Philippe Salembier and Michael H.F. Wilkinson

Connected Operators

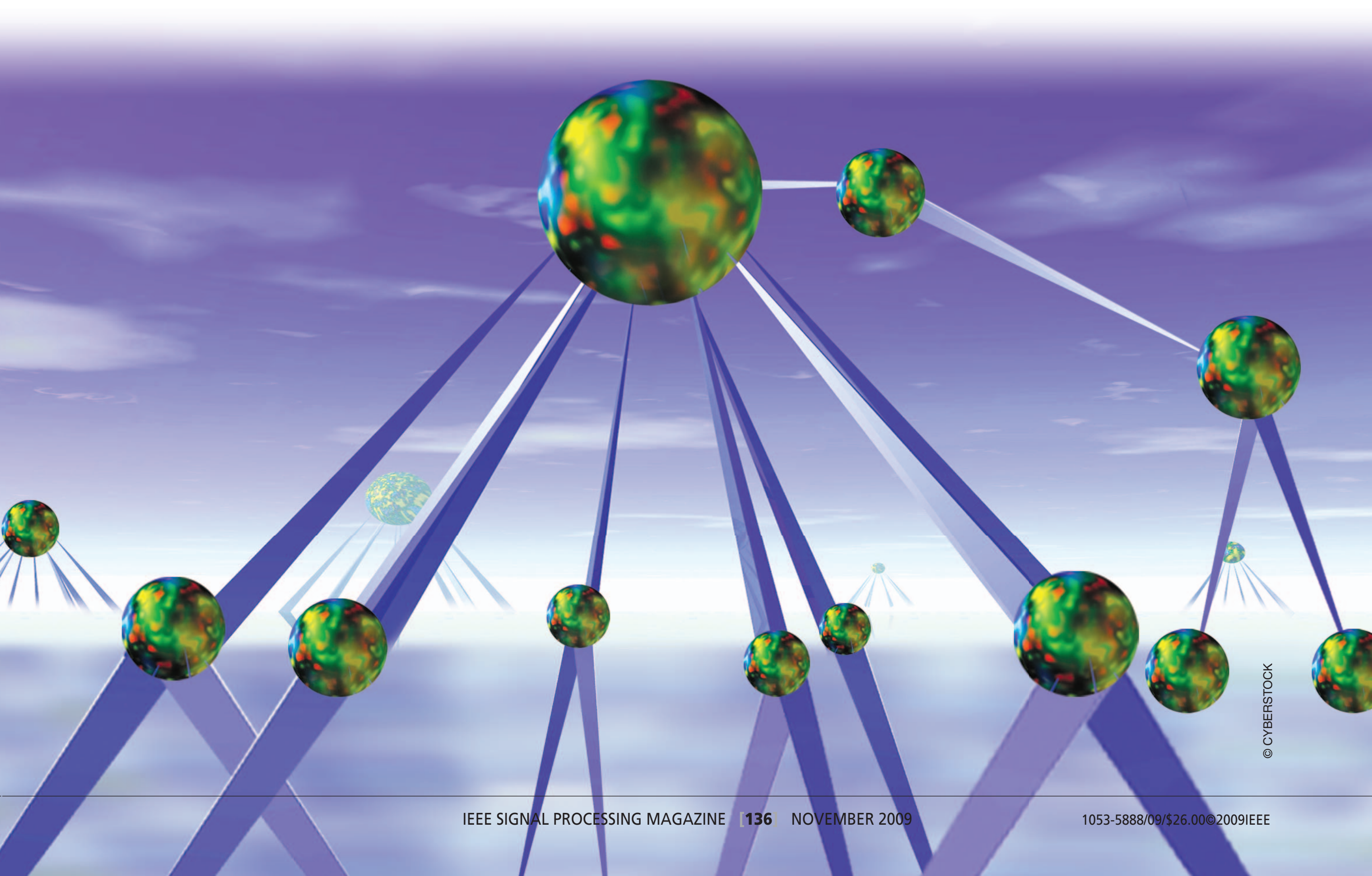
A review of region-based morphological image processing techniques

Connected operators are filtering tools that act by merging elementary regions called flat zones. Connecting operators cannot create new contours nor modify their position. Therefore, they have very good contour-preservation properties and are capable of both low-level filtering and higher-level object recognition. This article gives an overview on connected operators and their application to image and video filtering. There are two popular techniques used to create connected operators. The first one relies on a reconstruction process. The operator involves first a simplification step based on a "classical" filter and then a reconstruction process. In fact, the reconstruction can be seen as a way to create a connected version of an arbitrary operator. The simplification effect is defined and limited by the first step. The examples we show include simplification in terms of size or contrast.

The second strategy to define connected operators relies on a hierarchical region-based representation of the input image, i.e., a tree, computed in an initial step. Then, the simplification is obtained by pruning the tree, and, third, the output image is constructed from the pruned tree. This article presents the most important trees that have been used to create connected operators and also discusses important families of simplification or pruning criteria.

We also give a brief overview on efficient implementations of the reconstruction process and of tree construction. Finally, the

Digital Object Identifier 10.1109/MSP.2009.934154



possibility to define and to use nonclassical notions of connectivity is discussed and illustrated.

INTRODUCTION

An increasing number of image or video processing applications are not efficiently dealt with using classical pixel-based filtering and processing approaches. Examples are multimedia applications such as content-based compression or indexing as well as many biomedical or remote sensing applications.

For content-based image or video compression, the representation based on an array of pixels is not appropriate if one wants to be able to act on objects, to selectively encode areas of interest, or to assign different behaviors to the entities represented in the image. In these applications, the notion of object is essential. As a consequence, the data modeling or at least the processing has to include the notion of regions to represent objects. Content-based indexing also faces the same kind of challenges. For instance, the video representation based on a flow of frames is inadequate for many video indexing applications. One would like to have access to a table of contents of the video where the notion of temporal regions is central. In a large number of biomedical as well as remote sensing applications, low-level processing of the data would benefit from a region-based processing as most of the processing is directed toward the identification or classification of regions into meaningful entities with respect to the application (detection of organs, cells, and specific types of vegetation).

In all of these examples, the notion of region turns out to be central in the processing. Note that regions may be two-dimensional (2-D) or three-dimensional (3-D) spatial connected components but also temporal or spatio-temporal connected components in the case of video. It must be recognized that most low-level image processing tools cannot handle the notion of region. The vast majority of low-level processing tools such as filters are very closely related to the classical pixel-based representation of signals. Examples include linear convolution with an impulse response, median or rank-order filter, morphological operators based on erosion, and dilation with a structuring element. In all cases, the processing strategy consists in modifying the values of individual pixels by a function of the pixels values in a local window.

Early examples of region-based processing can be found in the literature in the field of segmentation. For example, the classical split and merge algorithm first defines a set of elementary regions (the split process) and then interacts directly on these regions allowing them to merge under certain conditions.

Recently, connected operators from morphology have received much attention. Connected operators are region-based filtering tools because they act directly on the connected components where the image is constant, the so-called flat zones. Intuitively, connected operators can remove boundaries between flat zones but cannot add new boundaries nor shift existing ones. The related literature rapidly grows and involves theoretical studies [1]–[11], algorithm developments [12]–[20], and applications [14], [21]–[28]. The goals of this article are 1) to provide an introduction to connected operators for gray-level images and video sequences and 2) to discuss the techniques and algorithms that have been most successful within the framework of practical applications.

[CONNECTED OPERATORS ARE FILTERING TOOLS THAT ACT BY MERGING ELEMENTARY REGIONS CALLED FLAT ZONES.]

CLASSICAL FILTERING APPROACHES

In this section, we define the notation and the basic notions in complete lattice theory [29], [30] that are going to be useful in this article. Although the notion of connected operators can be defined for continuous images, we focus on the discrete case that is of interest in practice. Let $f[n]$ denote images and $f_t[n]$ video sequences where n denotes the pixel or space coordinate (a vector in the case of 2-D or 3-D images) and t the time instant in the case of a video sequence. In lattice theory, the notions of order relationship, supremum $\bigvee$ and infimum $\bigwedge$ are central. Let us first recall the definition of order between images. A gray-level image f is said to be smaller than a gray-level image g if and only if

$$f \le g \Leftrightarrow \forall n, f[n] \le g[n]. \tag{1}$$

If the lattice is complete, any set (finite or infinite) of images has a supremum and an infimum. The supremum is the smallest upper bound and the infimum is the highest lower bound.

Many lattice theory properties are concerned with the interaction of an operator with the order relationship. In particular, an operator ψ acting on an input f is said to be

- increasing if the order that may be present between two input images is preserved by the filtering, $\forall f, g, f \le g \Rightarrow \psi(f) \le \psi(g)$
- idempotent if the iteration of the operator always gives the same result as applying the filter once (for example, ideal linear filters are idempotent but are not usable in practice since they are unstable). Equivalently, an operator is idempotent if the filtering of an arbitrary signal produces an invariant, $\forall f, \psi(\psi(f)) = \psi(f)$
- extensive if the input image is always smaller than the output image $\forall f, f \le \psi(f)$
- antiextensive if the output image is always smaller than the input image $\forall f, \psi(f) \le f$.

Based on these elementary properties, important classes of morphological filters are defined as follows:

- A morphological filter is an increasing and idempotent operator.
- An opening is an antiextensive morphological filter.
- A closing is an extensive morphological filter.

Note that sometimes the notion of a morphological filter is relaxed [3] to mean any idempotent operator, as in the case of shape filters [27]. Finally, an operator is said to be self-dual if it processes symmetrically bright and dark image components, $\forall f, \psi(f) = -\psi(-f)$.

Figure 1 shows a number of classical approaches to image filtering, i.e., linear filtering, median filtering, and opening and closing using a 5×5 structuring element. The opening is given by $\gamma_h(f) = \delta_h(\epsilon_h(f))$ and the closing by $\varphi_h(f) = \epsilon_h(\delta_h(f))$. Both are based on morphological dilation and erosion. The dilation by a structuring element $h[n]$ is defined by $\delta_h(f)[n] = \bigvee_{k=-\infty}^{\infty}(h[k] + f[n-k])$. The erosion is given by $\epsilon_h(f)[n] = \bigwedge_{k=-\infty}^{\infty} f[n-k] - h[-k]$. The opening and closing are morphological filters (i.e., both increasing and idempotent). Moreover, the opening is antiextensive (it removes bright components) whereas the closing is extensive (it removes dark components).

For all of these classical approaches, the filter design consists in carefully choosing a specific signal $h[n]$ that may be the impulse response, the window or the structuring element. While this definition step is classically seen as the key point of the filter design, our point here is to highlight that, for image processing, the use of $h[n]$ has a major drawback: as $h[n]$ is based on shapes and structures that are not related to the input signal, its use inevitably introduces distortions in the output signal. The distortion effect depends on the specific filter, but for a large range of applications requiring high precision on contours, none of these filtering strategies is acceptable.

To reduce the distortion, one possible solution is to adapt $h[n]$ to the local structures of the input signal. This solution may improve the results but still remains unacceptable in many circumstances. An attractive solution to this problem is provided by connected operators. Most connected operators used in practice rely on a completely different filtering strategy: the filtering is done without using any specific signal such as an impulse response, a window, or a structuring element. In fact, the structures of the input signal itself are used to act on the signal. As a result, no new structures or distortions coming from unrelated signals are introduced in the output.

QUITE UNIQUELY, CONNECTED FILTERS CAN EASILY BE GIVEN MANY DESIRABLE INVARIANCE PROPERTIES, SUCH AS SCALE INVARIANCE.

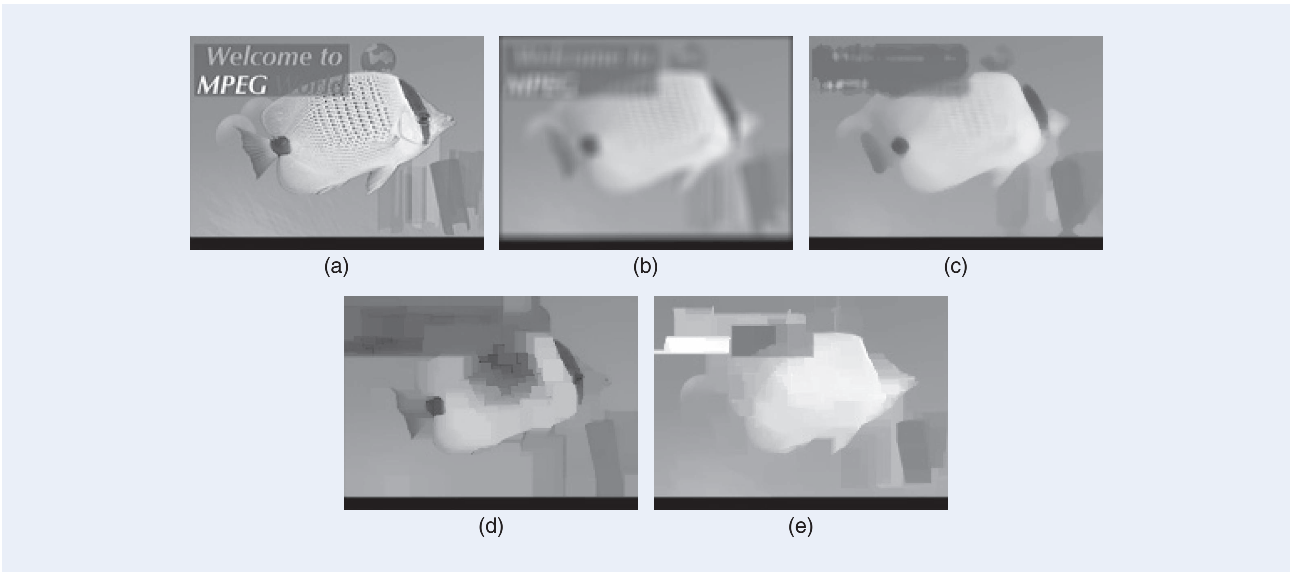
CONNECTED OPERATORS

DEFINITION AND BASIC PROPERTIES

Gray-level connected operators act by merging flat zones. The flat zones are the connected components where the image is constant. Note that in natural images, flat zones often involve only one or two pixels. Connected operators cannot create new contours, and, as a result, they cannot introduce in the output image a structure that is not present in the input image. Furthermore, they cannot modify the position of existing boundaries between regions and, therefore, have very good contour preservation properties.

Gray-level connected operators originally defined in [1] rely on the notion of partition of flat zones. A partition is a set of nonoverlapping, nonvoid regions that fills the entire space. Until the section "Nonclassical Connectivities," we assume that the connectivity is defined on the digital grid by a simple rule that defines the immediate neighbors of a pixel. Typical examples are the four- and eight-connectivity for 2-D images and six- and 26-connectivity for 3-D images. Let us denote a partition by $\mathcal{P}$ and the region of $\mathcal{P}$ that contains pixel n by $\mathcal{P}$(n). A partial order relationship among partitions can be created: $\mathcal{P}_1$ "is finer than" $\mathcal{P}_2$ (written as $\mathcal{P}_1 \sqsubseteq \mathcal{P}_2$), if $\forall n$, $\mathcal{P}_1(n) \subseteq \mathcal{P}_2(n)$.

It can be shown that the set of flat zones of an image f is a partition of the space, $\mathcal{P}_f$. Based on these notions, connected operators are defined next.



[FIG1] Example of filtering with classical filters: (a) original image, (b) low-pass filter (7×7 average), (c) median (window size: 5×5), (d) opening (structuring element size: 5×5), and (e) closing (structuring element size: 5×5).

