local curvature. On the other hand, surfacic approaches [10] remove small tunnels, but do not manage geometrical properties of surfaces, like geodesic distances or local curvature. In this paper, the specificity of these surfaces is managed during the manipulation steps, taking into account both topological and geometrical properties of the surfaces.

\*Université Blaise Pascal / LIMOS UMR 6158 Jean-Marie.Favreau@isima.fr

†Université Blaise Pascal / LIMOS UMR 6158 Vincent.Barra@isima.fr

## 2 Background

### 2.1 Topological definitions

Let  $X$  and  $Y$  be two topological spaces.  $X$  and  $Y$  are said to be *homeomorphic* if there is an homeomorphism between these spaces, *i.e.* if there is a bijective continuous function  $f : X \rightarrow Y$ , with a continuous inverse function  $f^{-1}$ . A *2-manifold* or surface  $\mathcal{M}$  without boundary is a topological space locally homeomorphic to a disc. A surface with boundary can also be locally homeomorphic to an half disc. A *non-separating curve* in  $\mathcal{M}$  is a curve that can be drawn on the surface without separating it. A *simple closed curve* in  $\mathcal{M}$  is a curve homeomorphic to a circle. The *genus*  $g$  of a surface  $\mathcal{M}$  is defined as the largest number of non-intersecting non-separating simple closed curves.

### 2.2 2-mesh definition

In the following, we assume that the first process of surface reconstruction has been applied, producing a *2-mesh*  $\mathcal{M}$ , defined as a *2-manifold simplicial 2-complex* (with boundaries or not), *i.e.*  $\mathcal{M}$  is an union of 0-simplices called vertices, 1-simplices called edges and 2-simplices called triangles, and locally homeomorphic to either a disc or an half disc. We call *a boundary* each connected component of the simplicial 1-complex defined by the boundary of  $\mathcal{M}$ . Surfaces are usually classified by their genus and the number of their boundaries. In the following, we assume that the triangulated surfaces are 2-meshes, *i.e.* without local artifacts.

### 2.3 Mapping a surface

An usual approach to visualize the surface of a 2-mesh  $\mathcal{M}$  is to map it to a piece of the plane. It is also a natural approach of UV mapping for texture drawing on complex surfaces. We define a *mapping* between a 2-mesh  $\mathcal{M}$  and a piece of the plane as an one-to-one continuous function  $f : \mathcal{M} \rightarrow \mathbb{R}^2$ . This definition does not allow foldings, and mappings thus only exists for surfaces homeomorphic to a piece of the plane.

Mapping algorithms can be subdivided into fixed-border and free-border methods [7]. Fixed-border methods allow the boundary to be constrained. The resulting map does not take into account the shape

of the boundary, and does not preserve angles or distances. Free-border methods produce only locally bijective mappings: the boundary of the unfolded surface may contain self-crossings on the plane. However, these self-crossings do not forbid a local use of the mapping and if necessary, local arrangements of the mapping may be applied [9]. For practical purposes, the majority of the methods imposes that  $\mathcal{M}$  is homeomorphic to a disc. In section 3, this topological constraint has been relaxed using a cutting step.

## 2.4 Cuttings using paths and loops

Most of the manipulated surfaces are not homeomorphic to a piece of the plane. Unfolding the surface thus make some preprocessings mandatory, reducing the genus and introducing boundaries. These transformations have to be as unnoticeable as possible. The most intuitive method to produce such a transformation is the use of cuttings along paths. A 1-*path* of a 2-mesh  $\mathcal{M}$  is defined as a list of  $n$  oriented edges  $s_0, \dots, s_{n-1}$  where for each  $i = \{0, \dots, n-2\}$ , the terminal vertex of  $s_i$  is the initial vertex of  $s_{i+1}$ . A 1-*cycle* of a 2-mesh  $\mathcal{M}$  is defined as a 1-path  $s_0, \dots, s_{n-1}$  where the initial vertex of  $s_0$  is the terminal vertex of  $s_{n-1}$ . A cutting along a 1-path  $p$  is naturally performed by duplicating vertices and edges of  $p$ , thus removing junctions between triangles adjacent to  $p$ .

## 3 Topological constraints

Generally speaking, the mapping methods compute a transformation from a surface of genus 0 with at least one boundary to a piece of the plane. In practice, methods described in section 2.3 unfold surfaces homeomorphic to a disc, *i.e.* with exactly one boundary. A preliminary topological step is thus needed to be able to apply those unfolding methods.

