Topologically-based animation for describing geological evolution

Pierre-François Léon, Xavier Skapin, Philippe Meseure

Topologically-based animation for describing geological evolution

P.-F. Léon – X. Skapin – P. Meseure
Laboratoire SIC - Université de Poitiers
{leon, skapin, meseure}@sic.univ-poitiers.fr

June 25, 2007

Abstract. This paper presents a topologically-based animation system aiming at representing the topological evolution of structured objects over time. Robustness and versatility both relie on the n-dimensional generalised map formalism. The animation is modelled as a series of maps ordered in time and represent each topological modification of the structure. We define several dedicated topological operations that are translated into a script defining the animation. We finally show the usefulness of the approach by means of a specific application in geology, namely the representation of a subsoil evolution in 2D.

Key words: topologically-based animation, generalised maps, geology

1. Introduction

The goal of animation systems is to produce consecutive images representing moving objects. Much work in the litterature has focused on the generation of animation by providing several approaches to create and control motion. Such systems do not intend to represent the history of animated objects, and no time link is established between generated images. However, some applications of animation, mainly in the simulation field, may want to follow the evolution of structured objects. Since usual animation systems fail at providing enough information about the object modifications, these kind of applications require the use of a specific structure allowing a temporal description of relationships between the animation compounds. To our knowledge, there is no animation model based on elementary topological operations in dimension n.

This paper, proposes an animation model based on topological structure. More precisely, we use a boundary representation, where topology is given by the n-dimensional generalised map ("n-G-map") model whereas the temporal embedding is provided by the user. The n-dimensional model is particularly useful since it allows to define both 2D and 3D animations. The temporal structure is represented by successive maps corresponding to the object topological modifications. We therefore define a set of atomic topological operations combined in high-level operations. A script is used to describe the succession of operations. Since these high-level operations are specific to some particular applications, we have investigated the use of our system to describe the history of the section of subsoil, that is, the 2D evolution of various geological layers.

The remainder of the paper is organised as follows: we first present our animation model based on n-G-map and keyframing. We then define some useful low-level operations. Next, we apply our system to the geological field and define some specific high-level operations. Finally, we detail the script creation before concluding and discussing on future work.

2. Animation model

We have chosen a robust topological model for defining our animation system. The n-dimensional generalised map model is defined as [6]:

Def. 2.1. Let $n \geq 0$; a n-G-map is defined by a $(n+2)$-tuple $G = (B, \alpha_0, \alpha_1, \ldots, \alpha_n)$, such that:

- B is a finite, non-empty set of darts;
- $\alpha_0, \alpha_1, \ldots, \alpha_n$ are involutions on B (i.e. $\forall i, 0 \leq i \leq n$, $\forall b \in B$, $b\alpha_i^2 = b$);
- $\forall i \in \{0, \ldots, n-2\}$, $\forall j \in \{i+2, \ldots, n\}$, $\alpha_i \alpha_j$ is an involution.

α_i represents the adjacency and incidence relationships between the i-cells (vertices, edges, surfaces, and so on) of the object.

The main idea of our model is to represent the animation of a structured object as a succession of instantaneous topological changes. We therefore rely on an adaptation of the keyframing approach, where each frame is associated with a map. It also represents all the topological changes of a given time. Between two consecutive frames f_i and f_{i+1}, no topology variation appears and the animation is obtained by interpolating the embedding of the edges.

More precisely, a keyframe is a closed n-G-map that representing a state of the animation at i_n. For all the topological modifications occurring at time i_{k+1}, we search for the last frame i_k. This frame is duplicated and the new frame time is set at i_{k+1}, then this new frame is altered by a set of topological operations (see Fig. 1). This construction methodology implies to have a temporally-sorted collection of topological modifications.

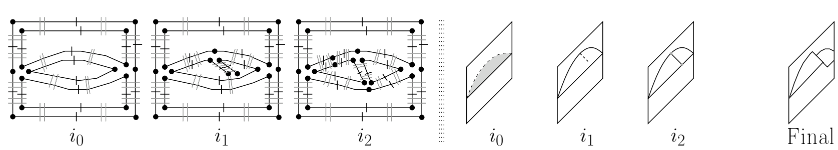


Fig. 1. A keyframing sample. The 2-G-map representation of objects is on the left and their geometrical representation is on the right. The animation steps show an half-circle (time i_1) cut by a segment (time i_2) and the sliding of one part (time i_3). The last image of the animation has the same topology than i_2.

