MOTION ANALYSIS OF IMAGE SEQUENCES USING CONNECTED OPERATORS

Luis Garrido, Albert Oliveras and Philippe Salembier

Universitat Politècnica de Catalunya
Campus Nord UPC, Módulo D5
c/ Gran Capita, s/n
08034 Barcelona, SPAIN
E-mail: {oster,albert,philippe}@gps.tsc.upc.es

ABSTRACT

This paper deals with a class of morphological operators called connected operators. *These operators interact with the signal by merging* flat zones. *As a result, they do not create any new contours and are very attractive for filtering tasks where the contour information has to be preserved. This paper focuses on a class of operators dealing with motion information. They remove from the original sequence the components that do not undergo a specific motion. They have a large number of applications including image sequence analysis with motion multiresolution decomposition and motion estimation.*

Keywords: *Mathematical morphology, Connected operators, Connectivity, Motion criterion, Optimization, Viterbi algorithm, Sequence processing.*

1 INTRODUCTION

The first *connected operators* reported in the literature are known as *binary opening by reconstruction*[4] . They consist in removing the connected components of a binary image that are totally removed by an erosion and in preserving the other components. This filtering approach offers the advantage of simplifying the image, because some components are removed, as well as preserving the contour information, because the components that are not removed are perfectly preserved.

This approach has been generalized for gray level functions using the so-called *reconstruction* process[16] . Beside opening by reconstruction, $\lambda - max$ operators, area opening[15] , dynamics filters[2] and, more recently, volumic[14] , complexity[6] , moment-oriented[1] and motion[8,9] operators have been proposed. These operators offer various simplification criteria (size, contrast, shape, etc.) while preserving contours. This property makes them very attractive for a very large number of applications such as noise cancellation, segmentation, pattern recognition, etc. The purpose of this paper is to focus on the class of motion-oriented connected operators and extend the results reported in[9] . In particular, the problem of non-increasing motion criterion partially discussed in[9] is analyzed with more details and its solution is formulated as an optimization problem that can be very efficiently solved by a Viterbi algorithm.

The organization of this paper is as follows: Section 2 is devoted to the notion of binary and gray level connected operators. This presentation highlights the usefulness of a tree representation of the image. Section 3 discusses the filtering step relying on a motion criterion whereas the image restitution, after the filtering of the tree, is presented in Section 4. Finally, Section 5 is devoted to the conclusions.

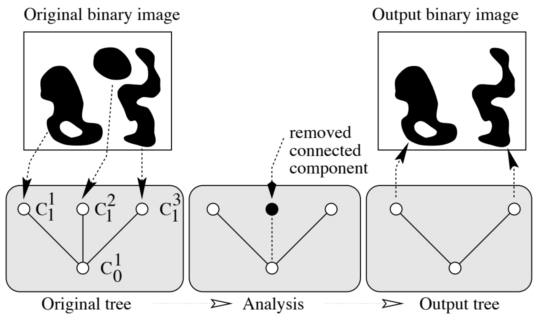


Figure 1: Binary connected operator

2 CONNECTED OPERATORS

21 Theoretical definition

Binary connected operators are formally defined as follows[13] :

Definition 2.1 (Binary connected operator). *A binary operator ψ is connected when for any binary image X, the set difference $X \setminus \psi(X)$ is exclusively composed of connected components of X or of its complement X^c.*

The extension of connected operators for gray level functions relies on the concept of partition[13,11] . Let us recall that a partition of the space E is a set of connected components $\{A_i\}$ which are disjoint and the union of which is the entire space. Each A_i is called a partition class. Moreover, a partition $\{A_i\}$ is said to be *finer* than another partition $\{B_i\}$ if any pair of points belonging to the same class A_i also belongs to a unique partition class B_j.

Let us call *flat zones* of a gray level function f the set of the largest connected components of the space where f is constant (note that a flat zone can be reduced to a single point). The set of flat zones of a function constitutes a partition of the space, called the *partition of flat zones* and leads to the following definition[13,11] :

Definition 2.2 (Gray level connected operator). *An operator Ψ acting on gray level functions is connected if, for any function f, the partition of flat zones of f is finer than the partition of flat zones of $\Psi(f)$.*

22 Binary anti-extensive connected operators

Let us see how this definition means that the operator works on a structured representation of the image. In the sequel, we restrict ourselves to the case of anti-extensive operators ($\forall X, \psi(X) \subseteq X$). Therefore, a binary anti-extensive connected operator is an operator that can only remove connected components of X. The filtering process can easily be explained if a tree representation of the image is used as shown in Fig. 1.

