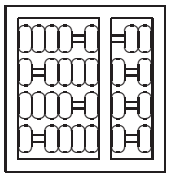


INSTITUTO DE COMPUTAÇÃO
UNIVERSIDADE ESTADUAL DE CAMPINAS

Riverbed: A Novel User-Steered Image Segmentation Method Based on Optimum Boundary Tracking

Paulo A.V. Miranda *Alexandre X. Falcão*
Thiago V. Spina

Technical Report - IC-11-04 - Relatório Técnico

January - 2011 - Janeiro

Riverbed: A Novel User-Steered Image Segmentation Method Based on Optimum Boundary Tracking

Paulo A.V. Miranda Alexandre X. Falcão Thiago V. Spina *

Abstract

This work presents an optimum user-steered boundary tracking approach for image segmentation, which simulates the behavior of water flowing through a riverbed. The riverbed approach was devised using the Image Foresting Transform with a never exploited connectivity function. We analyze its properties in the derived image graphs and discuss its theoretical relation with other popular methods, such as live wire and graph cuts. Several experiments show that riverbed can significantly reduce the number of user interactions (anchor points) as compared to live wire for objects with complex shapes. This work also includes a discussion about how to combine different methods in order to take advantage of their complementary strengths.

1 Introduction

Discrete Mathematics provides an elegant framework for image processing, rich of efficient algorithms with proofs of correctness. As a consequence, many image segmentation methods have been modeled as graph-search problems [38, 3, 6, 15, 28]. In these approaches, a graph derived from the image is computed and the user indicates hard constraints (seed pixels, anchor boundary points) for recognition, while subsequent delineation is performed by the computer in interactive time. These segmentation methods can be classified according to the strategy of representation used to segment the objects as *boundary-based* and *region-based* methods.

Region-based methods can be derived from several core frameworks, such as watershed [39, 34, 12], random walks [40, 20], fuzzy connectedness [36, 38, 10], image foresting transform (IFT) [15, 14], and graph cuts [7, 6]. The IFT framework subsumes many types of watershed and fuzzy-connectedness methods [2, 1, 9]. It can also produce optimum graph cuts, being more efficient and robust to seed position than other graph-cut methods based on the min-cut/max-flow algorithm [28, 9].

Boundary-based methods solve the problem by tracking the edge segments that compose the object's boundary. The idea of transforming the problem of detecting edge segments in a problem of finding the minimum cost path in a weighted and directed graph was

*The authors are with the Department of Information Systems, Institute of Computing - Unicamp, CP 6176, 13084-971, Campinas, SP, Brazil E-mail: {pavm,afalcao}@ic.unicamp.br, tvspina@liv.ic.unicamp.br Phone: +55-19-3521-5881 Fax: +55-19-3521-5847

initially proposed by Martelli [26, 27]. Martelli proposed a solution using the heuristic algorithm A^* [32]. In an attempt to reduce the processing time of algorithm A^*, variations of the Martelli's algorithm using different heuristics were subsequently proposed in the literature [4, 37]. These methods compromised the optimality of the results and could even fail to find any segment between two given points on the object's boundary. Other global approaches for optimal detection of edge segments based on 1D dynamic programming were also proposed [13, 30, 33], but imposing restrictions to the shape of the boundary and limiting the solution of the problem only to some types of applications.

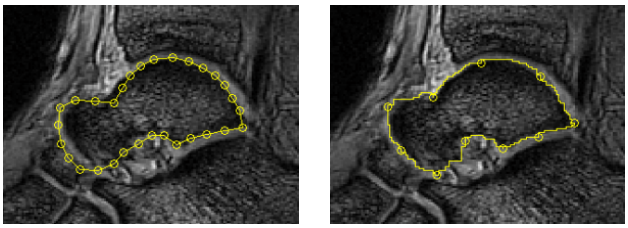
Considering the increased computing power and the development of fast algorithms for solving the problem of minimal cost path, the idea of Martelli resurfaced later in an interactive segmentation method called *live wire* [18, 31]. The solution proposed in this method can be viewed as a 2D dynamic programming that does not impose boundary restrictions and does not dependent on the application, and also as a graph approach that always guarantees an optimal solution. In relation to other methods like *snakes* [23], live wire also provides much tighter control to users since the desired path can be interactively selected from multiple candidate paths.

The live-wire algorithm was further extended in order to ensure real time response regardless the image size, leading to a live-wire-on-the-fly [17]. Later, an extension called G-wire was also proposed which incorporates internal energies on its formulation, resulting in better segmentation in noisy images [22]. The research involving live wire continued in recent years [21, 25, 19]. For example, some recent works [24, 8] have addressed automatic segmentation by combining model-based approaches like Active Shape Models (ASM) [11] with live wire in a synergistic way (Figure 1).

In this work we propose a different boundary tracking approach for image segmentation that simulates the behavior of water flowing through a riverbed. The water crosses the riverbed always seeking lower ground levels, snaking through the river bends, instead of short-cutting the path. A careful analysis of its properties in the graphs derived from the images is presented, as well its theoretical relation with other state-of-the-art methods, such as live wire [17] and graph cuts [6]. Indeed, the riverbed approach can be seen as the boundary-based version of the IFT-SC method [28], which solves region-based segmentation by optimum seed competition, and extends many popular region based segmentation approaches, like *iterative relative-fuzzy connectedness* (IRFC) [10], and *watershed transforms from markers* (WT) [15]. As advantages, riverbed requires fewer anchor points to handle complex shapes as compared to live wire; it can easily incorporate boundary orientation to resolve between very similar nearby boundary segments, unlike IFT-SC; and, as a boundary-based approach, riverbed can also be combined with ASM as proposed in [24, 8] for live wire.

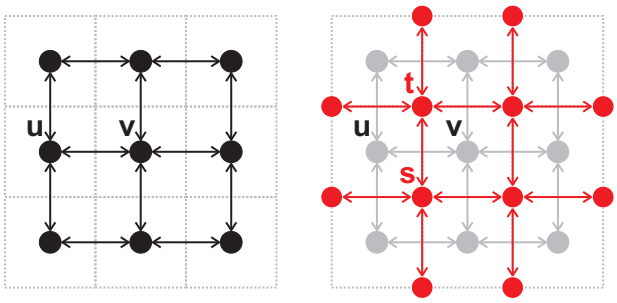
In this work, all methods are presented under the framework of the *image foresting transform* (IFT) [15] — a tool for the design of image processing operators based on *connectivity functions* (path-value functions) in graphs derived from the image. This greatly helps the understanding between the different methods, clarifying their relationships.

For the sake of completeness in presentation, Sections 2 and 3 include an overview of concepts on image graph and a revision of the IFT. Sections 4 and 5 present the related methods IFT-SC and live wire. The riverbed approach is presented in Section 6. We



(a) (b)

Figure 1: Talus of foot in MRI. (a) ASM result with 30 control points. (b) If methods with complementary strengths are combined, the effectiveness of segmentation can be improved (OASM with 10 control points).