3. Definition of the topological operations

The potentialities of our animation model heavily rely on the topological operations that we can offer. Well-known operations such as edge removing or edge contraction [3] are undoubtedly relevant for our purpose. However some "new" operations are necessary. In particular, Vertex split [5], edge path split and vertex identification must be redefined in our context. Note that other operations could be required for specific animations. Nevertheless, when applying our system to the 2D description of subsoil layer evolution in geology, the previously cited operations have shown themselves sufficient.

The definition of these operations must rely on the G-map formalism. Our methodology consists in building a map in terms of modifications applied to the original map. The historical links must then be established between the darts of the two maps. This process is general and could be applied to all the required new topological operations.

3.1. Vertex split

A vertex split separates a set of edges incident to a vertex into two distinct subsets of edges (see Fig. 2(a)). We consider two specific edges in the set, called a_1 and a_2. They delimit one subset (shown in dark), while the other subset contains all the other edges (shown in gray). Splitting the vertex results in two vertices incident to each subset (see Fig. 2(b)).

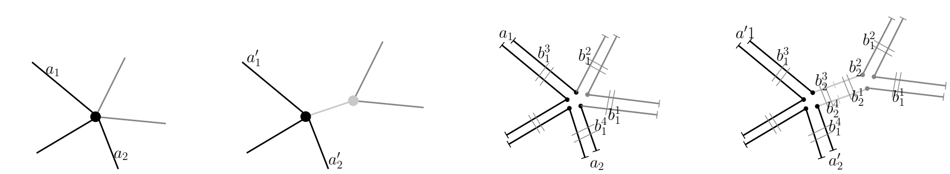
Def. 3.1. Let $G = (B, \alpha_0, \alpha_1, \alpha_2)$ be a 2-G-map. Let b_1^1, b_1^2, b_1^3 and b_1^4 be darts in B such as:

- $\exists p > 0$, $b_1^1 = b_1^2(\alpha_2\alpha_1)^p\alpha_2$; (3.1a)
- $b_1^3 = b_1^2\alpha_1$, $b_1^4 = b_1^1\alpha_1$.

In dimension 2, a vertex split generates a 2-G-map $G' = (B', \alpha'_0, \alpha'_1, \alpha'_2)$ such as:

- $B' = B \cup \{b_2^1, b_2^2, b_2^3, b_2^4\}$; (3.1b)
- $\forall b \in B$, $b\alpha'_0 = b\alpha_0$, $b\alpha'_2 = b\alpha_2$;
- $\forall b \in B - \{b_1^1, b_1^2, b_1^3, b_1^4\}$, $b\alpha'_1 = b\alpha_1$;
- $b_2^1\alpha'_0 = b_2^4$, $b_2^2\alpha'_0 = b_2^3$, $b_2^1\alpha'_2 = b_2^2$, $b_2^3\alpha'_2 = b_2^4$;
- $\forall i \in 1, \ldots 4$, $b_2^i\alpha'_1 = b_1^i$.

In definition (3.1a), darts b_1^1 and b_1^2 are bounding one subset of edges. In definition (3.1b), darts b_2^1, b_2^2, b_2^3 and b_2^4 are created by the vertex split operation.



(a) Before the vertex split. (b) After the vertex split. (c) Before the vertex split (topological view). (d) After the vertex split (topological view).

Fig. 2. A vertex split ($2D$).

3.2. Edge path split

An edge path split creates a new face inserted between several faces (see Fig. 3). It corresponds to a face-adding operator. An edge path is a sequence of edges selected along the boundary separating several faces.

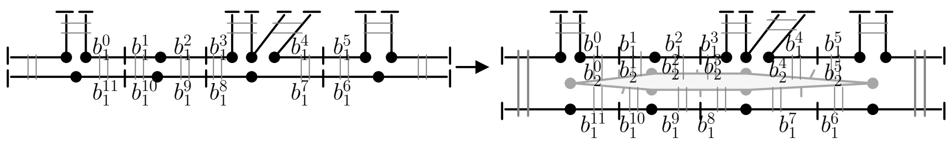


Fig. 3. An edge path between darts b_1^0 and b_1^5 is split to give the gray face.

Def. 3.2. In dimension 2, an edge path $\mathcal{C}$ is a sequence of darts $(b^0, b^1, \ldots, b^{2k+1})$ such as $b^{2p+1} = b^{2p}\alpha_0$ and $b^{2q} = b^{2q-1}(\alpha_1\alpha_2)^m\alpha_1$ with $1 \le q \le k$ and $m \ge 0$.

