Removing non-signicant regions in hierarchical clustering and segmentation

Benjamin Perret, Jean Cousty, Silvio Jamil Ferzoli Guimarães, Yukiko Kenmochi, Laurent Najman

Removing non-significant regions in hierarchical clustering and segmentation

Benjamin Perret[1], Jean Cousty[1], Silvio Jamil Ferzoli Guimarães[2], Yukiko Kenmochi[1] and Laurent Najman[1]

[1]Université Paris-Est, Laboratoire d'Informatique Gaspard-Monge UMR 8049, UPEMLV, ESIEE Paris, ENPC, CNRS, F-93162 Noisy-le-Grand, France
[2]PUC Minas–ICEI–DCC–VIPLAB

October 22, 2019

Abstract

We propose an efficient algorithm that removes unimportant regions from a hierarchical partition tree, while preserving the hierarchical partition structure. Various experiments demonstrate that applying this algorithm on various classification or segmentation problems does indeed improve the results by a large margin. Code is available online at `https://github.com/higra/Higra` [22].

1 Introduction

Many algorithms for image segmentation or data clustering contain a step that removes unimportant regions or clusters. In this paper, we are dealing with the more general problem of removing unimportant regions from a hierarchy of partitions, while still preserving the hierarchical partition structure. This is a common problem that appears in many different situations. For example, constrained connectivity [25] solves the chaining problem well-known as one of the issues with minimum spanning tree based approaches, but it may creates a series of small undesirable regions in situation where there is a ramp discontinuity (see [26] for an analysis of this particular case).

One way to achieve such a hierarchical simplification would be to extract all the possible segmentations from the hierarchy, and for each one of them, remove the non-important regions by merging these regions with one of their neighbours. One of the issues is that those merging steps have to be performed in a consistent way, so that the set of simplified segmentations is still a hierarchy. Another important issue is that such a process would be slow.

In the literature on transformations of hierarchical segmentations [11, 25, 5, 4, 27], there is not guarantee that unimportant regions are removed from the hierarchy. For example, small regions (with small area) can appear at very high level in the hierarchical tree, and the methods do not remove them. Thus, there is a need for post-processing the hierarchy. To the best of our knowledge, no algorithm has ever been presented for performing such a task.

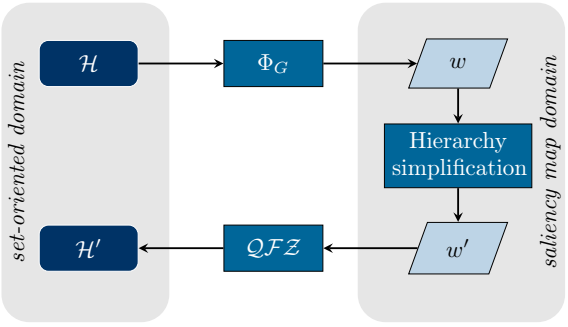
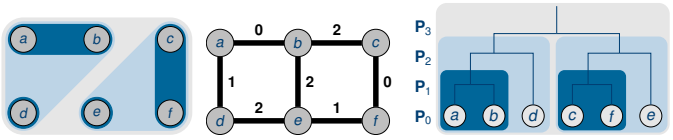


Figure 1: A flowchart of the proposed method for removing non-significant regions from a given hierarchy $\mathcal{H}$ and obtaining a new hierarchy $\mathcal{H}'$.

In order to provide such an efficient algorithm for removing unimportant regions from a hierarchy of partitions, we rely on the framework proposed in [7], where the equivalence between various hierarchical representations (dendrograms, saliency maps or minimum spanning trees) is demonstrated (see Section 2). As shown in Fig. 1, our algorithm makes use of these different representations to efficiently achieve its goal. This algorithm has been briefly introduced in the appendix of [12], but a detailed analysis and clear explanations were missing; they are provided in Section 3. Furthermore, an empirical evaluation demonstrating its practical effectiveness is performed in Section 4.



(a) (b) (c)

Figure 2: (a) An example of hierarchy of partitions, $\mathcal{H} = (\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3)$ where $\mathbf{P}_0 = \{\{a\}, \{b\}, \{c\}, \{d\}, \{e\}, \{f\}\}$, $\mathbf{P}_1 = \{\{a, b\}, \{c, f\}, \{d\}, \{e\}\}$, $\mathbf{P}_2 = \{\{a, b, d\}, \{c, e, f\}\}$, $\mathbf{P}_3 = \{\{a, b, c, d, e, f\}\}$; (b) the saliency map of $\mathcal{H}$ on a graph G, $\Phi_G(\mathcal{H})$; (c) the dendrogram of the hierarchy $\mathcal{H}$.

2 Basic notions for graph-based hierarchy processing

Any hierarchy can be equivalently represented by sets as series of nested partitions or with a characteristic function defined on the edges of a graph and called a saliency map. The core of the hierarchy simplification method which we propose in this article and which is precisely described in Section 3 considers the saliency map representations of the hierarchies. In this section, we provide the formal definitions of the set representation of hierarchies and of the saliency maps. We also highlight how one can switch between the set and the functional representations of a hierarchy.

2.1 Hierarchies of partitions

In this article, the symbol V denotes a finite set which stands for the working space. In applications to image analysis, it can be for instance the set of all image pixels or superpixels. A *partition of* V is a set $\mathbf{P}$ of nonempty disjoint subsets of V whose union is V. Each element of $\mathbf{P}$ is called a *region of* $\mathbf{P}$.