The original image X is composed of three connected components. It can be represented by a tree structure with four nodes: the root node C_0^1 represents the set of pixels belonging to the background X^c, and $\{C_1^k\}_{1\leq k\leq 3}$ represent the three connected components of the image. In this representation, the filtering process consists in analyzing each node C_1^k by assessing the value of a particular criterion. Assume for example that the criterion consists in counting the number of pixels belonging to a node (area opening[15]). Then, for each node, the criterion value is compared to a given threshold λ and the node is removed if the criterion is lower than λ. In the example of Fig. 1, node C_1^2 is removed because its area is small. As a result, its pixels are moved to the background node C_0^1 (the connected component is removed). As can be seen, the tree links represent the pixels' migration

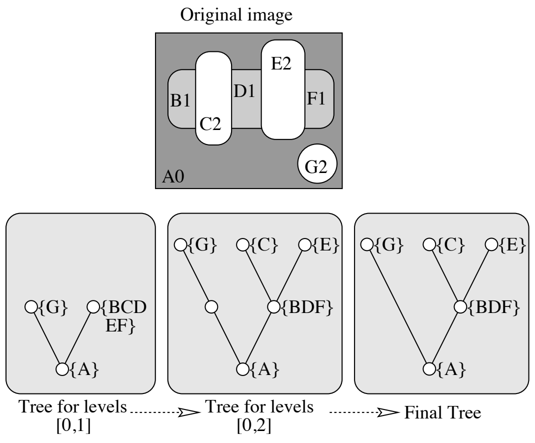


Figure 2: *Max-Tree* creation

(towards the father) when a node is removed. Note that this process leads to a simplification of the image (some connected components are removed) as well as a perfect preservation of the contour information of the remaining components (components that are not removed are perfectly preserved). All anti-extensive binary connected operators can be described by this process, the only modification being the criterion that is assessed.

23 Gray-level anti-extensive connected operators

As seen previously, the extension of connected operators to gray-level images is done via the notions of flat zones and the corresponding partition. This extension can also be seen as a simple generalization of the tree representation to the gray level case.

The idea consists in creating recursively the tree by a study of the thresholded versions of the image at all possible gray levels. An example is presented in Fig. 2. The original image is composed of seven flat zones identified by a letter $\{A, B, C, D, E, F, G\}$. The number following each letter defines the gray level value of the flat zone. In our example, the gray level values range from 0 to 2. In the first step, the threshold h is fixed to the gray level value 0. The image is binarized: all pixels at level $h = 0$, that is pixels of region A, are assigned to the root node of the tree $C_0^1 = \{A\}$. Furthermore, the pixels of gray level value strictly higher than $h = 0$ form two connected components that are temporarily assigned to two temporary nodes: $TC_1^1 = \{G\}$ and $TC_1^2 = \{B, C, D, E, F\}$. This creates the first tree (for gray levels $[0, 1]$). Note that this procedure is the same as the one used for the binary image. In a second step, the threshold is increased by one $h = 1$. Each node $TC_{h=1}^k$ is processed as the original image: consider, for instance, the node $TC_1^2 = \{B, C, D, E, F\}$. All pixels belonging to this node that are at level $h = 1$ remain assigned to this node which is called C_1^2. However, pixels of gray level value strictly higher than h (here $\{E, C\}$) create two different connected components and are moved to two temporary child nodes $TC_2^2 = \{C\}$ and $TC_2^3 = \{E\}$. The complete tree construction is done by iterating this process for all nodes k at level h and for all possible thresholds h (from 0 to the highest gray level value). The algorithm can be summarized saying that, at each temporary node TC_h^k, a “local” background is defined by keeping all pixels of gray level value equal to h (note that the “local” background itself is not *a priori* connected) and that the various connected components formed by the pixels of gray level value higher than h create the child nodes of the tree.

Note that in this procedure, some nodes may become empty. Therefore, at the end of the tree construction, the empty nodes are removed. The final tree is called a *Max-Tree* in the sense that it is a structured representation of the image which is oriented towards the maxima of the image (maxima are simply the leaves of the tree) and towards the implementation of anti-extensive operators. By duality, a *Min-Tree*, devoted to the implementation of extensive operators, can be defined.

The filtering itself is similar to the one used for the binary case. A criterion $\mathcal{M}(.)$ is assessed for each node C_h^k. Based on this value, $\mathcal{M}(C_h^k)$, the node is either preserved or removed. In this last case, the node's pixels are moved towards its father's node. At the end of the process, the output *Max-Tree* is transformed into a gray level image by assigning to the pixels of each node C_h^k the gray value h.

24 Connected operators and *Max-Tree* representation

Based on the previous description, the general filtering scheme is made of three steps. It involves a first step of *Max-Tree* creation, the goal of which is to structure the pixels in a suitable way for the filtering process[1]. The second step ("filtering step") is the filtering itself which analyzes each node and takes a decision on which node has to be preserved and which has to be removed. Finally, the last step ("restitution step") restores the filtered image by transforming the output *Max-Tree* into a gray level image. Note that the discussion of this paper will focus on anti-extensive operators and *Max-Trees*. By the duality $f \longleftrightarrow -f$, the same notions can be applied to extensive operators and *Min-Trees*. In the following, we describe the filtering and restitution steps.

3 FILTERING

31 Classical Criteria

Once the *Max-Tree* has been created, the filtering step analyzes each node by measuring a specific criterion and takes a decision on the elimination or preservation of the node. As examples, let us briefly recall some classical criteria used for the *opening by reconstruction*, the *area opening* and the $\lambda - max$ operator. In this section, we assume that node C_h^k is analyzed and denote by $\widehat{C_h^k}$ the set of pixels belonging to C_h^k and all its descendant nodes (note that in section 2, this notion has been used to explain the *Max-Tree* creation and the corresponding set of pixels was stored in a temporary node called TC_h^k).

