Anti-extensive connected operators with application to image sequences

L. Garrido, P. Salembier and A. Oliveras
Universitat Politècnica de Catalunya, Campus Nord - Módulo D5
C/ Gran Capità, 08034 Barcelona, Spain
E-mail: {oster, philippe, albert}@gps.tsc.upc.es

Abstract

Connected operators [3, 1, 2] are increasingly used in image processing. They are attractive in applications where the signal has to be simplified without loosing information about the contours. A large number of simplification criteria such as size, area [6], contrast or complexity [3] can be obtained with these operators. In this paper we deal with a motion-oriented connected operator. This operator eliminates from the original sequence the components that do not undergo a specific motion (defined as filtering parameter) while the remaining objects are almost perfectly preserved.

Key Words: Connected Operators, Motion Criterion, Optimization, Viterbi Algorithm, Sequence analysis.

1 Introduction

Motion information is a difficult issue in image sequence processing. Most of the time, motion is extracted from a local estimation that does not take into account the structure of the signal, that is the various objects in the scene. This is the case for the popular block-matching algorithm. The objective of this paper is to define a filtering tool taking into account the image structure and allowing the simplification of the image following a motion criterion.

As will be seen, the motion-oriented operator simplifies while preserving the information of the contours of the non-simplified objects. This filtering technique can be used for a large set of applications such as motion estimation, object tracking and motion-oriented multi-resolution decomposition.

The organization of this paper is as follows: the next section discusses the notion of *connected operators* based on a structured representation of the image called "*Max-Tree*". Section 3 is devoted to the definition of the motion criterion and to the non-increasingness of the criterion which may lead to instabilities in the filtered sequences. Finally, filtering examples are reported in section 4.

2 Connected Operators

2.1 Binary connected operators

Let X denote a binary image. As defined in [5], a binary *connected operator* Ψ is an operator that only removes connected components of X or of its complement X^c. In the sequel, we will restrict ourselves to the case of anti-extensive operators ($\forall X, \Psi(X) \subseteq X$). In this case, a binary *connected operator* is an operator that only removes connected components of X.

The filtering process can be easily explained if a tree representation of the image is used. This idea is illustrated in figure 1. The original image X is composed of three connected components. This image can be represented by a tree 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

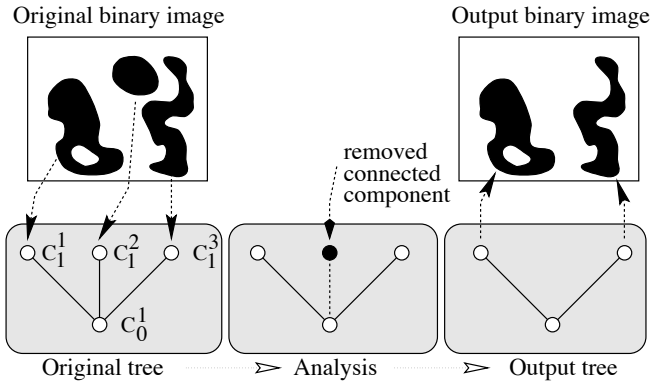

Figure 1: Binary connected operator

components of the image. In this representation, the filtering process consists in analyzing each node C_k^1 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 [6]). 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 figure 1, the node C_1^2 is removed because its area is small and its pixels are moved to the background node C_0^1 (the connected component is removed). As it can be seen, the tree links represent the pixels migration (towards the father) when a node is removed.

Note that his process leads to a simplification of the image (some connected components are removed) as well as to a perfect preservation of the remaining components. All anti-extensive binary connected operators can be described by this process; the only modification is the criterion that is assessed.

2.2 Gray level connected operators

The extension of connected operators to gray-level images can be done via the notion of *flat zone* and the corresponding partition [5]. We present now intuitively this extension by a simple generalization of the tree representation to the gray-level case. The idea consists in creating recursively the tree by a study of thresholded versions of the image at all possible gray levels. An example is presented in figure 2. The original image is composed of seven *flat zones* identified by a letter $\{A, B, C, D, E, F\}$. 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 lowest gray level value, in this case 0. The image is binarized: all the flat zones at level $h = 0$, that is region A, are assigned to the root node of the tree $C_0^1 = \{A\}$. Furthermore, the flat zones with gray level value strictly higher than h form two connected components: $C_1^1 = \{G\}$ and $C_1^2 = \{B, C, D, E, F\}$. 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 $C_{h=1}^k$ is processed as the original image: consider, for instance, the node $C_1^2 = \{B, C, D, E, F\}$. All flat zones belonging to this node with gray level $h = 1$ remain assigned to this node. However, the flat zones with gray level strictly higher than h (here $\{E, C\}$) create two different connected components and are moved to two child nodes: $C_2^2 = \{C\}$ and $C_2^3 = \{E\}$. 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 node C_h^k, a "local" background is defined by keeping all flat zones of gray level value equal to h and that the various connected components formed by the flat zones of gray