A *hierarchy on* V is a sequence $\mathcal{H} = (\mathbf{P}_0, \ldots, \mathbf{P}_\ell)$ of partitions of V such that, for any $i \in \{1, \ldots, \ell\}$, any region of the partition $\mathbf{P}_{i-1}$ is included in a region of $\mathbf{P}_i$.

Figure 2 (a) and (c) illustrate an example of a hierarchy and of its dendrogram, respectively. Dendrograms are commonly used in applications. Intuitively, the dendrogram of a hierarchy embeds the inclusion relationship between the regions of this hierarchy. More precisely, it is a tree where the nodes correspond to the regions of the hierarchy and where each region R is linked to the largest (non-empty) regions of the hierarchy which are proper subsets of R, called the children of R.

2.2 Saliency map

Any hierarchy can be represented by an edge-weighted graph [7] spanning the elements of the space V. We provide in this section the definition of such a representation called a saliency map. *A graph (spanning* V*)* is a pair $G = (V, E)$ such that E is a subset of the set of all unorderd pairs of distinct elements of V, *i.e.*, the set E is a subset of $\{\{x, y\} \subseteq V \mid x \neq y\}$. If $G = (V, E)$ is a graph, each element of V is called a *vertex of* G, and each element of E is called an *edge of* G. A subgraph of a graph $G = (V, E)$ is a graph (V', E') such that V' and E' are subsets of V and of E, respectively. If X is a graph, its vertex and edge sets are denoted by $V(X)$ and $E(X)$, respectively.

A sequence $(x_0, \ldots, x_\ell)$ of vertices of a graph G is called a *path from* x_0 *to* x_ℓ *in* G if any two successive vertices in the sequence form an edge of G, *i.e.*, for any i in $\{1, \ldots, \ell\}$, the unordered pair $\{x_{i-1}, x_i\}$ is an edge of G. A *graph is connected* whenever there is a path from any of its vertices to every other one. Let G be a graph, by extension, we say that a *subset* R *of* V *is connected* (for G) if the subgraph of G induced by R is connected, *i.e.*, the subgraph $(R, \{\{x, y\} \in E(G) \mid x \in R, y \in R\})$ is connected. A *connected component of* G is a subset R of V which is connected and maximal for this property: any proper superset of R is not connected.

In the sequel of this article, we assume that the space V is structured by a graph $G = (V, E)$. For instance, in applications to image analysis, if the set V contains the set of all pixels or superpixels of an image, the edge set E can be obtained by any pixel or superpixel adjacency relation such as the one induced by the classical 4-, 6- or 8-adjacency relations. Furthermore, we will also assume that any hierarchy on V is connected for G meaning that any region of any considered hierarchy is connected for the graph G. These assumptions correspond to the situations which are the most often encountered in hierarchical image analysis. However, they can be dropped by considering that the graph G is the complete graph on V so that any subset of V is always connected. In such case the notion of a saliency map, whose definition is recalled hereafter, corresponds exactly to the notion of ultrametric distance which is well known in classification [14].

A map w from E into the set $\mathbb{R}$ of real numbers is called a *weight map on* G. For any edge u of E, the value $w(u)$ is called the weight of u, and the pair (G, w) is called an *edge-weighted graph*. Given a graph $G = (V, E)$ and a hierarchy $\mathcal{H}$ on V, we show below how to define the saliency map $\Phi_G(\mathcal{H})$ of $\mathcal{H}$ from E to $\mathbb{R}$, which is an equivalent representation of the hierarchy $\mathcal{H}$; knowing $\mathcal{H}$ one can infer $\Phi_G(\mathcal{H})$ and, conversely, knowing $\Phi_G(\mathcal{H})$ one can recover $\mathcal{H}$.

Let us consider a hierarchy $\mathcal{H} = (\mathbf{P}_0, \ldots, \mathbf{P}_\ell)$ on V. The *saliency map of* $\mathcal{H}$ is the map $\Phi_G(\mathcal{H})$ from E to $\mathbb{L} = \{0, \ldots, \ell\} \subset \mathbb{R}$, such that the weight of any edge $u = \{x, y\}$ of G for $\Phi_G(\mathcal{H})$ is the largest value λ in $\mathbb{L}$ such that x and y belong to two distinct regions of $\mathbf{P}_\lambda$. Figure 2 (b) shows the saliency map $\Phi_G(\mathcal{H})$ of the hierarchy $\mathcal{H}$ given in Figure 2 (a).

There is a bijection between the set of all hierarchies on V and the set containing every map which is the saliency map of a hierarchy (see Theorem 1 of [7]). In the next

section, we present the quasi-flat-zone transform, denoted by $\mathcal{QFZ}$ which is the inverse of Φ_G and allows to recover the hierarchy $\mathcal{H}$ knowing only its saliency map $\Phi_G(\mathcal{H})$. These two transforms, namely Φ_G and $\mathcal{QFZ}$, make it possible to treat a hierachy either in a "set-oriented domain" (left part of Figure 1) or in the "saliency map domain" (right part of Figure 1).

An algorithm for computing the saliency map of any hierarchy $\mathcal{H}$ in linear time with respect to the size of the graph G, *i.e.*, $\mathcal{O}(|V| + |E|)$, is described in [7]. This algorithm can be sketched as follows:

1. preprocess $\mathcal{H}$ for least common ancestors searches;
2. for each edge $u = \{x, y\}$ of G taken in any order,
   (a) find the least common ancestor R of x and of y in the dendrogram of $\mathcal{H}$;
   (b) set the weight of u to the level of R in the hierarchy.

