Uniquely-Determined Thinning of the Tie-Zone Watershed Based on Label Frequency

ROMARIC AUDIGIER* AND ROBERTO LOTUFO

School of Electrical and Computer Engineering, State University of Campinas–UNICAMP, C.P. 6101, 13083-852, Campinas (SP), Brazil

audigier@dca.fee.unicamp.br

Abstract. There are many watershed transform algorithms in literature but most of them do not exactly correspond to their respective definition. The solution given by such algorithms depends on their implementation. Others fit with their definition which allows multiple solutions. The solution chosen by such algorithms depends on their implementation too. It is the case of the watershed by image foresting transform that consists in building a forest of trees with minimum path-costs. The recently introduced tie-zone watershed (TZW) has the advantage of not depending on arbitrary implementation choices: it provides a unique and, thereby, unbiased solution. Indeed, the TZW considers all possible solutions of the watershed transform and keeps only the common parts of them as catchment basins whereas parts that differ form a tie zone disputed by many solutions. Although the TZW insures the uniqueness of the solution, it does not prevent from possible large tie zones, sometimes unwanted in segmentation applications. This paper presents a special thinning of the tie zone that leads to a unique solution. Observing all the possible solutions of the watershed by image foresting transform, one can deduce the frequency of the labels associated with each pixel. The thinning consists in assigning the most frequent label while preserving the segmented region connectivity. We demonstrate that the label frequency can be computed both from an immersion-like recursive formula and the proposed fragmented drop paradigm. Finally, we propose an algorithm under the IFT framework that computes the TZW, the label frequency and the thinning simultaneously and without explicit calculation of all the watershed solutions.

Keywords: watershed, image segmentation, mathematical morphology, graph

1. Introduction

The watershed transform (WT)[1] is a famous and powerful segmentation tool in morphological image processing. First introduced by Beucher and Lantuéjoul [5] for contour detection and applied in digital image segmentation by Beucher and Meyer [22], it is inspired from a physical principle well-known in geography: if a drop of water falls on a topographic surface, it will follow the greatest slope until reaching a valley. The set of points which lead to the same valley is called a catchment basin (CB). The

*Corresponding author.

watershed lines separate different catchment basins. In the WT, an image is seen as a topographic surface where grey-level corresponds to altitude. Generally, to segment an image by WT, a gradient of the image is used as topographic surface. In this case, it is expected that a region with low gradient, a valley, corresponds to a rather homogeneous region and possibly to the same object. Ideally, catchment basins correspond to segmented objects separated by watershed lines.

Many definitions for WT exist in litterature. Definitions in the continuous space have been proposed [5, 20, 25, 26] and consider the watershed as a skeleton by influence zones (SKIZ) generalised to greyscale images. Each proposal gives a unique

0	1	1	0

(a) I_1

A	A	B	B

(b) Strict WT definition

A	A	W	B

(c) Line algorithm 1

A	W	B	B

(d) Line algorithm 2

0	1	0

(e) I_2

A	W	B

(f) Strict WT definition

A	A	B

(g) Region algorithm 1

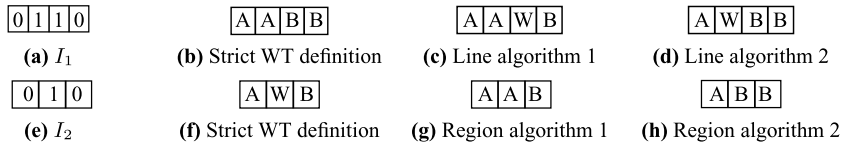| A	B	B

(h) Region algorithm 2

Figure 1. Data processing order introduces a 1-pixel bias. (a)–(d) Input image, WT according to the definition, and two possible labelled WT outputs (raster or anti-raster scan) of a line-algorithm. W represents the watershed line. (e)–(h) Input image, WT according to the definition, and two possible labelled WT outputs (raster or anti-raster scan) of a region-algorithm.

solution for the watershed. In the digital space (of interest in this paper), there are many definitions using different paradigms. We recall some of them.

The recursive definitions of digital WT by "immersion" [28] or "flooding" [6] and the definition of digital WT by "topographic distance" [20] can be seen as digital versions of the generalised SKIZ paradigm. Each definition, if strictly applied, gives a unique solution. But the watershed is not necessarily thin [28]. Moreover, not all CBs are separated by watershed lines. Some can have direct contacts (e.g., see the case of an even plateau in Fig. 1(a) and (b)). This is due to the discretization of a paradigm designed in the continuous space.

The topological watershed proposed by Couprie and Bertrand [10] cannot be viewed as a generalised SKIZ but as the ultimate topological thinning that transforms the image while preserving some topological properties [4, 24]. Furthermore, the separating watershed lines are valued such as the saliency between CBs is preserved [23]. Multiple solutions are allowed by the topological watershed definition and each CB is necessarily separated from the others by a watershed line (cf. Fig. 2(a) and (d)–(f)).

The image foresting transform (IFT), a graph-based framework introduced by Falcão, Lotufo and Stolfi [15], also defines a digital WT. Yet, this WT is made of CBs only (no separating watershed line): it is a "region WT". In this paradigm, the image is seen as a graph where pixels are nodes, and the WT seen as a problem of trees of minimal paths [17]. Indeed, the WT by IFT consists in creating an optimal forest from the image-graph, i.e. a set of trees that have minimum path-costs. These trees correspond to the CBs. Multiple solutions (optimal forests) are allowed by the definition (cf. Fig. 2(a)–(c)).

Next to the many definitions, there are lots of algorithms that compute a digital WT. The WT algorithms can be classified in two types: (i) *line-algorithms* that return watershed lines separating each CB from the others [11, 12, 28]; (ii) *region-algorithms* that return labelled regions (the CBs) which constitute a partition of the image without any watershed pixel [6–9, 15, 19, 20, 27]

Yet, many algorithms do not correspond to their respective digital definition. For example, the Vincent-Soille's algorithm [28] imposes lines between CBs so

1	1	1	1	1	1	1
1	20	20	20	20	20	1
1	20	16	14	17	20	1
30	30	14	13	15	30	30
2	30	13	12	14	30	3
2	30	30	11	30	30	3
2	2	2	10	3	3	3

(a) I_3

A	A	A	A	A	A	A
A	A	A	A	A	A	A
A	A	B	B	B	A	A
A	B	B	B	B	B	C
B	B	B	B	B	C	C
B	B	B	B	C	C	C
B	B	B	B	C	C	C

(b) WT by IFT 1

A	A	A	A	A	A	A
A	A	A	A	A	A	A
A	A	C	C	C	A	A
B	C	C	C	C	C	C
B	B	C	C	C	C	C
B	B	B	C	C	C	C
B	B	B	C	C	C	C

(c) WT by IFT 2

1	1	1	1	1	1	1
1	1	20	20	20	1	1
1	20	3	3	3	20	1
20	3	3	3	3	3	20
2	10	3	3	3	3	3
2	2	10	3	3	3	3
2	2	2	10	3	3	3

(d) Topological WT 1

1	1	1	1	1	1	1
1	1	20	20	20	1	1
1	20	2	10	3	20	1
20	2	2	10	3	3	20
2	2	2	10	3	3	3
2	2	2	10	3	3	3
2	2	2	10	3	3	3

(e) Topological WT 2

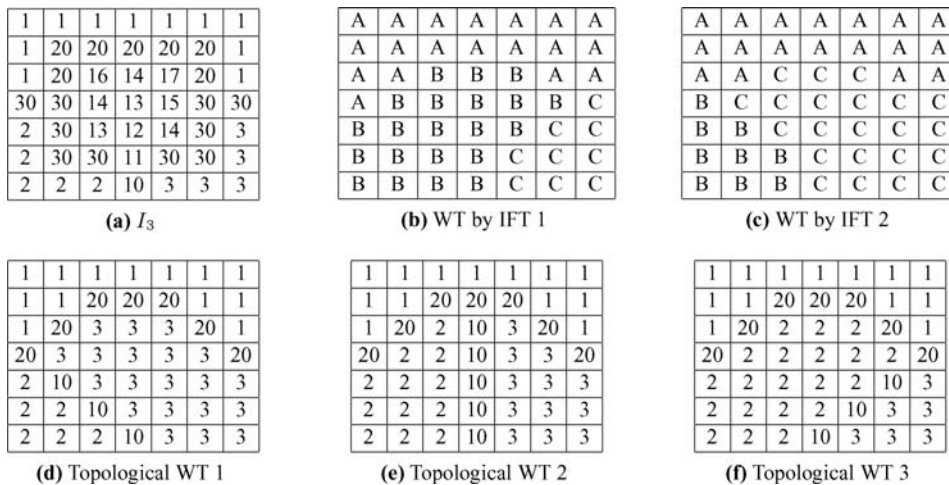| 1	1	1	1	1	1	1
1	1	20	20	20	1	1
1	20	2	2	2	20	1
20	2	2	2	2	2	20
2	2	2	2	2	10	3
2	2	2	2	10	3	3
2	2	2	10	3	3	3

(f) Topological WT 3

Figure 2. Multiple solutions do not contradict the definition but are consequences of specific implementations. (a) Input greyscale image with 3 minima. (b) and (c) Possible label maps of the WT by IFT using 4-adjacency (raster or anti-raster scan respectively). (d)–(f) Possible topological watersheds for different data processing orders. Minima's grey-levels are extended in CBs and watershed lines are valued.
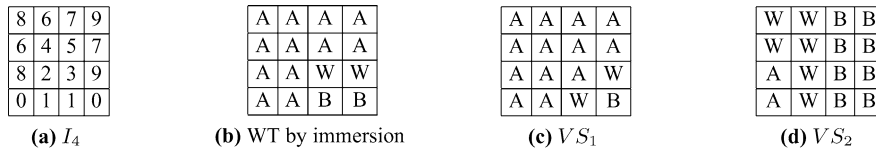
8	6	7	9
6	4	5	7
8	2	3	9
0	1	1	0

(a) I_4

A	A	A	A
A	A	A	A
A	A	W	W
A	A	B	B

(b) WT by immersion

A	A	A	A
A	A	A	A
A	A	A	W
A	A	W	B

(c) VS_1

W	W	B	B
W	W	B	B
A	W	B	B
A	W	B	B

(d) VS_2

Figure 3. (a) Input greyscale image with 2 minima. (b) Solution given by the strict definition of WT by immersion: watershed line is not separating. (c) and (d) Possible watershed lines (W) by Vincent and Soille's algorithm (raster or anti-raster scan respectively) using 4-adjacency. Multiple solutions are not predicted by the definition but are consequences of the bias introduced by implementation.

that it does not fit with any definition and many solutions are possible depending on the implementation (cf. Figs. 1(c)–(d) and 3). On the contrary, the algorithms presented by Meyer in ref. [20] "do not produce watershed labels and, therefore, are not exact implementations of the definition. All pixels are merged with some basin, so that, dependent on the order in which pixels are treated, different results may be produced" [27]. It is clear (see Fig. 1) that, by adapting the definitions to adequate with the output type constraint (lines or regions), these algorithms introduce a bias in the output according to the processing order (implementation choice). There is a lack of robustness of the result due to the implementation.

