Towards connected filtering based on component-graphs

Benoît Naegel, Nicolas Passat

Towards connected filtering based on component-graphs*

Benoît Naegel[1] and Nicolas Passat[2]

[1] ICube, UMR CNRS, Université de Strasbourg, France
[2] CReSTIC, EA 3804, Université de Reims, France

Abstract. In recent works, a new notion of component-graph has been introduced to extend the data structure of component-tree –and the induced antiextensive filtering methodologies– from grey-level images to multivalued ones. In this article, we briefly recall the main structural key-points of component-graphs, and we present the first algorithmic results that open the way to the actual development of component-graph-based antiextensive filtering procedures.

Keywords: Component-graph, component-tree, multivalued images, partially ordered sets, connected operators, antiextensive filtering.

1 Introduction

Among the connected filtering approaches, that aim to process images without altering their contours, the component-tree [17] has received a specific attention. The component-tree is a data structure that models the characteristics of grey-level images by considering their successive binary level-sets. It is well-suited for processing grey-level images, based on hypotheses related to the topology (connectedness) and the specific intensity (local extrema) of structures of interest. In particular, it has been involved in several approaches (see, *e.g.*, [9, 19]), especially for filtering and segmentation.

The success of the component-tree in the field of grey-level image processing, together with the increasing need for applications involving multivalued images, motivate its extension to the case of such images, which do not take their values in totally ordered sets, but in any (possibly partially) ordered ones.

After a preliminary study of the relations between component-trees and multivalued images [11], an extension of the component-tree to a more general notion of *component-graph* has been initiated in [13]. A study of the structural properties of these component-graphs has been proposed in [14], and we now consider the algorithmic key-points that will lead to the effective development of antiextensive filtering procedures.

This article is organised as follows. Sec. 2 briefly recalls some previous works on multivalued image handling in mathematical morphology. Sec. 3 describes the way to extend the notion of component-tree into a compliant notion of component-graph. The next sections explain how to build (Sec. 4), prune (Sec. 5) and recover a filtered multivalued image (Sec. 6) from a component-graph. The article is concluded by an illustrative example and perspective works, in Secs. 7 and 8, respectively.

* The research leading to these results has received funding from the French *Agence Nationale de la Recherche* (Grant Agreement ANR-2010-BLAN-0205 03).

2 Previous works

2.1 Mathematical morphology and multivalued images

The extension of mathematical morphology –initially defined on binary, and then on grey-level images [8]– to multivalued (*e.g.*, colour, label, multi- and hyperspectral, etc.) images is an important task, motivated by potential applications in multiple areas. Several contributions have been devoted to this specific purpose. A whole state of the art is beyond the scope of this article, and the reader is referred to [3] for a recent survey.

By opposition to the grey-level case, the spaces in which such multivalued images take their values are not canonically equipped with total orders, but with partial ones. Several strategies have been considered to deal with this issue. Except in few works (see, *e.g.*, [15]), the proposed attempts generally consist of decomposing these value spaces into several totally ordered ones (marginal processing), or to define *ad hoc* total order relations on them (vectorial processing), with several variants (see, *e.g.*, [7, 2, 20]).

These approaches embed multivalued images into simpler frameworks, which authorise to process them similarly to grey-level ones, reducing in particular the complexity induced by partial orders. However, they also potentially bias the information intrinsically carried by these –more complex but richer– partially ordered value spaces.

2.2 The case of tree-based approaches

In the specific field of approaches based on tree structures (or more generally on partition hierarchies) the difficulties raised by multivalued images vary according to the proximity degree that exists between the data structure and the value space.

In the case of trees, or partition hierarchies, whose construction is not directly induced by the value space, and more precisely by its associated order (*e.g.*, partition trees [16], hierarchical watershed [6]), the use of intermediate functions (*e.g.*, a gradient for watershed, or more complex metrics for partition trees [18]) enables to "hide" the complexity of the space, but necessarily induces a bias in the obtained data structure.

In the case of trees whose construction directly derives from the value space –and its order–, passing from total orders to partial ones leads to structural and algorithmic open issues. The main problems are caused by the fact that such data structures inherit from the structural complexity of the considered orders, and actually decuple this complexity via their hierarchical structure. Among such kinds of trees, we find the component tree [17] and its autodual version, the tree of shapes [10]. Researches about the extension to color images of the tree of shapes are currently developed by other authors [5].

In this article, we consider the component tree, and we investigate its extension to a more general hierarchical data structure that is no longer a tree, namely, the *component-graph*. In particular, we focus on the algorithmic consequences of this last property.

3 From component-trees to component-graphs

We now recall basic notions related to component-trees (Sec. 3.2). Then we introduce the recently proposed notion of component-graph [13, 14] (Sec. 3.3). Due to space limitations, we present the minimal set of definitions and properties that are required to make this article self-contained. A more complete description may be found in [14].

3.1 Definitions and hypotheses

Let Ω be a nonempty finite set equipped with a given connectivity. In particular, for any $X \subseteq \Omega$, the set of the connected components of X is noted $C[X]$.

Let V be a nonempty finite set equipped with an order relation $\leqslant$. We assume that $(V, \leqslant)$ admits a minimum, noted $\bot$.

Let I be an image defined on Ω and taking its values in V, *i.e.*, a function $I : \Omega \to V$. Without loss of generality, we can assume that $I^{-1}(\{\bot\}) = \{x \in \Omega \mid I(x) = \bot\} \neq \emptyset$.