Def. 3.3. Let $G = (B, \alpha_0, \alpha_1, \alpha_2)$ be a 2-G-map. Let $\mathcal{C} = \{b_1^0, \ldots, b_1^{2k+1}\}$ be an edge path and $\mathcal{C}' = \{b_1^{2k+2}, \ldots, b_1^{4k+3}\}$ its image by α_2 such as:

- $\mathcal{C} \subset B$, $\mathcal{C}' \subset B$ and $\mathcal{C} \cap \mathcal{C}' = \varnothing$;
- $\forall i \in [2k+2 \ldots 4k+3]$, $b_1^i = b_1^{i-(2k+2)}\alpha_2$.

In dimension 2, the edge path split generates a 2-G-map $G' = (B', \alpha'_0, \alpha'_1, \alpha'_2)$ such as:

- $B' = B \cup \{b_2^0, \ldots, b_2^{4k+3}\}$; (3.1a)
- $\forall b \in B$, $b\alpha'_0 = b\alpha_0$ and $b\alpha'_1 = b\alpha_1$;
- $\forall b \in B - (\mathcal{C} \cup \mathcal{C}')$, $b\alpha'_2 = b\alpha_2$;
- $b_2^{2a}\alpha'_0 = b_2^{2a+1}$ with $0 \le a < 2k+1$;
- $b_2^i\alpha'_2 = b_1^i$ with $0 \le i \le 4k+3$;
- $b_2^{2a+1}\alpha'_1 = b_2^{(2a+2) mod (4k+4)}$ with $0 \le a \le 2k+1$.

In definition (3.3a), darts $b_2^0, \ldots, b_{4k+3}^4$ represent the created face.

3.3. Vertex identification

A vertex identification collapses two distinct non-linked vertices into one.

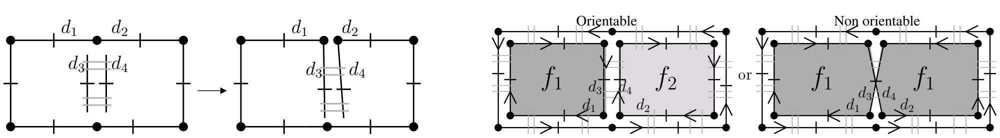


Fig. 4. Vertex identification.

Def. 3.4. Let $G = (B, \alpha_0, \alpha_1, \alpha_2)$ be a 2-G-map and $D \subset B$ a set of darts forming a polyline, let d_3, $d_4 \in D$ and d_1, $d_2 \in B - D$ such as:

- $\forall d \in D, d\alpha_2 \ne d \wedge d\alpha_0 \ne d \wedge (d(\alpha_1\alpha_2)^2 = d \vee d\alpha_1\alpha_2 = d \vee d\alpha_1 = d)$; (a)
- $d_1\alpha_1 = d_2 \wedge d_3\alpha_2 = d_4 \wedge (d_3\alpha_1 = d_3 \vee d_3\alpha_1\alpha_2 = d_3)$. (b)

In dimension 2, identifying d_1 and d_3 generates a 2-G-map $G' = (B, \alpha_0, \alpha'_1, \alpha_2)$ such as:

- $\forall d \in B - \{d_1, d_2, d_3, d_4\}$, $d\alpha'_1 = d\alpha_1$;
- $d_1\alpha'_1 = d_3, d_2\alpha'_1 = d_4$.

Definition (3.4a) defines a polyline. In definition (3.4b), d_3 and d_4 respectively represent one free polyline extremity. Figure 4(b) shows that vertex identification can generate a non-oriented object after the last identification step. To ensure that the resulting object is well-oriented, the propriety $d_3 \in< \beta_1 > (d_1\alpha_0)$ with $\beta_1 = \alpha_0\alpha_1$ has to be true.

4. Application to geology

In order to use the previously defined topological operations in a pratical way, we have chosen the field of geological modelling. Several approches in the litterature currently aim at creating an accurate representation of the geological layers of the underground, but all these models are static: None of them defines the temporal evolution of these layers.

To our knowledge, geometric models describing geological Earth subsoil are built from raw data via dedicated softwares like GOCAD [7] and RML [4], but they do not intend to describe the evolution of the geological layers.

