University of Hamburg, Dept. Informatik, Technical Report FBI-HH-M-308/0 (December 2001)

XPMaps and Topological Segmentation – a Unified Approach to Finite Topologies in the Plane

Ullrich Köthe

Cognitive Systems Group, University of Hamburg
koethe@informatik.uni-hamburg.de

Abstract: Finite topological spaces are now widely recognized as a valuable tool of image analysis. However, their practical application is complicated because there are so many different approaches. We show that there are close relationships between those approaches which motivate the introduction of XPMaps as a concept that subsumes the important characteristics of the other approaches. The notion of topological segmentations then extends this concept to a particular class of labelings of XPMaps. We show that the new notions lead to significant simplifications from both a theoretical and practical viewpoint.

Introduction

Many computer vision researchers now agree that the notion of finite topological spaces is very useful in image analysis and segmentation, as it allows for consistent descriptions of neighborhood relations, boundaries, and so on. Meanwhile, a number of different methods for the representation of finite topologies has been proposed, including cellular complexes [9, 7], block complexes [10], the star topology [1], the Khalimsky grid [6, 3], combinatorial maps [16, 5] and border maps [2]. All methods approach the problem somewhat differently, but they have a lot in common as well. Unfortunately, the commonalties are not immediately apparent from the literature because most authors present their method in isolation or emphasize the differences to other methods. This makes it unnecessarily difficult to understand and compare the different approaches and to provide reusable implementations in a general image analysis framework.

In this contribution I'm going to propose a unified approach to finite topologies in the plane. I'll show that, under fairly general assumptions, different models can be transformed into each other. This will motivate the introduction of two generalized concepts (the *XPMap* and the *topological segmentation*) which subsume most of the existing concepts. This paper complements a previous paper [8] where solutions for some software design and implementation problems of the new approach were presented. Here, we look at the problem from a theoretical viewpoint in order to show formally why the underlying unification is possible and which properties it has.

Models for Finite Topological Spaces in the Plane

In this section, we will introduce the topological models we are going to analyze later. At first, we will only give definitions but do not yet discuss any relationships between them.

The most fundamental definition of a finite topological space is obtained on the basis of a complete *division of the Euclidean plane*. A plane division is a finite set of disjoint cells whose union completely covers the plane. In particular, there are three types of cells:

Vertices: $V = \{v_1, \ldots, v_n\}$ is a set of distinct points of the plane

Arcs: $A = \{a_1, \ldots, a_m\}$ is a set of disjoint open arcs whose end points are vertices from V. (An arc is a homeomorphic image of the interval [0, 1]. An open arc is an arc less its end points, the images of 0 and 1. The two end points need not be distinct.)

Regions: $R = \{r_1, \ldots, r_k\}$ are the maximally connected components of the complement of the union of V and A. Since $V \cup A$ is a closed set, all regions are open.

The Euclidean topology of the plane induces a neighborhood relation between the cells, which is used to define the smallest open neighborhood (the *open star*) of each cell. The set of open stars forms a basis for a finite topological space:

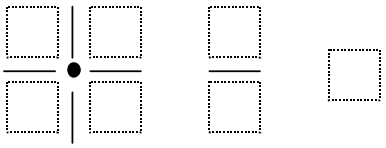

Fig. 1. The open stars of a vertex, arc and region (from left to right) in a square tessellation of the plane

Definition 1: Let T be a division of the Euclidean plane into cells (regions, arcs, vertices). Then the *open star* (smallest open neighborhood) of a cell z is defined as the set of cells that intersect with every Euclidean neighborhood of the points in z:

$$open-star(z \in T) = \{z' \in T \mid \forall q \in z, \forall \varepsilon > 0 : B_\varepsilon(q) \cap z' \neq \emptyset\}$$

($B_\varepsilon(q)$ denotes an open ball of radius ε around the point q). The set of open stars is a basis of a finite topological space over the cells of the plane division.

Two divisions of the plane are topologically equivalent if there exists a 1-to-1 mapping between the cells of the division so that the open stars are preserved. Some authors, e.g. Ahronovitz et al. [1], further restrict the definition by requiring that all regions be convex polytopes. We will not do this in this paper. The most common example for a star topology is obtained by tessellating the plane into equal squares. Then the open stars of the vertices, arcs, and regions look as depicted in figure 1.

The biggest drawback of the plane division definition is its reliance on geometric concepts which prevents separation between topology and geometry. Therefore, a number of definitions not relying on the Euclidean space exist. The best known is the *abstract cellular complex* introduced to image analysis by Kovalevsky [9]:

Definition 2: A cell complex is a triple (Z, *dim*, B) where Z is a set of *cells*, *dim* is a function that associates a non-negative integer *dimension* to each cell, and $B \subset Z \times Z$ is the *bounding relation* that describes which cells bound other cells. A cell may bound only cells of larger dimension, and the bounding relation must be transitive. If the largest dimension is k, we speak of a *k-complex*.

A cell complex becomes a topological space by additionally defining the *open sets* as follows: a set of cells is called open if, whenever cell z belongs to the set, all cells bounded by z do also belong to the set. Kovalevsky proved that every finite topological space can be represented as a cell complex. In image analysis, we want to represent the topological structure of 2-dimensional images, so we are naturally interested in cell complexes whose largest cell dimension is 2, and whose bounding relation is consistent with a division of the plane as defined above. We will call those *planar cell complexes*.

A special case of a cell complex is the *block complex* [10]. It is defined on a given cell complex by grouping cells into sets called *block cells*, or *blocks* for short. Blocks are defined by the property that they either contain a single 0-cell, or are homeomorphic to an open *k-sphere*. (A block is homeomorphic to an open k-sphere if it is isomorphic to a *simply connected* open set in some k-complex – see [10] for details.) A block's dimension equals the highest dimension of any of its cells.

Definition 3: A *block complex* is a complete partition of a given cell complex into blocks. The bounding relation of the blocks is induced by the bounding relation of the contained cells: a block bounds another one iff it contains a cell that bounds a cell in the other block.

Note that the induced bounding relation must meet the requirements of definition 2, i.e. blocks may only bound blocks of smaller dimension, and the relation must be transitive. This poses some restrictions on valid partitions. In particular, all 1-blocks must be sequences of adjacent 0- and 1-cells, and all junctions between 1-blocks must be 0-blocks.

Another approach to finite topological spaces originates from the field of graph theory – the *combinatorial map* [16, 5]:

Definition 4: A *combinatorial map* is a triple (D, σ, α) where D is a set of *darts* (also known as *half-edges*), σ is a permutation of the darts, and α is an involution of the darts.

In this context, a *permutation* is a mapping that associates to each dart a unique predecessor and a unique successor. By following successor chains, we can identify the *cycles* or *orbits* of the permutation. An *involution* is a permutation where each orbit contains exactly two elements. It can be shown that the mapping $\varphi = \sigma^{-1}\alpha$ is also a permutation. [1]

[1] Combinatorial maps are sometimes defined by using four darts per edge ("quad-edges" or "crosses"), e.g. [16]. This allows the realization of non-orientable manifolds and the definition of generalized maps (G-maps) that are easily extended to arbitrary dimensions [12]. However, the half-edge definition suffices in the present context.