2.3 Quasi-flat zone hierarchy

Quasi-flat zone transform [19, 17, 7] maps any edge-weighted graph into a hierarchy. In particular, if the departing map is the saliency map of a hierarchy, this transform allows to recover the initial hierarchy. As the hierarchy simplification method which we propose in Section 3 treats the hierarchies from their saliency maps, the quasi-flat zone transform allows us to recover the hierarchy associated with the saliency maps produced by our simplification method (see the overview diagram of Figure 1). Intuitively, this transform considers the series of the connected component partitions induced by the successive level sets of the edge-weight map.

Given an edge-weighted graph (G, w) and a value $\lambda \in \mathbb{R}$, the *λ-level set of E for w* is defined by $w_\lambda(E) = \{u \in E \mid w(u) < \lambda\}$ and its associated subgraph $(V, w_\lambda(E))$, denoted by $w_\lambda(G)$, is called the *λ-level graph of G for w*. The set of all connected components of $w_\lambda(G)$, denoted by $\mathbf{C}(w_\lambda(G))$, is a partition of V called the *λ-level partition of G for w*. Given an edge-weighted graph (G, w), the *quasi-flat zone hierarchy $\mathcal{QFZ}(G, w)$ of (G, w)* is then the finite sequence of all λ-level partitions of G for w, ordered by increasing values of λ, namely,

$$\mathcal{QFZ}(G, w) = (\mathbf{C}(w_\lambda(G)) \mid \lambda \in \mathbb{R}).$$

3 Hierarchy simplification with an attribute criterion

The proposed simplification method is based on a regional attribute, such as region area (size) and contrast, which measures the significance of any region. It aims at transforming an initial hierarchy into a new one such that:

- the new hierarchy does not contain any region with an attribute value below a given threshold;
- the regions of the new hierarchy are either regions of the initial hierarchies or regions obtained by merging adjacent regions of the initial hierarchy.

As mentioned in the introduction, in order to efficiently perform such simplification, the hierarchies are represented by weight maps. More precisely, we consider the (spatially and functionally) minimal representation of a hierarchy introduced in [7]: it consists of a minimal weighted subgraph (in terms of inclusion relation on graphs) whose quasi-flat zone hierarchy is precisely the hierarchy that we aim to represent. Such minimal representation of a hierarchy can be obtained by considering first the graph G weighted by the saliency map of the given hierarchy and then restricting it to one of its minimum spanning trees (Theorem 12 in [7]), leading to a weighted tree (T, w). The core of the method is then to produce a new weight map w' for this graph T, standing for the saliency map representing the resulting simplified hierarchy. In order to produce such map, the edges of the tree are considered in any order. For each edge $\{x, y\}$, the largest region of the hierarchy which contains x but not y and the largest region of the hierarchy which contains y but not x are analyzed. If the attribute value of one of these two regions is below the given threshold, then the two regions must be merged. This is done by setting to 0 the weight of $\{x, y\}$ for w'. On the contrary, if the attributes of both regions are above the given threshold, then the two regions must be kept and we replicate the weight of $\{x, y\}$ for w into w'. In order to efficiently implement this method, a fundamental operation consists of finding the largest region of a hierarchy which contains one extremity of an edge but not the other. This can be done with the help of a data structure called a binary partition tree by altitude ordering. Hence, before giving a precise presentation of the simplification algorithm in Section 3.2, we first present, in Section 3.1, binary partition trees by altitude ordering together with a simple algorithm to compute them.

3.1 Binary partition tree by altitude ordering

Binary partition trees by altitude ordering (BPTAOs) are deeply related to Kruskal's minimum spanning tree algorithm [13].

More precisely, the BPTAO data structure can be seen as the (tree-based representation of a) hierarchy of partitions of V obtained during Kruskal's minimum-spanning-tree algorithm. A formal definition of this structure can be found in [8] and algorithms to construct them are presented in [20]. In this article, for the sake of completeness, we present a simple algorithm to construct it. However, the reader interested into a more efficient construction is refered to [20].

Algorithm 1: Playing with Kruskal

Data: An edge-weighted graph $((V, E), w)$
Result: An array L_{MST} to store the edges of an MST of G in non-decreasing order of weight with respect to w
Result: Its associated BPTAO B

```
1 e := 0 ; /* Initialize the index for L_MST */
2 foreach x ∈ V do B.AddNode(x) ;
  /* Assuming that V = {0, ..., n − 1} */
3 foreach {x, y} ∈ E in non-decreasing order of w
    do
4   r_x := B.FindRoot(x);
5   r_y := B.FindRoot(y);
6   if r_x ≠ r_y then
7     B.CreateParent(r_x, r_y);
8     L_MST[e] := {x, y};
9     e += 1;
```

Function B.AddNode(x)

```
1 B.Parent[x] = −1;
2 B.Size += 1;
```

Function B.FindRoot(x)

```
1 while B.Parent[x] ≥ 0 do x := B.Parent[x];
2 return x
```

Function B.CreateParent(x,y)

```
1 i := B.Size; /* index for the new node */
2 B.AddNode(i);
3 B.Parent[x] := i;
4 B.Parent[y] := i;
5 B.LeftChild[i] := x;
6 B.RightChild[i] := y;
```

