J Math Imaging Vis (2007) 29: 141–162
DOI 10.1007/s10851-007-0035-4

Automatic Image Segmentation by Tree Pruning

Felipe P.G. Bergo · Alexandre X. Falcão · Paulo A.V. Miranda · Leonardo M. Rocha

Published online: 9 November 2007

Abstract The *Image Foresting Transform* (IFT) is a tool for the design of image processing operators based on connectivity, which reduces image processing problems into an optimum-path forest problem in a graph derived from the image. A new image operator is presented, which solves segmentation by pruning trees of the forest. An IFT is applied to create an optimum-path forest whose roots are *seed pixels*, selected inside a desired object. In this forest, object and background are connected by optimum paths (*leaking paths*), which cross the object's boundary through its "most weakly connected" parts (*leaking pixels*). These leaking pixels are automatically identified and their subtrees are eliminated, such that the remaining forest defines the object. Tree pruning runs in linear time, is extensible to multidimensional images, is free of *ad hoc* parameters, and requires only internal seeds, with little interference from the heterogeneity of the background. These aspects favor solutions for automatic segmentation. We present a formal definition of the obtained objects, algorithms, sufficient conditions for tree pruning, and two applications involving automatic segmentation: 3D MR-image segmentation of the human brain and image segmentation of license plates. Given that its most competitive approach is the watershed transform by markers, we also include a comparative analysis between them.

Keywords Image segmentation · Graph-search algorithms · Image foresting transform · Image processing · Watershed transform

F.P.G. Bergo · A.X. Falcão (✉) · P.A.V. Miranda
LIV, Institute of Computing, State University of Campinas (UNICAMP), CP 6176, 13083-970 Campinas, SP, Brazil
e-mail: afalcao@ic.unicamp.br

F.P.G. Bergo
e-mail: bergo@liv.ic.unicamp.br

P.A.V. Miranda
e-mail: pavm@liv.ic.unicamp.br

L.M. Rocha
DECOM, FEEC, State University of Campinas (UNICAMP), CP 6101, 13083-970 Campinas, SP, Brazil
e-mail: leorocha@decom.fee.unicamp.br

1 Introduction

The problem of defining the precise spatial extent of a desired object in a given image, namely *image segmentation*, is addressed. The difficulties stem from the absence of global object information (location, shape, appearance) and the similarities between object and background with respect to image features (color and texture). These difficulties usually call for user assistance [3, 6, 7, 9, 15, 20], making automatic segmentation viable only in an application-dependent and tailored fashion. Methods for automatic segmentation should separate the part that is application-dependent from the application-independent part, such that the former can be easily tailored for different applications. We present a method, called *tree pruning*, which is consistent with this strategy.

Tree pruning uses the *Image Foresting Transform* (IFT)—a tool for the design of image processing operators based on connectivity [13]. The IFT has been applied to compute distance transforms, multiscale skeletonizations, morphological reconstructions, watershed transforms, boundary tracking, fractal dimension, and shape saliences [11, 12, 14, 23, 24, 30, 31]. Tree pruning is the first IFT-based operator that exploits a *combinatorial property* of the forest—the number of descendants that each node has in the image's border.

In the IFT framework, an image is interpreted as a graph whose nodes are image pixels and whose arcs are defined by an *adjacency relation* between pixels. For a given set of *seed pixels* and suitable *path-cost function*, the IFT computes an optimum-path forest in the graph whose roots are drawn from the seed set. Each tree in the forest consists of pixels more strongly connected to its root than to any other seed. In tree pruning, the seeds are chosen inside the object and the choice of the path-cost function intends to connect object and background by optimum paths (*leaking paths*), which cross the object's boundary through its "most weakly connected" parts (*leaking pixels*). The above combinatorial property is exploited to automatically identify the leaking pixels and eliminate their subtrees, such that the remaining forest defines the object.

Tree pruning runs in time proportional to the number of pixels, is extensible to multidimensional images, is free of *ad hoc* parameters, and requires only internal seeds, with little interference from the heterogeneity of the background. These aspects favor solutions which exploit image features and object information for automatic segmentation. For example, we can estimate seeds using approaches for object location [34]. Candidate seeds can otherwise be used to obtain a set of possible objects, and the desired one can be chosen based on objective functions [21, 29, 35] or object features and pattern classifiers [8, 19, 22]. One can also exploit some combination between tree pruning and deformable models [4, 6, 7, 20] in order to achieve a better agreement between the geometry of the model and local image features. Even in the context of interactive segmentation, it is highly desirable to make the user's actions simple and minimal. Tree pruning reduces user intervention to a few markers in the image.

In comparison with region-growing approaches based on optimum paths from internal seeds [26, 32], the criterion to disconnect object and background does not use the costs of the optimum paths but the above combinatorial property. Optimum paths that reach object pixels are assumed to not pass through the background, instead of having costs strictly lower than the costs of paths that reach the background. In tree pruning, internal seeds compete among themselves and only a few seeds become roots of leaking paths. Other approaches based on optimal seed competition [2, 23, 27, 28, 33] can be roughly described in three steps: (i) seed pixels are selected inside and outside the objects, (ii) each seed defines an *influence zone* that contains pixels which are more strongly connected to that seed than to any other, and (iii) each object is defined by the union of the influence zones of its internal seeds. The absence of boundary information and/or heterogeneity of the background usually cause invasion (leaking) of object seeds in influence zones of background seeds and vice-versa. In interactive segmentation, the user corrects leaking by adding and removing seeds. In the context of the IFT, these corrections can be done in sublinear time [1, 9] (e.g., by differential watershed transforms). Tree pruning exploits the leaking problem and also favors solutions for automatic segmentation, as discussed above. On the other hand, tree pruning and the IFT-watershed transform by markers [23] use the same image graph and path-cost function which makes them competitive approaches when the external seeds for watershed can be found automatically. This aspect is also evaluated in Sect. 6.

Approaches for image segmentation usually exploit image features and some object information to emphasize the discontinuities between object and background. It is desirable, for example, that the external energy in snakes be lower along the object's boundary than inside and outside it [20]; the arc costs in live wire be lower along the object's boundary than within a neighborhood around it [15]; the local affinities in relative fuzzy connectedness [28] be higher inside and outside the object than on its boundary; the gradient values in watershed transform be higher for pixels on the object's boundary than inside and outside it [2, 23, 33]; and the arc weights in graph-cut segmentation be lower across the object's boundary than inside and outside it [3, 21, 29]. Additionally, the energy minimization in [3, 21] using min-cut/max-flow algorithms from source to sink nodes [16] also requires lower arc-weights between source and object pixels, higher arc-weights between sink and object pixels, lower arc-weights between sink and background pixels, and higher arc-weights between source and background pixels. Clearly the effectiveness of these approaches is affected when the above *desirable conditions* are not fully satisfied, but they can still work under certain hard constraints (usually, involving the user). For example, Boykov and Jolly [3] allow the user to force the arc weights with source and sink by selecting seed pixels inside and outside the object.

Tree pruning can take advantage of the same image features and object information to create a *gradient-like image* (Fig. 1a) in which pixel values are higher on the object's boundary than inside it and, at least, in a neighborhood outside it. The *cost of a path* in tree pruning is given by the maximum gradient value along it. Considering all possible paths from the internal seed set to a given pixel, the IFT assigns a path of minimum cost to that pixel. Therefore, the leaking pixels will be those with lower values along the object's boundary and the leaking paths will reach all background pixels around the object with costs equal to their leaking pixel values. By connectivity, the rest of the background will be also conquered by leaking paths that pass through the same leaking pixels. As explained above, the automatic identification of these leaking pixels solves the problem (Fig. 1b). Under the same conditions, the watershed transform may fail whenever the heterogeneity of the background results in gradient values similar to those of the

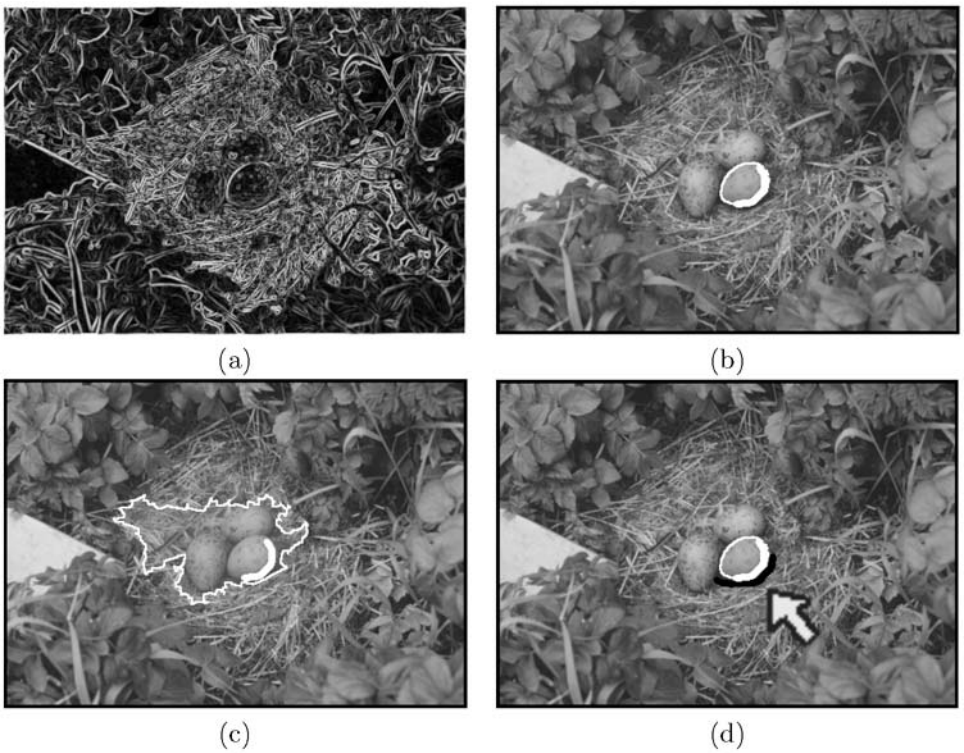

Fig. 1 **a** A gradient-like image where the object is an egg. **b** Segmentation result with tree pruning and the *white marker* as internal seeds. **c–d** Segmentation results with watershed under the same conditions and using the image's border and the *black marker* (indicated by the *arrow*) as external seeds

desired boundary, because the costs of optimum paths from external seeds may saturate before these paths reach the internal ones at the object's boundary (Fig. 1c). This will make the location of the external seeds more important to solve the problem in watershed-based approaches (see arrow in Fig. 1d).

Tree pruning was first presented in [10], with two approaches to detect leaking pixels. One is interactive where the user can visually identify leaking pixels and select them with the mouse pointer. The other is automatic, but relies on a parameter that is difficult to be adjusted in real applications. In [25], we revisited the method to propose an automatic solution for leaking pixel detection, which is free of *ad hoc* parameters, and to provide a comparative analysis of tree pruning and watershed transform for automatic segmentation, including one experiment for license plate segmentation. In this paper, we provide more details in the presentation of the method (formal definition of the obtained objects, different examples, algorithms, sufficient conditions, gradient-like images, and geometrical issues), improve the license plate segmentation method and presentation, and add another application (automatic 3D MR-brain segmentation), in which the methods are compared with a template-based approach widely used for medical research [17].

We give the main definitions and instantiate the IFT for tree pruning and watershed transform in Sect. 2; describe tree pruning with algorithms in Sect. 3; discuss sufficient conditions and geometrical issues in Sect. 4 and gradient-like images in Sect. 5; evaluate the methods in Sect. 6; and state conclusions in Sect. 7.

2 Background

Tree pruning (TP) and watershed (WS) algorithms rely on a *gradient-like image* (gradient image for short), being both approaches extensible to multidimensional and multiparametric images. In several situations, the gradient image can be simply the magnitude of some gradient operator, such as the Sobel's gradient (Fig. 1a). In other situations, it is better to assign an image feature vector to each pixel and compute the gradient image as a function of the differences between feature vectors of adjacent pixels (Sect. 5).

Gradient Condition In WS, it is desirable to have a gradient image with higher pixel values on the object's boundary than inside and outside the object. In TP, the lower pixel values outside the object are desirable only within a small neighborhood around the object's boundary. These gradient conditions are important to understand the methods, but they are not necessary conditions (see Sect. 4).

In the following, we present the image foresting transform and its algorithm for TP and WS.