The cycles of the σ permutation are called the *nodes* of the map, the cycles of α are the *edges*, and the cycles of φ the *faces*. A combinatorial map is planar (i.e. can be embedded in the plane) if its *Euler characteristic* equals two (cf. [16]):

$$n - e + f = 2 \quad (1)$$

(with n, e, and f denoting the number of nodes, edges, and faces). Each planar combinatorial map has a unique *dual map* that is obtained by reversing the roles of the σ and φ orbits. It should be noted that the graph of a combinatorial map (i.e. the remains of the map when all faces are removed) must be connected, since each face is associated with exactly one orbit. Thus, a combinatorial map cannot represent faces that have holes.

The final approach we are going to deal with is *Khalimsky's grid* [6, 3]. It is defined as the product topology of two 1-dimensional topological spaces. In particular, one defines a *connected ordered topological space* (COTS) as a finite ordered set of points which alternate being open and closed. The smallest open neighborhood of an open point is the point itself, of a closed point it is the point and its predecessor and successor (in case of a closed endpoint, the neighborhood consists of two points only). Now Khalimsky's grid is defined like this:

Definition 5: A Khalimsky grid is defined as the space $X \times Y$ with the product topology, where X and Y are COTS with at least three points each.

The product topology defines the neighborhood of each grid point as $N((x,y))=N(x)\times N(y)$. When x and y are both open, the resulting point (x, y) is also open. When they are closed, (x, y) is closed as well, and its neighborhood consists of the 8 incident points. These two kinds of points are also called *pure*. When an open point is combined with a closed point, a *mixed* point results, whose neighborhood contains the two incident closed points. In other words, we get neighborhoods analogous to the open stars in figure 1, with obvious modifications at the border of the grid.

Khalimsky's grid is interesting on two reasons: first, its regular structure simplifies theorems and algorithms. Second, we can associate *topological coordinates* with the grid points. To do so, we assign numbers to the points of X and Y and interpret each pair of numbers (x, y) as the topological coordinate of the associated grid point. We can now relate the topological coordinates of the open points to the geometric coordinates of the pixels in an image, e.g. by the relation $pixel(i, j) \Leftrightarrow point(2i, 2j)$ if we arrange for the open points to have even coordinates. Thus, we can relate geometric properties to topological entities without giving up the conceptual separation between topology and geometry.

Relationships Between Finite Planar Topological Spaces

In this section we show how the concepts defined above relate to each other by describing how a topological space represented in one concept can be translated into another. Proofs to the theorems can be found in the appendix.

Divisions of the Plane vs. Abstract Cell Complexes

Theorem 1: Any finite topology induced by a division of the plane can be represented by a planar cell complex.

The transformation is achieved by the following algorithm: Associate a 0-, 1- or 2-cell with every vertex, arc, and region respectively. Define the bounding relation so that cell z bounds another cell z' whenever z' is part of the open star of z. In the appendix, we prove that the algorithm actually creates a valid cell complex.

The transformation in the opposite direction is not so simple because not every abstract cell complex corresponds to a division of the plane. We will treat the problem in two steps: first transform the cell complex into a combinatorial map (or rather an XPMap) and second imbed the map into the plane. Both steps are discussed below.

Divisions of the Plane vs. Combinatorial Maps

The transformation of a division of the plane into a combinatorial map is easy as long as the union of the vertices and arcs (the *boundary set* of the division) is a connected set. Otherwise, at least one region's boundary consists of several connected components, which is inconsistent with the requirement that every orbit of the φ permutation corresponds to a distinct face. We will first describe the transformation for the restricted case, and then extend the map definition in order to handle the general case.

Theorem 2: A finite topology induced by a division of the plane can be represented by a planar combinatorial map if the union of the arcs and vertices is connected.

To create a map from a division of the plane, first associate two dart with each arc, one for either direction. If the arc is defined by the function $a(t)$, $0<t<1$, one dart corresponds to $a(t)$ and the other to $a(1-t)$. The vertices at $a(0)$ and $a(1)$ are the start points of the darts respectively. Each pair of darts belonging to the same arc defines an orbit in the α involution. Then, for each vertex take the darts who start at that vertex, and sort them in the order induced by circling the vertex in mathematically positive direction. This gives the orbits of the σ permutation. Under this definition the orbits of $\varphi = \sigma^{-1}\alpha$ indeed correspond to the regions of the plane division as is proved also proved in the appendix.

Two degenerated cases (not covered by this algorithm) require explicit definitions:

- the trivial map: it contains a single empty orbit in the φ permutation and corresponds to the trivial division of the plane that contains only a single region.
- the vertex map: it contains a single empty orbit in both the σ and φ permutations and corresponds to one region that contains an isolated vertex.

In order to handle arbitrary divisions of the plane, including holes in regions, we have to extend the notion of planar combinatorial map. In particular, we must allow the graph of the extended map to be non-connected. Such a map is defined as follows:

Definition 6: An *extended planar map* (*XPMap*) is a tuple (C, c_0, *exterior*, *contains*) where C is a set of non-trivial planar combinatorial maps (the *components* of the XPMap), c_0 is a trivial map that represents the infinite face of the XPMap, *exterior* is a relation that labels one face of each component in C as the exterior face, and *contains* is a relation that assigns each exterior face to exactly one non-exterior face or the infinite face.

Thus, if exterior face f_1 is contained in face f_2, the two faces are actually identical, and the component of f_1 becomes a hole in f_2. Euler's equation must be slightly modified for a XPMap with k non-trivial components (that is, $k=|C|$):

$$n - e + f - k = 1 \qquad (2)$$

Proof: This is easy to prove by induction. The trivial map has $f=1$ and $n=e=k=0$, the vertex map has $n=f=k=1$, $e=0$, and a connected map has $k=1$ and $n-e+f=2$, so the formula is valid for these cases. Suppose we have proved the formula for some k, i.e. $n^{(k)}-e^{(k)}+f^{(k)}-k=1$. Now add a new component with $n^{(1)}-e^{(1)}+f^{(1)}-1=1$. We may add the two equations together: $n^{(k)}+n^{(1)}-e^{(k)}-e^{(1)}+f^{(k)}+f^{(1)}-k-1=2$. It holds that $n^{(k)}+n^{(1)}=n^{(k+1)}$, $e^{(k)}+e^{(1)}=e^{(k+1)}$, but $f^{(k)}+f^{(1)}=f^{(k+1)}+1$, because the exterior face of the newly added component has already been present in the existing XPMap and must not be counted twice. Thus, $n^{(k+1)}-e^{(k+1)}+f^{(k+1)}-(k+1)=1$ as claimed. ■

It should be noted that XPMaps are essentially equivalent to the *border maps* proposed by Bertrand et al. [2] (but their paper lacks a formal definition). Their approach is based on G-maps because they are interested in 2- and 3-dimensional problems. Since this is not needed here, our definition is sufficient.

