An extension of component-trees to partial orders

Nicolas Passat, Benoît Naegel

AN EXTENSION OF COMPONENT-TREES TO PARTIAL ORDERS

Nicolas Passat

LSIIT, UMR CNRS 7005
Strasbourg University
France

Benoît Naegel

LORIA, UMR CNRS 7503
Nancy 1 University
France

ABSTRACT

Component-trees provide efficient ways to define filtering-based procedures on grey-level images. We propose an extension of the notion of component-trees to the case of "non grey-level" images (*i.e.* images taking their values in partially-ordered sets) including in particular - but not exclusively - colour images. Experiments performed on such images emphasise the interest of the approach.

Index Terms— Mathematical morphology, component-trees, filtering, partial orders, colour imaging

1. INTRODUCTION

The *component-tree* models some structural characteristics of an image by considering its "binary slices" obtained by threshold operations at successive levels. It has been defined in the theoretical framework of mathematical morphology and involved, in particular, in the development of morphological operators [1, 2].

Thanks to efforts devoted to the efficient computation of component-trees [2, 3], or their use in complex knowledge handling procedures, they have been considered for the design of various kinds of grey-level image processing methods, including image and video segmentation [2, 4], image registration [5] or image compression [2].

We propose to explore new ways for the use of component-trees by not only considering images taking their values in completely-ordered sets (*i.e.* "grey-level" images), but more generally in *partially-ordered* ones, such as colour images, for instance. A data structure and some strategies for filtering such images are described. They are, in particular, compliant with the classical approaches proposed in the grey-level case. Experimental results enable to illustrate the soundness and usefulness of the approach.

The article is organised as follows. In Sec. 2, theoretical background notions are provided. In Sec. 3, the definition of the component-tree, and classical ways for using it in filtering procedures are recalled. In Sec. 4, the proposed extension of component-trees to images taking their values in partially-ordered sets is described. Experimental results and perspectives will be found in Sec. 5.

2. THEORETICAL BACKGROUND

Let $n \in \mathbb{N}^*$. Let $E \subset \mathbb{Z}^n$ (generally, $E = \prod_{i=1}^{n}[0, d_i - 1]$ with $(d_i)_{i=1}^{n} \in (\mathbb{N}^*)^n$). Let V be a finite set. Let $\leq$ be an order relation on V. We suppose that there exists an infimum, noted $\bot$, for $(V, \leq)$. Let $I : E \to V$ (we also note $I \in V^E$), I is a *(discrete) image*, taking its values in V. If $(V, \leq)$ is a completely-ordered (resp. partially-ordered) set, then we say that I is a *grey-level image* (resp. a *label image*).

Let $x, y \in E$. We say that x and y are adjacent if $d(x, y) \leq \epsilon$ for a well-chosen distance $d(.,.)$ on E and a nonnegative value ϵ (with $\epsilon = 1$ and $d(.,.) = \|. - .\|_1$ (resp. $d(.,.) = \|. - .\|_\infty$), we retrieve the classical $(2n)$-adjacency (resp. $(3^n - 1)$-adjacency) associated to $\mathbb{Z}^n$).

Let $X \subseteq E$. Let $x, y \in X$. We say that x and y are connected (in X), and we note $x \sim y$, if there exists a sequence $(x_k)_{k=1}^{l}$ ($l \geq 1$) of elements of X such that $x_1 = x$, $x_l = y$ and x_k, x_{k+1} are adjacent for all $k \in [1, l-1]$. Note that $\sim$ is an equivalence relation on X. The connected components of X are the equivalence classes of X w.r.t. $\sim$, *i.e.* the elements of the quotient set $X/\sim$ (noted $\mathcal{C}[X]$ in the sequel).

Let $v \in V$. Let $X_v : V^E \to \mathcal{P}(E)$ be the threshold operator defined by $X_v(I) = \{x \in E \mid v \leq I(x)\}$ for all $I \in V^E$.

Let $v \in V$ and $X \subseteq E$. We define the cylinder function $C_{X,v} : E \to V$ by $C_{X,v}(x) = v$ if $x \in X$ and $\bot$ otherwise. Based on this definition, a discrete image $I \in V^E$ can be expressed as $I = \bigvee_{v \in V} C_{X_v(I),v} = \bigvee_{v \in V} \bigvee_{X \in \mathcal{C}[X_v(I)]} C_{X,v}$, where $\bigvee$ is the pointwise supremum of a set of functions, *i.e.* $I(x) = \sup_{v \in V}\{C_{X_v(I),v}(x)\}$ $(= \max_{v \in V}\{C_{X_v(I),v}(x)\}$ as V is finite).