DEFINITION 1

Connected Operators
A gray-level operator ψ is connected if the partition of flat zones of its input f is always finer than the partition of flat zones of its output, that is

$$\mathcal{P}_f \sqsubseteq \mathcal{P}_{\psi(f)}, \forall f.$$

This definition clearly highlights the region-based processing of the operator: indeed, regions of the output partition are created by union of regions of the input partition. New connected operators can be derived from the combination of primitive connected operators. In particular, if ψ_1, ψ_2 are two connected operators, their composition $\psi_2\psi_1$ is also connected. If $\{\psi_i\}$ represents a family of connected operators, their supremum $\bigvee_i \psi_i$ and infimum $\bigwedge_i \psi_i$ are connected. Finally, if ψ is a connected operator, its dual ψ^* defined by: $\psi^*(f) = -\psi(-f)$, is also connected.

As can be seen, connected operators heavily rely on the notions of flat zone partition and connectivity. In the section "Nonclassical Connectivities," we will show that the notion of flat zones can be defined with some flexibility allowing flat zones to present some gray-level fluctuations. We will illustrate the practical interest of these extensions.

Finally, connected operators are filtering tools in the sense that they transform an input gray-level image into a filtered gray-level image. However, as they are conceptually based on the notion of partition, they are often claimed to bridge the gap between classical filtering and segmentation [31], [32]. Moreover, some of the theoretical notions involved in connected operators have been recently extended for pure segmentation applications. The resulting approach is known as connective segmentation. The reader is referred to [33] and [34] for more information on this subject.

EARLY EXAMPLE OF CONNECTED OPERATORS

The first known connected operator was defined for binary images. It is known as the binary opening by reconstruction [35]. This operator eliminates the connected components that would be totally removed by an erosion with a given structuring element and leaves the other components unchanged. This filtering approach offers the advantage of simplifying the image (some components are removed) as well as preserving the contour information (the components that are not removed are perfectly preserved). It can be shown that the process is increasing, idempotent, and antiextensive, that is, an opening. Moreover, it was called "by reconstruction" because of the algorithm used for its implementation. From the algorithmic viewpoint, if X is the original binary image, the first step is to compute an opening with a structuring element B_k of size k, $\delta_{B_k}(\varepsilon_{B_k}(X))$ (or an erosion $\varepsilon_{B_k}(X)$ if B_k is connected and contains the origin). This opening is used to "mark" the connected components that should be preserved. The final result is obtained by progressively dilating the opening conditionally to the mask defined by the original image (see Figure 2)

1) $Y_0 = \delta_{B_k}(\varepsilon_{B_k}(X))$

2) $Y_k = \delta_C(Y_{k-1}) \cap X$, where C is a binary structuring element defining the connectivity, e.g., square of 3×3 (cross) for the eight-connectivity (four-connectivity)

3) Iterate Step 2 until idempotence (until no change is observed between two iterations).

The first gray-level connected operator was obtained by a transposition of the previous approach to the lattice of gray-level functions [13], [30]. It is known as an opening by reconstruction

1) $g_0 = \delta_{h_k}(\varepsilon_{h_k}(f))$, where f is the input and h_k a structuring element of size k

2) $g_k = \delta_C(g_{k-1}) \bigwedge f$, where C is a flat structuring element (a structuring element is flat if the values of $h[k]$ are 0 on its support S and $-\infty$ outside S. In this case, the dilation simply consists in computing the supremum of the samples corresponding to k such that $h[h] = 0$:$\delta_{h_{\text{flat}}}(f)[n] = \bigvee_{k \in S} f[n-k]$) defining the connectivity, e.g., square or cross

3) iterate Step 2 until idempotence.

It was shown in [1] that this operator is connected. Intuitively, the erosion acts as a simplification step by removing small, bright components. The reconstruction process restores the contours of the components that have not been completely removed by the erosion.

There are several ways to construct connected operators, and many new operators have been recently introduced. From the practical viewpoint, the most successful strategies rely on a reconstruction process or on region-tree pruning. Operators resulting from these two strategies are discussed next.

CONNECTED OPERATORS BASED ON RECONSTRUCTION

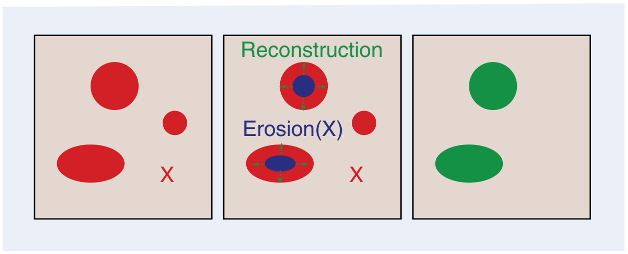
ANTIEXTENSIVE RECONSTRUCTION

Let us start by discussing openings that are filters that can simplify images by removing some of their maxima. The most classical way to construct connected openings is to use an antiextensive reconstruction process.

DEFINITION 2

Antiextensive Reconstruction
If f and g are two images (called the "reference" and the "marker" image, respectively), the antiextensive reconstruction $\rho^{\downarrow}(g|f)$ of g under f is given by

$$g_k = \delta_C(g_{k-1}) \bigwedge f \text{ and}$$
$$\rho^{\downarrow}(g|f) = \lim_{k\to\infty} g_k, \tag{2}$$



[FIG2] Example of reconstruction: (a) original image X, (b) erosion of X and conditional dilation, and (c) final reconstructed image.

where $g_0 = g \leq f$ and δ_C is the dilation with the flat structuring element defining the connectivity (3×3 square or cross for 2-D images and $3 \times 3 \times 3$ or a 3-D cross for 3-D images).

It can be shown that the series, g_k, always converges and the limit always exists. The operator $\delta_C(\cdot) \bigwedge f$ is sometimes called a conditional dilation as the elementary dilation is conditioned to remain below f. In this section, we will rely on this definition based on iterative conditional dilations. Later, we will see that efficient implementations of the reconstruction process do exist and do not rely on iterations.

Of course by duality, an extensive reconstruction may be defined in Definition 3.

DEFINITION 3

Extensive Reconstruction

If f and g are two images (called the "reference" and the "marker" image, respectively), the extensive reconstruction $\rho^{\uparrow}(g|f)$ of g above f is given by

$$g_k = \varepsilon_C(g_{k-1}) \bigvee f \text{ and}$$
$$\rho^{\uparrow}(g|f) = \lim_{k\to\infty} g_k, \tag{3}$$

where $g_0 = g \geq f$ and ε_C is the erosion with the flat structuring element defining the connectivity.

In practice, useful connected operators are obtained by considering that the marker image g is a transformation $\tau(f)$ of the input image f that also plays the role of the marker image. As a result, most connected operators ψ obtained by reconstruction can be written as

$$\psi(f) = \rho^{\downarrow}(\tau(f)|f) \quad \text{(antiextensive operator), or}$$
$$\psi(f) = \rho^{\uparrow}(\tau(f)|f) \quad \text{(extensive operator).} \tag{4}$$

A few examples are discussed next.

SIZE FILTERING

The simplest size-oriented connected operator is obtained by using as marker image, $\tau(f)$, the result of an erosion with a structuring element h_k of size k. It is the opening by reconstruction of erosion

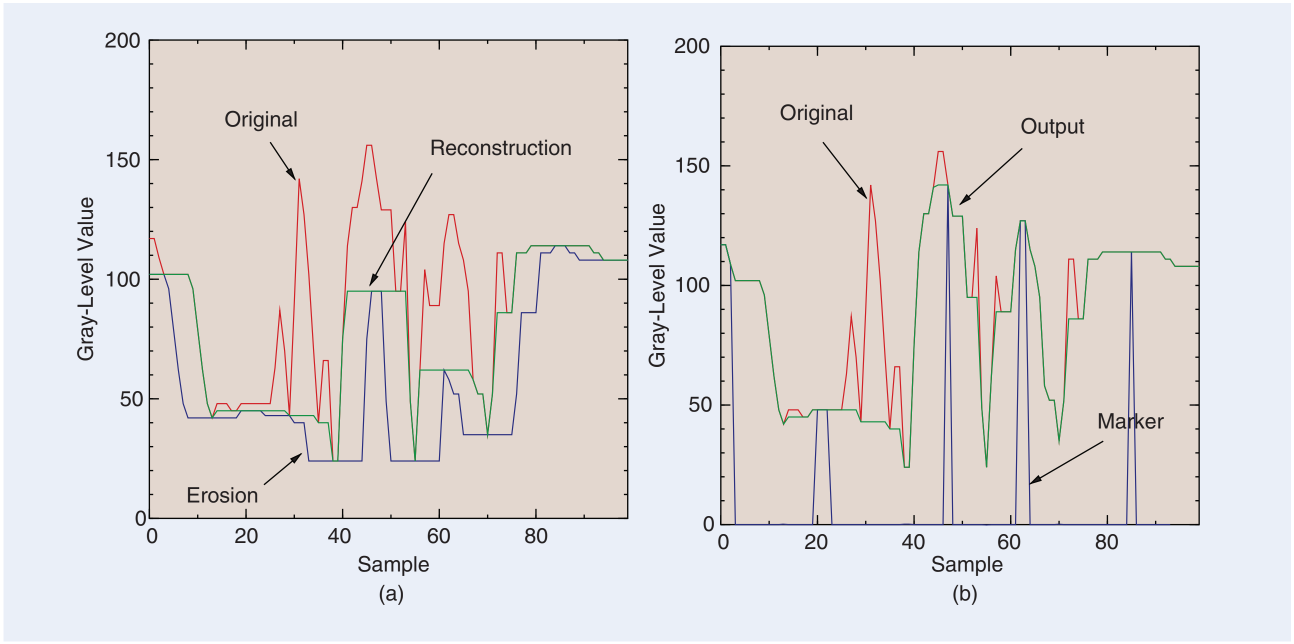
$$\psi(f) = \rho^{\downarrow}(\varepsilon_{h_k}(f)|f). \tag{5}$$

Note that it can be demonstrated that the same operator is obtained by changing the erosion, ε_{h_k}, by an opening, γ_{h_k} with the same structuring element, h_k, provided h_k is connected and contains the origin. It can be demonstrated that this operator is an opening. By duality, the closing by reconstruction is given by

$$\psi^*(f) = \rho^{\uparrow}(\delta_{h_k}(f)|f). \tag{6}$$

An example of opening by reconstruction of erosion is shown in Figure 3(a). In this example, the original signal f has 11 maxima. The marker signal g is created by an erosion with a flat structuring element which eliminates the narrowest maxima. Only five maxima are preserved after erosion. Finally, the marker is reconstructed. In the reconstruction, only the five maxima that were present after erosion are visible and narrow maxima have been eliminated. Moreover, the transitions of the reconstructed signal correspond precisely to the transitions of the original signal.

As can be seen, the simplification effect, that is the elimination of narrow maxima is almost perfectly done. However, the preservation effect may be criticized: although the maxima contours are well preserved, their shape and height



[FIG3] **Size-oriented connected operators: (a) opening by reconstruction and (b) new marker indicating where the reconstruction has been inactive and second reconstruction.**

are distorted. To reduce this distortion, a new connected operator can be built on top of the first one. Let us construct a new marker image, $m[n]$, indicating the maxima where the reconstruction has been inactive, i.e., where the final result is equal to the erosion

WHEN USING PATH CONNECTIVITY, WE MIGHT NOT OBTAIN SETS OF PIXELS THAT CORRESPOND CLOSELY TO PERCEPTUAL GROUPS RELEVANT TO HUMAN OBSERVERS.

$$m[n] = \begin{cases} f[n], & \text{if } \rho^{\downarrow}(\varepsilon_{h_k}(f)|f)[n] = \varepsilon_{h_k}(f)[n], \\ & \text{and } \varepsilon_{h_k}(f)[n] \text{ belongs to a maxima} \\ 0 & \text{otherwise.} \end{cases} \quad (7)$$

This marker image is illustrated in Figure 3(b). As can be seen, it is equal to zero except for the five maxima that are present after erosion. At that locations, the gray-level values of the original image, $f[n]$, are assigned to the marker image. Finally, the second connected operator is created by the reconstruction of the marker, m under f

$$\psi(f) = \rho^{\downarrow}(m|f). \quad (8)$$

This operator is also an opening by reconstruction. The final result is shown in Figure 3(b). The five maxima are better preserved than with the first opening by reconstruction whereas the remaining maxima are perfectly removed. The difference between both reconstructions is clearly visible when the initial erosion uses a rather large structuring element as in the example of Figure 4. The first opening by reconstruction removes small bright details of the image: the text in the upper left corner. The fish is a large element and is not removed. It is indeed visible after the first opening by reconstruction [Figure 4(b)], but its gray-level values are not well preserved. This drawback is avoided by using the second reconstruction. Finally, let us mention that by duality closings by reconstruction can be defined. They have the same effect than the openings but on dark components (minima).

CONTRAST FILTERING

The previous section considered size simplification. Contrast simplification can be obtained by substituting the erosion in (5) by a subtraction by a constant, c, from the original image f

$$\phi(f) = \rho^{\downarrow}(f - c|f). \quad (9)$$

This operator, known as the hmax operator, is connected, increasing, and antiextensive but not idempotent. Its effect is illustrated in Figure 5(a). As can be seen, the maxima of small contrast are removed and the contours of the maxima of high contrast are well preserved. However, the height of the remaining maxima are not well preserved. As in the previous section, this drawback can be removed if a second reconstruction process is used. This second reconstruction process is exactly the same as the previous one defined by (7) ($m[n] = f[n]$ if $\rho^{\downarrow}(f - c|f)[n] = f[n] - c$ and belongs to a maxima). This second connected operator is called a dynamic opening [36].

The operators effect is illustrated in Figure 6. Both operators remove maxima of contrast c lower than 100 gray-level values. However, the hmax operator produces an output image of low contrast, even for the preserved maxima. By contrast, the dynamic opening successfully restores the retained maxima.

SELF-DUAL RECONSTRUCTION AND LEVELING

The connected operators discussed in the previous section were either antiextensive or extensive. They allow the simplification of either bright or dark image components. For some applications, this behavior is a drawback and one would like to simplify in a symmetrical way, all components. From the theoretical viewpoint, this means that the filter has to be self-dual, that is $\psi(f) = -\psi(-f)$.

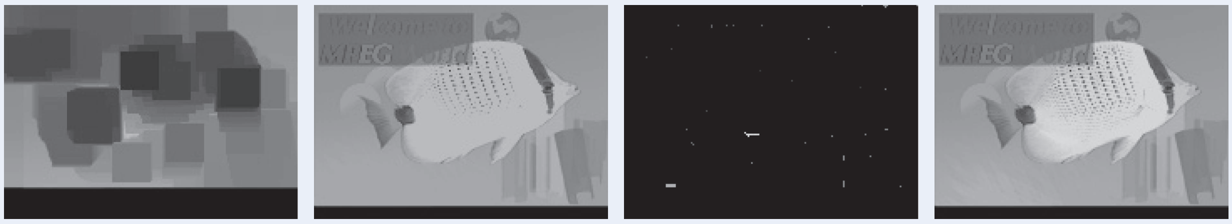
With the aim of constructing self-dual connected operators, the concept of levelings was proposed in [5] by adding some restrictions in the definition of connected operators.

