Meaningful Disjoint Level Lines Selection

Yongchao Xu, Edwin Carlinet, Thierry Géraud, Laurent Najman

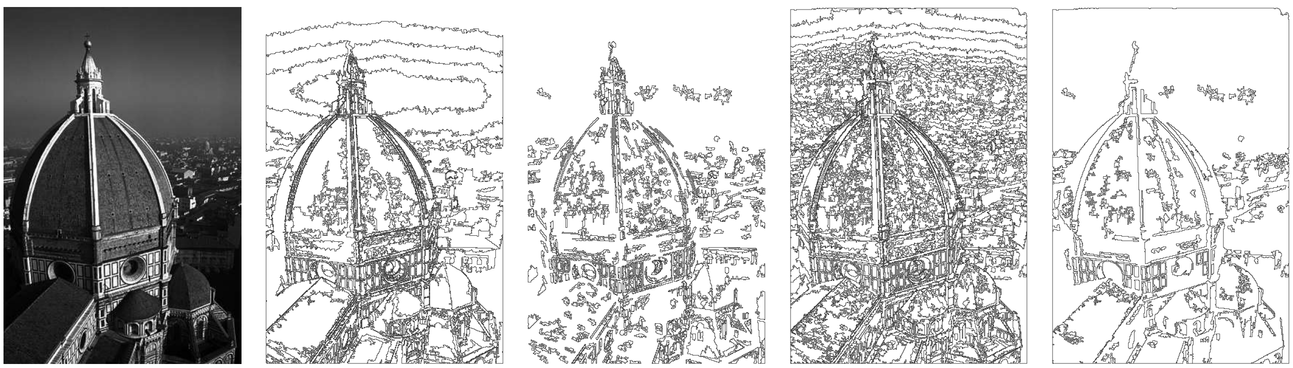
MEANINGFUL DISJOINT LEVEL LINES SELECTION

Yongchao Xu[1,2], *Edwin Carlinet*[1,2]

Thierry Géraud[1], *Laurent Najman*[2]

[1] EPITA Research and Development Laboratory (LRDE)
14-16, rue Voltaire
FR-94276 Le Kremlin-Bicêtre, France

[2] Université Paris-Est,
Laboratoire d'Informatique Gaspard-Monge (LIGM),
A3SI, ESIEE Paris, Cité Descartes, BP 99
FR-93162 Noisy-le-Grand, France



a: Original image. b: Level lines drawn every 10 levels; 717 level lines. c: Bottom-up level lines selection; 704 level lines. d: Top-down selection; 404 level lines. e: Selection by gradient's magnitude; 181 level lines.

Fig. 1: An example of the proposed method. (b-d): selected level lines.

ABSTRACT

Many methods based on the morphological notion of *shapes* (*i.e.*, connected components of level sets) have been proved to be very efficient in shape recognition and shape analysis. The inclusion relationship of the level lines (boundaries of level sets) forms the tree of shapes, a tree-based image representation with a high potential. Numerous applications using this tree representation have been proposed. In this article, we propose an efficient algorithm that extracts a set of disjoint level lines in the image. These selected level lines yield a simplified image with clean contours, which provides an intuitive idea about the main structure of the tree of shapes. Besides, we obtain a saliency map without transition problems around the contours by weighting level lines with their significance. Experimental results demonstrate the efficiency and usefulness of our method.

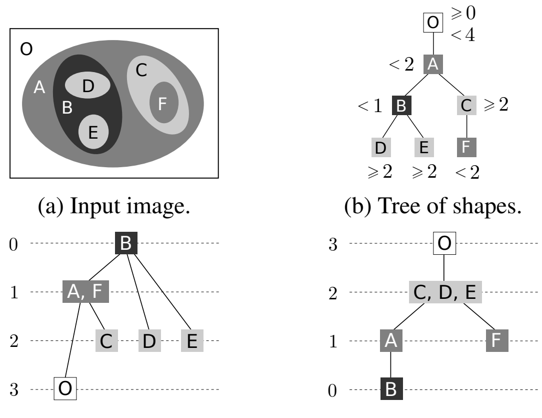
***Index Terms*—** Mathematical morphology, Level lines, Tree of shapes, Saliency map, Connected filters.