### 3.1 Cutting the surface

Let  $\mathcal{M}$  be a 2-mesh of genus  $g$  with  $b$  boundaries. As mentioned in section 2.4, the topological transformations have to be as unnoticeable as possible, and cutting is a solution that reduces the genus and the number of boundaries with only small modifications. We call *cutting*  $C$  of a 2-mesh  $\mathcal{M}$  a set of edges whose complement in  $\mathcal{M}$  is homeomorphic to a disc.

Some algorithms have already been proposed to produce such a cutting. Colin de Verdière *et al.* described a topological algorithm [2] producing a minimal cutting in terms of number of 1-paths. However, this cutting only focuses on topological properties of the 2-mesh, and does not handle the geometry. Erickson and Har-Peled [5] computed an almost minimal cutting in terms of cutting's length, using a distance  $d$  on edges. In our application, this method

seems to be the most interesting approach, since it manages topology without ignoring the geometry of the 2-mesh. In [5], authors defined a  $o(n \log(n))$  algorithm to compute the shortest non-separating 1-loop (1-nsloop) starting from a given point  $b$ , called basepoint. The shortest 1-path and 1-loop are computed using the Dijkstra algorithm [4]. Algorithm 1 describes the global cutting algorithm.

```

Data: 2-mesh  $\mathcal{M}$  non homeomorphic to a sphere
while genus of  $\mathcal{M}$   $\neq 0$  do
   $c$  = shortest 1-nsloop;
  Cut  $\mathcal{M}$  according to  $c$ ;
while number of boundaries of  $\mathcal{M}$   $\neq 1$  do
   $c$  = shortest 1-path between 2 boundaries;
  Cut  $\mathcal{M}$  according to  $c$ ;

```

**Algorithm 1:** Cutting a surface into a disc

We used an alternative method to build a better approximation of the optimal cutting, using Algorithm 2 to compute the shortest 1-nsloop.

```

Data: 2-mesh  $\mathcal{M}$  non homeomorphic to a sphere
Result: The shortest 1-nsloop  $l$ 
 $B$  = a complete set of basepoints;
foreach  $p \in B$  do
   $l_p$  = shortest 1-nsloop containing  $p$ ;
  if first step or  $length(l_p) < length(l)$  then
     $l \leftarrow l_p$ 

```

**Algorithm 2:** Finding the shortest 1-nsloop

The main difference between the original method and our approach is the set of basepoints  $B$  of  $\mathcal{M}$ . In [5], this set was reduced to optimize the complexity, but with a loss of precision. In our approach,  $B$  is a set of points such as if a 1-nsloop exists on  $\mathcal{M}$  it should have a point contained in  $B$ . Basepoints can be computed using a growing surface process: a point  $p$  is first selected on the interior of  $\mathcal{M}$ . Triangles of the 2-mesh are then added as a growing surface, preserving the original connectivity and assuming that this wavefront is continually homeomorphic to a disc. When the wavefront stops, some boundaries of the resulting surface may not be boundaries of  $\mathcal{M}$ , and the set of their points is equal to  $B$ . Our approach has a complexity  $O(gn|B| \log n)$ , where  $n$  is the number of points of  $\mathcal{M}$ . Some enhancements of this algorithm can then be considered to improve the practical running time, focusing on the computation of the shortest 1-nsloop.

### 3.2 Using a bounded Dijkstra algorithm

Algorithm 2 computes for each basepoint  $p$  the corresponding shortest 1-loop, and compares its length to the length of the shortest already computed 1-loop  $l$ .

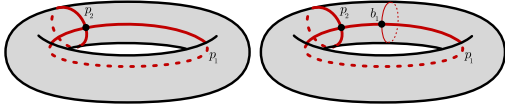


Figure 1: The potential good basepoint selection using shortest paths. Left: The selected basepoints, structured into two paths. Right: The potential good basepoint  $b_1$  selected using the shortest path between the two sides of  $p_1$ .

Paths longer than  $l$  are ignored. Moreover, the shortest 1-loop computation from  $p$  is mainly based on a Dijkstra algorithm: points are ordered according to the result of this algorithm to grow the wavefront.