It is now possible to construct an XPMap from an arbitrary division of the plane: First take each connected component of the divisions boundary set and create the corresponding combinatorial map. These maps become the components of the XPMap. Each component has exactly one φ orbit that is traversed in mathematically negative orientation which becomes the exterior face. Now construct the containment relation according to the inclusion of components within regions of the plane division.

To go in the opposite direction, we must find a plane imbedding of a given XPMap. This is problem is part of the standard problem of imbedding a planar graph, suitable algorithms can be found in [4].

XPMaps vs. Cell Complexes

It is easy to see that any XPMap can be transformed into a cell complex:

Theorem 3: Any XPMap defines a corresponding cell complex.

Proof: For every node, edge, and face of the map create a corresponding 0-, 1-, or 2-cell, respectively. Let a 1-cell bound a 2-cell whenever a dart of the corresponding edge is part of any φ orbit associated with the corresponding face. Let a 0-cell bound a 1-cell whenever a dart of the edge is part of the corresponding node's σ orbit. Let a 0-cell bound a 2-cell whenever its σ orbit contains a dart whose corresponding 1-cell bounds that 2-cell. The bounding relation obviously conforms to definition 2. ■

The bounding relation of the cells induces an equivalent bounding relation of the nodes, edges, and faces. By extension of terminology, and we will say that a node (edge) bounds an edge (face) if the corresponding 0-cell (1-cell) bounds the corresponding 1-cell (2-cell). The bounding relation in turn defines the topology of an XPMap:

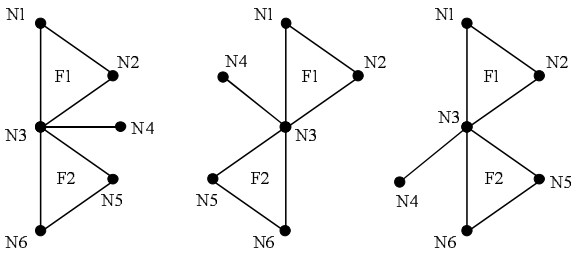

Fig. 2. Three imbeddings of the same cell complex that differ in the ordering of the darts in the σ and φ orbits. Note that the cell complex' graph is only 1-connected.

Definition 7: A set of nodes, edges, and faces in a XPMap is called *open* if the corresponding cells form an open set in the corresponding cell complex. The open star of a node, edge, or face (the smallest open set containing this entity) is the set containing the entity itself and all entities bound by it.

Transformations of cell complexes into XPMaps are more difficult. A 2-complex is not necessarily embeddable into the plane – it might correspond to a non orientable surface or a surface of higher genus. So, some necessary conditions must be met. First, one 2-cell must be denoted the infinite face (this can be chosen arbitrarily). Second, we must require Euler's formula in its second form to be fulfilled (again, k is the number of components of the cell complex' graph, i.e. the remains after removing all 2-cells). Third, all 1-cells must be bounded by one or two 0-cells since all darts must be incident to a node. Unfortunately, its not clear to me whether these requirements are sufficient.

Even when these requirements are fulfilled, the transformation of a 2-complex to a XPMap is not necessarily unique. From graph theory it is known that every 3-connected planar graph has a unique imbedding (up to topological equivalence)[2]. In case of planar cell complexes, a slightly stronger statement can be made: since the faces (including their bounding relation) are already known, ambiguities can be resolved in the 2-connected case. However, if the complex' graph is only 1-connected or disconnected, ambiguities cannot be avoided. In particular, although the incidence between darts and nodes is uniquely determined, the order within each σ orbit is not. Figure 2 shows some examples of different XPMaps (or rather their planar imbeddings) resulting from the same cell complex. An algorithm that produces one possible XPMap out of a given planar cell complex is given in the appendix. This algorithm terminates with no free darts remaining, if the cell complex is planar. The algorithm implies a method for generating a planar imbedding of a cell complex: first transform the cell complex into an XPMap, and then imbed the map into the plane by a standard imbedding algorithm (see [4]). However, although the algorithm may be interesting from a theoretical point of view, its practical value is probably limited: in practice, a cell complex will usually be equipped with additional geometric information that uniquely determines the correct order of the darts in the orbits and the imbedding into the plane.

Khalimsky Grid vs. Combinatorial Map

To allow the establishment of relationships between Khalimsky's grid and the other topological spaces, we must modify the definition of the grid somewhat so that it gets an infinite point that can be mapped to the infinite face of the map.

Definition 8: A *Khalimsky grid with infinite point* is defined as follows: the COTS X and Y constituting the grid must be chosen so that their first and last points are all closed. Then the border of the grid consists only of closed and mixed points. Add an additional point – the infinite point – that by definition belongs to the neighborhood of all border points.

This extension allows formulation of the following theorem:

Theorem 4: Any Khalimsky grid with infinite point defines a corresponding XPMap.

The transformation is obtained as follows: First, we associate a node with every closed point, an edge with every mixed one, and a face with every open one. Then we create two darts for every mixed point, starting at either incident closed point. The pairs form the orbits of the α involution. It is convenient to identify darts by the coordinates of the start point and the relative orientation between the darts's mixed point and start point (east, north, west, and south). This choice immediately yields the σ permutation by taking the darts associated with the same start point and sorting them according to their orientation. This results in the expected φ orbits, as is shown in the appendix.

In contrast, a combinatorial map can only be represented as a Khalimsky grid if it has the required regular structure: except at the exterior border, all nodes and faces must be adjacent to exactly four darts. Since this is a

[2] A graph is k-connected if at least k vertices must be removed to make the graph disconnected.

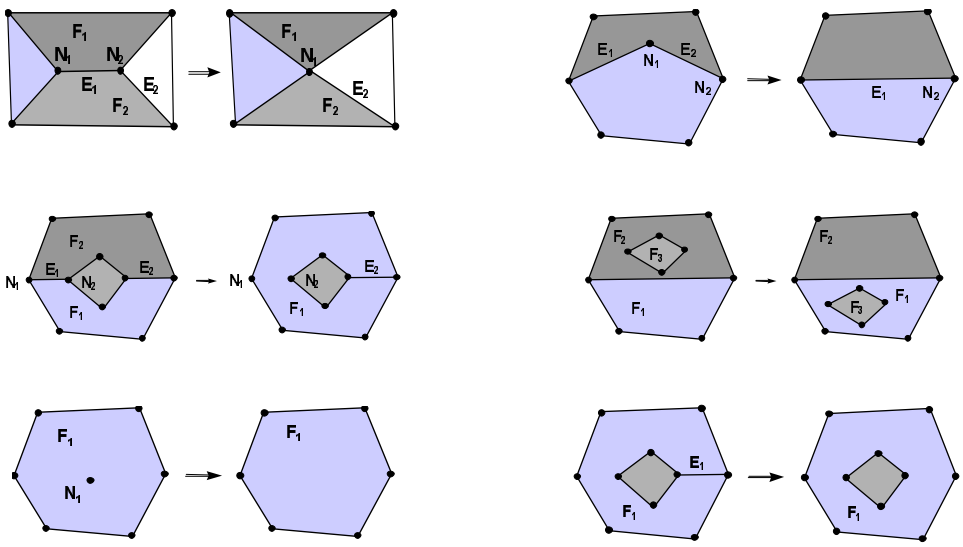

Fig. 3. Examples for Euler operators. first row: edge contraction and merge edges, second row: merge faces and move component, third row: remove isolated node and remove bridge

rather strong limitation, Khalimsky's grid itself is not useful for the representation of arbitrarily shaped topological spaces. Instead, one builds *block complexes* on top of the grid. That is, one creates irregularly shaped cells by grouping the grid points into appropriate blocks. We will discuss this in detail in the section on block complexes.