For any $v \in V$, let $\lambda_v : V^\Omega \to 2^\Omega$ be the thresholding function at value v, defined for any image I, by $\lambda_v(I) = \{x \in \Omega \mid v \leqslant I(x)\}$.

3.2 Component-trees

Here, we assume that $\leqslant$ is a total order. In other words, the image I is a grey-level image.

Let us define Ψ as the set of all the connected components obtained from all the thresholdings of I, that is

$$\Psi = \bigcup_{v \in V} C[\lambda_v(I)] \tag{1}$$

Definition 1 (Component-tree [17]) *The component-tree of I is the Hasse diagram $\mathfrak{T}$ of the partially ordered set $(\Psi, \subseteq)$.*

The component-tree has several virtues. Firstly, it can be built quite efficiently [17, 12, 4]. Secondly, it models the associated image in a lossless way. Indeed, we have

$$I = \bigvee_{v \in V}^{\leq} \bigvee_{X \in C[\lambda_v(I)]}^{\leq} C_{(X,v)} \tag{2}$$

where $\leq$ is the pointwise order relation on V^Ω induced by $\leqslant$, and $C_{(X,v)} : \Omega \to V$ is the cylinder function defined, for any $x \in \Omega$ by $C_{(X,v)}(x) = v$ if $x \in X$, and $\bot$ otherwise. Thirdly, any subset $\widehat{\Psi} \subseteq \Psi$ leads –by "substituting $\widehat{\Psi}$ to Ψ" in Eq. (2)– to a well-defined image $\widehat{I} : \Omega \to V$ that verifies $\widehat{I} \leq I$.

From these properties, an antiextensive filtering framework, based on component-trees, has been developed [17, 9]. This framework, illustrated in Diagram (3), consists of three successive steps:

(*i*) the construction of the component-tree $\mathfrak{T}$ associated to I;
(*ii*) the pruning of $\mathfrak{T}$, based on an *ad hoc* criterion and a pruning policy, leading to a reduced component-tree $\widehat{\mathfrak{T}}$, corresponding to the Hasse diagram of $(\widehat{\Psi}, \subseteq)$; and
(*iii*) the reconstruction of the filtered image $\widehat{I} \leq I$ induced by $\widehat{\mathfrak{T}}$.

$$\begin{array}{ccc} I & \xrightarrow{\text{Filtering}} & \widehat{I} \leq I \\ {\scriptstyle (i)}\downarrow & & \uparrow{\scriptstyle (iii)} \\ \mathfrak{T} & \xrightarrow{(ii)} & \widehat{\mathfrak{T}} \end{array} \tag{3}$$

The main purpose of this article is to provide algorithmic solutions (Secs. 4–6) for making this antiextensive filtering framework tractable in the case of component-graphs, that extend the component-trees to multivalued images. Before discussing such algorithmic issues, let us first introduce briefly this notion of component-graph.

3.3 Component-graphs

We now relax the hypothesis of totality on $\leqslant$, which can then be either a total or a partial order. In Eq. (2), any cylinder function $C_{(X,v)}$ is generated by a couple (X, v) where $X \in C[\lambda_v(I)]$ is a connected component of the thresholded image $\lambda_v(I) \subseteq \Omega$ of I at value v. In the sequel, (X, v) will be called a *valued connected component*. In particular, we define the set Θ of all the valued connected components of I as follows

$$\Theta = \bigcup_{v \in V} C[\lambda_v(I)] \times \{v\} \tag{4}$$

From the order relation $\leqslant$ defined on V, and the inclusion relation $\subseteq$ on 2^Ω, we then define the order relation $\trianglelefteq$ on Θ as follows

$$(X_1, v_1) \trianglelefteq (X_2, v_2) \iff (X_1 \subset X_2) \ \vee ((X_1 = X_2) \wedge (v_2 \leqslant v_1)) \tag{5}$$

In first approximation, the *component-graph* $\mathfrak{G}$ of I is the Hasse diagram of the ordered set $(\Theta, \trianglelefteq)$. However, three variants of component-graphs can relevantly be considered by defining two other subsets $\ddot{\Theta} \subseteq \dot{\Theta} \subseteq \Theta$ of valued connected components

$$\dot{\Theta} = \bigcup_{X \in \Psi} \{X\} \times \overset{\leqslant}{\bigtriangledown} \{v \mid X \in C[\lambda_v(I)]\} \tag{6}$$

$$\ddot{\Theta} = \Big\{(\Omega, \bot)\Big\} \cup \bigcap \Big\{\Theta' \subseteq \Theta \;\Big|\; I = \overset{\leq}{\bigvee_{K \in \Theta'}} C_K\Big\} \tag{7}$$

where $\bigtriangledown$ denotes the set of the maximal elements. Broadly speaking, Θ gathers all the valued connected components induced by I; $\dot{\Theta}$ gathers the valued connected components of maximal values for any connected components; and $\ddot{\Theta}$ gathers the valued connected components associated to cylinders functions which are sup-generators of I. We note $\blacktriangleleft$ (resp. $\dot{\blacktriangleleft}$, resp. $\ddot{\blacktriangleleft}$) the cover relation associated to the order relation $\trianglelefteq$ on Θ (resp. to the restriction of $\trianglelefteq$ to $\dot{\Theta}$, resp. to the restriction of $\trianglelefteq$ to $\ddot{\Theta}$). We then have the following definition for the three variants of component-graphs.

