Fast image and video segmentation based on alpha-tree multiscale representation

François Merciol, Sébastien Lefèvre

Fast Image and Video Segmentation based on α-tree Multiscale Representation

François Merciol Sébastien Lefèvre
Université de Bretagne-Sud, IRISA
Vannes, France
Email: {Francois.Merciol,Sebastien.Lefevre}@univ-ubs.fr

***Abstract*—In this paper, we propose to rely on a recent image representation model, namely the α-tree, to achieve efficient segmentation of images and videos. The α-tree is a multiscale representation of an image, based on its quasi-flat zones. An in-depth study of this tree reveals some interesting features of image pixels and regions. These features are then used in the design of both automatic and interactive segmentation algorithms. Interactivity is achieved thanks to a new and efficient implementation scheme. Experiments on the Berkeley Segmentation Dataset lead to very promising results.**

***Keywords*-α-tree ; Multiscale Representation ; Image Segmentation; Video Matting**

I. Introduction

Image segmentation is a fundamental problem in computer vision. While it has been deeply studied for decades, it is still an open issue. Indeed, a generic image segmentation hardly achieves accurate results on a wide panel of images. Conversely, interactive methods may achieve better results by involving the user in the process, but they require a low computational cost and are irrelevant if a large set of images has to be segmented.

Recently, a new image model, namely the α-tree [1], has been introduced as a powerful tool for multiscale image representation. It offers a compact and efficient way to access image content, and can be further exploited in various image analysis and processing tasks. We consider here the α-tree for image segmentation purpose, and study how this new image model can be used in such a context. Indeed, some relevant image features can be extracted from the tree, leading then in segmentation methods operating either automatically or interactively. Moreover, we propose an efficient implementation scheme which ensures user interactivity and extension to video data. Preliminary results obtained on the Berkeley Segmentation Dataset are very promising and show the relevance of the α-tree in image processing and analysis.

This paper is organized as follows. In the next section, we recall the definitions of flat and quasi-flat zones, that lead to the α-tree model for image representation. We then describe our contribution in Sec. III where we study how the α-tree can provide relevant features for image segmentation before introducing a new segmentation method and its efficient implementation. In Sec. IV, we discuss parameter settings and provide an experimental evaluation of our method on the Berkeley Segmentation Dataset. We also provide an insight into video matting. The last section is devoted to concluding remarks and future directions.

II. Background

The α-tree image model is a multiscale representation of an image through its α-zones. We recall here the notions of flat zones, quasi-flat zones (including α-zones) and finally the recent α-tree model.

In the following, we will use the notations used in [1]. We will denote by I a digital image and E its definition domain. Let us recall that an image segmentation is a partition $\mathbf{P}$ of E, i.e. a mapping $x \to \mathbf{P}(x)$ from E into $\mathcal{P}(E)$ such that $\forall x \in E \Rightarrow x \in \mathbf{P}(x)$ and $\forall x, y \in E \Rightarrow \mathbf{P}(x) = \mathbf{P}(y)$ or $\mathbf{P}(x) \cap \mathbf{P}(y) = \emptyset$, with $\mathbf{P}(x)$ indicating a cell of $\mathbf{P}$ containing a point $x \in E$. We thus have $\bigcup_{x \in E} \mathbf{P}(x) = E$. Moreover, we will write $\pi(x \rightsquigarrow y)$ a path of length N between any two elements $x, y \in E$, i.e. a chain of pairwise adjacent elements $\langle x = x_0, x_1, \ldots, x_{N-1} = y \rangle$. Finally, let $\Pi \neq \emptyset$ be the set of all possible paths between x and y. The minimum dissimilarity metric between x and y is defined as

$$\hat{d}(x,y) = \bigwedge_{\pi \in \Pi} \left\{ \bigvee_{i \in [0,\ldots,N-1]} \left\{ d(x_i, x_{i+1}) \mid x_i, x_{i+1} \in \Pi \right\} \right\} \quad (1)$$

with $d(x,y)$ a predefined dissimilarity measure between attributes of x and y (i.e. pixel intensities).