1. INTRODUCTION

Many authors [1, 2, 3] claim that significant contours of objects coincide with segments of level lines in images. Each level line is the contour of a *level set* or a *shape* which is a connected component with holes filled. The set of shapes equipped with the inclusion relationship allows to represent an image by a tree structure, called *tree of shapes* [4] or *topographic map* [1]. Numerous applications using the tree of shapes have been proposed: image filtering (grain filter [5], our proposed morphological shaping [6]), image simplification [2, 7, 8], image segmentation [9, 10, 11], scenery images analysis [12, 13], object recognition [14], and texture indexing [15]. In spite of all these abundant applications, the strong potential of the tree of shapes is still under-exploited.

The number of connected components in the tree of shapes is about as large as the number of pixels in the image. Hence, it is not straightforward to know *a priori* the structure of the tree (as shown in Fig. 1a). However, this information is fundamental for a deep tree analysis and developing tree-based methods. In this paper, we propose an efficient algorithm that selects a set of disjoint level lines of the tree of shapes from meaningful ones to meaningless ones. These level lines represent the main structure of the original tree, which can now be easily perceived from the contour clean image reconstructed from those level lines. This is the main contribution of this paper. An example of selected level lines is given in Fig. 1: In (b), too many level lines are preserved that makes it difficult to perceive the main structure of the image. In (c) and (d), the bottom-up and top-down level lines selection do not retrieve correctly the image geometry, whereas the selection by meaningfulness in (e) reveals the main structure of the image.

There exist many similar applications relying on the ex-



(a) Input image. (b) Tree of shapes. (c) Max-tree. (d) Min-tree.

Fig. 2: An example of the tree-based image representations relying on threshold decomposition.

traction of level lines from the tree of shapes to achieve image filtering, simplification, and segmentation tasks. Salembier and Wilkinson [16] give a detailed review of the tree-based filtering strategies. The works in [9, 10, 17, 2, 11, 7, 18, 8] are closest to what we propose here. In [9], the author proposes a segmentation algorithm that selects the perceptible level lines matching some criteria, *e.g.*, number of T-junctions, compactness, and contrast. In [10], the authors remove the level lines that enclose a region similar to its parent w.r.t. a histogram-based distance. Then they select interesting regions by identifying parts of the tree having an *homogeneous* histogram. Desolneux et al. [17] and Cao et al. [2] define the *meaningfulness* of a given level line using the *a contrario* model based on the computation of a number of false alarms (NFA). In [11], we also propose a context-based energy estimator to measure the significance. In [7, 18, 8], the authors proposed to extract level lines by minimizing piecewise-constant Mumford-Shah functional. To the best of our knowledge, there is no method in the literature that tries to extract a subset of level lines for visualizing the tree. The most similar one might be the tree of bilinear level lines in the chapter 7 of [19]. The authors use a bilinear interpolation to compute a continuous version of the image. So level lines from different gray levels do not touch each other, which permits a multiscale visualization.

The rest of the paper is organized as follows. Some background information about the tree of shapes and Géraud et al. [20]'s tree computation algorithm are provided in Section 2. Our method is detailed in Section 3, and we present in Section 4 some experimental results. Finally, we conclude and give some perspectives in Section 5.