DEFINITION 4

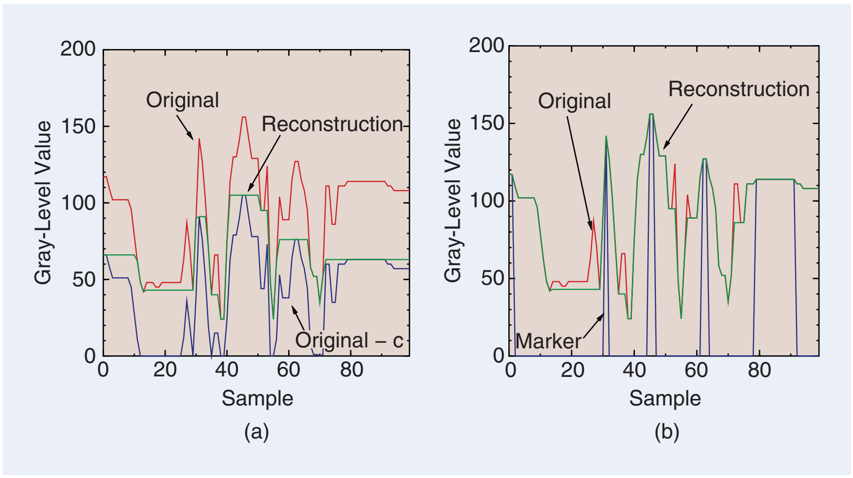
Leveling

The operator ψ is a leveling if $\forall n, n'$ neighboring pixels, $\psi(f)[n] > \psi(f)[n'] \Rightarrow f[n] \geq \psi(f)[n]$ and $\psi(f)[n'] \geq f[n']$.

This definition not only states that if a transition exists in the output image, it was already present in the original image



[FIG4] Size filtering with opening by reconstruction: (a) erosion of the original image of Figure 1(a) by a flat structuring element of size 30×30, (b) reconstruction of the erosion, (c) marker indicating maxima where the first reconstruction has not been active (7), and (d) second reconstruction.



[FIG5] **Contrast-oriented connected operators: (a) reconstruction of $f-c$ and (b) second reconstruction: dynamic opening.**

The most popular technique to create levelings relies on the self-dual reconstruction process described next.

DEFINITION 5

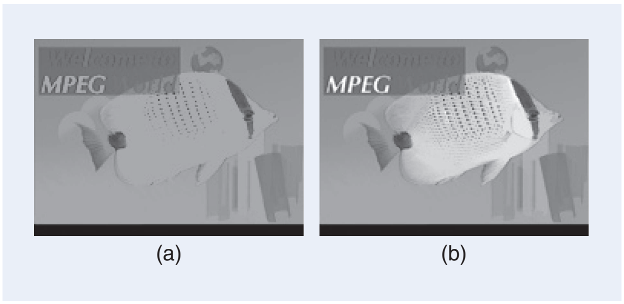
Self-Dual Reconstruction

If f and g are two images (respectively called the "reference" and the "marker" image), the self-dual reconstruction $\rho^{\updownarrow}(g|f)$ of g with respect to f is given by

$$\begin{aligned} g_k &= \varepsilon_C(g_{k-1}) \bigvee [\delta_C(g_{k-1}) \bigwedge f] \\ &= \delta_C(g_{k-1}) \bigwedge [\varepsilon_C(g_{k-1}) \bigvee f] \\ &\quad \text{(equivalent expression) and} \\ \rho^{\updownarrow}(g|f) &= \lim_{k\to\infty} g_k, \end{aligned} \tag{10}$$

(This is equivalent to say that the operator is connected) but also that 1) the sense of gray-level variation between n and n' has to be preserved and 2) the variation interval between $\psi(f)[n]$ and $\psi(f)[n']$ must contain the variation interval between $f[n]$ and $f[n']$.

The theoretical properties of levelings are studied in [5] and [6]. In particular, it has been shown that any opening or closing by reconstruction is a leveling. If ψ_1, ψ_2 are levelings, their composition $\psi_2\psi_1$ is also a leveling. Finally, if $\{\psi_i\}$ are levelings, their supremum $\bigvee_i \psi_i$, and infimum $\bigwedge_i \psi_i$, are levelings.



[FIG6] **Contrast filtering: (a) hmax operator and (b) dynamic opening.**

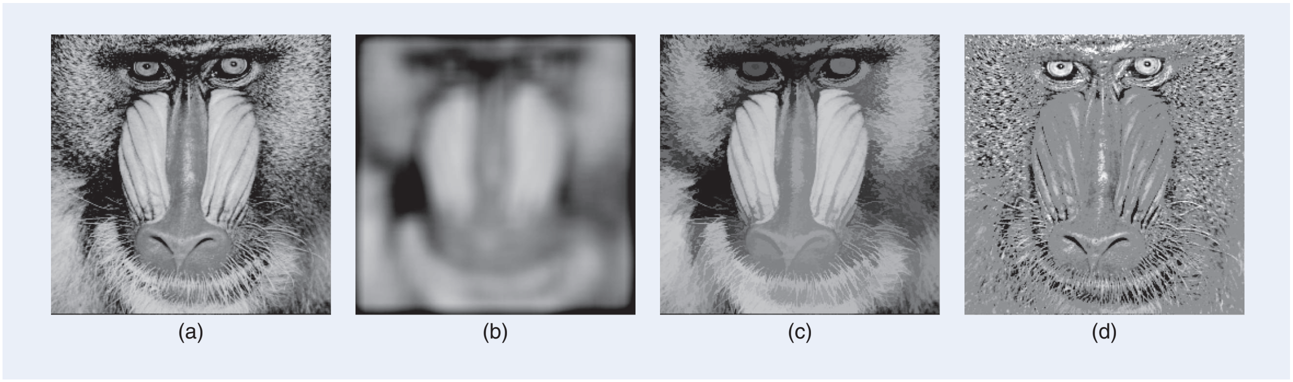
where $g_0 = g$ and δ_C and ε_C are respectively the dilation and the erosion with the flat structuring element defining the connectivity (3×3 square or cross).

In fact, the self-dual reconstruction is the antiextensive reconstruction of (2) for the pixels where $g[n] < f[n]$ and the extensive reconstruction of (3) for the pixels where $f[n] < g[n]$. In practice, the self-dual reconstruction is used to restore the contour information after an initial filtering process. In other words, the reconstruction allows the creation of a connected version $\rho^{\updownarrow}(\psi(f)|f)$ of any filter $\psi(f)$.

A typical example of initial filter $\psi(f)$ is an alternating sequential filter

$$\psi(f) = \varphi_{h_k}\gamma_{h_k}\varphi_{h_{k-1}}\gamma_{h_{k-1}} \ldots \varphi_{h_1}\gamma_{h_1}(f), \tag{11}$$

where φ_{h_k} and γ_{h_k} are respectively a closing and an opening with a structuring element h_k. In [26], the initial filter is a linear low-pass filter based on the convolution with a Gaussian impulse response. As can be seen in Figure 7, the low-pass filter removes most of the texture of the original image. The leveling provides then the structural part of the image, that is, the image content, with a precise definition of



[FIG7] **Example of image decomposition in structural and texture parts with leveling: (a) original image, (b) marker: low-pass filtering with Gaussian filter, (c) leveling: structural part, and (d) residue: texture part.**

the contours but without the texture. Finally, the residue $f - \rho^{\updownarrow}(\psi(f)|f)$ gives the texture part. This strategy can be viewed as a morphological approach to the classical problem of structure/texture image decomposition.

Variational Formulation of Reconstruction and Leveling

The definitions of the reconstruction process and the leveling done above rely on the notion of conditional erosion and dilation. However, erosion and dilation can be formulated as the result of simple variational problems with the following partial differential equations (pdes) defining the flows [37], [38]

$$\text{Dilation flow: } \frac{\partial g}{\partial t} = \|\nabla g\| \text{ and}$$
$$\text{Erosion flow: } \frac{\partial g}{\partial t} = -\|\nabla g\|, \tag{12}$$

where ∇ represents the classical gradient operator. As the leveling is a conditional dilation (erosion) of g at locations where $g < f$ $(g > f)$, it can be shown [39] that the leveling of the reference f with marker g can be obtained as a steady state of the following pde:

$$\frac{\partial g}{\partial t} = \operatorname{sign}(f - g)\|\nabla g\|. \tag{13}$$

This alternative formulation of the leveling not only provides an alternative implementation approach but also new insights on the interpretation of leveling and its scale-space properties (see [39] for more details).

EFFICIENT IMPLEMENTATIONS OF RECONSTRUCTION PROCESSES

Equations (2) and (3) define the reconstruction processes but do not provide efficient implementations. Indeed, the number of iterations is generally fairly high. The most efficient reconstruction algorithms rely on the definition of a clever scanning of the image and are implemented by hierarchical first-in, first-out (FIFO), or priority queues. A review of the most popular reconstruction algorithms can be found in [13]. Here, we describe a simple but efficient one: the basic idea of the algorithm is to start from the regional maxima of the marker image g and to propagate them under the original image f. In the case of (2), the algorithm works in the following two steps:

1) The initialization consists in putting in the queue the location of pixels that are on the boundary of the regional maxima of the marker image. Regional maxima are the set of connected components where the image has a constant gray-level value and such that every pixel in the neighborhood of the regional maxima has strictly a lower value. Algorithms to compute regional maxima can be found in [40].

2) The propagation extracts the first pixel, with the highest gray level, n, from the queue (note that n is a pixel of the marker image g). Then, it assigns to each of its neighbors, n', that have a strictly lower gray-level value than $g[n]$ (that is, if $g[n'] < g[n]$), the minimum between the gray-level value of n and the gray-level value of the pixel of the original image at the same location than n', that is $g[n'] = g[n] \bigwedge f[n']$. Finally, the pixel n' is introduced in the queue, using gray level as priority or hierarchy level. This propagation process has to be carried on until the queue is empty. This is efficient since pixels are processed only once.

CONNECTED OPERATORS BASED ON REGION TREE PRUNING

TREE REPRESENTATION OF SIGNALS

The reconstruction strategies discussed in the previous section can be viewed as tools working on a pixel-based representation of the image that provide a way to create connected operators, which, in essence, are region-based processing tools. In this section, we present a different approach. The first step of the filtering process is to construct a hierarchical region-based representation of the image represented by a tree structure, then the simplification effect is obtained by pruning of the tree, and finally the filtered image is created from the pruned tree. The approach may be considered as being conceptually more complex than the reconstruction, however, it provides more flexibility in the choice of the image structures to be simplified and the simplification criterion.

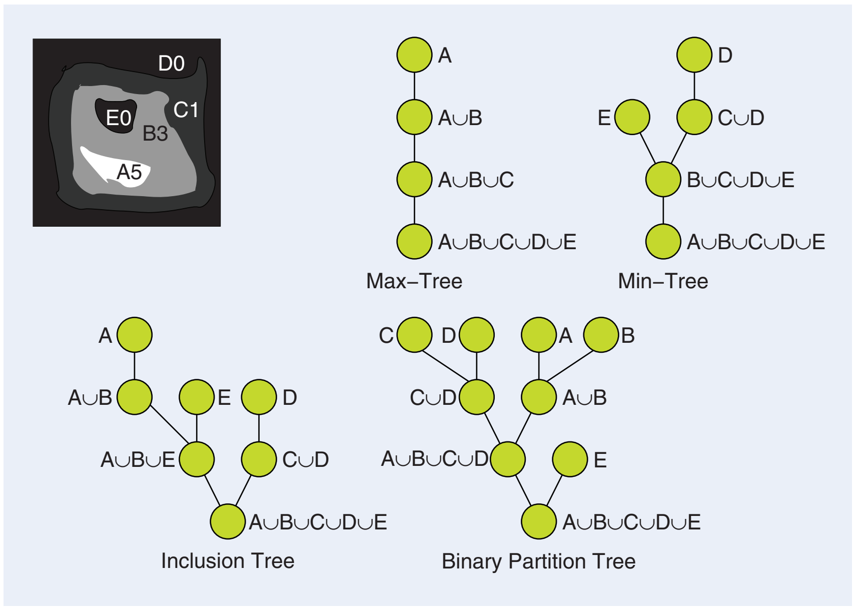
Four region-based representations are discussed next: the max-tree/min-tree [15], the inclusion tree [17], and the binary partition tree (BPT) [18]. The first two lead to antiextensive or extensive connected operators whereas the second two are the basis for self-dual connected operators. Let us discuss first the max-tree, min-tree, and the inclusion tree.

MAX-TREE, MIN-TREE, AND INCLUSION TREE

One of the simplest tree representations is known as the max-tree [15]. It structures the connected components of level set based on their inclusion relationship. Each tree node $\mathcal{N}_k$ represents a connected component of the space that is extracted by the following thresholding process: for a given threshold T, consider the set of pixels X_T that have a gray-level value larger than T and the set of pixels Y_T that have a gray-level value equal to T

$$X_T = \{n, \text{ such that } f[n] \geq T\}$$
$$Y_T = \{n, \text{ such that } f[n] = T\}. \tag{14}$$

The tree nodes $\mathcal{N}_k$ represent the connected components C of X such that $C \cap Y \neq \varnothing$. The links between the nodes represent the inclusion relationship between the connected components of X_T. A simple example is illustrated in Figure 8. The original image is composed of five flat zones identified by the letters A, B, C, D, and E. The numbers represent the gray-level value of the flat zones. The leaves of the max-tree are the image maxima. In this example, as the image has only one maxima, the max-tree has only one leaf, therefore one branch. The dual structure is called a min-tree. In this case, the leaves correspond to minima. It can be computed by duality by creating the max-tree of $-f$. As the image of Figure 8 has two minima, the min-tree has two leaves.



[FIG8] **Tree representations. The root is at the bottom.**

Various fast algorithms have been proposed in the literature to efficiently compute these trees [10], [15], [19]. They are described in the section "Efficient Computation of the Tree Representations." The pruning of a max-tree (min-tree) allows simplification, i.e., partial or total removal, of image maxima (minima). If one wants to interact with both maxima and minima, the processing strategy may either involve a combination of max- and min-tree-based filters or use a self-dual tree representation of the image as the inclusion tree. This structure (also known as a tree of shapes) represents maxima and minima in a symmetrical way. It structures the connected components of the level sets X_T based on the inclusion relationship of their saturation $sat(X_T)$. The saturation is an operator that fills the holes of X_T, that are the connected components of the background that are completely surrounded by X_T. An example of the inclusion tree is illustrated in Figure 8. As the image has three extrema, the tree has three leaves. The filtering by pruning of these tree structures will be illustrated in the section "Tree Simplification."

BINARY PARTITION TREE

The second example of region-based representation of images is the BPT [18]. It represents a set of regions obtained from an initial partition that we assume to be the partition of flat zones (but in practice any initial segmentation can be used). The leaves of the tree represent the flat zones of the original signal. The remaining tree nodes represent regions that are obtained by merging the regions represented by the children. The root node represents the entire image support. The tree represents a fairly large set of regions at different scales. This is why this representation can be viewed as a hierarchical region-based representation of the image. Large regions appear close to the root whereas small details can be found at lower levels. This representation should be considered as a compromise between representation accuracy and processing efficiency. Indeed, all possible merging of regions belonging to the initial partition are not represented in the tree. Only the most "likely" or "useful" merging steps are represented in the BPT. The connectivity encoded in the tree structure is binary in the sense that a region is explicitly connected to its sibling (since their union is a connected component represented by the father), but the remaining connections between regions of the original partition are not represented in the tree. Therefore, the tree encodes only part of the neighborhood relationships between the regions of the initial partition. However, the main advantage of the tree representation is that it allows the fast implementation of sophisticated processing techniques.