2.1 Image Foresting Transform

A gradient image $\hat{I}$ is a pair (D_I, I) where $D_I \subset Z^2$ is the image domain and $I(p)$ assigns to each pixel $p \in D_I$

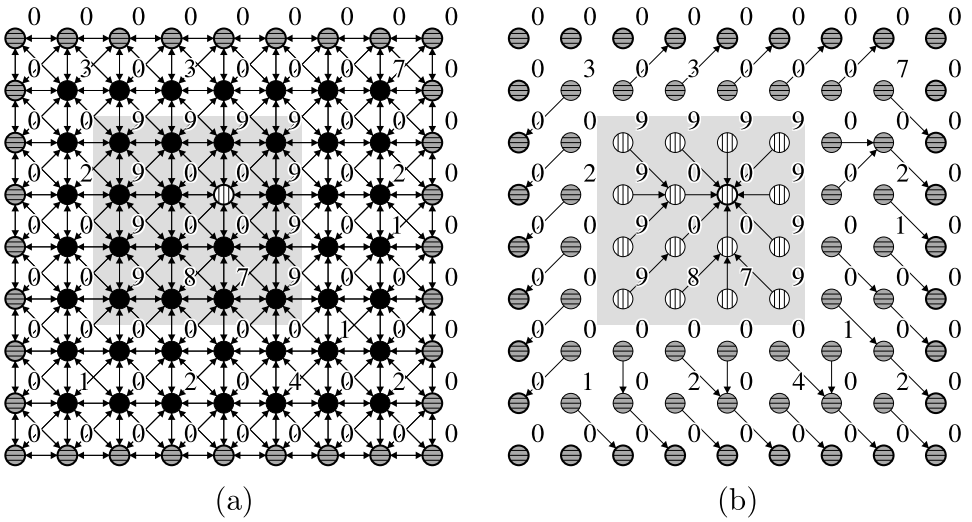

Fig. 2 **a** An 8-connected image graph, where the numbers indicate the pixel values and the object is the *shaded square*. Internal and external seeds for WS are shown with distinct node patterns (*non-black labels*). **b** The optimum-path forest, where the numbers indicate minimum costs and distinct labels separate object and background. The *arrows* link each node to its predecessor in the forest

a scalar value. The gradient image is interpreted as a graph (D_I, A) whose nodes are the pixels in D_I and whose arcs are defined by an *adjacency relation* A between pixels [13]. We are interested in 4- or 8-connected relations for 2D region-based image segmentation (Fig. 2a). A *path* in the graph is a sequence of adjacent pixels and a *path-cost function* c assigns to each path π a *path cost* $c(\pi)$.

Definition 1 (Optimum path) A path π is *optimum* if $c(\pi) \leq c(\tau)$ for any other path τ with the same destination of π.

For both WS and TP, the cost $c(\pi)$ of a path is defined as the *maximum gradient value* of its pixels, when π starts in a set S of seed pixels; and as *infinity cost* otherwise.

$$c(\pi) = \begin{cases} \max_{\forall p \in \pi}\{I(p)\} & \text{if } org(\pi) \in S, \\ +\infty & \text{otherwise} \end{cases} \quad (1)$$

where $org(\pi)$ is the origin of path π. Marker imposition [2, 23, 33] is important in most situations and it is implemented by setting $I(p)$ to 0 for pixels $p \in S$.

The IFT assigns one optimum path (Definition 1) from S to every pixel $p \in D_I$. These paths form an *optimum-path forest* rooted in S which is stored in a *predecessor map* P, such that WS can separate object and background by propagating distinct root labels to their respective trees in the forest (Fig. 2b).

Definition 2 (Optimum-path forest) A predecessor map P is a function that assigns to each pixel $p \notin S$ its predecessor $P(p)$ in the optimum path from S or a marker *nil* when $p \in S$ (in which case p is said to be a root of the forest). An optimum-path forest is a predecessor map which contains no cycles—in other words, one which takes every pixel to *nil* in a finite number of iterations.

The IFT algorithm, as presented next, computes at the same time an optimum-path forest in P and a label map in L, being the former useful for TP and the latter applicable for WS.

2.2 The IFT Algorithm

Let $S = S_o \cup S_b$ be the union of two sets of seed pixels, such that S_o and S_b contain only object and background seeds, respectively. Then, S_b is empty for TP and WS requires S_b not empty.

Algorithm 1 runs in linear time when Q is implemented as described in [14]. Lines 1–3 initialize maps and insert seeds in Q. The main loop computes an optimum path from S to every pixel p in a non-decreasing order of cost (Lines 4–11). At each iteration, a path of minimum cost $C(p)$ is obtained in P when we remove its last pixel p from Q (Line 5). Ties are broken in Q using first-in-first-out (FIFO) policy. That is, when two optimum paths reach an ambiguous pixel p with the same minimum cost, p is assigned to the first path that reached it. The rest of the lines evaluate if the path that reaches an adjacent pixel q through p is cheaper than the current path with terminus q and update Q, $C(q)$, $L(q)$ and $P(q)$ accordingly.

The label propagation in L assigns 1 to pixels that belong to the trees rooted inside the object and 0 to pixels of the trees rooted in the background. In WS, it is expected that the object be defined by image components with label 1, which can be directly obtained from L (Figs. 1d and 2b).

Clearly, WS solves segmentation by seed competition for object and background pixels. It also allows simultaneous multiple object segmentation by modifying Algorithm 1 to propagate a distinct label per object. TP requires the identification of leaking pixels in an optimum-path forest P with no external seeds. The object is obtained by pruning all subtrees rooted in the background (Fig. 1b).

Algorithm 1 Image Foresting Transform for WS and TP

INPUT: Gradient image $\hat{I} = (D_I, I)$, adjacency relation A, seed sets S_o and S_b.

OUTPUT: Optimum-path forest P and label map L.

AUXILIARY: Cost map C, priority queue Q, and variable cst.

1. *For all* $p \in D_I$, *set* $P(p) \leftarrow nil$ *and* $C(p) \leftarrow +\infty$.
2. *For all* $p \in S_o$, *set* $C(p) \leftarrow 0$, $L(p) \leftarrow 1$, *and insert* p *in* Q.
3. *For all* $p \in S_b$, *set* $C(p) \leftarrow 0$, $L(p) \leftarrow 0$, *and insert* p *in* Q.
4. *While* Q *is not empty, do*
5. *Remove from* Q *a pixel* p *such that* $C(p)$ *is minimum.*
6. *For each* q *such that* $(p, q) \in A$ *and* $C(q) > C(p)$, *do*
7. *Compute* $cst \leftarrow \max\{C(p), I(q)\}$.
8. *If* $cst < C(q)$, *then*
9. *If* $C(q) \neq +\infty$, *remove* q *from* Q.
10. *Set* $P(q) \leftarrow p$, $C(q) \leftarrow cst$, $L(q) \leftarrow L(p)$.
11. *Insert* q *in* Q.

The term "subtree of a node" is used in several parts of the text. A subtree of a node p is a tree rooted at a child node q (i.e., $P(q) = p$), which is obtained by removing from P the arc (p, q).

3 Tree-Pruning Segmentation

Figure 3a shows the same image of Fig. 2a, except that the image's border is not used as external marker. Therefore, the optimum-path forest connects object and background through the leaking pixel $(5, 6)$[1] in Fig. 3b. The object can be obtained by removing the subtrees of this leaking pixel.

Note that the IFT algorithm computes optimum paths in a non-decreasing order of costs. Therefore, the optimum paths from S_o will reach object pixels before background pixels whenever the gradient condition for TP (Sect. 2) is satisfied. Moreover, if a pixel p is the only one with lowest value $I(p)$ on the object's boundary, then all pixels around the object will be reached by leaking paths with minimum cost $I(p)$, which pass through the leaking pixel p. By connectivity, the rest of the background will be also conquered by leaking paths that pass through p. When the gradient condition is not fully satisfied, the method may still work (Fig. 4). The same property can be verified when there are multiple leaking pixels, which can be automatically detected for object definition as follows.

[1] Pixel locations are given as (x, y) pairs and the top left pixel is $(1, 1)$.

3.1 Object Definition

Let $\mathcal{R}$ be the set of the roots in the optimum-path forest (Definition 2). By removing the roots of the forest, we get a forest of subtrees. Each tree of this new forest is classified as being either an *object tree* or as a *leaking tree*.

Definition 3 (Object tree) An object tree is a subtree of the root nodes, which is contained within the object.

Definition 4 (Leaking tree) A leaking tree is a subtree of the root nodes, which also contains optimum paths that reach the background (leaking paths).

Figure 3b shows several object trees (e.g., one rooted at pixel $(4, 4)$) and a single leaking tree rooted at pixel $(5, 5)$, each one with a distinct node pattern. Let $B \subset D_I$ be the image's border. We compute the number of descendants that every node of the new forest has in B to obtain a *descendant map* D (Fig. 3c). This combinatorial property of the forest allows the identification of the leaking pixels. The map D is different from the one presented in [10], which computes all descendants in the forest.

Definition 5 (Descendant map) A descendant map is a function D that assigns to each pixel $p \in D_I \backslash \mathcal{R}$ the number of descendants of p in B.

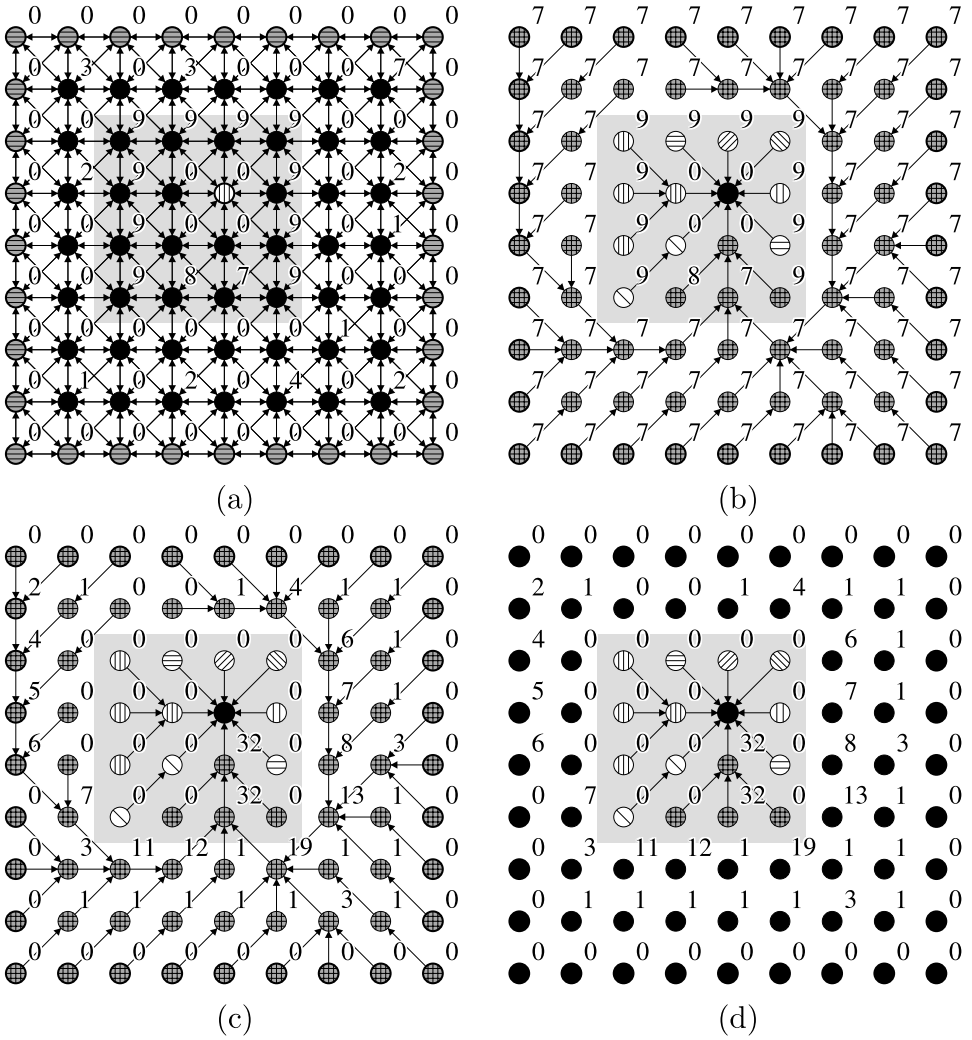

Fig. 3 **a** The same image graph of Fig. 2a. Internal seed and pixels in the image's border B are shown with distinct node patterns. **b** Optimum-path forest in TP, where the numbers indicate minimum costs. Object and leaking trees are shown with distinct node patterns. **c** The numbers indicate the descendant count in B for each pixel, except for the root node. **d** After pruning, the remaining forest defines the object

Each leaking tree is supposed to cross the object's boundary through a single pixel, named leaking pixel, and the paths that reach B must pass through all leaking pixels (Fig. 3b).

Definition 6 (Leaking pixel) A leaking pixel is defined as an object pixel whose successors in the leaking paths belong to the background.

The leaking pixels can be usually detected from the predecessor and descendant maps as the *last one* with the highest descendant value along an optimum path that reaches B. That is, by following backwards any optimum path in P, which has terminal node in B, the leaking pixel is the *first one* with the highest descendant value in the way back to the root of its tree (pixel $(5, 6)$ in Fig. 3c). We can repeat this procedure for all nodes in B to detect all leaking pixels automatically.

This property usually holds at the object's boundary thanks to the FIFO tie-breaking policy of the priority queue Q. After leaking, the gradient condition makes ambiguous the pixels in the neighborhood outside the object, ramifying the leaking path into several branches (Fig. 5). This ramification drastically reduces the descendant count for pixels of the subtrees rooted at the leaking pixels. The object is finally obtained by removing the subtrees of the leaking pixels from the original forest (Figs. 3d, 4f and 5c).