Euler Operators

Before we can turn our attention to block complexes, we must define and discuss *Euler operators* [13, 7]. Euler operators turn a given planar topological structure into a different on. The term stems from the restriction that Eulers equation must remain valid in the new structure, so that planarity is always preserved (some authors call such operators *superficial*). The proposed Euler operators are elementary in the sense that they change the number of nodes, edges, faces, and components by at most one. They are complete in the sense that any desired transformation can be expressed as a sequence of Euler operations. Completeness follows from theorem 5 below: the entire topological space is always an open set and thus can be reduced to a single face by a sequence of Euler operations. By reversing the sequence, any XPMap can be built out of the trivial map.

Each operator will be formally defined and proved in the context of XPMaps only. The proofs (see appendix) check that Euler's equation indeed remains valid. Analogous operators in other topological models are implied by the transformations already discussed. We will graphically illustrate the effect of each operator on a plane division (figure 3 – space constraints do not allow to give proofs for this representation). Application in the Khalimsky grid is forbidden because the grid's structure is fixed. On the other hand, adaptation of the definitions for cell complexes is straightforward: the changes which the operator causes to the bounding relation are easily inferred from the changes in the map's orbits. We will consider two kinds of operators: first, those that do not change the number of components of the map's graph (so that we can restrict the discussion to standard combinatorial maps), and second the ones that do (and thus require a XPMap). All operators are illustrated in figure 3.

Euler Operators that Preserve the Number of Components

Node Split and Edge Contraction

A node split creates two nodes from one. Some existing edges are moved to the new node, and a new edge connecting the nodes is added. The split must be arranged so that the order of the edges in the orbits is not changed (formal definition below). The inverse operation is called edge contraction, because the two ends of an edge are contracted into a single node.

Definition 9: The operator *node split* is defined as follows: Suppose the node to be split is associated with the orbit $\sigma_1=(d_0,\ldots,d_{n-1})$. Remove σ_1 from the set of orbits and chose darts d_i and d_j $(0\le i\le j<n)$. Split the orbit after these darts into the sequences $(d_{j+1},\ldots,d_i)$ and $(d_{i+1},\ldots,d_j)$ where addition is taken modulo n. If $i=j$, the second sequence is empty. Create a new pair of darts d, d' and insert the pair as a new orbit into α. Add one new dart to either sequence and complete them to the new orbits $\sigma_1'=(d_{j+1},\ldots,d_i, d)$ and $\sigma_1''=(d_{i+1},\ldots,d_j, d')$ which are inserted into the set of orbits. For the inverse operation *edge contraction*, the steps are reversed. The edge to be contracted must not be a self-loop.

By definition, the operator increases the number of orbits in the σ and α permutation by one. The number of components does not change since the newly inserted edge ensures that the graph of the map remains connected. Thus we require $(n+1)-(e+1)+(f+\Delta f)-k=1$, where n, e, f, and k are the numbers of nodes, edges, faces, and components before applying the operator. Because $n-e+f-k=1$ holds for the original map, we get $\Delta f=0$. The proof that the number of φ orbits indeed remains constant is in the appendix.

Edge Split and Edge Merge

Edge split creates two edges by placing a new node into an existing edge. Edge merge reverses this operation. Merging may only be applied to a pair of distinct edges whose common node has degree 2. Otherwise, forbidden configurations would arise.

Definition 10: The operator *edge split* is defined as follows: Suppose the edge to be split consists of the darts d_1 and d_1', and d_1 is part of the orbit σ_1. Create a new pair of darts d_2 and d_2', insert the pair (d_2, d_2') as a new α orbit, replace d_1 by d_2 in σ_1, and insert (d_2', d_1) as a new σ orbit. Edge merge is performed by inverting those steps.

Edge split increases the number of nodes and edges by one and leaves the number of faces and components constant. So Euler's equation is still fulfilled afterwards. The proof of this statement is trivial by observing that edge split is just a special case of node split with $j=i+1$, and d_j is the edge to be split. Nevertheless, its useful to define edge split as an operator of its own because of its special semantic.

Face Merge and Face Split

The face merge operator removes an edge, so that its adjacent faces are merged into one. Face split is the inverse operation. The edge to be removed must bound two distinct faces (i.e. it must not be a *bridge*) as otherwise the number of components would be increased (this case is treated in the next section).

Definition 11: The operator *face split* is defined as follows: Create a new pair of darts d and d' and insert the pair as a new orbit into α. Select two darts d_1 and d_2 which must belong to the same φ orbit. Insert d after d_1 and d' after d_2 into their respective σ orbits. The operator *face merge* is executed be reversing those steps, provided that the edge (d, d') is not a bridge.

This operator increases the number of edges and face by one and leaves the number of nodes and components constant, so that Euler's equation remains again valid. We could prove this explicitly, similar to the proof of node split, but there is a much easier way by observing that face split is the *dual operation* of node split. Face split is equivalent to a node split in the dual map, which immediately implies the proof.

Euler Operators that Involve Several Components of a XPMap

Move Component

This operation is very simple and does not involve any orbits:

Definition 12: The operator *move component* is defined as follows: Take an entry of an XPMap's containment relation and replace the enclosing face by another one (which must not itself be an exterior face).

Since this operation does not add or remove anything, Euler's equation is trivially fulfilled. This operation is especially useful in the context of "split face": if the face to be split contains some components of the XPMap, the definition of the face split operator doesn't tell which of the newly created faces will contain those components afterwards. By means of move component containment can be adjusted without changing the definition of split face.

Insert/Remove Isolated Node

This operation is the simplest possibility to create a new component within an XPMap:

Definition 13: The operator *insert isolated node* is defined as follows: Create a new vertex map. Make its single face the exterior one and associate it with an arbitrary non-exterior face by adding appropriate entries to the XPMap's *exterior* and *contains* relations.

This operation adds a node and a component to the XPMap, but the number of faces (and edges) remains constant since the exterior face of the new component is identified with a face already present in the map. Thus, Euler's equation is not violated.