In a digital image, flat zones are defined as connected sets of pixels sharing the same value. Formally, the flat zone of x is defined as

$$\mathcal{Z}(x) = \{x\} \cup \{y \mid \exists \pi(x \rightsquigarrow y) : \forall x_i \in \pi(x \rightsquigarrow y) \wedge x_i \neq y \Rightarrow d(x_i, x_{i+1}) = 0\}. \quad (2)$$

In the field of Mathematical Morphology, flat zones have been shown to be elements with nice properties [2]. Indeed, the partition of an image into its flat zones most often includes any relevant image segmentation, since objects edges are located between neighboring pixels with different values, i.e. belonging to different flat zones. However, the practical usage of flat zones is limited since it leads to an extreme oversegmentation, flat zones being made of only a few pixels. To counter this problem, softer definitions have

been introduced under the name quasi-flat zones. A recent survey related to quasi-flat zones is provided by Soille in [3].

The simpler and most widely used definition of quasi-flat zones is called α-zone. For a given pixel x, its α-zone noted α-$\mathcal{Z}(x)$ is made of all pixels reachable from x through a path with intermediary steps not higher than α. Using the previous definitions, we have

$$\alpha\text{-}\mathcal{Z}(x) = \{x\} \cup \{y \mid \exists \pi(x \rightsquigarrow y) : \forall x_i \in \pi(x \rightsquigarrow y) \wedge x_i \neq y \Rightarrow d(x_i, x_{i+1}) \leq \alpha\}, \quad (3)$$

the specific case of $\alpha = 0$ leading to standard flat zones. Let us observe that α-zones define a partition or segmentation, i.e. $\bigcup_{x \in E} \alpha\text{-}\mathcal{Z}(x) = E$. The main drawback of α-zones is their purely local behavior, which can lead to an artifact called chaining effect. This is observed when the successive steps in a path $\pi(x \rightsquigarrow y)$ are low (w.r.t α) while the dissimilarity measure between x and y is high, for instance in the case of a gradual transition from black to white.

Soille [3], and later Soille and Grazzini [4], have then introduced new α-zone definitions with additional constraints. Among them, the (α, ω)-zones rely on both a local range to be compared to α and a global range to be compared with ω:

$$(\alpha,\omega)\text{-}\mathcal{Z}(x) = \bigvee \{\alpha_i\text{-}\mathcal{Z} \mid \alpha_i \leq \alpha \vee \mathrm{R}(\alpha_i\text{-}\mathcal{Z}) \leq \omega\} \quad (4)$$

with $\mathrm{R}(\cdot)$ denoting the global range of the quasi-flat zone. Such quasi-flat zones have been already used in image segmentation through an interactive framework [5]. However, this require to set predefined values for α and ω parameters, which is not an easy task in practice.

Another way to use the α-zones has been recently reported by Ouzounis and Soille in [1]. In this seminal work, they introduce the concept of α-tree based on a partition pyramid of E. This latter is a mapping $\Delta^A : E \rightarrow \Pi^A(E)$ defined by

$$\Delta^A = \{\mathbf{P}^{\alpha=\mathbf{0}}, \mathbf{P}^{\alpha=\mathbf{1}}, \ldots, \mathbf{P}^{\alpha=\alpha_{\max}}\} \mid \mathbf{P}^{\alpha'} \preceq \mathbf{P}^{\alpha}, \forall \alpha, \alpha' \in A,\ \alpha' < \alpha \quad (5)$$

with $\Pi^A(E)$ the set of all α-partitions of the definition domain of I (i.e. E) and $A = [0, 1, \ldots, \alpha_{\max}]$ the range of α values. The relation $\preceq$ denotes a notion of order with respect to $\alpha \in A$:

$$\forall x \in E, \alpha' < \alpha \Rightarrow \alpha'\text{-}\mathcal{Z}(x) \subseteq \alpha\text{-}\mathcal{Z}(x) \Rightarrow \mathbf{P}^{\alpha'} \preceq \mathbf{P}^{\alpha} \quad (6)$$

