2D Topological Map Isomorphism for Multi-Label Simple Transformation Definition

Guillaume Damiand, Tristan Roussillon, Christine Solnon

2D Topological Map Isomorphism for Multi-Label Simple Transformation Definition*

Guillaume Damiand, Tristan Roussillon, and Christine Solnon

Université de Lyon, CNRS, LIRIS, UMR5205, INSA-Lyon, F-69622 France

Abstract. A 2D topological map allows one to fully describe the topology of a labeled image. In this paper we introduce new tools for comparing the topology of two labeled images. First we define 2D topological map isomorphism. We show that isomorphic topological maps correspond to homeomorphic embeddings in the plane and we give a polynomial-time algorithm for deciding of topological map isomorphism. Then we use this notion to give a generic definition of multi-label simple transformation as a set of transformations of labels of pixels which does not modify the topology of the labeled image. We illustrate the interest of multi-label simple transformation by generating look-up tables of small transformations preserving the topology.

Keywords: Combinatorial maps; 2D topological maps isomorphism; labeled image; simple points; simple sets.

1 Introduction

Image processing often needs to group pixels into clusters having some common properties (which can be colorimetric, semantic, geometric...). One way to describe these clusters is to use labeled images where a label is associated with each pixel. Given two labeled images, it is interesting to be able to decide if they have the same topology. This can be useful for example for object tracking or image analysis. A related question is to decide whether a labeled image can be transformed into another one while preserving the topology of the partition. This second question is interesting to propose a deformable model with a topological control for example. In this paper, these two problems are addressed.

Many data-structures were proposed to describe the topology of labeled images. A well-known one is the Region Adjacency Graph (RAG) [12]: the image is partitioned into regions corresponding to sets of connected pixels; the RAG associates a node with every region and an edge with every pair of adjacent regions. RAGs are used in different image processings like image segmentation [14] or object recognition [9]. However a RAG does not fully describe the topology of

* Paper published in Proceedings of 18th International Conference on Discrete Geometry for Computer Imagery, LNCS 8668, pp. 39-50, September 2014. Thanks to Springer Berlin Heidelberg. The original publication is available at `http://dx.doi.org/10.1007/978-3-319-09955-2_4`

a partition in regions: it does not represent multi-adjacency relations nor the order of adjacent regions when turning around a given region. Thus two partitions having different topologies may be described by isomorphic RAGs.

2D topological maps [3] are more powerful data structures for describing the topology of subdivided objects as they fully describe the topology of labeled images. They combine 2D combinatorial maps (2-maps [10]), describing the topology of subdivided objects (multi-adjacency relations as well as the order of adjacent regions when turning around a given region) with enclosure trees, describing region enclosure relations.

In this paper, we define isomorphism between topological maps and we show that two labeled images are homeomorphic if their associated topological maps are isomorphic. We describe a polynomial-time algorithm for checking whether two topological maps are isomorphic, thus providing an efficient way of deciding whether two labeled images are homeomorphic. Then we use the isomorphism definition of topological maps in order to give a simple definition of multi-label simple transformation (called *ML-simple*), i.e. a set of modifications of labels of pixels that preserve the topology of the whole partition. Lastly we use all these tools in order to generate different look-up tables of ML-simple transformations allowing to test in amortized constant time if two local configurations are equivalent, and allowing to retrieve in linear time in the size of the output all the configurations that are equivalent.

In Sect. 2 we introduce all the preliminary notions on labeled images, combinatorial maps, isomorphism and homotopic deformation. In Sect. 3 we define topological map isomorphism and present the algorithm allowing to test if two topological maps are isomorphic. Section 4 introduces the definition of ML-simple transformation and shows that it is possible to restrict the test of topological map isomorphism on bounding boxes around the modified pixels. In Sect. 5 we present the construction of look-up tables of ML-simple transformations. We conclude and give some perspectives in Sect. 6.

2 Preliminary Notions

2.1 Labeled Images and Partitions into Regions

A 2D *labeled image* is a triple (I_d, L, l), where $I_d \subseteq \mathbb{Z}^2$ is a set of pixels (the image domain), L is a finite set of labels, and $l : \mathbb{Z}^2 \to L$ is a labeling function which associates a label with every pixel. Two pixels $p_1 = (x, y)$ and $p_2 = (x', y')$ are 4-*adjacent* (resp. 8-*adjacent*) if $|x-x'|+|y-y'| = 1$ (resp. $\max(|x-x'|, |y-y'|) = 1$). A k-*path* (with $k = 4$ or 8) is a sequence of pixels such that two consecutive pixels of the sequence are k-adjacent. A set of pixels S is k-*connected* if for each pair of pixels $(p_1, p_2) \in S^2$ there is a k-path from p_1 to p_2 having all its pixels in S.