Definition 7 (Object by tree pruning) Let $\mathcal{P}$ be the set of pixels that belong to the subtrees of leaking pixels, $\mathcal{T}_l$ be the set of pixels that belong to the leaking trees, and $\mathcal{T}_o$ be the set of pixels that belong to the object trees. The object is defined as $\mathcal{R} \cup \mathcal{T}_o \cup \{\mathcal{T}_l \backslash \mathcal{P}\}$.

3.2 Algorithms

Algorithm 2 computes the descendant map D (Definition 5) in linear time. It visits all pixels of the forest in reverse breadth-first order, accumulating the number of descendants from the leaf pixels to the root pixels. Lines 1–3 insert the

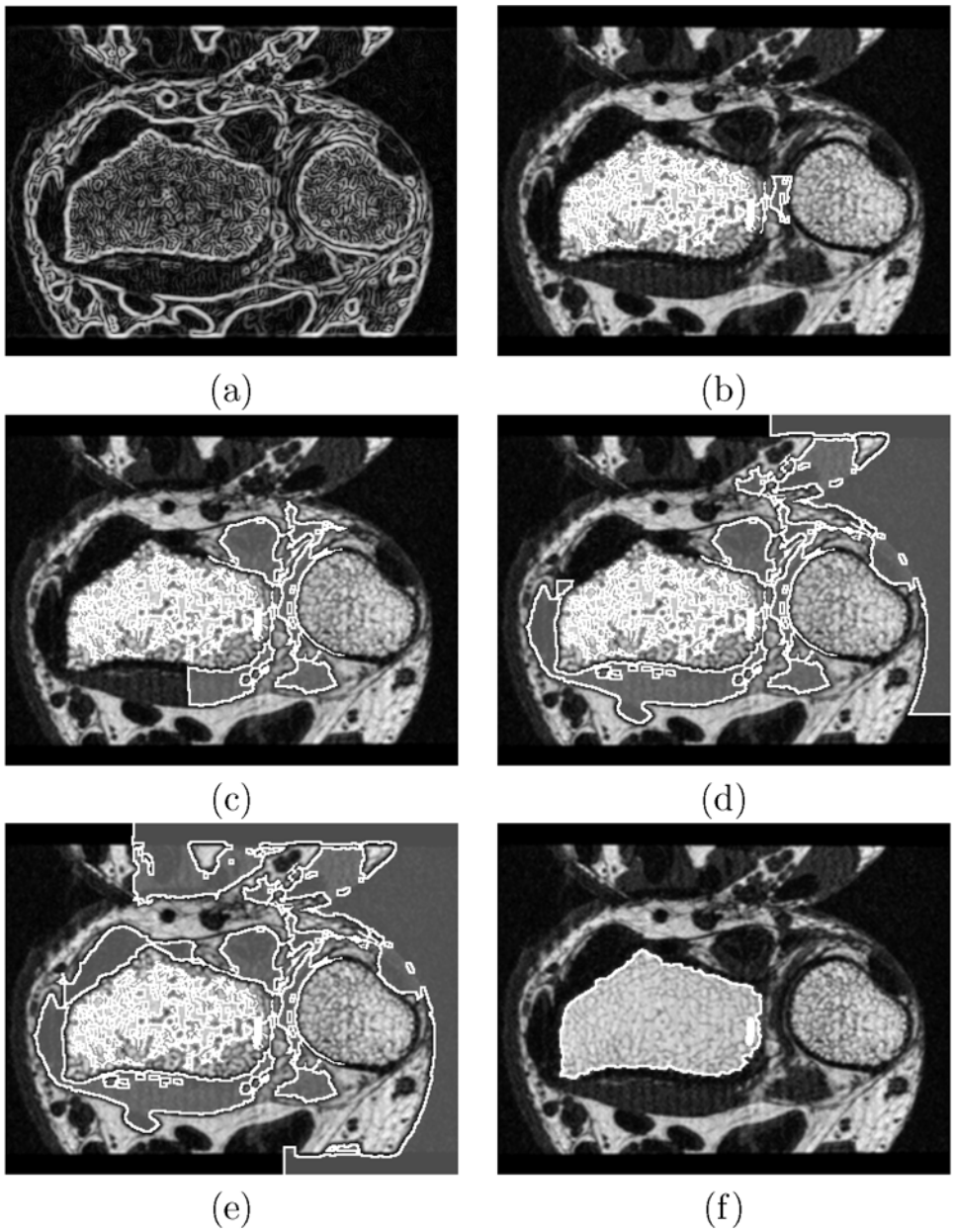

Fig. 4 **a** A gradient image where the object is a bone of the wrist. **b–e** The region growing of the IFT from internal seeds. The leaking occurs before filling the entire object, but these leaking paths surround the object, avoiding further connections between object and background. **f** The object is obtained by automatic leaking pixel detection

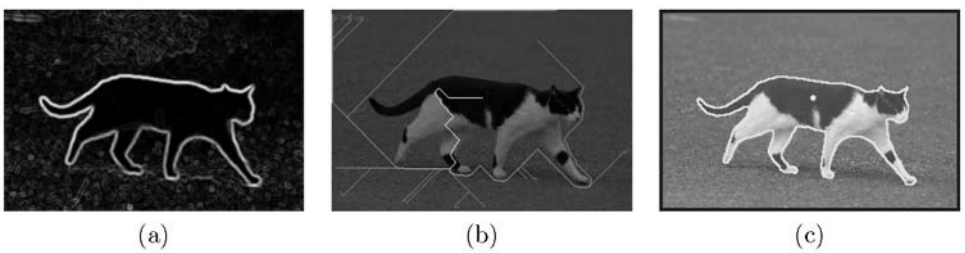

Fig. 5 **a** A gradient image. **b** The original image overlaid by the descendant map. The leaking path ramifies into several branches on the object's boundary, provoking a decreasing in D (the lines become darker) and avoiding further connections between object and background. **c** The resulting segmentation with TP

Algorithm 2 Descendant Map Computation (Linear Time)

INPUT: Optimum-path forest P, adjacency relation A, set B.

OUTPUT: Descendant map D.

AUXILIARY: Queue Q, Stack S.

1. *For all* $p \in D_I$, *do*
2. $\quad$ *Set* $D(p) \leftarrow 0$.
3. $\quad$ *If* $P(p) = nil$, *insert* p *in* Q.
4. *While* Q *is not empty, do*
5. $\quad$ *Remove a pixel* p *from* Q.
6. $\quad$ *Insert* p *in* S.
7. $\quad$ *For each pixel* q *such that* $(p, q) \in A$, *do*
8. $\quad\quad$ *If* $P(q) = p$, *insert* q *in* Q.
9. *While* S *is not empty, do*
10. $\quad$ *Remove a pixel* p *from* S.
11. $\quad$ *If* $P(p) \neq nil$, *then*
12. $\quad\quad$ *Set* $D(P(p)) \leftarrow D(P(p)) + D(p)$.
13. $\quad\quad$ *If* $p \in B$, *set* $D(P(p)) \leftarrow D(P(p)) + 1$.

Algorithm 3 Descendent Map Computation (Alternative)

INPUT: Optimum-path forest P and set B.

OUTPUT: Descendant map D.

1. *For all* $p \in D_I$, *set* $D(p) \leftarrow 0$.
2. *For each pixel* $p \in B$, *do*
3. $\quad$ *Set* $q \leftarrow p$.
4. $\quad$ *While* $P(q) \neq nil$, *do*
5. $\quad\quad$ *Set* $D(P(q)) \leftarrow D(P(q)) + 1$, *and* $q \leftarrow P(q)$.

forest roots in a FIFO queue Q and initialize the descendant count map D. Lines 4–8 traverse the optimum-path forest in breadth-first order, inserting every visited pixel in the stack S. Lines 9–13 use the stack S to visit the forest in reverse breadth-first order, calculating and propagating the descendant count up to the roots of the forest. The resulting map D counts, for each pixel, the number of descendant pixels in the optimum-path forest that belong to B. Algorithm 2 visits each pixel exactly three times, and its running time is $\Theta(|D_I|)$.

Algorithm 3 computes the same descendant map D with a different approach: For each pixel in B, the algorithm follows backwards its optimum-path up to the root, updating the descendants-in-the-border count. In the worst case—an unlikely situation in which all pixels of B belong to a same optimum-path that contains all image pixels—Algorithm 3 will visit each pixel $|B|$ times, leading to $O(|D_I||B|)$ performance. For 2D images, $|B| \propto \sqrt{|D_I|}$, and its running time becomes $O(|D_I|^{\frac{3}{2}})$. For 3D images, $|B| \propto \sqrt[3]{|D_I|^2}$, and its running time becomes $O(|D_I|^{\frac{5}{3}})$.

Algorithm 4 Leaking Pixel Detection

INPUT: Optimum-path forest P, descendant map D, and set B.

OUTPUT: Set L_k of leaking pixels.

```
1. For each pixel p ∈ B, do
2.     Set q ← p, set d_max ← −∞.
3.     While P(q) ≠ nil, do
4.         If D(q) > d_max Then set d_max ← D(q), r ← q.
5.         Set q ← P(q).
6.     Set L_k ← L_k ∪ {r}.
```

In real applications, Algorithm 3 visits much less than $|D_I|$ pixels since most pixels belong to optimum paths that do not reach set B, and thus they are never visited. While Algorithm 2 guarantees linear performance, Algorithm 3 leads to reduced running times in practical applications. Experimental comparison of these algorithms in real 3D segmentation applications showed that Algorithm 3 is 1.8 times faster than Algorithm 2. The implementation of Algorithm 3 is also shorter and simpler than the implementation of Algorithm 2.

The descendant map D is used to detect the leaking pixel associated with each node in B. For every backwards path from B to its root, the first pixel with the highest value in D along the backwards path is a leaking pixel. The set L_k of leaking pixels is computed by Algorithm 4.

4 Sufficient Conditions and Geometrical Issues

This section discusses the main aspects to understand the differences between watershed transform (WS) and tree pruning (TP), their advantages and limitations. In all examples, we use the same gradient image and internal seeds S_o for WS and TP. Additionally, WS uses the image's border as external seeds S_b and TP uses the image's border B to extract the descendant map D. Therefore, the role of the image's border is very different in these approaches.

The gradient conditions are desirable but not necessary conditions (Sect. 2). In WS, it is not difficult to see that the optimum paths from S_o and S_b will meet at the object's boundary, regardless of seed location, whenever the gradient values are strictly higher on the object's boundary than inside and outside the object. Indeed, WS will work even with one external seed and one internal seed. The gradient condition for TP is more relaxed, but its sufficient conditions are:

(C1) All subtrees of leaking pixels must belong to the background.

(C2) Each leaking tree may have at most one leaking pixel.

(C3) Each leaking pixel must have at least two children nodes with descendants in B.

The object by Definition 7 is a set of pixels resulting from the union of the roots, object trees, and leaking trees after removing the subtrees of all leaking pixels (background). By definition, roots and object trees belong to the object. Therefore, conditions (C1)–(C3) are sufficient to guarantee that object pixels will not belong to subtrees of leaking pixels and Algorithm 4 will detect all leaking pixels. If (C1) is violated, then object pixels may belong to the subtree of some leaking pixel (because some leaking path left and then returned to the object or a leaking pixel also belongs to some non-leaking path). Condition (C2) guarantees that Algorithm 4 will not detect the highest descendant count inside the object (a false leaking pixel). If a leaking tree has two or more leaking pixels, then the respective leaking paths must have common ancestor nodes inside the object with descendant count strictly higher than the leaking pixels. However, (C2) does not prevent Algorithm 4 to detect a false leaking pixel outside the object. Then (C3) together with (C2) guarantee that all leaking pixels are detected by Algorithm 4. (C3) implies by induction that all leaking pixels have descendant count strictly higher than any of their descendant nodes. Given that the only entrance to the object is through a leaking pixel, the first highest descendant count is detected on the object's boundary for any backwards path from B. Therefore, conditions (C1)–(C3) guarantee the correctness of the method. On the other hand, if the method segments the object then it is not difficult to prove that conditions (C1)–(C3) hold.