(a) (b)

Figure 2: (a) A 4-neighborhood graph, where the pixels are the nodes. (b) A dual graph, where the nodes are the pixel vertices. The arcs (u, v) in (a) have complementary weights $w(u, v) = K - w(s, t)$ with respect to the weights of the corresponding pixel edge (s, t) in (b).

also include comments concerning how to effectively combine the different approaches in Section 7. The experiments and our conclusions are stated in Sections 8 and 9.

2 Basic Concepts on Image Graphs

A multi-dimensional and multi-spectral image $\hat{I}$ is a pair $(\mathcal{I}, \vec{I})$ where $\mathcal{I} \subset Z^n$ is the image domain and $\vec{I}(t)$ assigns a set of m scalars $I_i(t)$, $i = 1, 2, \ldots, m$, to each pixel $t \in \mathcal{I}$. The subindex i is removed when $m = 1$. An image $\hat{I}$ can be interpreted as a graph $(\mathcal{N}, \mathcal{A})$ whose nodes in $\mathcal{N}$ are image elements (e.g., pixels in $\mathcal{I}$ or pixel vertices, as shown in Figure 2) and arcs (s, t) are defined by an *adjacency relation* $\mathcal{A} \subset \mathcal{N} \times \mathcal{N}$. We use $t \in \mathcal{A}(s)$ and $(s, t) \in \mathcal{A}$ to indicate that t is adjacent to s. In this work, we are interested in irreflexive and symmetric relations. For example, one can take $\mathcal{A}$ to consist of all pairs of nodes (s, t) in the Cartesian product $\mathcal{N} \times \mathcal{N}$ such that $d(s, t) \leq \rho$ and $s \neq t$, where $d(s, t)$ denotes the Euclidean distance and ρ is a specified constant (e.g., 4-neighborhood, when $\rho = 1$).

Each arc $(s,t) \in \mathcal{A}$ has a fixed weight $0 \leq w(s,t) \leq K$ which may be computed from local image and object properties extracted from $\vec{I}$ [29], such that lower weights $w(s,t)$ in Figure 2b are assigned to arcs along the object's boundary (or higher weights $w(u,v) = K - w(s,t)$ in Figure 2a are assigned to arcs across the object's boundary).

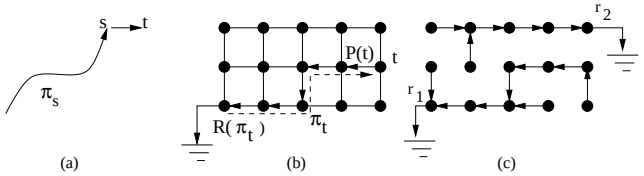


Figure 3: (a) Path $\pi_t = \pi_s \cdot \langle s,t \rangle$ indicates the extension of path π_s by an arc (s,t). (b) A 4-neighborhood graph showing a path π_t (dashed line) represented in backwards, where $P(t)$ is the predecessor node of t and $R(\pi_t)$ is the root pixel. (c) A spanning forest P with two root nodes, r_1 and r_2.

For a given image graph $(\mathcal{N}, \mathcal{A})$, a path $\pi_t = \langle t_1, t_2, \ldots, t \rangle$ is a sequence of adjacent nodes with terminus at a node t. A path is *trivial* when $\pi_t = \langle t \rangle$. A path $\pi_t = \pi_s \cdot \langle s,t \rangle$ indicates the extension of a path π_s by an arc (s,t) (Figure 3a). All paths considered in this work are simple paths, that is, paths with no repeated vertices.

A *predecessor map* is a function P that assigns to each node t in $\mathcal{N}$ either some other adjacent node in $\mathcal{N}$, or a distinctive marker *nil* not in $\mathcal{N}$ — in which case t is said to be a *root* of the map. A *spanning forest* is a predecessor map which contains no cycles — i.e., one which takes every node to *nil* in a finite number of iterations (Figures 3b and 3c, where $R(\pi_t)$ is a root node and $P(t)$ is the predecessor node of t in the path π_t). For any node $t \in \mathcal{N}$, a spanning forest P defines a path π_t recursively as $\langle t \rangle$ if $P(t) = nil$, and $\pi_s \cdot \langle s,t \rangle$ if $P(t) = s \neq nil$.

3 Image Foresting Transform (IFT)

A *connectivity function* computes a value $f(\pi_t)$ for any path π_t, usually based on its arc weights. Let $\Pi(\mathcal{N}, \mathcal{A}, t)$ be the set of all paths in the graph $(\mathcal{N}, \mathcal{A})$ with terminus at t. In this work, a path is *optimum* according to the following definition.

Definition 1 (Optimum path)**.** *A path π_t is* optimum *if $f(\pi_t) \leq f(\tau_t)$ for any other path $\tau_t \in \Pi(\mathcal{N}, \mathcal{A}, t)$.*

By assigning one optimum path with terminus t to each node $t \in \mathcal{N}$, we obtain the optimum-path value $V(t)$, which is uniquely defined by

$$V(t) \quad = \quad \min_{\forall \pi_t \in \Pi(\mathcal{N}, \mathcal{A}, t)} \{f(\pi_t)\}. \tag{1}$$

The *image foresting transform* (IFT) algorithm solves the above optimization problem by dynamic programming [15]. The IFT takes an image $\hat{I}$, a path-value function f and an adjacency relation $\mathcal{A}$; and assigns one optimum path π_t to every node $t \in \mathcal{N}$ such that an *optimum-path forest* P is obtained — i.e., a spanning forest where all paths are optimum. However, the path optimality is only guaranteed for *smooth* functions (Definition 2) as demonstrated in [15]. The attributes of the forest include the map V, the roots $R(\pi_t)$, root labels $L(t)$, and the predecessor $P(t)$ of t in the optimum path. The image operators are then reduced to a local processing of these attributes [15]. For example, region-based operators for segmentation usually rely on the label map L, while the boundary-based operators exploit the predecessor map P, as will be explained later.

Definition 2 (Smooth path-value function)**.** *A path-value function f is* smooth *if for any node $t \in \mathcal{N}$, there is an optimum path π_t which either is trivial, or has the form $\tau_s \cdot \langle s,t \rangle$ where*

(C1) $f(\tau_s) \leq f(\pi_t)$,

(C2) τ_s *is optimum,*

(C3) for any optimum path τ'_s, $f(\tau'_s \cdot \langle s,t \rangle) = f(\pi_t)$.

The connectivity functions are specified by an initialization rule and a path-extension rule. Examples of smooth path-value functions, used for region-based and boundary-based segmentation are $f_{\max}$ and f_{Σ}, respectivelly.

$$\begin{aligned} f_{\max}(\langle v \rangle) &= H(v) \\ f_{\max}(\pi_u \cdot \langle u,v \rangle) &= \max\{f_{\max}(\pi_u), w(u,v)\} \qquad (2) \\ f_{\Sigma}(\langle t \rangle) &= H(t) \\ f_{\Sigma}(\pi_s \cdot \langle s,t \rangle) &= f_{\Sigma}(\pi_s) + w^{\beta}(s,t) \qquad (3) \end{aligned}$$

where $H(t)$ is a handicap value and the parameter $\beta \geq 1$ favors *longer optimum paths* (boundary segments).