A *region* in a labeled image $i = (I_d, L, l)$ is a maximal set of 4-connected pixels having the same label. An additional region is defined, denoted *infinite*(i), which is the complement of I_d, i.e., *infinite*$(i) = \mathbb{Z}^2 \setminus I_d$. The set of regions of i, including *infinite*(i), is denoted $regions(i)$ and is a partition of $\mathbb{Z}^2$.

A region R is *enclosed* in another region R' if all 8-paths from one pixel of R to a pixel of *infinite*(i) contains at least one pixel of R'. Region R is *directly enclosed* in R' if there is no region $R'' \neq R'$ such that R is enclosed in R'' and R'' is enclosed in R'. Every region except the infinite one has exactly one direct enclosing region whereas it may have 0, 1 or more directly enclosed regions. Direct enclosure relations may be described by an enclosure tree rooted in *infinite*(i) (see Fig. 1 for an example of a partition).

Digital contours of regions are made explicit by using the *interpixel* topology [8]. In interpixel topology, the cellular decomposition of the euclidean space $\mathbb{R}^2$ into regular elements is considered. *Pixels* are 2-dimensional elements (unit squares), *linels* are 1-dimensional elements (unit segments) and *pointels* are 0-dimensional elements (points). Two linels are connected if they share a pointel in their boundary. A *frontier* between two regions R and R' is a maximal set of connected linels separating pixels belonging to R and R'.

The *boundary* of a region R is the set of linels which separate pixels of R from 4-adjacent pixels not in R. This boundary is partitioned in two sets: the *external boundary* contains the linels separating R from non enclosed regions; *internal boundaries* contain the linels separating R from its enclosed regions. Note that every region except the infinite one has a non empty external boundary whereas it may have an empty internal boundary. The infinite region has an empty external boundary but its internal boundary is not empty (unless $I_d = \emptyset$).

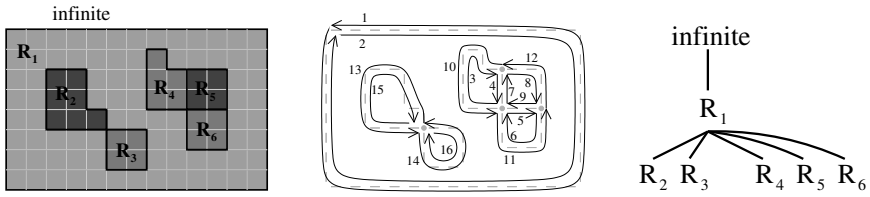
2.2 Combinatorial Maps

Combinatorial maps [10] were defined to describe the subdivision of objects in cells (vertices, edges, faces...) plus the incidence and adjacency relations between these cells. In 2D, a combinatorial map (2-map) can be seen as a graph where each edge is cut in two darts (also known as half-edges). Darts are oriented and two relations are defined on the set of darts: $\beta_1(d)$ is the dart following d when turning around the face which contains d and $\beta_2(d)$ is the dart opposite to d in the face adjacent to the face which contains d. More formally, a 2-map is defined by a triple $M = (D, \beta_1, \beta_2)$ such that D is a finite set of darts, β_1 is a permutation on D, and β_2 is an involution on D. A 2-map is *connected* if for every pair of darts $(d, d') \in D^2$, there exists a sequence of darts $(d_1, \ldots, d_n)$ such that $d_1 = d$, $d_n = d'$, and $\forall 1 \leq i < n, d_{i+1} = \beta_1(d_i)$ or $d_{i+1} = \beta_2(d_i)$.

An example of 2-map is given in Fig. 1(b). This combinatorial map contains 16 darts (drawn by oriented curves). Two darts linked by β_1 are drawn consecutively (e.g. $\beta_1(10) = 11$). Note that a dart may be linked with itself in case of loops (e.g. $\beta_1(1) = 1$). Two darts linked by β_2 are drawn in parallel and have reverse orientations (e.g. $\beta_2(1) = 2$ or $\beta_2(10) = 3$). This 2-map is not connected and is composed of 3 different connected components.