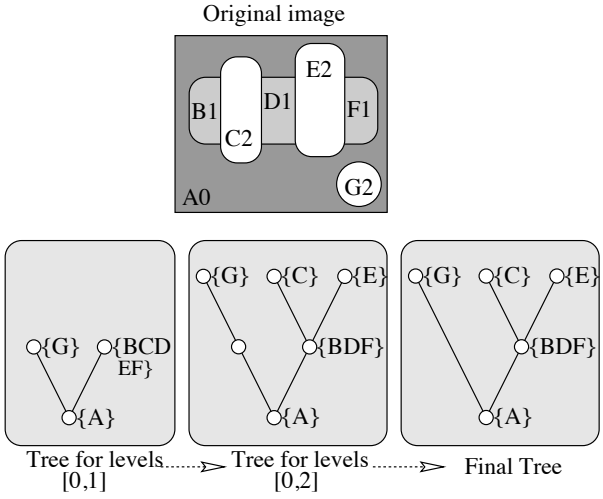

Figure 2: Max-Tree creation

level value higher than h create the child nodes of the tree.

Note that in this procedure, some nodes may become empty. Empty nodes do not give us additional information. Therefore, at the end of the tree construction, they are removed. The final tree is called a *Max-Tree* in the sense that it is a structured representation oriented towards the regional maxima of the image (maxima are simply the leaves of the tree) and towards the implementation of anti-extensive connected operators. By duality, the *Min-Tree* 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, the decision process takes a decision on each node preserving or removing it. As examples, let us briefly recall some classical criteria used for the *opening by reconstruction*, the *area opening* and the λ-*max operator*. We'll denote by $\overline{C}_h^k$ the set of pixels belonging to C_h^k and all its descendant nodes.

- *Opening by reconstruction*: this filter preserves a node C_h^k if the binary erosion of the set of pixels included in $\overline{C}_h^k$ 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 that the structuring element.

- *Gray level area opening* [6]: this filter has been used in section 2.1. It is similar to the previous one except that it preserves the C_h^k if the number of pixels of $\overline{C}_h^k$ is larger or equal to a limit λ. It has also a size simplification effect, but the notion of size is different from the one used in the *opening by reconstruction*.

- λ-*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 $h + \lambda$ or greater ($\exists k'$, $\exists h' \geq h + \lambda$ such that $\overline{C}_{h'}^{k'} \cap \overline{C}_h^k \neq \emptyset$). 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 $\lambda - max$ is an operator but not a morphological filter because it is not idempotent.

These filters correspond to increasing operators, i.e. $\forall x \leq y \Rightarrow \Psi(x) \leq \Psi(y)$. In the context of the *Max-Tree* representation, this means that if C_h^k is a child node of C_j^m (that is $\overline{C}_h^k \subseteq \overline{C}_j^m$) $\mathcal{M}(C_h^k) \leq \mathcal{M}(C_j^m)$. In this case, all nodes such that $\mathcal{M}(C_h^k) < \lambda$ are removed and are moved to the first ancestor node such that $\mathcal{M}(C_h^k) \geq \lambda$. After 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 value h.

3 Motion Connected Operator

The goal of this section if to present the motion criterion. As will be seen, the criterion is non increasing. This issue will be studied in section 3.2.

3.1 Motion criterion

Denote by $f_t(i,j)$ and image sequence where i and j represent the coordinates of the pixels and t the time instant. Our objective 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.

The sequence processing is performed as follows: each frame is transformed intro its corresponding *Max-Tree* representation and each node C_h^k is analyzed. To check whether or not the component associated to a given node is moving in accordance to the motion field $\{\Delta_i(i,j), \Delta_j(i,j)\}$ a simple solution consists in computing the opposite of the Mean Displaced Frame Difference 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 of the region has to be removed (i.e. in accordance to *area opening* criterion). More formally, the criterion can be expressed as [4]:

$$\mathcal{D}_{f_t}^{f_{t-1}}(\overline{C}_h^k) = -\sum_{i,j\in\overline{C}_h^k} |f_t(i,j) - f_{t-1}(i-\Delta_i, j-\Delta_j)| \Big/ Area\left(\overline{C}_h^k\right) \tag{1}$$

In practice, however, it is not very reliable to state the motion 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 DFD's 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*).

$$\mathcal{M}(C_h^k) = (1-\alpha)\mathcal{D}_{f_t}^{f_{t-1}}(\overline{C}_h^k) + \alpha\mathcal{D}_{f_t}^{\Psi(f_{t-1})}(\overline{C}_h^k) \tag{2}$$

where $0 \leq \alpha \leq 1$. If $\alpha = 0$, the criterion is memoryless, whereas higher values of α ($0 \ll \alpha \leq 1$) 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 ($\alpha \approx 0$), whereas if a more reliable decision involving the observation of a larger number of sequences of frames is necessary, the the system should rely heavily on the recursive part ($0 \ll \alpha \leq 1$).