A pyramid level $\Delta^A_\alpha \in \Delta^A$ is a partition $\mathbf{P}^\alpha$ of E, with $\alpha \in A$. Let $j \in J^\alpha$, in which $J^\alpha \subseteq \mathbb{Z}$ is an index set, employed to address the α-zones making up $\mathbf{P}^\alpha$. Finally, the α-partition hierarchy $\mathbf{\Lambda}^A$ is a family of ordered mappings $\mathbf{\Lambda}^A_\alpha : J^\alpha \rightarrow K^\alpha$ with $K^\alpha \subseteq J^\alpha$, given by:

$$\mathbf{\Lambda}^A = \{\mathbf{\Lambda}^A_{\alpha=0}, \mathbf{\Lambda}^A_{\alpha=1}, \ldots, \mathbf{\Lambda}^A_{\alpha=\alpha_{\max}}\} \mid \mathbf{\Lambda}^A_{\alpha'} \prec \mathbf{\Lambda}^A_{\alpha}, \forall \alpha, \alpha' \in A,\ \alpha' < \alpha \quad (7)$$

and $\forall \alpha \in A \setminus 0$ and $\forall j \in J^\alpha$:

$$\mathbf{\Lambda}^A_\alpha = \{\alpha_j\text{-}\mathcal{Z} | (\alpha_j\text{-}\mathcal{Z} \in \Delta^A_\alpha) \wedge (\alpha_j\text{-}\mathcal{Z} \notin \Delta^A_{\alpha-1})\} \quad (8)$$

III. From α-tree to Image Segmentation

In the previous section, we have recalled from [1] the new concept of α-tree. This hierarchical image model leads to new and efficient ways of accessing and manipulating the image content. We study here how such a model can be used in the context of image segmentation. Thus, we first discuss how to build the α-tree from color images and to derive from this tree some segmentation criteria. Then we introduce a new segmentation algorithm based on this model. Finally, we present an efficient algorithm for α-tree computation which ensures low computation time.

A. α-tree on Color Images

The definitions provided in the previous section apply directly to the greyscale case. However, when dealing with color images, dissimilarity metrics require some specific attention. Indeed, each pixel is not anymore described by a single scalar value but rather by a vector of color components, most often the RGB triplet. Contrary to other morphological operators, it is not necessary here to select a vector ordering to compare pixels [6] since the ordering is imposed on the (scalar) dissimilarity values. The local dissimilarity metric computed between two pixels then measures the dissimilarity between their two colors. While various metrics are available, we have here chosen to use the Chebyshev distance rather than other measures such as Euclidean or Manhattan distance. Let us consider two colors $\mathbf{c} = (r, g, b)$ and $\mathbf{c'} = (r', g', b')$, the Chebyshev distance between $\mathbf{c}$ and $\mathbf{c}'$ is given by $\max(|r - r'|, |g - g'|, |b - b'|)$. Since the Chebyshev distance is the L_∞ norm of the absolute difference vector $|c - c'|$, the range of possible distance values is kept low and similar to the input range of each color component (i.e. 256 for an 8-bit image). On the opposite, Manhattan or (worst) Euclidean distance lead to a range respectively equal to 3×256 and 256^3 possible distance values. Since this range is to be compared with the various α values, we prefer to set the depth of α-tree $A = 256$ rather than $A = 16,777,216$ for practical reasons related to computation time. Let us notice that a quantization might be involved to further reduce the range A, with the risk to consider two pixels with different colors in the same flat zone. The same motivation led us to keep the RGB space instead of performing a conversion to a possibly more relevant space for image analysis, e.g. Lab or HSL.

As we will show later, we do not involve global constraints (e.g. related to ω) at the time of α-tree computation,

but impose them on the tree in a post-processing before the final segmentation step.