Other algorithms, like the topological watershed (line) and the watershed by IFT (region), fit with their respective definitions but give only one solution among the multiple solutions allowed, wich depends on the (arbitrary) implementation. Therefore, they can give different solutions depending on the processing order (cf. Fig. 2). Again, there is a lack of segmentation robustness due to the implementation but it does not conflict with the definitions.

In summary, the dichotomy between line and region algorithms is incompatible with most of the proposed digital definitions. And when it is compatible, the WT is not uniquely defined.

In this paper, we focus on the WT by image foresting transform (IFT). By returning only one of the multiple solutions, the algorithm makes necessarily an arbitrary choice. The returned solution is, therefore, biased by the algorithm implementation. This variation due to implementation can be insignificant in some cases (1-pixel bias for the line or region position, see Fig. 1(g) and (h)) but in other cases, it becomes considerable and even may be unacceptable for some applications (e.g., reliable measures on segmented objects). In some images an entire region is reached passing by a bottleneck pixel and consequently included to the first (or other arbitrary) CB that invades the bottleneck (like in Fig. 2(b) and (c)). Notice that the problem does not occur only on plateaux but also in the buttonhole configurations: regions invaded by passing through a bottleneck whose

merging with a CB or a watershed line is affected by the 1-pixel bias depicted in Fig. 1. It is important to remark that buttonhole cases "correspond to special pixel configurations which are not so rare in practice" as reported by ref. [23, 28].

How to provide a unique solution for the WT by IFT independent of the implementation? This question motivated the recent introduction of the tie-zone watershed (TZW). Roughly speaking, the definition of TZW [2] takes into account all the possible solutions (optimal forests) derived from the strict WT definition to generate a unique solution: when the multiple solutions disagree with each other on the segmentation result of a region (i.e., the label to be assigned), the region is included to the tie-zone (TZ) and a specific tie label is assigned. The TZ is therefore a litigious zone. The TZW avoids to make arbitrary decisions.

The size and distribution of the TZ can be used as a tool that gives a measure of the non-robustness or unreliability of a segmentation in relation to the results given by other implementations of IFT-WS. In other words, the size of the TZ indicates how different could be the segmentation if another implementation were used [1].

For a segmentation purpose, the possibly large TZ may be unwanted. As a large TZ indicates a poor reliability of the segmentation, the user may add some extra markers (see ref. [6, 22] for the concept of marker) to modify the segmentation and try to untie the TZ, and then increase the reliability of the segmentation. Observe that a reliable or robust solution is not necessarily a semantically correct solution. It is reliable in relation to the segmentation paradigm that was defined: in this case, the optimality of the forest.

An alternative to extra markers is to apply an automatic thinning on the TZ that assigns the TZ to already segmented regions (CBs). The TZ thinning must be unique and independent of the implementation, otherwise it is not worth having firstly applied the TZW. However, there is still a trade-off between uniqueness of the solution and thinness of the TZs, so that perhaps only part of the TZ can be thinned to preserve the uniqueness of the solution.

A first proposal of TZ thinning was introduced in ref. [2]. It consists in applying iteratively the TZW on a topography whose altitude is the number of different labels that tied together. Consequently, a distance criterion indirectly unties the TZ. This thinning drastically simplifies the topography of the original TZ.

In addition, an accidental leak of water in watershed can create a non-representative TZ. For example, if almost all the WT solutions agree in assigning a region to a same CB but only one WT solution assigns it to another one, the region becomes a TZ, equally disputed by both CBs. Thus, one solution has as much weight as all the others. On the contrary, it could be desirable that all the solutions have the same importance.

We propose in this paper a thinning of the TZ keeping the property of uniqueness and based on the label frequency: observing all the possible watershed solutions by IFT, the TZ can be untied when a region is most frequently assigned to a specific CB. Considering the drop of water paradigm, the TZ corresponds to regions where a drop of water could fall and follow several ways and slide down to different valleys. Imagine now that every time the drop of water can follow several ways, it is split in smaller equal amounts of water which follow the possible ways. The fractions of water the valleys receive are evaluated. At the end, the greatest fraction determines which CB the drop belongs to, as long as the CB connectivity is preserved, i.e., if there exists a path linking the locus the drop fell to the valley, entirely included in the CB. We demonstrate that the label frequency used by the thinning of TZW can be computed both from the fragmented drop paradigm and an immersion-based formula.

This paper is organised as follows. The notation and definitions on graphs necessary to understand the IFT framework are introduced in Section 2. Then, Section 3 gives an overview of TZW and recalls the watermerging paradigm that sustains the idea of tie-zones. The multipredecessor optimal graph, a special graph describing the water flows and mergings, is defined too. Finally, Section 4 deals with the several ways of determining the label frequency, defines the thinning of the TZ based on label frequency, and presents an algorithm to compute at the same time the TZW and its frequency-based thinning.

2. Watershed Transform by Image Foresting Transform

In this section, the notation and definitions for the watershed by image foresting transform (IFT) are recalled.

The IFT is a general framework based on graph theory in which an image is interpreted as a graph and pixels as its nodes. The key idea of this transform is to obtain, according to a path-cost function, a *shortest path forest* from an input image-graph. Depending on the path-cost function and some other input parameters (adjacency, arc weights), the IFT can compute different image processing operations [14, 15]: distance transforms, connected filters, interactive object delineation ("live-wire"), segmentation by fuzzy connectedness and segmentation by watershed.

2.1. Notation and Definitions

Under the IFT framework, an image is seen as a weighted graph $G = (V, A, I)$ where each pixel (or voxel in 3D) is represented by a node or vertex $v \in V$ with intensity $I(v)$. For digital image, I is a map from V to $\mathbb{Z}$. An arc $\langle u, v \rangle \in A$ exists between vertices u and v when the corresponding pixels are adjacent according to the defined adjacency (usually 4- or 8-adjacency in 2D and 6- or 26-adjacency in 3D). A path $\pi(u, v)$ from a node u to a node v in a graph (V, A, I) is a sequence $\langle u = v_1, v_2, \ldots, v_n = v \rangle$ of nodes of V such that $\forall i = 1 \ldots n - 1$, $\langle v_i, v_{i+1} \rangle \in A$. A path is said simple if all its nodes are different from each other. Let $S \subseteq V$ be a set of particular nodes s_i called seeds. The graph $G' = (V', A')$ is a subgraph of G if $V' \subseteq V$, $A' \subseteq A$ and $A' \subseteq V' \times V'$. A directed forest F of G is a directed acyclic subgraph F of G. A tree of the forest F is a connected component of F.

For a given weighted graph $G = (V, A, I)$ and a set $S = \{s_i\}$ of seeds, the *image foresting transform* (IFT) returns an *optimal forest*, i.e. a directed forest F of G such that (i) there exists for each node $v \in V$ a unique and directed simple path $\pi(s_i, v)$ in F from a seed node $s_i \in S$ to v and (ii) each such path has a minimum (or "optimum") cost for linking v to any seed of S, according to a specified path-cost function f_C.

Assume that the arcs $\langle u, v \rangle$ are weighted with the grey-level $I[v]$ of the pixel corresponding to v. Assume that the seed nodes correspond to the regional minima of the image (or to imposed minima, i.e. markers [6]). If the path-cost function f_C is defined as the 'maximum arc' function $f_{\max}$,

$$f_{\max}(\langle v_1, v_2, \ldots, v_n \rangle) = \max\{h(v_1), I(v_2), \ldots, I(v_n)\}$$

where h is a fixed but arbitrary handicap cost [18], the IFT returns a region-WT where the trees of the forest
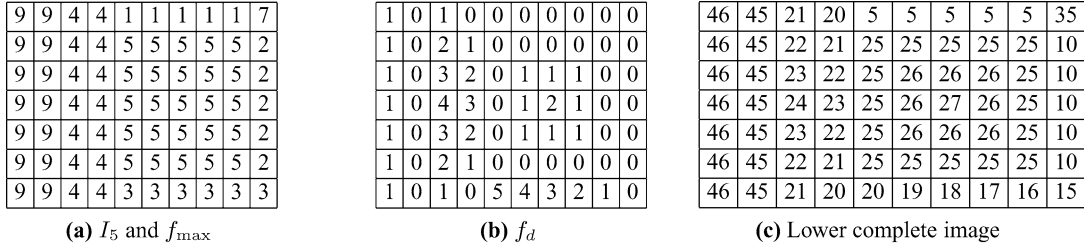
9	9	4	4	1	1	1	1	1	7
9	9	4	4	5	5	5	5	5	2
9	9	4	4	5	5	5	5	5	2
9	9	4	4	5	5	5	5	5	2
9	9	4	4	5	5	5	5	5	2
9	9	4	4	5	5	5	5	5	2
9	9	4	4	3	3	3	3	3	3

(a) I_5 and $f_{\max}$

1	0	1	0	0	0	0	0	0	0
1	0	2	1	0	0	0	0	0	0
1	0	3	2	0	1	1	1	0	0
1	0	4	3	0	1	2	1	0	0
1	0	3	2	0	1	1	1	0	0
1	0	2	1	0	0	0	0	0	0
1	0	1	0	5	4	3	2	1	0

(b) f_d

46	45	21	20	5	5	5	5	5	35
46	45	22	21	25	25	25	25	25	10
46	45	23	22	25	26	26	26	25	10
46	45	24	23	25	26	27	26	25	10
46	45	23	22	25	26	26	26	25	10
46	45	22	21	25	25	25	25	25	10
46	45	21	20	20	19	18	17	16	15

(c) Lower complete image

Figure 4. (a) Input greyscale image I_5 with 2 minima is equal to the first component ($f_{\max}$) of the lexicographic cost. (b) The second component (f_d) of the lexicographic cost corresponds to the geodesic distance to the lower border. (c) The lower complete image. Lower completion increases the image grey-level range.

correspond to the CBs. Note that all vertices (pixels) are covered by this forest. The handicap cost is generally set to $I(v_1)$ or 0 in case of minima imposition [6].

The IFT can result in many optimal forests because many paths of (same) minimum cost are sometimes possible for some nodes. The set of all the optimal forests F is denoted by Φ. Observe that a forest can be simply represented by a predecessor map P where $P[v]$ denotes the predecessor of node v in the minimum path. In addition, a cost map C can indicate for each node v the cost $C[v]$ of the minimum path from the tree root to v. When $f_C = f_{\max}$, it corresponds exactly to the morphological superior-reconstruction [15] of the input image I from the specified seeds (natural or imposed minima). For segmentation purpose (WT by IFT), a label map L is generally associated with the optimal forest, so that, for each node v, $L[v]$ represents the label of the corresponding minimum-path root. Notice that the final cost map C is unique (values of the minimum paths are unique) while the predecessor map P and then the labelling L may be multiple.