3.1 General IFT Algorithm

Algorithm 1 obtains an optimum-path forest P, in which all paths satisfy conditions $(C1) - (C3)$, by minimizing a smooth path-value funtion f.

Algorithm 1. – General IFT Algorithm

Input:	Image graph $(\mathcal{N}, \mathcal{A})$, path-value function f, and an initial labeling function $\lambda(t)$.
Output:	Optimum-path forest P, the minimum path-value map V and label map L.
Auxiliary:	Priority queue Q, variable *tmp*, and an array of *status*.

```
1.  For each t ∈ 𝒩, do
2.      Set L(t) ← λ(t), P(t) ← nil and V(t) ← f(⟨t⟩).
3.      Set status(t) ← 0.
4.      If V(t) ≠ +∞, then insert t in Q.
5.  While Q ≠ ∅, do
6.      Remove s from Q such that V(s) is minimum.
7.      Set status(s) ← 1.
8.      For each t ∈ 𝒜(s), such that status(t) = 0, do
9.          Compute tmp ← f(π_s · ⟨s,t⟩).
10.         If tmp < V(t), then
11.             If V(t) ≠ +∞, then remove t from Q.
12.             Set P(t) ← s, V(t) ← tmp.
13.             L(t) ← L(s) and insert t in Q.
```

Lines 1-4 initialize maps and insert nodes with finite trivial-path values in Q. The minima of the initial map V compete with each other and some of them become roots of the forest. The main loop computes optimum paths from the minima to every node s in a non-decreasing order of value (Lines 5–13). At each iteration, a path π_s of minimum value $V(s)$ is obtained in P when we remove its last node s from Q (Line 6). The rest of the lines evaluate if the path $\pi_s \cdot \langle s,t \rangle$ that reaches an adjacent node t through s is cheaper than the current path π_t in P and update Q, $V(t)$, $L(t)$, and $P(t)$ accordingly.

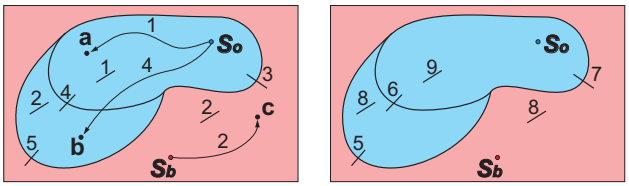
As observed in reference [15], the optimum-path forest may not be unique. For example, if all paths have the same value, then any spanning forest will be optimum. In many implementations of Algorithm 1, ties are usually broken in Q using first-in-first-out (FIFO) policy. That is, when two optimum paths reach an ambiguous node s with the same minimum value, s is assigned to the first path that reached it [15].

4 Segmentation by IFT-SC

A region-based segmentation can be defined by a labeled image $\hat{L} = (\mathcal{I}, L)$, which must satisfy two sets of hard constraints, $\mathcal{S}_o$ and $\mathcal{S}_b$ ($\mathcal{S}_o \cap \mathcal{S}_b = \emptyset$), containing seed pixels selected inside and outside the object, respectively (i.e., $L(u) = 1$ for all $u \in \mathcal{S}_o$ and $L(u) = 0$ for all $u \in \mathcal{S}_b$).

The method referred to as IFT-SC (*IFT segmentation by Seed Competition*) [28] may consider the undirected graph $(\mathcal{N}, \mathcal{A})$ in Figure 2a with weights $w(u,v) = K - w(s,t)$ and with $\mathcal{N} = \mathcal{I}$. Under this paradigm, the connectivity function most commonly used is $f_{\max}$. In this case, $H(v) = -1$, if $v \in \mathcal{S}_o \cup \mathcal{S}_b$, and $H(v) = +\infty$ otherwise (Equation 2). Note that the search for optimum paths is constrained to start in the set of seeds (roots by imposition).

The internal and external seeds compete with each other for their most strongly connected nodes, such that the image graph is partitioned into two optimum-path forests — one rooted at $\mathcal{S}_o$, defining the object, and the other rooted at $\mathcal{S}_b$, representing the background [14]. Both the internal and external forests are encoded in the same predecessor map P returned by the IFT. The segmentation $\hat{L}$ is obtained by Algorithm 1 which is already propagating the root labels to all graph nodes. Hence, we only have to set the initial label $\lambda(v) = 1$ for all $v \in \mathcal{S}_o$ and $\lambda(v) = 0$ for all $v \in \mathcal{S}_b$ before calling the algorithm.



(a) (b)

Figure 4: (a) Schematic graph representation where only some representative arc weights $w(u, v)$ are depicted for each region and border segment. The optimum connectivity values $V(a) = 1$, $V(b) = 4$, and $V(c) = 2$ are also indicated. (b) The graph with complemented weights $\bar{w}(u, v) = K - w(u, v)$, assuming $K = 10$. The boundary segment with arc weights equal to 5 has preference over the segment with weight 6, due to the piecewise optimum property. Note that both solutions lead to the same minimum cut with maximum value 7.

From the graph cut point of view, it was shown that, under the conditions stated in [28] (i.e., fixed labeling to tie-zone pixels), IFT-SC minimizes the graph-cut measure, as defined by Equation 4, among all possible segmentation results satisfying those hard constraints.

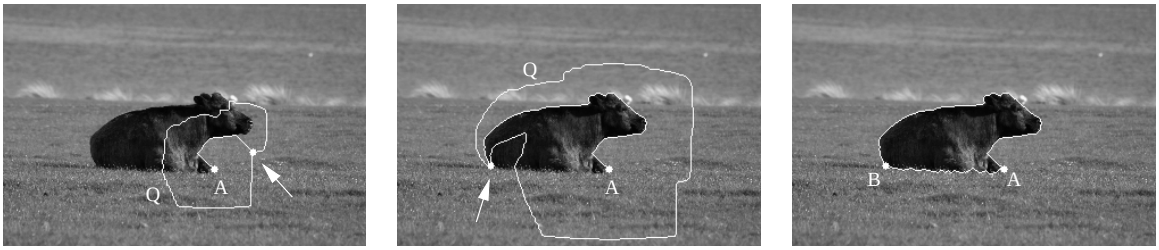
$$E(\hat{L}) \quad = \quad \max_{\forall (u,v) \in \mathcal{A} | \; L(u)=1, L(v)=0} \bar{w}(u, v), \tag{4}$$

where $\bar{w}(u, v) = K - w(u, v)$. In fact, these cut boundaries obtained by IFT-SC are also *piecewise optimum* (Figure 4). That is, the minimization of energy E also applies recursively to all subparts of the boundary, as proved in [28]. In other words, any part of a cut boundary is chosen as one that minimizes its maximum complemented weights $\bar{w}(u, v) = K - w(u, v) = w(s, t)$ of the graph. This will be especially important in the theoretical analysis of the relations with other algorithms.