- *Opening by reconstruction*[4]: this filter preserves a node C_h^k if the binary erosion of the set of pixels included in $\widehat{C_h^k}$ by a structuring element of size λ is not the empty set. This operator has a size-oriented simplification effect: it removes the bright components that are smaller than the structuring element. By duality, a closing by reconstruction can be defined. Its simplification effect is similar to that of the opening but on dark components.
- *Gray level area opening*[15]: this filter has been used in section 2. It is similar to the previous one except that it preserves the C_h^k if the number of pixels of $\widehat{C_h^k}$ is larger than a limit λ. It has also a size-oriented simplification effect, but the notion of size is different from the one used in the *opening by reconstruction*. By duality an *area closing* can be defined.
- $\lambda - max$ operator: the criterion here is to preserve the node C_h^k if this node has at least one non-empty descendant node at level $h + \lambda$ ($\exists k'$ such that $C_{h+\lambda}^{k'} \cap \widehat{C_h^k} \neq \emptyset$). The simplification effect of this operator is contrast-oriented in the sense that it eliminates image components with a contrast lower than λ. Note that, the $\lambda - max$ is an operator and not a morphological filter because it is not idempotent. By duality, the $\lambda - min$ operator can be defined.

[1]The *Max-Tree* representation has also the advantage of leading to very efficient implementations of connected operators (see[10] for more details)

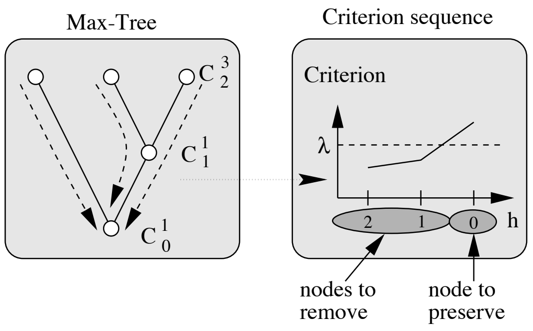


Figure 3: *Max-Tree* and *criterion sequence* for each local maximum

All these operators are extensively used in practice. A large set of examples can be found in particular in[5,7,16,15,11] and in the references they contain.

32 Non increasing criteria

The classical criteria and resulting operators are increasing. In the lattice theory, an operator ψ is said to be increasing if it does not modify the order between any pair of elements of the lattice: $\forall x \leq y, \psi(x) \leq \psi(y)$. In the context of connected operators, this property is directly related to the criterion assessed for each node. Assume that the node $C_{h_1}^{k_1}$ is a child node of $C_{h_2}^{k_2}$ (that is $C_{h_1}^{k_1} \cap \widehat{C_{h_2}^{k_2}} \neq \emptyset$). Because of the tree structure, we have the following relations $h_1 \geq h_2$ and $\widehat{C_{h_1}^{k_1}} \subseteq \widehat{C_{h_2}^{k_2}}$. If the criterion $\mathcal{M}(.)$ is increasing, we have also $\mathcal{M}(C_{h_1}^{h_1}) \leq \mathcal{M}(C_{h_2}^{k_2})$.

Let us analyze the effect of having an increasing or a non-increasing criterion on the *Max-Tree* representation and the decision process. Consider a maximum of the image, that is a leaf node of the *Max-Tree*, and the sequence of all its ancestor nodes going down to the root node. In the example of Fig. 3, if we start by the maximum corresponding to C_2^3, the sequence is $C_2^3 \rightarrow C_1^1 \rightarrow C_0^1$. Consider now the sequence of the criterion values $\mathcal{M}(C_h^k)$ obtained by scanning successively all the ancestors of a maximum. In the example of Fig. 3, the *criterion sequence* starting from C_2^3 is $M(h) = [\mathcal{M}(C_2^3), \mathcal{M}(C_1^1), \mathcal{M}(C_0^1)]$ and is represented as a curve (function of h) on the right side. Note that the parameter h itself is decreasing because the nodes are scanned starting for the maximum and going down to the root. If the criterion is increasing, the *criterion sequence* is itself increasing and there is no problem to define the level h where the criterion is higher than a given limit λ. In this case, all nodes such that $\mathcal{M}(C_h^k) < \lambda$ are removed and the corresponding pixels are moved to the first ancestor node such that $\mathcal{M}(C_h^k) \geq \lambda$.

If the criterion is non-increasing, the *criterion sequence* $M(h)$ may fluctuate around the λ value and the definition of the set of nodes to remove is less straightforward. Three rules have been reported in the literature[3,13,11] to deal with the non-increasing case:

1. "Direct" decision: This is the most straightforward approach which consists in considering that a node C_h^k is preserved if and only if $\mathcal{M}(C_h^k) \geq \lambda$. The content of the nodes that are removed are merged with their nearest preserved ancestors. This solution is illustrated in Fig. 4.B. In this figure, we show on the upper level, a simple gray level function (input) and the corresponding output function. Components with a criterion value higher than λ are shown in dotted line on the output function. Components with a criterion value lower than λ are not shown. On the lower level of the figure, we show the evolution of the *criterion sequence* and the corresponding nodes (and therefore components) to be preserved.