The BPT should be created in such a way that the most "interesting" or "useful" regions are represented. This issue can be application dependent. However, a possible solution, suitable for a large number of cases, is to create the tree by keeping track of the merging steps performed by a segmentation algorithm based on region merging (see [41] and [42], for example). In the following, this information is called the merging sequence. Starting from the partition of flat zones, the algorithm merges neighboring regions following a homogeneity criterion until a single region is obtained. An example is shown in Figure 8. The original partition involves five regions. The algorithm merges the five regions in four steps. In the first step, the pair of most similar regions, C and D, are merged. Then, region A is merged with region B. Then regions $A \bigcup B$ and $C \bigcup D$ are merged together and finally, region E is merged with region $A \bigcup B \bigcup C \bigcup D$ and this creates a region corresponding to the region of support of the whole image. In this example, the merging sequence is: $(C, D)|(A, B)|(A \bigcup B, C \bigcup D)|(A \bigcup B \bigcup C \bigcup D, E)$. This merging sequence defines the BPT as shown in Figure 8.

To completely define the merging algorithm, one has to specify the region merging order and the region model, i.e., the model used to represent the union of two regions. To create the BPTs used to illustrate the processing examples discussed in this article, we have used a merging algorithm following the color homogeneity criterion described in [42]. Let us define the merging order $O(R_1, R_2)$ and the region model M_R:

- *Merging order*: at each step the algorithm looks for the pair of most similar regions. The similarity between regions R_1 and R_2 is defined by

$$O(R_1, R_2) = N_1 \| M_{R_1} - M_{R_1 \cup R_2} \|_2 + N_2 \| M_{R_2} - M_{R_1 \cup R_2} \|_2, \quad (15)$$

where N_1 and N_2 are the numbers of pixels of regions R_1 and R_2 and $\|.\|_2$ denotes the $\mathcal{L}_2$ norm. M_R represents the model for region R. It consists of three constant values describing the YUV components. The interest of this merging order, compared to other classical criteria, is discussed in [42]. However, alternative order are discussed and evaluated in [28].

- *Region model*: as mentioned previously, each region is modeled by a constant vector YUV value: M_R. During the merging process, the YUV components of the union of two regions, R_1 and R_2, are computed as follows [42]:

$$\begin{aligned} &\text{if } N_1 < N_2 \Rightarrow M_{R_1 \cup R_2} = M_{R_2} \\ &\text{if } N_2 < N_1 \Rightarrow M_{R_1 \cup R_2} = M_{R_1} \\ &\text{if } N_1 = N_2 \Rightarrow M_{R_1 \cup R_2} = (M_{R_1} + M_{R_2})/2. \end{aligned} \quad (16)$$

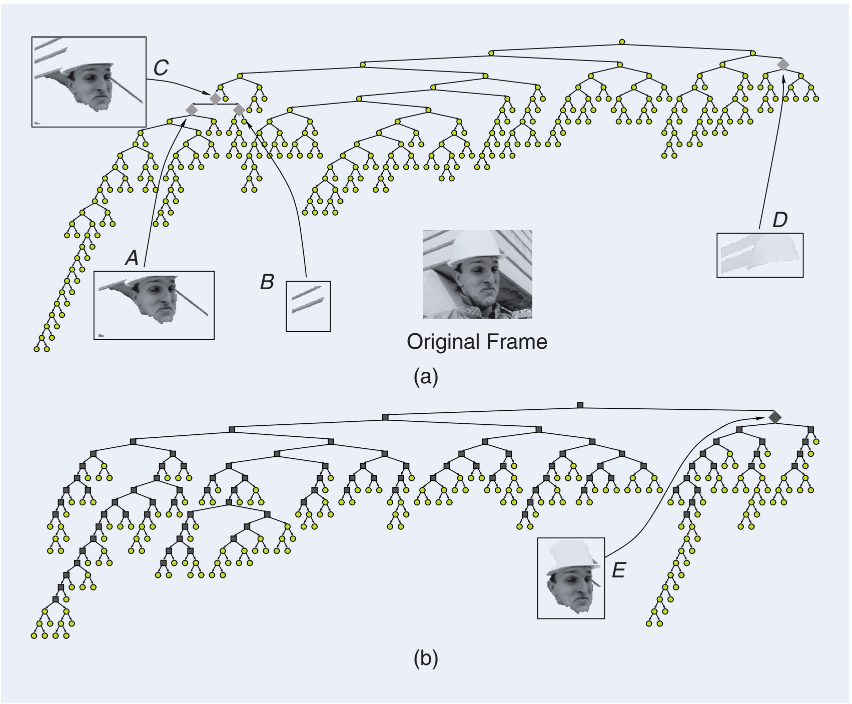
As can be seen, if $N_1 \neq N_2$, the model of the union of two regions is equal to the model of the largest region. This zero-order (constant) model is the simplest possible one. More advanced modeling strategies, for example, relying on the region histogram, are described in [43]. They are particularly useful in the case of highly textured images.

It should be noticed that the homogeneity criterion does not need to be restricted to color. For example, if the image for which we create the BPT belongs to a sequence of images, motion information can also be used: in the first stage, regions may be merged for example using a color homogeneity criterion, whereas a motion homogeneity criterion may be used in the second stage. Figure 9 shows an example of the Foreman sequence. In Figure 9(a), the BPT has been constructed exclusively with the color criterion described above. In this case, it is not possible to concentrate the information about the Foreground object (head and shoulder regions of Foreman) within a single subtree. For example, the face mainly appears in the subtree hanging from region A, whereas the helmet regions are located below region D. In practice, the nodes close to the root have no clear meaning because they are not homogeneous in color. Figure 9(b) presents an example of BPT created with color and motion criteria. The nodes appearing as white circles correspond to the color criterion, whereas the dark squares correspond to a motion criterion. The motion criterion is formally the same as the color criterion except that the YUV color distance is replaced by the YUV displaced frame difference (DFD). The process starts with the color criterion as in Figure 9(a) and then, when a given peak signal-to-noise ratio (PSNR) is reached, it changes to the motion criterion. Using motion information, the face and helmet now appear as a single region E.

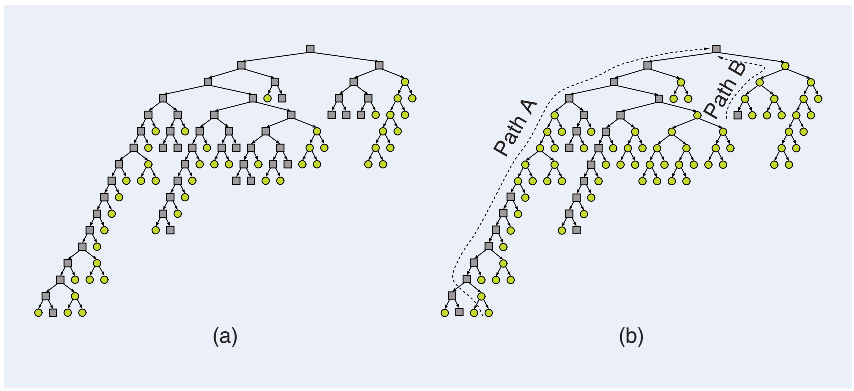
As can be seen, the construction of a BPT is considerably more complex than the creation of a max-or min-tree. However, BPTs offer more flexibility because one can choose the homogeneity criterion through the proper selection of the region model and the merging order. This choice has a direct impact on the definition of the structures that may be removed or simplified by the pruning process, which are not restricted to be maxima, minima or extrema. Because of this, it can handle vector-data without problems. Furthermore, if the functions defining the region model and the merging order are self-dual, the tree itself is self-dual. The same BPT can be used to represent f and $-f$. Note that in all cases, trees are hierarchical region-based representations. They encode a large set of regions and partitions that can be derived for the flat zones partition of the original image without adding new contours.

TREE SIMPLIFICATION

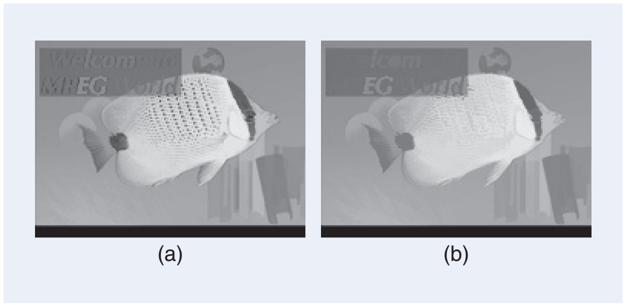
Once the representation has been created, the filtering strategy consists in simplifying the tree and in reconstructing an image from the simplified tree. Two general approaches might be recognized: i) pruning strategies and ii) nonpruning strategies. In the former, a single cut is made along each path from leaf to root, and all nodes leaf-side of the cut are collapsed onto the highest surviving ancestor. In the latter case, any number of nodes might be preserved or removed along a root path. Pruning strategies are often used because the idea is to eliminate the image components that are represented by the leaves



[FIG9] Examples of creation of BPT: (a) color homogeneity criterion and (b) color and motion homogeneity criteria.



[FIG10] (a) Example of increasing criterion (size). If a node has to be removed, all its descendants have also to be removed. Gray squares: nodes to be preserved, white circles: nodes to be removed. (b) Example of nonincreasing criterion (perimeter). No relation exists between the decisions among descendants (see decisions along Path A or Path B).



[FIG11] Area filtering: (a) area opening, γ^{area} and (b) area opening followed by area closing, $\varphi^{\text{area}}\gamma^{\text{area}}$.

and branches of the tree. The nature of these components depends on the tree. In the case of max-trees, min-trees, or inclusion trees, the components that may be eliminated are respectively regional maxima, minima, or extrema, whereas the elements that may be simplified in the case of BPTs are unions of the most similar flat zones, the notion of similarity depending on the merging criterion used to build the tree. The simplification itself is governed by a criterion that may involve simple notions such as size, contrast, or more complex ones such as texture, motion, or even criteria close to semantic notions, such as similarity to predefined shapes.

In tree representations, the set of possible merging steps is fixed (represented by the tree branches). As a result, sophisticated pruning strategies may be designed. An example of such a strategy deals with nonincreasing simplification criteria. Mathematically, a criterion $\mathcal{C}$ assessed on a region R is said to be increasing if the following property holds:

$$\forall R_1 \subseteq R_2 \Rightarrow \mathcal{C}(R_1) \leq \mathcal{C}(R_2). \tag{17}$$

INCREASING CRITERION

Assume that all nodes corresponding to regions where the criterion value is lower than a given threshold should be pruned. If the criterion is increasing, the pruning strategy is straightforward: merge all nodes that should be removed. It is indeed a pruning strategy since the increasingness of the criterion guarantees that, if a node has to be removed, all of its descendants have also to be removed. An example of BPT with increasing decision criterion is shown in Figure 10(a). The criterion used to create this example is the size, measured as the number of pixels belonging to the region, which is indeed increasing. Note that this example involves a BPT but the same issue also applies to all tree representations.

A typical filter corresponding to this situation, known as the area opening [12], [21] is illustrated in Figure 11. One possible implementation of the area opening consists in creating a max-tree and in measuring the area (the number of pixels) $\mathcal{A}_k$ contained in each node $\mathcal{N}_k$. If the area $\mathcal{A}_k$ is smaller than a threshold, $\mathcal{T}_{\mathcal{A}}$, the node is removed. The area criterion is increasing. It can be shown that the area opening is equal to the supremum of all possible openings by a connected structuring element involving $\mathcal{T}_{\mathcal{A}}$ pixels. The simplification effect of the area opening is illustrated in Figure 11(a). As expected, the operator removes small bright components of the image. If this simplified image is processed by the dual operator, the area closing, small dark components are also removed [see Figure 11(b)].

Using the same strategy, a large number of connected operators can be obtained. For example, if the criterion is the volume: $\sum_{n \in R} f[n]$ (also increasing), the resulting operator is the volumic operator. Many other properties or attributes such as area or diagonal length of the smallest bounding rectangle, moment of inertia, convex-hull area, and erosion width were suggested in [14] where the resulting connected filters were called attribute opening or closing. The idea behind this approach is to think of attribute filtering as selecting objects, represented by flat zones, in images based on prior knowledge of the objects of interest. The reader is also referred to [18] to see examples of this situation involving a BPT.

NONINCREASING CRITERION

If the criterion is not increasing, the simplification strategy is not straightforward, since the descendants of a node to be removed have not necessarily been removed. An example of such criterion is the region perimeter. If a region R_1 is included in region R_2, no specific relation can be stated about their respective perimeter.

Figure 10(b) illustrates a case of nonincreasing criterion. If we follow either Path A or Path B in Figure 10(b), we see that there are some oscillations of the remove/preserve decisions. Several simple strategies [15] can be used to deal with this case. Four of these strategies are listed next.

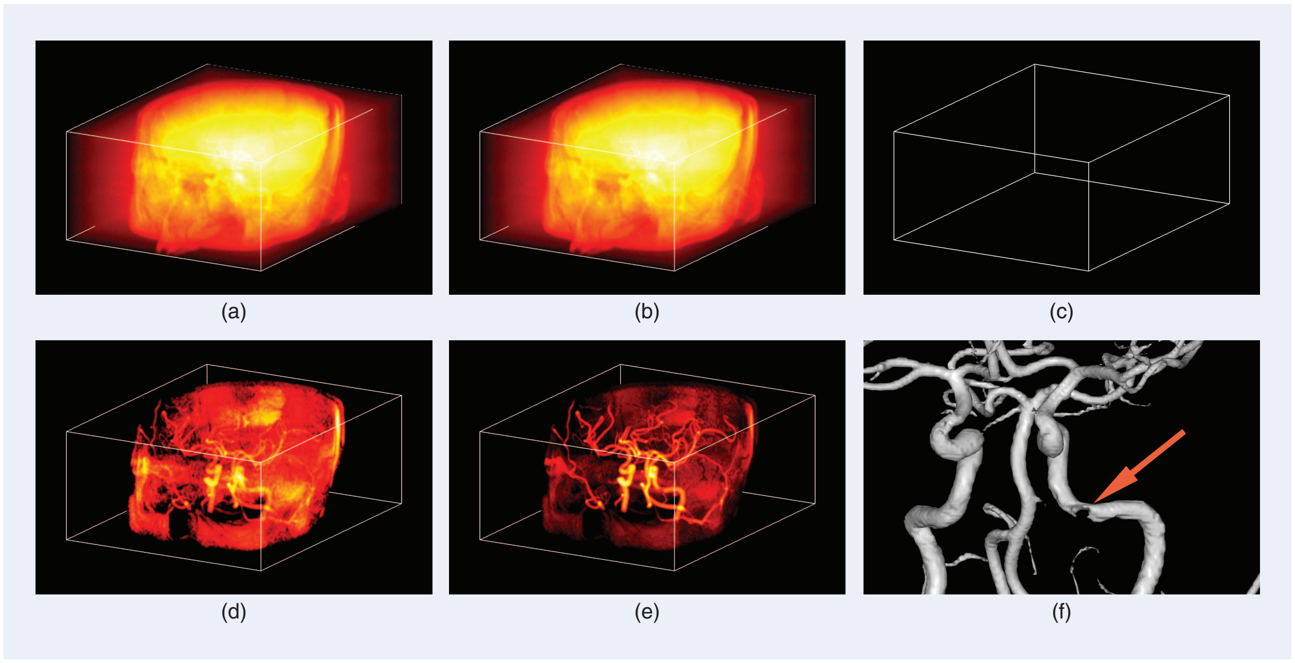
- The max rule prunes the branches from the leaves up to the first node that has to be preserved.
- The min rule prunes the branches from the leaves up to the last node that has to be removed.