B. Analysis of α-tree

The α-tree image model can be studied from a pixel basis. Indeed, for every given pixel x of an image, it is possible to focus on the different α-zones (or nodes in the α-tree) it belongs to, or in other words $\Delta^A(x)$. Various measures might be considered to perform this analysis and to describe the nodes $\mathbf{P}^\alpha(x)$ related to a pixel x at the different scales or α values. These measures provide a characterization of the nodes, e.g. related to their color, texture, shape, size, etc. In the following, we will consider the zone area or size (i.e. number of pixels) as a comprehensive measure to perform a characterization of each pixel based on the α-tree:

$$\Omega^A(x) = \{|\mathbf{P}^{\alpha=\mathbf{0}}(x)|, |\mathbf{P}^{\alpha=\mathbf{1}}(x)|, \ldots, |\mathbf{P}^{\alpha_{\max}}(x)|\} \quad (9)$$

with $\forall \alpha, \alpha', \ \alpha' < \alpha, \ |\mathbf{P}^{\alpha'}(x)| < |\mathbf{P}^{\alpha}(x)|$ and $|E|$ the cardinality of set E (computed as its number of pixels). Of course, other measures might be involved, as long as they are also monotonically increasing.

To illustrate the descriptive power of the α-tree and its related measures (e.g. Ω), we have considered the sample image of Fig. 1 taken from the Berkeley Segmentation Dataset [7]. In this figure, 3 α-zones are highlighted, corresponding to the sky, the wall and the sea. While the sky is a very wide and uniform region (α close to 0), the wall is a textured area and the sea is a uniform region but surrounded by complex objects. Ω^A curves for pixels belonging to these 3 zones are then plotted in Fig. 2. From these plots we can made several observations:

- even in the case of very uniform areas, flat zones do not spread among a lot of pixels. Such areas are better represented by quasi-flat zones with a low (but not null) α value;
- textured areas require higher values of α to be correctly described. The required α value is directly proportional to the contrast of the texture. In practice, some textures such as the wall can be extracted with relatively low value of α;
- in some cases, the complexity of the scene leads to a large number of small plateaus separated by relatively small steps. This is observed with the sea which is surrounded by many objects. In such cases, an automatic segmentation process most often fails to identify correct object edges;
- for all pixels, the monotonically increasing curve ends with the same plateau which correspond to the whole image after the inclusion of some outlier pixels.

More generally, we can distinguish between several patterns in these curves: steps, plateaus, and slopes of various strength.

Steps are observed when increasing α value leads to a high growth of the α-zone area. In other words, the region has passed a border and now includes significant neighboring region(s). The height of the step is directly related to the size of objects added to the growing α-zone.

Figure 1. A sample image with 3 selected objects.

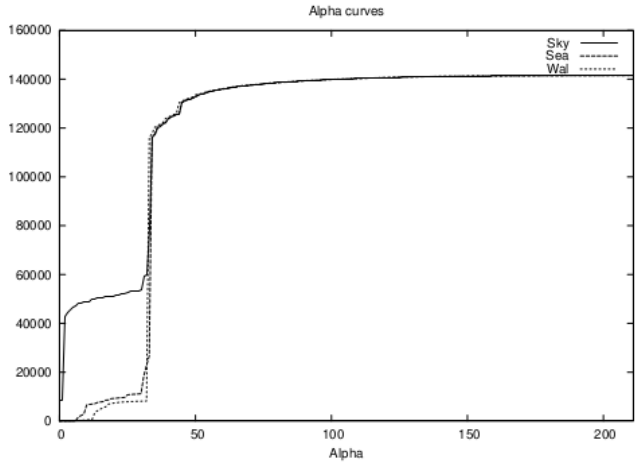

Figure 2. Ω^A evolution curves (zone area w.r.t α) for the 3 objects of Fig. 1.

Plateaus are zones for which a slightly increment of α will not change the α-zones. They are most often located between two steps, i.e. between two jumps over borders. This means that plateaus represent indeed border areas. The length of the plateau is then directly related to the strength of the edge, with most significant edges being characterized by wide plateaus.

Slopes may have two different explanations. They are observed in case of very blurred edges (leading to some non flat plateaus) but also in the case of successive inclusions of small objects (as with the sea in Fig. 1).

C. Enriching the α-tree with Additional Features

We have seen that Ω curves built from α-tree may be analyzed to identify accurate segmentation criteria. Nevertheless, these measures only gather local dissimilarity and size information. To bring further knowledge to the

segmentation process, we propose to equip the tree with additional information, similar to the constraints imposed on quasi-flat zones [1], [4]. More precisely, we consider a global range constraint related to the hue and/or brightness of the α-zones. To do so, we compute for each node both a reduced average hue and an average brightness using a HSB color space.