Remove/Insert Bridge

A bridge is an edge that bounds only one face. That is, both darts belong to the same φ orbit. Thus, removal of the bridge will cut the orbit into two parts, which results in a new component. Here is the formal definition:

Definition 14: To define operator *remove bridge* we must distinguish several cases:

1. The bridge bounds the exterior face of its component. First, remove this component's entry from the *exterior* and *contains* relations, but remember the enclosing face (say f). Suppose the bridge contains darts d and d'. Remove the pair (d, d') from the set of α involutions. Remove d from its σ orbit. If this orbit becomes empty, remove it from the set of σ orbits and create a new vertex map whose single face is marked as *exterior*. If the σ orbit doesn't become empty, its still contains d_1, the σ successor of d. Then $d_1'=\alpha(d_1)$ was the φ predecessor of d. d_1' is part of a newly created component, and the new φ orbit of d_1' becomes its exterior face. Associate the new exterior face with f by a new entry in the *contains* relation. Perform the same steps with d'.
2. The bridge does not bound an exterior face. Suppose the bridge's darts are d and d', and the face is f. Remove the pair (d, d') from the set of α involutions. For both end nodes of the bridge, find the length of the shortest path to any node incident to the exterior face of the present component. The two lengths always differ by one because the path of one end necessarily includes the bridge. Remove d and d' from their σ orbits. Suppose that d was incident to the node with higher distance to the exterior border. If the σ orbit of d has become empty, remove it and create a vertex map that is associated with f in the *contains* relation. Otherwise, let $d_1'=\varphi(d)$ be the φ predecessor of d prior to removal. Then d_1' is part of a new component, and the φ orbit of d_1' becomes its exterior face. Insert the appropriate entry in the containment relation.

As always, the inverse operation *insert bridge* performs these steps in opposite order.

Remove bridge obviously decreases the number of edges by one, while it doesn't change the number of nodes. Since a bridge is necessarily a cut edge of the graph of its map, bridge removal creates a new component. Thus, to fulfill Euler's equation the number of faces must remain constant. This is indeed the case: Although the operator increases the number of φ orbits by one (as is shown in the appendix), it also defines a new exterior face. This face gets identified with an existing face, which leads to the required preservation of face count.

Block Complexes and Topological Segmentation

As can be seen from definition 3, block complexes are defined on top of cell complexes. It would be easy to adapt the definition to the other topological structures we are considering. However, we are not going to do so because the restrictions posed upon the blocks (k-blocks must be homeomorphic to open k-spheres) are not acceptable in the context of image analysis. Applied to 2-blocks, this restriction means that all regions must be simply connected, so holes in regions are not allowed. Applied to 0-blocks, it follows that each 0-block is just a single 0-cell. This means that, if the block complex were built on top of a Khalimsky grid, each 0-block could have at most degree 4. Thus, many interesting and practically important objects could not be represented by a block complex, which is clearly not desirable.

Therefore, we will develop a more general concept for grouping the cells of a digital topological structure which we will call a *topological segmentation*. Informally, a topological segmentation is defined as a complete partition of a digital topological space into *k-segments* (k=0, 1, 2), such that every k-segment could be transformed into a single k-cell by a suitable sequence of Euler operations that do not change the structure of the segment's complement. Before we can give a formal definition we need some preliminaries:

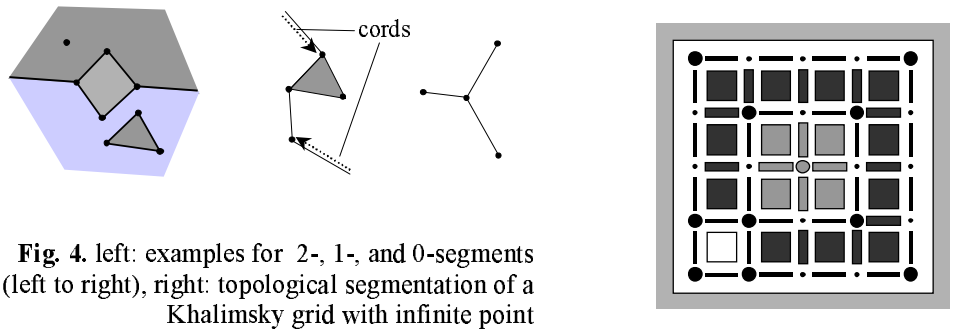

Fig. 4. left: examples for 2-, 1-, and 0-segments (left to right), right: topological segmentation of a Khalimsky grid with infinite point

Theorem 5: Any connected open set in a XPMap can be reduced to a single face by a sequence of Euler operators, without changing the structure of the set's complement. Likewise, any simply connected closed set can be reduced to a single node.

The proof is again in the appendix. Furthermore, we need the notion of *hammocks*:

Definition 15: A *hammock* is the union of a simply connected closed set and two edges emanating from the set such that removal of the two edges would turn the closed set into a new component of the XPMap. An edge emanates from a set if one if its end nodes belongs to the set but not the other. The darts starting at these external nodes are called the *cords* of the hammock.

Hammocks are useful because they can be reduced to single edges by a sequence of Euler operations: first reduce the closed set to a single node. By definition of a hammock, this node must bound the two remaining edges of the hammock, but no other edge. Thus we can apply the operator "merge edges" to get a single edge. We can now define the topological segmentation:

Definition 16: A *topological segmentation* is a complete partition of an underlying XPMap into k-segments (k=0, 1, 2) such that every 0-segment is a simply connected closed set, every 1-segment is a hammock, and every 2-segment is a connected open set.

Figure 4 shows examples of k-segments and a topological segmentation. According to the discussion above, any 0-segment can be reduced to a single node, any 1-segment to a single edge, and any 2-segment to a single face. By reducing all segments we arrive at a new XPMap which we will call the *reduced XPMap* of the topological segmentation. Definition 16 generalizes the notion of a block complex by replacing the restriction "every k-block must be homeomorphic to an open k-cell" by "every k-segment must be reducible to a single k-cell". A topological segmentation may contain 2-segments with holes and 0-segments that have higher degree than any node of the underlying XPMap.

Although every topological segmentation can be reduced to a XPMap, it is not necessary to actually perform the reduction. We can also just label the segments in the underlying XPMap and work with the segments directly. In fact, this is the *only possibility to represent irregular divisions of the plane by means of a Khalimsky grid*: since the grid's structure is rigid, one cannot perform Euler operations on it, but one can still label points into segments.

In order to make a labeled XPMap or Khalimsky grid behave like a reduced XPMap, we must specify the *orbits of the topological segmentation.* To avoid confusion, we will denote these orbits by Greek capitals A, Φ, Σ, in contrast to the lower case α, φ, and σ orbits of the underlying XPMap. The *cords* of the hammocks are the basis of the orbit definitions. These definitions amount to contour following algorithms that go around segments in order to find all incident cords. Efficient and elegant implementations of these algorithms can be obtained by means of *generic programming*, as described in [8]:

A orbits of a topological segmentation: The A orbits associate the two cords of a hammock: given one cord, we want to find the other. To do so, observe that the left face of the first cord is the right face of the second. Thus, we can follow the φ orbit of the first cord until we arrive at a dart that doesn't belong to the present hammock. Its σ-successor is the sought for A successor of the first cord.

Σ orbits of a topological segmentation: Σ orbits associate cords with 0-segments. This is easy if the 0-segment consists of a single node: then the Σ orbit is obtained from the node's σ orbit by simply dropping the darts that are not cords. In case of a more complicated 0-segment, the principle is the same, but instead of just considering a single σ orbit we must now follow the entire contour of the segment. The following algorithm finds the Σ successor of a given cord:

1. Find the σ successor of the present dart. If the successor is a cord, stop. If it belongs to an edge that is part of the 0-segment, go to 2. Otherwise go to 1.