Definition 2 (Component-graph(s) [14]) *The Θ- (resp. $\dot{\Theta}$-, resp. $\ddot{\Theta}$-)*component-graph *of I is the Hasse diagram $\mathfrak{G} = (\Theta, \blacktriangleleft)$ (resp. $\dot{\mathfrak{G}} = (\dot{\Theta}, \dot{\blacktriangleleft})$, resp. $\ddot{\mathfrak{G}} = (\ddot{\Theta}, \ddot{\blacktriangleleft})$) of the ordered set $(\Theta, \trianglelefteq)$ (resp. $(\dot{\Theta}, \trianglelefteq)$, resp. $(\ddot{\Theta}, \trianglelefteq)$). (The term $\mathring{\Theta}$-component-graph and the notation $\mathring{\mathfrak{G}} = (\mathring{\Theta}, \mathring{\blacktriangleleft})$ will sometimes be used for the three kinds of component-graphs.)*

The component-graph is a relevant extension of the component-tree, since (*i*) both notions are compliant for totally ordered sets $(V, \leqslant)$, and (*ii*) the component-graph satisfies the image (de)composition model associated to component-tree, defined in Eq. (2).

Property 1 ([14]) *If $\leqslant$ is a total order, then two of the three variants of component-graphs, namely $\dot{\mathfrak{G}}$ and $\ddot{\mathfrak{G}}$, are isomorphic to the component-tree $\mathfrak{T}$.*

Property 2 ([14]) *For the three variants of component-graphs, we have*

$$I = \overset{\leq}{\bigvee_{v \in V}} \overset{\leq}{\bigvee_{X \in C[\lambda_v(I)]}} C_{(X,v)} = \overset{\leq}{\bigvee_{K \in \mathring{\Theta}}} C_K \tag{8}$$

4 Building the ($\ddot{\Theta}$-)component-graph

Efficient algorithms [17, 12, 4] have been proposed to build the component-tree, leading to algorithmic complexities which are nearly linear with respect to the image size. Such a linear bound is hard to reach in the case of multivalued images, in particular due to the structural properties of $\leqslant$, whose Hasse diagram is not necessarily a chain.

In the sequel, we specifically deal with the construction[3] of the $\ddot{\Theta}$-component-graph. Our motivation is twofold. Firstly, the $\ddot{\Theta}$-component-graph is the only of the three variants that guarantees to avoid the appearance of new values in the filtered images, since any valued connected component of $\ddot{\Theta}$ actually contributes to the formation of the image I (see Eq. (7)). Secondly, due to the increasing cardinality of $\ddot{\Theta}$-, $\dot{\Theta}$- and Θ, the algorithmic process that is considered for building the $\ddot{\Theta}$-component-graph may be further used as a basis to develop (more complex) algorithms for building the other variants.

4.1 Algorithmics

Algorithm 1 describes the main procedure to compute the $\ddot{\Theta}$-component-graph. The collection of valued connected components (the *nodes* of the graph) is maintained using Tarjan's union-find algorithm, based on the `makeSet`, `find` and `link` operations, similarly to [12]. The array `graph` stores, for each canonical element p , the *set* of fathers of the node of p. To avoid to insert the same link twice, each cell of `graph` is managed as a *set* data structure. The pixels are processed by decreasing values, using a priority queue `pq`. More precisely (since the values are not totally ordered) a pixel of value v is processed only if all the pixels of values $v' > v$ have been processed.

A key point of the algorithm relies on the `lowestNodes` function. This function returns the set of minimal ancestors (the *lowest* nodes which are not comparable) of a node given a value. This function plays the same role as the array `lowestNode` in the component-tree computation, in Najman and Couprie's algorithm [12]. However, in the case of the component-tree, the `lowestNode` array can be maintained efficiently: by opposition, in the case of the component-graph, the `lowestNodes` function must be recomputed each time since its result depends on the value given in parameters.

4.2 Example

We illustrate the steps of the algorithm on a toy example. Fig. 1(a) depicts the Hasse diagram of the partially ordered set $(V, \leqslant)$ used by the image I (Fig. 1(b)). Successive threshold sets of I are depicted on Fig. 1(c–g).

Let us suppose that we have processed the pixels[4] having values f, d and b. The current computed graph is depicted on Fig. 2(a). The level c is processed, and the pixel 4 is extracted from the priority queue. It has four processed neighbors: $\{2, 3, 5, 6\}$. We observe that 4 and 2 are not comparable, while $I(4) < I(3)$; then 4 is an ancestor of 3. The result of the function `lowestNodes(3,`c`)` is the set $\{3\}$, so the pixel 4 is a direct

[3] Source code, additional explanations and experiments are available at the following url: http://code.google.com/p/component-graph

[4] There is no pixel having the value e; then this value is not present in the $\ddot{\Theta}$-component-graph.