Besides, some 3D topological models derived from G-maps are currently used to model geological layers. For example, Schneider [1] describes the layers by a set of parametric surfaces which intersect each other and represent the boundary of geological volumes (called "blocks"). These blocks may have been fragmented during some geological events. After processing, Schneider uses its extended 3-G-map model not only to represent the blocks, but also to link some disconnected blocks belonging to a former single layer before fragmentation. Schneider's works have been extended in [2] with subdivided surfaces. Both models are static (i.e. don't describe any temporal evolution).

In the late nineties, Perrin has defined a geological syntax to describe causes and effects of geological phenomena and their succession ([8]). Those rules ensure geological consistency of the layer geological model. The rules are part of the Geological Evolution Scheme (GES). A GES is an acyclic oriented tree: Nodes represent surfaces or sub-GESs (for level of detail), arcs represent chronological (for example, "surface B is older than surface A") or topological relationships (e.g. "surface B cuts surface A"). Geological rules give some information about node interactions. In short, GES provides topological and anteriority information, but this information is not complete enough to create an animation, since we need temporal data and durations to describe any evolution.

4.1. Modelling geological phenomena with topological operations

There are several kinds of geological phenomena. We have identified four of them: Sedimentation, erosion, fault creation and sliding.

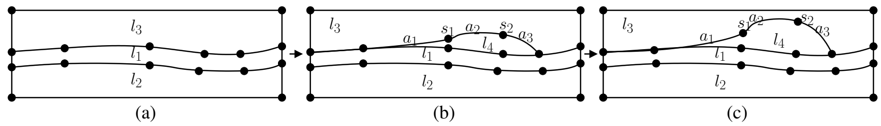


Fig. 5. Exaggerated sedimentation (for explanation purpose). We see the new layer l_4 creation between l_1 and l_3, then l_4 is deformed.

4.1.1. Sedimentation

The sedimentation process consists of two parts. First, a new layer is created, as shown in figure 5. This layer (Fig. 5(b)) generates the creation of new vertices (s_1, s_2) and new edges (a_1, a_2, a_3). This phenomenon uses the edge path split operation. The second step consists in increasing or decreasing the shape of l_4 (Fig. 5(c)). This last operation does not modify the topology.

The sedimentation process can make several layers merge, as shown in figure 6. In this case, topology modification is made up of one contraction (edge a on Fig. 6(a)) followed by a vertex split (the gray vertex on Fig. 6(b)), resulting into a pair of new vertices (s_3 and s_4 in Fig. 6(c)).

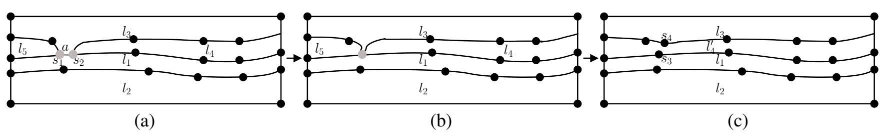


Fig. 6. A sedimentation with layer merging (l_4 and l_5 in l'_4).

4.1.2. Erosion

Erosion is the symmetrical case of sedimentation. Erosion removes sediments and can lead to layer destruction.

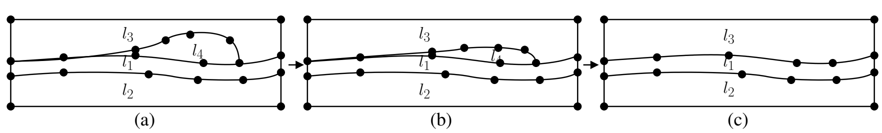


Fig. 7. l_4 layer erosion.

Figure 7 shows an erosion sample. The shrinking of layer l_4 is modelled by a simple deformation of the interface (l_3, l_4) (Fig. 7(a)) and 7(b)). There is no topological modification. But the process can lead to a layer destruction as shown in figure 7(c). Instead of removing the face corresponding to l_4 layer (which is a complex and not well-defined process [3]), we equivalently remove the edge path representing the interface between l_3 and l_4 and thus merge the two faces l_3 and l_4.

4.1.3. Fault

A fault is a (partial or complete) breaking of a layer. It is modelled by a polyline which extends during its evolution. The starting vertex of a fault either belongs to the upper layer interface (faults F_1 and F_2 in Fig. 8(b)), or is inside a layer (fault F_1 in Fig. 8(c)). In the second case, a fictive edge[1] is used for linking the starting vertex and some vertex of the upper layer interface.