5 Segmentation by Live Wire

In order to segment the object with live wire [17], the user selects a starting point on the object's boundary (e.g., point A in Figure 5a), and, for any subsequent position of the mouse cursor, the method computes an optimum path from A to that position in real time. As the user moves the cursor close to the boundary, the optimum segment snaps on to it. The user can quickly verify the longest segment, as the one with terminus at point B in Figure 5b, and deposit the mouse cursor at that position. The process is then repeated from B until the user decides to close the contour (Figure 5c).

The closed contour is an optimum curve that is constrained to pass through a sequence $\langle s_1, s_2, \ldots, s_N \rangle$ of N anchor points (seeds) selected by the user on the object's boundary, in that order, starting from s_1 and ending in s_N, where $s_1 = s_N$. The optimum curve that satisfies those constraints consists of $N - 1$ segments $\pi_{s_2}, \pi_{s_3}, \ldots, \pi_{s_N}$, where each π_{s_i} is an optimum path connecting s_{i-1} to s_i. Therefore, we can solve this problem by $N - 1$ executions of the IFT and the optimum contour can be obtained from the predecessor map P after the last execution.



(a) (b) (c)

Figure 5: Contour tracking with live wire. (a) Initial point A is selected on the boundary and the user moves the mouse. (b) A second point B is selected on the boundary. (c) Final contour with 2 segments.

In its simplest form, for $i = 2, 3, \ldots, N$, the IFT for live wire is computed using the initial point s_{i-1} as seed, and path-value function f_Σ with $H(t) = 0$ if $t = s_{i-1}$, and $H(t) = +\infty$ otherwise. Although the method can be used in different graphs, for the theoretical aspects we consider the dual graph with nodes at the pixel's vertices (Figure 2b). We use a slightly modified version of Algorithm 1. At each iteration i, all previous segments $\pi_{s_2}, \pi_{s_3}, \ldots, \pi_{s_{i-1}}$ are kept unchanged during the algorithm, so their nodes can not be revisited or reset during the first loop (Lines 1-4). The only exception is when we compute the last segment, in this case we make $V(s_1) = +\infty$ and $status(s_1) = 0$.

Each IFT execution ($i = 2, 3, \ldots, N$) can also exploit the Bellman's optimality principle [5] for early termination and incremental computation, as proposed in live-wire-on-the-fly (LWOF) [17]. That is, the computation stops when the pixel c, corresponding to the cursor's position, is removed from Q (Line 6), and an optimum-path is shown from s_{i-1} to c. At this moment, a wavefront indicates the pixels in Q (Figure 5). If the user moves the cursor inside the region surrounded by the wavefront, then the respective optimum path is shown with no further computation. As the user moves the cursor outside it (increasing the search region), optimum paths are incrementally computed from the wavefront Q by extending previous optimum paths already computed. This process is repeated until the user selects s_i. We use the LWOF algorithm to implement live wire in this paper.

5.1 Oriented boundaries

The orientedness property in live wire makes it resolve nearby, otherwise very similar, boundary segments. This formulation favors segmentation on a single orientation, but allows longer boundary segments. In this case we consider oriented pixel edges, by using the IFT algorithm with early termination and function $f_\Sigma^{\circlearrowleft}$ which finds optimum paths from a starting point s^* on *counter-clockwise* orientation.

$$
\begin{aligned}
f_{\Sigma}^{\circlearrowleft}(\langle t \rangle) &= \begin{cases} 0 & \text{if } t = s^* \\ +\infty & \text{otherwise} \end{cases} \\
f_{\Sigma}^{\circlearrowleft}(\pi_s \cdot \langle s,t \rangle) &= \begin{cases} f_{\Sigma}^{\circlearrowleft}(\pi_s) + w^{\beta}(s,t) & \text{if } O(u) \geq O(v) \\ f_{\Sigma}^{\circlearrowleft}(\pi_s) + K^{\beta} & \text{otherwise,} \end{cases}
\end{aligned}
$$

where u and v are the pixels at the left and right sides of arc $\langle s,t \rangle$ (Figure 2), and O is a reference map expected to be brighter inside the object. Another option based on a dot product with the gradient vector is discussed in [29].

6 Riverbed

The idea of this new boundary tracking approach is to simulate the behavior of water flowing through a riverbed. The water crosses the riverbed always seeking lower ground levels, snaking through the river bends, instead of short-cutting the path. The prime moving force of water is gravity. Therefore, at any instant of the algorithm, its decision of where to go, does not depend on the past history. The water will always tend to flow down the slope. This leads to the following connectivity function for a starting seed point s^*:

$$
\begin{aligned}
f_w(\langle t \rangle) &= \begin{cases} 0 & \text{if } t = s^* \\ +\infty & \text{otherwise} \end{cases} \\
f_w(\pi_s \cdot \langle s,t \rangle) &= w(s,t)
\end{aligned} \tag{5}
$$

That is, at any moment the IFT algorithm with f_w will move through the arc with lowest weight $w(s,t)$. This algorithm with all the features discussed in Section 5 (i.e., previous segments kept unchanged, early termination, and incremental computation) results in the riverbed approach. Figures 6a-f demonstrate its advantages and disadvantages in relation to live wire.

Following the ideas first described in [18], we can also extend the riverbed to include the orientedness property by using connectivity function $f_w^{\circlearrowleft}$. This favors segmentation in *counter-clockwise* orientation, allowing longer boundary segments (Figure 7).

$$
\begin{aligned}
f_w^{\circlearrowleft}(\langle t \rangle) &= \begin{cases} 0 & \text{if } t = s^* \\ +\infty & \text{otherwise} \end{cases} \\
f_w^{\circlearrowleft}(\pi_s \cdot \langle s,t \rangle) &= \begin{cases} w(s,t) & \text{if } O(u) \geq O(v) \\ K & \text{otherwise.} \end{cases}
\end{aligned}
$$

Although function f_w is not smooth (Definition 2), and consequently the computed paths are not necessarily optimum according to Definition 1, riverbed ensures optimum boundary segments from another point of view which encloses the main features of IFT-SC, as described next.

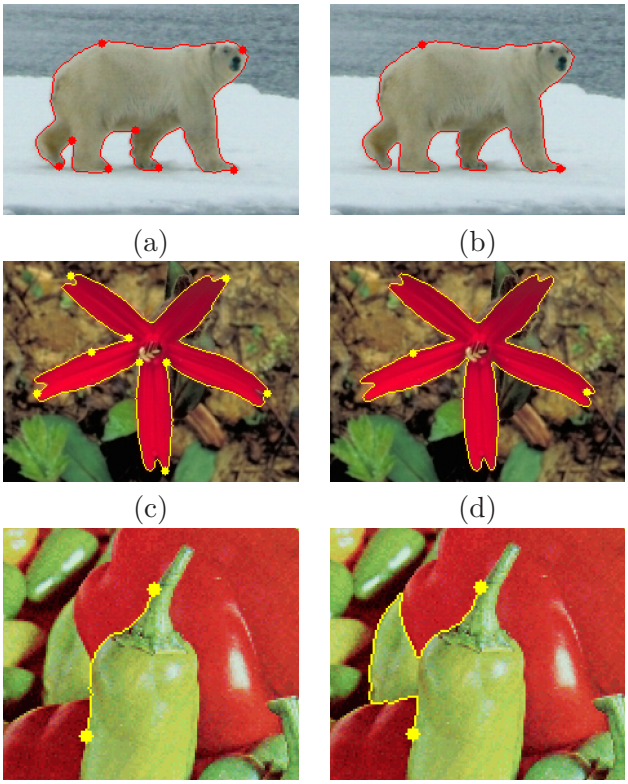


Figure 6: Live wire (first column), and riverbed (second column) results. (a-d) Riverbed requires fewer anchor points to handle complex shapes. (e-f) On the other hand, live wire favors shortest-distance jumps across regions where the boundary is not well defined.