Let $\mathcal{K} = \bigcup_{v \in V}(\mathcal{C}[X_v(I)] \times \{v\})$. Let $\subseteq_{\mathcal{K}}$ be the relation defined by $(X, v_X) \subseteq_{\mathcal{K}} (Y, v_Y)$ if $X \subset Y$ or ($X = Y$ and $v_Y \leq v_X$). The inclusion relation $\subseteq_{\mathcal{K}}$ is a partial order on $\mathcal{K}$. By abuse of notation, in the sequel, we will sometimes note $X \in \mathcal{K}$ for $(X, v_X) \in \mathcal{K}$, and $\subseteq$ for $\subseteq_{\mathcal{K}}$.

3. COMPONENT-TREES

In this section, we assume that $(V, \leq)$ is a *completely-ordered set*. In particular, we identify $(V, \leq)$ and $([0, |V| - 1], \leq)$.

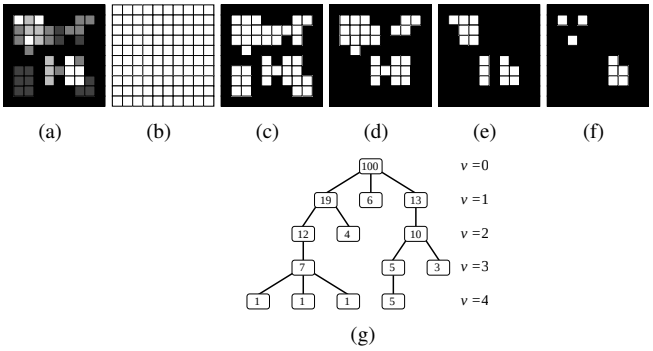

Fig. 1. (a) A grey-level image $I : [0,9]^2 \rightarrow [0,4]$. (b-f) Threshold sets $X_v(I)$ for v from 0 (b) to 4 (f). (g) The component-tree of I; its successive levels correspond to increasing threshold values v. (h) The same tree, enriched by an attribute (the size of the connected component of each node).

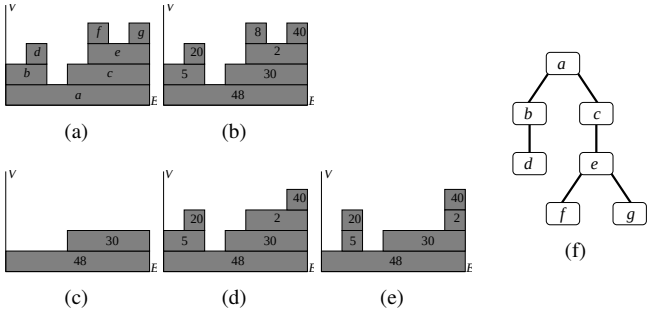

Fig. 2. (a) A 1-D grey-level image I and (f) its component-tree. (b) (Fictive) non-increasing attributes associated to the nodes of I. (c-e) Filtering results I_ρ obtained by processing the component-tree w.r.t. a predicate ρ true if and only if the attribute of the node is higher than 15: (c) *Min*, (d) *Max* and (e) *Direct* policies.

Let $v_1 \leq v_2 \in V$. Let $B_1, B_2 \subseteq E$ be the binary images defined by $B_k = X_{v_k}(I)$ for $k \in \{1,2\}$. Let $C_2 \in \mathcal{C}[B_2]$ be a connected component of B_2. Then, there exists a (unique) connected component $C_1 \in \mathcal{C}[B_1]$ of B_1 such that $C_2 \subseteq C_1$ (see Fig. 1 (b-f) for an illustration of this assertion). Note that, in particular, we necessarily have $B_2 \subseteq B_1$.

Based on these considerations, it easily derives that the Hasse diagram of the partially-ordered set $(\mathcal{K}, \subseteq)$ is a tree (*i.e.* a connected acyclic graph), the "root" of which is the supremum $\sup(\mathcal{K}, \subseteq) = X_\perp(I) = E$. This tree is called the *component-tree of* I (see Fig. 1(g)).