2. Find the α successor of the present dart, and find the σ successor of the resulting dart. If this dart is a cord, stop. If it belongs to an edge that is part of the 0-segment, go to 2. Otherwise go to 1.

Φ orbits of a topological segmentation: The Φ orbits associate cords with 2-segments. The Φ successor of a given cord is found as follows: go to the A successor of the present cord, and find its Σ predecessor. Thus, $\Phi = \Sigma^1 A$.

It can be shown that the orbits of a topological segmentation are equivalent to the corresponding orbits in the reduced XPMap. Furthermore, we can define *Euler operators for a topological segmentation*. They perform the same logical operation as their XPMap equivalent, but do not merge cells. Instead, they re-label the segments in order to reflect the modified structure. The only difference to XPMaps is that split operations can only be applied as long as the segments contain more than one cell.

Thus, from a theoretical point of view, it doesn't matter much whether we work with the reduced map or with the segments directly. This opens up a very general approach to image segmentation: any algorithm that produces labelings consistent with definition 16 can be analyzed in the theoretical framework of XPMaps. Applications include many well-known segmentation algorithms such as split-and-merge, the watershed algorithm, and irregular pyramids [14, 11]. This topic will be further explored in an upcoming paper.

Some Aspects of Software Realization

To make the proposed topological models useful in practice, they must be implemented in software. This leads to another requirement that is not so apparent in the purely mathematical description: if two data objects code identical topological structures, the operations should behave identically. The objects' behavior must not depend on the history of the objects. It should not matter how the current state was reached. This means two things:

- When we list the cells, the order of the listing must agree in both objects. The same applies to the list of components within a face.
- When we access the orbit of a cell, the first dart visited must be the same in both objects.

The second requirement may be somewhat surprising at first. After all, no dart in an orbit is special in any way. But when we actually access an orbit, we must begin with *some* dart. For the sake of reproducibility, this dart must be well defined. Therefore, we mark one dart of each orbit as its *anchor*, and whenever the orbit is accessed the anchor will be listed first. Definition of orders and anchors is done according to the following rules:

- Each cell must carry an attribute that defines a total order of the cells. Usually, this is just an integer ID. Cells are always listed in increasing order.
- The components contained in a face are listed in the order determined by the smallest node that bounds each component's exterior face.
- The anchor of an α orbit (edge) is the dart incident to the smaller node. In case of a self-loop, the dart with smaller left face is chosen.
- The anchor of a σ orbit (node) is the dart belonging to the smallest edge. If this edge is a self-loop, the edge's anchor is chosen.
- The anchor of a φ orbit (face) is the dart belonging to the smallest edge. If this edge is a bridge, the edge's anchor is chosen.

Whenever the structure of the data object is changed by applying an Euler operator, the anchors must be updated accordingly. This ensures that anchors depend only on the current state of the object and not on its history.

A good way to actually implement topological data structures is the method of *generic programming* [15]. This is a new programming paradigm especially tailored to complex algorithmic domains such as computational topology and image analysis. The most important characteristic of generic programming is the use of *iterators* to traverse lists of cells and orbits. This ensures that XPMaps, Khalimsky grid, and topological segmentation share the *same software interface*, so that algorithms can be implemented independently of the data structure selected. A detailed description of this approach can be found in [7] and [8].

Conclusions

In this paper, we have presented a novel, unified approach to the representation of finite topologies in the plane. XPMaps incorporate the full range of functionality from cell complexes, combinatorial maps and plane divisions. The concept of a topological segmentation allows to define labelings of XPMaps that again have the properties of a topological space. This provides a smooth transition from the pixel plane to an abstract topologi-

cal space as it allows for the definition of irregular topologies on top of regular grids. Furthermore, by applying labeling and reduction repeatedly, a wide variety of irregular pyramids can be realized.

Of course, it would be desirable to carry the analysis to 3-dimensional spaces. However, given the wealth of unsolved problems in 2D image analysis and segmentation, its appears equally worthwhile to utilize the power of the new approach here. The general formal definitions presented provide a firm basis for studying the relationships between topology and the imaging process. By using XPMaps as a common representation for segmentation results, we will be able to develop more robust segmentation algorithms and achieve a unification of approaches in general.

References

[1] E. Ahronovitz, J.P. Aubert, C. Fiorio: *"The Star Topology: a Topology for Image Analsysis"*, Proc. 5th Intl. Conf. Discrete Geometry for Computer Imagery, DGCI'95, 1995

[2] Y. Bertrand, C. Fiorio, Y. Pennaneach: *"Border Map: a Topological Representation for nD Image Analysis"*, G. Bertrand, M. Couprie, L. Perroton (eds.): Proc. 8th Intl. Conf. Discrete Geometry for Computer Imagery, DGCI'99, Springer LNCS 1568, 1999

[3] J.-P. Braquelaire, J.-P. Domenger: *"Representation of Region Segmented Images with Discrete Maps"*, Université Bordeaux, Laboratoire Bordelais de Recherche en Informatique, Technical Report 1127-96, 1996

[4] G. Di Battista, P. Eades, R. Tamassia, I.G. Tollis: *"Graph Drawing"*, Prentice Hall, 1999

[5] J.-F. Dufourd, F. Puitg: *"Functional specification and prototyping with oriented combinatorial maps"*, Computational Geometry 16 (2000) 129-156

[6] E. Khalimsky, R. Kopperman, P. Meyer: *"Computer Graphics and Connected Topologies on Finite Ordered Sets"*, J. Topology and its Applications, vol. 36, pp. 1-27, 1990

[7] U. Köthe: *"Generische Programmierung für die Bildverarbeitung"*, PhD thesis, Computer Science Department, University of Hamburg, 2000 (in German)

[8] U. Köthe: *"Generic Programming Techniques that Make Planar Cell Complexes Easy to Use"*, G.Bertrand, A.Imiya, R.Klette (eds.): Proc. of Dagstuhl WS on Digital and Image Geometry 2000, Springer LNCS, to appear

[9] V. Kovalevsky: *"Finite Topology as Applied to Image Analysis"*, Computer Vision, Graphics, and Image Processing, 46(2), pp. 141-161, 1989

[10] V. Kovalevsky: *"Computergestützte Untersuchung topologischer Eigenschaften mehrdimensionaler Räume"*, Preprints CS-03-00, Computer Science Department, University of Rostock, 2000 (in German)

[11] W. Kropatsch: *"Building Irregular Pyramids by Dual Graph Contraction"*, IEE Proceedings – Vision, Image and Signal Processing, 142(6), 366-374, 1995

[12] P. Lienhardt: *"Topological models for boundary representation: a comparison with n-dimensional generalized maps"*, computer aided design, 23(1), 59-82, 1991

[13] M. Mäntylä: *"An Introduction to Solid Modeling"*, Computer Science Press, 1988

[14] A. Montanvert, P. Meer, A. Rosenfeld: *"Hierarchical Image Analysis Using Irregular Tessellations"*, IEEE Trans. Pattern Anal. and Machine Intelligence, 13(4), 307-316, 1991 [15] D. Musser , A. Stepanov: *"Algorithm-Oriented Generic Libraries"*, Software – Practice and Experience, 24(7), 623-642, 1994

[16] W.T. Tutte: *"Graph Theory"*, Cambridge University Press, 1984

Appendix

Proof of Theorem 1

We show that the bounding relation induced by the open stars of a plane division fulfills the requirements of definition 2:

1. It is transitive. We show that, whenever 0-cell c_0 (associated with vertex v) bounds 1-cell c_1 (associated with arc a), and a bounds 2-cell c_2 (associated with region r), then c_0 also bounds c_2 (other cases of transitivity cannot occur in two dimensions). If c_0 bounds c_1, every neighborhood of v contains some point p of a. According to the properties of Euclidean topology, the neighborhood of v also contains a neighborhood of p, which in turn contains a point of r. Therefore, every neighborhood of v contains a point of r. Thus, r is in the open star of v and c_0 bounds c_2 as required.