2. BACKGROUND

2.1. The Tree of Shapes

Let f be an image defined on a domain Ω and with values on an ordered set V (typically $\mathbb{R}$ or $\mathbb{Z}$). For any $\lambda \in V$, the upper level sets $\mathcal{X}_\lambda$ and lower level sets $\mathcal{X}^\lambda$ of an image f are respectively defined by $\mathcal{X}_\lambda(f) = \{p \in \Omega \mid f(p) \geq \lambda\}$ and $\mathcal{X}^\lambda(f) = \{p \in \Omega \mid f(p) \leq \lambda\}$. Both upper and lower level sets have a natural inclusion structure: $\forall\, \lambda_1 \leq \lambda_2,\ \mathcal{X}_{\lambda_1} \supseteq \mathcal{X}_{\lambda_2}$ and $\mathcal{X}^{\lambda_1} \subseteq \mathcal{X}^{\lambda_2}$, which leads to two distinct and dual representations of an image, the Max-tree and the Min-tree [21]. The tree of shapes is a fusion of the Max-tree and Min-tree via the notion of *shapes* [4]. A shape is defined as a connected component of an upper or lower level set with its holes filled in. Thanks to the inclusion relationship of both kinds of level sets, the set of shapes can be structured into a tree structure, called the tree of shapes. This tree features several interesting properties: it is invariant to contrast changes and forms a self-dual, non-redundant, and complete representation of an image. Furthermore, such a tree inherently embeds a morphological scale-space (the parent of a node/shape is a larger shape) w.r.t. the theory of multi-scale analysis [22]. An example of these trees is depicted in Fig. 2.

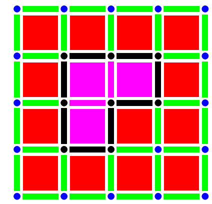


Fig. 3: An example of Khalimsky's grid by materializing the points in a 2D image with 0-faces (blue disks), 1-faces (green strips), and 2-faces (red squares).

2.2. Tree computation and representation

Though many tree of shapes computation algorithms exist [4, 23, 19], we rely on the one proposed in [20] that ensures a worst-case quasi-linear complexity. Even if this paper does not aim at explaining this algorithm since it has already been fully described by its authors, we still need to remind that it does not work directly on the domain Ω but on a simplicial version of the 2D discrete grid: the Khalimsky grid. We note $\mathcal{K}_\Omega$, the domain Ω immersed on this grid. In Fig. 3, the original points of the image are the 2-faces, the boundaries are materialized with 0-faces and 1-faces. Géraud at al. [20]'s algorithm ensures that shapes are open connected sets (*e.g.*, the purple shape in Fig. 3) and that levels lines (shape's borders) are composed of 0-faces and 1-faces only (*e.g.*, the dark curve in Fig. 3). Given a connected open set A, we note $\bar{A}$ the closure of A and ∂A the frontier of $\bar{A}$, *i.e.*, the level line associated to the shape A.

Using the tree representation from [20], a node is represented by a single pixel (2-face) called the canonical element. The tree is encoded through an image $parent : \mathcal{K}_\Omega \to \mathcal{K}_\Omega$ that states the parenthood relationship between nodes. In $parent$, each non-canonical element point is attached to the canonical element representing the node it belongs to. In the following, we denote by *getCanonical* : $\mathcal{K}_\Omega \to \mathcal{K}_\Omega$, the routine that returns the canonical element of each point in the image. We also rely on a sorted vector of pixels S such that processing S in the direct order is a top-down traversal of the tree and the reverse order is a bottom-up traversal.

```
COMPUTE_SES(parent, S, depth)
foreach x in K_Ω do SES(x) ← getCanonical(x)
foreach 2-face x in reverse order of S do
    foreach 0 and 1-face e in x̄ do
        if depth(e) < depth(SES(x)) then
            SES(x) ← getCanonical(e)
    q ← parent(x)
    if depth(SES(x)) < depth(SES(q)) then
        SES(q) ← SES(x)
return SES
```

Algorithm 1: Computation of SES.

