Object definition by $\kappa$-connected components

A.X. Falcão  P.A.V. Miranda  A. Rocha
F.P.G. Bergo


September - 2005 - Setembro

The contents of this report are the sole responsibility of the authors.
O conteúdo do presente relatório é de única responsabilidade dos autores.
Object definition by $\kappa$-connected components

Alexandre X. Falcão$^1$  Paulo A.V. Miranda$^1$  Anderson Rocha$^1$  Felipe P.G. Bergo$^1$

$^1$Instituto de Computação, Universidade Estadual de Campinas
Av. Albert Einstein, 1251, CEP 13084-970, Campinas, SP, Brasil

Abstract

The notion of “strength of connectedness” between pixels has been successfully used in image segmentation. We present extensions to these works, which can considerably improve the efficiency of object definition tasks. A set of pixels is said a $\kappa$-connected component with respect to a seed pixel, when the strength of connectedness of any pixel in that set with respect to the seed is higher than or equal to a threshold. We discuss two approaches that define objects based on $\kappa$-connected components with respect to a given seed set: with and without competition among seeds. While the previous approaches either assume no competition with a single threshold for all seeds or eliminate the threshold for seed competition, we show that seeds with different thresholds can improve segmentation in both paradigms. We also propose automatic and user-friendly interactive methods to determining the thresholds. The proposed methods are presented in the framework of the image foresting transform, which naturally leads to efficient and correct graph algorithms. The improvements are demonstrated through several segmentation experiments involving medical images.

1 Introduction

Many image segmentation methods have been proposed, but their success usually depends on user intervention, being automatic segmentation only verified for some specific situations. In view of that, it is important to develop interactive methods, which minimize the user’s time and involvement in the segmentation process, such that their automation becomes feasible under certain conditions. For example, we are interested in reducing the user intervention to simple selection of a few pixels in the image.

Fuzzy connectedness/watersheds are image segmentation approaches based on seed pixels, still in progress, and successfully used in many applications [1–6]. In relative fuzzy connectedness [7–9], as well as in watershed-based segmentations [10, 11], the seeds are specified inside and outside the object, each seed defines an influence zone composed by pixels more strongly connected to that seed than to any other, and the object is defined by the union of the influence zones of its internal seeds. In absolute fuzzy connectedness [12], a seed is specified inside the object and the strength of connectedness of each pixel with respect to that seed is computed, such that the object is obtained by thresholding the resulting connectivity image. Clearly, these approaches represent two distinct region-based segmentation paradigms, with and without competition among seeds.
We present extensions to these works, using the framework of the image foresting transform (IFT) [13]—a general tool for the design, implementation, and evaluation of image processing operators based on connectivity. In the IFT, the image is interpreted as a graph, whose nodes are the image pixels and whose arcs are defined by an adjacency relation between pixels. The cost of a path in this graph is determined by an application-specific path-cost function, which usually depends on local image properties along the path—such as color, gradient, and pixel position. For suitable path-cost functions and a set of seed pixels, one can obtain an image partition as an optimum-path forest rooted at the seed set. That is, each seed is root of a minimum-cost path tree whose pixels are reached from that seed by a path of minimum cost, as compared to the cost of any other path starting in the seed set. The IFT essentially reduces image operators to a simple local processing of attributes of the forest [14–18].

The strength of connectedness of a pixel with respect to a seed is inversely related to the cost of the optimum path connecting the seed to that pixel in the graph. In absolute fuzzy connectedness, the object can be obtained by selecting pixels reached from an internal seed by an optimum path whose cost is less than or equal to a number $\kappa$. In this case, the object is said a single $\kappa$-connected component (a minimum-cost path tree). The object can also be defined as the union of all $\kappa$-connected components created from each seed separately, which requires one IFT for each seed. In relative fuzzy connectedness, seeds selected inside and outside the object compete among themselves, partitioning the image into an optimum-path forest, and the object is defined by the union of the optimum-path trees rooted at its internal seeds. The initial appeal for relative fuzzy connectedness was the possibility to detect multiple objects simultaneously, without depending on thresholds. However, the use of thresholding together with seed competition provides a hybrid approach which turns out to be more efficient than the previous ones in many situations. While the previous approaches either assume no competition with a single value of $\kappa$ for all seeds or eliminate $\kappa$ for seed competition, we show that seeds with different values of $\kappa$ can considerably improve segmentation in both paradigms. Of course, this comes with the problem of finding the values of $\kappa$ for each seed, but we provide automatic and user-friendly interactive ways to determining them.