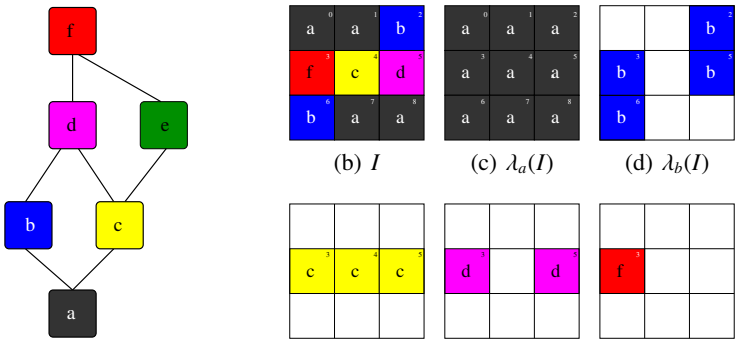
Algorithm 1: Computation of the $\ddot{\Theta}$-component-graph

```
Data: image I : Ω → V
Result: Θ̈-component-graph 𝔊̈ (array [0...N − 1] storing, for each canonical element, the
        set of its "fathers")
foreach p ∈ Ω do
    makeSet(p);
    pq.put(p,I(p));
while pq ≠ ∅ do
    p ← pq.front();
    foreach already processed neighbors q of p do
        adjNode ←find(q);
        if I(adjNode) = I(p) ∧ adjNode ≠ p then
            link(adjNode,p); // p is the new canonical element of the
               node
        else if I(adjNode) < I(p) then
            nodesList ← lowestNodes(adjNode,I(p));
            foreach n ∈ nodesList do
                if I(p) < I(n) then graph[n].insert(p);
                else link(n,p); // p is the new canonical element of the
                   node
```

Function lowestNodes(node,value)

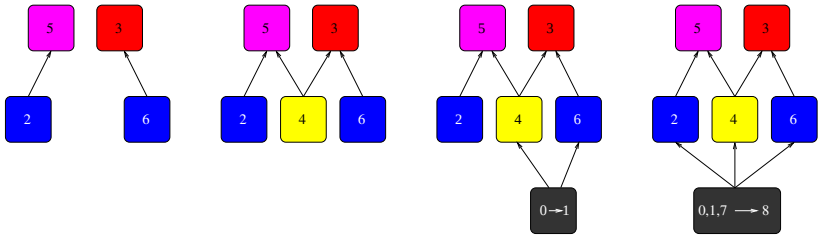
```
Data: active (array [0...N − 1] initialized to true)
Data: lowestNode (array [0...N − 1] initialized to true)
Result: nodesList: lowest ancestors of node having a value superior or equal to value
fifo.push(node);
while fifo ≠ ∅ do
    p ←fifo.get();
    foreach father f ∈ graph[p] do
        q ← find(f);
        if I(q) ⩾ v∧ active[q] then
            fifo.push(q);
            active[q]← false;
        if active[q]= false then lowestNode[p]= false;
    if lowestNode[p]= true then nodesList.add(p);
return nodesList
```

father of 3. It is inserted in the set `graph[3]`. Similarly, the pixel 4 is a direct father of 5 (see Fig. 2(b)). The level a is now processed. The point 0 is extracted. It has two processed neighbors of higher value: $\{3,4\}$. The result of the function `lowestNodes(3,`a`)` is the set $\{4,6\}$: 0 is then a direct father of these points (which are canonical elements). The result of the function `lowestNodes(4,`a`)` is the set $\{0\}$ (since 0 is now a father of 4 and 6), *i.e.*, the current visited point: nothing has to be done. The point 1 is pro-



(a) $(V, \leqslant)$ (b) I (c) $\lambda_a(I)$ (d) $\lambda_b(I)$ (e) $\lambda_c(I)$ (f) $\lambda_d(I)$ (g) $\lambda_e(I)$

Fig. 1. (a) Hasse diagram of $(V, \leqslant)$. (b) Image I. (c–g) Components of $\ddot{\Theta}$-component-graph.



(a) Proc. level b (b) Proc. level c (c) Proc. level a (d) $\ddot{\mathfrak{G}}$

Fig. 2. $\ddot{\Theta}$-component-graph computation: illustration of some steps.

cessed and compared to its five neighbors $\{0, 2, 3, 4, 5\}$. We have $I(1) = I(0)$, then it becomes the canonical element of the (partial) node $\{0, 1\}$. We have $I(1) < I(2)$ and `lowestNodes(2,`a`)`$= \{2\}$, consequently 1 becomes a father of 2. The other comparisons do not change the graph, since `lowestNodes(`i`,`a`)`$= \{1\}$ for $i = 3, 4, 5$ (see Fig. 2(c)). Finally, the points 7 and 8 are processed. When 7 is compared to any of its neighbors of greater value, the function `lowestNodes` returns the set $\{1\}$. Since 7 and 1 have the same value but belong to different nodes, 7 and 1 are linked to the same node. At last, 8 is linked to 7 and becomes the new canonical element of the node $\{0, 1, 7, 8\}$. The final $(\ddot{\Theta})$-component-graph is depicted in Fig. 2(d).

5 Pruning the component-graph

Similarly to the antiextensive filtering framework based on component-trees, filtering with component-graphs requires to define a subset $\widehat{\Theta} \subseteq \mathring{\Theta}$. This choice is based on (*i*) a selection (Boolean) criterion ρ on $\mathring{\Theta}$, and (*ii*) a pruning policy which determines, together with ρ, which parts of the component-graph should be preserved.