In ref. [17], a two-component lexicographic cost function f_{LC} was proposed to mimic the flooding process and handle with plateaux too: $f_{LC} = (f_{\max}, f_d)$. The first component, of highest priority, is the max-arc function and represents the flooding process. The second one corresponds to the geodesic distance to the lower boundary of the plateau and makes different waters propagate on plateau at a same speed rate:

$$f_d(\langle v_1, v_2, \ldots, v_n \rangle) = \max_{k \in [0, n-1]} \{k, C[v_n] = C[v_{n-k}]\}$$
$$C[v_n] = f_{\max}(\langle v_1, v_2, \ldots, v_n \rangle)$$

The use of the lexicographic cost provides partitions that seem to be more equitable (on plateaux) than those obtained with the maximum cost component only. In fact, the lexicographic cost avoids a prior lower completion on image with plateaux but has strictly the same role.

In Fig. 4(a), a greyscale image with two minima and many plateaux is showed. Figures 4(a) and (b) show the two cost maps corresponding to the two lexicographic cost components returned by the IFT. The first component ($f_{\max}$) always corresponds to the reconstruction of the input image from the specified seeds (natural or imposed minima). In this case, all the regional minima are considered seeds, so the reconstruction is the input image itself. The second component (f_d) corresponds exactly to the geodesic distance (minus one) to the lower boundary of the plateau. With these geodesic distances on plateaux, it is possible to transform (see Definition 3.4 of ref. [27]) the original input image to a lower complete image (see Fig. 4(c)), as required in many algorithms (e.g., hill climbing, topographic distance).

2.2. Algorithms for IFT

The algorithm for IFT computes three attributes for each vertex $v \in V$: its predecessor $P[v]$ in the minimum path, the cost $C[v]$ ($C[v] = (C_1[v], C_2[v])$ when the lexicographic cost is used) of that path, and the corresponding root label $L[v]$.

The efficient ordered queue-based algorithm for IFT proposed in ref. [15, 17] is essentially Dijkstra's algorithm [13], extended for multiple sources and a more general path-cost function. It is denoted Dijkstra-IFT in this paper. Note that, for the 4- or 8-adjacency in 2D or 6- or 26-adjacency in 3D, the ordered queue can be implemented such that the IFT algorithm will run in time proportional to the number of vertices [15].

The lexicographic path-cost is very simple to compute: only the first component (maximum arc) is explicitly computed. The second component is implicitly computed by using a priority first-in-first-out (FIFO) queue [17]. As we said, the use of this lexicographic path-cost substitutes the lower completion step. Observe that the explicit lower completion as presented in

Fig. 4(c) increases the range of the image grey-levels. So, in addition to the computation cost of the lower completion step, there may be, in some images, an extra storage cost.

Note also that other algorithms are able to compute the IFT. For example, the ordered queue is not necessary. One can process the image data in raster-scan and anti raster-scan order alternatively until stability of the result (algorithm not presented here and based on Berge's one [3], cited in Section 4.3 of ref. [20]).

3. Optimal Forest Paradigm and Tie-Zone Watershed

We recall in this section the main definitions and concepts on tie-zone watershed introduced in ref. [2].

3.1. The Tie-Zone Watershed (TZW) Transform

3.1.1. Definition.   As we saw in the previous section, many optimal forests and so, many partitions may correspond to an input image. We propose then a new definition of WT in the IFT context which results in a unique partition, i.e. a unique label map.

A node is included in a specific catchment basin CB_i when it is linked by a path to a same seed s_i in all the optimal forests, otherwise it is included in the Tie-Zone T:

$$CB_i = \{v \in V, \quad \forall F \in \Phi, \ \exists \pi(s_i, v) \text{ in } F\}$$

$$T = V \setminus \bigcup_i CB_i$$

If a node is in the tie-zone, it means that it could be included in different CBs without affecting the forest optimality. CBs are only the common part of all optimal solutions whereas differing parts are considered TZ. Therefore, the tie-zone existence prevents from making any arbitrary choice between optimal solutions. Consequently, the TZW solution is defined without ambiguity.

Note that this definition does not produce watershed lines but only regions: catchment basins and tie-zone. They form together a unique optimal partition of the image. If all pixels are assigned to catchment basins, the tie-zone will be empty. This situation can occur when the lexicographic path-cost function unties growing CBs on plateaus (e.g., Fig. 1(b) also corresponds to the TZW of image I_1 but has no tie-zone). So, this watershed transform possibly does not contain any tie-zone. In some cases, the TZ can be quite large (see Fig. 5).
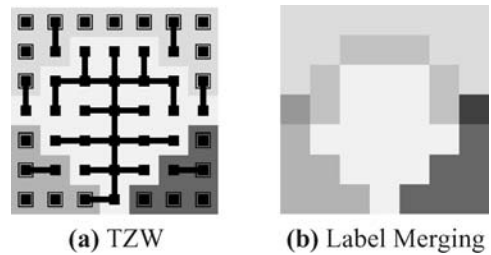

Figure 5.   (a) TZW applied on image I_3 of Fig. 2 using 4-adjacency: 3 CBs (grey), TZ (white), a possible forest (black). (b) Result of the Label Merging algorithm.

Unlike in the WT by IFT where each CB corresponds to entire trees with same label, in the TZW by IFT, CBs may correspond to rooted parts of trees while the TZ is composed of many terminal parts of trees as in the example of Fig. 5(a).

3.1.2. Dijkstra-Based Algorithm.   Now, we transcribe an efficient algorithm that labels the image in order to obtain a TZW [2]. It is based on Dijkstra's shortest path algorithm [13] and utilises an ordered queue Q where each bucket has a FIFO policy.

The algorithm input is: the image as a weighted graph $G = (V, A, I)$, the seed set S with associated labelling function λ and handicap function h. We denote the neighbourhood of a node $p \in V$ by: $N_G(p) = \{q \in V, \langle p, q \rangle \in A\}$.

In the output, we have the label map L corresponding to the TZW result, map P giving each node's predecessor in the tree and maps (C_1, C_2) giving the lexicographic cost of an optimal path from a seed to each node. Note that, unlike the algorithm for IFT [15], the second component C_2 of the lexicographic cost is not intrinsically computed by the FIFO policy and must be explicit in the TZW by IFT in order to prevent 1-pixel bias.

The priority queue Q is initially empty. DequeueMin removes from Q the node of minimum cost and returns it; Enqueue(p, c) inserts node p in Q at priority (cost) c bucket. QueueNotEmpty indicates that the queue is not empty. The state flag $done(p)$ is TRUE when the node has already been processed, i.e. it has its definitive attributes.

Algorithm 1: Dijkstra-based TZW with lexicographic path-cost.

Inputs:	image (V, A, I), neighbourhood N_G (derived from A), seeds S, handicap h and labelling λ functions.

Outputs: label L, predecessor P and cost (C_1, C_2)
                maps.
Auxiliary Data: empty ordered queue Q, state flag *done*,
                cost variable c.

1. $\forall p \in V, \ C_2[p] \leftarrow 0; \quad done(p) \leftarrow$ FALSE;
2. $\forall p \notin S, \ C_1[p] \leftarrow \infty; \quad L[p] \leftarrow$ NIL; $\ P[p] \leftarrow$ NIL;
3. $\forall p \in S, \ C_1[p] \leftarrow h(p); L[p] \leftarrow \lambda(p); P[p] \leftarrow p$;
    Enqueue$(p, h(p))$;

4. **while** QueueNotEmpty,
5. $v \leftarrow$ DequeueMin; $done(v) \leftarrow$ TRUE;
6. $\forall p \in N_G(v)$ **and** $done(p) =$ FALSE,
7. $c \leftarrow \max\{C_1[v], I[p]\}$;
8.   **if** $c < C_1[p]$,
9.   **if** p in Q, Dequeue(p);
10. $C_1[p] \leftarrow c; L[p] \leftarrow L[v]; P[p] \leftarrow v$;
11.   Enqueue$(p, C_1[p])$;
12.   **if** $c = C_1[v], C_2[p] \leftarrow C_2[v] + 1$;
13.   **else**, **if** $c = C_1[p]$ **and** $L[p] \neq L[v]$,
14.   **if** $c = C_1[v]$,
15.   **if** $C_2[p] = C_2[v] + 1, L[p] \leftarrow$ TZ;
16.   **else** $L[p] \leftarrow$ TZ;

The beginning of the algorithm (lines 1–11) is identical to the IFT algorithm in ref. [15] based on Dijkstra's. Lines 12–16 are TZW-specific (line numbers are bold-faced). After cost, label and predecessor initialisations (*l.* 1–3), a loop for emptying the priority queue Q starts (*l.* 4–16). This ordered queue has firstly been filled with seed nodes, properly labelled and with their respective handicap cost (*l.* 3). The node v of highest priority, i.e. lowest cost, is removed from Q (*l.* 5) with its definitive attributes (cost $C[v]$, label $L[v]$, and predecessor $P[v]$). This indicates that the minimum path $\pi(s_i, v)$ from some seed $s_i \in S$ to the node v has already be found. For each node p neighbour of v, such that p has not been definitively processed, the cost c of a candidate path with terminus p passing by v is evaluated (*l.* 6–7). If c is lower than the already assigned cost $C_1[p]$ (*l.* 8), then the path to p passing by v is considered better (cheaper) than the current path that reaches p and the three attributes of p are updated (*l.* 10). If p has never been visited, i.e. p is not in Q, it is inserted in Q with cost c (*l.* 11). Otherwise, only its position in Q is updated (*l.* 9,11).

In line 12, the second component of lexicographic cost is incremented, as water propagates on plateau. Lines 13–16 detect the nodes p where paths from (at least) two seeds with different labels ($L[p] \neq L[v]$) tie together, i.e. have same costs (C_1, C_2). The special label TZ is assigned to such nodes.

Note that the algorithm is efficient because it has the same complexity as the algorithm of ref. [15] that computes a simple WT by IFT, and it is not necessary to compute explicitly all the WT solutions to obtain the TZW. The solution of TZW, based on IFT, is optimal because it keeps therefore the optimality of the shortest-path forest solution as demonstrated in ref. [15, 17]. In addition, label map L cannot be biased by arbitrary processing order in queue remotion nor neighbour selection.

The TZW can also be obtained without using an ordered queue by processing the image data in an unordered way, for example by sequential forward and backward scannings alternatively, until stability of the result (algorithm not presented here and based on Berge's one [3, 20]).

The area of the TZs, their distribution and number and distribution of their sources, the so-called bottlenecks, can be correlated with the robustness of a segmentation, i.e. with the degree of confidence a particular segmentation by WT has [1].

The definition of TZW can be extended to other WT definitions: similarly, all solutions have to be taken into account and the regions where labelling differences occur constitute the tie-zone.