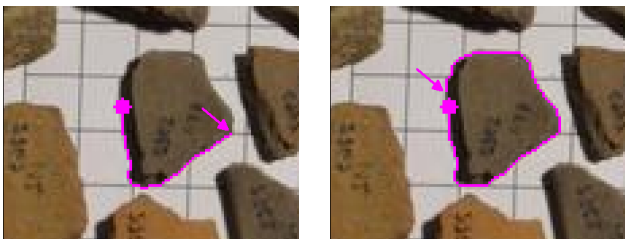


Figure 7: The orientedness property favors longer boundary segments on a single orientation. (a) The longest segment using riverbed with f_w from an initial point. (b) Riverbed from the same initial point can actually delineate the desired contour, by using $f_w^{\circlearrowleft}$.

Given that each pair of adjacent pixels (u, v) in the graph of Figure 2a has a corresponding pair of pixel vertices (s, t) (the edge shared by u and v) in the dual graph of Figure 2b, traversing the object's boundary along the arcs (s, t) is the same of cutting the corresponding arcs (u, v). In this sense, live wire with f_Σ on the dual graph of Figure 2b can be seen as a boundary-based version of the min-cut/max-flow segmentation on the graph of Figure 2a using $\bar{w}(u, v) = w(s, t)$ as weights [6]. Similarly, riverbed with f_w on the dual graph can be seen as a boundary-based version of the IFT-SC method [28] on the graph of Figure 2a.

Theorem 1 (Duality between riverbed and IFT-SC)**.** *Let IFT-SC obey the conditions stated in [28] (i.e., fixed labeling to tie-zone pixels). The riverbed approach with f_w on the dual graph of Figure 2b traverses arcs that minimize recursively the same energy corresponding to the graph-cut measure that IFT-SC presents on the regular graph of Figure 2a. Their difference concerns on how the seeds are interpreted, and on their dynamic of execution: IFT-SC finds a global optimum using the seeds as regional constraints, while riverbed performs successive energy minimization between the ordered pairs of anchor points that act as boundary hard constraints.*

Proof. From Section 4, we know that any part of a cut boundary obtained by IFT-SC, under the stated conditions, is chosen as one that minimizes recursively its maximum complemented weights $\bar{w}(u, v) = w(s, t)$. Thus, riverbed must always traverse a path on the dual graph, such that in any part of this path the method minimizes recursively the maximum arc weight $w(s, t)$ along it. Figure 8 shows an example illustrating that only riverbed has indeed this property.

Let π_1 and π_2 be two arbitrary boundary segments with common origin and destination nodes over the dual graph, and let E_1 and E_2 be their maximum weights respectively. In order to prove the aforementioned statement, we must show that the choice of riverbed is always made in a manner consistent with the piecewise optimum property discussed in Section 4.

Without loss of generality, let's assume that $E_1 > E_2$. Hence, E_1 is greater than any arc weight along π_2. Since the arcs with lower weight are removed first from the priority queue Q in the IFT algorithm using f_w, then π_2 is necessarily traversed before π_1. The usage of the array of status in Algorithm 1 guarantees that this choice will not change.

Since these segments are generic, and it is not assumed that they start at any seed, we have then that the assertion is true. Note that, in the case of function $f_{\max}$, the optimum value from the seed may saturate before reaching the boundary segments at issue (i.e., π_1 and π_2) leading to inconsistent path selection (Figure 8d). □

In [28], it has been shown that the results of the min-cut/max-flow algorithm [6] approximate the IFT-SC's results when the weights are raised to an increasing power tending to infinity. From the above considerations we may conclude that, in theory, by raising the parameter β in function $f_\Sigma^{\breve{\cup}}$, the result of live wire will approximate the riverbed outcome. However, an exact match may not be possible because multiple contours may comply with

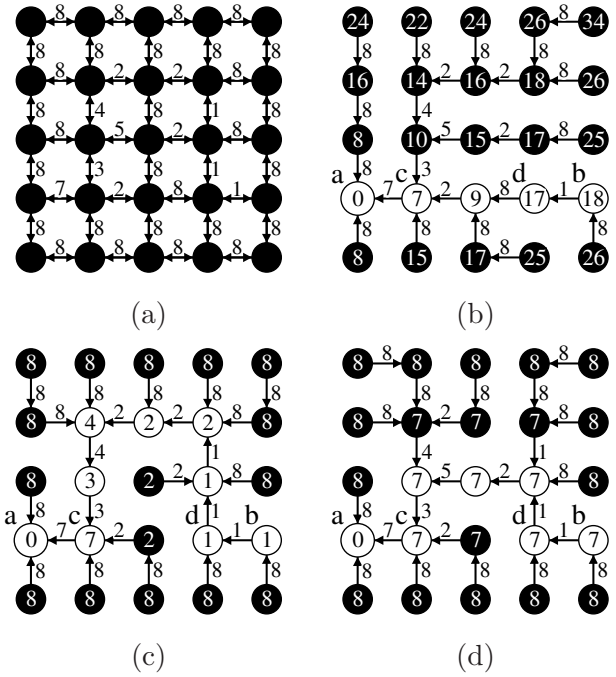


Figure 8: (a) A dual graph with arc weights $w(s,t)$. (b) An optimum live-wire segment with f_Σ from seed a to node b (paths are shown in backwards). The optimum connectivity values $V(t)$ are indicated inside the nodes. (c) A riverbed segment with f_w from a to b. It minimizes the maximum arc weight in any part of this path (e.g., from c to d). (d) This can not be verified for $f_{\max}$ with $w(s,t)$ and $H(a) = 0$.

(a) (b) (c)

Figure 9: (a) Live wire requires many anchor points on complex shape parts. (b) Riverbed gets stuck on poorly defined boundaries, demanding user intervention. (c) The combined result requires only two segments (live wire in cyan, riverbed in red).

the riverbed formulation for some images. Moreover, in practice, high power values create overflow problems in linear time implementations of the queue, which constrains the power value, besides the extra computational burden.

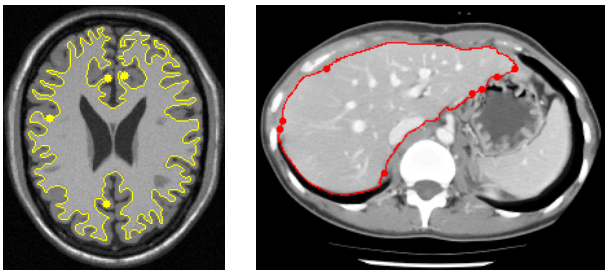
7 Multiple Intuitive User Inputs & Methods

The use of multiple intuitive user inputs to better reflect the user's intention has drawn attention in recent studies [40]. Since riverbed and IFT-SC are boundary and region dual versions of the same energy optimization, as demonstrated in Section 6, they can be used in conjunction in order to accomplish this goal.