Section 2 describes some definitions related to the IFT, making them more specific for region-based image segmentation. For sake of simplicity, we will describe the methods for gray-scale and two-dimensional images, but they are extensile to multi-parametric and multi-dimensional data sets. The proposed variants and their algorithms are presented in Sections 3 and 4. Section 5 demonstrates the improvements with respect to the previous approaches. Conclusion and future work are presented in Section 6.

2 Background

An image $\hat{I}$ is a pair $(D_I, I)$ consisting of a finite set $D_I$ of pixels (points in $\mathbb{Z}^2$) and a mapping $I$ that assigns to each pixel $p$ in $D_I$ a pixel value $I(p)$ in some arbitrary value space.

An adjacency relation $A$ is a binary relation between pixels $p$ and $q$ of $D_I$. We use
Figure 1: (a) The main elements of a spanning forest with two roots, where the thicker path indicates $P^*(q)$. (b) An image graph with 4-adjacency, where the integers are the image values $I(p)$ and the bigger dots indicate two seeds. One is inside the brighter rectangle and one is in the darker background outside it. Note that, the background also contains brighter parts. (c) An optimum-path forest for $f_{\text{max}}$, with $\delta(p,q) = |I(q) - I(p)|$. The integers are the cost values and the rectangle is obtained as an optimum-path tree rooted at the internal seed.

$q \in A(p)$ and $(p,q) \in A$ to indicate that $q$ is adjacent to $p$. Once the adjacency relation $A$ has been fixed, the image $\hat{I}$ can be interpreted as a directed graph $(D_I,A)$ whose nodes are the image pixels in $D_I$ and whose arcs are the pixel pairs $(p,q)$ in $A$. We are interested in irreflexive, symmetric, and translation-invariant relations. For example, one can take $A$ to consist of all pairs of pixels $(p,q) \in D_I \times D_I \setminus \{(0,0)\}$ such that $d(p,q) \leq \rho$, where $d(p,q)$ denotes the Euclidean distance and $\rho$ is a specified constant (i.e. 4-adjacency, when $\rho = 1$, and 8-adjacency, when $\rho = \sqrt{2}$).

A path is a sequence $\pi = \langle p_1, p_2, \ldots, p_n \rangle$ of pixels where $(p_i, p_{i+1}) \in A$, for $1 \leq i \leq n-1$. The path is trivial if $n = 1$. Let $\text{org}(\pi) = p_1$ and $\text{dst}(\pi) = p_n$ be the origin and destination of a path $\pi$. If $\pi$ and $\tau$ are paths such that $\text{dst}(\pi) = \text{org}(\tau) = p$, we denote by $\pi \cdot \tau$ the concatenation of the two paths, with the two joining instances of $p$ merged into one. In particular, $\pi \cdot \langle p, q \rangle$ is a path resulting from the concatenation of its longest prefix $\pi$ and the last arc $(p,q) \in A$.

A predecessor map is a function $P$ that assigns to each pixel $q \in D_I$ either some other pixel in $D_I$, or a distinctive marker nil not in $D_I$ — in which case $q$ is said to be a root of the map. A spanning 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. For any pixel $q \in D_I$, a spanning forest $P$ defines a path $P^*(q)$ recursively as $\langle q \rangle$, if $P(q) = \text{nil}$, or $P^*(p) \cdot \langle p, q \rangle$ if $P(q) = p \neq \text{nil}$ (see Figure 1a).

A pixel $q$ is connected to a pixel $p$ if there exists a path in the graph from $p$ to $q$. In this sense, every pixel is connected to itself by its trivial path. Since $A$ is symmetric, we can also say that $p$ is connected to $q$, or simply $p$ and $q$ are connected. Therefore, a connected component is a subset of $D_I$ wherein all pairs of pixels are connected.
A path-cost function $f$ assigns to each path $\pi$ a path cost $f(\pi)$, in some totally ordered set $V$ of cost values, whose maximum element is denoted by $+\infty$. A path $\pi$ is optimum if $f(\pi) \leq f(\tau)$ for any other path $\tau$ with $\text{dst}(\tau) = \text{dst}(\pi)$, irrespective of its starting point. The IFT establishes some conditions applied to optimum paths, which are satisfied by only smooth path-cost functions. That is, for any pixel $q \in D_I$, there must exist an optimum path $\pi$ ending at $q$ which either is trivial, or has the form $\tau \cdot \langle p, q \rangle$ where

(C1) $f(\tau) \leq f(\pi)$,
(C2) $\tau$ is optimum,
(C3) for any optimum path $\tau'$ ending at $p$, $f(\tau' \cdot \langle p, q \rangle) = f(\pi)$.