2.3 Topological Maps

A topological map is a combinatorial data-structure which fully describes the topology of a partition into regions of a labeled image. It is composed of three



(a) (b) (c)

Fig. 1. Example of topological map. (a) A labeled image i. (b) The minimal combinatorial map describing i. (c) The region enclosure tree.

parts: a combinatorial map describing the adjacency relations between regions in an ordered way, an enclosure tree describing the direct enclosure relations between regions and an interpixel matrix describing the geometry of the different contours. In this paper, we focus on topology and do not use geometry so that we do not consider the interpixel matrix.

Definition 1 (2D topological map). *Given a 2D labeled image $i = (I_d, L, l)$, its 2D topological map is defined by $TM(i) = (M, T, r, d)$ where:*

- *$M = (D, \beta_1, \beta_2)$ is a 2-map such that each face of M corresponds to a boundary of a region of i, and β_2 describes adjacency relationships between the faces. Furthermore, M is minimal in its number of darts;*
- *$T = (N, E)$ is an enclosure tree of regions: each node of T corresponds to a region of i, and has a child for every region which is directly enclosed in it; the root of T is* infinite(i)*;*
- *$r : D \to N$ associates each dart $d \in D$ with the region $r(d) \in N$ whose boundary contains d;*
- *$d : N \to D$ associates each region $R \in N$ with a dart $d(R) \in D$ which is a dart of the face corresponding to the external boundary of R, except for the infinite region which does not have an external boundary so that d*(infinite(i)) *belongs to its internal boundary.*

Given a 2D labeled image, its topological map is unique (up to isomorphism between 2-maps). Indeed, the 2-map is minimal in number of darts. Thus, each pair of darts $(d, \beta_2(d))$ describes a frontier between two regions R and R' (i.e. a maximal set of connected linels separating pixels belonging to R and R').

In a 2D topological map, each region is represented as a node in the enclosure tree and as face(s) in the 2-map (see example in Fig. 1). Two 4-adjacent regions share a common frontier represented as pair(s) of darts linked by β_2 in the 2-map. In Fig. 1, darts 8 and 12 represent the frontier between regions R_1 and R_5. The relation between two 8-adjacent but not 4-adjacent regions is implicitly represented by a third region, which is 4-adjacent to both. In Fig. 1, the two consecutive darts 13 and 14, linked by β_1, which represent one internal boundary of region R_1, are respectively linked by β_2 to darts 15 and 16, which represent the external boundary of regions R_2 and R_3.

2.4 Isomorphisms and Signatures

Two 2-maps $M = (D, \beta_1, \beta_2)$ and $M' = (D', \beta'_1, \beta'_2)$ are *isomorphic* if there exists a bijection $f : D \to D'$ (called *map isomorphism function*) such that $\forall d \in D$, $\forall i \in \{1, 2\}$, $f(\beta_i(d)) = \beta'_i(f(d))$ [10]. In [7], map signatures are defined such that two connected 2-maps are isomorphic iff their signatures are equal. Given a connected map $M = (D, \beta_1, \beta_2)$ such that $|D| = k$, the signature of M is a sequence of $2k$ integer values $\sigma(M) =< d_1, d_2, \ldots, d_{2k} >$, with $d_i \in [1; k]$ for all $i \in [1; 2k]$. This signature may be used to define the *canonical form* of M: $canonical(M) = (D', \beta'_1, \beta'_2)$ with $D' = \{1, \ldots, k\}$ and $\forall i \in D', \beta'_1(i) = d_{2i-1}$ and $\beta'_2(i) = d_{2i}$ (see [7] for more details).

Two trees $T = (N, E)$ and $T' = (N', E')$ are *isomorphic* if there exists a bijection $f : N \to N'$ (called *tree isomorphism function*) such that $\forall u, v \in N, (u, v) \in E \Leftrightarrow (f(u), f(v)) \in E'$. If the trees are rooted in r and r', respectively, then f must map the roots of the trees, i.e., $f(r) = r'$. If the complexity of graph isomorphism is still an open question, the complexity of tree isomorphism is polynomial and [1] describes an algorithm in $\mathcal{O}(|N|)$ which associates signatures with nodes of the trees. The algorithm of [1] may be extended to integrate labels in signatures: the signature of a leaf is its label; the signature of a node which is not a leaf is the concatenation of its label with the sorted sequence of its children signatures concatenated with the corresponding edge labels.