A specific attention is given to hue due to its particular behavior. Indeed, we consider the hue of a pixel or zone only if related saturation is higher than a given threshold $T_{\text{saturation}}$. Then, we compute the mean of relevant hues through circular statistics [8]. To ensure a robust measure, we have designed a specific scheme to compute the reduced average hue. Thus, we only consider hues which are similar (w.r.t. a given threshold T_{angle}) to the most representative hue to remove the influence of outlier hue values in the computation of the mean of several hues. Finally, a hue average is considered as significant only if the number of removed hues is kept lower than another threshold T_{outliers}. In the other case, we mark the hue node as insignificant. The brightness and hue values and the hue significance flag will be further used in the segmentation process.

D. Image Segmentation Algorithm

The α-tree, equipped with some complementary features making possible the use of additional constraints, is ready to be used for image segmentation. To do so, we aim to focus on the characteristic elements of pixelwise curves described previously. Plateaus represent objects of the image which are stable over α values (thus sharing probably some similarities with Maximally Stable Extremal Regions (MSER), but this is beyond the scope of this paper). Unfortunately, a given pixel may lead to several plateaus. In order to focus on relevant objects only, we consider nodes in the tree with an area comprised in a predefined interval $[T_{\min}, T_{\max}]$.

Starting from the tree root, we further analyze the nodes using the global hue constraint discussed previously. This additional constraint has shown great improvement (thus we do not show in this paper the use of local constraint only). The detailed algorithm is given in Alg. 3. It is made of two steps: (1) attribute computation which is a bottom-up approach required to compute average hue and brightness values; (2) automatic node selection which is a top-down approach starting from the root and selecting only relevant nodes. If these nodes fulfill the predefined criteria, their related subtrees are not further explored.

Segmenting color images based on a global hue constraint coupled with local differences in the RGB space might not seem to be the most relevant strategy. Let us recall that our goal here is to illustrate the potential of the α-tree model in the context of image segmentation. Our proposal is not at all limited to these local and global criteria, and can be set up with other more robust criteria. However, this point is not addressed here and is left for future studies.

Result: automatic segmentation

```
// Step #1 bottom up attribute
   computation
computeAttributes( root, parameters) begin
    if alpha = 0 then
        set size of pixels;
    else
        foreach spare in siblingSpares do
            foreach subset in spare do
                computeAttributes(subset,
                parameters);
        foreach spare in siblingSpares do
            foreach subset in Spare do
                define allAttributes with attributes of
                subset;
        // to be sure to start with
           significant data
        inverseSort( allAttributes);
        forall the allAttributes do
            aggregate pixels according to their features;
// Step #2 automatic node selection
autoSelect( root, parameters) begin
    if minArea then
        // nothing else in that branch
    else if not maxArea and (valid hue or valid
    brightness) then
        addSelection( root) ;
    else if root is a leaf then
        // nothing else in that branch
    foreach spare in root.siblingSpares do
        for subset in spare do
            autoSelect(subset, parameters);
```

Figure 3. Alpha-Tree Automatic Segmentation

The complexity of the proposed procedure has been estimated as follows: $O(N \log(A))$ for the first step, where every nodes have to be scanned; $O(K \log(A))$ for the second step. Here N denotes the number of pixels in the image while K is the number of relevant nodes or objects to be kept (limited to 100 in practice).

E. Efficient Implementation of α-tree Computation

Manipulating an image through its α-tree leads to very efficient algorithms (such as segmentation as addressed in this paper). However, the computational cost of the α-tree building step might also require some study to improve efficiency.