- The direct rule consists of simply removing the nodes that have to be removed even if this does not create a pruning strategy. The pixels belonging to the nodes that have been removed are merged to the node of their first ancestor that has to be preserved.
- The subtractive rule [27] is the same as the direct rule except that the gray levels of surviving descendants of removed nodes are also lowered, so that the contrast with the local background remains the same.

These four simple strategies are illustrated in Figure 12, which shows a 3 tesla (3T) magnetic resonance time-of-flight angiography data set of a human head (the original is courtesy of Özlem Gürvit from the Institute for Neuroradiology, Frankfurt, Germany). Here the aim is to enhance the blood vessels without distortion. We observe that size does not identify blood vessels, but shape does. This means we need a scale-invariant criterion. This is obtained through the ratio of the trace of the moment-of-inertia tensor, and the volume to the power 5/3, which is just the 3-D counterpart of Hu's first moment invariant. This parameter is minimal for a solid sphere, and increases as the object becomes less compact. The attribute threshold is set to 3.7 in this case. Both nonpruning strategies [Figure 12(d) and (e)] suppress the background effectively, while preserving the blood vessels. However, the direct filter enhances the contrast of the parts of the skin that are also accepted at this attribute threshold. The reason is that their original gray level is preserved, whereas the background is reduced from around 55 to zero. This means their contrast is increased from around 10–50 to 65–105 gray levels. By contrast, the subtractive rule preserves contrast and the blood vessels show up more clearly. Note that this means that we are confident that the blood clot-like structure in Figure 12(f), found using the subtractive rule must be present at a contrast level of at least 50–60 gray levels in the original data.

All pruning strategies perform very poorly here, because one of the main concerns is removal of the background, i.e., nodes near the root of the tree. The max rule fails to remove this because the vessels are preserved at a higher level. By contrast the min rule removes everything, yielding a slow way to zero the volume.

Many nonincreasing criteria have useful invariance properties, such as scale invariance used above. Scale invariance is needed whenever we want to characterize objects by shape rather than size. This is the case when, e.g, the distance to the observer or focal length of the camera are unknown, or distances vary widely with in the same image (e.g., galaxies in astronomical images). The objects themselves might be better characterized by shape than size, e.g., blood vessels, or other organs, the size of which varies considerably during development and growth, but which have a distinctive shape. Often scale invariant image processing is approximated by multiscale filtering using either wavelets, granulometries, or scale spaces [44]. However, if using tree-based connected operators, this can be achieved using a single filtering step with considerable speed benefits [27], [45]. Examples of scale invariant criteria are the ratio of area and square of the perimeter length (sometimes called circularity), ratio of convex-hull area to area, or moment invariants such as those of Hu. A 3-D counterpart of the latter was used to effectively enhance blood vessels in computed tomography and magnetic resonance angiograms [45]. Other 3-D examples are given in [46]. We need not even limit ourselves to scale invariance. It has been shown that if a



[FIG12] An example of 3-D attribute filtering: (a) X-ray rendering of magnetic resonance angiogram; (b)–(e) result of attribute filter using different filtering rules (b) max; (c) min; (d) direct; (e) subtractive; and (f) detail of iso-surface rendering, showing blood clot-like structure. Images generated using the mtdemo program (http://www.cs.rug.nl/~michael/MTdemo/).

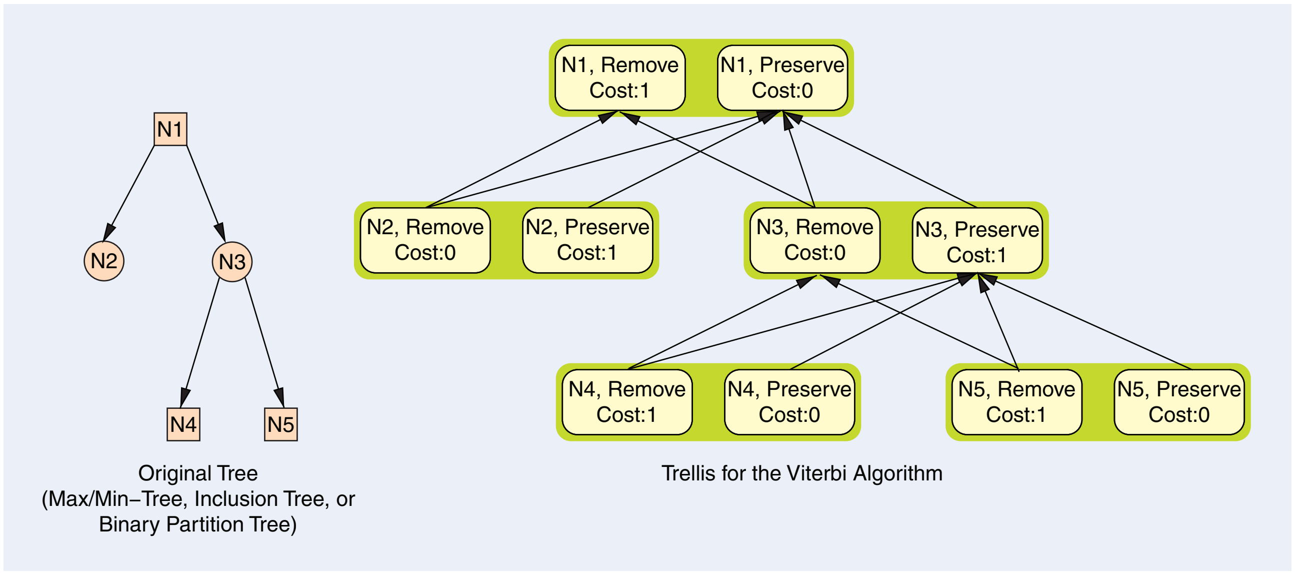
criterion T is invariant to any group of transformations, so is the tree-based connected operators derived from it, provided the connectivity is also invariant to the same transformations, i.e., each flat zone of the original image is mapped precisely onto a corresponding flat zone in the transformed image [27].

In some practical applications, the nonincreasingness of the criterion implies a lack of robustness of the operator. An example might be perimeter length in 2-D or surface area in 3-D. This is neither scale-invariant nor increasing. In such cases, similar images may produce quite different results or small modifications of the criterion threshold involve drastic changes on the output. In these situations, a possible solution to the nonincreasingness of the criterion consists of applying a transformation on the set of decisions. The transformation should create a set of increasing decisions while preserving the decisions defined by the criterion as much as possible. This approach may be viewed as dynamic programming problem that can be efficiently solved with the Viterbi algorithm.

The definition of the dynamic programming algorithm is explained and illustrated in the sequel assuming that the tree is binary but the extension to N-ary trees is straightforward. An example of trellis on which the Viterbi algorithm [47] is applied is illustrated in Figure 13. The trellis has the same structure as the tree except that two trellis states, preserve and remove correspond to each node of the tree. The two states of each child node are connected to the two states of its parent. However, to avoid nonincreasing decisions, the preserve state of a child is not connected to the remove state of its parent. As a result, the trellis structure guarantees that, if a node has to be removed, its children also have to be removed. The cost associated to each state is used to compute the number of modifications the algorithm has to apply to create an increasing set of decisions. If the criterion value states that the node of the tree has to be removed, the cost associated to the remove state is equal to zero (no modification) and the cost associated to the preserve state is equal to one (one modification). Similarly, if the criterion value states that the node has to be preserved, the cost of the remove state is equal to one and the cost of the preserve state is equal to zero. Although some modifications may be much more severe than others, the cost choice has no strong effect on the final result. This issue of cost selection is similar to the hard-versus-soft decision of the Viterbi algorithm in the context of digital communications [47]. The cost values appearing in Figure 13 assume that nodes $\mathcal{N}_1$, $\mathcal{N}_4$, and $\mathcal{N}_5$ should be preserved and that $\mathcal{N}_2$ and $\mathcal{N}_3$ should be removed. The goal of the Viterbi algorithm is to define the set of decisions such that

$$\text{Min} \sum_k \text{Cost}(\mathcal{N}_k) \text{such that the decisions are increasing.} \quad (18)$$

To find the optimum set of decisions, a set of paths going from all leaf nodes to the root node is created. For each node, the path can go through either the preserve or the remove state of the trellis. The Viterbi algorithm is used to find the paths that minimize the global cost at the root node and the trellis itself guarantees that this optimum decision is increasing. The optimization is achieved in a bottom-up iterative fashion. A detailed explanation of the algorithm can be found in [15] and [18]. A complete example is shown in Figure 14. The original tree corresponds to the one shown in Figure 10(b). The Viterbi algorithm has modified five decisions along Path A and one decision along Path B to get the optimum set of increasing decisions. This approach is particularly useful when the criterion has a certain "tendency" to increase but potentially defines several points along the branch where the pruning may be done. If the criterion has a tendency to increase, the various candidate pruning points are localized in the branch and the Viterbi algorithm



[FIG13] Trellis structure for the Viterbi algorithm. A circular (square) node on the tree indicates that the criterion value states that the node has to be removed (preserved). The trellis on which the Viterbi algorithm is run duplicates the structure of the tree and defines a preserve state and a remove state for each tree node. Paths from remove states to child preserve states are forbidden so that the decisions are increasing.

can efficiently define a single pruning point. If the criterion does not exhibit any tendency to increase, the potential pruning points may be randomly distributed along the branch and the Viterbi algorithm may not provide a useful solution. In these cases, the max, min, direct, and subtractive rules may provide interesting and simple strategies.

These strategies to define a pruning even if the criterion is not increasing are important because of practical applications, many useful criteria are not increasing. Various connected operators involving nonincreasing criteria such as entropy, simplicity, and perimeter can be found in [14], [15], and [18].

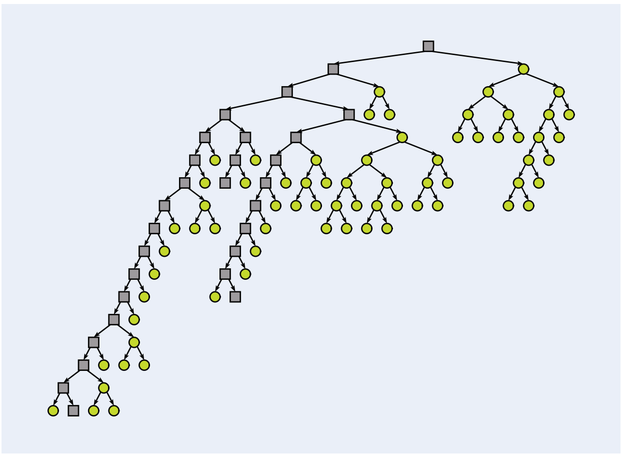
This situation is illustrated here by a motion-oriented connected operator based on a max-tree [15] and $f_t[n]$ represents an image sequence. The goal of the connected operator is to eliminate the image components that do not undergo a given motion defined by a displacement field at each position $\Delta[n]$. The field can be constant Δ if one wants to extract objects following a translation, but in general, the displacement depends on the spatial position n to deal with more complex motion models.

The sequence processing is performed as follows: each frame is transformed into its corresponding max-tree representation and each node $\mathcal{N}_k$ is analyzed. To check whether or not the pixels contained in a given node $\mathcal{N}_k$ are moving in accordance to the motion field $\Delta[n]$, a simple solution consists in computing the mean DFD of this region with the previous frame

$$\text{DFD}_{f_t}^{f_{t-1}}(\mathcal{N}_k) = \sum_{n\in\mathcal{N}_k} |f_t[n] - f_{t-1}[n-\Delta[n]]| / \sum_{n\in\mathcal{N}_k} 1. \quad (19)$$

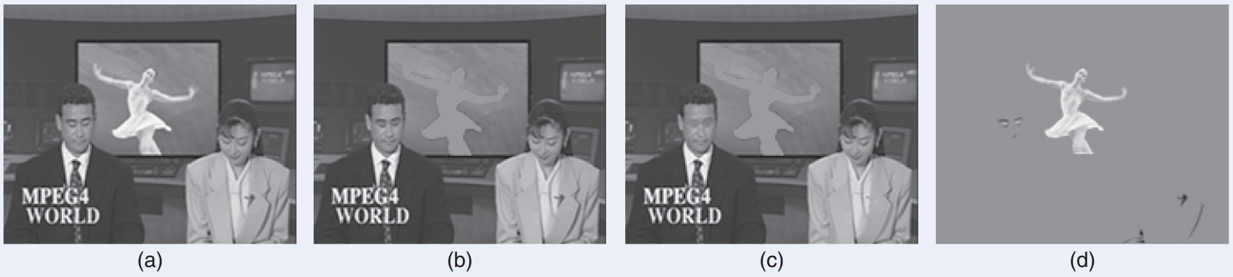
In practice, however, it is not very reliable to assess the motion on the basis of only two frames. The criterion should include a reasonable memory of the past decisions. This idea can be easily introduced in the criterion by adding a recursive term. Two mean DFD are measured: one between the current frame f_t and the previous frame f_{t-1} and a second one between the current frame and the previous filtered frame $\psi(f_{t-1})$ (ψ denotes the connected operator). The motion criterion is finally defined as

$$\text{Motion}(\mathcal{N}_k) = \alpha \text{DFD}_{f_t}^{f_{t-1}}(\mathcal{N}_k) + (1-\alpha)\text{DFD}_{f_t}^{\psi(f_{t-1})}(\mathcal{N}_k) \quad (20)$$


[FIG14] Set of increasing decisions resulting from the Viterbi algorithm used on the original tree of Figure 10(b). Five decisions along Path A and one decision along Path B have been modified. Gray squares: preserve, green circles: remove.

with $0 \le \alpha \le 1$. If α is equal to one, the criterion is memoryless. Low values of α allow the introduction of an important recursive component in the decision process. In a way similar to recursive filtering schemes, the selection of a proper value for α depends on the application: if one wants to very rapidly detect any changes in motion, the criterion should be mainly memoryless ($\alpha \approx 1$), whereas if a more reliable decision involving the observation of a larger number of frames is necessary, then the system should rely heavily on the recursive part ($0 \le \alpha << 1$).

The motion criterion described by (19) and (20) deals with one set of motion parameters. Objects that do not follow the given motion produce a high DFD and should be removed. The criterion is not increasing and the Viterbi algorithm has to be used. A motion filtering example is shown in Figure 15. The operator goal is to remove all moving objects. The motion model is defined by $\Delta[n] = (0,0), \forall n$. In this sequence, all objects are still except the ballerina behind the two speakers and the speaker on the left side. The connected operator $\psi(f)$ removes all bright, moving objects [Figure 15(b)]. The dual operator $\psi^*(f) = -\psi(-f)$ removes all dark, moving objects [Figure 15(c)]. The residue (the difference with the original



[FIG15] Example of motion connected operator preserving fixed objects: (a) original frame, (b) motion connected operator ψ, (c) dual operator: $\psi^*\psi(f)$, and (d) residue: f – $\psi^*(\psi(f)$.

image) presented in Figure 15(d) shows what has been removed by the operator. As can be seen, the operator has precisely extracted the ballerina and the (moving) details of the speaker's face.