A simple and effective way to combine these methods is given as follows: Given that the image graph is usually computed based on object properties extracted from markers selected inside and outside the object [29], it is more natural to start with the region based approach (i.e., IFT-SC). The user can then compute optimum segments using riverbed in the dual graph. By removing the corresponding arcs (u, v) of theses segments from the first graph (Figure 2a), the IFT-SC method will naturally converge to the same final result.

This same procedure may be adopted to combine methods based on different energies (e.g., Riverbed & min-cut/max-flow; live wire & IFT-SC). This combination is even more interesting, because it allows the user to take advantage of methods with complementary strengths. For example, live wire may be used to effectively correct the leaking problem which is common in the IFT-SC approach. On the other hand, riverbed can help to circumvent the bias toward small boundaries of the min-cut/max-flow.

Another very promising solution is simply to combine a sequence of optimum segments computed by live wire and riverbed. This allows us to explore the advantages of both methods. That is, riverbed contributes to reduce anchor points along complex shapes, while the live wire can better solve badly defined borders (Figure 9).



(a) (b)

Figure 10: (a) External white-matter boundary by riverbed with $f_w^{\circlearrowleft}$ requires only four anchor points, while live wire would require almost one point for each more pronounced shape corner. (b) Sample slice of a CT image showing in highlight a boundary by riverbed for the liver.

8 Evaluation

From the discussion of Figure 6, we expect that the advantages of riverbed over live wire, or vice-verse, will depend on the characteristics of the application. Hence, outside the context of a specific application, we can not say, in definitive, that one method is better than the other. In this section we present the results of three experiments involving homogeneous and heterogeneous image datasets. Arc weights were computed as proposed in [29] and β was set to 1.5 in f_Σ, and $f_\Sigma^{\circlearrowleft}$.

Table 1 presents mean and standard deviation of the Dice measures (accuracy) and the number of anchor points (efficiency) resulting from the segmentation of two distinct objects. In the first experiment, the object was the white-matter tissue in 20 MRT1-images of phantoms (available at the BrainWeb site[1]). This experiment demonstrated that riverbed can be far more effective than live wire when objects have complex shapes (Figure 10a). Indeed, this was already expected given our previous discussion. However, the second experiment involved the liver, an object with regular shape, in 22 CT-images from 10 distinct subjects (Figure 10b). In this case, the accuracy was much higher in both methods, being slightly better for riverbed. In regard to efficiency, riverbed required a considerably lower number of anchor points than live wire with f_Σ (a reduction factor of about 53%-60%). Its efficiency gain over live wire with $f_\Sigma^{\circlearrowleft}$ was less pronounced (a reduction factor of about 27%-37%). The lower performance of live wire in this case was due to the presence of stronger nearby boundaries (i.e., lower arc weights) from other objects. Riverbed was less affected by this problem, offering better adhesion force to the desired boundary. Clearly, the orientedness property was more important to live wire than to riverbed.

A third experiment was conducted using a heterogeneous dataset with 22 natural images from the public GrabCut dataset [35] (Figure 11). In this case, we also combined the methods, as proposed in Section 7, involved two different users, and implemented riverbed with an improvement for the case of poorly defined boundaries. That is, when the user moves

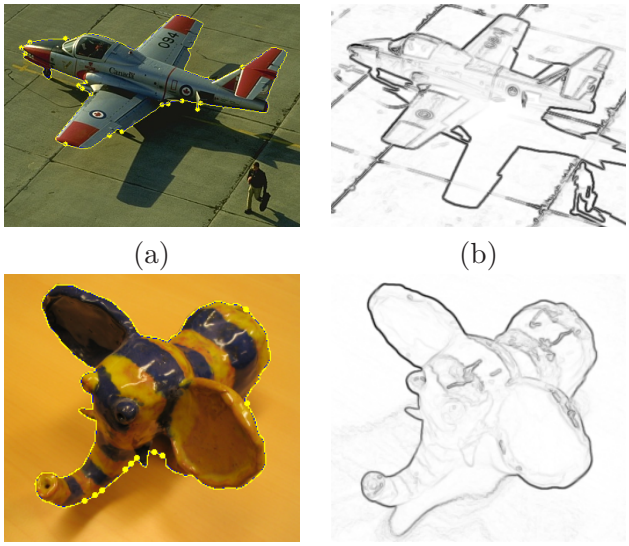
[1]URL: http://www.bic.mni.mcgill.ca/brainweb/

White matter (MRI-T1 phantoms)				
	Live wire		Riverbed	
	f_Σ	$f_\Sigma^\circlearrowleft$	f_w	$f_w^\circlearrowleft$
Dice (%)	μ=95.00 σ=0.93	μ=95.02 σ=0.88	μ=95.15 σ=1.02	μ=95.14 σ=0.99
anchor points	μ=35.85 σ=12.77	μ=32.30 σ=10.38	μ=4.65 σ=1.18	μ=3.75 σ=0.91
Liver (CT)				
Dice (%)	μ=98.03 σ=0.51	μ=98.22 σ=0.35	μ=98.31 σ=0.34	μ=98.28 σ=0.37
anchor points	μ=17.73 σ=2.85	μ=11.23 σ=2.60	μ=8.23 σ=2.41	μ=7.09 σ=1.77

Table 1: Mean (μ) and standard deviation (σ) of the Dice similarity and number of anchor points for all methods.

the mouse's cursor too close to the previous anchor point (e.g., less than 15 pixels), he/she gives an indication that the method is having difficulties to follow the correct boundary. In this case, we build a duct for the water to flow through a straight line, keeping the user's control over the segmentation process. The results are presented in Table 2, which also includes some boundary-based metrics. The mean Euclidean distance error between the segmented and the ground-truth boundaries is given by $\overline{ED}$. This measure is further divided into $\overline{ED}_{fp}$ and $\overline{ED}_{fn}$, which consider the false positive and false negative errors separately (i.e., $\overline{ED} = \overline{ED}_{fp} + \overline{ED}_{fn}$).