2.5 Transformations Preserving Topology

In digital topology, the topology of a binary image is usually defined from a pair of adjacency relations (e.g. 4 for one label, 8 for the other) so that a digital version of the Jordan Curve theorem holds [13]. The drawback of this popular solution is that we have two adjacency relations for one partition. Its topology may change if we interchange the two colors, because the chosen adjacency pair is not an intrinsic feature of the partition, but depends on the object to represent. There is no good choice if the topology of the object is intricate (e.g. there are many nested connected components) or if both labels represent regions of interest. This is especially true for images of more than two labels.

In a binary image, a point is *simple* if its label can be changed without changing the connectedness properties of the support of either label [13]. This concept can be extended to labeled images if we independently consider the support of each label and its complement. However, to take into account the adjacency relations between regions, it is proposed in [2] to also consider the union of two labels. A generalization of this idea may be found in [11]. In [6], a new definition of multi-label simple point is proposed to guarantee that the topology of a partition is preserved when a pixel is flipped from a region to an adjacent region. The drawback of these methods is that only elementary transformations are taken into account. However, two partitions can be homeomorphic even if there does not exist a sequence of elementary transformations mapping the two partitions.

In this work, we represent a labeled image by a topological map, which is an intrinsic feature of its partition into regions. Isomorphism between two topological maps, which is equivalent to homeomorphism between two partitions,

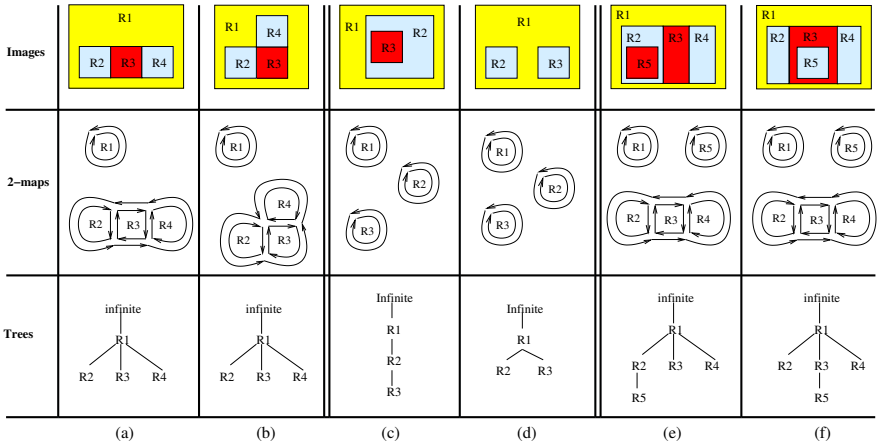


Fig. 2. Examples of non isomorphic topological maps. (a) and (b) have isomorphic trees but not isomorphic 2-maps (R_2 and R_4 are not adjacent in (a) whereas they are 8-adjacent in (b)). (c) and (d) have isomorphic 2-maps but not isomorphic trees. (e) and (f) have isomorphic trees and 2-maps, but isomorphism functions are not compatible.

provides a way of deciding whether a global transformation preserves the topology of the partition or not.

3 Topological Map Isomorphism

In [5], 2-map isomorphism is extended to consider plane isomorphism, i.e. isomorphism between connected 2-maps drawn on the plane. This preliminary definition is extended here to 2D topological maps by considering additional information given by region enclosure trees.

Definition 2 (2D topological map isomorphism). *Two 2D topological maps* $TM = (M = (D, \beta_1, \beta_2), T = (N, E), r, d)$ *and* $TM' = (M' = (D', \beta'_1, \beta'_2), T' = (N', E'), r', d')$ *are* isomorphic *iff: (1) there exists a map isomorphism function* $f_m : D \to D'$*; (2) there exists a tree isomorphism function* $f_t : N \to N'$*; (3) two faces of* M *and* M' *are matched by* f_m *iff the corresponding regions of* T *and* T' *are matched by* f_t*, i.e.,* $\forall d_i \in D$*:* $f_t(r(d_i)) = r'(f_m(d_i))$.