3. DISJOINT LEVEL LINES SELECTION

In natural images, there are about as many level lines as pixels, which make the tree of shapes difficult to be visualized for further analysis. In fact, most shapes only differ from a few pixels when climbing the tree of shapes, thus many level lines overlap. In this section, we propose an efficient algorithm that allows to select a set of disjoint level lines whatever the selection ordering is.

3.1. Disjoint nodes computation

As shortly reviewed in Section 2.2, a shape A is an open connected set in $\mathcal{K}_\Omega$ and the corresponding level line is the border ∂A composed of 0 and 1-faces only. In this section, we focus on an algorithm that enables to access for each shape, the shapes having part of borders in common. In other words, for any shape A, we aim at computing the set of shapes $\mathbb{B}$ such that for each $B \in \mathbb{B}$, either $A \subset B \wedge \partial A \cap \partial B \neq \emptyset$ or $B \subset A \wedge \partial A \cap \partial B \neq \emptyset$. Actually, these relations are dual. We can focus on computing only the first one: the set of shapes in the parenthood sharing a border with ∂A. Let $\mathrm{SES}(A)$ denotes the Smallest Enclosing Shape of $\bar{A}$. Then, $\mathrm{SES}(A)$ is the smallest open set that totally includes the border of A. Since $\mathrm{SES}(A)$ is open, its border does not belong to $\mathrm{SES}(A)$ and thus, $\partial\mathrm{SES}(A) \cap \partial A = \emptyset$. In other words, the level line $\partial\mathrm{SES}(A)$ is the first (w.r.t the inclusion relationship) disjoint level line of ∂A. On the contrary, for each $B \in [A \rightsquigarrow \mathrm{SES}(A)[$ (where $x \rightsquigarrow y$ is the path from x to y in the tree), the borders of A and B intersect.

Computing $\mathrm{SES} : \mathcal{K}_\Omega \rightarrow \mathcal{K}_\Omega$ for each shape is a standard incremental attribute computation. This is given in Algorithm 1. For each 2-face x in $\mathcal{K}_\Omega$, we retain the lowest node which contains $\bar{x}$ (*i.e.*, the closure of x). This is straightforward if the depth of each point in the tree is given. When processing each 0 and 1-face e connecting to x, we have two cases: either e is an internal edge or an external edge of the shape containing x. In the former case, we have $depth(e) \geq depth(x)$, thus e is ignored. In the latter case, we have $depth(e) < depth(x)$. The node that contains e is a potential smallest enclosing shape. Since $\bar{A} = \cup\, \{\bar{x} \mid x \in A\}$, we propagate the SES from the bottom to the top of the tree incrementally, and eventually we get the SES for each node.

```
SELECT_LEVEL_LINES(parent, SES, Order)
foreach x in K_Ω do status(x) ← Null
S' = ∅
foreach canonical element x in Order do
    if status(x) ≠ Unactive then
        y ← parent(x)
        while y ≠ SES(x) and status(y) ≠ Active do
            y ← parent(y)
        if y = SES(x) then
            status(x) ← Active
            S' ← S' ∪ {x}
            y ← parent(x)
            while y ≠ SES(x) do
                status(y) ← Unactive
                y ← parent(y)
return S'
```

Algorithm 2: Disjoint level lines selection algorithm.

3.2. Final disjoint nodes selection

Based on the Smallest Enclosing Shapes, we are able to select a set of nodes in an arbitrary order such that any pair of nodes have disjoint borders. In this paper, we consider three selection strategies: top-down, bottom-up and another one based on a meaningfulness criterion. Let $Order$ denotes the sequence of shapes sorted by one of the previous strategy. $Order$ is actually an array of canonical elements. The final disjoint nodes selection is given in Algorithm 2 which proceeds as follows. Every shape has a three-state status: *Null*, the shape has not yet been processed, it is a candidate shape; *Active*, the shape has been selected; *Unactive*, the shape has been dismissed. At first, every shape has the *Null* status and we process the candidate shapes in the order given by $Order$. Given a candidate shape A, we have to check that the border does not intersect with an already *Active* shape, that is: A has not been marked *Unactive* by a sub-shape and that any shape in $[A \rightsquigarrow SES(A)[$ is not *Active*. If both conditions are fulfilled, A is selected, marked *Active* and we dismiss every shape in $]A \rightsquigarrow SES(A)[$ since they intersect A's border.