3.2. Watermerging Paradigm

The WT is frequently compared to the flooding of a topography where dams are built to prevent distinct coloured waters from merging, supposing a colour is assigned to each minimum or marker [6, 28]. The watermerging paradigm does not build such dams. For an intuitive comprehension of this paradigm, flip up-to-down the topography representing the image like in Fig. 6. Imagine that each marker (former minimum that is now regional maximum) is a source of coloured water. When coloured waters meet together, no dam is
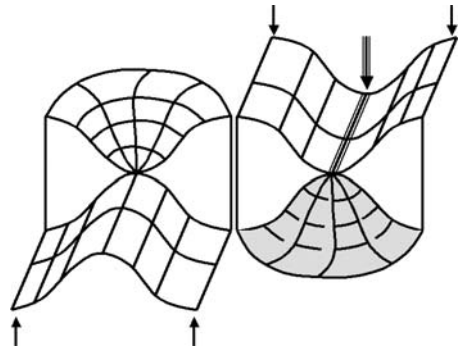

Figure 6. Watermerging paradigm. On the left (right), the arrows point the 2 minima (maxima) of the topography. The triple line and the grey zone constitute the tie-zone where waters merged.

built but the coloured waters naturally merge into a water of blended colour. Holes are punched in the regional minima (former maxima) for draining the water. Supposing waters propagate along all negative slopes (not only the steepest as occurs in reality), we get coloured hills and possibly regions with blended colours. The blended regions correspond to the tie-zone.

This intuitive up-to-down transformation is actually not used in implementations. When label *a* and label *b* tie together in a region, a merged-label $\{a, b\}$ is assigned to this region. The tie-zone is therefore differentiated in several regions according to the labels that could be assigned to them. The Label Merging algorithm is a useful variation of the previous Algorithm 1. Substitute TZ label on lines 15–16 by `MergeLabels`$(\mathbb{L}[p], \mathbb{L}[v])$ and the simple label map L by a merged-label map $\mathbb{L}$ that returns a set of labels for each pixel. There is no more one TZ label but so many as the distinct label mergings (four in example of Fig. 5(b)).

3.3. *Multipredecessor Optimal Graph*

We introduce now a special graph, unique for each image, that will be used in Section 4.

Roughly speaking, the multipredecessor optimal graph (MOG) $\Gamma = (V, A^*)$ of a weighted gaph $G = (V, A, I)$ is the "union" of all the optimal forests $F \in \Phi$. More precisely, it is a directed acyclic subgraph of G such as its arc set A^* is the set of the optimal oriented arcs, i.e. the union of the (oriented) arcs of all optimal forests $F \in \Phi$.

$$\Gamma = (V, A^*) = \left(V, \bigcup_{\forall F = (V, A') \in \Phi} A' \right)$$

Note that the graph is unique but is not necessarily a forest. Once we have a lexicographical cost map of the image, i.e. its reconstruction from the seeds and the distance to the border of plateaux, the following local property is valid. A node p is predecessor of a node v in the MOG if and only if p is a neighbour of v such that its lexicographic cost $C_L[p] = (C_1[p], C_2[p])$ is strictly lower than the lexicographic cost $C_L[v] = (C_1[v], C_2[v])$ of v:

$$p \in \mathbb{P}[v] \Leftrightarrow p \in N(v), \quad C_L[p] \prec C_L[v].$$

where $\mathbb{P}[v]$ denotes the set of predecessors of node v.

The number of predecessors by node is no longer restricted to one. But it is bounded by the maximum number of neighbours by node, given by the adjacency definition (e.g., four in the example of Fig. 7(c)). Furthermore, the number of connected components of the MOG no longer corresponds to the number of seeds as in the case of optimal forest of trees.

An analogy can be observed between the MOG and the 'lower complete graph' (Definition 3.5 of ref. [27]). Both are directed acyclic graphs. In the former case, the lower neighbours in the lexicographic cost map are predecessors of the nodes. In the latter case, only the steepest lower neighbours in the lower complete image are predecessors of the nodes.

4. Thinning of the Tie-Zone Based on Label Frequency

As said in Section 1, thin lines (or no line) between segmented regions can be wished. It is why a thinning of the TZ can be useful when the user does not want to add more markers (seeds) to untie a large TZ. Other criteria than path-cost have to untie the TZ to assign this region to existing CBs while preserving the uniqueness property of the TZW. Indeed, the TZ thinning must be unique and independent of the implementation, otherwise it is not worth having firstly applied the TZW. However, as we saw in Section 1, there is still a trade-off between uniqueness of the solution and thinness of the TZs, so that perhaps only part of the TZ can be thinned to preserve the uniqueness of the solution.

A first proposal of TZ thinning was introduced in ref. [2]. It consists in applying iteratively the TZW on a topography whose altitude is the number of different labels that tied together. More labels tied in a region, more disputed it is and higher its altitude is. The topography change creates plateaux with altitude proportional to the number of CBs disputing the tie-zone. By applying such a topography change, the original image is drastically simplified no matter which labels tied and how many times they dispute this region. For example, a CB label may dispute a TZ even if there is only one path reaching the TZ while labels from other CBs have much more paths leading to the same TZ. Consequently, a distance criterion unties the TZ in the next iteration(s) because of the use of lexicographic cost on the created plateaux. One can believe that a unique path to a TZ could be an accidental "leak of water" creating a non-representative TZ. At least, this path has certainly not the same weight as the others in the TZ formation.

This argument motivated the TZ thinning proposed in this section. Indeed, a TZ is created when at least an

optimal forest does not assign the same label to a pixel, no matter how many optimal forests are discording with this labelling. Thus, one optimal forest might have as much weight as all the others. We propose to assign the same importance to each optimal forest and keep the most frequent labelling.

Remember that any optimal forest is a solution of the WT by IFT, and is not a biased solution in itself. But the arbitrary choice of one optimal forest would be equivalent to ignoring all the other possibilities of path optimisation in the image-graph. This simplistic choice constitutes a bias. Similarly, giving an equal importance to every label responsible for a TZ would be equivalent to ignoring that perhaps a certain CB is "more frequently linked" to a TZ than other CBs. And this fact reflects the specificity of the image topography, information that should be taken into account.

The TZ thinning proposed here keeps the property of uniqueness and is based on the label frequency: observing all the possible watershed solutions, the TZ can be untied when a region is most frequently assigned to a specific CB. Each optimal forest has the same relevance. Considering the collection of all these possible realisations of the WT, one can deduce the relative frequency for a pixel to be included in a particular CB and finally assign to it the most frequent CB label.

Considering the drop of water paradigm, the TZ corresponds to regions where a drop of water could fall and follow several ways and slide down to different valleys. Imagine now that every time the drop of water can follow several ways, it is split in smaller equal amounts of water which follow the possible ways. We demonstrate that the fractions of water the valleys receive correspond to the respective label frequencies.

Section 4.1 demonstrates how relative label frequencies can be computed, by the fragmented drop paradigm (Section 4.1.1) or recursively (Section 4.1.2). Section 4.2 explicits the way the most frequent labelling is obtained and special care to take for preserving uniqueness and consistency of the segmentation. Section 4.3 presents an algorithm derived from Dijkstra-IFT-TZW algorithm (Algorithm 1) that computes the TZW and the label frequency-based thinning simultaneously.

4.1. Computing Label Frequency

Let $G = (V, A, I)$ be the weighted graph corresponding to an image. Let S be the set of seed nodes (pixels). Let $\Gamma = (V, A^*)$ be the multipredecessor optimal graph (MOG) of G. We denote by $\mathbb{P}(p)$ the set of predecessors of node p in the MOG.

Remember that only one optimal cost map C can result from the IFT of a weighted graph $G = (V, A, I)$. But many predecessor maps, the optimal forests F, can exist and each one is a support for a (distinct or not) label map L when labels are associated with seeds. We wish to find the map of the most frequent labels. We do not search for the most frequent optimal forest (as each distinct optimal forest is observed only once), nor for the forest built with the set of most frequent arcs.

First, we have to compute all the optimal forests $F \in \Phi$ and associate with each pixel the relative frequency of each label. In a final step, the most frequent label will be assigned to each pixel (Section 4.2). As we will compute the labels' relative frequency for each pixel considering a collection of optimal forests, we have to know how many optimal forests F exist. Thus, the problem is to count how many optimal forests can derive from Γ. To build an optimal forest from the MOG, we have to choose one and only one predecessor per node except for the optimal tree roots that have no predecessor. As setting a specific predecessor among $\mathbb{P}(p)$ for a node p does not discard choices of predecessor for any other node in the graph, choices of node predecessor are independent events. Therefore, we have $|\Phi|$ possible optimal forests:

$$|\Phi| = \prod_{p \in V, \ |\mathbb{P}(p)| \neq 0} |\mathbb{P}(p)|$$

The condition $|\mathbb{P}(p)| \neq 0$ is necessary to exclude the tree roots.

Notice that in the label frequency-based thinning, we do not necessarily assign a pixel to the CB from which there are most optimal paths to this pixel. Because this would assume that each linking path is equally frequent for a specific node. But rather, pixel is assigned to the label it is most frequently associated with, when considering all the optimal forests. So, it assumes that each forest is equally frequent, i.e., each possible predecessor arc is equally frequent for a specific node. Figure 7 shows that if in some cases these assumptions lead to the same result in term of frequency (case of I_6), in general they do not (cases of I_7 and I_8).