   The "Direct" approach has the advantage of being simple, however, in practice it is not very robust because the decisions are very local and do not depend on the decision of neighboring nodes (note that, in the case

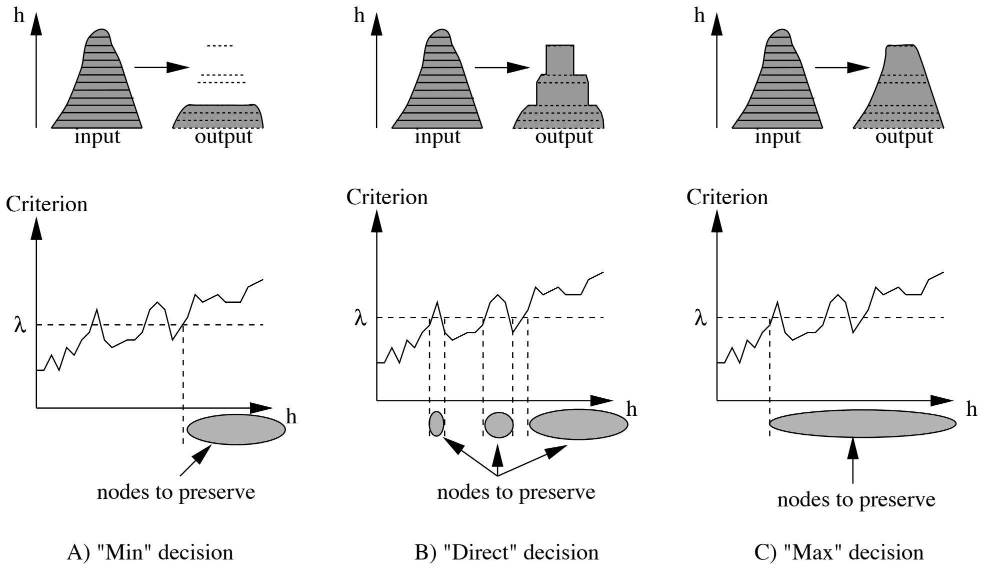


Figure 4: Illustration of various decision rules in the case of non-increasing criterion

of an increasing criterion, the decision is also local but, because of the increasing property, the relation between the decisions on various levels is *a priori* known).

2. "Min" decision: A node C_h^k is preserved if $\mathcal{M}(C_h^k) \geq \lambda$ and if all its ancestors are also preserved. This rule preserves less nodes than the first one. It is illustrated in Fig. 4.A.

3. "Max" decision: The last rule in the dual of the previous one: a node is removed if $\mathcal{M}(C_h^k) < \lambda$ and if all its descendant nodes are also removed (see Fig. 4.C).

From our practical experience, the "Min" and "Max" rules are generally more robust and lead to more coherent decisions. In particular, the "Direct" decision is the less robust and gives noisy results (presence of isolated small connected components that should "intuitively" be removed but are not). However, this conclusion is highly influenced by the type of criterion and its "degree of nonincreasingness". In[9] , an improvement of the decision robustness is proposed. It consists in filtering the decision sequence, for example, with a median filter. This solution actually provides more robustness, however, in the following, a different solution is proposed. It turns out to be much more robust than any of the previous ones. It relies on a formulation of the decision process as an optimization problem.

For each node C_h^k, a binary decision: preserve or remove, has to be taken. Assume that a decision cost is assigned to each possible decision for each node. In this case, the decision problem can naturally be seen as finding the set of lowest cost paths that go from the leaves of the tree down to the root node. Let us describe this approach in detail with a simple tree involving a single branch as shown in Fig. 5.

First, assign to each node C_h of the *Max-Tree* two states, $\mathcal{P}_h$ and $\mathcal{R}_h$, describing the two possible decisions: "preserve" or "remove". Second, a trellis is constructed, as shown in Fig. 5, by creating transitions linking the possible decisions of one node and of its father. Between C_h^k and C_{h-1}^k, there are four possible transitions: $\mathcal{R}_h \rightarrow \mathcal{R}_{h-1}$, $\mathcal{R}_h \rightarrow \mathcal{P}_{h-1}$, $\mathcal{P}_h \rightarrow \mathcal{R}_{h-1}$ and $\mathcal{P}_h \rightarrow \mathcal{P}_{h-1}$. Furthermore, a cost is assigned to each transition. The same cost is assigned to the two transitions going to a "preserve" state. This cost should reflect the reliability of the "preserve" decision for that node. This reliability can be measured for example by the difference between the decision threshold and the criterion value $\lambda - \mathcal{M}(C_h)$ (if the reliability is very high, the cost is very low). In the case of a transition emanating from a "remove" state and going to a "remove" state, the situation is similar and