The smallest enclosing shapes computation described in Algorithm 1 has a linear complexity. For the disjoint level lines selection shown in Algorithm 2, the bottom-up selection has also a linear complexity, whereas the other two strategies have a worst case $O(N^2)$ complexity, where N is the number of shapes. We have implemented the proposed method using our C++ image processing library [24]. Processing a 640×960 pixels image takes about 3s on a regular PC station, where the tree computation takes half of the time.

4. EXPERIMENTAL RESULTS

We have experienced three different orders for disjoint level lines selection: top-down (from the root to the leaves), bottom-up (from the leaves to the root), and by meaningfulness, *e.g.*, the average of the gradient's magnitude along the level lines. An example of such disjoint level lines selec-

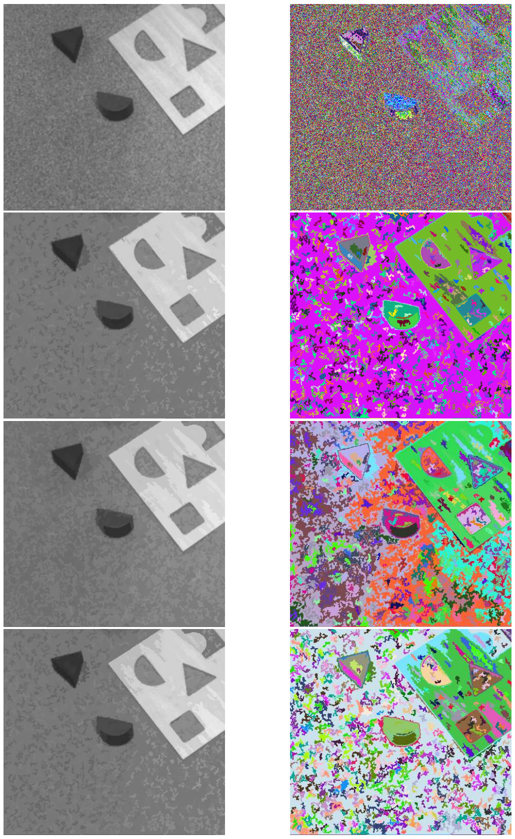
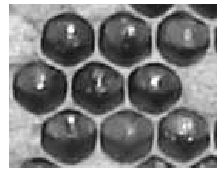
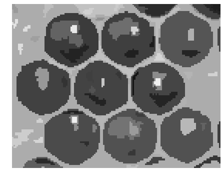


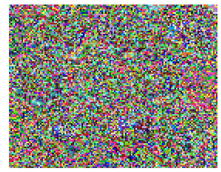
Fig. 4: An illustration of the disjoint level lines selection with different orders. From top to down: original image, bottom-up, top-down, average of gradient's magnitude ordering. Left: grayscale image; Right: corresponding randomly colorized image.



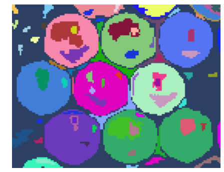
(a) Input image f.



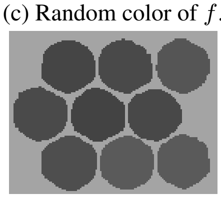
(b) Simplified image f'.



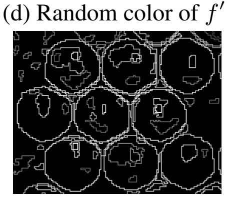
(c) Random color of f.



(d) Random color of f'.



(e) Segmented objects.

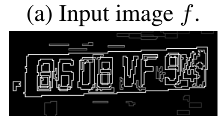


(f) Saliency map.