Using the above definitions and notations, the component-tree - as a data structure - can be described as follows.

Definition Let $I \in V^E$ be a grey-level image. The *component-tree* of I is the rooted tree $(\mathcal{K}, L, R)$ defined by

$$R = E = \sup(\mathcal{K}, \subseteq), \tag{1}$$

$$L = \{(X,Y) \in \mathcal{C}[X_v(I)] \times \mathcal{C}[X_{v+1}(I)] \mid Y \subseteq X\}. \tag{2}$$

The elements $\mathcal{K}$, R and L are the set of the *nodes*, the *root* and the set of the *edges* of the tree, respectively (in the sequel, we will use the usual terminology associated to trees). In Fig. 1(g), $\mathcal{K}$ is the set of all white rectangles (R is the one located at the highest level) and L is visualised by the set of all black edges (linking the elements of each pair).

Component-trees enable the storage - at each node - of elements of information, also called *attributes*, related to the binary connected component associated to the node. For instance, in Fig. 1(h), the size (*i.e.* the cardinal) of the connected components has been added at their corresponding nodes[1].

Component-trees, equipped with attributes related to their image, are generally considered for filtering operations consisting in "pruning" the branches of the tree, based on a chosen predicate ρ on $\mathcal{K}$, indicating if the attribute of a node satisfies a given criterion. In case of non-increasing criteria, various pruning policies can be considered (see [6]):
(i) *Min*: A node $N \in \mathcal{K}$ is removed if $\neg\rho(N)$ or the parent node $P \in \mathcal{K}$ of N has been removed;
(ii) *Max*: A node $N \in \mathcal{K}$ is removed if $\neg\rho(N)$ and all its children $C \in \mathcal{K}$ have been removed;
(iii) *Direct*: A node $N \in \mathcal{K}$ is removed if $\neg\rho(N)$.

Once the set $\mathcal{K}$ has been reduced to a set $\mathcal{K}_\rho \subseteq \mathcal{K}$ according to one of these three policies, a reconstructed image I_ρ of I can be defined as $I_\rho = \bigvee_{(X,v_X) \in \mathcal{K}_\rho} C_{X,v_X}$. Filtering and reconstruction results of these strategies are illustrated on a 1-D example in Fig. 2.

4. EXTENSION TO PARTIAL ORDERS

In this section, we now assume that $(V, \leq)$ is a *partially-ordered set* and that $(V, \leq)$ has an infimum, noted $\perp$ (otherwise we enrich $(V, \leq)$ with such an infimum).

Let $I \in V^E$ be a label image. Let $v \in V$. Let $B \subseteq E$ be the binary image defined by $B = X_v(I)$. Let $C \in \mathcal{C}[B]$ be a connected component of B. By opposition to the case of completely-ordered sets (Sec. 3), there may exist *several* values $v_i \leq v$ ($i \in \mathbb{N}$), such that $C \subseteq C_i \in \mathcal{C}[X_{v_i}(I)]$, and v_i is maximal w.r.t. $\leq$ for these properties.

As in Sec. 3, the relation $\subseteq$ is still a partial order on $\mathcal{K}$, but the Hasse diagram of the partially-ordered set $(\mathcal{K}, \subseteq)$ *is not necessarily a tree*. This derives from the fact that - using tree terminology - a node of the diagram can be the child of *several* parent nodes (and not only one). Note that such a diagram (called *component-graph* in the sequel) has however a supremal element (which can be assimilated to its "root"), namely $X_\perp(I) = E$, while it can possibly have several minimal elements.

[1] Here, the attribute is a single numerical value. It is however possible to consider any kind of (quantitative/qualitative and scalar/vectorial) attributes, provided they can be conveniently formalised.