This simple construction of a BPTAO from an edge weighted graph (G, w) is given in Algorithm 1. It corresponds to a particular implementation of Kruskal's algorithm. The auxiliary functions called in Algorithm 1, namely AddNode, FindRoot and CreateParent, are also described below Algorithm 1. In Algorithm 1, we initially consider a partition into singletons (Line 2) which is the first level of the BPTAO. Then, when an edge is selected by Kruskal's algorithm, we build the next level by merging the largest regions containing the vertices of the selected edge $\{x, y\}$ (Lines 3-9). In terms of tree, the newly created region R is a new node of the BPTAO B, which becomes the parent of the two nodes associated with the merged regions (Line 7). There is a direct relation between the newly created region R and the edge $\{x, y\}$ that is considered for the merging which creates the region R. In Algorithm 1, we observe that the edge $\{x, y\}$ is stored at an index e (see Line 8) of the array L_{MST} and that the index of the region R corresponding to the node in the tree data structure B is $n + e = |V| + e$ (Line 7), allowing to keep track of the relation between the edge $\{x, y\}$ and the region R for further processing. When the algorithm terminates, we then obtain:

- a minimum spanning tree of (G, w) whose edges are stored, following a non-decreasing order of weight (called an altitude ordering), in the array L_{MST};
- a tree B, called the BPTAO of (G, w) associated with L_{MST}, whose non-leaf nodes correspond to the edges the minimum spanning tree produced by Kruskal's algorithm (Line 8) and whose leaves correspond to the vertices of G (Line 2);
- an implicit mapping between the nodes of the BPTAO B and the vertices and edges of the minimum spanning tree. Any node of B stored at an index between 0 and $n - 1$ is mapped to the vertex of the graph G at the same index, whereas any node of B with an index i between n and $2n - 1$ is mapped to the edge of the minimum spanning tree stored in $L_{MST}[i - n]$, where $i \in \{0, \ldots, n - 1\}$ is the vertex set of G.

Figure 3 illustrates the relationship between a minimum spanning tree of G and its associated BPTAO B.

It should be also noticed that, if the quasi-flat zone hierarchy $\mathcal{QFZ}(G, w)$ is a binary hierarchy (*i.e.*, each region is either a singleton or the result from the merging of exactly two regions), then it is equal to the BPTAO produced by Algorithm 1 [8]. Otherwise, the hierarchy $\mathcal{QFZ}(G, w)$ can be straightforwardly recovered from B as shown in [8, 20].

For a more efficient implementation of Algorithm 1, readers are referred to [20]. Provided that the edges of the graph G are either already sorted or can be sorted in linear time, the efficient algorithm of [20] has a quasi-linear time complexity, $\mathcal{O}(|E(G)| \times \alpha(|V(G)|))$, where α is

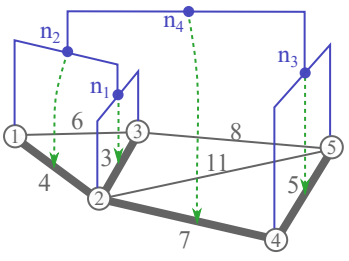


Figure 3: Given a weighted graph, its minimum spanning tree T (whose edges are thick and gray) is represented by the binary partition tree by altitude ordering B (in blue). Each leaf node corresponds to a vertex of T while each non-leaf node n_i of B corresponds to an edge of T; the correspondences are depicted in green arrows.

Algorithm 2: Hierarchy simplification by attribute

Data: A graph $G = (V, E)$ that is the working space
Data: The saliency map w of a hierarchy $\mathcal{H}$
Data: An attribute threshold value m
Result: A (saliency) saliency map w' defined on the edges of an MST T of (G, w)

1 Calculate the ordered edge array L_{MST} and its associated BPTAO B (Algorithm 1);
2 Calculate the region attribute of each node n of B and store it in $A[n]$;
3 **foreach** *non-leaf node* n *of* B /* n iterates over the set $\{|V|, \ldots, 2|V| - 1\}$ */
4 **do**
5 $\quad a_1 \coloneqq A[B.\text{LeftChild}[n]]$;
6 $\quad a_2 \coloneqq A[B.\text{RightChild}[n]]$;
7 $\quad u \coloneqq L_{MST}[n - |V|]$;
8 $\quad$ **if** $a_1 \geq m$ *and* $a_2 \geq m$ **then** $w'(u) \coloneqq w(u)$;
9 $\quad$ **else** $w'(u) \coloneqq 0$;

the extremely slowly growing inverse of the single-valued Ackermann function.

3.2 Hierarchy simplification algorithm

The hierarchy simplification method is precisely described, with the help of Playing with Kruskal algorithm (namely Algorithm 1), in Algorithm 2. In the first line of the algorithm, an MST of the saliency map w of a given hierarchy $\mathcal{H}$ and its associated BPTAO B are obtained from Algorithm 1. After calculating the attribute for every region R in B (Line 2), we can efficiently carry out the two main steps of the method for each edge $u \in E(G)$, thanks to the two structures L_{MST} and B: (1) get the attribute values of the associated connected components in Lines 5 and 6, and (2) set the new edge weight $w'(u)$ depending on the verification of the attribute criterion for the two regions merged by the edge u (Lines 8 and 9). It can be observed that Line 7 uses the mapping between the nodes of B and the edges of the considered MST, which was presented at the end of Section 3.1 and which is illustrated with the green arrows in Fig. 3.