Condition (C1) may hold even when the gradient condition for TP is not satisfied (Fig. 4), but the gradient condition implies in (C1). This can be easily verified because the IFT algorithm computes optimum paths in a non-decreasing order of costs with FIFO tie-breaking policy (Sect. 2.2). This

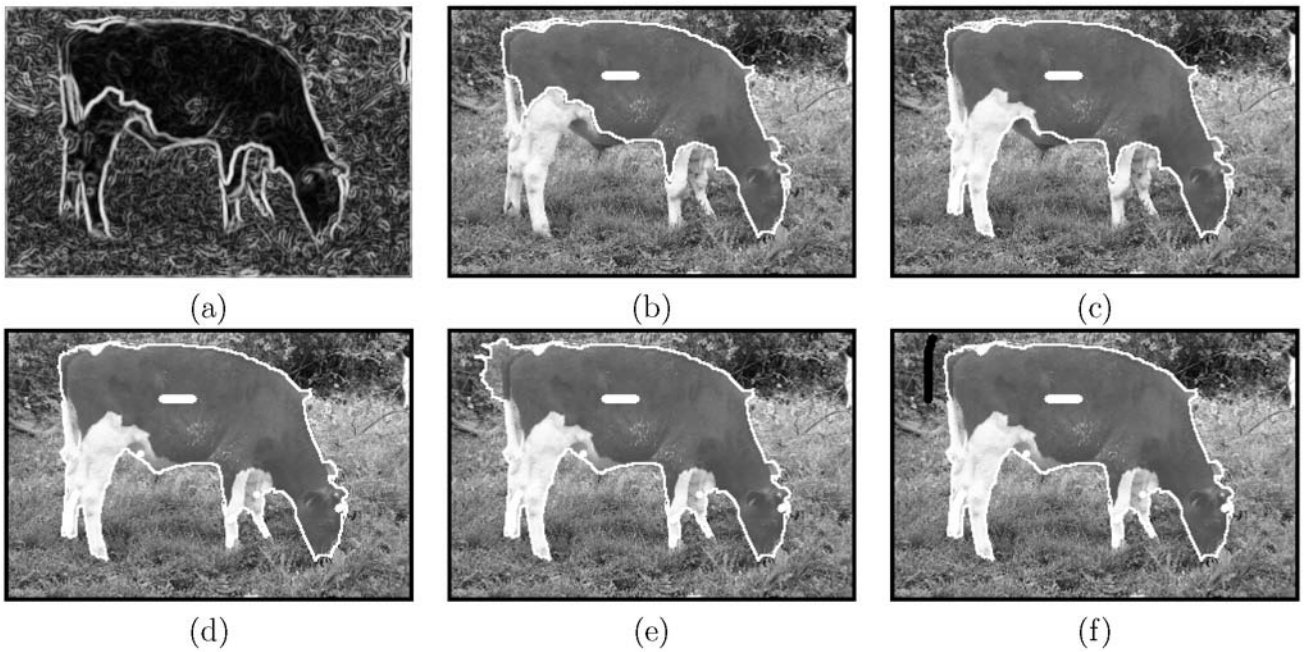

Fig. 6 **a** A gradient image. **b–d** Results of segmentation with TP and incremental seed sets (*white markers*). **e–f** WS requires the image's border and additional external seeds (*black markers*) to work

guarantees that under the gradient condition all object pixels will be reached by optimum paths before the background pixels and pixels on the object's boundary will be reached by optimum paths from interior pixels before other boundary pixels.

In interactive segmentation, we may correct the results of TP by adding seeds to S_o whenever (C1) and (C2) fail. This forces object pixels to be reached before background pixels in (C1) and breaks leaking trees with multiple leaking pixels into object trees and/or trees with a single leaking pixel each. Similarly (C3) can be satisfied when a leaking pixel has no descendants in B, by adding pixels of its subtrees to set B. In the worst case, the predecessor of the leaking pixels are added to S_o and their children nodes are added to B, forcing the first highest descendant value to be at the leaking pixel. In [10], the descendant map highlights all leaking paths and not only those that reach B. This is better for 2D interactive segmentation, because the user can see exactly where the leaking occurs and cut the leaking paths.

Figures 6a–d illustrate the case that requires more internal seeds to satisfy (C1) and (C2) in TP. The same example does not satisfy the gradient condition for WS. Figures 6e–f show that, besides the image's border, additional external seeds are needed in S_b to correct segmentation in WS. Figures 7a–b illustrate the case that requires additional external pixels in B to satisfy (C3). The same example shows in Figures 7c–d that WS may require more external seeds in S_b for correction than the pixels in B.

On the other hand, Fig. 8 suggests that TP may be more sensitive to the location of the internal seeds than WS. TP fails in Fig. 8c due to violation of (C2) (the dot in the bigger fragment is a disk with multiple pixels in B). One seed close to the leaking pixel corrects segmentation.

Note that in the gradient condition for WS, both methods should provide similar results. We will show this experimentally in Sect. 6.2. Condition (C1) is satisfied as described above and the lower gradient values outside the object together with the FIFO tie-breaking policy favor (C3), as explained in Sect. 3.1. In order to favor (C2), either we select seeds close to the leaking pixels or, in the case of automatic segmentation, estimate as large as possible the seed set.

4.1 Heterogeneity of the Background

The examples in Figs. 1, 6, and 7 suggest that TP may be more robust than WS with respect to the heterogeneity of the background. Figure 9a shows a gradient image of a circular object, which satisfies the gradient condition for TP but not for WS. Ambiguous regions are shown in Fig. 9b as white pixels. These regions are plateaus of cost (tie zones in WS) whose pixels are reached by optimum paths with costs greater than or equal to the leaking pixel values. The watershed lines might be anywhere on these plateaus (Fig. 9c). Given that TP does not depend on the costs of the optimum paths outside the object, it is less susceptible to the heterogeneity of the background (Fig. 9d).

The Variant of Maximum Gradient A problem may occur when multiple objects with similar gradient values are very

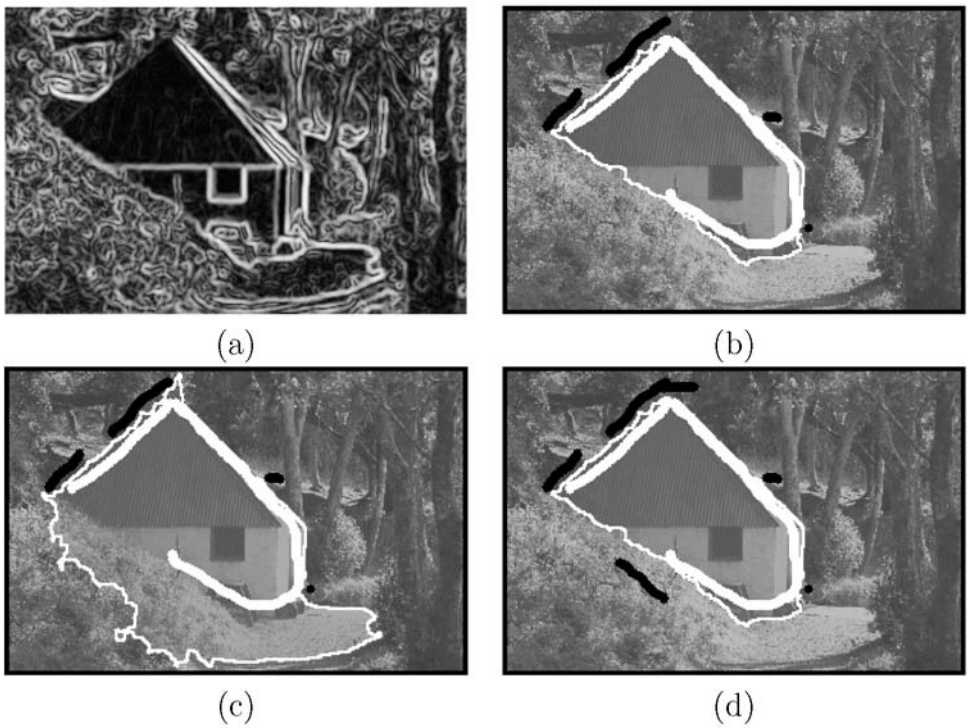

Fig. 7 **a** A gradient image. **b** Resulting segmentation with TP, internal seeds (*white markers*) and additional pixels in B (*black markers*). **c–d** Resulting segmentation with WS and additional external seeds, besides the image's border (*black markers*)

close to each other. The absence of space in between the objects to ramify the path at the leaking pixel may create a false pruning point outside the object (Fig. 10a), failing condition (C3). If the gradient condition for TP is satisfied, we may assume that the correct location of a leaking pixel is always at the maximum gradient value in the path segment between the detected point and its root. We call this *variant of maximum gradient*. Note that it does not affect the location of the true leaking pixels, but solves the problem as illustrated in Fig. 10b. This variant has been used in all examples and experiments of this paper.

4.2 Geometrical Issues

The failure probability of the TP conditions increases with the number of leaking pixels. The competition among seeds might prevent some leaking trees to reach set B, failing (C3), and leaking trees might have multiple leaking pixels, failing (C2). Although the gradient condition for TP is not necessary, these observations and its implication in (C1) indicate that it is at least desirable.

The number of leaking pixels achieves the worst case for perfect boundaries and perfect gaps (Fig. 11), but there are alternative solutions. Figures 11a and 11b show a perfect boundary and a wrong segmentation result with TP. A simple solution is to order the pixels by their gradient values such that pixels with the same intensities are randomly ordered (Figs. 11c and 11d). In perfect gaps (Fig. 11e), we can add internal seeds in S_o and external pixels in B around the gaps to solve segmentation (Fig. 11f). A similar solution is required for WS. However, a more elegant solution is to run an edge detection method, compute the Euclidean distance transform to the edges, and create an edge distance map as the complement of the pixel distances to the edges. By adding the edge distance values to the original gradient value, we obtain a new gradient image in which the number of leaking pixels is drastically reduced (Fig. 11g), solving the problem with perfect gaps (Fig. 11h).

Fortunately, images usually have noise that considerably reduces the number of leaking pixels. Therefore, we never needed to use these solutions in practice.

Condition (C3) may also fail in the case of nested boundaries, being them a problem for WS as well. External (true or false) closed boundaries may prevent leaking paths of internal boundaries to reach set B. The problem can be solved with an alternative pixel in B, which should be selected between the boundaries. Similar solution can be obtained with S_b in WS. In [25], we proposed an alternative solution for TP which iteratively searches the desired boundary (leaking pixel) by matching candidates with a template. This solution goes backwards along the optimum path from the first detected pixel to its root, looking for a next pixel whose gradient value is maximum in the remaining segment. A third and probably best option is to improve gradient computation in order to avoid such situations (Fig. 12).

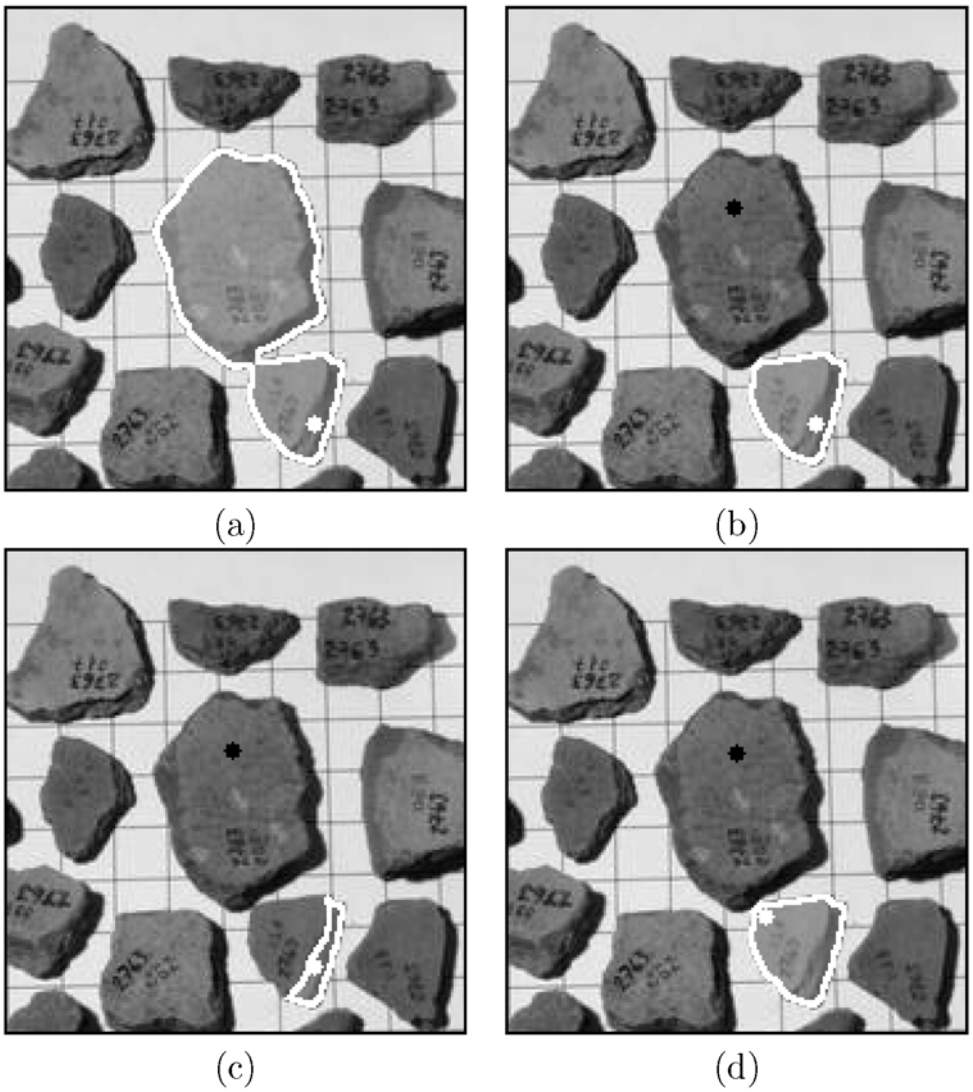

Fig. 8 **a** Result of segmentation with TP. WS obtains similar result with the image's border as external marker. **b** WS segments the object with one additional external seed (*black dot*). **c** TP fails with the same seed selection. **d** TP works when we change the location of the internal seed (*white dot*)

5 Gradient Images

It should be clear that under the gradient condition for WS, TP and WS provide similar results, except for some pixels on the object's boundary because WS divides the boundary between influence zones of internal and external seeds. In order to create suitable gradient images for both methods, we should be able to exploit image features which distinguish object and background.