Using  $l$  as an additional piece of information from a given basepoint, the computation may then be rewritten as follows: *Compute the shortest 1-nsloop  $l_p$  containing  $p$  and shortest than  $length(l)$* . This truncates the Dijkstra algorithm and the wavefront growing stage. The complexity of this modified 1-loop computation is less than  $o(|B|(\log(n) + l_l \log(l)))$  where  $l_l$  is the higher length of the shortest 1-loop starting from a basepoint of  $B$ . For practical purposes, this maximum bound is rather small and this improvement clearly speeds up the 1-loop computation (Table 1).

If this improvement dramatically increases the shortest 1-nsloop computation, it often depends on the order of the basepoints: the sooner the shortest 1-loops are founded, the sooner next searches are truncated, increasing their computation time. We select the first basepoint from each path  $p$  of  $B$  using the shortest path from one side of  $p$  on the complementary of  $B$  to the other (Fig. 1).

### 3.3 Using previous cuttings

Algorithm 1 uses successively several calls of Algorithm 2. After each computation of the shortest 1-loop, the 2-mesh is cut according to this result. The length of the  $i$ st 1-loop will thus be greater or equal than the length of the  $(i - 1)$ st 1-loop. When a junction of the growing surface occurs during the construction of the  $i$ st 1-loop, the length of the 1-loop corresponding to this junction may thus be computed before computing the number of connected components of the complement of the growing surface. If this length is greater than the length of the  $(i - 1)$ st 1-loop, there is no need to compute the number of connected components.

## 4 Geometrical constraints

The cutting method (section 3.1) considers only a part of the geometry by the shortest geodesic distances. Much more geometrical properties of the surface can be handled by enhancing this original method.

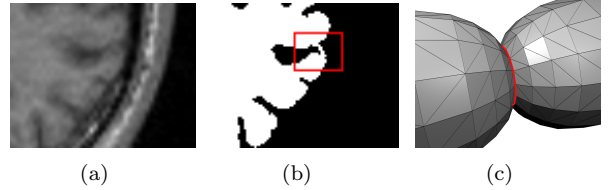


Figure 2: Fig. (a), (b): A topological error (junction between two parts of the surface) after segmentation step. Fig. (c): A cutting path (drawn in red) on the non-zero local curvature area.

### 4.1 Patching the surface

Surfaces mainly originate from low resolution acquisitions. In this context, some topological errors are common for specific geometric configurations. If partial volume effects, undersampling or interpolation artifacts occurs in the original data, some junctions between parts of the surface may be observed (Fig. 2 (a), (b)). To remove these errors, we used a variation of the topological and geometrical algorithm previously described. Given a threshold length  $m$ , this algorithm cuts all the junctions with perimeter smaller than  $m$  and patches the corresponding boundaries by adding small discs to close it. The original boundaries of  $\mathcal{M}$  with perimeter smaller than  $m$  are also patched.

When used before Algorithm 1, this method manages the specificity of the low resolution surfaces and creates some allowable unfolding, with few overlaps (Fig. 3(c)).

### 4.2 Using non-Euclidean distances

The cutting and patching algorithms both use Dijkstra algorithm to select the 1-loops. Thus the only requirement is a non-negative cost function on the edges, seen as a pseudo-distance  $d$  (section 3.1). The obvious cost function is the Euclidean distance  $d_E$  between two vertices in  $\mathbb{R}^3$ , but other choices, guided by application, are also possible.

During the patching process, the algorithm looks for surface junctions. In many applications like medical imaging, junctions are cut along a valley (Fig. 2(c)). Each edge  $e$  of the cutting path takes part in two triangles, and the local curvature can be described by the angle  $\alpha(e) \in [-\pi, \pi]$  between their normals. The Euclidean distance may then easily be modified to address this local curvature. Let  $c \in [0, 1]$ , we define  $d_c(e) = d_E(e)m(e)$ , for each edge  $e$  of  $\mathcal{M}$ , with  $m : e \mapsto 1 - c \frac{|\alpha(e)|}{\pi}$ .  $m$  has the good following property: if  $\alpha(e) = 0$ ,  $e$  takes part in a flat area, and  $m(e)$  should reach the maximum. Otherwise, the highest is  $|\alpha(e)|$ , the smallest  $d_c$ .  $d_c$  thus grants the paths along paths with high local curvature, in particular in the junctions.

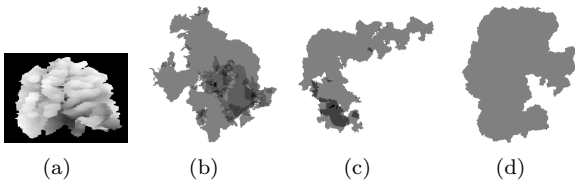


Figure 3: Fig. (a): A neighbourhood of a central sulcus extracted from the original volumic data [6]. Fig. (b), (c), (d): Cortical maps unfolded by Circle Packing after cutting steps (naïve, topological, and patching + topological). Dark grey shows regions of overlappings.