Pruning policies If ρ is a non-increasing criterion, then various pruning policies can be considered. For component-trees, several classical policies have been defined (for example min, direct, max, subtractive, Viterbi [17]). In the case of component-graphs, the direct and max policies can be directly transposed, while the min one leads to two variants, $\min_1$ and $\min_2$, that can be axiomatically (and recursively) defined by

$$\rho(K_1) \Longrightarrow \Big((\forall K_2 \blacktriangleright K_1, K_2 \in \widehat{\Theta}_{\min_1}) \Rightarrow K_1 \in \widehat{\Theta}_{\min_1}\Big) \tag{9}$$

$$\rho(K_1) \Longrightarrow \Big((\exists K_2 \blacktriangleright K_1, K_2 \in \widehat{\Theta}_{\min_2}) \Rightarrow K_1 \in \widehat{\Theta}_{\min_2}\Big) \tag{10}$$

These four policies lead to increasing results, *i.e.*, for a same criterion ρ, we have

$$\widehat{\Theta}_{\min_1} \subseteq \widehat{\Theta}_{\min_2} \subseteq \widehat{\Theta}_{\text{direct}} \subseteq \widehat{\Theta}_{\max} \tag{11}$$

Moreover, in the case where $(\mathring{\Theta}, \mathring{\blacktriangleleft})$ has a tree structure, the $\min_1$ and $\min_2$ policies are equivalent. In this case, which happens in particular when $(V, \leqslant)$ is totally ordered (*i.e.*, for grey-level images), we retrieve the standard min policy defined for component-trees.

Algorithmic remarks From their very definition, the $\min_1$ and $\min_2$ policies require to define $\widehat{\Theta}$ in a top-down fashion, *i.e.*, by starting from $(\Omega, \perp)$. By contrast, the max policy requires a bottom-up strategy. The direct policy –which can be applied indifferently in both directions– may be more relevantly involved in a bottom-up strategy.

The $\min_1$ and max pruning The pruning of a component-graph based on the $\min_1$ and max policies is globally straightforward, since the edges of the pruned component-graph constitute a subset of the edges of the initial component-graph. This property leads in particular to pruning procedures whose algorithmic complexity is $O(|\mathring{\Theta}| + |\mathring{\blacktriangleleft}|)$.

The $\min_2$ and direct pruning In the case where $(\mathring{\Theta}, \mathring{\blacktriangleleft})$ is not a tree, by opposition to the previous two policies, the $\min_2$ and direct ones do not imply that $\widehat{\blacktriangleleft} \subseteq \mathring{\blacktriangleleft}$. Indeed, when a element $K \in \mathring{\Theta}$ is not preserved in $\widehat{\Theta}$, each pair of edges of the form $K' \mathring{\blacktriangleleft} K \mathring{\blacktriangleleft} K''$ may lead to an edge $K' \widehat{\blacktriangleleft} K''$. However, the existence of such an edge in $(\widehat{\Theta}, \widehat{\blacktriangleleft})$ is conditioned by the non-existence of a series of edges $K' \widehat{\blacktriangleleft} K_1 \widehat{\blacktriangleleft} \ldots \widehat{\blacktriangleleft} K_t \widehat{\blacktriangleleft} K''$. Based on these considerations, a relevant approach consists of first computing $(\widehat{\Theta}, \overline{\blacktriangleleft})$, where $\overline{\blacktriangleleft}$ is a superset of $\widehat{\blacktriangleleft}$, containing redundant edges which may be obtained by transitivity from the edges of $\widehat{\blacktriangleleft}$. Such an approach, that presents an algorithmic complexity $O(|\trianglelefteq|)$, may then be followed by a standard transitive reduction procedure [1] to recover $\widehat{\blacktriangleleft}$ from $\overline{\blacktriangleleft}$.

6 Recovering a filtered image from a pruned component-graph

Once $\widehat{\Theta}$ is defined, the filtered image $\widehat{I} : \Omega \to V$ should be obtained from the cylinder functions $\{C_K \mid K \in \widehat{\Theta}\}$. However, the expression of $\widehat{I}$ via Eq. (8) (by substituting $\widehat{I}$ to I, and $\widehat{\Theta}$ to $\mathring{\Theta}$) is not necessarily well-defined. Indeed, there is no guarantee that for any $x \in \Omega$, the set $\{C_K(x) \mid K \in \widehat{\Theta}\} \subseteq V$ admits a maximum (or even a supremum) for $\leqslant$.

In such conditions, it is then necessary to define a reconstructed image $\widetilde{I} : \Omega \to V$ formed by a set $\widetilde{\Theta}$ being "as similar as possible" to $\widehat{\Theta}$. In this first study, we chose to consider the sets $\widetilde{\Theta}$ leading to reconstructed images $\widetilde{I}$ being either greater or lower than the putative image $\widehat{I}$ (w.r.t. $\leq$). To this end, let us first define the following notions.

6.1 Well-defined sets of valued connected components

We say that $\Theta' \subseteq \mathring{\Theta}$ is *well-defined* if $\bigvee^{\leq}_{K\in\Theta'} C_K$ exists, *i.e.*, if for any $x \in \Omega$, the set $\{C_K(x) \mid K \in \Theta'\}$ admits a maximum for $\leqslant$. We note $\mathring{\Xi} \subseteq 2^{\mathring{\Theta}}$ the set of all the well-defined subsets of $\mathring{\Theta}$, and for any $\Theta' \in \mathring{\Xi}$ we note $I_{\Theta'} = \bigvee^{\leq}_{K\in\Theta'} C_K$, namely the image reconstructed from Θ'. Let $\sim$ be the equivalence relation on $\mathring{\Xi}$ defined by

$$(\Theta' \sim \Theta'') \Longleftrightarrow (I_{\Theta'} = I_{\Theta''}) \tag{12}$$

that gathers the well-defined sets of valued connected components which lead to similar images. Let $\sqsubseteq$ by the (partial) relation order on the quotient set $\mathring{\Xi}/\!\sim$, defined by