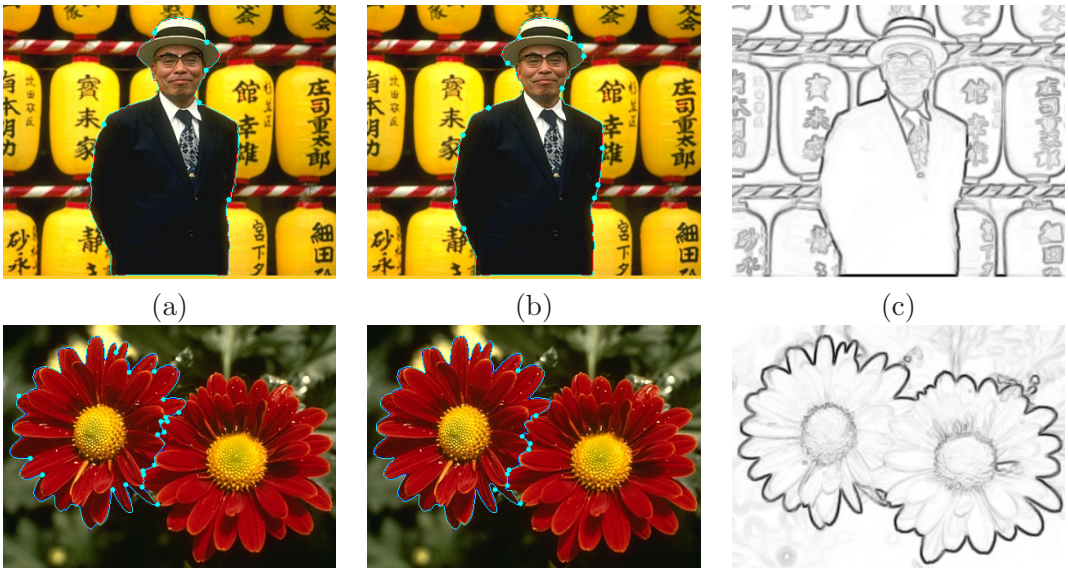
Note that, all methods obtained high accuracy with mean boundary errors less than one pixel. These errors were uniformly distributed, with false positives predominating over false negatives. This is a consequence of the simpler graph model adopted from [29], which, although being sufficient for this application, has less precision than the formulation based on the dual graph (Figure 2b). In regard to the arc weights, they were computed by a supervised process in a step before delineation, as recommended in [29]. Clearly, the user #1 conducted a more careful arc-weight assignment than the second user, as we may note by the higher accuracy and fewer anchor points used during the subsequent boundary delineation. In the images with good arc-weight assignment, riverbed usually prevailed over live wire because the latter may still be affected by complex shapes. For the remaining images, the analysis is more complex. Some images were better segmented by riverbed and others by live wire, demonstrating that they are truly competitive approaches (Figure 12). Although the mean values did not show a very high advantage for the combined approach (i.e., riverbed & live wire) in relation to riverbed, we note that for some images it provided a reduction factor of about 6 anchor points with respect to the best individual approach. Hence, it might be very useful depending on the database composition.



(a) (b)

(c) (d)

Figure 11: Riverbed results on the GrabCut dataset. (a,c) The yellow and blue borders are the representation of the segmentation and ground-truth labels, respectively. (b,d) The corresponding weight assignment used.



(a) (b) (c)

(d) (e) (f)

Figure 12: (a,d) Live wire results, (b,e) riverbed results, and (c,f) the weight assignment used. On the first row, live wire uses fewer seeds (7 against 13), and on the second row, riverbed prevails (10 against 20). The cyan and red/blue borders are the segmentation label and ground-truth, respectively.

User #1			
	Live wire $f_{\Sigma}^{\bigcirc}$	Riverbed $f_{w}^{\bigcirc}$	Riverbed & Live wire
Dice (%)	μ=99.05 σ=0.40	μ=99.11 σ=0.37	μ=99.09 σ=0.40
anchor points	μ=12.32 σ=7.43	μ=9.14 σ=6.86	μ=7.91 σ=6.55
$\overline{ED}$	μ=0.546 σ=0.078	μ=0.526 σ=0.083	μ=0.533 σ=0.076
$\overline{ED}_{fp}$	μ=0.431 σ=0.093	μ=0.434 σ=0.090	μ=0.438 σ=0.092
$\overline{ED}_{fn}$	μ=0.115 σ=0.094	μ=0.092 σ=0.068	μ=0.095 σ=0.070
User #2			
	Live wire $f_{\Sigma}^{\bigcirc}$	Riverbed $f_{w}^{\bigcirc}$	Riverbed & Live wire
Dice (%)	μ=98.67 σ=1.15	μ=98.94 σ=0.42	μ=98.90 σ=0.44
anchor points	μ=13.86 σ=7.19	μ=11.32 σ=8.01	μ=9.00 σ=6.16
$\overline{ED}$	μ=0.654 σ=0.182	μ=0.640 σ=0.174	μ=0.655 σ=0.179
$\overline{ED}_{fp}$	μ=0.495 σ=0.222	μ=0.509 σ=0.205	μ=0.512 σ=0.218
$\overline{ED}_{fn}$	μ=0.159 σ=0.142	μ=0.132 σ=0.116	μ=0.143 σ=0.127

Table 2: Mean (μ) and standard deviation (σ) of several quantitative metrics for 22 natural images.

9 Conclusion

We have presented a new user-steered boundary tracking approach for image segmentation, which simulates the behavior of water flowing through a riverbed. The method, named "riverbed", was developed in the framework of the *image foresting transform* (IFT) [15], by showing for the first time the importance of non-smooth connectivity functions.

We have presented a comparative analysis of riverbed with its most natural competitive method, live wire [17], and proved its duality with the region-based IFT-SC method [28], which provides image segmentation by optimum internal and external seed competition. We have also showed that all these methods provide an optimum graph cut segmentation, according to some suitable measure. The experiments used several images from different datasets, involving objects with complex and simpler shapes. The results indicated that riverbed is a truly competitive approach, which can provide significantly better results than live wire.

Riverbed can be used for photo, video, and medical image segmentation. In the last two cases, segmentation is performed in a frame-by-frame or slice-by-slice fashion. This is actually the most suitable way for objects in video, but it may be interpreted as a limitation of the method for the case of medical images, where exist other 3D approaches and the IFT already provides very efficient interactive segmentation tools via 3D IFT-SC [14]. On the other hand, one can further investigate 3D extensions of riverbed as it has been done for live wire [16, 25]. Besides, in several cases, the slice-by-slice segmentation of medical images may be desirable or very effective from the practical point of view. It is also easier to incorporate internal energies on a purely 2D formulation similarly to what was done in G-wire approach [22]. Hence, as future work, we intend to investigate possible combinations between riverbed and model-based approaches for automatic segmentation, as it has been successfully done in medical imaging [24, 8].

Acknowledgment

The authors thank FAPESP (2009/16428-4, 07/52015-0, and 2009/11908-8), and CNPq (481556/2009-5, 201732/2007-6 and 302617/2007-8) for the financial support. The public domain images of Figures 6a-d have been obtained from the National Digital Library `http://www.fws.gov/digitalmedia`.

References

[1] R. Audigier and R.A. Lotufo. Seed-relative segmentation robustness of watershed and fuzzy connectedness approaches. In *Proceedings of the XX Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI)*, pages 61–68, Belo Horizonte, MG, Oct 2007. IEEE CPS.

[2] R. Audigier and R.A. Lotufo. Watershed by image foresting transform, tie-zone, and theoretical relationship with other watershed definitions. In *Proceedings of the 8th Intl. Symposium on Mathematical Morphology and its Applications to Signal and Image Processing (ISMM)*, pages 277–288, Rio de Janeiro, Brazil, Oct 2007. MCT/INPE.