2. Cells bound only cells of higher dimension: Regions are open sets, thus their open star cannot contain any other cells. Vertices are points, therefore any vertices has a neighborhood that doesn't contain any other (separation axiom of Euclidean topology). Thus vertices cannot bound other vertices. To prove the properties of arcs, take an arc a and draw a new arc a_1 between a's end points such that a_1 is entirely within one of a's incident regions. This is always possible since regions are connected so that any two points on a region's boundary can be connected by an interior arc. Draw another arc a_2 on the other side of a. Then the union of a_1, a_2 and their end points is a Jordan curve. Its interior is an open set which contains the arc a but no other arc and no vertex. Thus, the open star of an arc can only contain regions.

Proof of Theorem 2

We prove that the φ orbits obtained when transforming a plane division into a combinatorial map indeed correspond to the regions of the plane division. Assign a left and a right region to each dart: when a vertex is circled in mathematically positive direction, denote the region found immediately before the dart its *right* region, and the region found immediately after it its *left* one (these need not be distinct). Now observe that the two darts of the same arc have their left and right regions reversed. Observe also that the right region of a given dart is the left region of the dart preceding it in the σ orbit. Consider the left region of some dart d_1. If we apply α to this dart, we get to its twin d_2, who has the given region as right region. By applying σ^{-1} to the twin, we get to its predecessor d_3 whose left region equals the right region of d_2 which equals the left region of d_1. By induction, we see that all darts of a given φ orbit have the same left region.

It remains to be shown that exactly one φ orbit contains all darts with a given left face. We will show this by means of a triangulation of the plane division: insert auxiliary arcs into the regions until all resulting regions are incident to at most three vertices. Triangulation is always possible since any pair of points on the boundary of an open region can be connected by an open arc lying completely in the region. If the boundary set is connected, all auxiliary arcs split some region into two. Thus, there are as many new regions as auxiliary arcs, and no auxiliary arc is a bridge.

Now transform the triangulated plane division into a combinatorial map. We show indirectly that exactly one φ orbit is generated for each region. Suppose to the contrary that the orbits φ_1, ..., φ_k all belong to a single region r. Since the boundary of all regions is connected, at least two orbits (say φ_1 and φ_2) must meet at a common vertex v. Let v be the start point of dart d_1 from φ_1 and d_2 from φ_2. Let $d_3=\sigma(d_1)$ and $d_4=\sigma(d_2)$ their σ successors and $d_5=\alpha(d_3)$ and $d_6=\alpha(d_2)$ their φ predecessors. We can draw yet another new arc lying entirely in r that starts at vertex v between d_1 and d_3 and returns to v between d_2 and d_4. The union of this new arc and v is a Jordan curve which separates the plane into an interior and an exterior part. Since r's contour consists of at most 3 arcs (triangulation!), either exterior or interior contains only one of those arcs. Therefore, either d_2 and d_5 or d_1 and d_6 must actually be identical, so that φ_1 and φ_2 cannot be distinct – contradiction.

Finally, remove the edges corresponding to the auxiliary arcs of the triangulation by *face merge* operations (this is possible since no auxiliary arc is a bridge). Since each face merge reduces the number of faces by one, this will result in a combinatorial map that contains exactly as many faces (φ orbits) as there were regions in the original plane division.

Algorithm to Transform a Planar Cell Complex into an XPMap

The following algorithm transforms a planar cell complex into an XPMap. If the graph of the cell complex is only 1-connected or disconnected, the result is not uniquely determined: whenever there is a choice for the next dart to be processed in steps 3, 5.1, and 7, different XPMaps may result. However, the algorithms always terminates with no remaining darts if the cell complex is planar. Space limitations do not permit to formally prove this algorithm.

1. For each 0-cell create a node, for each 1-cell create two darts, and for each 2-cell a face. To each dart associate one of the nodes that correspond to the 0-cells bounding the dart's 1-cell as its start node. If the 1-cell is bounded by only one 0-cell, it is a loop, and the same node is associated with both darts. Further associate with each dart the two faces which correspond to the 2-cells bounded by the dart's 1-cell. If a 1-cell bounds only one 2-cell, it is a bridge (or isthmus), and the darts are associated with the same face twice.
2. Each pair of darts corresponding to the same 1-cell forms a cycle in the α involution. (The darts in each pair will be called twins.)
3. Select a dart that is associated with the infinite face of the cell complex.
4. Designate that face as the dart's "left" face. The dart's other face becomes the "right" face. In the dart's twin, the opposite orientation is induced by this designation. Append the selected dart to the σ orbit of its start node.

5. Select all darts who have the same start node as the one just inserted. Recursively append them to the same orbit as follows:
   5.1. Among the darts not yet inserted, select one whose right face equals the left face of the dart appended last. If no such dart exists, there must be a dart that is associated with that face, but its orientation (left and right face) has not yet been fixed. Select that dart and make the current face the right one (as before, this induces the opposite designation in the dart's twin). Append the selected dart to the orbit.
   5.2. Repeat until all darts of the node have been inserted into the orbit.
6. Select any free dart whose left face has already been decided. Insert it into its start node's orbit and go to 5.
7. If no such dart exists, a connected component of the complex' graph has been completed. Start the next component by selecting a dart that is associated with the infinite face or a face already inserted in the map. Update the containment relation accordingly. Go to 4 until no suitable darts remain.

Proof of Theorem 4

We prove that the φ orbits created for an open point of the Khalimsky grid by the proposed algorithm are correct. For a given orientation o, let o_- and o_+ denote the previous and next orientation in counter-clockwise order. If the dart has orientation o, the point corresponding to its left face is located in direction o_+ relative to the dart's point. According to the definition of α and σ, he darts φ successor is found by going one step in direction o and one in direction o_+. This darts left face is then found by a third step in direction $(o_+)_+$. Since $(o_+)_+$ is the opposite direction of o, the sequence o, o_+, $(o_+)_+$ is equivalent to o_+. Therefore, the two darts have the same left face. In a similar way, it can be shown that the φ orbit's length is 4 as required.