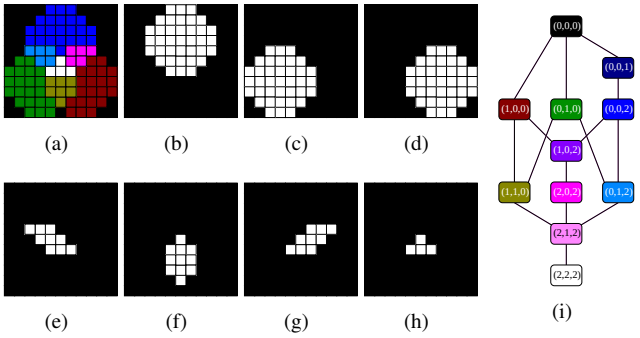

Fig. 3. (a) A label image $I : [0, 10]^2 \to V$ (where $V \subset [0, 2]^3$ is composed of 11 distinct values depicted in (i)) visualised as an RGB image. (b-h) Some threshold images $X_v(I)$ (from (b) to (h), $v = (0, 0, 2)$, $(0, 1, 0)$, $(1, 0, 0)$, $(0, 1, 2)$, $(1, 1, 0)$, $(2, 0, 2)$, $(2, 2, 2)$. (i) The component graph of I.

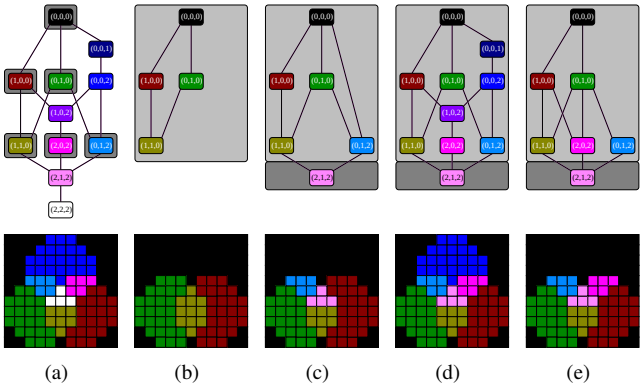

Fig. 4. (a) I and its component-graph, with nodes verifying a given predicate ρ (grey-background). (b-e) Light-grey background: graph $(\mathcal{K}_\rho, L_\rho)$ obtained with the *Min*$_1$ (b), *Min*$_2$ (c), *Max* (d), and *Direct* (e) policies; light/dark-grey background: corrected graph $(\mathcal{K}^c_\rho, L^c_\rho)$.

4.1. Component-graphs

Let $I \in V^E$ be a label image. The *component-graph* of I is the "rooted" oriented graph $(\mathcal{K}, L, R)$ defined such that $(\mathcal{K}, L)$ is the Hasse diagram of the partially-ordered set $(\mathcal{K}, \subseteq)$, and $R = \sup(\mathcal{K}, \subseteq) = X_\perp(I) = E$. Keeping the same terminology as previously, $\mathcal{K}$, R and L are the set of the *nodes*, the *root* and the set of the *edges* of the graph, respectively.

Fig. 3 illustrates a label image I (a), some of its threshold images (b-h), and its component graph (i): $\mathcal{K}$ is the set of all rectangles (R is the black one) and L is visualised by the set of all black edges (linking the elements of each pair).

Similarly to component-trees, attributes can be stored at each node of a component-graph. The preservation/removal of nodes of the graph w.r.t. such attributes can then also lead to filtering strategies. However, the less restrictive properties of component-graphs (by opposition to those of component-trees) do not enable to straightforwardly apply the filtering strategies designed for component-trees, described in Sec. 3. To tackle this problem, we propose hereafter a two step filtering procedure.

4.2. Filtering strategy

Let $I \in V^E$ be a label image and $(\mathcal{K}, L, R)$ be its component-graph (the image I of Fig. 3(a) will be considered for illustration). Let ρ be a predicate on $\mathcal{K}$. The following strategy is devoted to the filtering of I by "pruning" its component-graph w.r.t. ρ.

4.2.1. Step 1: Node selection

Let $\mathcal{K}' = \mathcal{K} \setminus \{R\}$. In case of *non-increasing* criteria (notice that with *increasing* ones, all policies are equivalent), the following policies can be considered to select a subset $\mathcal{K}_\rho \subseteq \mathcal{K}$ of nodes to be preserved in the component-graph:
(i) *Min*$_1$: A node $N \in \mathcal{K}'$ is removed if $\neg\rho(N)$ or at least one of the parent node $P \in \mathcal{K}$ of N has been removed;
(ii) *Min*$_2$: A node $N \in \mathcal{K}'$ is removed if $\neg\rho(N)$ or all the parent nodes $P \in \mathcal{K}$ of N have been removed;
(iii) *Max*: A node $N \in \mathcal{K}$ is removed if $\neg\rho(N)$ and all its children $C \in \mathcal{K}'$ have been removed;
(iv) *Direct*: A node $N \in \mathcal{K}$ is removed if $\neg\rho(N)$.
Note that with the *Min*$_1$ and *Min*$_2$ (resp. *Max*) policy(ies), $\mathcal{K}_\rho$ has to be computed in a top-down (resp. bottom-up) fashion. Moreover, with the first three policies, we should assume that $R \in \mathcal{K}_\rho$ (otherwise, we would obtain a - useless - trivial set $K_\rho = \emptyset$). More formally, with the *Min*$_1$ (3), *Min*$_2$ (4), *Max* (5), and *Direct* (6) policies, we have (recursively)