PRUNING STRATEGIES INVOLVING GLOBAL OPTIMIZATION UNDER CONSTRAINT

In this section, we illustrate a pruning strategy involving global optimization under constraint that also takes benefit of the tree structure. To fix the notations, let us denote the criterion that has to be optimized (we assume, without loss of generality, that the criterion has to be minimized) by $\mathcal{C}$ and the constraint by $\mathcal{K}$. The problem is minimizing the criterion $\mathcal{C}$ with the restriction that the constraint $\mathcal{K}$ is below a given threshold $\mathcal{T}_K$. Moreover, we assume that both the criterion and the constraint are additive over the regions represented by the nodes $\mathcal{N}_k$: $\mathcal{C} = \sum_{\mathcal{N}_k} \mathcal{C}(\mathcal{N}_k)$ and $\mathcal{K} = \sum_{\mathcal{N}_k} \mathcal{K}(\mathcal{N}_k)$. The problem is therefore to define a pruning strategy such that the resulting partition is composed of nodes $\mathcal{N}_i$ such that

$$\text{Min} \sum_{\mathcal{N}_i} \mathcal{C}(\mathcal{N}_i), \text{ with } \sum_{\mathcal{N}_i} \mathcal{K}(\mathcal{N}_i) \leq \mathcal{T}_K. \tag{21}$$

It has been shown [48] that this problem can be reformulated as the minimization of the Lagrangian $\mathcal{L} = C + \lambda K$, where λ is the so-called Lagrange parameter. Both problems have the same solution if we find λ^* such that $\mathcal{K}$ is equal (or very close) to the constraint threshold $\mathcal{T}_K$. Therefore, the problem is to find a set of nodes that creates a partition and also minimizes the criterion defined by (22). Now, to find this set of nodes, we use a pruning process applied on the tree

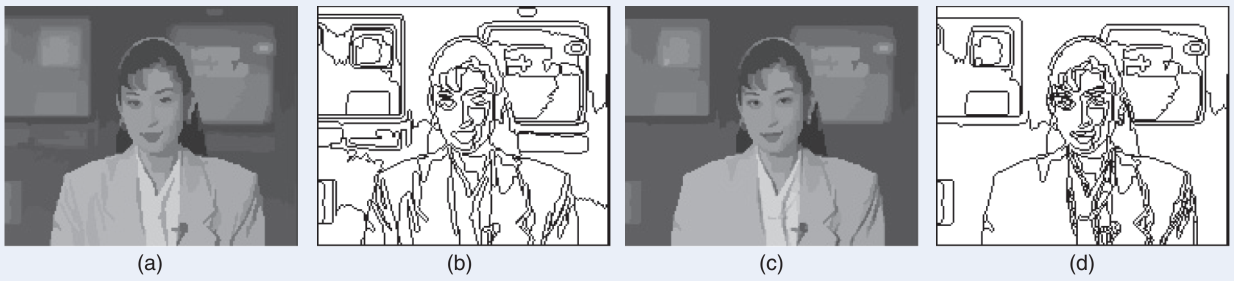
$$\text{Min}\left(\sum_{\mathcal{N}_i} \mathcal{C}(\mathcal{N}_i) + \lambda^* \sum_{\mathcal{N}_i} \mathcal{K}(\mathcal{N}_i)\right). \tag{22}$$

Assume, in a first step, that the optimum λ^* is known. In this case, the pruning is done by a bottom-up analysis of the tree. If the Lagrangian value corresponding to a given node $\mathcal{N}_0$ is smaller than the sum of the Lagrangians of the children nodes $\mathcal{N}_i$, then the children are pruned

$$\text{If } \mathcal{C}(\mathcal{N}_0) + \lambda^* \mathcal{K}(\mathcal{N}_0) < \sum_{\mathcal{N}_i} \mathcal{C}(\mathcal{N}_i) + \lambda^* \sum_{\mathcal{N}_i} \mathcal{K}(\mathcal{N}_i), \text{ prune the children nodes } \mathcal{N}_i. \tag{23}$$

This procedure is iterated up to the root node. In practice, the optimum λ^* is not known and the previous bottom-up analysis is embedded in a loop that searches for the best λ parameter. The computation of the optimum λ parameter can be done with a gradient search algorithm. The bottom-up analysis itself is not expensive in terms of computation since the algorithm has simply to perform a comparison of Lagrangians for all nodes of the tree. The part of the algorithm that might be expensive is the computation of the criterion and the constraint values associated to the regions. Note, however, that this computation has to be done once.

This type of pruning strategy is illustrated by two examples relying on a BPT. In the first example, the goal is to simplify the input image by minimizing the number of flat zones of the output image: $\mathcal{C}_1 = \sum_{N_k} 1$. In the second example, the criterion is to minimize the total length of the flat zones contours: $\mathcal{C}_2 = \sum_{N_k} \text{Perimeter}(N_k)$. In both cases, the criterion has no meaning if there is no constraint because the algorithm would prune all nodes. The constraint we use is to force the output image to be a faithful approximation of the input image: the squared error between the input and output images $\mathcal{K} = \sum_{\mathcal{N}_k} \sum_{n \in \mathcal{N}_k} (\psi(f)[n] - f[n])^2$ is constrained to be below a given threshold. In the examples shown in Figure 16, the squared error is constrained to be of at least 31 dB. Figure 16(a) shows the output image when the criterion is the number of flat zones. The image is visually a good approximation of the original image but it involves a much lower number of flat zones: the original image is composed of 14,335 flat zones whereas only 87 flat zones are present in the filtered image. The second criterion is illustrated in Figure 16(c). The approximation provided by this image is of the same quality as the previous one (squared error of 31 dB). However, the characteristics of its flat zones are quite different. The total length of the perimeter of its flat zones is equal to 3,684 pixels whereas the example of Figure 16(a) involves a total perimeter length of 4,491 pixels. The reduction of perimeter length is obtained at the expense of a drastic increase of the number of flat zones: 219 instead of 87. Figure 16(b) and (d) shows the flat zone contours. As can be seen, the flat zone contours are more complex in the first example but the number of flat zones is higher in the second one.



[FIG16] Example of optimization strategies under a squared error constraint of 31 dB: (a) minimization of the number of the flat zones, (b) contours of the flat zones of (a) (number of flat zones: 87, perimeter length: 4,491), (c) minimization of the total perimeter length, and (d) contours of the flat zones of (b) (number of flat zones: 219, perimeter length: 3,684).

This kind of strategy can be applied for a large number of criteria and constraints. Note that without defining a tree structure such as a max-tree or a BPT, it would be extremely difficult to implement this kind of connected operator.

EFFICIENT COMPUTATION OF TREE REPRESENTATIONS

Building trees is usually done in one of two approaches. The first approach is flood-filling, which generally starts at the root, and performs a depth-first or breadth-first flooding process to build the final tree. The second uses union-find approach, which was first introduced into connected filtering as an efficient means to perform area openings.

THE UNION-FIND APPROACH

An efficient algorithm [20] for area and attribute openings is based on Tarjan's union-find [49]. Union-find is an attractive choice as a basis for connected filter, because it is explicitly designed to administrate and merge disjoint sets. Connected filters always need to find disjoint sets based on flat zones and then merge them based on some criterion.

The way in which union-find represents sets, here, flat zones, is by trees, in which each member of the set points, either directly or indirectly at the canonical element of the set, which lies at the root of the tree. Two objects, `(n)` and `(p)`, are members of the same set if and only if `(n)` and `(p)` are nodes of the same tree, or equivalently that they share the same root of the tree they are stored in. There are four basic operations:

- `MakeSet(n)`: Create a new singleton set `{n}`. This operation assumes that `n` is not a member of any other set.
- `FindRoot(n)`: Return the root of the tree containing `x`.
- `Union(n,p)`: Form the union of the two sets that contain `n` and `p`.
- `Criterion(n,p)`: a symmetric criterion that determines whether `(n)` and `(p)` belong to the same set.

Pixels or voxel positions in the image or volume are given by a single index (the algorithm treats both as one-dimensional arrays). Before proceeding with the main loop, the pixels are sorted into a predetermined processing order. The union-find trees are stored in an array parent, which holds either a reference to the parent of a pixel `p`, or the area of the component as a negative number. After the resolving stage, the output image is contained in the same array. It is important to ensure that no cycles occur in the `parent` array. This is done by demanding that pointers always point towards the most recently processed pixels. This means we never need to perform a `FindRoot` on the current pixel, because it is always a root.

In the case of the area opening, this processing order is from high gray levels to low, and in lexicographic order within each gray level. Figure 17 shows the basic operations in the case of area openings. Here, `MakeSet(n)` sets the parent to −1, indicating the pixel is a root, and the area is 1, `Criterion(n,p)` assuming both parameters are roots, decide for a merge if either `n` and `p` have the same gray level, or the gray level of `n` differs, but the area indicated by `-parent[n]` is smaller than the area threshold. In `Union(n,p)`, `p` always indicates the current pixel, and therefore a root, and `n` a neighbor that has been processed before (and is therefore in a union-find tree). First the root of `n` is found. If the root `r` differs from `p`, and if `Criterion(r,p)` is true, we add `parent[r]` to `parent[p]`, which ensures that `parent[p]` now stores the sum of the two areas (as a negative number). Only then can we set `parent[r]` to `p`, which ensures the two sets are merged. If `Criterion(r,p)` is false, but `p` and `r` differ, we have encountered a higher gray-level component that is preserved, and therefore the current component (indicated by `p`) must be preserved. This is done by setting its area to the threshold, which prevent further mergers between regions with different gray levels.

Once all pixels have been processed, as shown in Figure 18, for example, we have a forest of trees representing the components in the filtered image. The root of each tree always has the lowest gray level within the component. We now need to resolve

```
void MakeSet ( int n )
{ parent[n] = -1;
}

int FindRoot ( int n )
{ if ( parent[n] >=0 )
    { parent[n] = FindRoot( parent[n] );
      return parent[n];
    }
  else return n;
}

boolean Criterion ( int n, int p )
{ return ( (f[n] == f[p]) ||
           ( -parent[n] < AreaThreshold ) ) ;
}

void Union ( int n, int p )
{ int r=FindRoot(n);
  if ( r != p )
    { if ( Criterion(r, p) )
        { parent[p] = parent[p] + parent[r];
          parent[r] = p;
        }
      else
        parent[p] = -AreaThreshold;
    }
}
```

[FIG17] Implementation of the basic operations for area openings and closings. Note that the areas of flat zones are stored as negative numbers in the corresponding roots. The parameters for `Criterion` must be root nodes.

```
/* array S contains sorted pixel list */
for (p=0; p<Length(S); p++)
  {
    pix = S[p];
    MakeSet(pix);
    for all neighbors nb of pix do
      if ((f[pix] < f[nb]) ||
          ((f[pix] == f[nb]) && (nb<pix)))
        Union(nb,pix);
  }
/* Resolving phase in reverse sort order */
for (p=Length(S)-1; p>=0; p--)
  {
    pix = S[p];
    if (parent[pix] >= 0)
        parent[pix] = parent[parent[pix]];
    else
        parent[pix] = f[pix];
  }
```

[FIG18] Code showing how to perform an area opening with the union-find approach using the operations of Figure 17.

the `parent` array. In this case, all pixels are assigned the original gray level of their root nodes. This is done by processing the pixels in reverse order, which guarantees that the first pixel we encounter of any component is its root. During the resolving phase, we check whether the current pixel is a root (i.e., `parent[pix]` is negative), and if so, assign `parent[pix]` the original gray level (stored in `f[pix]`). Otherwise, `parent[pix]` points to a pixel in the parent array that has already been resolved, and assign it that gray level. This simple algorithm does not build a full tree, but it can be extended to do this [19]. Further thoughts on the use of the use of union-find in connected filtering can be found in [50].

COMPLETE CONSTRUCTION OF THE TREE

Complete construction of the tree has many advantages over the partial construction just described. Computing the complete tree has the disadvantage of higher memory footprint, but it does allow far more complicated manipulation of the representation. This is particularly important in the case of a nonincreasing criterion. Moreover, it also allows multiple manipulations to be applied to the same tree. In practice, the tree can be built once and multiple filtered images can be created from this tree. Because the construction of the tree is generally the most costly stage, this allows speeding up the process of selecting the correct criteria in interactive applications, especially in 3-D [46].

The classical algorithm for building max-trees (and by duality, min-trees) uses a flood-filling approach [15] based on a hierarchical queue. It performs a depths-first sweep of the tree, starting at the lowest gray level of the image, which represents the root of the tree, and moving upwards in gray scale. As the tree is built, a STATUS array of the same dimensions as the image assigns labels to each pixel to indicate which node of the tree they belong to.

The union-find approach was combined with the flood-filling approach in [19]. Though the resulting algorithm is generally slower than the hierarchical queue approach, it has a lower theoretical complexity, especially when high numbers of gray levels (more than 12 b) are used. Indeed, it is the most efficient approach to date for floating point data [51].

The construction of an inclusion tree [17] can rely on the construction of the max- and min-trees. Three steps are necessary. First, the max- and min-trees of the image are computed taking into account the holes in each connected component stored in the tree nodes. Second, for each hole of a connected component corresponding to a node of the max-tree (min-tree), find the connected component that corresponds to it in the min-tree (max-tree). And finally, both trees are merged by putting connected components corresponding to holes as descendants of the ones containing them.

As explained in the section "Binary Partition Tree," the creation of a BPT can be done by keeping track of the merging steps performed by a segmentation algorithm based on region merging. Starting from an initial partition (potentially the flat zones or any other precomputed partition), the algorithm merges neighboring regions following a homogeneity criterion until a single region is obtained. When two regions are merged, two basic computation steps have to be done: first, compute the model of the union of the regions and second, update the value of the homogeneity criterion (also called merging order) corresponding to the neighboring regions. To have a fast algorithm, it is important to avoid going back to the pixels of the original image to compute the new model and update the merging order. To define a new node, it is indeed much more efficient to work recursively and rely on information that has been stored as descriptors of the children nodes. Finally, the last key point of the implementation is to be able to efficiently access the pair of regions that has to be merged that is the closest pair of regions following the homogeneity criterion.

PARALLEL COMPUTATION

Through the advent of multicore processors, parallel computation of connected filters becomes more important due to images and volume data sets becoming larger and parallel computers becoming cheaper. Most image processing operators are easily parallelized, either because of their local nature, or because they are separable. Connected filters are difficult, because they are not separable nor localized. Especially in the case of shape-based attributes, connected components of any size might be either accepted or rejected.

Regarding the max- and min-trees, a solution has been proposed [52] in the case of shared memory parallel computers. The solution lies in building separate max-trees for disjoint parts of the image or volume, and merging them efficiently, while maintaining attribute information. The image or volume is stored as a contiguous block in memory, and one of N_p contiguous sections of the data is assigned to each of the N_p processors. The method then uses an adaptation of the algorithm from [15] to build N_p max-trees. However, instead of assigning arbitrary labels to max-tree nodes, the first detected pixel of a node is used to indicate the label. This effectively has every pixel of a node point to this canonical element, and the node representation is done in a union-find style. The advantage of this approach lies in the stage in which the individual max-trees are merged. Consider a node of the max-tree of the entire data that straddles the boundary between to sections of the data. If arbitrary labels were assigned to the node, some equivalence table would be needed, with further memory burdens, or a relabeling would have to be performed. In the union-find approach, it is enough to let the canonical element of one region points to that of the other, and the merging is done.

NONCLASSICAL CONNECTIVITIES

Until now, we have implicitly considered the connected components and flat zones on which connected filters work to be the usual four- or eight-connectivity for 2-D images and six- or 26-connectivity for 3-D images. These connectivities rely on graph-based path connectivity: two pixels belong to the same connected component or flat zone if they can be connected by a path composed of a succession of neighboring pixels with the same value. The terminology "N-connectivity" used before specifies the number of neighbors each pixel has.