It can be observed that, as presented in Figure 1, the hierarchy given to Algorithm 2 and the one resulting from its execution are in the form of saliency maps denoted by w and w' respectively. The tree-based representation of the simplified hierarchy resulting from Algorithm 2 can be obtained by computing the quasi-flat zone hierarchy of w' for the MST stored in L_{MST}. Such computation can be done, for instance, with the algorithm presented in [20].

4 Illustrations and assessments

This section presents qualitative and quantitative assessments of the proposed method. Our tests focus on two different hierarchical segmentation methods: the quasi-flat zone hierarchy (QFZ, see Section 2.3) and the watershed hierarchy by area (WS-Area). Watershed hierarchies were first proposed in [3, 21, 16] and have since been formalized in the context of minimum spanning forests [6]. Intuitively, the WS-Area hierarchy of an edge-weighted graph is obtained by sequentially filtering the edge weights of the graph with area closings of increasing sizes and then computing the sequence of watershed segmentations of these filtered edge weights.

Then, we consider two regional attributes to simplify those hierarchies:

1. the area of a region, defined as the number of vertices in the region; and

2. the frontier strength of a region, defined as the mean weight of the edges linking the region with its sibling, *i.e.*, the edges on the frontier between the two children of the parent region.

The area attribute of each region can be computed in linear time from the BPTAO by traversing the tree from the leaves to the root, the area of the leaves being 1 and the area of a non-leaf node being the sum of the area of its two children. The frontier strength can also be computed in linear time, by traversing the edges of the graph G, finding the lowest common ancestor of the two vertices of the edge in the BPTAO (this query can be done in constant time thanks to a linear time pre-processing of the tree [2]) and accumulating the edge weights in this region. The area attribute is used to identify non significant nodes in the QFZ hierarchy, while the frontier strength attribute is used in conjunction with the WS-Area hierarchy.

We first present illustrations of non-significant node removal on hierarchies built on point clouds and images. Then, we present extensive quantitative assessments of the benefits of our procedure for natural image analysis.

4.1 Illustrations

We first demonstrate the effectiveness of the proposed method on the hierarchical analysis of two simulated 2D point clouds (see Figure 4)[1]. Each point cloud is generated from three random distributions corresponding to three classes. We then consider the graph induced by the Delaunay triangulation of the points and we weight the edges by the Euclidean distance between the points. We observe that very small regions are branching at very high levels in the dendrogram of the QFZ hierarchy. Hence, the

[1] All the illustrations presented in this section can be reproduced using the Python Notebooks available at `https://higra.readthedocs.io/en/latest/notebooks.html`.

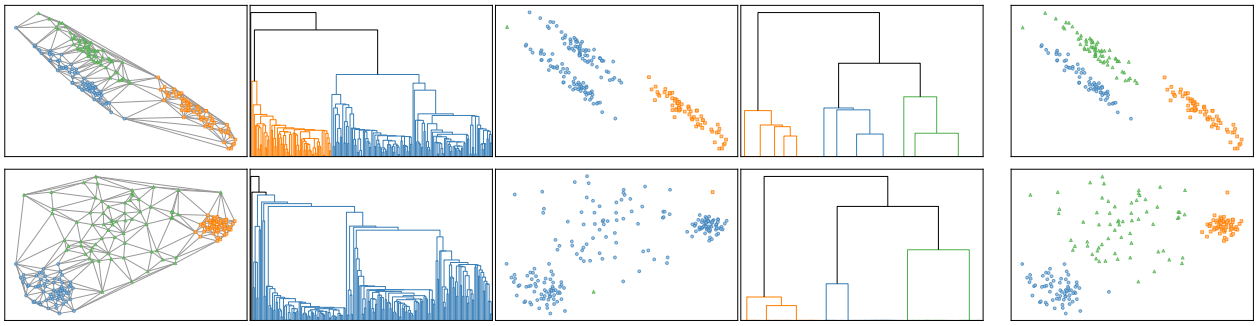


Figure 4: Removal of non-significant nodes on the QFZ hierarchical clustering of two point clouds (first and second lines). For each graph, we show from left to right: the graph with the three ground-truth clusterings of the graph vertices (red, green, and blue), the dendrogram of the QFZ hierarchy, the clustering into 3 classes for this dendrogram, the dendrogram of the simplification of the QFZ hierarchy, and the clustering into 3 classes for this simplified dendrogram. Note that the colors used to represent clusterings are arbitrary and do not represent an explicit correspondence between two different clusterings.

partition containing three regions in the hierarchy fails to correctly recovers the three clusters. By removing non-significant nodes from the QFZ hierarchy based on an area attribute (nodes containing less than 7 points are considered non-significant), we ensure that the hierarchy does not contain any small region anymore (neither at high nor at low levels). We observe that the partitions containing three regions in the simplified hierarchy correctly recovers the three clusters.

Another illustration of the effectiveness of the proposed method on hierarchical natural image analysis is demonstrated in Figure 5. In this Figure, the saliency map of a hierarchy (Section 2.2) is represented in the 2D Khalimsky grid [1, 21]: in this representation, the brightness of a contour is inversely proportional to the number of partitions of the hierarchy it belongs to, *i.e.*, dark contours are the strongest ones. We can observe that in the QFZ hierarchy, most strong contours represent very small regions located on thick transitions between different regions of the images. When the saliency map is plotted in the 2D Khalimsky grid, this suppression of small regions looks like a sharpening, in other words, thick and blurred transitions become sharp. On the contrary, we can see that WS-Area hierarchy already produces thin contours. However, it also produces a lot of non-significant contours in large homogeneous regions. After a region removal procedure with a small contour strength from the WS-Area hierarchy, most spurious contours disappear.