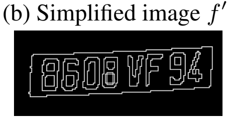
Fig. 5: An illustration of the disjoint level lines selection applied to an image of bee board.

(a) Input image f.

(b) Simplified image f'.



(c) Saliency map m.



(d) Thresholding of m.

Fig. 6: An illustration of the disjoint level lines selection applied to an image of license plate.

tions is illustrated in Fig. 4 and Fig. 1. Usually, the selection order based on meaningfulness is preferred, since it preserves better the meaningful level lines. Consequently, it offers better understanding of the tree structure for further analysis.

In Fig. 5, we show an illustration of this method on a bee board image that displays disjoint black hexagons representing places without honey. The simplified image reconstructed from the extracted disjoint level lines is shown in Fig. 5b. The selection is based on the decreasing order of average of the gradient's magnitude along the level lines. The main structure of the tree of shapes can be easily perceived in Fig. 5b and Fig. 5d. By removing those small nodes, we obtain the object segmentation result presented in Fig. 5e. In Fig. 5f, the level lines are weighted with the average of gradient's magnitude. It thus provides a hierachical segmentation represented by its saliency map.

In Fig. 6, we show an illustration of our method applied to an image of license plate. The simplified image reconstructed from the set of disjoint level lines and the saliency map are depicted respectively in Fig. 6b and in Fig. 6c. An optimal thresholding of the saliency map is illustrated in Fig. 6d.

5. CONCLUSION AND PERSPECTIVES

In this paper, we have presented an efficient algorithm that selects disjoint level lines from a tree of shapes. The selection process is based on the incremental computation of the smallest enclosing shapes. The method extracts a set of disjoint level lines from meaningful ones to meaningless ones, that briefly reflect the main structure of the tree of shapes. Not only does it allow to analyse the tree structure, but it can also be a start for more advanced tree-based processing methods. Indeed, we have shown that it provides a relevant simplification of the images where the meaningful objects are separated one from each other and also provides a saliency map without transition problem around image contours. However, in natural images, the meaningful level lines are not all strictly separated, so in the future, we would like to relax this condition and tolerate a small superposition between the selected level lines. As a second future work, we aim at applying level lines shortening that yields an image curvature microscope [25] before extracting disjoint level lines. Another major perspective is to extend this method to 3D images and the tree of shapes for color images [26, 27].

References

[1] V. Caselles, B. Coll, and J. Morel, "Topographic maps and local contrast changes in natural images," *International Journal of Computer Vision*, vol. 33, no. 1, pp. 5–27, 1999.

[2] F. Cao, P. Musé, and F. Sur, "Extracting meaningful curves from images," *Journal of Mathematical Imaging and Vision*, vol. 22, pp. 159–181, 2005.

[3] J. L. Lisani, L. Moisan, P. Monasse, and J. M. Morel, "On the theory of planar shape," *Multiscale Modeling & Simulation*, vol. 1, no. 1, pp. 1–24, 2003.

[4] P. Monasse and F. Guichard, "Fast computation of a contrast-invariant image representation," *IEEE Trans. on Image Processing*, vol. 9, no. 5, pp. 860–872, 2000.

[5] V. Caselles and P. Monasse, "Grain filters," *Journal of Mathematical Imaging and Vision*, vol. 17, no. 3, pp. 249–270, 2002.

[6] Y. Xu, T. Géraud, and L. Najman, "Morphological Filtering in Shape Spaces: Applications using Tree-Based Image Representations," in *Proc. of Intl. Conf. on Pattern Recognition*, 2012, pp. 485–488.

[7] C. Ballester, V. Caselles, L. Igual, and L. Garrido, "Level lines selection with variational models for segmentation and encoding," *Journal of Mathematical Imaging and Vision*, vol. 27, pp. 5–27, 2007.