Enhancements	Brain (a)	Synthetic (b)
none	> 5 h 7 mn	7.68 s
sect. 3.2	720 s	0.84 s
all (section 3)	548 s	0.81 s

Table 1: Computation time (on a 2.6 Ghz PC) of the first step of Algorithm 1 according to the improvements of section 3, (a) on a cortical surface [16,057 points, 31,705 triangles], a genus of 56 and 7 boundaries; (b) on a synthetic non-boundary surface of genus 4 [2,202 points, 4,416 triangles].

## 5 Results

A C++ implementation of these modified algorithms has been performed to illustrate their performing on low resolved surfaces. The method has also been applied on classical surfaces, notably with high genus and structures similar to the junctions of the low resolution acquisitions.

In medical imaging, an unfolding approach has been described in a previous work [6], to produce a flattened map of the region of interest on the cortical surface, using T1-weighted Magnetic Resonance Imaging scans. After some segmentation steps and a surface reconstruction, a 2-mesh is obtained, modeling the region of interest on the cortical surface. The cutting method used in [6] did not take into account geometry, producing maps with overlaps for high genus original surfaces (Fig. 3(b)). The approach described in this paper was applied to the 2-meshes stemming from the original data. The resulting surface was then unfolded using classical unfolding tools. Fig. 3(c) and 3(d) illustrate the quality of the generated maps. The cortical map generated using a patching step before the cutting step is clearly more satisfactory.

We applied the cutting method on the cortical mesh using the various improvements of the cutting algorithm (section 3). Table 1 compiles the computation time of the first part of Algorithm 1 using these improvements. If the evaluation of the theoretical computation time of our improvements is hard, Table 1 nevertheless illustrates the significant time-saving ob-

served with our implementation.

## 6 Conclusion

We presented here a topological and geometrical approach to cut the surface before mapping it onto the plane. By improving the complexity of the cutting algorithm, we obtained a useful tool that produces maps without coverings, allowing the user to navigate using the 2D map on the 3D surface thanks to the bijective mapping.

## References

- [1] J. C. Carr, R. K. Beatson, et al. Reconstruction and representation of 3d objects with radial basis functions. In E. Fiume, editor, *SIGGRAPH 2001, Computer Graphics Proceedings*, pages 67–76. ACM Press / ACM SIGGRAPH, 2001.
- [2] É. Colin de Verdière and F. Lazarus. Optimal system of loops on an orientable surface. *Discrete & Computational Geometry*, 33:507–534, 2005.
- [3] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *SIGGRAPH*, pages 303–312, 1996.
- [4] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.
- [5] J. Erickson and S. Har-Peled. Optimally cutting a surface into a disk. In *SCG '02: Proceedings of the eighteenth annual symposium on Computational geometry*, pages 244–253, New York, USA, 2002. ACM.
- [6] J.-M. Favreau, S. Hemm, C. Nuti, et al. A tool for topographic analysis of electrode contacts in human cortical stimulation. In *MMBIA '07*, Rio de Janeiro, Brazil, October 2007. ICCV 2007.
- [7] M. S. Floater and K. Hormann. Surface parameterization: a tutorial and survey. In N. A. Dodgson, M. S. Floater, and M. A. Sabin, editors, *Advances in multiresolution for geometric modelling*, pages 157–186. Springer Verlag, 2005.
- [8] J.-O. Lachaud. Topologically defined iso-surfaces. In *DCGA '96: Proceedings of the 6th International Workshop on Discrete Geometry for Computer Imagery*, pages 245–256, London, UK, 1996. Springer-Verlag.
- [9] A. Sheffer and E. de Sturler. Parameterization of Faceted Surfaces for Meshing using Angle-Based Flattening. *Engineering with Computers*, 17(3):326–337, 2001.
- [10] Z. Wood and I. Guskov. Topological noise removal. *Proceedings of Graphics Interface*, pages 19–26, 2001.
- [11] Q.-Y. Zhou, T. Ju, and S.-M. Hu. Topology repair of solid models using skeletons. *IEEE Transactions on Visualization and Computer Graphics*, 13(4):675–685, 2007.