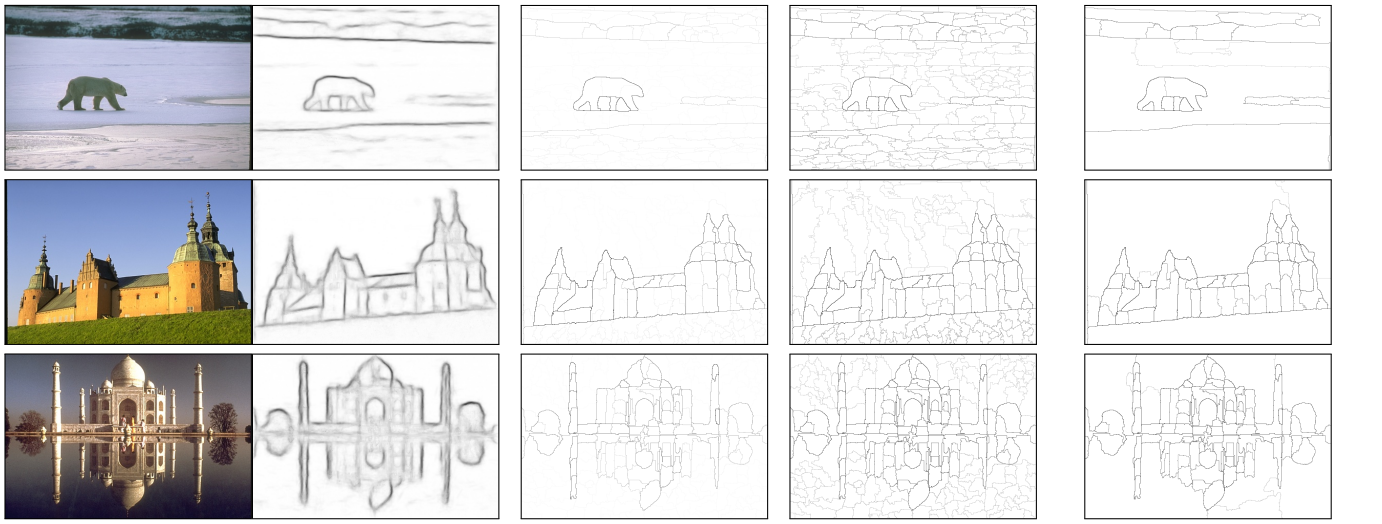
4.2 Quantitative assessment

This section presents a quantitative assessment of the proposed method on natural image analysis. We first explain the assessment methodology, the evaluation measures, and the image datasets. Then, we give the results comparing the QFZ and WS-Area hierarchies to their simplified counterparts. Finally, we also compare our results with the one obtained by the transformation of a hierarchy into its optimal cut hierarchy for the piecewise constant Mumford-Shah energy [11].

Methodology We mainly follow the supervised assessment framework proposed in [23]. We give an overview of the quality measures and readers can refer to the provided references to get detailed descriptions. The assessment framework relies on three types of measures to encompass various aspects of hierarchical representations:

1. precision-recall and F-measure on boundaries (FB) [1]. This measure evaluates the quality of the boundaries of each partition of a hierarchy with respect to a ground-truth segmentation. To evaluate a hierarchical method on a whole dataset, two aggregated measures are then defined: 1) the optimal image scale (OIS) measuring the best achievable score when taking the optimal partition in each hierarchy, and 2) the optimal data-set scale (ODS) measuring the best achievable score when taking partitions at the same level (the optimal scale) in every hierarchy;

2. fragmentation curves on the bidirectional-consistency error (BCE) [23]. The fragmentation of a partition is defined as the number of regions in the partition divided by the number of regions in the ground-truth. The fragmentation curve on BCE evaluates the quality of the regions of partitions of the hierarchy as the fragmentation increases, also with respect to a ground-truth segmentation. We consider two categories of partitions that can be extracted from a hierarchy: the partitions of the hierarchy (horizontal cuts), and the optimal partitions constructable from regions taken from any partition of the hierarchy (the



Input image | QFZ hierarchy | simplified QFZ hierarchy | WS-Area hierarchy | simplified WS-Area hierarchy

Figure 5: Removal of non-significant nodes of the QFZ hierarchy and of the WS-Area hierarchy on 4 images of the BSDS 500 dataset [1]. For each image, we show from left to right: the input image, the saliency map of the QFZ hiearchy, the saliency map of the simplified QFZ hierarchy, the saliency map of the WS-Area hierarchy, and the saliency map of the simplified WS-Area hierarchy.

Table 1: Comparison between the QFZ hierarchy, the simplified QFZ hierarchy at a given threshold, and the hierarchy of optimal Mumford-Shah (MS) cuts of QFZ.