4.1.1. Fragmented Drop Paradigm. Let us determine the relative frequency $f_p(\lambda)$ of a pixel p being associated with a label λ using the definition of frequency normalised by the number of observations (optimal forests):

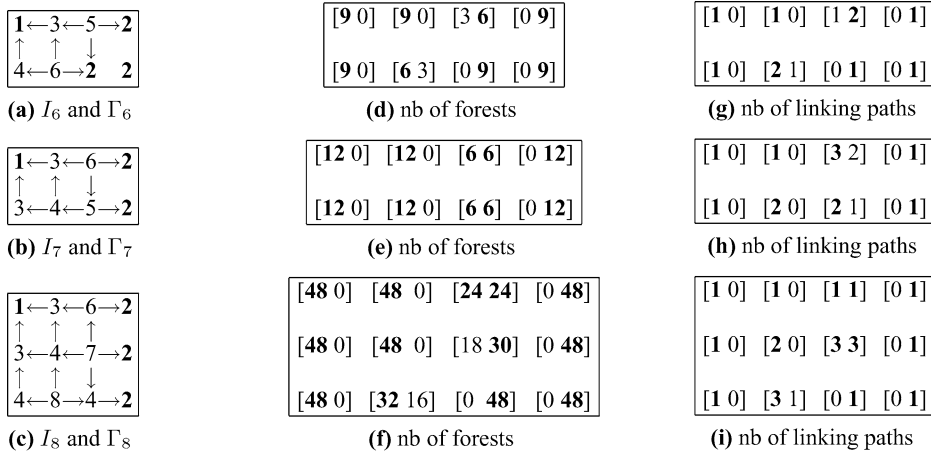$$f_p(\lambda) = \frac{|\Phi_{\pi(p, \lambda)}|}{|\Phi|} \tag{1}$$

$\begin{array}{c}1{\leftarrow}3{\leftarrow}5{\rightarrow}2\\ \uparrow\quad\uparrow\quad\downarrow\\ 4{\leftarrow}6{\rightarrow}2\quad 2\end{array}$	$\begin{array}{cccc}[9\ 0] & [9\ 0] & [3\ 6] & [0\ 9]\\ {}\\ {}[9\ 0] & [6\ 3] & [0\ 9] & [0\ 9]\end{array}$	$\begin{array}{cccc}[1\ 0] & [1\ 0] & [1\ 2] & [0\ 1]\\ {}\\ {}[1\ 0] & [2\ 1] & [0\ 1] & [0\ 1]\end{array}$
(a) I_6 and Γ_6	**(d)** nb of forests	**(g)** nb of linking paths
$\begin{array}{c}1{\leftarrow}3{\leftarrow}6{\rightarrow}2\\ \uparrow\quad\uparrow\quad\downarrow\\ 3{\leftarrow}4{\leftarrow}5{\rightarrow}2\end{array}$	$\begin{array}{cccc}[12\ 0] & [12\ 0] & [6\ 6] & [0\ 12]\\ {}\\ {}[12\ 0] & [12\ 0] & [6\ 6] & [0\ 12]\end{array}$	$\begin{array}{cccc}[1\ 0] & [1\ 0] & [3\ 2] & [0\ 1]\\ {}\\ {}[1\ 0] & [2\ 0] & [2\ 1] & [0\ 1]\end{array}$
(b) I_7 and Γ_7	**(e)** nb of forests	**(h)** nb of linking paths
$\begin{array}{c}1{\leftarrow}3{\leftarrow}6{\rightarrow}2\\ \uparrow\quad\uparrow\quad\uparrow\\ 3{\leftarrow}4{\leftarrow}7{\rightarrow}2\\ \uparrow\quad\uparrow\quad\downarrow\\ 4{\leftarrow}8{\rightarrow}4{\rightarrow}2\end{array}$	$\begin{array}{cccc}[48\ 0] & [48\ \ 0] & [24\ 24] & [0\ 48]\\ {}[48\ 0] & [48\ \ 0] & [18\ 30] & [0\ 48]\\ {}[48\ 0] & [32\ 16] & [0\ \ 48] & [0\ 48]\end{array}$	$\begin{array}{cccc}[1\ 0] & [1\ 0] & [1\ 1] & [0\ 1]\\ {}[1\ 0] & [2\ 0] & [3\ 3] & [0\ 1]\\ {}[1\ 0] & [3\ 1] & [0\ 1] & [0\ 1]\end{array}$
(c) I_8 and Γ_8	**(f)** nb of forests	**(i)** nb of linking paths

Figure 7. Difference between frequency of forests (or predecessor arcs) and linking paths. (a)–(c) Images and their respective MOG (minima are bold-faced). (d)–(f) For each pixel, [x y] are the numbers of optimal forests in which the pixel is labelled with 1 and 2 respectively. (g)–(i) For each pixel, [x y] are the numbers of paths linking the pixel with the minimum 1 and 2 respectively.

where $|\Phi_{\pi(p,\lambda)}|$ corresponds to the number of optimal forests that include a ("descending") path between pixel p and a root with label λ (necessary condition for labelling p with λ).

Let $\Pi(p,\lambda)$ be the set of the paths $\pi(p,\lambda)$ in Γ that link p to a root with label λ. Equation (1) is equivalent to:

$$f_p(\lambda) = \frac{\sum_{\pi\in\Pi(p,\lambda)}|\Phi_\pi|}{|\Phi|} \qquad (2)$$

A path π can be written in this form: $\pi = \langle q_n, q_{n-1}, \ldots, q_2, q_1 \rangle$ where q_{i-1} is predecessor of q_i. Let π' be the set of successor vertices of the arcs contained in path π: $\pi' = \{q_n, q_{n-1}, \ldots, q_2\}$. The number of optimal forests that contain a particular path π is given by:

$$|\Phi_\pi| = \prod_{p\notin\pi',\ |\mathbb{P}(p)|\neq 0}|\mathbb{P}(p)| = \frac{\prod_{p\in V,\ |\mathbb{P}(p)|\neq 0}|\mathbb{P}(p)|}{|\mathbb{P}(q_n)||\mathbb{P}(q_{n-1})|\ldots|\mathbb{P}(q_2)|}$$
$$= \frac{|\Phi|}{\prod_{q\in\pi'}|\mathbb{P}(q)|}$$

because each pixel q_i in π' must have the specified predecessor q_{i-1} whereas all the other pixels p can have any one of the possible predecessors in $|\mathbb{P}(p)|$. From (2) it follows that:

$$f_p(\lambda) = \frac{\sum_{\pi\in\Pi(p,\lambda)}\frac{|\Phi|}{\prod_{q\in\pi'}|\mathbb{P}(q)|}}{|\Phi|}$$

And finally, we obtain:

$$f_p(\lambda) = \sum_{\pi\in\Pi(p,\lambda)}\frac{1}{\prod_{q\in\pi'}|\mathbb{P}(q)|} \qquad (3)$$

Therefore, for each distinct ("descending") path between p and a root with label λ, it is necessary to calculate the inverse of the product of predecessor numbers for each node of the path. After summing all the obtained results, we get the relative frequency of having a path "between p and label λ", the whole set of optimal forests being observed. We saw in the introduction of this paper (Section 1) that, considering the image as a topography, a drop of water falling at a locus (pixel) that belongs to a CB will slide down to the corresponding valley (minimum). Otherwise, it means that the locus belongs to the watershed and the drop of water can slide down to several valleys. Equations (1) and (3) demonstrate that computing the label frequency for a pixel when all the optimal forests are observed is equivalent to evaluating the amount of the drop of water that reaches the respective labelled CB. Indeed, in Eq. (3), the label frequency is equal to the sum of the drop fragments considering all the possible paths between the pixel and the respective CB. Every time the sliding fragmented drop passes through a pixel with many predecessors, the amount of water at that locus is divided in equal parts among all the predecessors to continue the descent until minima (see Fig. 8). For a pixel in CB, the drop will entirely arrive in the same minimum. For a pixel in the TZ, the drop of water is broken up into at least two minima. In Section 4.2, we will see that the frequency-based thinning assigns the pixel to the label whose minimum caught the maximum fraction of the drop of water. And it is equivalent to assigning the most frequent label after observing all the labelled optimal forests.
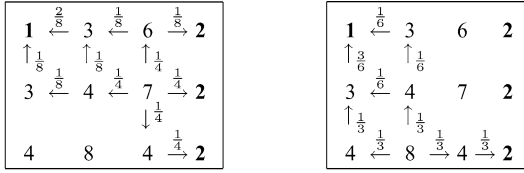
(a) Sliding down from pixel 7   **(b)** Sliding down from pixel 8

Figure 8. Drop of water is sliding down from a TZ pixel and broken up in equal parts whenever it meets an intersection of descending paths (image I_8). (a) Descending paths encountered by the drop from pixel 7 and respective fractions of water sliding down. Minima 1 and 2 catch respectively 3/8 and 5/8 of the drop. This result fits with the proportion of optimal forests leading to each labelling (18 and 30, see Fig. 7(f)). (b) Idem for pixel 8: 2/3 and 1/3 are in the same proportion as 32 and 16.

4.1.2. Recursive Computation by Immersion.

We demonstrate now that Eq. (3) corresponding to the fragmented drop paradigm is equivalent to a recursive formula (4) corresponding to an immersion simulation.

In Eq. (3) where it appears a sum in relation to the different paths π between p and λ, let us group the paths depending on whether they pass through one or another of the $|\mathbb{P}(p)|$ predecessors v of p —as any path linking p to λ is necessarily the concatenation of the arc $\langle p, v \rangle$ with a path linking v to λ. So, grouping according to possible predecessors v and factoring the sum, we get:

$$f_p(\lambda) = \sum_{v \in \mathbb{P}(p)} \sum_{\pi \in \Pi(v,\lambda)} \left[\frac{1}{|\mathbb{P}(p)| \prod_{q \in \pi'} |\mathbb{P}(q)|} \right]$$
$$= \frac{1}{|\mathbb{P}(p)|} \sum_{v \in \mathbb{P}(p)} \left[\sum_{\pi \in \Pi(v,\lambda)} \frac{1}{\prod_{q \in \pi'} |\mathbb{P}(q)|} \right]$$

According to (3), replace the expression between brackets:

$$f_p(\lambda) = \frac{1}{|\mathbb{P}(p)|} \sum_{v \in \mathbb{P}(p)} f_v(\lambda) \qquad (4)$$

As we can see, this recursive formula implicitly describes an immersion process where label frequencies are first computed at lower levels (predecessors v) and then at higher levels (p). Note also that the normalisation term allows to take into account all the possible predecessors. In summary, we have three ways of computing label frequency, as Eq. (4) is equivalent to Eqs. (1) and (3). In practice, Eq. (4) is used by the frequency-based thinning algorithm proposed in Section 4.3.

4.2. Computing the Map of Most Frequent Labels

Now we have computed the frequencies of each label to be assigned to each pixel, we want to choose the most frequent label for each pixel (see Fig. 9(a)). To obtain a unique label map, we cannot decide which of two or more labels is assigned to a pixel in case of equal frequency (EF). Indeed, there can be many EF pixels. They are pixels where at least two labels have the same maximum frequency of being assigned to (see Fig. 9(c) and (d)). They constitute the first type of pixel that cannot be thinned with the maximum frequency criterion.