Let $\vec{f}(p) = (f_1(p), f_2(p), \ldots, f_n(p))$ be a vector at pixel p such that the values $f_i(p)$, $i = 1, 2, \ldots, n$, are the brightness values of p and of its 8-neighbors, for instance. The differences $d_j(p, q_j)$, $j = 1, 2, \ldots, 8$, measure the brightness variations around arcs (p, q_j) between p and each of its 8-neighbors.

$$d_j(p, q_j) = \frac{1}{n} \sum_{i=1}^{n} f_i(q_j) - f_i(p). \tag{2}$$

The vector $\vec{d}_j(p, q_j) = d_j(p, q_j) \frac{q_j - p}{|q_j - p|}$ represents the brightness variation in the direction of (p, q_j). The gradient vector $\vec{G}(p)$ at p is obtained by the sum of its projections on these directions.

$$\vec{G}(p) = \sum_{j=1}^{8} \vec{d}_j(p, q_j). \tag{3}$$

The magnitude of $\vec{G}$ is used as gradient image $\hat{I}$. In the case of colored images (the originals of Figs. 6, 7, 8, and 10), we obtain one gradient image for each channel; red $\hat{I}_r$, green $\hat{I}_g$, and blue $\hat{I}_b$; and compute the final gradient value $I(p) = \max\{I_r(p), I_g(p), I_b(p)\}$ for all pixels $p \in D_I$.

6 Evaluation

TP and WS require some approach for seed selection (object location) in automatic segmentation. We present two appli-

cations in which robust procedures exist for object location: (i) license plate segmentation and (ii) 3D MR-image segmentation of the human brain.

The experiments compare the results with some ground truth, which was obtained by interactive differential watershed segmentation [9] for both applications. Let O and G be the pixel sets that represent segmented object and ground truth. We use three normalized measurements: *false negatives* (FN), *false positives* (FP), and *error* (E).

$$FN = \frac{|G \setminus O|}{|G|}, \tag{4}$$

$$FP = \frac{|O \setminus G|}{|O|}, \tag{5}$$

$$E = \frac{|(O \setminus G) \cup (G \setminus O)|}{|O \cup G|} \tag{6}$$

where $|X|$ is the cardinality of the set X.

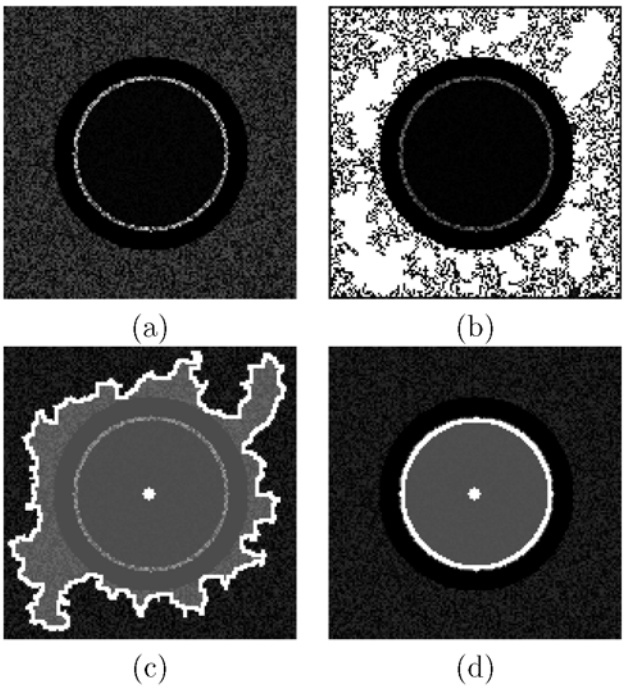

Fig. 9 **a** A synthetic gradient image of a circular object, where the heterogeneity of the background is given by a random external noise. **b** Watershed tie zones in white. **c–d** Respective segmentation results by WS (**c**) and TP (**d**) for a same internal seed (*white dot*)

6.1 License Plate Segmentation

The experiments used 990 images (352 × 240 pixels) from a database of license plates. The goal is to find the precise location and spatial extent of the plates (Fig. 13a). Seed selection is a difficult task, because any attempt to estimate seeds inside a plate is likely to find seeds in other parts of the image. Besides, we need to estimate seed pixels outside the plate numbers to avoid problems with nested boundaries (Fig. 13b).

6.1.1 Seed Selection and Gradient Image

For automatic seed selection, we chose a method proposed by Zheng et al. [36]. This approach is very effective for plate location, but could not be used as baseline for segmentation. The reason is that they ignore shape deformations which do not occur in their database, but are very common in ours (Fig. 14a). The magnitude of the Sobel's operator was chosen as gradient image. This choice illustrates our observation that WS is more dependent of the heterogeneity of the background than TP.

The original image is first enhanced, then vertical edges are extracted using Sobel's operator (Fig. 14b). An edge density map is computed using a rectangular window

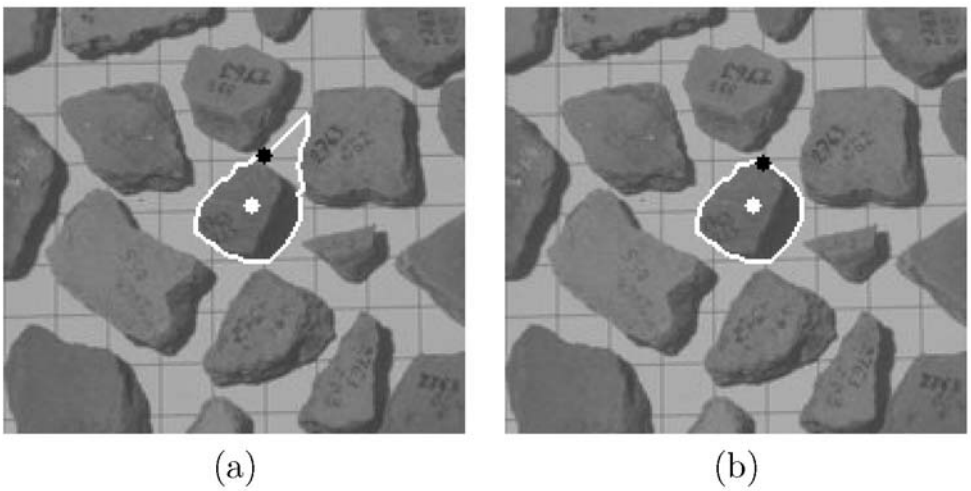

Fig. 10 Image with one object fragment marked for detection. **a** The detected pixel is outside the fragment due to its proximity to the other fragments. **b** The correct leaking pixel is automatically detected using the variant of maximum gradient

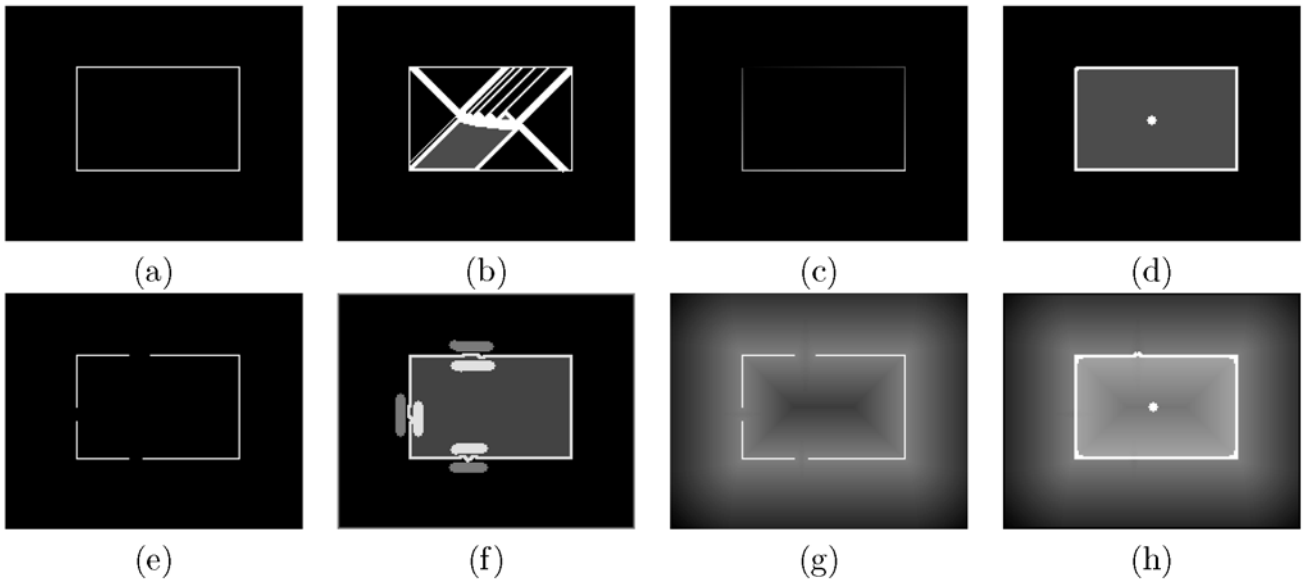

Fig. 11 **a** Synthetic gradient with perfect boundary. **b** TP fails because it missed several leaking pixels. **c** A randomly ordered gradient. **d** TP works on (**c**) with a single seed. **e** Synthetic gradient with perfect gaps. **f** A hard solution using markers around the gaps. **g** A gradient image with edge distance map. **h** TP works on (**g**) with a single seed

Table 1 License plate segmentation: Mean and standard deviation of the average values of error (E), false negatives (FN) and false positives (FP) for each method with respect to the ground truth

Method	Error (E)	False negatives (FN)	False positives (FP)
TP	$2.72\% \pm 0.07\%$	$0.89\% \pm 0.09\%$	$1.90\% \pm 0.10\%$
WS	$4.84\% \pm 0.10\%$	$2.09\% \pm 0.07\%$	$2.85\% \pm 0.11\%$
PTP ($T = 1\%$)	$5.85\% \pm 0.74\%$	$3.34\% \pm 0.76\%$	$2.61\% \pm 0.03\%$
PTP ($T = 3\%$)	$4.15\% \pm 0.07\%$	$0.56\% \pm 0.06\%$	$3.66\% \pm 0.10\%$
PTP ($T = 5\%$)	$5.37\% \pm 0.05\%$	$0.56\% \pm 0.02\%$	$4.90\% \pm 0.06\%$

(Fig. 14c). The center of the plate is expected to be at the highest value in this map.

Seed estimation around the point detected by Zheng's approach is done by "supervised learning". The training set consists of 15% of the plates randomly selected from the database. We compute the average of their binary images (ground truths), with all plates translated to a same reference point (a common geometric center). The pixels with average value 1 are interior pixels in these plates, and so they are assumed to be interior pixels of the remaining plates in the database. To avoid seeds on the border of the plates, their erosion with a disk of radius 1 is used as internal seed set for TP and WS (Fig. 14d). Pixels of the image's border are used as external seeds S_b for WS and set B for TP.

The seed estimation based on Zheng's approach found seed sets inside 93.3% of the plates in our database. We could further improve this result to 96.4% by taking into account the fact that there is no license plate in the top region of the image (30% of the height) in our database.

6.1.2 *Experiments and Results*

We compare the segmentation results of TP, WS, and a previous version of TP [10] (PTP—previous TP), which requires a parameter T for leaking pixel detection. In order to show robustness with respect to the seed estimation process, we executed segmentation 10 times for each method, using a distinct training set each time, and computed mean and standard deviation of the average values of E, FN, and FP (see (4–6)). Table 1 summarizes the results of each method using as test set the images where seed estimation was inside the plates (i.e., 96.4% of the 990 plates).

Note that TP was more accurate than PTP for any fixed value T, because it is difficult to adjust T in this application. Figures 15a–b illustrate cases of errors in seed estimation. In some cases Zheng's method detects a point outside the plate and in other cases the detected point is far from the center of the plate (as consequence, internal seeds are wrongly estimated outside the plate). Table 1 also shows that WS was more sensitive than TP with respect to the heterogeneity of the background (Figs. 15c–d). Figures 15e–i show correct segmentation results using TP under various conditions.