$$([\Theta']_\sim \sqsubseteq [\Theta'']_\sim) \Longleftrightarrow (I_{\Theta'} \leq I_{\Theta''}) \tag{13}$$

that embeds the relation $\leq$ on images into the space of the (sets of) generating valued connected components. Given a subset of valued connected components $\widehat{\Theta} \subseteq \mathring{\Theta}$, we set

$$\mathring{\Xi}^{+}(\widehat{\Theta}) = \{\Theta' \in \mathring{\Xi} \mid \forall K \in \widehat{\Theta}, C_K \leq I_{\Theta'}\} \tag{14}$$

$$\mathring{\Xi}^{-}(\widehat{\Theta}) = \{\Theta' \in \mathring{\Xi} \mid \forall K \in \widehat{\Theta}, I_{\Theta'} \leq C_K\} \tag{15}$$

Since our purpose is to define a result image $\widetilde{I}$ being "as close as possible" to the putative image $\widehat{I}$, the choice of the solution set of valued connected components has to be made among the minimal (resp. maximal) equivalence classes $[\Theta']_\sim$, associated to the sets Θ' of $\mathring{\Xi}^{+}(\widehat{\Theta})$ (resp. $\mathring{\Xi}^{-}(\widehat{\Theta})$), for $\sqsubseteq$. More precisely, in order to respect both the content of the image I, and the nature of the component-graph ($\mathfrak{G}$, $\dot{\mathfrak{G}}$, or $\ddot{\mathfrak{G}}$), the sets Θ' have actually to be considered within $\mathring{\Xi}^{-}(\widehat{\Theta}) \cap 2^{\widehat{\Theta}}$ (resp. $\mathring{\Xi}^{+}(\widehat{\Theta}) \cap 2^{\mathring{\Theta}}$).

Note that in the case where a solution $\widetilde{\Theta}^{+}$ is determined among $\mathring{\Xi}^{+}(\widehat{\Theta})$ (*i.e.*, when $\widetilde{I}$ is greater than the putative image $\widehat{I}$), we do not necessarily have $\widehat{\Theta} \subseteq \widetilde{\Theta}^{+}$. However, we do have $(\widehat{\Theta} \cup \widetilde{\Theta}^{+}) \in \mathring{\Xi}^{+}(\widehat{\Theta})$, and $(\widehat{\Theta} \cup \widetilde{\Theta}^{+}) \sim \widetilde{\Theta}^{+}$. Broadly speaking (an equivalent version of) $\widetilde{\Theta}^{+}$ can be defined from $\widehat{\Theta}$ by relevantly adding new valued connected components.

Under such assumptions, we have in particular

$$[\emptyset]_\sim = [\{(\Omega, \bot)\}]_\sim \sqsubseteq [\widetilde{\Theta}^{-}]_\sim \sqsubseteq [\widehat{\Theta}]_\sim \sqsubseteq [\widehat{\Theta} \cup \widetilde{\Theta}^{+}]_\sim = [\widetilde{\Theta}^{+}]_\sim \sqsubseteq [\mathring{\Theta}]_\sim \tag{16}$$

The pruned set $\widehat{\Theta}$, obtained from $\mathring{\Theta}$, leads to a partition of Ω into two sets: $\Omega_w(\widehat{\Theta})$ that contains the points x such that $\{C_K(x) \mid K \in \widehat{\Theta}\}$ admits a maximum for $\leqslant$, and $\Omega_{\overline{w}}(\widehat{\Theta})$ that contains the points x such that $\{C_K(x) \mid K \in \widehat{\Theta}\}$ has several maximal elements for $\leqslant$, *i.e.*, where $\widehat{I}$ is not well-defined. We now discuss the way to deal with $\Omega_{\overline{w}}(\widehat{\Theta})$.

6.2 Algorithmics

Let us first assume that $\leqslant$ is a lower piecewise total order (LPTO), *i.e.*, that $(\{v' \in V \mid v' \leqslant v\}, \leqslant)$ is totally ordered for any $v \in V$. In such conditions, a component-graph has a tree structure [14], and we have the following property.

Property 3 *If $\leqslant$ is a LPTO, we have $\mathring{\Xi} = 2^{\mathring{\Theta}}$ and $[\widehat{\Theta}]_\sim = \bigvee^{\sqsubseteq}[\mathring{\Xi}^{-}(\widehat{\Theta})]_\sim = \bigwedge^{\sqsubseteq}[\mathring{\Xi}^{+}(\widehat{\Theta})]_\sim$.*

In other words, $\Omega_{\overline{w}}(\widehat{\Theta}) = \emptyset$, and $\widehat{I} = \widetilde{I}$ is then straightforwardly obtained from $\widehat{\Theta}$ by applying Eq. (8). In the sequel, we now suppose that $\leqslant$ is no longer a LPTO.

Reconstruction from $\mathring{\dot{\Xi}}^{+}(\widehat{\Theta})$ Let us first assume that $\leqslant$ is a lower piecewise lattice (LPL), *i.e.*, that $(\{v' \in V \mid v' \leqslant v\}, \leqslant)$ is a lattice for any $v \in V$. In such conditions, we have the following property for the Θ and the $\dot{\Theta}$-component-graphs.