The fault evolution ends when its pending extremity stops on another interface and breaks the layer in two parts (F_1 in Fig. 8(b)) or inside the bottom layer (F_2 in Fig. 8(b)).

We then use vertex identification each time a fault extremity is linked with a layer interface vertex.

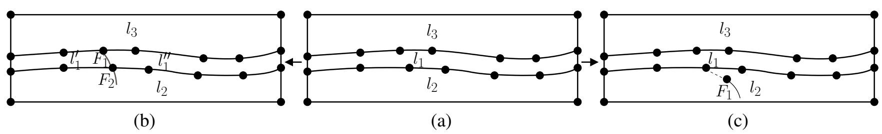


Fig. 8. A fault creation. (a-b) F_1 splits the layer, F_2 is inside a block. (c) Another fault creation: F_1 is not linked with the boundary of l_2 so we use a fictive edge between F_1 and interface (l_1, l_2).

4.1.4. Sliding

Sliding describes the movement of one or several geological blocks along a fault. For example, Figure 9 shows the evolution of interface (l_1, l_4). Two vertices and two edges are created (shown in gray in Fig. 9(b)). At last, removing the contact edge between l_1 and l_4 leads to the disappearance of interface (l_1, l_4).

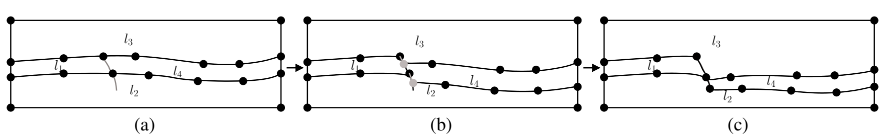
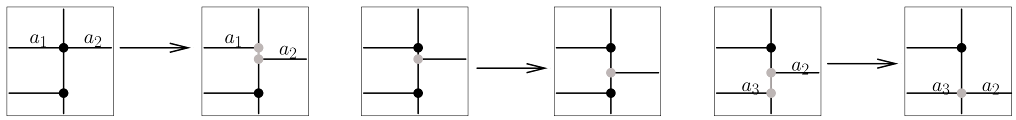


Fig. 9. A sliding of layer l_4 along the fault between l_1 and l_4.

Figure 10 describes the sliding steps in geometrical terms.

[1] A fictive edge is a topological edge with no geometrical embedding.



(a) Vertex splitting. (b) Gray vertex translation. (c) Gray edge contraction.

Fig. 10. Sliding steps.

5. Sample script and results

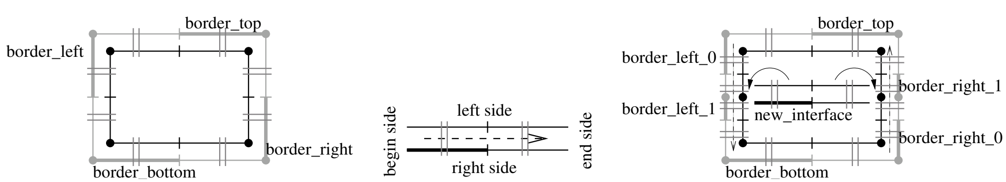
Real geological events often result from combinations of specific geological phenomena, such as those we have presented in this article. In order to control those combinations, we use an animation script.

There are at least two degrees of control for a script: High (end-user) level and low (developer) user. High level control is application dependent: For geological application, each line in the script should be like "layer sedimentation", "layer erosion", and so on. However we want our work to be as independent on the application as possible, so we currently work at low level.

5.1. Writing the animation script

The animation script is a sequence of topological operations and frame duplications. The edge embedding is defined by a function $f(p,t)$, where p is a set of linear coordinates and t is a time. Since we manipulate $2D$ objects, all operations are defined on edges designated by a name. A vertex can be named as well by an extremity of an edge ("begin" or "end") which implies that we use oriented objects. An edge is incident to two faces (the orientation permits us to define left and right sides, as shown later) and incident to two vertices at most.

Figure 11(a) shows an empty frame in the topological view and its associated names. This frame is made of four edges named "border_left", "border_top", "border_right" and "border_bottom".



(a) An empty frame. (b) The orientation is deducted from the tag. (c) Sample construction.

Fig. 11. Tags, orientation and an example.

The example below, describes the steps for building a new interface inside a frame with a sinusoidal embedding.