However, this approach to connectivity has several limitations. Because of strict preservation of edges, boundary noise cannot be removed. Furthermore, objects broken up by, e.g., sampling errors, are treated as different objects, rather than parts of the same. Conversely, disjoint real objects connected by noise pixels cannot be treated separately. More fundamentally, when using path connectivity, we might not obtain sets of pixels that correspond closely to perceptual groups relevant to human observers. Human observers might consider clusters of distinct, path-connected objects as a single perceptual group. An example might be a flock of birds or a cluster of stars. Conversely, objects that touch each other, and are therefore path connected, might be considered separate objects by human observers. It is quite easy to accommodate such perceptual groupings within the context of connected filters. The key notion for this is that of connections or connectivity classes [7], [8], [53]. An in-depth review of the mathematical foundations can be found in [9]. A connection is simply the set of all subsets of some set (e.g., the image domain) that are to be considered as connected sets. To define a connection, the following three rules in Definition 6 must be obeyed.

DEFINITION 6

Connection

A connection $\mathcal{C}$ is defined on the subsets of a set E when

1) the empty set is connected
2) any singleton is connected (i.e., any single pixel is connected)
3) for any set of connected sets, if the intersection is not empty, then their union is connected.

It can easily be verified that these properties hold for classical path connectivity. It was shown in [53] that this connection definition is equivalent to the definition of a family of connectivity openings $\{\Gamma_n\}$, associated to each pixel n belonging to the image support E. Intuitively, the opening $\Gamma_n(X)$ extracts the connected component of a binary set X that contains x. Let us recall this result in Definition 7.

DEFINITION 7

Connectivity Characterized by Openings

The definition of a connection is equivalent to the definition of a family of connectivity openings $\{\Gamma_n, n \in E\}$ such that

1) $\forall$ pixel n, $\Gamma_n(\{n\}) = \{n\}$
2) $\forall$ pixels n', n'' and a binary set X, $\Gamma_{n'}(X)$ and $\Gamma_{n''}(X)$ are either equal or disjoint
3) $\forall$ pixel n and binary set X, if the pixel n does not belong to X, then $\Gamma_n(X)$ returns the empty set $\varnothing$.

To change the notion of connectivity, the connectivity opening has simply to be modified while preserving the basic properties mentioned above. Later, we will briefly review the most classical examples.

SECOND-GENERATION CONNECTIVITIES

Most nonclassical connectivities are derived from path connectivity in some way. The most intuitive of these cases are so-called clustering-based connectivities. These take the path-connected components of an image, and group them into clusters, based on the size of the separation between them. It is the clusters of path-connected components that now form the new connected components. In this case, any set that is connected in the base connection, is also connected in the new connectivity class, but the reverse is not true.

To define a clustering-based connection, a simple approach consists in constructing a connectivity opening by first applying an extensive operator such as a dilation or a closing to the original image. These operators will tend to bridge narrow gaps between the original components. The connected components are then defined on this modified image, and then intersected with the original image. Assume that $\{\Gamma_n \mid n \in E\}$ is the family of connectivity openings associated with $\mathcal{C}$. If ψ is a clustering operator, the derived family of connectivity openings $\{\Gamma_n^\psi \mid n \in E\}$ is given by

$$\Gamma_n^\psi(X) = \begin{cases} \Gamma_n(\psi(X)) \cap X, & \text{if } n \in X \\ \varnothing, & \text{otherwise.} \end{cases} \qquad (24)$$

This definition means that the new connected components are defined on $\psi(X)$. But as $\psi(X)$ may be larger than X, we have to finally compute the intersection of the connected components extracted from $\psi(X)$ with X. The theoretical properties that have to be fulfilled by the clustering operator ψ are discussed in [54]. In practice, the classical choices for ψ in (24) are extensive dilations and closings by connected structuring elements. This example is illustrated in the first row of Figure 19. The original image is composed of two footprints. Using classical path connectivity, each footprint is composed of five connected components. If ψ is a closing with a flat structuring element whose size is larger than the distance between the connected components of the footprints, then $\psi(X)$ will involve only two connected components. As a result, only two connected components can be extracted by the connectivity opening Γ_n^ψ (referred to $\Gamma_p^\psi(X)$ and $\Gamma_q^\psi(X)$ in Figure 19).

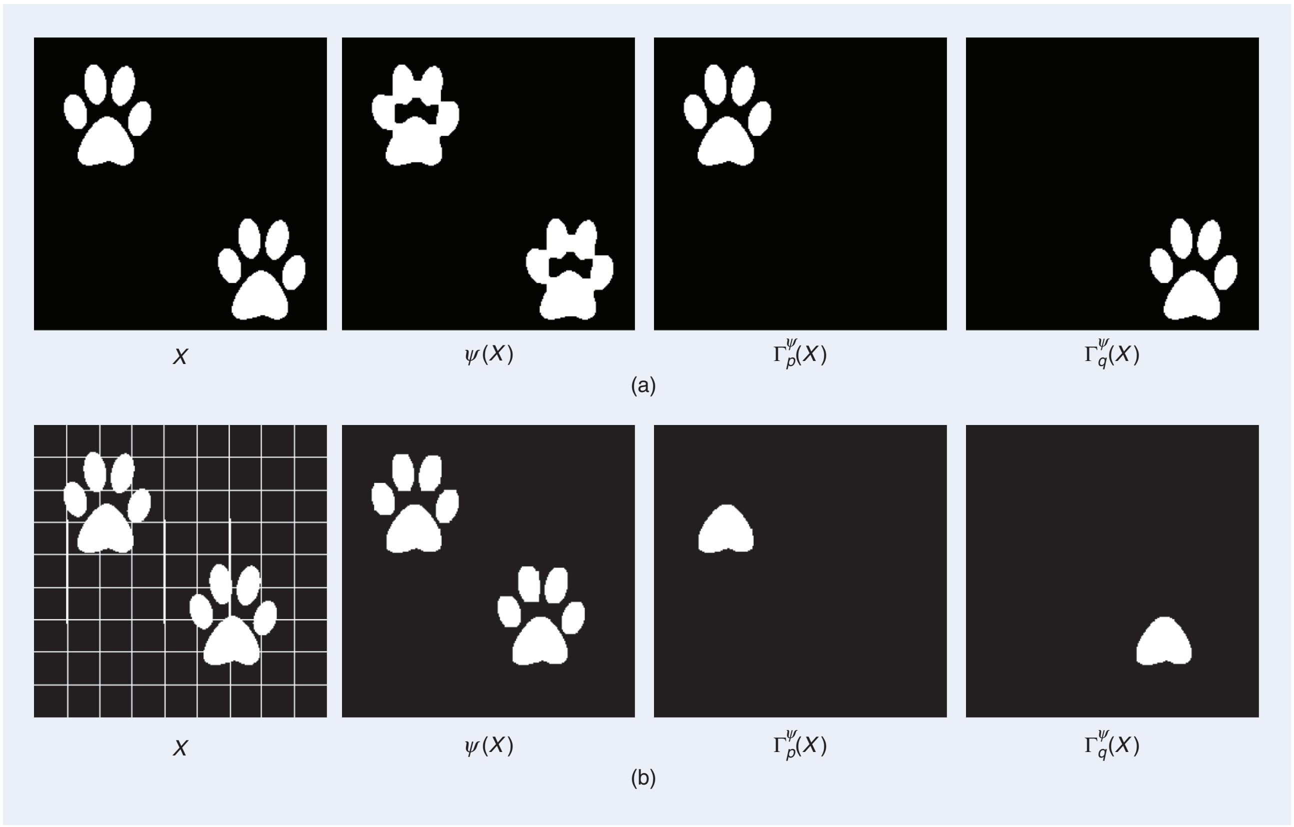
Alternatively, we can partition path-connected components into multiple fragments by cutting them at, e.g., narrow connecting "bridges" between wider regions. This idea leads to contraction-based connectivity [8]. In this case, ψ has to be an antiextensive operator, for example, an opening by a connected structuring element. Now, three situations can be distinguished: either $n \in \psi(X)$, $n \in X \setminus \psi(X)$, or $n \notin X$. The connectivity opening returns different results in each case

$$\Gamma_n^\psi = \begin{cases} \Gamma_n(\psi(X)) & \text{if } n \in \psi(X) \\ \{n\} & \text{if } n \in X \setminus \psi(X) \\ \varnothing & \text{otherwise.} \end{cases} \quad (25)$$

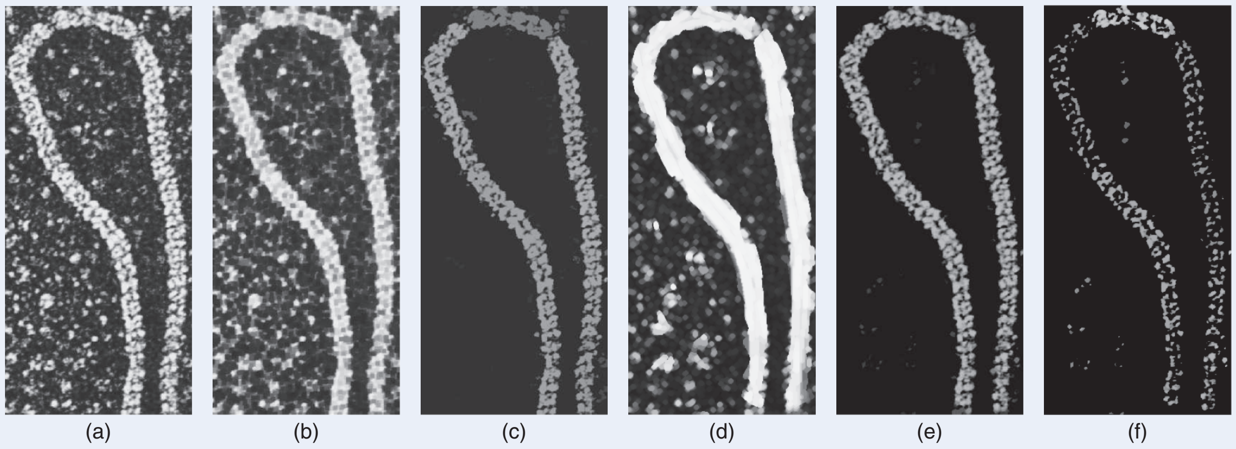
What this means is that if n is in the contracted image, the corresponding contracted connected component is returned. If n is a pixel in the original image that is removed by the contracting operator, it is considered a singleton (not connected to any other pixel). An example of contraction-based connectivity openings can be seen in the second row of Figure 19. In this example, using a simple path connectivity would be difficult as the image is composed of a single connected component. However, if ψ is an opening with a small structuring element, then $\psi(X)$ is composed of ten different connected components. Each of these individual connected components can be extracted by the connectivity opening Γ_n^ψ.

MASK-BASED CONNECTIVITY

In some cases, we might want to cluster in certain regions, but partition in others. In this case, the framework of mask-based connectivity is suitable [10]. The idea here is to compute a mask image M by any method and use the connected components of this mask to modify the connectivity of the image X under study. Thus, rather than using an operator to modify the connectivity openings, a mask image is used as



[FIG19] Second generation connectivity: (a) clustering-based connectivity and (b) contraction-based connectivity. On each row from left to right we see the original image, the result of $\psi(X)$, and the result of the connectivity openings at points p and q. Note that $p = (65, 85)$ and $q = (200, 225)$.



[FIG20] **Modifying connectivity for an electron micrograph of protein: (a) original image, (b) the mask by a closing with an SE of size 5 × 5, (c) the filtered output with $\lambda = 6$, (d) the mask described in the text, (e) the filtered output with $\lambda = 6$, and (f) the difference image after contrast enhancement. Image from [10].**

parameter for the connectivity openings. Once this mask M has been obtained, the connectivity openings can be defined as follows:

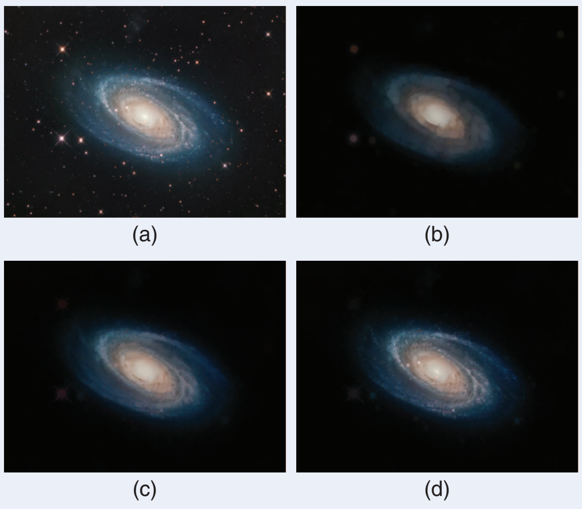
$$\Gamma_n(X|M) = \begin{cases} \Gamma_n(M) \cap X & \text{if } n \in M \cap X \\ \{n\} & \text{if } n \in X \setminus M \\ \varnothing & \text{otherwise.} \end{cases} \quad (26)$$

In other words, the mask takes the place of $\psi(X)$. However, M need not stem from an operation on X. Instead, M and X might be images of the same scene obtained at different wavelengths (e.g., IR and visible in astronomy), or in different modalities (e.g., registered magnetic resonance imaging and functional MRI data). In Figure 20, we can see an example of attribute filtering using the first moment invariant of Hu. For humans, the protein molecule in Figure 20(a) is considered a single perceptual group, despite its granular appearance. Using a closing with a small structuring element to "glue" the protein together in Figure 20(b), the resulting operator-based clustering connectivity yields a reasonable result in Figure 20(c). Some background objects have already been glued to the protein, however. Figure 20(d) shows a mask that first determines the dominant direction at each point using a Gabor filter bank. If such a dominant direction exists, an adaptive pseudodilation is performed in that direction at each point. Using this mask, we obtain a far better result, without gluing noise to the protein as can be seen in Figure 20(e) and (f).

MODIFYING FLAT ZONES

Another approach to changing connectivity is to change the definition of flat zones. The simplest approach is based on quasiflat zones [6], also known as λ-flat zone [15]. These are connected regions where, from any given pixel in the component, any other pixel of the same component can be reached through a path in which neighbors differ by no more than λ. This approach, first proposed by [55], reduces the fragmentation of images into very small flat zones but increases problems with leakage, because a narrow path of very low gradient might link to otherwise distinct regions, because λ only restricts the slope along minimum-slope paths, not the range of gray levels allowed.

An alternative is to use k-flat zones [56], which are defined as connected regions of maximal extent, where the total gray-level variation is no more than k. This prevents zones from becoming arbitrarily large, thus avoiding the under segmentation problem. However, k-flat zones do not provide a partition



[FIG21] **Separating galaxies from stars: (a) spiral galaxy M81, (original image, image courtesy of Giovanni Benintende), (b) stars suppressed by an opening with a disk of radius 8, (c) area attribute filter with 2,000 ≤ area ≤ 240,000, and (d) filtered result using *k*-flat version [56] of area attribute filter used in (c). The latter shows the best preservation of edge detail, and the best suppression of stellar signal.**

of the image, because they can overlap each other. The only way to use them is through hyperconnectivity (see [56] for their application to max- and min-trees). An example is shown in Figure 21. Hyperconnectivity [7] steps beyond the confines of connectivity and allows hyperconnected image regions to overlap without their union necessarily being hyperconnected. A similar effect is obtained using attribute-space connectivity [57], in which the image is first mapped into a higher dimensional space where a connection is applied, and the resulting structures are projected back onto the image domain.