A second type of pixel cannot be labelled: the isolated (ISO) pixels. To ensure that a segmentation is achieved, the labelling must admit at least an optimal forest as its support. And we saw that this forest is necessarily made of the possible arcs of the MOG. If
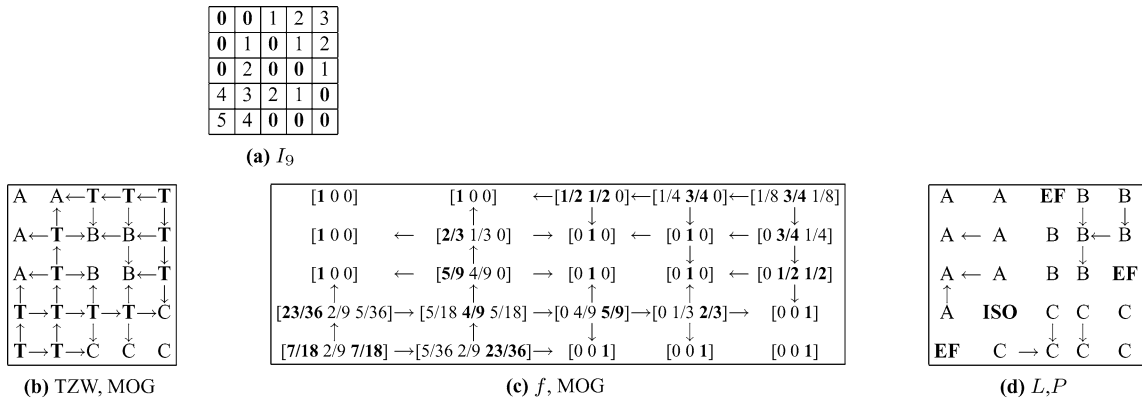
pixel	$f(A)$	$f(B)$	$f(C)$	L(pixel)	P(pixel)
q_1	**0.5**	0.3	0.2	**A**	...
q_2	**0.5**	0.3	0.2	**A**	...
q_3	0.2	0.3	**0.5**	C	...
p	**0.4**	0.3	0.3	**A**	q_1 or q_2

(a)

pixel	$f(A)$	$f(B)$	$f(C)$	L(pixel)	P(pixel)
q_1	**0.6**	0.0	0.4	A	...
q_2	0.2	**0.5**	0.3	B	...
q_3	0.1	**0.5**	0.4	B	...
p	0.3	0.33	**0.37**	**ISO**	NIL

(b)

pixel	$f(A)$	$f(B)$	$f(C)$	L(pixel)	P(pixel)
q_1	**0.5**	0.4	0.1	**A**	...
q_2	0.4	**0.5**	0.1	**B**	...
q_3	0.2	0.2	**0.6**	C	...
p	**0.37**	**0.37**	0.26	**EF**	NIL

(c)

pixel	$f(A)$	$f(B)$	$f(C)$	L(pixel)	P(pixel)
q_1	**0.5**	0.1	0.4	A	...
q_2	0.2	**0.5**	0.3	**B**	...
q_3	0.1	**0.5**	0.4	**B**	...
p	0.26	**0.37**	**0.37**	**ISO**	NIL

(d)

Figure 9. Computing the most frequent label. Pixels q_1, q_2, q_3 are possible predecessors of p. Label frequencies f of p are computed according to the recursive formula. Then, label L and predecessor P are assigned when possible. (a) Most frequent label for p is A and two predecessors are possible (see Eq. (5)). (b) Isolated (ISO) pixel case. Label C is the most frequent for p but no predecessor has the same label (see Eq. (6)). (c) Equal frequency (EF) case between labels A and B. No predecessor can be chosen (see Eq. (7)). (d) Other isolated pixel case. Equal frequency case between labels B and C but no predecessor is labelled with C (see Eq. (8)).

0	0	1	2	3
0	1	0	1	2
0	2	0	0	1
4	3	2	1	0
5	4	0	0	0

(a) I_9

(b) TZW, MOG

(c) f, MOG

(d) L,P

Figure 10. (a) Input greyscale image with 3 minima (in bold). (b) Multipredecessor Optimal Graph (arrows to predecessors) and TZW (label T is tie-zone). (c) MOG and frequency of label assignments. Each vector represents the frequencies of labelling the pixel with A, B and C respectively. Maximum frequency is bold-faced. (d): Resulting label map L and one of its supports P. Observe the cases of equal frequency (EF) and isolated pixel (ISO).

such a support does not exist, it means that the labelling is not a segmentation because some pixels can be disconnected from their seeds. So, the case of isolated pixel occurs when no arc can link the labelled pixel with another one of same label (see Fig. 9(b) and (d)). Figure 10 shows the frequency-based thinning of an image with EF and ISO cases.

Observe that the frequency of each label assignment is not a monotonically increasing function. If a label is most frequently assigned to a pixel, it does not mean that it is most frequently assigned to its successor in the path: its frequency may decrease at the expense of other assignment frequencies.

In conclusion, the result of the frequency-based thinning is a thinning of the TZ consisting of assigning pixels of the TZ to the most frequent CB's label when it is unique and when there exists a path linking to the respective CB root within the CB. Otherwise, the pixels have undefined (EF or ISO) labels. Thus, the labelled pixels have a support made of minimum cost trees while some pixels (EF and ISO) are disconnected from these trees. The solution is not necessarily an optimal forest because of these special cases.

Hereafter are presented the conditions (rules) of the labelling process for each pixel p, depicted in Fig. 9(a)–(d) respectively. The most frequent label map is denoted by L and the predecessor map by P (it is only one of its possible supports). To summarise, when the most frequent label for a node is unique and at least one of its predecessors is assigned to this label, the label is propagated from the predecessor to this node (rule (5)). But if no predecessor is assigned to this most frequent label, the node is isolated (rule (6)). When there are many most frequent labels for a node and each of

them is assigned to at least one of the predecessors, the node is an equal frequency case (rule (7)). But if at least one of the equally most frequent labels is not present among the predecessors, the node is considered isolated (rule (8)).

$$\left.\begin{array}{l} \exists \lambda_{ML}, \forall \lambda \neq \lambda_{ML},\ f_p[\lambda_{ML}] > f_p[\lambda] \\ \exists q \in \mathbb{P}[p], \qquad L[q] = \lambda_{ML} \end{array}\right\}$$
$$\implies \begin{cases} L[p] \leftarrow \lambda_{ML} \\ P[p] \leftarrow q \end{cases} \quad (5)$$

$$\left.\begin{array}{l} \exists \lambda_{ML}, \forall \lambda \neq \lambda_{ML},\ f_p[\lambda_{ML}] > f_p[\lambda] \\ \forall q \in \mathbb{P}[p], \qquad L[q] \neq \lambda_{ML} \end{array}\right\}$$
$$\implies \begin{cases} L[p] \leftarrow \text{ISO} \\ P[p] \leftarrow \text{NIL} \end{cases} \quad (6)$$

$$\left.\begin{array}{l} \exists \Lambda_{ML}, \forall \lambda_{ML} \in \Lambda_{ML}, \forall \lambda \neq \lambda_{ML},\ f_p[\lambda_{ML}] \geq f_p[\lambda] \\ \forall \lambda_{ML} \in \Lambda_{ML}, \exists q \in \mathbb{P}[p], \qquad L[q] = \lambda_{ML} \end{array}\right\}$$
$$\implies \begin{cases} L[p] \leftarrow \text{EF} \\ P[p] \leftarrow \text{NIL} \end{cases} \quad (7)$$

$$\left.\begin{array}{l} \exists \Lambda_{ML}, \forall \lambda_{ML} \in \Lambda_{ML}, \forall \lambda \neq \lambda_{ML},\ f_p[\lambda_{ML}] \geq f_p[\lambda] \\ \exists \lambda_{ML} \in \Lambda_{ML}, \forall q \in \mathbb{P}[p], \qquad L[q] \neq \lambda_{ML} \end{array}\right\}$$
$$\implies \begin{cases} L[p] \leftarrow \text{ISO} \\ P[p] \leftarrow \text{NIL} \end{cases} \quad (8)$$

Remember that the frequency-based thinning uses frequencies of label assignment given by the set of optimal forests and alternatively, these frequencies can be viewed as the fractions of a divisible drop of water sliding down from pixel to minima. Under this approach, the frequency-based thinning assigns to each pixel the label of the minimum that caught the greatest fraction of the drop of water if there exists a descending path from the pixel to the minimum that is entirely included in the CB.

4.3. Dijkstra-IFT-Based Algorithm for the Label-Frequency Thinning of the Tie-Zone

Now the frequency criterion was defined, an algorithm to thin the TZ is proposed (see Algorithm 2). It is based on Algorithm 1. It has therefore the same structure and needs the same input (image-graph (V, A, I), seeds S, labelling λ and handicap h functions) and the same ordered queue and cost maps (C_1, C_2). Observe that instead of predecessor and label maps, there are multi-predecessor set map $\mathbb{P}$ and label set map $\mathbb{L}$ for internal processing. Frequencies f, most frequent label map L (the frequency-based thinning in output) and its support P (a predecessor map) are computed during the main loop (i.e. the emptying of the ordered queue). The lines with bold-faced number are specific of the frequency-based thinning algorithm and differ from the Label Merging algorithm mentioned in Section 3.2.

Let us comment some distinctive particularities of the frequency-based thinning algorithm. In line 1, the frequency map is also initialised with 0. In line 5, whenever a pixel is removed from the ordered queue, the frequency map has to be updated (`UpdateLabelFreq` function). Its definitive frequencies are computed from the frequencies already assigned to its predecessors (the already processed neighbours).

The `UpdateLabelFreq` function (lines 22–28) corresponds to rules (5)–(8). If only one label can be assigned to a pixel (line 22), frequency of this labelling is 1. A predecessor is assigned but is not unique. If many labels can be assigned (line 23), the frequencies of each labelling are computed from all the predecessors' frequencies by using the recursive formula. Lines 24–26 describe the case of one unique label of maximum frequency. If it has no predecessor with same label (line 26), it is an isolated pixel. Lines 27–28 describe

Algorithm 2: Label frequency-based thinning algorithm.

1. $\forall p \in V, \; C_2[p] \leftarrow 0; \quad done(p) \leftarrow$ FALSE; $\; \forall \lambda \in \Lambda, \;\; f_p[\lambda] \leftarrow 0;$

2. $\forall p \notin S, \; C_1[p] \leftarrow \infty; \;\; \mathbb{L}[p] \leftarrow \{\}; \qquad \mathbb{P}[p] \leftarrow \{\};$

3. $\forall p \in S, \; C_1[p] \leftarrow h(p); \mathbb{L}[p] \leftarrow \{\lambda(p)\}; \quad \mathbb{P}[p] \leftarrow \{p\}; \;\; $ Enqueue$(p, h(p));$

4. **while** `QueueNotEmpty`,

5. $\quad v \leftarrow$ DequeueMin; $\;\; done(v) \leftarrow$ TRUE; UpdateLabelFreq$(v);$

6. $\quad \forall p \in N_G(v)$ **and** $done(p) =$ FALSE,

7. $\qquad c \leftarrow \max\{C_1[v], I[p]\};$

8. $\qquad$ **if** $c < C_1[p]$,

9. $\qquad\quad$ **if** p in Q, Dequeue$(p);$

10. $\qquad\quad C_1[p] \leftarrow c; \;\; \mathbb{L}[p] \leftarrow \mathbb{L}[v]; \;\; \mathbb{P}[p] \leftarrow \{v\};$