The IFT takes an image $\hat{I}$, a smooth path-cost function $f$ and an adjacency relation $A$; and returns an optimum-path forest—a spanning forest $P$ such that $P^*(q)$ is optimum for every pixel $q \in D_I$. In the forest, there are three important attributes for each pixel: its predecessor in the optimum path, the cost of that path, and the corresponding root (or some label associated with it). The IFT-based image operators result from simple local processing of one or more of these attributes.

For given seed set $S \subset D_I$, the concept of strength of connectedness [12, 19] of a pixel $q \in D_I$ with respect to a seed $s \in S$ can be interpreted as an image property inversely related to the cost of the optimum path from $s$ to $q$ according to the max-arc path-cost function $f_{\text{max}}$:

$$f_{\text{max}}(\langle q \rangle) = \begin{cases} 0, & \text{if } q \in S, \\ +\infty, & \text{otherwise.} \end{cases}$$
$$f_{\text{max}}(\pi \cdot \langle p, q \rangle) = \max\{f_{\text{max}}(\pi), \delta(p, q)\},$$

(1)

where $(p, q) \in A$, $\pi$ is any path ending at $p$ and starting in $S$, and $\delta(p, q)$ is a non-negative dissimilarity function between $p$ and $q$ which depends on image properties, such as brightness and gradient (see Figures 1b and 1c).

One may think of smoothness as a more general definition for strength of connectedness. In this work, we discuss only $f_{\text{max}}$ because the comparison with previous approaches and our practical experience in region-based segmentation, which shows that $f_{\text{max}}$ often leads to better results than other commonly known smooth cost functions.

3 Image segmentation by $\kappa$-connectivity

We assume given a seed set $S$ either interactively, by simple mouse clicks, or automatically, based on some a priori knowledge about the approximate location of the object. The adjacency relation $A$ is usually a simple 8-neighborhood, but sometimes it is important to allow farther pixels be adjacent. This may reduce the number of seeds required to detect nearby components of a same object, such as letters of a word in the image of a text. Some examples of $\delta$ functions for $f_{\text{max}}$ are given below:
Object definition by $\kappa$-connected components

\[
\delta_1(p, q) = K \left( 1 - \exp \left( -\frac{1}{2\sigma^2} (I(p) - I(q))^2 \right) \right)
\]  
\[
\delta_2(p, q) = G(q)
\]  
\[
\delta_3(p, q) = K \left( 1 - \exp \left( -\frac{1}{2\sigma^2} \left( \frac{I(p) + I(q)}{2} - I(s) \right)^2 \right) \right)
\]  
\[
\delta_4(p, q) = \min_{s \in S} \{\delta_3(p, q)\}
\]  
\[
\delta_5(p, q) = a\delta_1(p, q) + b\delta_3(p, q)
\]  
\[
\delta_6(p, q) = \begin{cases} 
\delta_3(p, q)(1 + \vec{g}(p, q) \cdot \vec{\eta}(p, q)), & \text{if } E_r(p, q) > D_r(p, q), \\
K, & \text{otherwise}
\end{cases}
\]

where $K$ is a positive integer (e.g., the maximum image intensity), $\sigma$ is an allowed intensity variation, $G(q)$ is a gradient magnitude computed at $q$, and $I(s)$ is the intensity of a seed $s \in S$, such that $s = \text{org}(P^*(p))$ in $\delta_3$ and $\delta_1$ considers all seeds in $S$. The parameters $a$ and $b$ are constants such that $a + b = 1$, and $\vec{g}(p, q)$ is a normalized gradient vector computed at arc $(p, q)$, $\vec{\eta}(p, q)$ is the unit vector of the arc $(p, q)$, $E_r(p, q)$ and $D_r(p, q)$ are the pixel intensities at a distance $r$ to the left and right sides of the arc $(p, q)$, respectively.