As illustrated in Fig. 2, the three conditions of Def. 2 are necessary to ensure topological map isomorphism. Moreover, Def. 2 leads to the following result:

Theorem 1. *Let* i *and* i' *be two labeled images, and* $TM(i)$ *and* $TM(i')$ *be their associated 2D topological maps.* $TM(i)$ *and* $TM(i')$ *are* isomorphic *iff the partitions into regions of* i *and* i' *are homeomorphic embeddings in the plane.*

Two embeddings are homeomorphic in the plane when they describe the same regions and the same adjacency relations and region enclosure relations.

In [15] the definition of maptree is given. This is a 2D combinatorial map plus a black and white adjacency tree. This tree has one white node for each region of the partition and one black node for each connected component of the combinatorial map (no two adjacent nodes have the same color). The white father (resp. sons) of a black node is the region (resp. regions) corresponding to the external face (resp. internal faces) incident to a same connected component of the combinatorial map. It is thus easy to see that a maptree is equivalent to a topological map.

Proposition 1 in [15] proves that a maptree provides a unique representation (up to homeomorphism of the sphere) of the embedding of a non-connected combinatorial map. Moreover, if the maptree is rooted, the representation is unique up to homeomorphism of the plane. Theorem 1 is a direct consequence of this proposition.

Now, let us describe an algorithm for deciding whether two 2D topological maps are isomorphic or not. Let us consider a 2D topological map $TM = (M, T = (N, E), r, d)$. Each node $R \in N$ of the region enclosure tree corresponds to an image region and is associated with a dart $d(R)$ which belongs to a face of a connected component of M. Let us note F_R the face of M which contains $d(R)$, and M_R the connected component of M which contains $d(R)$. We label the tree $T = (N, E)$ by defining the labeling function λ as follows:

- every region $R \in N$ is labeled with the signature of the connected component of M which contains $d(R)$, i.e., $\lambda(R) = \sigma(M_R)$;
- every edge $(R, R') \in E$ is labeled with the smallest dart of the face corresponding to F_R in $canonical(M_R)$.

The labeling of the region enclosure trees provides a way of deciding whether two 2D topological maps are isomorphic or not:

Theorem 2. *Two 2D topological maps are isomorphic iff their labeled region enclosure trees are isomorphic.*

Indeed, the labels associated with the nodes ensure that two regions mapped by a tree isomorphism function actually belong to isomorphic connected components of the 2-maps; the labels associated with the edges ensure that two regions mapped by a tree isomorphism function are enclosed in regions which correspond to a same face in the canonical form of the corresponding connected components of the 2-maps.

Hence, to decide whether two 2D topological maps are isomorphic we first compute the labeling functions of the region enclosure trees and then use the algorithm of [1] to decide whether the two trees are isomorphic. The time complexity of the construction of the labeling function mainly involves computing the signature and the canonical form of each connected component of the 2-map. This may be done in $\mathcal{O}(k \cdot t^2)$ where k is the number of connected components of the 2-map and t is the maximum number of darts in a connected component (see Property 10 of [7]). The time complexity for deciding whether two labeled trees are isomorphic is linear w.r.t. the number of nodes, i.e., the number of regions.

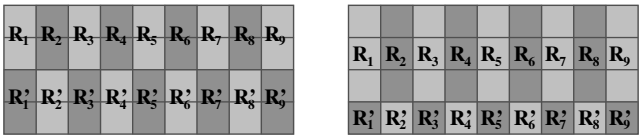


Fig. 3. Two partitions having homeomorphic embeddings in the plane. In the left image, there is no ML-simple transformation of less than 9 pixels. A possible ML-simple transformation is the one modifying simultaneously the labels of the 9 pixels of a same line and giving the right image.

Our contribution here is the definition of 2D topological map isomorphism (which could be also given for maptrees) and the description of a polynomial time algorithm for deciding of isomorphism. The second main contribution of this paper, given in the next section, is the definition of ML-simple transformation.

4 ML-simple Transformation

Given a labeled image (I_d, L, l), the goal is to modify the label of some pixels while preserving the topology of the partition. To achieve this goal, we start to define what is a transformation in a labeled image.