3.2 Non-increasingness issue

The criterion defined by equation 2 is not increasing. That means that if a region X is included in a region Y, there is a priori no relation between the two measured criterion. The decision (node preservation and elimination) presented for the increasing criterion (called "direct" decision) leads to unstable decisions, that appear as random changes between elimination and preservation of some objects. In this paper, we formulate the decision as an optimization problem: based on the "direct" decision tree, our objective is to find an increasing decision rule that minimizes the cost.

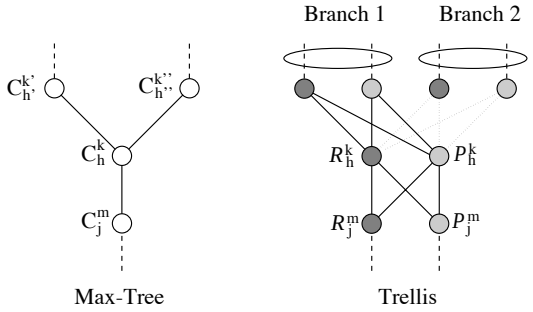

Figure 3: Optimization problem

For each node C_h^k, a binary decision: preserve or remove, has to be taken. Therefore, the first step consists in assigning to each node C_h^k of the *Max-Tree* two states, $\mathcal{P}_h^k$ and $\mathcal{R}_h^k$, describing the two possible decisions. Second, a trellis is constructed by creating transitions linking the possible decisions of one node and of its father. There are four possible transitions between C_h^k and it's father C_j^m (see figure 3): $\mathcal{R}_h^k \to \mathcal{R}_j^m$, $\mathcal{R}_h^k \to \mathcal{P}_j^m$, $\mathcal{P}_h^k \to \mathcal{R}_j^m$ and $\mathcal{P}_h^k \to \mathcal{P}_j^m$. Furthermore, a cost is assigned to each transition. The cost assigned to the transitions ending in the "preserve" state is the same and should reflect the reliability of the decision for that node. This reliability can be measured for example by the difference between the decision threshold and the criterion value, $\lambda - \mathcal{M}(C_j^m)$. For the transition $\mathcal{R}_h^k \to \mathcal{R}_j^m$, the situation is similar and the value $\mathcal{M}(C_j^m) - \lambda$ can be assigned as transition cost. This is however not the case for the transition $\mathcal{P}_h^k \to \mathcal{R}_j^m$: these transitions should be avoided because we want to obtain an increasing decision rule.

The problem of finding the path that minimizes the total cost (the cost of a path is defined as the sum of the costs of its transitions) can be very efficiently solved by the Viterbi algorithm. This algorithm has to be extended in order to deal with trees with various branches: figure 3 shows us the particular case for two branches but the procedure is general and can deal with an arbitrary number of branches. Let us analyze the case of the state $\mathcal{P}_h^k$: there is not one but two optimum paths ending at this state. Note that they are independent from each other. Therefore, we'll apply the Viterbi algorithm for each set of transitions (here identified with solid and dotted lines) and consider their union as the "optimum path" ending in $\mathcal{P}_h^k$. The cost is equal to the sum of the costs of the two paths.

4 Examples and conclusions

The first filtering example is shown in figure 4. Our goal is to apply the motion connected operator in order to remove all moving objects. Therefore, the motion model is defined by $(\Delta_i, \Delta_j) = (0, 0)$. The application of the motion connected operator $\Psi(f)$ described in section 3 removes all bright moving objects (figure 4b). The application of the dual operator $\Psi^*(f) = -\Psi(-f)$ removes all dark moving objects. The result of the composition of the two operator is illustrated in figure 4c. The residue presented in figure 4d show what has been removed by the operator: as it can be seen, the operator has very precisely extracted the ballerina and the (moving) details of the two speakers.

The example illustrated in figure 5 shows a decomposition of the original image into three sequences: objects with a translation of $(\Delta_i, \Delta_j) = (2, 0)$ (figure 5b), still objects $(\Delta_i, \Delta_j) = (0, 0)$ (figure 5c) and the remaining objects (figure 5d). This is a decomposition of the original sequence in the sense that the sum of the tree sequences restores the original sequence.

Motion connected operator can be potentially be used for a large set of applications. It opens the door in particular to different ways of handling motion information. Indeed, generally, motion information is measured without knowing anything about the image structure. *Connected*

a) Original image f | b) Motion operator $\Psi(f)$ | c) Dual operator $\Psi^*(\Psi(f))$ | d) Residue $f - \Psi^*(\Psi(f))$

Figure 4: Extraction of the moving objects using the motion oriented connected operator.

a) Original image | b) Objects with translation (2,0) | c) Objects with translation (0,0) | d) Remaining objects

Figure 5: Motion-oriented sequence decomposition (a = b + c + d)

operators take a different viewpoint by making decisions taking into account the structure of the signal. 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.

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] F.K. Potjer. Region adjacency graphs and connected morphological operators. In P. Maragos, R.W. Schafer, and M.A. Butt, editors, *International Symposium on Mathematical Morphology*, pages 111–118, Atlanta (GA), USA, May 1996. Kluwer Academic Publishers.

[3] 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.

[4] 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.

[5] 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.

[6] 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.