11. $\qquad\quad$ Enqueue$(p, C_1[p]);$

12. $\qquad\quad$ **if** $c = C_1[v], \;\; C_2[p] \leftarrow C_2[v] + 1;$

13. $\qquad$ **else**, **if** $c = C_1[p]$,

14. $\qquad\quad$ **if** $\mathbb{L}[p] \neq \mathbb{L}[v]$,

15. $\qquad\qquad$ **if** $c = C_1[v]$,

16. $\qquad\qquad\quad$ **if** $C_2[p] = C_2[v] + 1, \;\; \mathbb{L}[p] \leftarrow \mathbb{L}[p] \cup \mathbb{L}[v]; \;\; \mathbb{P}[p] \leftarrow \mathbb{P}[p] \cup \{v\};$

17. $\qquad\qquad$ **else**, $\;\; \mathbb{L}[p] \leftarrow \mathbb{L}[p] \cup \mathbb{L}[v]; \;\; \mathbb{P}[p] \leftarrow \mathbb{P}[p] \cup \{v\};$

18. $\qquad\quad$ **else**,

19. $\qquad\qquad$ **if** $c = C_1[v]$,

20. $\qquad\qquad\quad$ **if** $C_2[p] = C_2[v] + 1, \;\; \mathbb{P}[p] \leftarrow \mathbb{P}[p] \cup \{v\};$

21. $\qquad\qquad$ **else**, $\;\; \mathbb{P}[p] \leftarrow \mathbb{P}[p] \cup \{v\};$

UpdateLabelFreq(v):

22. **if** $|\mathbb{L}[v]| = 1, \; \exists \lambda \in \mathbb{L}[v], \exists p \in \mathbb{P}[v], \;\; f_v[\lambda] \leftarrow 1; \;\; L[v] \leftarrow \lambda; \;\; P[v] \leftarrow p;$ /* not necessarily unique */

23. **else**, $\;\; \forall p \in \mathbb{P}[v], \;\; \forall \lambda \in \mathbb{L}[v], \;\; f_v[\lambda] \leftarrow f_v[\lambda] + \frac{1}{|\mathbb{P}[v]|} f_p[\lambda];$

24. $\qquad$ **if** $\exists \lambda_{ML}, \; \forall \lambda \neq \lambda_{ML}, \;\; f_v[\lambda_{ML}] > f_v[\lambda]$,

25. $\qquad\quad$ **if** $\exists p \in \mathbb{P}[v], L[p] = \lambda_{ML}, \;\; L[v] \leftarrow \lambda_{ML}; P[v] \leftarrow p;$ $\qquad$ /* not necessarily unique */

26. $\qquad\quad$ **else**, $\qquad\qquad\qquad\qquad L[v] \leftarrow$ ISO; $\;\; P[v] \leftarrow$ NIL;

27. $\qquad$ **else**, $\;\; L[v] \leftarrow$ EF; $P[v] \leftarrow$ NIL;

28. $\qquad\quad$ **if** $\exists \lambda_i, \; f_v[\lambda_i] \geq f_v[\lambda], \forall \lambda \neq \lambda_i, \forall p \in \mathbb{P}[v], L[p] \neq \lambda_i, \;\; L[v] \leftarrow$ ISO;

the case of at least two labels of maximum frequency: equal frequency case EF. But if one of the equally most frequent label has no predecessor with same label (line 28), the pixel is considered isolated in some sense.

Coming back to the main loop of the algorithm, we analyse the case of cost equality (former TZ case): when proposed cost is equal to current cost in cost map (lines 13–21). Notice that whenever the sets of labels are different (lines 14–17) or equal (lines 18–21), a new predecessor must be added in the multipredecessor set. The label merging is actually necessary only when the propagating label set is not already included to the current label set: $\mathbb{L}[v] \cup \mathbb{L}[p] \neq \mathbb{L}[p]$. Elsewhere, the label merging is redundant. And avoiding it is desirable. According to the implementation of these sets, the above criterion of non-inclusion can be more costly than a simple inequality operation. The proposed algorithm uses this alternative criterion ($\mathbb{L}[v] \neq \mathbb{L}[p]$ in line 14) that can allow unnecessary label merging. This choice of criterion will depend on the cost of merging, inclusion and equality test operations on label sets. Besides, observe that the implementation difficulty is with managing the sets $\mathbb{L}$ and $\mathbb{P}$. The size of $\mathbb{P}$ is variable but limited by the number of neighbours. Implementation of set $\mathbb{L}$ is more problematic: its size depends on the number of labels that tied together. The propagation of the frequencies to the neighbours are another related problem.

Notice that the computations of frequencies, most frequent label map and its support could be integrated in the IFT loop because of the recursive formula for frequencies and the ordered computing: frequencies and label assignment of a pixel only depend on its lower neighbours.

4.4.   Illustration

To illustrate the concepts introduced in this paper, we applied the TZW and the frequency-based thinning on a rather simple real image of airplane. The gradient image is presented in Fig. 11(a) whereas TZW is in Fig. 11(b). White areas are the tie-zone and grey ones are the CBs. Each CB corresponds to a regional minimum. Observe that in practice, the detection of the markers (set of the seeds with same label) should be achieved manually or automatically by filtering, for example, the minima according to their dynamics [16, 21]. Here, the minima were not filtered on purpose. This allows to visualise the location of

large TZ and the behaviour of the proposed thinning in this case. The quite large TZ areas of Fig. 11(b) desappeared in Fig. 11(c): the frequency-based thinning was applied. We can see that some pixels (in white) remain undefined (EF or ISO cases). Most of them are on the frontier between CBs and constitute thin segments of line. However, two regions are still thick. They can be visualised in the detailed view of Fig. 11(d). Grey CBs are constituted of optimal trees (in black) rooted to regional minima (nodes contoured by a square) while EF pixels (in white with a central dot) are disconnected of any tree. Most ISO pixels (in white with a central black square) occur by transitivity, because their predecessors are already EF pixels without defined label. Even if they have a most frequent label, there is no labelled descending path to the corresponding CB. EF pixels are blocking their way. Therefore, ISO pixels are also disconnected of any tree.

Note that there may be cases where the frequency-based thinning does not thin TZ. For example, symmetric images whose topography is similar to pyramid or cone may contain only EF pixels and ISO pixels (by transitivity) in their tie-zone.

Figure 12(a) shows one of the optimal forests defined by the WT by IFT. It was randomly obtained from the multipredecessor optimal graph. If we compare this figure with Fig. 11(b) and (c), the catchment basins are the same, out of the tie-zone. Variations occur in the tie-zone: TZW does not make any arbitrary decision, the thinning unties the TZ according to the frequency criteria and the random optimal forest makes arbitrary decisions. Observe the important differences between the thinning and the random IFT in the tie-zone: e.g. (white) tie-zones in the top right border, near the bottom left corner, on the contours of the wings and in the rear and front parts of the airplane.

Figure 12(b) shows the results given by an implementation of the Vincent and Soille's algorithm for four different image scannings. There are many little differences in the segmentations on the contours, in the wings, and in many other catchment basins of the background. Note that other implementations could choose other internal processing orders and possibly give other solutions than these ones. Observe that the watershed line becomes thicker in the front regions of the airplane. The frequency-based thinning returns EF/ISO labels in these regions. The minima in these regions should be filtered or redefined to remove the tie-zone.
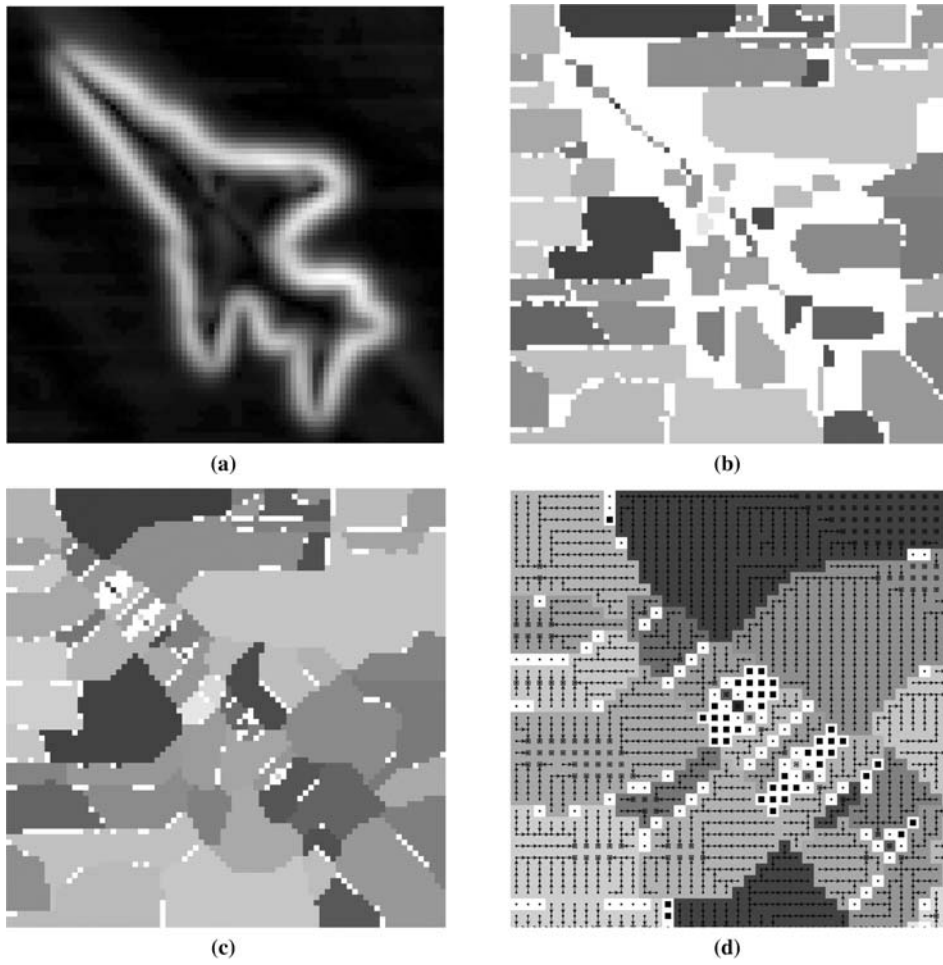
Figure 11. (a) Input gradient image of an airplane. (b) TZW (TZ in white). (c) Frequency-based thinning (EF and ISO pixels in white). (d) Detail of the frequency-based thinning showing optimal trees in CBs, EF (white with dot) and ISO (white with black square) pixels. The seeds (nodes contoured by a square) represent the regional minima of the gradient image.
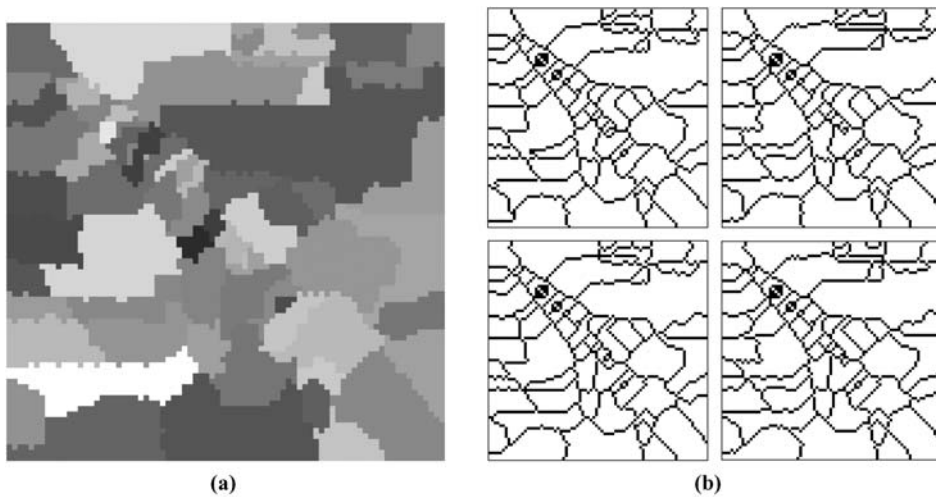

Figure 12. (a) WT corresponding to a random optimal forest (one of the solution of WT by IFT). (b) WT by Vincent-Soille's algorithm (watershed lines in black): four different image scannings.