Definition 3 (Transformation). *Given a 2D labeled image $i = (I_d, L, l)$, a transformation T is a set of pairs $\{(p_1, v_1), \ldots, (p_n, v_n)\}$, with $\forall k \in \{1, \ldots, n\}$, $p_k \in I_d$ a pixel and $v_k \in L$ a label. The labeled image obtained by applying transformation T on i is the labeled image $T(i) = (I_d, L, l')$ such that $\forall k \in \{1, \ldots, n\}$, $l'(p_k) = v_k$ and $\forall p \in \mathbb{Z}^2 \setminus \{p_1, \ldots, p_n\}$, $l'(p) = l(p)$.*

Now, thanks to the definition of transformation and the definition of topological map isomorphism, it is straightforward to define the notion of *ML-simple Transformation* (*ML* stands for multi-label). Intuitively a transformation is ML-simple if the two topological maps of the initial image and of the transformed image are isomorphic.

Definition 4 (ML-simple transformation). *Let $i = (I_d, L, l)$ be a 2D labeled image. A transformation T on i is ML-simple if $TM(i)$ is isomorphic to $TM(T(i))$.*

Theorem 1 says that two isomorphic topological maps represent two homeomorphic embeddings in the plane, thus an ML-simple transformation does not modify the topology of the embedding in the plane. Note that a ML-simple transformation that modifies only one pixel corresponds to the notion of ML-simple point [6]. Figure 3 shows that transformation of many pixels could be required in some configurations. In this example, there is no ML-simple transformation of sets having less than 9 pixels.

The definition of ML-simple transformation gives a straightforward way to test if a transformation is ML-simple: this may be done by computing the two

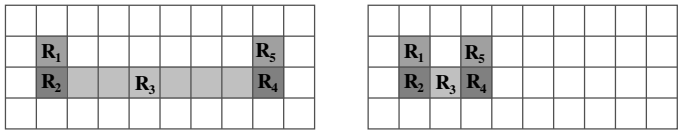


Fig. 4. Two partitions having homeomorphic embeddings in the plane. There is no sequence of flips of ML-simple points allowing to transform the image to the left in the image to the right. However there is a sequence of ML-simple transformations of four pixels which transforms the left image into the right one.

topological maps and testing if they are isomorphic. This simple algorithm can be improved thanks to Theorem 3, which shows that it is enough to compare subimages surrounding the pixels of the transformation.

Given a transformation T on a labeled image (I_d, L, l), let $(I_{d|T}, L, l)$ be the subimage concerned by the transformation. More precisely, $min(I_{d|T}) = (min\{p_1.x, \ldots, p_k.x\} - 1, min\{p_1.y, \ldots, p_k.y\} - 1)$ and $max(I_{d|T}) = (max\{p_1.x, \ldots, p_k.x\} + 1, max\{p_1.y, \ldots, p_k.y\} + 1)$ are the two extremal pixels of the bounding box lying at distance 1 from the extremal points of T. In other words, $(I_{d|T}, L, l)$ is the subimage enclosing all the pixels of the transformation by a border of 1 pixel in thickness.

Theorem 3 shows that it is enough to compare the topological map of the subimage $(I_{d|T}, L, l)$ before and after the transformation in order to test if a transformation is ML-simple.

Theorem 3. *Let $i = (I_d, L, l)$ be a 2D labeled image and T be a transformation on i. T is ML-simple for i iff T is ML-simple for $(I_{d|T}, L, l)$.*

The main argument of the proof uses the fact that the labels of the pixels in the complement of $I_{d|T}$ are not modified by the transformation, nor the pixels in the border of $I_{d|T}$ (because the bounding box has been enlarged by a band of 1 pixel in thickness). Intuitively, this band of pixels around the bounding box guarantees that the frontiers are preserved between $I_{d|T}$ and its complement.

5 Look-up Tables of ML-simple Transformations

One interest of ML-simple transformations is to allow more deformations preserving the topology than ML-simple points. This is illustrated in the example given in Fig. 4. Both partitions are two homeomorphic embeddings in the plane, however there is no sequence of flips of ML-simple points allowing to transform the image on the left into the image on the right. This becomes possible if we use ML-simple transformations instead of flips of ML-simple points.

Deforming a labeled image while preserving the topology of the partition in regions can be done by searching for a ML-simple transformation and applying it. Given a local configuration of pixels, the question is thus to find all the possible ML-simple transformations of the current configuration. Indeed Theorem 3

ensures that the ML-simple transformation test can be restricted to the local window around the modified pixels.