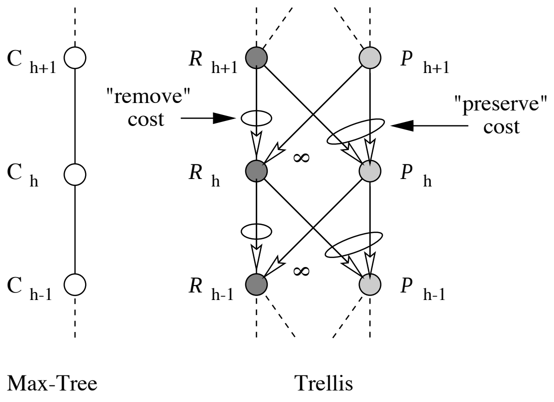


Figure 5: Trellis construction for the decision in the case of a single branch tree

the value $\mathcal{M}(C_h) - \lambda$ can be assigned as transition cost. This is however not the case for the transition emanating from a "preserve" state and coming to a "remove" state. Indeed, these transitions should be avoided because we want to define a level h above which all nodes are removed and below which all nodes are preserved. We choose this strategy to get back to the situation where the criterion is indeed increasing. These transitions can be easily eliminated if an infinite cost is assigned to them. Now, the decision consists in finding in this trellis, the path of lowest cost that starts from the maximum and ends at the "preserve" state of the root node (at least, the root node should be preserved). The cost of a path is defined as the sum of the costs of its transitions.

This formulation is a classical dynamic programming problem that can be very efficiently solved by the Viterbi algorithm[17] . Let us briefly describe this algorithm: assume that the two optimum (lowest cost) paths starting from the maximum and ending at $\mathcal{P}_{h+1}$ and $\mathcal{R}_{h+1}$ are known. Let us call $Path^{\mathcal{P}}_{h+1}$ and $Path^{\mathcal{R}}_{h+1}$ these two optimum paths. The definition of the two optimum paths ending at $\mathcal{P}_h$ and $\mathcal{R}_h$ can be easily defined by a local decision. For example, the optimum path ending in $\mathcal{R}_h$, that is $Path^{\mathcal{R}}_h$, is defined by the following rule:

$$\begin{aligned} \text{If} \quad & \text{Cost}(Path^{\mathcal{P}}_{h+1}) + \text{Cost}(\mathcal{P}_{h+1} \to \mathcal{R}_h) \;\leq\; \text{Cost}(Path^{\mathcal{R}}_{h+1}) + \text{Cost}(\mathcal{R}_{h+1} \to \mathcal{R}_h) \\ & \text{then } (Path^{\mathcal{R}}_h) = (Path^{\mathcal{P}}_{h+1}) \cup \{\mathcal{P}_{h+1} \to \mathcal{R}_h\} \\ & \text{else } (Path^{\mathcal{R}}_h) = (Path^{\mathcal{R}}_{h+1}) \cup \{\mathcal{R}_{h+1} \to \mathcal{R}_h\} \end{aligned} \tag{1}$$

This rule simply states that the optimum path ending at state $\mathcal{R}_h$ has to go through either state $\mathcal{R}_{h+1}$ or state $\mathcal{P}_{h+1}$ and that the best path is the one leading to the lowest additive cost. A similar decision rule can be defined for the best path ending at state $\mathcal{P}_h$. This process is iterated until the root node $h = 0$ and the optimum path is progressively constructed on the basis of local decisions. Finally, once the optimum path is found, the states it goes through define the decisions for each node.

This simple procedure has to be extended to deal with trees with various branches. The extension is described in Fig. 6 in the case of the junction of two branches but the procedure is general and can deal with an arbitrary number of branches. Let us analyze the case of the state $\mathcal{R}_h$: there is not one but two optimum paths ending at this state. One path comes from branch 1 whereas another one comes from branch 2. Note that they are independent from each other. As a result, we have to define independently these two paths. In Fig. 6, two sets of transitions, identified by solid and doted lines, can be seen. The decision defined by Eq. 1 is used on both sets of transitions. Once these two optimum paths have been defined, their union is considered as "the optimum path" ending at state $\mathcal{R}_h$ and its cost is equal to the sum of the costs of two paths.

In practice, this algorithm turns out to be very robust. The robustness means that similar input images lead to similar output results. From our practical experience, the decision using the Viterbi algorithm is drastically

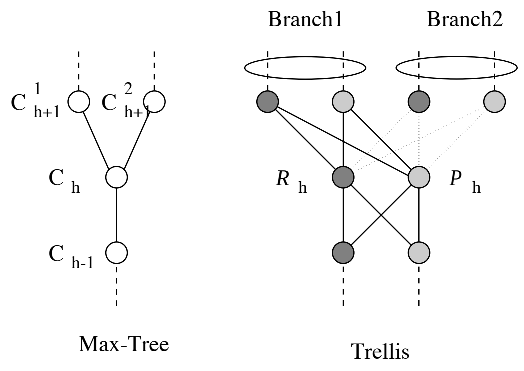


Figure 6: Trellis construction for the decision in the case of a multiple branches tree

superior to the other approaches. This advantage is obtained because the decision is global on the entire tree and not local as for the "Direct", "Min" or "Max" decisions. The robustness of the Viterbi approach is also reflected by the fact that decisions do not strongly depend on the cost assigned to each transition. For example, in practice, similar results are obtained if the costs proposed previously are replaced by their sign: either 1 or -1. Finally, note that when the criterion is increasing, all restitution techniques are equivalent.

The decision algorithm proposed in this section allows us to deal in a robust way with a very large number of criteria. For a given application, one has simply to characterize by a measure the flat zones of interests. In the following section, we concentrate on a motion criterion for image sequences.