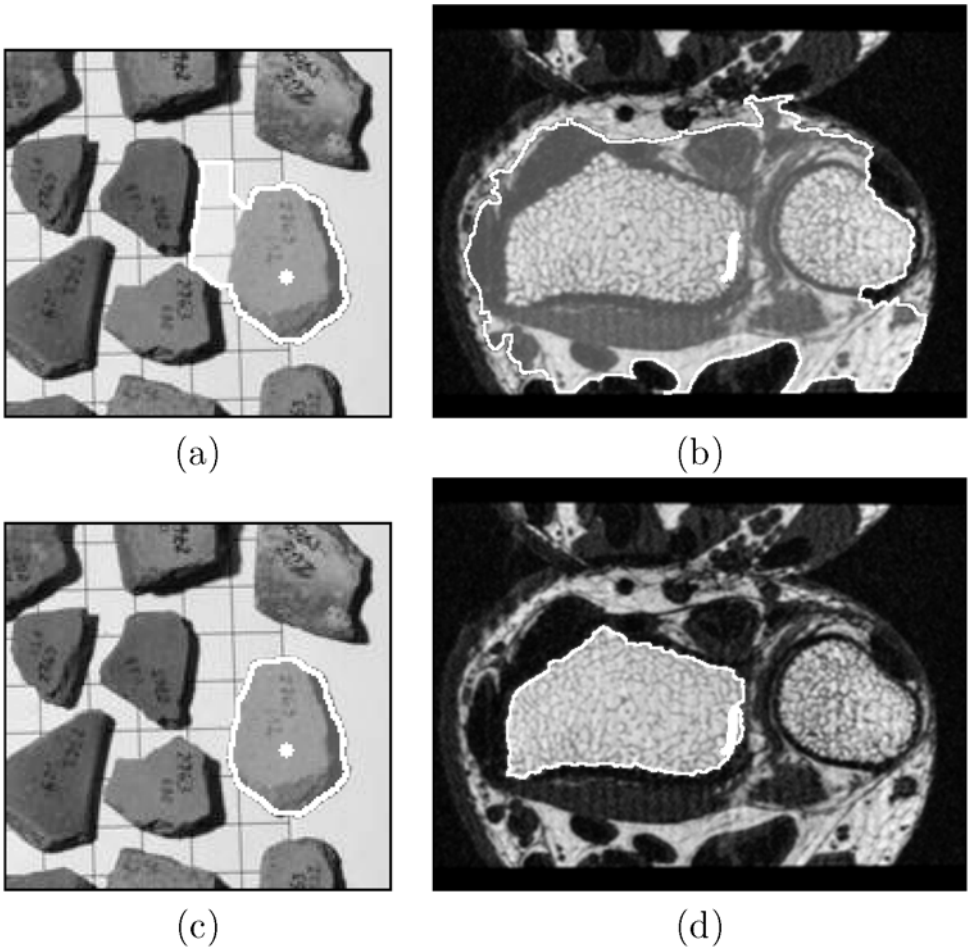

Fig. 12 **a–b** Results of TP for a fragment and a bone of the wrist using the magnitude of the Sobel's gradient and **c–d** the improved gradient presented in Sect. 5

Fig. 13 **a** Original image with the result of TP. **b** The magnitude of the Sobel's gradient

If we consider the maximum error $E = 0.10$ (10%) as an acceptable segmentation, 93% of the 990 plates are detected with TP and 91% are detected with WS.

Deformable models [7, 20] could also be used to segment the plates from the detected location. However, the edges inside and outside the plates might present a local minima problem (Fig. 13b) and the model should take into account shape deformations and imperfect plates (Figs. 15e–h). In this case, it seems at least simpler to use approaches like TP and WS.

In comparison with the TP-based plate segmentation method presented in [25], the current version gives similar results but works faster (55 ms per image on an Athlon XP 2400+) using a more elegant procedure for seed estimation and a single tree pruning execution per image.

6.2 3D MR-Image Segmentation of the Brain

Segmentation of the human brain from Magnetic Resonance (MR) images has been addressed in several different ways, each one with its pros and cons [18]. Figure 16 shows an

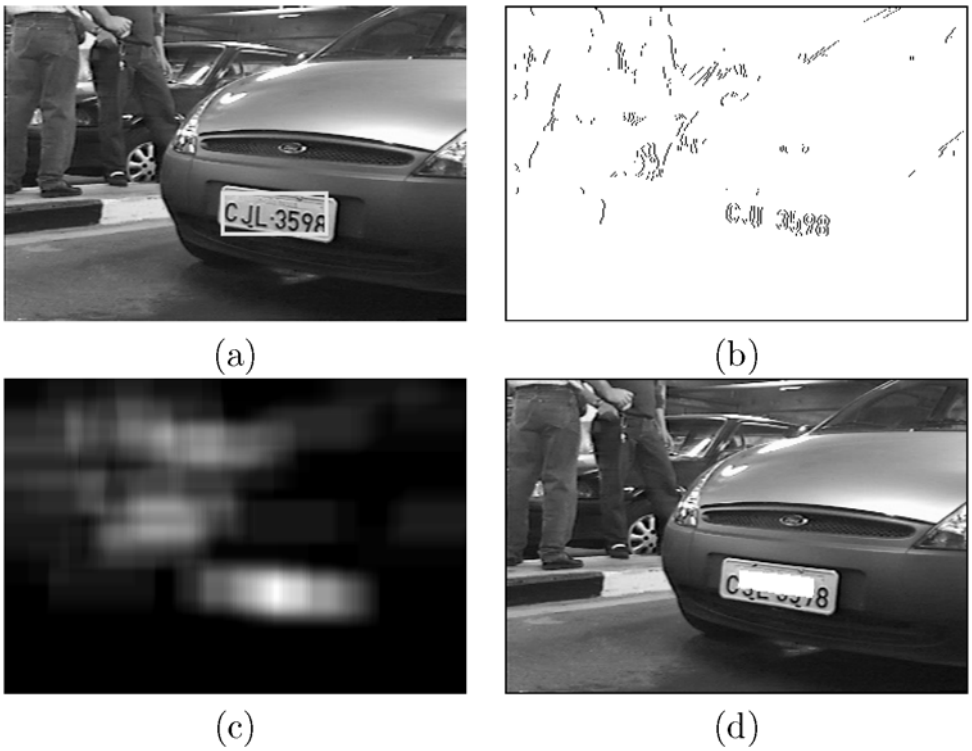

Fig. 14 **a** Original image with the result of the method proposed by Zheng. **b** Vertical edges after filtering. **c** The edge density map. **d** Internal seed set

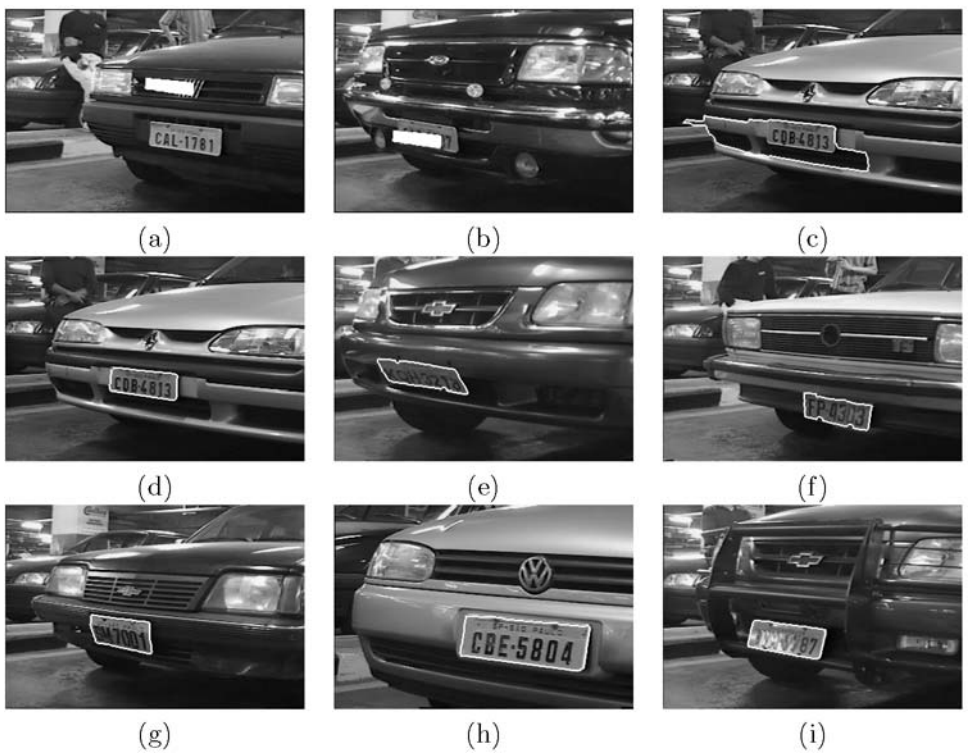

Fig. 15 **a–b** Errors in the seed selection procedure. **c** Watershed fails segmentation. **d** Tree pruning correctly segments the same image in (**c**). **e–i** License plate segmentations by TP under shape distortions, scale changes and lighting effects

MR-T1 slice image of the brain, where some structures and tissues are indicated. The goal is to separate the gray matter (GM) and white matter (WM), as a single object, from the rest of the image. We have evaluated tree pruning using phantom MR-T1 images (which are available at the Brain-Web site[2] [5] together with their ground truth) and real MR-T1 images from control subjects.

[2]URL: http://www.bic.mni.mcgill.ca/brainweb/.

6.2.1 Seed Selection and Gradient Image

Figure 17a shows an MR-T1 slice image of the brain. Note that GM and WM can not be obtained by automatic thresholding on Fig. 17a and simple morphological operations on Fig. 17b. On the other hand, a morphological erosion on Fig. 17b with a sphere of radius 5 mm can assuredly separate GM and WM from other structures, and the largest connected component resulting from erosion can be used as a seed set inside the brain (Fig. 17c). The gradient image is computed as described in Sect. 5, but using a three-dimensional 6-neighborhood instead of the two-dimensional 8-neighborhood (Fig. 17d). This choice illustrates our observation that WS and TP produce similar results when the gradient condition for WS is satisfied.

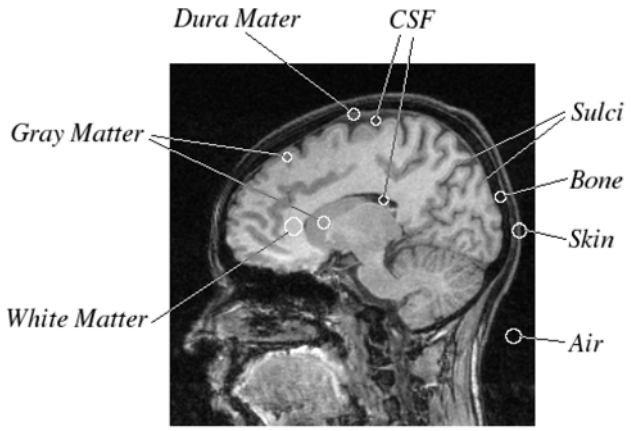

Fig. 16 Tissues in MR-T1 images of the brain. CSF is the cerebrospinal fluid (liquor)

6.2.2 Experiments with Phantoms

On the first set of experiments, we generated 8 MR-T1 phantoms with varying noise and inhomogeneity (INU) settings (Fig. 18), and automatically segmented them with tree pruning (TP) and watershed (WS), using the same internal seeds, gradient image, and the image's border as background seeds for WS and set B for TP. The results were compared with the available ground truth.

Since TP and WS detect the external boundary of the brain, the resulting objects still contain small CSF regions (sulci and ventricles). To properly classify them as background, we take the intersection between the pruned object and the voxels with intensities above the Otsu's threshold.

6.2.3 Experiments with Real MR-T1 Images

On the second set of experiments, we selected 20 MR-T1 images of control subjects with no known anomalies, acquired with a 2T Elscint scanner and voxel size of $0.98 \times 0.98 \times 1.00\ \text{mm}^3$. All volumes were interactively segmented

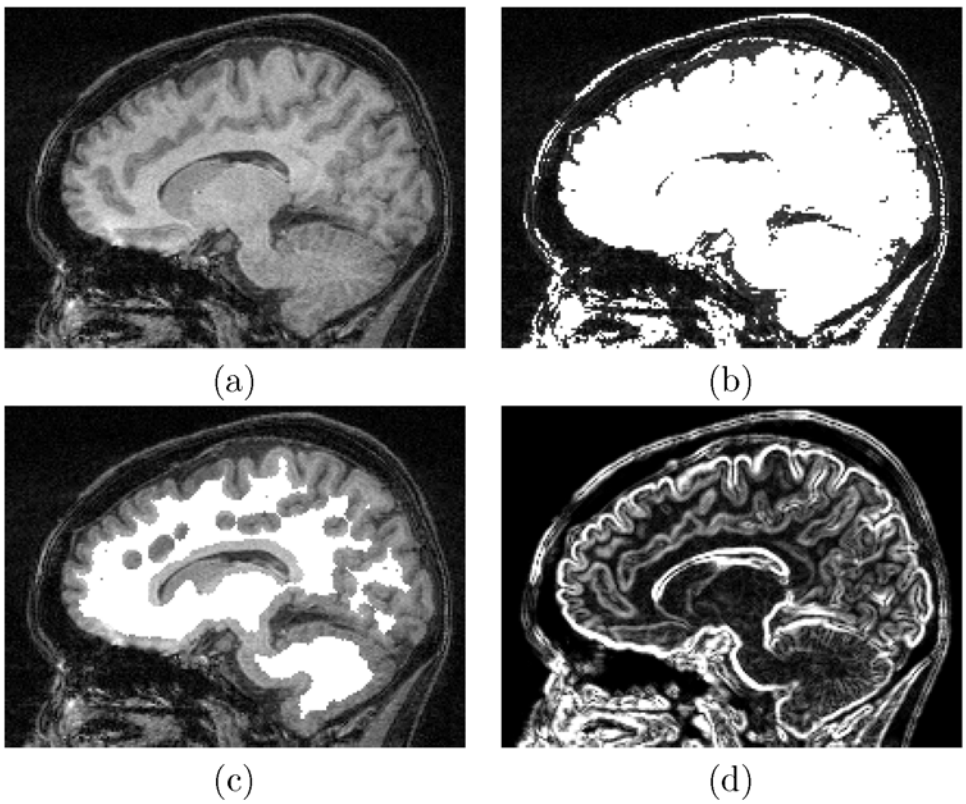

Fig. 17 **a** Sample slice of MR-T1 image of the brain. **b** Binary slice resulting from Otsu's thresholding overlaid on image (**a**). **c** Binary slice of seed voxels overlaid on image (**a**). **d** Slice of the gradient image