To answer this question in an efficient way, we generated look-up tables of ML-simple transformations of three sizes (s_x, s_y): 1×1 (which is thus equivalent to the notion of ML-simple point), 2×1 and 1×2 (modification of two 4-adjacent pixels) and 2×2 (modification of four pixels forming a square).

To generate these look-up tables, we first generated all the possible labeled images of size $(s_x + 2, s_y + 2)$. We associate to each image i a bitset $b(i)$ having $2 \times (s_x + 2) \times (s_y + 2)$ bits. Bit number $2 \times (x + y \times (s_x + 2))$ is equal to 1 if pixel (x, y) belongs to the same region as pixel $(x - 1, y)$, and bit number $1 + 2 \times (x + y \times (s_x + 2))$ is equal to 1 if pixel (x, y) belongs to the same region as pixel $(x, y - 1)$. Each bitset describes a labeled image up to relabeling of labels, which is what we want since configurations do not consider the value of the labels but only the partition in regions.

The different bitsets are stored in two data-structures. The first one is an array of lists of bitsets, each element of the array being the list of all the bitsets having the same values in their boundaries and having isomorphic topological maps. Each list contains all the possible ML-simple transformations of a given configuration, i.e. all the local configurations belonging to the same equivalence class. The second data structure is an associative array (for example a hash-table) allowing to retrieve, given a bitset, the index of its equivalence class in the array of lists.

Thanks to these two data-structures, it is easy to test if two local configurations (of a given size) are equivalent by computing their two bitsets (by a linear scan of the pixels of the two images) and computing their two indices of their equivalence class. If the two indices are equal, the two configurations are topologically equivalent. Thanks to the hash-table, this can be done in amortized constant time.

Moreover, given a local configuration, we can directly apply successively all the possible ML-simple transformations (of a given size) while preserving the topology of the partition. We first compute the bitset of the local configuration and get the index of its equivalence class. Thanks to this index, we have a direct access to the list of all the bitsets belonging to the same equivalence class, i.e. having the same topology. For each bitset in the list, we can locally modify the labeled image according to the label of pixels not modified by the transformation (belonging to the boundary of the image) and we are sure that the modified image has the same topology as the initial one.

Table 1 shows some information on the generated look-up tables with size 1×1, 2×1 and 2×2. The number of configurations increases quickly as the size of the ML-transformation increases. The number of equivalence classes is significant (between 75% and 84% of the total number of configurations). Indeed there are many non ML-simple configurations. Lastly, we can notice the important time required to compute the look-up tables (about 2 hours for 2×2 pixels), however this generation was done only once as a preprocessing step.

size	nb configs	nb classes	time	max/classe
1×1	1 002	850	0.04s	3
2×1	16 239	13 211	5.4s	6
2×2	756 436	567 728	7042.s	30

Table 1. Generation of look-up tables of all ML-simple transformations of size 1×1, 2×1 and 2×2. *nb configs* is the total number of configurations, *nb classes* the number of equivalence classes, *time* the time spent to compute the look-up tables and *max/classe* the maximal number of elements belonging to the same class.

An upper bound on the number of configurations can be computed. For a transformation of size (s_x, s_y), there are $p = (s_x+2)\times(s_y+2)$ pixels. If we denote by k the number of different labels, each pixel can be labeled with an integer between 1 and k ($k \in \{1, \ldots, p\}$). Thus the number of different configurations with k labels is $\frac{p!}{(p-k)!}$ and the total number of configurations is $\sum_{k=1}^{p} \frac{p!}{(p-k)!}$, that is 986 409 (resp. 1 302 061 344 and 56 874 039 553 216) for size 1×1 (resp. 2×1 and 2×2). We can observe a major difference between these results and the experimental ones because the previous formula gives an upper bound and not exactly the number of configurations. Indeed, different permutations of labels can give the same configuration (i.e. the same partition into regions) while they could be recounted several times in the formula. Note also that it is very difficult to estimate the number of classes, and to give a better upper bound than the total number of configurations.

6 Conclusion

In this paper, we defined the isomorphism between 2D topological maps and showed that two topological maps are isomorphic if and only if the corresponding images are homeomorphic embeddings in the plane. Thanks to this definition, we defined the ML-simple transformation as a set of modifications of labels of pixels that preserve the topology of the partition. Thanks to these notions, we were able to generate different look-up tables of ML-simple transformations for different sizes of modified pixels.