33 Motion criterion

This criterion deals with motion information for sequences. Denote by $f_t(i,j)$ an image sequence where i and j represent the coordinates of the pixels and t the time instant. Our objective now is to define a connected operator able to eliminate the image components that do not undergo a given motion. The first step is therefore to define the motion model giving for example the displacement field at each position $\{\Delta_i(i,j), \Delta_j(i,j)\}$. The field can be constant $\{\Delta_i, \Delta_j\}$ if one wants to extract all objects following a translation, but in general the displacement can depend on the spatial position (i,j) to deal with more complex motion models such as affine or quadratic models.

The sequence processing is performed as follows: each frame is transformed into its corresponding *Max-Tree* representation and each node C_h^k is analyzed. To check whether or not the information contained in a given node C_h^k is moving in accordance to the motion field $\{\Delta_i(i,j), \Delta_j(i,j)\}$ a simple solution consists in considering the region created by the pixels of $\widehat{C_h^k}$ and to compute the opposite of the Mean Displaced Frame Difference ($\mathcal{D}$) of this region with the previous frame. Note that, the opposite of the mean DFD is used so that the criterion value for a region that has to be preserved is higher than the corresponding value when the region has to be removed. More formally, the criterion can be expressed as[9]:

$$\mathcal{D}_{f_t}^{f_{t-1}}(C_h^k) = - \sum_{i,j \in \widehat{C_h^k}} |f_t(i,j) - f_{t-1}(i - \Delta_i, j - \Delta_j)| \;/\; \sum_{i,j \in \widehat{C_h^k}} 1 \qquad (2)$$

In practice, however, it is not very reliable to state on the motion of part of the image on the basis of only two frames. The criterion should have 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

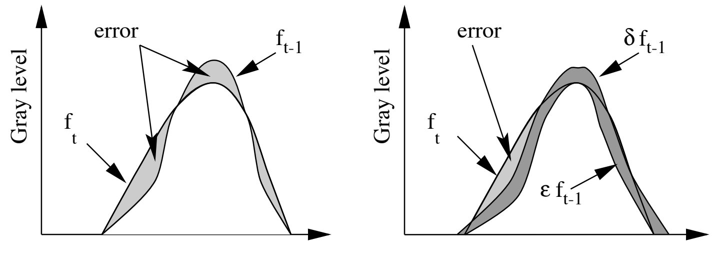


Figure 7: Motion criterion: Left: Criterion for one motion parameter, Right: criterion for a range of motion

denotes the connected operator). The motion criterion is finally defined as:

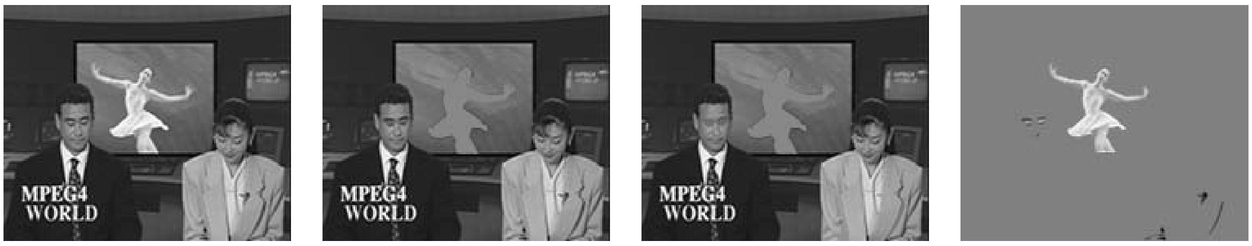
$$Motion(C_h^k) = \alpha \mathcal{D}_{f_t}^{f_{t-1}}(C_h^k) + (1-\alpha)\mathcal{D}_{f_t}^{\Psi(f_{t-1})}(C_h^k) \tag{3}$$

where $0 \leq \alpha \leq 1$. If α is equal to 1, the criterion is memoryless, whereas low values of α allow the introduction of an important recursive component in the decision process. In a way similar to all recursive filtering schemes, the selection of a proper value for α depends on the application: if one wants to detect very rapidly 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 ($0 \leq \alpha \ll 1$).

The motion criterion described by Eqs. 2 & 3 deals with one set of motion parameters. Objects that do not follow exactly the given motion are removed. For some applications, it is useful to preserve objects that are within a given range of motion (notion of "motion bandwidth"). To this end, the criterion of Eq. 2 can be modified by introducing an erosion ϵ and a dilation δ of the previous frame. The difference $|f_t - f_{t-1}|$ in the DFD ($\mathcal{D}$) is replaced at each point (i,j) either by $f_t - \delta(f_{t-1})$, if $f_t > \delta(f_{t-1})$; by $\epsilon(f_{t-1}) - f_t$, if $f_t < \epsilon(f_{t-1})$ or by 0 if $\delta(f_{t-1}) \leq f_t \leq \epsilon(f_{t-1})$. This approach is illustrated in Fig. 7. As can be seen, the erosion and the dilation of f_{t-1} create a "tube" in which the function f_t can remain without contributing to the DFD. The size of the structuring element used in the dilation and the erosion defines the motion "bandwidth".