We are interested in using the above functions under two possible segmentation paradigms: with and without seed competition. Functions $\delta_1$ and $\delta_2$ assume low inhomogeneity within the object. They represent gradient magnitudes with different image resolutions and lead to smooth functions in both paradigms. In fact, $f_{\text{max}}$ is smooth whenever $\delta(p, q)$ is fixed for any $(p, q) \in A$. In the case of seed competition, the IFT with these functions becomes a watershed transform [14]. Function $\delta_3$ exploits the dissimilarity between object and pixel intensities, being the object represented by its seed pixels. Although $f_{\text{max}}$ is smooth for $\delta_3$ with no seed competition, it may not be smooth in the case of competition among seeds [13] (i.e., the IFT results a spanning forest, but it may be non-optimal). This problem was the main motivation for $\delta_4$ [7]. However, sometimes $\delta_3$ with seed competition provides better segmentation results than $\delta_4$ (see Section 5). Function $\delta_3$ may also limit the influence zones of the seeds, when the intensities inside the object vary linearly toward the background. Function $\delta_5$ reduces this problem, and in the case of seed competition, one can also replace $\delta_3$ by $\delta_4$ in Equation 6. Function $\delta_6$ exploits situations where the intensities of the background on one side of the object region are higher than the object intensities and the other way around is true for the other side of the object (Figure 2). Other interesting ideas of dissimilarity functions for $f_{\text{max}}$ are presented in [7, 8, 12, 20, 21].

The basic differences between the formulations proposed in [8] and [7, 9] are that (i) the former assumes $\delta(p, q) = \delta(q, p)$ for all $(p, q) \in A$, and requires smooth path-cost functions, and (ii) the later allows asymmetric dissimilarity relations (e.g., $\delta_2$), and non-smooth cost functions (e.g., $f_{\text{max}}$ with $\delta_3$ and seed competition). The strength of connectedness between image pixels in (i) is a symmetric relation, while it may be asymmetric in (ii). The main theoretical differences between our formulation and these ones are presented next.
Figure 2: An MR-T1 image of the brain with one seed $s$ inside the cortex. (a-b) The maximum influence zones of $s$ within the cortex for $f_{\text{max}}$ with $\delta_3$ and with $\delta_6$, respectively. The asymmetry of $\delta_6$ favors segmentation in anti-clockwise orientation, increasing the influence zone of $s$.

3.1 Object definition without seed competition

We say that a pixel $p$ is $\kappa$-connected to a seed $s \in S$, if there exists an optimum path $\pi$ from $s$ to $p$ such that $f(\pi) \leq \kappa$. This $\kappa$-connectivity relation will be asymmetric whenever the dissimilarity $\delta(p, q)$ is asymmetric.

An object is a maximal subset of $D_I$ wherein all pixels $p$ are at least $\kappa$-connected to one pixel $s \in S$. Similarly to the method presented in [12], the object is the union of all $\kappa$-connected components with respect to each seed $s \in S$, which must be computed separately. This makes $f_{\text{max}}$ smooth for all dissimilarity functions described in Equations 2-7.

The algorithm described in [12] assumes that the object can be defined by a single value of $\kappa$ for all seeds in $S$. Figure 3a illustrates an example where this assumption works. However, a simple change in the position of a seed can fail segmentation (Figure 3b), because the influence zone of each seed inside the object is actually limited by a distinct value of cost $\kappa$ (Figure 3c). Moreover, the choice of seeds with distinct values of $\kappa$ usually reduces the number of seeds required to complete segmentation. This situation is better understood when we relate the concepts of minimum-spanning tree [22] and minimum-cost path tree for $f_{\text{max}}$ and symmetric $\kappa$-connectivity relations.

A minimum-spanning tree is a spanning forest $P$ with a single arbitrary root, where the sum of the arc weights $\delta(p, q)$ for all pairs $(p, q) \in A$, such that $P(q) = p$, is minimum, as compared to any other minimum-spanning tree obtained from the original graph $(D_I, A)$ (Figures 4a and 4b). If we remove the orientation of the arcs in Figure 4b, every pair of pixels in $P$ is connected by a path which is also optimum according to $f_{\text{max}}$ (Figure 4c). That is, the minimum-spanning tree encodes all possible minimum-cost path trees for $f_{\text{max}}$.