Property 4 *If $\leqslant$ is a LPL, then $\bigwedge^{\sqsubseteq}[\Xi^{+}(\widehat{\Theta})]_{\sim}$ and $\bigwedge^{\sqsubseteq}[\dot{\Xi}^{+}(\widehat{\Theta})]_{\sim}$ exist. Moreover, in the case of the Θ-component-graph, $\widetilde{I} = I_{\widetilde{\Theta}^{+}}$ is directly defined from $\widehat{\Theta}$, by applying Eq. (8).*

In other words, $\Omega_{\overline{w}}(\widehat{\Theta}) \neq \emptyset$ in general, but the (unique) solution $\widetilde{I}$ can be straightforwardly obtained from $\widetilde{\Theta}^{+}$ in the case of $\mathfrak{G}$, and from $\widetilde{\Theta}^{+}$ and $\dot{\mathfrak{G}}$ in the case of $\dot{\mathfrak{G}}$.

For $\ddot{\mathfrak{G}}$ (and for any $\mathring{\mathfrak{G}}$ when $\leqslant$ is not a LPL), the definition of a solution $\widetilde{I}$ is no longer straightforward. Such a solution $\widetilde{I}$ can be obtained by defining $\widetilde{\Theta}^{+}$ by the following (non-deterministic) process. (For the sake of concision in the next property, we note the set of valued connected components of Θ' that are in conflict at a given point $x \in \Omega_{\overline{w}}(\Theta')$ as $\Theta'_x = \{(K', v') \in \Theta' \mid (x \in K') \wedge (v' \in \bigtriangledown^{\leqslant}\{v'' \mid ((K'', v'') \in \Theta') \wedge (x \in K'')\})\}$.)

Property 5 *A set $\widetilde{\Theta}^{+}$ can be defined as $\mathcal{F}^{+}(\widehat{\Theta})$ where $\mathcal{F}^{+} : 2^{\mathring{\Theta}} \to 2^{\mathring{\Theta}}$ is the extensive function (recursively) defined by*

$$\mathcal{F}^{+}(\Theta') = \begin{cases} \Theta' & \text{if } \Omega_{\overline{w}}(\Theta') = \emptyset \\ \mathcal{F}(\Theta' \cup \{(K, v)\}) & \text{if } \Omega_{\overline{w}}(\Theta') \neq \emptyset \end{cases} \tag{17}$$

where $x \in \arg_y \min |\Theta'_y|$, and $(K, v) \in \bigwedge^{\trianglelefteq}\{(K', v') \mid (\forall (K'', v''), (K'', v'') \trianglelefteq (K', v')) \wedge (x \in K')\}$

It has to be noted that in the case of $\ddot{\mathfrak{G}}$, this process is deterministic, and the solution $\widetilde{I}$ is then unique. However, this is not the case for $\mathfrak{G}$ and $\dot{\mathfrak{G}}$.

Reconstruction from $\mathring{\dot{\Xi}}^{-}(\widehat{\Theta})$ When $\widetilde{\Theta}^{-}$ is determined among $\mathring{\dot{\Xi}}^{-}(\widehat{\Theta})$, the solution $\widetilde{I}$ is not unique in general, independently from hypotheses about the kind of component-graph ($\mathfrak{G}$, $\dot{\mathfrak{G}}$, $\ddot{\mathfrak{G}}$), and the kind of order $\leqslant$ (except in the case of LPTO). Indeed, a solution $\widetilde{I}$ can be obtained by defining $\widetilde{\Theta}^{-}$ by the following (non-deterministic) process.

Property 6 *A set $\widetilde{\Theta}^{-}$ can be defined as $\mathcal{F}^{-}(\widehat{\Theta})$ where $\mathcal{F}^{-} : 2^{\widehat{\Theta}} \to 2^{\widehat{\Theta}}$ is the antiextensive function (recursively) defined by*

$$\mathcal{F}^{-}(\Theta') = \begin{cases} \Theta' & \text{if } \Omega_{\overline{w}}(\Theta') = \emptyset \\ \mathcal{F}^{-}(\Theta' \setminus \mathcal{X}) & \text{if } \Omega_{\overline{w}}(\Theta') \neq \emptyset \end{cases} \tag{18}$$

where $\mathcal{X} = \{(K', v') \in \Theta' \mid (K' \cap K \cap \Omega_{\overline{w}}(\Theta') \neq \emptyset) \wedge (v' \not\leqslant v)\}$ and $(K, v) \in \Theta'$ verifies $K \cap \Omega_{\overline{w}}(\Theta') \neq \emptyset$ and $v \in \bigtriangledown^{\leqslant}\{v' \mid ((K', v') \in \Theta') \wedge (K' \cap \Omega_{\overline{w}}(\Theta') \neq \emptyset)\}$.

7 An applicative example

In this section, we illustrate, on a simple application case, the interest of component-graphs for multimodal imaging. The proposed example involves both PET (Positron Emission Tomography) and standard CT (Computed Tomography) X-ray data. The CT

(a) (b) (c)

Fig. 3. (a) CT image. (b) PET image. (c) Detected components.

image provides homogeneous zones that characterise specific tissues and organs. The PET image provides local intensity minima where tumours are active, but with a spatial accuracy that is lower that CT information. Consequently, by coupling both grey-level value spaces into a single value space V, it may be possible to extract some valued connected components that gather the spatial accuracy of CT images and the spectral accuracy of PET ones, thus leading to accurate localisation of tumors.