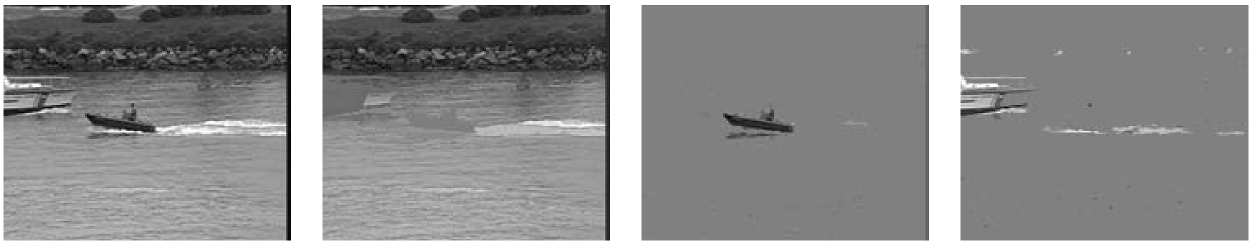
A first motion filtering example is shown in Fig. 8. The objective of the operator is to remove all moving objects. The motion model is defined by: $(\Delta_i, \Delta_j) = (0,0)$. In this sequence, all objects are still except the ballerina behind the two speakers and the speaker on the left side who is speaking. The application of the connected operator $\Psi(f)$ described previously removes all bright moving objects (Fig. 8.B.). The application of the dual operator: $\Psi^*(f)$ removes all dark moving objects (Fig. 8.C.). The residue (that is the difference with the original image) presented in Fig. 8.D shows what has been removed by the operator. As can be seen, the operator has very precisely extracted the ballerina and the (moving) details of the speaker's face.

The example illustrated in Fig. 9 shows a decomposition of the original image into three sequences: First the dominant translation is estimated giving the following motion model $(\Delta_i, \Delta_j) = (2,0)$. Objects following this translation are obtained by application of the motion operator followed by its dual (Fig. 9.B). Then, the difference



A) Original frame f B) Motion connected operator $\Psi(f)$ C) Dual operator $\Psi^*(\Psi(f))$ D) Residue f - $\Psi^*(\Psi(f))$

Figure 8: Example of motion connected operator preserving fixed objects



A) Original frame B) Objects with translation (2,0) C) Objects with translation (0,0) D) Remaining objects

Figure 9: Example of motion-oriented decomposition (A = B + C + D)

between the original frame and the filtered frame is computed and on this residue still objects $(\Delta_i, \Delta_j) = (0, 0)$ are extracted (Fig. 9.C). Finally, the remaining components are shown in Fig. 9.D. This is a decomposition[7] of the original sequence in the sense that the sum of the three sequences restores the original sequence. As can be seen, the filtering has clearly separated the background and the two boats moving in two different directions.

4 IMAGE RESTITUTION

After the *Max-Tree* creation, the criterion assessment and the decision, the last step of the filtering process consists in transforming the output *Max-Tree* into an output image. In all the previous examples, it has been assumed that the following procedure was used: assign to pixel $p(i, j)$ the gray level value h of the node C_h^k it belongs to. This is one of the simplest rules but it can be modified for specific applications.

Indeed, the decision classifies the nodes, and their corresponding pixels, into two classes: nodes to be removed and nodes to be preserved. A different restitution technique can therefore be assigned to each class. This approach can be seen as a "toggle mapping" problem[12] . It is quite natural to assume that, if a node has to be preserved, its content should not be modified by the connected operator. As a result, the original gray level values are used for the pixels of preserved nodes. By contrast, nodes to be removed correspond to areas that should disappear from the image. One approach consists in estimating the gray level values that would be seen if this area was actually not present in the image.

An example involving the motion-oriented operator is shown in Fig. 10. The objective is to preserve all image components that do not move. This sequence involves a fixed scene of a corridor with a person walking. The

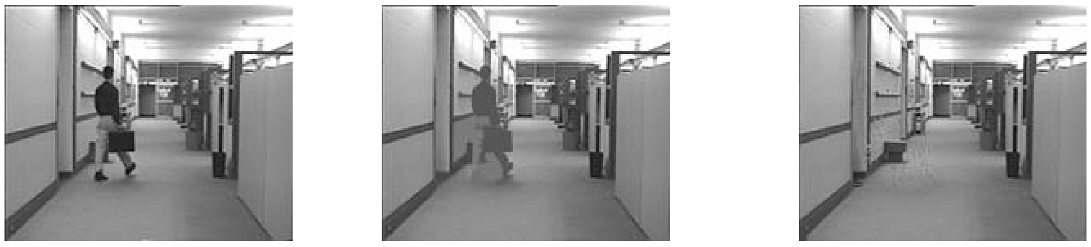

A) Original
B) Restitution without compensation
C) Restitution with compensation

Figure 10: Example of restitution with (C) and without (B) motion compensation in the case of motion connected operators

image of Fig. 10.B presents the result obtained by using in cascade the motion operator followed by its dual. The classical restitution where each removed node is merged with its first non-removed ancestor has been used. As can be seen, the person has been removed and replaced by flat zones of the background. However, since we are processing a time sequence, information of what is behind this person can be extracted from previous frames. This idea can be seen as a toggle mapping defined by the following rule: following the notations of section 33, if a pixel (i,j) belongs to a node to be preserved, the output gray level value $g_t(i,j) = \Psi(f_t(i,j))$ at time t is simply equal to the input gray level value: $g_t(i,j) = f_t(i,j)$. If the pixel belongs to a node to be removed, the output gray level value can be defined by motion compensation of the previous filtered frame: $g_t(i,j) = g_{t-1}(i - \Delta_i, j - \Delta_j)$. This example is illustrated in Fig. 10.C. As can be seen, the compensation has successfully estimated the image content behind the person.