	FB		BCE		OD		Mean score
	ODS	OIS	FOC	FHC	Mean	Median	
QFZ	0.479	0.477	0.358	0.358	0.500	0.550	0.454
QFZ + simplification 0.4‰	0.525	0.580	0.510	0.464	**0.552**	**0.613**	0.541
QFZ + simplification 0.8‰	0.537	0.589	0.533	0.475	0.533	0.579	0.541
QFZ + simplification 1.6‰	0.517	0.602	**0.550**	0.483	0.505	0.524	0.530
QFZ + simplification 3.2‰	0.515	0.543	0.541	0.479	0.467	0.448	0.499
QFZ + MS cuts [11]	0.525	0.523	0.368	0.368	0.503	0.551	0.473
QFZ + Simplification 0.8‰+ MS cuts	**0.595**	**0.637**	0.548	**0.505**	0.528	0.569	**0.564**

optimal non-horizontal cuts). Two aggregated measures are defined: the area under the curve for optimal cuts (FOC) and the area under the curve for horizontal cuts (FHC);

3. object detection measure [23, 24]. This last measure is based on supervised object detection with markers (one marker for the object and one for the background) and tries to describe an object as a set of regions taken from any partition of the hierarchy. It quantifies how well a specific object of a scene can be retrieved with different levels of information given on its position. Markers are automatically generated from the ground-truth and corresponds to: erosions of the ground-truth object/background masks (Er), skeletons of the ground-truth object/background masks (Sk), and the frame of the image (Fr). Three combination of background-foreground markers are considered: Er-Er, Fr-Sk, and Sk-Sk. The quality of a detection is measure with its Jaccard index.

The precision-recall curves and fragmentation curves are evaluated on the test set of the Pascal Context dataset [18] which consists of a pixel-wise segmentation of the last 2 498 images of the Pascal VOC'10 [10] validation set. The object detection measure is evaluated on the MS-COCO [15] dataset. Each object of the dataset is processed independently leading to a total of 291 875 objects from the 40 504 images of the MS-COCO 2014 validation set.

Results Each image of the test datasets presented in the previous section was first transformed into a 4-adjacency graph. The edge weights of the graph of an image are then defined as the mean gradient value of its two extremities, the gradient being obtained with the structured edge detector [9]. In order to evaluate the benefits of the proposed method we propose two comparisons:

Table 2: Comparison between the WS-Area hierarchy, the simplified WS-Area hierarchy at a given threshold , and the hierarchy of optimal Mumford-Shah (MS) cuts of WS-Area.

	FB		BCE		OD		Mean score
	ODS	OIS	FOC	FHC	Mean	Median	
WS-Area	0.512	0.591	0.588	0.440	0.518	0.552	0.534
WS-Area + simplification 0.05	0.522	0.592	0.589	0.445	0.519	0.554	0.536
WS-Area + simplification 0.08	0.527	0.596	0.591	0.452	0.519	0.554	0.540
WS-Area + simplification 0.10	0.530	0.599	0.591	0.457	0.518	0.553	0.541
WS-Area + simplification 0.15	**0.541**	0.604	0.593	0.470	0.511	0.539	0.543
WS-Area + simplification 0.20	**0.541**	**0.605**	0.592	0.482	0.490	0.503	0.536
WS-Area + MS cuts [11]	0.535	0.585	**0.615**	**0.514**	**0.531**	**0.576**	**0.559**

1. QFZ hierarchy versus a simplified QFZ hierarchy where small regions have been removed. The area threshold is expressed as a fraction of the total number of pixels in the image.

2. WS-Area hierarchy versus a simplified WS-Area hierarchy where regions with a common weak frontier have been merged. The strength threshold assumes that gradient values are normalized between 0 and 1.

Table 1 shows the results obtained with QFZ hierarchies. We can see that the removing of small regions provides significant improvements for the three measures. A threshold level of 0.4‰ or 0.8‰ (between 150 and 400 pixels on the tested images) offers a good compromise on the different measures.

Table 2 shows the results obtained with WS-Area hierarchies. In this case, the results are more contrasted. While a suppression of weak contours can provide significant improvement on precision-recall curves and fragmentation curves, the effect can be rapidly detrimental to the object detection measure. This issue can be due to the fact that the MS-COCO dataset contains a lot of poorly resolved objects with weak contours that can be deleted by the proposed method. However, we still see that a hierarchy simplification with moderate threshold values (between 0.05 and 0.1) improves all the considered quality measures.

Finally, the last lines of Tables 1 and 2 show the results obtained by the transformation of a hierarchy into its optimal cut hierarchy [11]. We recall that this transformation modify the level of the nodes of a hierarchy such that each partition of the transformed hierarchy is optimal for the piecewise constant Mumford-Shah energy whose regularization parameter is equal to the level of the partition. We can see that this transformation provides very good results on the WS-Area hierarchy where it can identify incorrect contours, thanks to the rich information provided by the Mumford-Shah energy, and then push them down to the bottom of the hierarchy. It is however unable to deal with the small regions present close to the top of the QFZ hierarchy as pushing them down to the bottom would require to completely collapse the hierarchy. Nevertheless, we can see that the combination of the two transformation methods, the proposed simplification strategy followed by the transformation into optimal cut hierarchy, on QFZ (last line of Table 1) gives the best result. This further support the idea that the proposed method can be used as a pre- or post-processing step to enhance the quality of hierarchical segmentation algorithms.

5 Conclusion and perspectives

In this paper, relying on the framework developed in [7], we have provided a generic solution to the common problem of removing non-significant regions from a hierarchy of partitions. The experiments demonstrate that applying this algorithm does indeed improve the results in a number of situations. Future work will combine our approach with probability functions (e.g., attention saliency) or some other criterion relying on deep learning techniques to achieve state-of-the-art results.

Acknowledgments

The authors are grateful to CNPq (Universal 421521/2016-3 and PQ 307062/2016-3), FAPEMIG (PPM-00006-16) and PUC Minas for the financial support to this work. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001 and by CAPES/COFECUB 88887.191730/2018-00.