A $\kappa$-connected object with respect to a seed $s$ can be obtained by taking the component
Object definition by $\kappa$-connected components

Figure 3: A CT image of a knee where the patella can be segmented with two seed pixels, $s_1$ and $s_2$, $f_{max}$ with $\delta_3$, and without seed competition. (a) The result with a single value of $\kappa$ for both seeds. (b) The segmentation with a single value of $\kappa$ fails when we change the position of $s_1$, because $s_1$ requires a higher value of $\kappa$ to get the brighter part of the bone and $B$ invades the background at this higher value of $\kappa$. (c) The result can be corrected with distinct values of $\kappa$ for each seed.

connected to $s$, after removing all arcs from $P$ whose $\delta(p,q) > \kappa$. Suppose, for example, that the object is the brighter rectangle in the center of Figure 4a. Figure 4c shows that only the left side of the rectangle is obtained with $s_1$ and $\kappa_1 = 3$. If $\kappa_1 = 4$, $s_1$ reaches the right side of the rectangle but invades the background. The rectangle can be obtained with three seeds and $\kappa = 2$. However, different values of $\kappa$ reduce the number of seeds to two, $s_1$ with $\kappa_1 = 3$ and $s_2$ with $\kappa_2 = 2$ (Figure 4d).

There may be many arcs connecting object and background in a minimum-spanning tree. The choice of a single value of $\kappa$ is equivalent to remove the arcs whose weight $\delta(p,q)$ is minimum among those connecting object and background. This usually divides the object into several $\kappa$-connected components (minimum-spanning trees) and the segmentation will require one seed for each component. When we allow different values of $\kappa$, the object components become larger, and consequently, the number of seeds is reduced.

3.2 Object definition with seed competition

In [7, 8], seeds are selected inside and outside the object, and the object is defined by the subset of pixels which are more strongly connected to its internal seeds than to any other. This is the same as removing the arcs of maximum weight from the paths that connect object and background in the minimum-spanning tree. For example, the rectangle in Figure 4c is obtained by changing the position of $s_1$ to any pixel in the background and selecting $s_2$ as shown in Figure 4d. The main motivation for this paradigm was to eliminate the choice of $\kappa$, favoring the simultaneous segmentation of multiple objects.

We define the object as the subset of pixels which are more strongly $\kappa$-connected to its
internal seeds than to any other. That is, the seeds will compete among themselves for pixels reached from more than one seed by paths whose costs are less than or equal to $\kappa$. In which case, the pixel is conquered by the seed whose path cost is minimum. Note that, even the internal seeds compete among themselves, and a distinct value of $\kappa$ may be required for each seed. When the seed competition fails, these thresholds should limit the influence zones of the seeds avoiding connection between object and background, and the pixels do not conquered by any seed should be considered as belonging to the background.

In general, the use of distinct values of $\kappa$ together with seed competition reduces the number of seeds required to complete segmentation. Figure 5a shows an example where many seeds have to be carefully selected in the background to detect the object. The segmentation fails when some of these seeds are removed (Figure 5b), but it works when we limit the extent of the internal seed to some value of $\kappa$ (Figure 5c).

The algorithms and the problem of determining these thresholds for the internal seeds are addressed next.

### 4 Algorithms

The IFT uses a variant of Dijkstra’s algorithm [23] to compute three attributes for each pixel $p \in D_I$ [13]: its predecessor $P(p)$ in the optimum path, the cost $C(p)$ of that path,
Object definition by \( \kappa \)-connected components

Figure 5: A CT image of the orbital region where the eye ball is obtained by seed competition. (a) One internal seed and many external seeds are required for segmentation, using \( f_{\text{max}} \) with \( \delta_4 \). (b) The segmentation fails when some of the external seeds are removed. (c) A value of \( \kappa \) is used to limit the influence zone of the internal seed in parts where the seed competition fails.

and the corresponding root \( R(p) \). In the algorithms presented in this section, we do not need to create the predecessor map \( P \) and the root map \( R \) is only used in the case of seed competition. The IFT propagates wavefronts \( W_{cst} \) of same cost \( cst \) around each seed, following the order of the costs \( cst = 0, 1, \ldots, K \). This process is exploited to compute the values \( \kappa_s \) of each seed \( s \in S \) automatically and interactively.

4.1 Automatic computation of \( \kappa_s \)

First consider the wavefronts around a seed \( s \) selected inside a given object. All pixels \( p \) in the wavefront \( W_{cst} \) around \( s \) have optimum cost \( C(p) = cst, 0 \leq cst \leq K \). If the object is a single \( \kappa \)-connected component with respect to \( s \), then there exists a threshold \( \kappa_s, 0 \leq \kappa_s \leq K \), such that the object can be defined by the union of all wavefronts \( W_{cst} \), for \( cst = 0, 1, \ldots, \kappa_s \). We can specify a fixed \( \kappa_s \) for this particular application, but this is susceptible to intensity variations. Another alternative is to search for matchings between the shape of the object and the shape of the wavefronts. One drawback is the speed of segmentation, but this may be justified in some applications. A more complex situation occurs when the object definition requires more than one seed pixel. Each seed defines its own maximal extent inside the object and we need to match the shape of the object with the shape of the union of their influence zones. The approach presented here is much simpler and yet effective. It stems from the following observation.