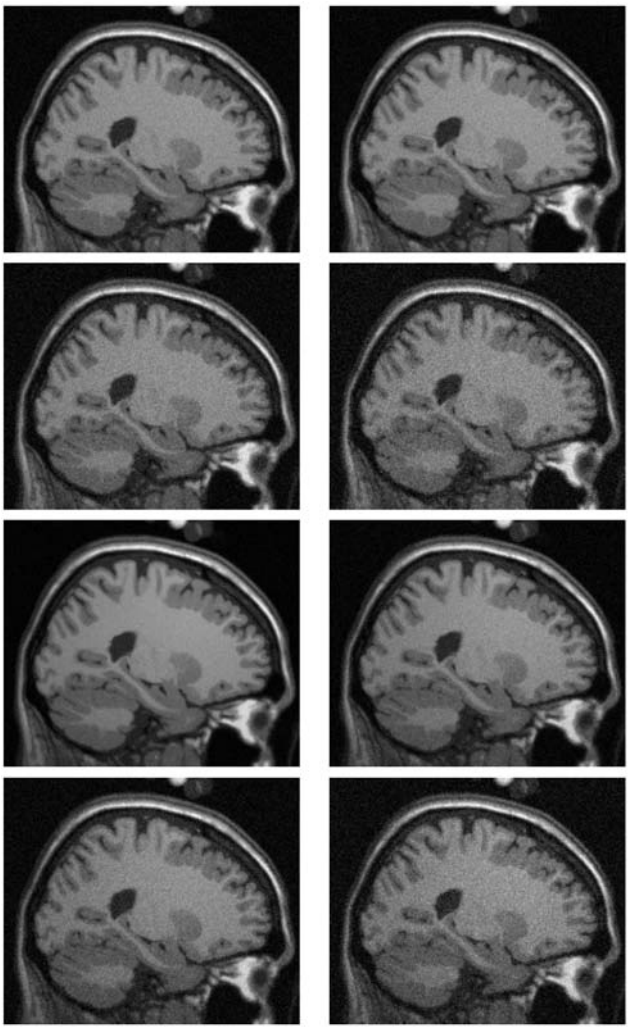

Fig. 18 Sample slices of the phantoms with different degrees of noise (N) and inhomogeneity (INU): **a** $N = 3\%$ and $INU = 20\%$, **b** $N = 5\%$ and $INU = 20\%$, **c** $N = 7\%$ and $INU = 20\%$, **d** $N = 9\%$ and $INU = 20\%$, **e** $N = 3\%$ and $INU = 40\%$, **f** $N = 5\%$ and $INU = 40\%$, **g** $N = 7\%$ and $INU = 40\%$, and **h** $N = 9\%$ and $INU = 40\%$

with differential watersheds [9] in order to provide a basis for comparison (a ground truth). The images were automatically segmented by 3 methods: TP, WS (with the image's border as external seeds), and SPM2 [17]—a widely used template-based approach for medical research.

6.2.4 Results

Table 2 shows the segmentation errors of TP and WS for the phantoms with respect to the available ground truth. Figure 19 shows 3D renditions of the obtained segmentations.

Table 3 shows the segmentation errors for the 3 methods on real images, using the interactive segmentation as the basis for comparison. Figure 20 shows renditions of the segmented brains.

Table 2 Brain phantom segmentation errors with tree pruning and watershed

Noise / INU	Tree pruning	Watershed
	Error (FN,FP)	Error (FN,FP)
3% / 20%	5.16% (3.20%,2.09%)	4.58% (2.53%,2.15%)
5% / 20%	5.03% (3.14%,2.01%)	4.41% (2.55%,1.95%)
7% / 20%	5.26% (3.65%,1.74%)	4.69% (2.96%,1.84%)
9% / 20%	6.00% (4.48%,1.66%)	5.42% (3.85%,1.70%)
3% / 40%	5.26% (3.16%,2.23%)	4.61% (2.45%,2.27%)
5% / 40%	5.18% (3.44%,1.86%)	4.70% (2.73%,2.08%)
7% / 40%	5.67% (4.06%,1.75%)	5.09% (3.38%,1.83%)
9% / 40%	6.68% (5.23%,1.61%)	6.14% (4.66%,1.64%)
Average	5.53% (3.80%,1.87%)	4.96% (3.14%,1.93%)

We can observe the high degree of agreement among TP, WS and the ground truths in both experiments. Note that SPM2 provided segmentations with noticeable artifacts, such as disconnected components outside the brain (Fig. 20). This illustrates how difficult is to segment our images.

TP and WS not only provided better segmentation results than SPM2 on the real images, but also were much faster. The entire segmentation task with TP (marker extraction, gradient computation, optimum-path forest computation, leaking point detection and tree pruning), for example, took 25 seconds for a $181 \times 217 \times 181$ MR-T1 volume, while SPM2 took 5 minutes, on the same Athlon64 3200+ workstation.

The background is homogeneous in most part of these MR-brain images and there are no strong barriers between the brain's border and the image's border. This favored gradient images that satisfy the gradient condition for WS. As we expected, such a situation makes TP and WS to provide similar results. The segmentation differences were mostly the segmentation of excess tissue (false positives) near the cerebellum (as seen in Fig. 20 for subjects 9, 17 and 20) and near the optical nerve (as seen in Fig. 20 for subjects 6, 8, 9 and 19), as well as a thin layer of misclassified voxels over the brain's cortex, indicating off-by-one errors in the location of the object's boundary. These mistakes can be minimized with the design of better application-dependent methods for the computation of the gradient image, with stronger barriers between object and background in the problematic regions. In the case of real images, the slight advantages of WS in performance are also related to the fact that the ground truth was created using the interactive watershed on the same 3D gradient image [9].

Table 3 Real brain segmentation errors with tree pruning, watershed and SPM2

Subject	Tree pruning		Watershed		SPM2	
	Error	(FN,FP)	Error	(FN,FP)	Error	(FN,FP)
01	6.63%	(2.88%, 2.69%)	5.62%	(2.51%, 2.16%)	10.49%	(4.87%, 3.92%)
02	8.79%	(1.62%, 6.94%)	8.28%	(1.42%, 6.75%)	14.56%	(5.85%, 6.77%)
03	7.83%	(2.33%, 4.82%)	7.30%	(1.80%, 5.08%)	13.00%	(4.31%, 7.62%)
04	11.60%	(3.46%, 6.07%)	7.62%	(3.04%, 2.32%)	15.10%	(6.39%, 4.46%)
05	9.85%	(2.34%, 6.71%)	10.24%	(2.06%, 7.74%)	19.51%	(10.21%, 4.04%)
06	9.59%	(2.39%, 6.02%)	8.32%	(2.05%, 5.27%)	16.47%	(4.05%, 10.71%)
07	7.98%	(1.55%, 6.17%)	8.24%	(1.27%, 6.88%)	11.70%	(3.89%, 6.71%)
08	8.37%	(3.36%, 3.75%)	7.05%	(2.87%, 3.12%)	13.20%	(5.37%, 6.15%)
09	9.32%	(2.70%, 5.74%)	9.35%	(2.33%, 6.45%)	12.36%	(5.21%, 5.20%)
10	13.96%	(1.80%,12.21%)	13.46%	(1.56%,12.05%)	15.50%	(5.70%, 7.92%)
11	6.82%	(1.82%, 4.67%)	6.77%	(1.56%, 4.98%)	12.68%	(5.09%, 6.40%)
12	12.88%	(2.22%,10.51%)	12.93%	(1.82%,11.14%)	15.78%	(5.20%, 9.01%)
13	8.51%	(1.98%, 6.22%)	8.12%	(1.56%, 6.47%)	12.41%	(4.62%, 6.56%)
14	8.96%	(3.89%, 4.15%)	8.49%	(3.43%, 4.27%)	12.36%	(6.00%, 5.09%)
15	7.00%	(2.07%, 3.96%)	6.02%	(1.78%, 3.42%)	11.82%	(4.99%, 4.22%)
16	9.71%	(2.45%, 6.44%)	12.61%	(2.20%,10.25%)	14.70%	(5.20%, 7.14%)
17	7.66%	(2.94%, 3.51%)	7.30%	(2.61%, 3.65%)	12.57%	(5.01%, 5.81%)
18	7.15%	(2.81%, 2.62%)	6.41%	(2.61%, 2.08%)	16.43%	(7.92%, 3.93%)
19	9.14%	(1.41%, 7.87%)	8.26%	(1.29%, 7.00%)	13.01%	(4.17%, 8.38%)
20	16.03%	(1.85%,14.71%)	14.31%	(1.70%,12.97%)	18.34%	(3.31%, 14.87%)
Average	9.39%	(2.39%, 6.29%)	8.83%	(2.07%, 6.20%)	14.10%	(5.37%, 6.74%)

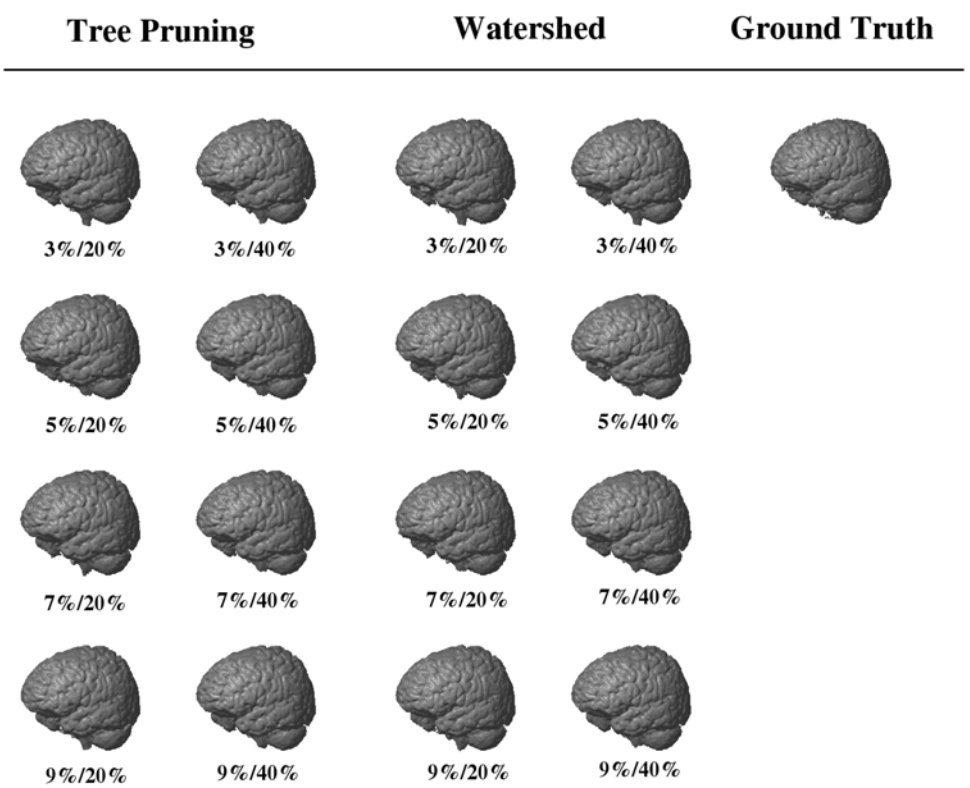

Fig. 19 Renditions of the 8 brain phantom segmentations by tree pruning (*first and second columns*), watershed (*third and fourth columns*) and of the provided ground truth (*fifth column*)