$$\mathcal{K}_\rho = \{R\} \cup \{N \in \mathcal{K}' \mid \rho(N) \wedge \forall (P, N) \in L, P \in \mathcal{K}_\rho\}, \quad (3)$$
$$\mathcal{K}_\rho = \{R\} \cup \{N \in \mathcal{K}' \mid \rho(N) \wedge \exists (P, N) \in L, P \in \mathcal{K}_\rho\}, \quad (4)$$
$$\mathcal{K}_\rho = \{R\} \cup \{N \in \mathcal{K}' \mid \rho(N) \vee \exists (N, C) \in L, C \in \mathcal{K}_\rho\}, \quad (5)$$
$$\mathcal{K}_\rho = \{N \in \mathcal{K} \mid \rho(N)\}. \quad (6)$$

Once $\mathcal{K}_\rho$ has been computed, a new set of edges L_ρ can straightforwardly be defined by building the Hasse diagram of $(\mathcal{K}_\rho, \subseteq)$. Examples of graphs $(\mathcal{K}_\rho, L_\rho)$ obtained with these different policies are illustrated in Fig. 4.

4.2.2. Step 2: Coherence recovery

By considering the *Min*$_1$ and *Min*$_2$ (resp. *Max*, *Direct*) policy(ies) proposed above, in the case of a completely-ordered set, we obviously retrieve the "classical" *Min* (resp. *Max*, *Direct*) policy defined in Sec. 3. The proposed approach is then compliant with the case of component-trees.

However, in the considered case of a partially-ordered set $(V, \leq)$, it may sometimes be impossible to reconstruct an image $I_\rho = \bigvee_{(X, v_X) \in \mathcal{K}_\rho} C_{X, v_X}$ defined by $I_\rho(x) =$

$\max_{(X,v_X)\in\mathcal{K}_\rho}\{C_{X,v_X}(x)\}$ for all $x \in E$, since such a maximum may be undefined, due to the partiality of $(V, \leq)$. As an example, in Fig. 4, the graph of (b) directly leads to a "correct" image I_ρ, while it is not the case for those of (c-e) since the lowest nodes of the graph correspond to intersecting connected components associated to non-comparable values.

In order to deal with this issue, a *coherence recovery* procedure has to be proposed to define unambiguously $\max_{(X,v_X)\in\mathcal{K}_\rho}\{C_{X,v_X}\}$ for all $x \in E$. This procedure consists in adding to the current graph $(\mathcal{K}_\rho, L_\rho)$ a set of nodes (actually removed during the previous step) and their associated edges, such that the resulting corrected graph $(\mathcal{K}^c_\rho, L^c_\rho)$ enables the generation of a *well-defined* image. Practically, this can be done by computing iteratively - and until stability - $\mathcal{K}^c_\rho$ (initialised to $\mathcal{K}_\rho$) as follows. Choose a point $x \in E$ and two nodes $N_i = (X_i, v_{X_i}) \in \mathcal{K}^c_\rho$ $(i \in \{1, 2\})$ (with $N_1 \not\subseteq N_2$, $N_2 \not\subseteq N_1$) such that $x \in X_1 \cap X_2$ and $\forall N = (X, v_X) \in \mathcal{K}^c_\rho \setminus \{N_i\}, x \in X \Rightarrow (N \not\subseteq N_i)$. Then, there exists $N = (X, v_X) \in \mathcal{K} \setminus \mathcal{K}^c_\rho$ such that $x \in X \subseteq X_1 \cap X_2$ and $v_{X_1}, v_{X_2} \leq v_X$. Choose such a node N and set $\mathcal{K}^c_\rho = \mathcal{K}^c_\rho \cup \{N\}$. Once $\mathcal{K}^c_\rho$ has been computed, L^c_ρ is obtained by computing the Hasse diagram of $(\mathcal{K}^c_\rho, \subseteq)$.