We consider here a parallel implementation to speedup the building process and introduce the parallel algorithm given in Alg. 4. In this algorithm, the function **distance** refers to the local dissimilarity measure in use (here the Chebyshev distance). The function **link** is more complex and aims to link two leaves by looking for a compatible node. When the

two considered nodes do not belong to the same subtree (i.e. they are not connected), this function creates a link between the two subtrees (which can be as high as the tree root if necessary). Thus the image model is always a single tree and not a forest.

```
Result: α-tree building
pix ← createPixels() ;
for x ← 0 to width do
    for y ← 0 to height do
        val[x][y] ← compute( h, s, b);
for scale in scales do
    for tile at scale do
        m ← (top+bottom)/2;
        for x ← left to right do
            d ← distance( pix[x][m-1], pix[x][m]);
            link( val[x][m-1], val[x][m], d);
        c ← (left+right)/2;
        for y ← top to bottom do
            d ← distance( pix[c-1][y], pix[c][y]);
            link( val[c-1][y], val[c][y], d);
reduceTree;
```

Figure 4. Alpha-Tree Creation

The input image is split into several areas which will be simultaneously processed with the same operations. This will lead to several independent subtrees. Let us note that depending on image content (some areas might be more complex than others), these trees may have different depth and require various computational costs to be built. There is thus a need for synchronization, followed by a merging procedure which occurs on both spatial dimensions. During this step, each pixel is compared to his previous neighbors in the horizontal and vertical axes, to create a new node which is inserted in the tree according to the local dissimilarity value.

We have observed that most of the computation time is spent during the update of node children. In particular, gathering flat zones leads to many merging operations. We thus have adopted a data structure which allows for a late merging of these zones. To do so, we first scan the image to build the α-tree. Then we require a second pass on the image to reduce the tree by performing late fusions and simplifying useless nodes. The overall complexity of this algorithm is then $O(2N\log(A))$ with N the number of pixels and $A = 256$ the tree height.

IV. Experiments

In order to evaluate the original segmentation method proposed in the previous section, we have conducted a set of experiments, which will be presented here. But first, we discuss how to set the different parameters involved in our algorithm.

Figure 5. Illustration of automatic image segmentation

A. Parameter settings

As indicated previously, our segmentation method relies on several parameters to be set. In the presented experiments, parameter settings has been achieved empirically, with a single set of parameters for all images to be segmented. To ensure reproducibility, we provide here the values used in our experiments. $T_{\text{saturation}} = 0.02$, $T_{\text{angle}} = 0.07$, $T_{\text{outlier}} = 0.14$ are used to describe the average hue of each α-zone. $[T_{\text{min}}, T_{\text{max}}] = [0.02, 51]$ denotes the interval of significant α-zones in terms of relative area w.r.t the whole image (i.e. the other nodes in the tree will not be selected). If an image comes in greyscale, or some image parts have a low saturation, the parameters involved are $T'_{\text{outlier}} = 0.02$ and $T'_{\text{outlier}} = 0.3$.

Fig. 5 illustrates the behavior of our automatic segmentation method. Each region is represented by the average hue of its highest (in the tree hierarchy) node. Grey and black colors are respectively used for low contrast and unsegmented areas.

B. Automatic image segmentation

Let us recall that our algorithm does not lead to a complete segmentation or partition of the image. Based on the α-tree features, it kept only relevant nodes and branches, leaving some parts of the image out of the resulting partition. Of course, a full partition might be obtained by adding all leaves of removed branches to the set of selected branches.

To achieve a fair evaluation w.r.t. the state-of-the-art, we have experimented our algorithm on the Berkeley Segmentation Dataset [7]. This dataset offers a wide range of natural images, along with some reference segmentation maps produced with manual user segmentation. We have measured the performance of our algorithm based on several criteria taken from [9] and [10]. More precisely, we have obtained an overall oversegmentation ratio (OV) equal to 23.66 and an overall maximal precision (PM) equal to 60%. While these preliminary results are still far from being

Figure 6. Illustration of video matting

perfect, they are very promising since we have not optimized the parameter settings yet. Let us precise that only 70% of the image content is analyzed (the insignificant nodes are removed). We can thus modify the PM measure to deal with such an incomplete segmentation, leading then to a new PM value of 87%. Of course, manual tuning of the parameters may lead to better results on some images, e.g. up to PM equal to 97% in our experiments.

C. Interactive video matting

Besides automatic image segmentation, the proposed scheme can also be used in an interactive framework. In this case, the user is expected to pick one or several pixels from which relevant branches from the α-tree are extracted. This processing is also known as image matting in the computer vision and graphics community and provides, for a given pixel or area, the main object-of-interest it belongs to (together with his edges). Since our algorithm comes with a very small computation time, it has been experimented in the context of matting of still images but also of video sequences. For the sake of conciseness, only experiments on video matting are presented here.