Topological map isomorphism could be helpful in work that need to verify that modifications preserve the topology of the partition. This is for example the case for rigid transformation, where it is possible to use topological map isomorphism as a control tool to check the results of the transformation.

The look-up tables will serve in order to propose more deformations in our framework of deformable partition [4]. We hope that these new deformations will improve the previous results since with ML-simple points, final results of the deformation process are sometimes blocked in local configurations that can not be modified with ML-simple points. Moreover, we want to increase the speed of the simulation thanks to the access in amortized constant time to the elements of the table. Note that it is possible to use directly the topological map isomorphism

in order to test if a transformation is ML-simple. This possibility is particularly interesting in order to use transformations of a large number of pixels.

Our first perspective is the extension of this work to 3D, where topological maps and signatures are already defined. The only problem to solve is the position of fictive edges in 3D topological maps. We must propose a canonical representation for these edges in order to retrieve the property that two maps are isomorphic if the corresponding images are homeomorphic embeddings in the 3D Euclidean space. Our second perspective is the generation of look-up tables with bigger size. The problem is then the memory storage of these tables while keeping an efficient access to all the elements.

Acknowledgement: This work has been partially supported by the French National Agency (ANR), projects DigitalSnow ANR-11-BS02-009 and Solstice ANR-13-BS02-01.

References

1. Aho A.V., Hopcroft J.E., and Ullman J.D. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, 1974.
2. P.-L. Bazin, L. M. Ellingsen, and D. L Pham. Digital homeomorphisms in deformable registration. In *Proc. of IPMI*, volume 20, pages 211–222, January 2007.
3. G. Damiand, Y. Bertrand, and C. Fiorio. Topological model for two-dimensional image representation: Definition and optimal extraction algorithm. *Computer Vision and Image Understanding*, 93(2):111–154, February 2004.
4. G. Damiand, A. Dupas, and J.-O. Lachaud. Combining topological maps, multi-label simple points, and minimum-length polygons for efficient digital partition model. In *Proc. of IWCIA*, volume 6636 of *LNCS*, pages 56–69, Madrid, Spain, May 2011. Springer Berlin/Heidelberg.
5. G. Damiand, C. Solnon, C. de la Higuera, J.-C. Janodet, and E. Samuel. Polynomial algorithms for subisomorphism of nd open combinatorial maps. *Computer Vision and Image Understanding*, 115(7):996–1010, July 2011.
6. A. Dupas, G. Damiand, and J.-O. Lachaud. Multi-label simple points definition for 3d images digital deformable model. In *Proc. of DGCI*, volume 5810 of *LNCS*, pages 156–167, Montréal, Canada, September 2009. Springer Berlin/Heidelberg.
7. S. Gosselin, G. Damiand, and C. Solnon. Efficient search of combinatorial maps using signatures. *Theoretical Computer Science*, 412(15):1392–1405, March 2011.
8. E. Khalimsky, R. Kopperman, and P.R. Meyer. Computer graphics and connected topologies on finite ordered sets. *Topology and its Applications*, 36:1–17, 1990.
9. P. Le Bodic, H. Locteau, S. Adam, P. Héroux, Y. Lecourtier, and A. Knippel. Symbol detection using region adjacency graphs and integer linear programming. In *Proc. of ICDAR*, pages 1320–1324, Barcelona, Spain, July 2009. IEEE Computer Society.
10. P. Lienhardt. N-Dimensional generalized combinatorial maps and cellular quasi-manifolds. *International Journal of Computational Geometry and Applications*, 4(3):275–324, 1994.
11. L. Mazo. A framework for label images. In *Proc. of CTIC*, volume 7309 of *LNCS*, pages 1–10. Springer Berlin Heidelberg, 2012.
12. A. Rosenfeld. Adjacency in digital pictures. *Information and Control*, 26(1):24–33, 1974.

13. A. Rosenfeld. Digital Topology. *The american Mathematical monthly*, 86(8):621–630, 1979.
14. A. Trémeau and P. Colantoni. Regions adjacency graph applied to color image segmentation. *IEEE Transactions on Image Processing*, 9:735–744, 2000.
15. M. Worboys. The maptree: A fine-grained formal representation of space. In *Geographic Information Science*, volume 7478 of *LNCS*, pages 298–310. Springer Berlin Heidelberg, 2012.