Various ways to get around this problem have been sought, and they are reviewed in [58]. They all try to provide simultaneous smoothness, as in λ-flat zones, but restrict the growth and prevent overlap. The solution offered by Soille can be considered $(k, \lambda_{\max})$-flat zones, in which a succession of λ flat zones is built with increasing slope parameter λ (up to some maximum $\lambda_{\max}$), none of which may have a gray-level range larger than k. The approach has the advantage of providing a unique partition of the image domain, which is very difficult to achieve in any other way. This has been used in image simplification and filtering of satellite images. An important feature is that the method is readily extended to multichannel images by generalizing the definitions of slope and range to their multichannel equivalents [58]. A general theoretical background to definition of pseudoflat zones of different types to partition the image domain is given in [33] and [34].

CONCLUSIONS

This article has presented and discussed a region-based processing technique involving connected operators. There is currently an interest in defining processing tools that do not act on the pixel level, but on a region level. Connected operators are examples of such tools that come from mathematical morphology.

Connected operators are operators that process the image by merging flat zones. As a result, they cannot introduce any contours or move existing ones. The two most popular approaches to create connected operators have been reviewed. The first one works on a pixel-based representation of the image and involves a reconstruction process. The operator first involves a simplification step based on a "classical" operator (such as morphological open, close, low-pass filter, and median filter) and then a reconstruction process. Three kinds of reconstruction processes have been analyzed: antiextensive, extensive, and self-dual. The goal of the reconstruction process is to restore the contour information after the simplification. In fact, the reconstruction can be seen as a way to create a connected version of an arbitrary operator. Note that the simplification effect is defined and limited by the first step. The examples we have shown include simplification in terms of size or contrast.

The second strategy used to create connected operators involves three steps, First, a region-based representation of the input image is constructed. Four examples have been discussed: max-tree, min-tree, inclusion tree, and binary partition tree. In the second step, the simplification is obtained by modifying the tree and, in the third step, the output image is constructed from the simplified tree. The tree creation defines the set of regions that the pruning strategy can use to create the final partition. It represents a compromise between flexibility and efficiency: on the one hand, not all possible merging of flat zones are represented in the tree, but on the other hand, once the tree has been defined, complex pruning strategies can be defined. In particular, it is possible to deal in a robust way with nonincreasing criteria. Criteria involving the notions of area, shape, and optimization under a quality constraint have been demonstrated.

Quite uniquely, connected filters can easily be given many desirable invariance properties, such as scale invariance. Normally, multiscale analysis is used, often requiring several filtering discrete steps. In connected filters, multiscale analysis is often highly efficient. This is possible, because the tree representations form a compact multiscale representation of the image, and a multiscale analysis of the image (e.g., using pattern spectra) is as time consuming as a single filtering of the image [27]. Another extension is the use of vector attributes, which allows filtering based on multiple properties [59]. By supplying one or more prototype objects, the user could train a filter to detect specific classes of objects efficiently.

Given the rapid development of connected filters and the availability of fast algorithms, we expect a large increase in their use in many domains in the future.

AUTHORS

Philippe Salembier (philippe.salembier@upc.edu) received a degree from the Ecole Polytechnique, Paris, France, in 1983 and a degree from the Ecole Nationale Suprieure des Telcommunications, Paris, France, in 1985. He received the Ph.D. degree from the Swiss Federal Institute of Technology in 1991 and was a postdoctoral fellow at the Harvard Robotics Laboratory, Cambridge, Massachusetts, in 1991. He is currently a professor at Universitat Politecnica de Catalunya. His research interests include image and video coding, compression and indexing, video sequence analysis, mathematical morphology, level sets, and nonlinear filtering. He was a member of the Image and Multidimensional Signal Processing Technical Committee of the IEEE Signal Processing Society and technical chair of the 2003 IEEE International Conference on Image Processing. He was associate editor of *IEEE Transactions on Image Processing* and *IEEE Signal Processing Letters*.

Michael H.F. Wilkinson obtained an M.Sc. degree in astronomy from the University of Groningen, The Netherlands, in 1993. He later worked on image analysis of intestinal bacteria at the Department of Medical Microbiology at the University of Groningen, which formed the basis of his Ph.D. at the Institute of Mathematics and Computing Science (IWI), also in Groningen, in 1995. He became a researcher at the Centre for High Performance Computing in Groningen working on simulating the intestinal microbial ecosystem on parallel computers. He coedited *Digital Image Analysis of Microbes* (Wiley, U.K., 1998). He is currently a senior lecturer at the IWI.

REFERENCES

[1] J. Serra and P. Salembier, "Connected operators and pyramids," in *Image Algebra and Mathematical Morphology*, E. R. Dougherty, P. D. Gader, and J. Serra, Eds., vol. 2030. San Diego, CA: SPIE, July 1993, pp. 65–76.

[2] J. Crespo, J. Serra, and R. Schafer, "Theoretical aspects of morphological filters by reconstruction," *Signal Process.*, vol. 47, no. 2, pp. 201–225, 1995.

[3] H. J. A. M. Heijmans, "Connected morphological operators for binary images," *Comput. Vision Image Understanding*, vol. 73, no. 1, pp. 99–120, 1999.

[4] J. Crespo and R. Schafer, "Locality and adjacency stability constraints for morphological connected operators," *J. Math. Imaging Vision*, vol. 7, no. 1, pp. 85–102, 1997.

[5] F. Meyer, "From connected operators to levelings," in *Proc. 4th Int. Symp. Mathematical Morphology, ISMM'98*. Amsterdam, The Netherlands: Kluwer, June 1998, pp. 191–198.

[6] F. Meyer, "The levelings," in *Proc. 4th Int. Symp. Mathematical Morphology, ISMM'98*. Amsterdam, The Netherlands: Kluwer, June 1998, pp. 199–206.

[7] J. Serra, "Connectivity on complete lattices," *J. Math. Imaging Vision*, vol. 9, no. 3, pp. 231–251, 1998.

[8] C. Ronse, "Set-theoretical algebraic approaches to connectivity in continuous or digital spaces," *J. Math. Imaging Vision*, vol. 8, no. 1, pp. 41–58, 1998.

[9] U. Braga-Neto and J. Goutsias, "A theoretical tour of connectivity in image processing and analysis," *J. Math. Imaging Vision*, vol. 19, no. 1, pp. 5–31, 2003.

[10] G. K. Ouzounis and M. H. F. Wilkinson, "Mask-based second generation connectivity and attribute filters," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 29, no. 6, pp. 990–1004, 2007.

[11] J. Crespo and V. Maojo, "The strong property of morphological connected alternated filters," *J. Math. Imaging Vision*, vol. 32, no. 3, pp. 251–263, 2008.

[12] L. Vincent, "Morphological area opening and closing for grayscale images," in *Proc. NATO Shape in Picture Workshop*. Driebergen, The Netherlands: Springer-Verlag, Sept. 1992, pp. 197–208.

[13] L. Vincent, "Morphological gray scale reconstruction in image analysis: Applications and efficients algorithms," *IEEE Trans. Image Processing*, vol. 2, no. 2, pp. 176–201, Apr. 1993.

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

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

[16] C. Gomila and F. Meyer, "Levelings in vector space," in *Proc. IEEE Int. Conf. Image Processing, ICIP'99*, vol. 2, Kobe, Japan, Oct. 1999, pp. 929–933.

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

[18] P. Salembier and L. Garrido, "Binary partition tree as an efficient representation for image processing, segmentation and information retrieval," *IEEE Trans. Image Processing*, vol. 9, no. 4, pp. 561–576, Apr. 2000.

[19] L. Najman and M. Couprie, "Building the component tree in quasi-linear time," *IEEE Trans. Image Processing*, vol. 15, no. 11, pp. 3531–3539, Nov. 2000.

[20] A. Meijster and M. H. F. Wilkinson, "A comparison of algorithms for connected set openings and closings," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 24, no. 4, pp. 484–494, 2002.

[21] F. Cheng and A. N. Venetsanopoulos, "An adaptive morphological filter for image processing," *IEEE Trans. Image Processing*, vol. 1, no. 4, pp. 533–539, 1992.

[22] P. Salembier and M. Kunt, "Size-sensitive multiresolution decomposition of images with rank order based filters," *Signal Process.*, vol. 27, no. 2, pp. 205–241, May 1992.

[23] P. Salembier and J. Serra, "Flat zones filtering, connected operators and filters by reconstruction," *IEEE Trans. Image Processing*, vol. 3, no. 8, pp. 1153–1160, Aug. 1995.

[24] J. Crespo, R. Shafer, J. Serra, C. Gratin, and F. Meyer, "A flat zone approach: A general low-level region merging segmentation method," *Signal Process.*, vol. 62, no. 1, pp. 37–60, Oct. 1997.

[25] N. Ray and S. Acton, "Inclusion filters: A class of self-dual connected operators," *IEEE Trans. Image Processing*, vol. 14, no. 11, pp. 1736–1746, Nov. 2005.

[26] P. Maragos and G. Evangelopoulos, "Leveling cartoons, texture energy markers, and image decomposition," in *Proc. Int. Symp. Mathematical Morphology (INPE)*, G. J. F. Banon, J. Barrera, U. d. M. Braga-Neto, and N. S. T. Hirata, Eds. Rio de Janeiro, Brazil, Oct. 10–13, 2007, pp. 125–138.

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

[28] V. Vilaplana, F. Marques, and P. Salembier, "Binary partition trees for object detection," *IEEE Trans. Image Processing*, vol. 17, no. 11, pp. 1–16, Nov. 2008.

[29] J. Serra, *Image Analysis and Mathematical Morphology*. New York: Academic, 1982.

[30] J. Serra, *Image Analysis and Mathematical Morphology, Vol. II: Theoretical Advances*. New York: Academic, 1988.

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

[32] D. Gatica-Perez, C. Gu, M. T. Sun, and S. Ruiz-Correa, "Extensive partition operators, gray-level connected operators, and region merging/classification segmentation algorithms: Theoretical links," *IEEE Trans. Image Processing*, vol. 10, no. 9, pp. 1332–1345, 2001.

[33] J. Serra, "A lattice approach to segmentation," *J. Math. Imaging Vision*, vol. 24, no. 1, pp. 83–130, 2006.

[34] C. Ronse, "Partial partitions, partial connections and connective segmentation," *J. Math. Imaging Vision*, vol. 32, no. 1, pp. 97–125, 2008.

[35] J. Klein, "Conception et réalisation d'une unité logique pour l'analyse quantitative d'images," Ph.D. dissertation, Nancy Univ., France, 1976.

[36] M. Grimaud, "A new measure of contrast: The dynamics," in *Proc. SPIE Visual Communications and Image Processing'92*, vol. SPIE 1769, S. Gader and E. R. Dougherty, Eds. San Diego, CA: SPIE, July 1992, pp. 292–305.

[37] R. van den Boomgaard and A. Smeulders, "The morphological structure of images, the differential equations of morphological scale-space," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 16, no. 11, pp. 1101–1113, 1994.

[38] P. Maragos, "Differential morphology and image processing," *IEEE Trans. Image Processing*, vol. 5, no. 6, pp. 922–937, 1996.

[39] F. Meyer and P. Maragos, "Nonlinear scale-space representation with levelings," *J. Vis. Commun. Image Represent.*, vol. 11, no. 2, pp. 245–265, Apr. 2000.

[40] L. Vincent, "Graphs and mathematical morphology," *Signal Process.*, vol. 16, no. 4, pp. 365–388, Apr. 1989.

[41] O. Morris, M. Lee, and A. Constantinidies, "Graph theory for image analysis: An approach based on the shortest spanning tree," *IEE Proc. F*, vol. 133, no. 2, pp. 146–152, Apr. 1986.

[42] L. Garrido, P. Salembier, and D. Garcia, "Extensive operators in partition lattices for image sequence analysis," *Signal Process.*, vol. 66, no. 2, pp. 157–180, Apr. 1998.

[43] F. Calderero and F. Marqués, "General region merging approaches based on information theory statistical measures," in *Proc. Int. Conf. Image Processing* 2008, pp. 3016–3019.

[44] Y. Sato, S. Nakajima, N. Shiraga, H. Atsumi, S. Yoshida, T. Koller, G. Gerig, and R. Kinikis, "3D multiscale line filter for segmentation and visualization of curvilinear structures in medical images," *Med. Image Anal.*, vol. 2, no. 2, pp. 143–168, 1998.

[45] M. H. F. Wilkinson and M. A. Westenberg, "Shape preserving filament enhancement filtering," in *Proc. MICCAI'2001* (Lecture Notes in Computer Science, vol. 2208), W. J. Niessen and M. A. Viergever, Eds. Utrecht, The Netherlands: Springer, 2001, pp. 770–777.

[46] M. A. Westenberg, J. B. T. M. Roerdink, and M. H. F. Wilkinson, "Volumetric attribute filtering and interactive visualization using the max-tree representation," *IEEE Trans. Image Processing*, vol. 16, no. 12, pp. 2943–2952, 2007.

[47] A. Viterbi and J. Omura, *Principles of Digital Communications and Coding*. New York: McGraw-Hill, 1979.

[48] Y. Shoham and A. Gersho, "Efficient bit allocation for an arbitrary set of quantizers," *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. 36, pp. 1445–1453, Sept. 1988.

[49] R. E. Tarjan, "Efficiency of a good but not linear set union algorithm," *J. ACM*, vol. 22, no. 2, pp. 215–225, 1975.

[50] T. Géraud, "Ruminations on Tarjan's union-find algorithm and connected operators," in *Proc. Int. Symp. Mathematical Morphology (ISMM) 2005*, Paris, Apr. 18–20, 2005, pp. 105–116.

[51] C. Berger, T. Geraud, R. Levillain, N. Widynski, A. Baillard, and E. Bertin, "Effective component tree computation with application to pattern recognition in astronomical imaging," in *Proc. Int. Conf. Image Processing 2007*, San Antonio, TX, Sept. 16–19, 2007, pp. IV 40–44.

[52] M. H. F. Wilkinson, H. Gao, W. H. Hesselink, J. E. Jonker, and A. Meijster, "Concurrent computation of attribute filters using shared memory parallel machines," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 30, no. 10, pp. 1800–1813, 2008.

[53] J. Serra, "Mathematical morphology for Boolean lattices," in *Image Analysis and Mathematical Morphology, II: Theoretical Advances*, J. Serra, Ed. London: Academic, 1988, ch. 2, pp. 37–58.

[54] U. Braga-Neto and J. Goutsias, "Connectivity on complete lattices: New results," *Comput. Vision Image Understanding*, vol. 85, no. 1, pp. 22–53, 2002.

[55] M. Nagao, T. Matsuyama, and Y. Ikeda, "Region extraction and shape analysis in aerial photographs," *Comput. Graph. Image Process.*, vol. 10, no. 3, pp. 195–223, 1979.

[56] G. K. Ouzounis, "Generalized connected morphological operators for robust shape extraction," Ph.D. dissertation, Univ. Groningen, Groningen, The Netherlands, 2009.

[57] M. H. F. Wilkinson, "Attribute-space connectivity and connected filters," *Image Vision Comput.*, vol. 25, no. 4, pp. 426–435, 2007.

[58] P. Soille, "Constrained connectivity and connected filters," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 30, no. 7, pp. 1132–1145, July 2008.

[59] E. R. Urbach, N. J. Boersma, and M. H. F. Wilkinson, "Vector-attribute filters," in *Proc. Int. Symp. Mathematical Morphology (ISMM) 2005*, Paris, Apr. 18–20, 2005, pp. 95–104.