5 CONCLUSIONS

This paper has focussed on morphological *connected operators*. These operators interact with the signal by merging *flat zones*. As a result, they do not create any new contours and are very attractive for filtering tasks where the contour information has to be preserved. Connected operators work implicitly on a structured representation of the image made of flat zones. The *Max-Tree* representation is a suitable and efficient structure to deal with the processing steps involved in anti-extensive connected operators.

The filtering step can directly act on the *Max-Tree* and consists in measuring a criterion and in taking a binary decision (remove or preserve) for each connected component. The issue of non increasing criterion has been extensively discussed. After presenting the classical solutions, an optimization formulation of the problem has been proposed for the decision step. The optimization problem can be very efficiently solved by the Viterbi algorithm. As application, the motion-oriented connected operator have been defined and illustrated. The last step called the restitution creates the output image from the filtered *Max-Tree*. Depending on the application, this step can be seen as a toggle mapping allowing several restitution strategies that may be particularly useful for image sequence processing.

The motion connected operator can potentially be used for a large set of applications. It opens the door in particular to different ways of handling the motion information. Indeed, generally, motion information is measured without knowing anything about the image structure. Connected operators take a different viewpoint by making decisions on the basis of the analysis of all possible flat zones, that is of all possible structures of the image. By using motion connected operators, we can "inverse" the classical approach to motion and, for example, analyze simplified sequences where objects are following a known motion. The application of theses operators to motion-oriented segmentation of sequences as well as to motion estimation seems to be a very interesting field of research.

6 REFERENCES

[1] E. Breen and R. Jones. An attribute-based approach to mathematical morphology. In P. Maragos, R.W. Schafer, and M.A. Butt, editors, *International Symposium on Mathematical Morphology*, pages 41–48, Atlanta (GA), USA, May 1996. Kluwer Academic Publishers.

[2] M. Grimaud. A new measure of contrast: the dynamics. In Serra Gader, Dougherty, editor, *Visual Communications and Image Processing'92*, volume 1769, pages 292–305, San Diego (CA), USA, July 1992.

[3] H. Heijmans. Theoretical aspects of gray level morphology. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6):568–592, 1991.

[4] J.C. Klein. *Conception et réalisation d'une unité logique pour l'analyse quantitative d'images.* PhD thesis, Nancy University, France, 1976.

[5] F. Meyer and S. Beucher. Morphological segmentation. *Journal of Visual Communication and Image Representation*, 1(1):21–46, September 1990.

[6] A. Oliveras and P. Salembier. Generalized connected operators. In Smith Ansari, editor, *Visual Communication and Image Processing, VCIP'96*, volume 2727, pages 761–773, Orlando (FL), USA, March 1996.

[7] P. Salembier and M. Kunt. Size-sensitive multiresolution decomposition of images with rank order based filters. *EURASIP Signal Processing*, 27(2):205–241, May 1992.

[8] P. Salembier and A. Oliveras. Practical extensions of connected operators. In P. Maragos, R.W. Schafer, and M.A. Butt, editors, *International Symposium on Mathematical Morphology, (Invited paper)*, pages 97–110, Atlanta (GA), USA, May 1996. Kluwer Academic Publishers.

[9] P. Salembier, A. Oliveras, and L. Garrido. Motion connected operators for image sequences. In *EUSIPCO'96*, volume II, pages 1083–1086, Trieste, Italy, September 1996.

[10] P. Salembier, A. Oliveras, and L. Garrido. Anti-extensive connected operators for image and sequence processing. *IEEE Transactions on Image Processing*, Submitted.

[11] P. Salembier and J. Serra. Flat zones filtering, connected operators and filters by reconstruction. *IEEE Transactions on Image Processing*, 3(8):1153–1160, August 1995.

[12] J. Serra. Toggle mappings. In J. C. Simon, editor, *From Pixels to Features*, pages 61–72. North Holland, Amsterdam, 1989.

[13] J. Serra and P. Salembier. Connected operators and pyramids. In SPIE, editor, *Image Algebra and Mathematical Morphology*, volume 2030, pages 65–76, San Diego (CA), USA, July 1993.

[14] C. Vachier. *Extraction de Caractéristiques, Segmentation d'Image et Morhpologie Mathématique.* PhD thesis, Paris School of Mines, 1995.

[15] L. Vincent. Grayscale area openings and closings, their efficient implementation and applications. In J. Serra and P. Salembier, editors, *First Workshop on Mathematical Morphology and its Applications to Signal Processing*, pages 22–27, Barcelona, Spain, May 1993. UPC.

[16] L. Vincent. Morphological gray scale reconstruction in image analysis: Applications and efficients algorithms. *IEEE, Transactions on Image Processing*, 2(2):176–201, April 1993.

[17] A.J. Viterbi and J.K. Omura. *Principles of Digital Communications and Coding.* Mc Graw-Hill, New York, 1979.