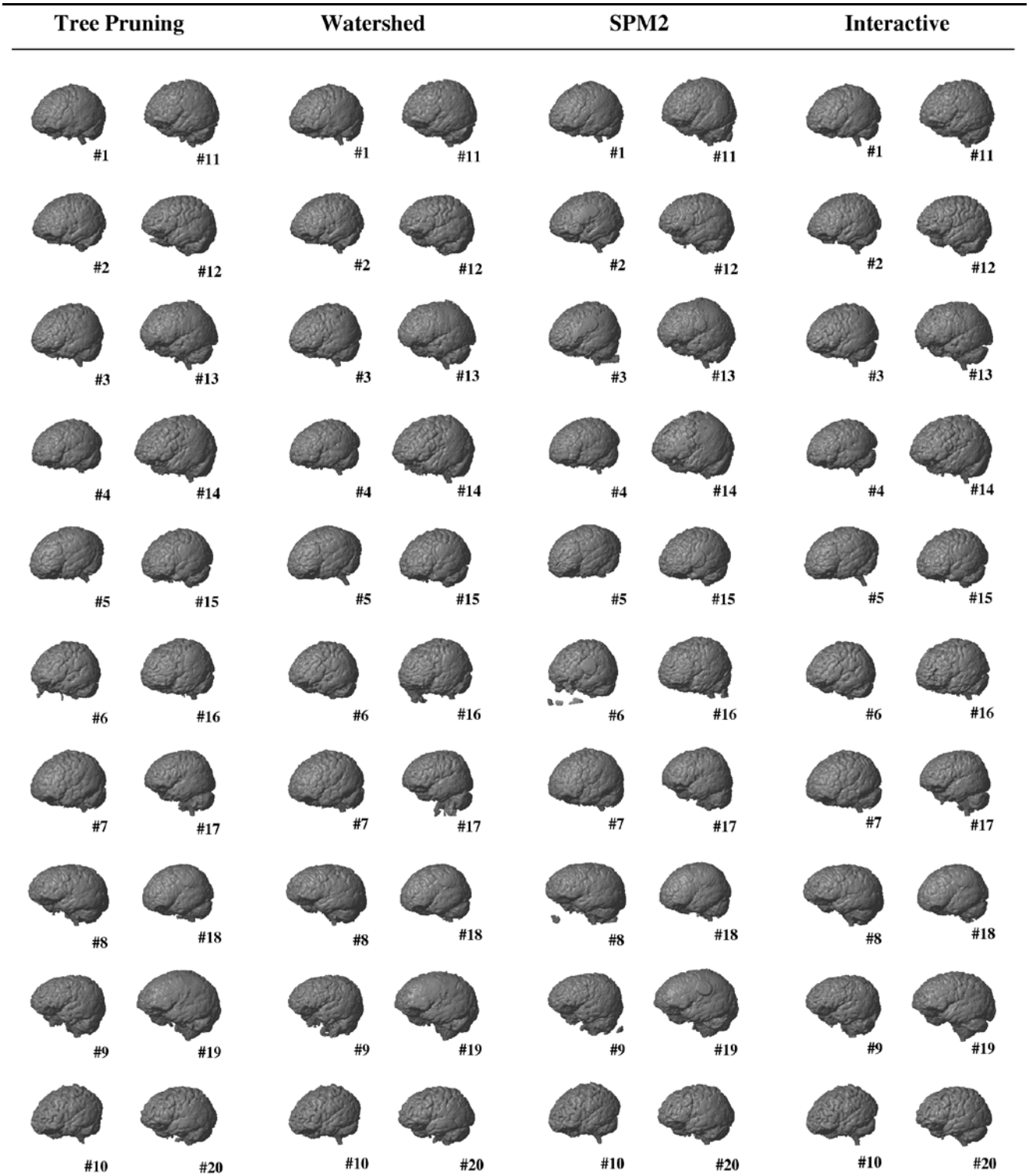

Fig. 20 Brain segmentation results with tree pruning, watershed and SPM2 for 20 control subjects, shown as 3D renditions

7 Conclusion

We have presented a considerably extended version of our previous work on tree-pruning segmentation [10, 25], which adds a formal definition of the obtained objects, several new examples, algorithms, sufficient conditions, geometrical issues, an improved gradient computation with respect to the Sobel's operator, and experiments for auto-

matic license plate image segmentation and 3D MR-image segmentation of the human brain. The experiments show that TP can provide good results in both applications, is less sensitive than WS with respect to the heterogeneity of the background, and that both approaches provide similar results when the gradient condition for WS is satisfied.

We may see image segmentation as consisting of two tightly coupled tasks: *recognition* and *delineation*. Delineation aims to define the precise spatial extent of an object in the image while its approximate location (e.g., seed estimation) is a recognition task. Recognition also involves other cognitive tasks, such as to verify the segmentation correctness or to identify a desired object among candidate ones. While computers usually outperform human beings in delineation, the other way around has been verified for recognition [15].

TP and WS are essentially delineation methods. At same time, model-based approaches [6, 7] have been proposed for recognition and delineation. While they are effective recognition methods, it is usually very difficult to model all possible variations of a given object. In this sense, we believe that methods such as WS and TP, which provide delineation based on optimum criteria, should be combined with object location by model-based approaches for automatic segmentation. Our future work follows in this direction.

Acknowledgements The authors thank Roberto Lotufo (FEEC-UNICAMP) for the images of license plates and Laurent Najman (ESIEE) for his comments on this paper. This work was supported by CNPq (Proc. 302427/04-0), CAPES and FAPESP (Proc. 03/09793-1 and Proc. 03/13424-1).

References

1. Audigier, R., Lotufo, R.A., Falcão, A.X.: 3D visualization to assist iterative object definition from medical images. Comput. Med. Imaging Graph. **30**(4), 217–230 (2006)
2. Beucher, S., Meyer, F.: The morphological approach to segmentation: the watershed transformation. In: Mathematical Morphology in Image Processing, pp. 433–481. Marcel Dekker, New York (1993)
3. Boykov, Y.Y., Jolly, M.-P.: Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images. In: International Conference on Computer Vision (ICCV), vol. 1, pp. 105–112 (2001)
4. Cohen, L.D.: On active contour models and balloons. Comput. Vis. Graph. Image Process. Image Underst. **53**(2), 211–218 (1991)
5. Collins, D.L., Zijdenbos, A.P., Kollokian, V., Sled, J.G., Kabani, N.J., Holmes, C.J., Evans, A.C.: Design and construction of a realistic digital brain phantom. IEEE Trans. Med. Imaging **17**(3), 463–468 (1998)
6. Cootes, T., Edwards, G., Taylor, C.J.: Active appearance models. In: European Conference on Computer Vision (ECCV), vol. 2, pp. 484–498 (1998)
7. Cootes, T., Taylor, C., Cooper, D., Graham, J.: Active shape models—their training and application. Comput. Vis. Image Underst. **61**(1), 38–59 (1995)
8. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification. Wiley-Interscience, New York (2000)
9. Falcão, A.X., Bergo, F.P.G.: Interactive volume segmentation with differential image foresting transforms. IEEE Trans. Med. Imaging **23**(9), 1100–1108 (2004)
10. Falcão, A.X., Bergo, F.P.G., Miranda, P.A.V.: Image segmentation by tree pruning. In: XVII Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI), October 2004, pp. 65–71. IEEE, New York (2004)
11. Falcão, A.X., Costa, L.F., da Cunha, B.S.: Multiscale skeletons by image foresting transform and its applications to neuromorphometry. Pattern Recognit. **35**(7), 1571–1582 (2002)
12. Falcão, A.X., da Cunha, B.S., Lotufo, R.A.: Design of connected operators using the image foresting transform. In: Proc. of SPIE on Medical Imaging, vol. 4322, pp. 468–479 (2001)
13. Falcão, A.X., Stolfi, J., Lotufo, R.A.: The image foresting transform: Theory, algorithms, and applications. IEEE Trans. Pattern Anal. Mach. Intell. **26**(1), 19–29 (2004)
14. Falcão, A.X., Udupa, J.K., Miyazawa, F.K.: An ultra-fast user-steered image segmentation paradigm: live-wire-on-the-fly. IEEE Trans. Med. Imaging **19**(1), 55–62 (2000)
15. Falcão, A.X., Udupa, J.K., Samarasekera, S., Sharma, S., Hirsch, B.E., Lotufo, R.A.: User-steered image segmentation paradigms: live-wire and live-lane. Graph. Models Image Process. **60**(4), 233–260 (1998)
16. Ford, L., Fulkerson, D.: Flows in Networks. Princeton University Press, Princeton (1962)
17. Frackowiak, R.S.J., Friston, K.J., Frith, C., Dolan, R., Price, C.J., Zeki, S., Ashburner, J., Penny, W.D.: Human Brain Function, 2nd edn. Academic Press, New York (2003)
18. Grau, V., Mewes, A.U.J., Alcaniz, M., Kikinis, R., Warfield, S.K.: Improved watershed transform for medical image segmentation using prior information. IEEE Trans. Med. Imaging **23**(4), 447–458 (2004)
19. Haykin, S.: Neural Networks: A Comprehensive Foundation. Prentice-Hall, New York (1998)
20. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: active contour models. Int. J. Comput. Vis. **1**, 321–331 (1987)
21. Kolmogorov, V., Zabih, R.: What energy functions can be minimized via graph cuts. IEEE Trans. Pattern Anal. Mach. Intell. **26**(2), 147–159 (2004)
22. Kuncheva, L.I.: Combining Pattern Classifiers: Methods and Algorithms. Wiley-Interscience, New York (2004)
23. Lotufo, R.A., Falcão, A.X.: The ordered queue and the optimality of the watershed approaches. In: Mathematical Morphology and its Applications to Image and Signal Processing, vol. 18, pp. 341–350. Kluwer, Dordrecht (2000)
24. Lotufo, R.A., Falcão, A.X., Zampirolli, F.: IFT-Watershed from gray-scale marker. In: Proc. of XV Brazilian Symp. on Computer Graphics and Image Processing, October 2002, pp. 146–152. IEEE, New York (2002)
25. Miranda, P.A.V., Bergo, F.P.G., Rocha, L.M., Falcão, A.X.: Tree-pruning: a new algorithm and its comparative analysis with the watershed transform for automatic image segmentation. In: XIX Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI), October 2006, pp. 37–44. IEEE, New York (2006)
26. Nyul, L.G., Falcão, A.X., Udupa, J.K.: Fuzzy-connected 3D image segmentation at interactive speeds. Graph. Models **64**(5), 259–281 (2003)

27. Roerdink, J.B.T.M., Meijster, A.: The watershed transform: definitions, algorithms and parallelization strategies. Fundam. Inform. **41**, 187–228 (2000)
28. Saha, P.K., Udupa, J.K.: Relative fuzzy connectedness among multiple objects: theory, algorithms, and applications in image segmentation. Comput. Vis. Image Underst. **82**, 42–56 (2001)
29. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **22**(8), 888–905 (2000)
30. Torres, R.S., Falcão, A.X.: Contour salience descriptors for effective image retrieval and analysis. Image Vis. Comput. **25**(1), 3–13 (2007)
31. Torres, R.S., Falcão, A.X., Costa, L.F.: A graph-based approach for multiscale shape analysis. Pattern Recognit. **37**(6), 1163–1174 (2004)
32. Udupa, J.K., Samarasekera, S.: Fuzzy connectedness and object definition: theory, algorithms, and applications in image segmentation. Graph. Models Image Process. **58**, 246–261 (1996)
33. Vincent, L., Soille, P.: Watersheds in digital spaces: an efficient algorithm based on immersion simulations. IEEE Trans. Pattern Anal. Mach. Intell. **13**(6) (1991)
34. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Int. Conf. on Computer Vision and Pattern Recognition (CVPR), vol. 1, pp. I-511–I-518 (2001)
35. Wang, S., Siskind, J.M.: Image segmentation with minimum mean cut. In: Int. Conf. on Computer Vision (ICCV), vol. 1, pp. 517–525 (2001)
36. Zheng, D., Zhao, Y., Wang, J.: An efficient method of license plate location. Pattern Recognit. Lett. **26**, 2431–2438 (2005)

Felipe P.G. Bergo received the Computer Engineering degree (2001) and the M.Sc. degree in Computer Science (2004) from the State University of Campinas (Brazil). Since 2004, he is pursuing a Ph.D. in Computer Science at the State University of Campinas. His research involves 3D image analysis, segmentation and visualization, neuroscience, medical image analysis, pattern recognition, and parallel and distributed algorithms.

Alexandre X. Falcão received a B.Sc. in Electrical Engineering (1988) from the University of Pernambuco, PE, Brazil. He has worked in image processing and analysis since 1991. In 1993, he received a M.Sc. in Electrical Engineering from the State University of Campinas, SP, Brazil. During 1994-1996, he worked at the University of Pennsylvania, PA, USA, on interactive image segmentation for his doctorate. He got his doctorate in Electrical Engineering from the State University of Campinas in 1996. In 1997, he developed video quality evaluation methods for Globo TV, RJ, Brazil. He has been Professor at the Institute of Computing, University of Campinas, since 1998, and his research interests include image segmentation and analysis, volume visualization, content-based image retrieval, mathematical morphology, digital TV, medical imaging applications and pattern recognition.

Paulo A.V. Miranda received the Computer Engineering degree (2003) and the M.Sc. degree in Computer Science (2006) from the State University of Campinas (Brazil). Since 2006, he is pursuing a Ph.D. in Computer Science at the State University of Campinas. His research involves image analysis, segmentation, medical image analysis, object recognition, and content-based image retrieval.

Leonardo M. Rocha received the Electrical Engineering degree (2000) from the Federal University of the Ceará (Brazil), and the M.Sc. degree in Computer Science (2002) from the State University of Campinas (Brazil). In 2005 he participated in the committee for the development of the Brazilian digital TV standard. Since 2004, he is pursuing a Ph.D. in Electrical Engineering at the State University of Campinas. His research involves 3D medical imaging and visualization, computer graphics, clustering techniques, digital video segmentation and digital video compression.