[3] X. Bai and G. Sapiro. Distance cut: Interactive segmentation and matting of images and videos. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, volume 2, pages II – 249–II – 252, San Antonio, Texas, 2007.

[4] D.H. Ballard and C.M. Brown. *Computer Vision*. Prentice-Hall Inc., Englewood Cliffs, NJ, USA, 1982.

[5] R. Bellman. *Dynamic Programming*. Princeton University, 1957.

[6] Y. Boykov and G. Funka-Lea. Graph cuts and efficient N-D image segmentation. *International Journal of Computer Vision*, 70(2):109–131, 2006.

[7] Y.Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images. In *International Conference on Computer Vision (ICCV)*, volume 1, pages 105–112, 2001.

[8] X. Chen, J.K. Udupa, X. Zheng, D. Torigian, and A. Alavi. Automatic anatomy recognition via multi object oriented active shape models. In *SPIE on Medical Imaging*, volume 7259, Orlando, Florida, USA, Feb 2009.

[9] K.C. Ciesielski, J.K. Udupa, A.X. Falcão, and P.A.V. Miranda. Fuzzy connectedness and graph cut image segmentation similarities and differences. In *Proceedings of SPIE on Medical Imaging: Image Processing*, 2011. to appear.

[10] K.C. Ciesielski, J.K. Udupa, P.K. Saha, and Y. Zhuge. Iterative relative fuzzy connectedness for multiple objects with multiple seeds. *Computer Vision and Image Understanding*, 107(3):160–182, 2007. doi: 10.1016/j.cviu.2006.10.005.

[11] T. Cootes, C. Taylor, D. Cooper, and J. Graham. Active shape models – their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, 1995.

[12] J. Cousty, G. Bertrand, L. Najman, and M. Couprie. Watershed cuts: Thinnings, shortest path forests, and topological watersheds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32:925–939, 2010.

[13] A.C.M. Dumay, M.N.A.J. Claessens, C. Roos, J.J. Gerbrands, and J.H.C. Reiber. Object delineation in noisy images by a modified policy-iteration method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(9):952–958, 1992.

[14] A.X. Falcão and F.P.G. Bergo. Interactive volume segmentation with differential image foresting transforms. *IEEE Trans. on Medical Imaging*, 23(9):1100–1108, 2004.

[15] A.X. Falcão, J. Stolfi, and R.A. Lotufo. The image foresting transform: Theory, algorithms, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(1):19–29, 2004.

[16] A.X. Falcão and J.K. Udupa. A 3D generalization of user-steered live wire segmentation. *Medical Image Analysis*, 4(4):389–402, Dec 2000. doi: 10.1016/S1361-8415(00)00023-2.

[17] A.X. Falcão, J.K. Udupa, and F.K. Miyazawa. An ultra-fast user-steered image segmentation paradigm: Live-wire-on-the-fly. *IEEE Transactions on Medical Imaging*, 19(1):55–62, Jan 2000.

[18] A.X. Falcão, J.K. Udupa, S. Samarasekera, S. Sharma, B.E. Hirsch, and R.A. Lotufo. User-steered image segmentation paradigms: Live-wire and live-lane. *Graphical Models and Image Processing*, 60(4):233–260, Jul 1998.

[19] M. Farber, J. Ehrhardt, and H. Handels. Live-wire-based segmentation using similarities between corresponding image structures. *Computerized Medical Imaging and Graphics*, 31(7):549–560, Oct 2007.

[20] L. Grady. Random walks for image segmentation. *IEEE Transactions Pattern Anaysis and Machine Intelligence*, 28(11):1768–1783, 2006.

[21] H.G. He, J. Tian, Y. Lin, and K. Lu. A new interactive segmentation scheme based on fuzzy affinity and live-wire. In *Fuzzy Systems and Knowledge Discovery*, volume 3613, pages 436–443, 2005.

[22] H.W. Kang. G-wire: A livewire segmentation algorithm based on a generalized graph formulation. *Pattern Recognition Letters*, 26(13):2042–2051, Oct 2005.

[23] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1:321–331, 1987.

[24] J. Liu and J.K. Udupa. Oriented active shape models. *IEEE Transactions on Medical Imaging*, 28(4):571–584, 2009.

[25] F. Malmberg, E. Vidholm, and I. Nystrom. A 3D live-wire segmentation method for volume images using haptic interaction. In *Discrete Geometry for Computer Imagery*, volume 4245, pages 663–673, 2006.

[26] A. Martelli. Edge detection using heuristic search methods. *Computer Graphics and Image Processing*, 1:169–182, 1972.

[27] A. Martelli. An application of heuristic search methods to edge and contour detection. *Communications of the ACM*, 19(2):73–83, 1976.

[28] P.A.V. Miranda and A.X. Falcão. Links between image segmentation based on optimum-path forest and minimum cut in graph. *Journal of Mathematical Imaging and Vision*, 35(2):128–142, 2009. doi:10.1007/s10851-009-0159-9.

[29] P.A.V. Miranda, A.X. Falcão, and J.K. Udupa. Synergistic arc-weight estimation for interactive image segmentation using graphs. *Computer Vision and Image Understanding*, 114(1):85–99, Jan 2010. doi: 10.1016/j.cviu.2009.08.001.

[30] U. Montanari. On the optimal detection of curves in noisy pictures. *Comm. of the ACM*, 14(5):335–345, 1971.

[31] E.N. Mortensen and W.A. Barrett. Interactive segmentation with intelligent scissors. *Graphical Models and Image Processing*, 60:349–384, 1998.

[32] N.J. Nilsson. *Principles of artificial intelligence*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1980.

[33] D. Pope, D. Parker, D. Gustafson, and P. Clayton. Dynamic search algorithms in left ventricular border recognition and analysis of coronary arteries. In *IEEE Proceedings of Computers in Cardiology*, volume 9, pages 71–75, 1984.

[34] J.B.T.M. Roerdink and A. Meijster. The watershed transform: Definitions, algorithms and parallelization strategies. *Fundamenta Informaticae*, 41:187–228, 2000.

[35] C. Rother, V. Kolmogorov, and A. Blake. "grabcut": Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics*, 23(3):309–314, 2004.

[36] P.K. Saha and J.K. Udupa. Relative fuzzy connectedness among multiple objects: Theory, algorithms, and applications in image segmentation. *Computer Vision and Image Understanding*, 82(1):42–56, 2001.

[37] M. Sonka, V. Hlavac, and R. Boyle. *Image Processing, Analysis and Machine Vision*. Chapman & Hall Computing, New York, NY, USA, 1993.

[38] J.K. Udupa, P.K. Saha, and R.A. Lotufo. Relative fuzzy connectedness and object definition: Theory, algorithms, and applications in image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:1485–1500, 2002.

[39] L. Vincent and P. Soille. Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6), Jun 1991.

[40] W.Yang, J.Cai, J. Zheng, and J. Luo. User-friendly interactive image segmentation through unified combinatorial user inputs. *IEEE Transactions on Image Processing*, 19(9):2470–2479, Sep 2010.