Proof of Validity of the Node Split Operator

We prove that "node split" keeps the number of faces (φ orbits) constant. Suppose that $d_a=\varphi^{-1}(d_i)$ is the φ predecessor of d_i before splitting. That is, the φ orbit of d_i contains the sequence (..., d_a, d_i,...), and $\alpha(d_a)=\sigma(d_i)=d_{i+1}$ by definition of φ. We consider two cases:

(1) Suppose $i=j$. Then we have the new orbits σ_1'=($d_{j+1}=d_{i+1}$,...,d_i, d) and σ_1''=(d'). That is, d has been inserted between d_i and d_{i+1}. Lets see what happens to the φ orbit of d_i: the φ successor of d_a in the modified map is $\varphi(d_a)=\sigma^{-1}(\alpha(d_a))=\sigma^{-1}(d_{i+1})=d$. Following the orbit further, we get $\varphi(d)=\sigma^{-1}(\alpha(d))=\sigma^{-1}(d')=d'$ (since d' is in a σ orbit of size 1) and then $\varphi(d')=\sigma^{-1}(\alpha(d'))=\sigma^{-1}(d)=d_i$. Thus, the modified φ orbit contains the sequence (..., d_a, d, d', d_i,...). Both new darts have been inserted into an existing φ orbit, and the number of φ orbits has not changed, as required.

(2) Suppose now that $i\neq j$. Then we have the new orbits σ_1'=(d_{j+1},...,d_i, d) and σ_1''=(d_{i+1},...,d_j, d'). Follow the modified φ orbit of d_a: $\varphi(d_a)=\sigma^{-1}(\alpha(d_a))=\sigma^{-1}(d_{i+1})=d'$. Going further gives $\varphi(d')=\sigma^{-1}(\alpha(d'))=\sigma^{-1}(d')=d_i$. Thus, the orbit of d_i now contains the sequence (..., d_a, d, d_i,...). An analogous argument can be put forward for the φ orbit of d_j (which might be another part of the d_i orbit). Again, the operator has only inserted two new darts into existing φ orbits, and the number of faces has not changed.

Proof of Validity of the Remove Bridge Operator

To prove the effect of the operator on φ orbits, we first observe that the two cases in the definition differ only in how the containment relation is updated. The orbits are modified in the same way in both cases. Suppose the bridge's darts are d and d'. By definition of a bridge, both darts are part of the same φ orbit. If both end nodes of the bridge have degree >1, this φ orbit contains the following sequence of darts: (..., $d_1=\varphi^{-1}(d)$, d, $d_2=\varphi(d)$,..., $d_3=\varphi^{-1}(d')$, d', $d_4=\varphi(d')$, ...). The two σ orbits contain the sequences (..., $\alpha(d_1)$, d, d_4,...) and (..., $\alpha(d_3)$, d', d_2,...) respectively. After removal of d, the σ orbit of d_1 becomes (..., $\alpha(d_1)$, d_4,...). Thus, the φ orbit of d_1 now contains (..., d_1, $\varphi(d_1)=\sigma^{-1}(\alpha(d_1))=d_4$,...). d_2 and d_3 are no longer part of this orbit. By applying the same argument to d_2, we see that exactly two φ orbits have been created.

If, on the other hand, an end nodes has degree 1, it is replaced by a vertex map which also contains exactly one (empty) φ orbit. So, in any case the number of φ orbits is increased by one. But the new orbit is marked as the exterior face of a new component, so that the number of faces remains constant as required.

Proof of Theorem 5

For the proof we first need some lemmas:

Lemma 6: Consider a connected open set in a XPMap. If the set contains bridges and/or isolated nodes they can be removed (by means of remove bridge/isolated node) so that the set remaining set is still connected and open.

Proof: We first show that the set remains connected. Recall that a set is connected if every pair of cells in the set can be connected by a path that is completely within the set. Here, a path is a sequence of cells where successive cells are always incident (that is, one of them bounds the other). Consider an isolated node. It always bounds only its surrounding face, and by definition of an open set, this face also belongs to the set. Thus, any path containing the node can be rewritten without it (i.e. (..., *f*, *n*, *f*, ...) is replaced by (..., *f*,...). Removal of the node does not change connectedness. Now consider a bridge. Again it bounds only one face which also belongs to the set. By transitivity of the bounding relation, any cell that bounds the bridge also bounds that face. Therefore, the bridge can be replaced by the face wherever it occurs in a path, and its removal does not change connectedness.

Now we prove that the set is still open after removal. We must show that the open stars of all other cells in the set remain in the set. This is trivial in case of an isolated node, since the only open star modified is the removed node's itself. A bridge is part of the open stars of its end nodes. If it is deleted, it is removed from these open stars. All other cells bound by these nodes remain untouched. The lemma follows. ■

Lemma 7: If a connected open set in an XPMap contains at least two faces, it also contains an edge that is not a bridge. Removal of such an edge by a face merge again results in a connected open set.

Proof: By the previous lemma, we may assume that the set does not contain bridges or isolated nodes. Since the set is connected and faces cannot be incident, there must be at least one additional cell in the set. If this is an edge, the first part of the lemma follows immediately. If it is a node, it must bound at least one edge (since it is not isolated) which, by definition of an open set, is also in the set. Again, the first part of the lemma follows.

To prove the second part, suppose the edge e is incident to the faces f_1 and f_2 and the set $\{f_1, e, f_2\}$ gets replaced by f'. By definition of face merge, any cell that was incident to one of f_1, e, f_2 becomes incident to f'. Therefore, any path containing one of the deleted cells can be rewritten in terms of f'. Likewise, any open star that contained one of them now contains f'. Since f' belongs to the set and is itself open, the lemma follows. ■

Lemma 8: Any connected open set in a combinatorial map can be reduced to a single face by a sequence of Euler operators, without changing the structure of the set's complement. Likewise, any simply connected closed set can be reduced to a single node.

Proof: If the open set contains several faces, remove all non-bridge edges until only one face remains in the set. By the previous lemma, the resulting set is still connected and open. The remaining edges in the set are all bridges, which can be removed by lemma 6 (this may temporarily turn the map into an XPMap). At the end of bridge removal, all cells in the sets besides the face must be isolated nodes. According to lemma 6 they can also be removed. The result is a connected open set that consists of a single face, as claimed.

The proof for closed sets follows from the observation that closed sets correspond to open sets in the dual map. We apply the first part of the lemma to the dual set and obtain a single face. By taking the dual of the dual, the face is turned into a single node and the claim follows. ■

In practice, one can reduce a closed set directly, without constructing the dual map. First, remove all non-bridge edges from the set by means of "face merge". The remaining set consists of only nodes and bridges which form a tree. Since a tree has one more nodes than edges and no loops, one can remove all bridges by means of edge contractions. The result is a single node.

Now we can finally prove theorem 5. If none of the faces in the set to be reduced (either open or closed) contains another component of the map, the set can be reduced by the previous lemma. Otherwise, observe that an entire component of an XPMap is always an open set. Thus we can split the task into two steps: first reduce all contained components to single faces by recursively applying the present procedure, and then reduce the remaining set (which no longer contains further components) by lemma 8. ■