In the considered example (Fig 3(a,b)), the resolution of the image is 1318×864 and $V = [0, 255] \times [0, 255]$. Since the purpose is to extract bright objects in one image and dark objects in the other one, we consider the partial order relation $\leqslant$ defined by: $(v_1, v_2) \leqslant (w_1, w_2) \Leftrightarrow (v_1 \leq w_1) \wedge (v_2 \geq w_2)$. The graph $\ddot{\mathfrak{G}}$ is computed[5] and we consider for each node the attributes "area" and "height"[6]. Graph pruning is performed by using a non-increasing criterion based on (minimal and maximal) thresholds on the attributes. The reconstruction is performed "from bottom" based on the set $\ddot{\Xi}^{-}(\widehat{\Theta})$, therefore ensuring the removal of all non-desired components. Fig. 3(c) shows the detected areas.

One may notice that using the same strategy on the component-tree of the CT-scan image is not sufficient to extract the component, while using only the PET image prevents the extraction of an accurate contour.

8 Conclusion

This article has proposed first algorithmic results that may lead to connected filtering methodologies relying on component-graphs, and thus handling multivalued images (and, more generally, any valued graph structures [22]) in a "component-tree" fashion. Some issues remain however to be considered on the way toward such methodologies.

From an algorithmic point of view, solutions to (smartly) define both Θ- and $\dot{\Theta}$-component-graphs still have to be found. Distributed strategies [21] may provide some solutions to deal with these issues. Moreover, the reduction of the algorithmic complexity of filtered image reconstruction(s), in the most general cases, also has to be carefully considered. To this end, discrete optimisation strategies may be investigated.

From a theoretical point of view, the links that may exist between this work and other attempts to extend tree structures to multivalued images [5] will also be studied.

[5] The graph computation takes 50 s on an Intel Core-i7 for 130 000 nodes and 800 000 edges.

[6] Maximal distance (L_1-norm) between the value of the node and the value of all its descendants.

References

1. A. V. Aho, M. R. Garey, and J. D. Ullman. The transitive reduction of a directed graph. *SIAM J. Comput.*, 1(2):131–137, 1972.
2. J. Angulo. Geometric algebra colour image representations and derived total orderings for morphological operators–Part I: Colour quaternions. *J. of Vis. Commun. Image R.*, 21(1):33–48, 2010.
3. E. Aptoula and S. Lefèvre. A comparative study on multivariate mathematical morphology. *Pattern Recogn.*, 40(11):2914–2929, 2007.
4. C. Berger, T. Géraud, R. Levillain, N. Widynski, A. Baillard, and E. Bertin. Effective component tree computation with application to pattern recognition in astronomical imaging. In *Proc. ICIP*, pages 41–44, 2007.
5. E. Carlinet. Extending the tree of shapes on colors. Master's thesis, ENS Cachan, 2012.
6. J. Cousty, G. Bertrand, L. Najman, and M. Couprie. Watershed cuts: Thinnings, shortest path forests, and topological watersheds. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(5):925–939, 2010.
7. J. Goutsias, H. J. A. M. Heijmans, and K. Sivakumar. Morphological operators for image sequences. *Comp. Vis. Imag. Under.*, 62(3):326–346, 1995.
8. H. Heijmans. Theoretical aspects of gray level morphology. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(6):568–592, 1991.
9. R. Jones. Connected filtering and segmentation using component trees. *Comp. Vis. Imag. Under.*, 75(3):215–228, 1999.
10. P. Monasse and F. Guichard. Scale-space from a level lines tree. *J. of Vis. Commun. Image R.*, 11(2):224–236, 2000.
11. B. Naegel and N. Passat. Component-trees and multivalued images: A comparative study. In *ISMM*, volume 5720 of *LNCS*, pages 261–171. Springer, 2009.
12. L. Najman and M. Couprie. Building the component tree in quasi-linear time. *IEEE Trans. Image Proc.*, 15(11):3531–3539, 2006.
13. N. Passat and B. Naegel. An extension of component-trees to partial orders. In *ICIP*, pages 3981–3984, 2009.
14. N. Passat and B. Naegel. Component-trees and multivalued images: Structural properties. Technical report, INRIA-00611714, 2012. url: `http://hal.inria.fr/inria-00611714`.
15. C. Ronse and V. Agnus. Morphology on label images: Flat-type operators and connections. *J. Math. Imaging Vis.*, 22(2):283–307, 2005.
16. P. Salembier and L. Garrido. Binary partition tree as an efficient representation for image processing, segmentation and information retrieval. *IEEE Trans. Image Proc.*, 9(4):561–576, 2000.
17. P. Salembier, A. Oliveras, and L. Garrido. Anti-extensive connected operators for image and sequence processing. *IEEE Trans. Image Proc.*, 7(4):555–570, 1998.
18. P. Soille. Constrained connectivity for hierarchical image partitioning and simplification. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(7):1132–1145, 2008.
19. 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 Trans. Pattern Anal. Mach. Intell.*, 29(2):272–285, 2007.
20. S. Velasco-Forero and J. Angulo. Supervised ordering in $\mathbb{R}^p$: Application to morphological processing of hyperspectral images. *IEEE Trans. Image Proc.*, 20(11):3301–3308, 2011.
21. M. H. F. Wilkinson, H. Gao, W. H. Hesselink, J.-E. Jonker, and A. Meijster. Concurrent computation of attribute filters on shared memory parallel machines. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(10):1800–1813, 2008.
22. Y. Xu, T. Géraud, and L. Najman. Morphological filtering in shape spaces: Applications using tree-based image representations. *CoRR*, abs/1204.4758, 2012.