We show in Fig. 6 the matting of 5 objects-of-interest marked by the user on the middle frame. Only a sample of the video frames are shown. We can observe that objects are accurately extracted. This process is achieved in less than a few milliseconds once the tree has been fully built. Indeed, the online matting has a complexity equals to $O(\log(A))$, with here $A = 256$.

V. Conclusion

In this paper, we have considered the α-tree image model in the context of image segmentation. From an in-depth study of the tree properties, we have introduced some relevant features to drive the segmentation process. The subsequent image segmentation algorithm comes with a low complexity thanks to an efficient implementation scheme. Moreover, this make possible to use this algorithm both in an interactive way and with video data. Preliminary results obtained with the Berkeley Segmentation Dataset are promising and appeal for further studies.

More precisely, the tree structure has to be further explored to understand how it can provide more image knowledge. Since it shares some similarities with the concept of MSER (Maximally Stable Extremal Region) [11], a comparative study is worth being achieved. Besides, various information (object size, edge sharpness) might be extracted to distinguish between object-of-interest and other objects in an automatic fashion. We are also replacing the manual and empirical step for parameter settings by a machine learning strategy, relying on a genetic algorithm. For experimental setups such as the Berkely Segmentation Dataset, the availability of some reference segmentation maps has indeed to be exploited. Finally, following this first experiment, we would like to consider other tree models (i.e. hyperconnected trees [12]) in such a segmentation paradigm.

References

[1] G. K. Ouzounis and P. Soille, "Pattern spectra from partition pyramids and hierarchies," in *International Symposium on Mathematical Morphology*, Verbania-Intra, Italy, 2011, pp. 108–119.

[2] J. Serra, "Anamorphoses and function lattices," in *Mathematical Morphology in Image Processing"*, E. R. Dougherty, Ed. New York: Marcel Dekker, 1993, ch. 13, pp. 483–523.

[3] P. Soille, "Constrained connectivity for hierarchical image partitioning and simplification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 7, pp. 1132–1145, July 2008.

[4] P. Soille and J. Grazzini, "Constrained connectivity and transition regions," in *International Symposium on Mathematical Morphology*, Groningen, The Netherlands, August 2009, pp. 59–69.

[5] J. Weber, S. Lefèvre, and P. Gançarski, "Interactive video segmentation based on quasi-flat zones," in *International Symposium on Image and Signal Processing and Analysis (ISPA)*, Dubrovnik, Croatia, September 2011, pp. 265–270.

[6] E. Aptoula and S. Lefèvre, "A comparative study on multivariate mathematical morphology," *Pattern Recognition*, vol. 40, no. 11, pp. 2914–2929, November 2007.

[7] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proceedings of the 8th International Conference on Computer Vision*, vol. 2, Vancouver, Canada, July 2001, pp. 416–425.

[8] E. Aptoula and S. Lefèvre, "On the morphological processing of hue," *Image and Vision Computing*, vol. 27, no. 9, pp. 1394–1401, August 2009.

[9] A. Carleer, O. Debeir, and E. Wolff, "Assessment of very high spatial resolution satellite image segmentations," *Photogrammetric Engineering & Remote Sensing*, vol. 71, no. 11, pp. 1285–1294, November 2005.

[10] S. Derivaux, G. Forestier, C. Wemmert, and S. Lefèvre, "Supervised image segmentation using watershed transform, fuzzy classification and evolutionary computation," *Pattern Recognition Letters*, vol. 31, no. 15, pp. 2364–2374, November 2010.

[11] J. Matas, O. Chum, U. Martin, and T. Pajdla, “Robust wide baseline stereo from maximally stable extremal regions,” in *British Machine Vision Conference*, 2002, pp. 384–393.

[12] B. Perret, S. Lefèvre, C. Collet, and E. Slezak, “Hyperconnections and hierarchical representations for grayscale and multiband image processing,” *IEEE Transactions on Image Processing*, vol. 21, no. 1, pp. 14–27, January 2012.