Note that the element N may be non-unique at a given iteration, leading to a possibly non-deterministic process, and thus various results (in the case where $(V, \leq)$ is a lattice, some deterministic strategies can however be developed ; this will be considered in further works). Examples of corrected graphs and their associated image are available in Fig. 4(c-e).

5. EXPERIMENTAL RESULTS AND PERSPECTIVES

We have introduced an extension of the notion of component-trees (namely the component-graphs) to the case of partially-ordered sets, enabling in particular to deal with colour images.

To illustrate the soundness of this new concept, a denoising example is proposed on a colour image (see Fig. 5). Considering a predicate ρ based on the size of the connected components of the nodes, a two-step component-graph filtering procedure (with the $\leq$ and $\geq$ partial orders on the RGB space, successively) is applied with a *Direct* policy on an image (a) corrupted with 10% (b) and 15% (c) random noise. It can be observed that the result images (e,f) present visual qualities quite similar to the original one (a,d).

This simple example (presented here due to space limitations) is - of course - not the only nor the main application field of component-graph approaches: more sophisticated ones (applied on both real colour images and "synthetic" ones [7] obtained from hierarchical classification processes, for instance) will be presented in further works. From a methodological point of view, we will also develop this preliminary study by extending classical concepts of the component-trees such that the *Subtractive* or *Viterbi* policies, and by developing alternative coherence recovery strategies.

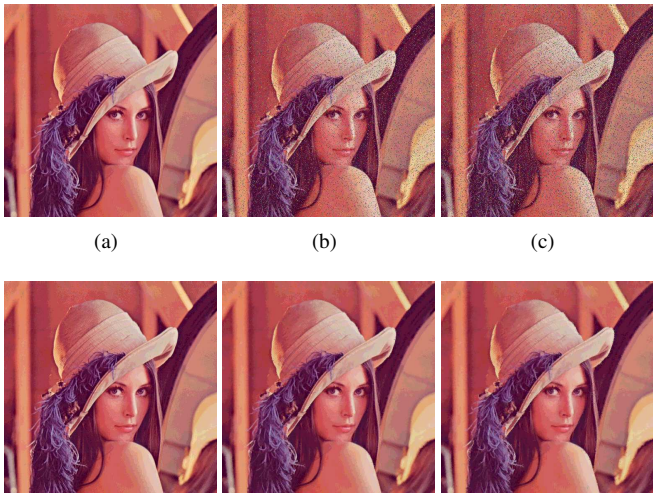

Fig. 5. (a,d) Original image (Lenna, ©Playboy). (b,c) Noisy image (10% and 15% random noise corruption, respectively). (e-f) Filtered images obtained from (b,c) by removing the connected component of size smaller than 10 pixels.

6. REFERENCES

[1] E.J. Breen and R. Jones, "Attribute openings, thinnings, and granulometries," *Computer Vision and Image Understanding*, vol. 64, no. 3, pp. 377–389, 1996.

[2] P. Salembier, A. Oliveras, and L. Garrido, "Anti-extensive connected operators for image and sequence processing," *IEEE Transactions on Image Processing*, vol. 7, no. 4, pp. 555–570, 1998.

[3] L. Najman and M. Couprie, "Building the component tree in quasi-linear time," *IEEE Transactions on Image Processing*, vol. 15, no. 11, pp. 3531–3539, 2006.

[4] R. Jones, "Connected filtering and segmentation using component trees," *Computer Vision and Image Understanding*, vol. 75, no. 3, pp. 215–228, 1999.

[5] J. Mattes, M. Richard, and J. Demongeot, "Tree representation for image matching and object recognition," in *DGCI*. 1999, vol. 1568 of *LNCS*, pp. 298–312, Springer.

[6] E.R. Urbach, J.B.T.M. Roerdink, and M.H.F. Wilkinson, "Connected shape-size pattern spectra for rotation and scale-invariant classification of gray-scale images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 2, pp. 272–285, 2007.

[7] C. Ronse and V. Agnus, "Morphology on label images: flat-type operators and connections," *Journal of Mathematical Imaging and Vision*, vol. 22, no. 2–3, pp. 283–307, 2005.