First, we create a new animation and the first frame (Fig. 11(a)):

```
animation = CAnimation:new(gmv, width, height)
firstFrame = animation:addFirstDiscretePlane2d(0)
```

Next, we split left and right frame borders to identify new vertices with the extremities of a new line named "new_line" (Fig. 11(c)):

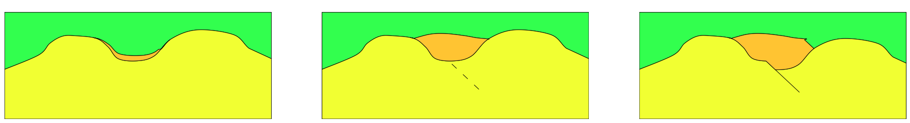
```
firstFrame:createInterface("new_line")
firstFrame:splitInterface({50}, "border_left")  -- Split at the middle (50%)
firstFrame:splitInterface({50}, "border_right") -- Split at the middle (50%)
firstFrame:identification("border_left_1", true, "new_line", true, true)
firstFrame:identification("border_right_1", true, "new_line", false, true)
```

The two first boolean parameters of the identification function refer to edge extremities (true = begin, false = end). The third boolean parameter is the left side parameter as shown in figure 11(b).

Next, we add embedding to "new_line":

```
firstFrame:setSymboleEdgeEmbedding("new_line", function(p, t)
  return width * p, sin(p * pi * 4) * cos(t / 3.5) * 4 + height / 2 end)
```

With combinations of topological operations, we can produce more sophisticated subsoil animation. Figure 12 shows images taken from a script-built animation. At the beginning, a sedimentation layer appears between two "hills". Next, a set of faults appears inside the lowest geological layer. Those faults grow up, and meet to form a single fault, until the fault top extremity comes into contact with the sedimentation layer bottom. This leads to the sliding of the bottom layers.



(a) Sedimentation begins.

(b) Separate faults appear and extend.

(c) Faults merge into a single one and a sliding occurs.

Fig. 12. Some animation steps of simultaneous geological phenomena.

6. Conclusion and future work

In this paper, we have shown a topologically-based animation system. We have defined some low-level topological operations for subsoil evolution modelling. We have presented a methodology to use those operations for animation creation inside scripts. Those scripts allow us to control the animation at each time steps.

Our system need several improvements. First, we have to complete the set of topological operations. We are looking for applications in order to identify other phenomena and analyse them in terms of low-level operations.

We also want to increase animation control by providing a user level depending on the application. It will be necessary to know how to analyse a set of operations defined by the user.

Finally, we want to extend the animation model to 3D. The n-G-map model should make the extension of 2D operations easy but we think there will be new 3D-specific operations to define.

7. Acknowledgments

We want to thank M. Perrin (Ecole des Mines de Paris) and JF. Rainaud (Petroleum French Institute) for their assessment in geography and their help in describing geological phenomena. We also want to acknowledge Pascal Lienhardt for his valuable comments about this work.

References

[1] Y. Bertrand, P. Lienhardt, N. Guiard, S. Brandel, S. Schneider, M. Perrin, and JF Rainaud. Automatic building of structured geological models. Juin 2004. ACM Symposium on Solid Modeling, Gênes, Italie, pp. 59-69.

[2] S. Brandel, S. Schneider, M. Perrin, N. Guiard, J-F. Rainaud, P. Lienhardt, and Y. Bertrand. Automatic building of structured geological models. *JCISEV*, 2004.

[3] G. Damiand and P. Lienhardt. Removal and contraction for n-dimensional generalized maps. In *Discrete Geometry for Computer Imagery*, number 2886 in Lecture Notes in Computer Science, pages 408–419, Naples, Italy, november 2003.

[4] M. Floater, Y. Hallbwachs, O. Hjelle, and M. Reimers. A cad-based approach to geological modelling. Nancy, France, Juin 1998.

[5] H. Helter. *Étude de structures combinatoires pour la représentation de complexes cellulaires*. PhD thesis, Université Louis Pasteur, 1994.

[6] P. Lienhardt. Subdivision of n-dimensional spaces and n-dimensional generalized maps. In *Annual Symposium on Computational Geometry SCG'89*, pages 228–236, Saarbruchen, Germany, June 1989. ACM Press.

[7] J.-L. Mallet. Geomodeling. *Oxford University Press*, 2002.

[8] M. Perrin. Geological consistency : an opportunity for safe surface assembly and quick model exploration. *3D Modeling of Natural Objects, A Challenge for the 2000's*, Juin 1998.