References

[1] Arbelaez, P., Maire, M., Fowlkes, C., and Malik, J. Contour detection and hierarchical image segmentation. *IEEE PAMI 33*, 5 (2011), 898–916.

[2] Bender, M., and Farach-Colton, M. The lca problem revisited. In *LATIN 2000: Theoretical Informatics* (2000), G. Gonnet and A. Viola, Eds., Springer Berlin Heidelberg, pp. 88–94.

[3] Beucher, S. Watershed, hierarchical segmentation and waterfall algorithm. In *ISMM*, J. Serra and P. Soille, Eds. 1994, pp. 69–76.

[4] Bosilj, P., Lefèvre, S., and Kijak, E. Hierarchical image representation simplification driven by region complexity. In *International Conference on Image Analysis and Processing* (2013), Springer, pp. 562–571.

[5] Cardelino, J., Caselles, V., Bertalmio, M., and Randall, G. A contrario selection of optimal partitions for image segmentation. *SIAM Journal on Imaging Sciences 6*, 3 (2013), 1274–1317.

[6] Cousty, J., and Najman, L. Incremental algorithm for hierarchical minimum spanning forests and saliency of watershed cuts. In *ISMM* (2011), Springer, pp. 272–283.

[7] Cousty, J., Najman, L., Kenmochi, Y., and Guimarães, S. Hierarchical segmentations with graphs: Quasi-flat zones, minimum spanning trees, and saliency maps. *JMIV 60*, 4 (2018), 479–502.

[8] Cousty, J., Najman, L., and Perret, B. Constructive links between some morphological hierarchies on edge-weighted graphs. In *ISMM* (2013), Springer, pp. 86–97.

[9] Dollár, P., and Zitnick, C. L. Fast edge detection using structured forests. *IEEE PAMI 37*, 8 (2015), 1558–1570.

[10] Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. The PASCAL Visual Object Classes Challenge 2010 (VOC2010) Results. http://www.pascal-network.org/challenges/VOC/voc2010/workshop/index.html

[11] Guigues, L., Cocquerez, J., and Le Men, H. Scale-sets image analysis. *International Journal of Computer Vision 68*, 3 (Jul 2006), 289–317.

[12] Guimarães, S., Kenmochi, Y., Cousty, J., Patrocinio, Z., and Najman, L. Hierarchizing graph-based image segmentation algorithms relying on region dissimilarity: the case of the felzenszwalb-huttenlocher method. *Mathematical Morphology - Theory and Applications 2* (2017), 55–75.

[13] Kruskal, J. B. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society 7*, 1 (1956), 48–50.

[14] Leclerc, B. Description combinatoire des ultramétriques. *Mathématiques et Sciences humaines 73* (1981), 5–37.

[15] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft COCO: Common Objects in Context. In *ECCV* (2014), D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., pp. 740–755.

[16] Meyer, F. The dynamics of minima and contours. In *ISMM* (1996), P. Maragos, R. Schafer, and M. Butt, Eds., pp. 329–336.

[17] Meyer, F., and Maragos, P. Morphological scale-space representation with levelings. In *Scale-Space Theories in Computer Vision* (1999), M. Nielsen, P. Johansen, O. Olsen, and J. Weickert, Eds., pp. 187–198.

[18] Mottaghi, R., Chen, X., Liu, X., Cho, N.-G., Lee, S.-W., Fidler, S., Urtasun, R., and Yuille, A. The role of context for object detection and semantic segmentation in the wild. In *IEEE CVPR* (2014).

[19] Nagao, M., Matsuyama, T., and Ikeda, Y. Region extraction and shape analysis in aerial photographs. *Computer Graphics and Image Processing 10*, 3 (1979), 195–223.

[20] Najman, L., Cousty, J., and Perret, B. Playing with kruskal: Algorithms for morphological trees in edge-weighted graphs. In *ISMM* (2013), C. Hendriks, G. Borgefors, and R. Strand, Eds., pp. 135–146.

[21] Najman, L., and Schmitt, M. Geodesic saliency of watershed contours and hierarchical segmentation. *IEEE PAMI*.

[22] Perret, B., Chierchia, G., Cousty, J., Guimarães, S. F., Kenmochi, Y., and Najman, L. Higra: Hierarchical graph analysis. *SoftwareX 10* (2019), 1–6.

[23] Perret, B., Cousty, J., Guimarães, S. J., and Maia, D. S. Evaluation of hierarchical watersheds. *TIP 27*, 4 (April 2018), 1676–1688.

[24] Perret, B., Cousty, J., Rivera Ura, J. C., and Guimarães, S. J. F. Evaluation of morphological hierarchies for supervised segmentation. In *ISMM* (2015), J. Benediktsson, J. Chanussot, L. Najman, and H. Talbot, Eds., vol. 9082 of *LNCS*, Springer, pp. 39–50.

[25] Soille, P. Constrained connectivity for hierarchical image partitioning and simplification. *IEEE PAMI 30*, 7 (2008), 1132–1145.

[26] Soille, P., and Grazzini, J. Constrained connectivity and transition regions. In *ISMM* (2009), Springer, pp. 59–69.

[27] Xu, Y., Carlinet, E., Géraud, T., and Najman, L. Hierarchical segmentation using tree-based shape spaces. *IEEE transactions on pattern analysis and machine intelligence 39*, 3 (2016), 457–469.