[8] Y. Xu, T. Géraud, and L. Najman, "Salient level lines selection using the mumford-shah functional," in *Proc. of the IEEE International Conference on Image Processing*, 2013, pp. 1227–1231.

[9] A. Pardo, "Semantic image segmentation using morphological tools," in *Proc. of IEEE Intl. Conf. on Image Processing*, 2002, pp. 745–748.

[10] J. Cardelino, G. Randall, M. Bertalmio, and V. Caselles, "Region based segmentation using the tree of shapes," in *Proc. of IEEE Intl. Conf. on Image Processing*, 2006, pp. 2421–2424.

[11] Y. Xu, T. Géraud, and L. Najman, "Context-based energy estimator: Application to object segmentation on the tree of shapes," in *Proc. of IEEE Intl. Conf. on Image Processing*. IEEE, 2012, pp. 1577–1580.

[12] Y. Song and A. Zhang, "Locating image background by monotonic tree," in *Proceedings of the 6th Joint Conference on Information Science*. Association for Intelligent Machinery, Inc., 2002, pp. 879–884.

[13] Y. Song and A. Zhang, "Analyzing scenery images by monotonic tree," *Multimedia Systems*, vol. 8, no. 6, pp. 495–511, 2003.

[14] Y. Pan, J. D. Birdwell, and S. M. Djouadi, "Preferential image segmentation using trees of shapes," *IEEE Transactions on Image Processing*, vol. 18, no. 4, pp. 854–866, 2009.

[15] G. Xia, J. Delon, and Y. Gousseau, "Shape-based invariant texture indexing," *International Journal of Computer Vision*, vol. 88, no. 3, pp. 382–403, 2010.

[16] P. Salembier and M. H. F. Wilkinson, "Connected operators," *IEEE Signal Processing Magazine*, vol. 26, no. 6, pp. 136–157, 2009.

[17] A. Desolneux, L. Moisan, and J. Morel, "Edge detection by helmholtz principle," *Journal of Mathematical Imaging and Vision*, vol. 14, no. 3, pp. 271–284, 2001.

[18] Y. Pan, "Top-down image segmentation using the Mumford-Shah functional and level set image representation," in *Proc. of the IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing*. IEEE, 2009, pp. 1241–1244.

[19] V. Caselles and P. Monasse, *Geometric Description of Images as Topographic Maps*, 1st ed. Springer Publishing Company, Incorporated, 2009.

[20] T. Géraud, E. Carlinet, S. Crozet, and L. Najman, "A quasi-linear algorithm to compute the tree of shapes of *n*D images," in *Proc. of Intl. Symp. on Mathematical Morphology*, 2013, pp. 98–110.

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

[22] L. Guigues, J. P. Cocquerez, and H. L. Men, "Scale-sets image analysis," *International Journal of Computer Vision*, vol. 68, no. 3, pp. 289–317, 2006.

[23] Y. Song, "A topdown algorithm for computation of level line trees," *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2107–2116, Aug. 2007.

[24] R. Levillain, T. Géraud, and L. Najman, "Why and how to design a generic and efficient image processing framework: The case of the Milena library," in *Proc. of IEEE Intl. Conf. on Image Processing*, 2010, pp. 1941–1944. http://olena.lrde.epita.fr .

[25] A. Ciomaga, P. Monasse, and J. M. Morel, "Level lines shortening yields an image curvature microscope," in *Proc. of IEEE Intl. Conf. on Image Processing*, 2010, pp. 4129–4132.

[26] E. Carlinet and T. Géraud, "A morphological tree of shapes for color images," in *Proc. of Intl. Conf. on Pattern Recognition*, 2014, to appear.

[27] E. Carlinet and T. Géraud, "Getting a morphological tree of shapes for multivariate images: paths, traps, and pitfalls," in *Proc. of IEEE Intl. Conf. on Image Processing*, 2014, to appear.