Table 1.   Table of abbreviations

CB	catchment basin
EF	"equal frequency" label
FIFO	first-in-first-out
IFT	image foresting transform
ISO	"isolated" label
MOG	multipredecessor optimal graph
SKIZ	skeleton by influence zones
TZ	tie-zone
TZW	tie-zone watershed
WT	watershed transform

5.   Conclusion

In this paper, we saw that there are many watershed transform (WT) definitions in both continuous and digital spaces and lots of algorithms to compute them. But there is an inadequacy between some algorithms and their respective definition because they impose a watershed solution containing either separating watershed lines, or only regions (catchment basins). Consequently, such algorithms may return different solutions depending on the implementation because they introduce a bias in relation to the definitions. Other definitions like the WT by image foresting transform (IFT) admit several solutions. The corresponding algorithms respect their definition but introduce a bias by choosing one of the multiple solutions. Again, the solution is not unique and depends on the implementation. The tie-zone watershed (TZW) returns a unique and unbiased solution by considering all the possible solutions. As large tie-zones may sometimes appear, we proposed a label frequency-based thinning that leads to a unique solution. If we define our space of observation as the set of the possible solutions (optimal forests), we can compute the frequency of the labels associated with each pixel. The thinning consists in assigning the most frequent label while preserving the segmented region connectivity, i.e., each labelled pixel must be connected to a root by a path whose pixels are all associated with the same label. Notice that the thinning does not necessarily correspond to an optimal forest: some pixels can be simply disconnected from the trees (i.e., unlabelled) because assigning the most frequent label to them would violate the segmentated region connectivity. We demonstrated that the label frequency computation can also be explained by the fragmented drop paradigm: a drop of water can be fragmented and go on until several minima. The fraction of water in each minimum corresponds to the frequency of assigning the corresponding label to the pixel on which the drop fell down. Alternatively, the label frequency can be computed by a recursive formula that simulates an immersion. Finally, we proposed an algorithm that computes at the same time the TZW, the label frequencies and the frequency-based thinning.

Acknowledgments

Note

1. Table 1 (at the end of the article) contains all the abbreviations used in the text.

References

1. R. Audigier and R. Lotufo, "Tie-zone watershed, bottlenecks and segmentation robustness analysis," in *XVIII Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI'05)*, IEEE Press: Natal, Brazil, October 2005, pp. 55–62.

2. R. Audigier, R. Lotufo, and M. Couprie, "The tie-zone watershed: Definition, algorithm and applications," in *Proceedings of IEEE International Conference on Image Processing (ICIP'05)*, Genova, Italy, September 2005, Vol. 2, pp. 654–657.

3. C. Berge, *Théorie des Graphes et ses Applications*, Dunod: Paris, France, 1958.

4. G. Bertrand, "On topological watersheds," *Journal of Mathematical Imaging and Vision*, Vol. 22, Nos. 2–3, pp. 217–230, May 2005. Special issue on Mathematical Morphology.

5. S. Beucher and Ch. Lantuéjoul, "Use of watersheds in contour detection," in *International Workshop on Image Processing, Real-Time Edge and Motion Detection/Estimation*, Rennes: France, 1979.

6. S. Beucher and F. Meyer, "The morphological approach to segmentation: The watershed transform," in *Mathematical Morphology in Image Processing*, E.R. Dougherty (Ed.), Marcel Dekker, Inc.: New York (NY), USA, 1993, chapter 12, pp. 433–481.

7. A. Bieniek, H. Burkhardt, H. Marschner, M. Nölle, and G. Schreiber, "A parallel watershed algorithmm" in *Proceedings of 10th Scandinavian Conference on Image Analysis (SCIA'97)*, Lappeenranta: Finland, June 1997, pp. 237–244.

8. A. Bieniek and A. Moga, "A connected component approach to the watershed segmentation," in *Mathematical Morphology and its Applications to Image and Signal Processing*, H.J.A.M. Heijmans and J.B.T.M. Roerdink (Eds.), Kluwer Acad., Dordrecht: The Netherlands, 1998, pp. 215–222.

9. A. Bieniek and A. Moga, "An efficient watershed algorithm based on connected components," *Pattern Recognition*, Vol. 33, No. 6, pp. 907–916, June 2000.

10. M. Couprie and G. Bertrand, "Topological grayscale watershed transformation," in *SPIE Vision Geometry V Proceedings*, 1997, Vol. 3168, pp. 136–146.

11. M. Couprie, L. Najman, and G. Bertrand, "Algorithms for the topological watershed," in *Proceedings of the 12th International Conference on Discrete Geometry for Computer Imagery (DGCI 2005)*, Pascal Lienhardt, Eric Andres, and Guillaume Damiand (Eds.), Poitiers: France, January 2005, Vol. 3429, pp. 172–182.

12. M. Couprie, L. Najman, and G. Bertrand, "Quasi-linear algorithms for the topological watershed," *Journal of Mathematical Imaging and Vision*, Vol. 22, Nos. 2–3, pp. 231–249, May 2005. Special issue on Mathematical Morphology.

13. E.W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, 1959.

14. A.X. Falcão, B.S. da Cunha, and R.A. Lotufo, "Design of connected operators using the image foresting transform," *SPIE on Medical Imaging*, Vol. 4322, pp. 468–479, February 2001.

15. A.X. Falcão, J. Stolfi, and R.A. Lotufo, "The image foresting transform: Theory, algorithms, and applications," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 26, No. 1, pp. 19–29, January 2004.

16. M. Grimaud, "A new measure of contrast: the dynamics," in *Image Algebra and Morphological Image Processing III*, E. Dougherty, J. Serra, and P. Gader (Eds.), SPIE: San Diego, CA, USA, 1992, Vol. 1769, pp. 292–305.

17. R.A. Lotufo and A.X. Falcão, "The ordered queue and the optimality of the watershed approaches," in *5th International Symposium on Mathematical Morphology*, Kluwer Academic: Palo Alto (CA), USA, June 2000, pp. 341–350.

18. R.A. Lotufo, A.X. Falcão, and F.A. Zampirolli, "IFT-watershed from gray-scale marker," in *Proceedings of the 15th Brazilian Symposium on Computer Graphics and Image Processing*, Fortaleza (CE): Brazil, IEEE Computer Society, October 2002, pp. 146–152.

19. A. Meijster and J.B.T.M. Roerdink, "A disjoint set algorithm for the watershed transform," in *Proceedings of IX European Signal Processing Conference (EUSIPCO'98)*, IEEE Computer Society: Rhodes, Greece, September 1998, pp. 1665–1668.

20. F. Meyer, "Topographic distance and watershed lines," *Signal Processing*, No. 38, pp. 113–125, 1994.

21. F. Meyer, "The dynamics of minima and contours," in *Mathematical Morphology and its Application to Image and Signal Processing (3rd)*, P. Maragos, R.W. Schafer, and M.A. Butt (Eds.), Computational Imaging and Vision, Kluwer Academic Publishers: Boston, 1996, pp. 329–336.

22. F. Meyer and S. Beucher, "Morphological segmentation," *Journal of Visual Communication and Image Representation*, Vol. 1, No. 1, pp. 21–46, 1990.

23. L. Najman and M. Couprie, "Watershed algorithms and contrast preservation," in *Discrete geometry for computer imagery*, Vol. 2886 of *Lecture Notes in Computer Science*, Springer, 2003, pp. 62–71.

24. L. Najman, M. Couprie, and G. Bertrand, "Watersheds, mosaics and the emergence paradigm," *Discrete Applied Mathematics*, Vol. 147, Nos. 2–3, pp. 301–324, April 2005. Special issue on DGCI.

25. L. Najman and M. Schmitt, "Watershed of a continuous function," *Signal Process.*, Vol. 38, No. 1, pp. 99–112, 1994.

26. F. Prêteux, "On a distance function approach for gray-level mathematical morphology," in *Mathematical Morphology in Image Processing*, E.R. Dougherty (Ed.), chapter 10, Marcel Dekker, Inc.: New York (NY), USA, 1993, pp. 323–349.

27. J.B.T.M. Roerdink and A. Meijster, "The watershed transform: Definitions, algorithms and parallelization strategies," *Fundamenta Informaticae*, Vol. 41, Nos. 1–2, pp. 187–228, January 2000. Special issue on mathematical morphology.

28. L. Vincent and P. Soille, "Watersheds in digital spaces: An efficient algorithm based on immersion simulations," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 13, No. 6, pp. 583–598, 1991.

Romaric Audigier received the Electrical Engineering Diploma from Institut National des Sciences Appliquées (INSA) of Lyon, France, in 2001, the M.Sc. degree in Images and Systems from École Doctorale EEA of Lyon, in 2001, and the M.Sc. degree in Electrical Engineering from the State University of Campinas (UNICAMP), SP, Brazil, in 2004. Since 2004, he has been a Ph.D. student in the Department of Computer Engineering and Industrial Automation at the State University of Campinas (UNICAMP), SP, Brazil. His research interests include image processing and analysis, mathematical morphology, image segmentation, medical imaging and volume visualization.

Roberto Lotufo obtained the Electronic Engineering Diploma from Instituto Tecnologico de Aeronautica, Brazil, in 1978, the M.Sc. degree from the University of Campinas, UNICAMP, Brazil, in 1981, and the Ph.D. degree from the University of Bristol, U.K., in 1990, in Electrical Engineering. He is a full professor at the School of Electrical and Computer Engineering, University of Campinas (UNICAMP), Brazil, where he has worked for since 1981. His principal interests are in the areas of Image Processing and Analysis, Mathematical Morphology, Image Segmentation and Medical Imaging. He is one of the main architects of two morphological toolboxes: MMach for Khoros, and SDC Morphology Toolbox for MATLAB. He is the executive director of Inova Unicamp, the agency for innovation at Unicamp, since 2004. Prof. Lotufo has published over 50 refereed international journal and full conference papers.