The effectiveness of segmentation using \( f_{\text{max}} \) depends on assigning lower values of \( \delta(p, q) \) to arcs inside (and outside) the objects and higher values to arcs across their boundaries. As consequence, the wavefronts usually present a considerable increase in number of pixels when they cross the object boundaries (Figures 6a and 6b). That is, many pixels in the background are reached by optimum paths whose cost is the value \( \delta(p, q) \) of some arc \((p, q)\) across the boundary.
Figure 6: A CT image of the orbital region with one seed inside the eye ball. (a) A wavefront of cost $\kappa$ which represents the maximum extent of this seed inside the eye ball. (b) The wavefront of cost $\kappa+1$ shows a considerable augment in size when it invades the background. (c) The pixel propagation order provides more continuous transitions of the wavefronts to select $\kappa$, interactively.

We substitute the choice of one value $\kappa_s$ for each seed $s \in S$ by a threshold $T$ (i.e. a percentage of the total number of pixels), which limits the maximum size of their wavefronts. The region growing process of a seed $s$ must stop when the size of its wavefront of cost $cst$ is greater than $T$, and the value of $\kappa_s$ is determined as $\max\{cst - 1, 0\}$. The algorithms are presented for single object detection without seed competition and multiple object definition with seed competition.

**Single object definition without seed competition:**

Input: Image $\hat{I} = (D_I, I)$, adjacency $A$, internal seeds $S$, path-cost function $f_{\text{max}}$, and the size threshold $T$.
Output: Binary image $\hat{L} = (D_I, L)$, where $L(p) = 1$, if $p$ belongs to the object, and $L(p) = 0$ otherwise.
Auxiliary: A priority queue $Q$, variables $\text{tmp}$, $\kappa$, $cst$ and $\text{size}$, and cost map $C$ defined in $D_I$.

1. For every pixel $p \in D_I$, set $L(p) \leftarrow 0$.
2. While $S \neq \emptyset$, do
3.  For every pixel $p \in D_I$, set $C(p) \leftarrow +\infty$.
4.  Remove a seed $s$ from $S$.
5.  Set $C(s) \leftarrow 0$, $\text{size} \leftarrow 0$, $cst \leftarrow 0$, $\kappa \leftarrow +\infty$, and insert $s$ in $Q$.
6.  While $Q \neq \emptyset$ and $\kappa = +\infty$, do
7.    Remove a pixel $p$ from $Q$ such that $C(p)$ is minimum.
8.    For every $q \in A(p)$, such that $C(q) > C(p)$, do
9.      Set $\text{tmp} \leftarrow \max\{C(p), \delta(p, q)\}$. 


Object definition by $\kappa$-connected components

10. If $\text{tmp} < C(q)$, then
11. \quad If $C(q) \neq +\infty$, then remove $q$ from $Q$.
12. \quad Set $C(q) \leftarrow \text{tmp}$ and insert $q$ in $Q$.
13. \quad If $C(p) \neq \text{cst}$, then set $\text{size} \leftarrow 1$ and $\text{cst} \leftarrow C(p)$.
14. \quad Else, set $\text{size} \leftarrow \text{size} + 1$.
15. \quad If $\text{size} > T$ then set $\kappa \leftarrow \max\{\text{cst} - 1, 0\}$.
16. For every pixel $p \in D_I$, do
17. \quad If $C(p) \leq \kappa$, then set $L(p) \leftarrow 1$.
18. Remove any remaining pixels from $Q$.

Multiple object definition with seed competition:

Input: Image $\hat{I} = (D_I, \hat{I})$, adjacency $A$, path-cost function $f_{\max}$, size threshold $T$, and a labeled image $\hat{L} = (D_I, L)$, where $L(p) = i$, $0 \leq i \leq k$, if $p$ is a seed pixel selected inside object $i > 0$ from $k$ objects, being $i = 0$ reserved for seeds in the background, and $L(p) = -1$ otherwise.

Output: A labeled image $\hat{L} = (D_I, L)$, where $L(p) = i$, $0 \leq i \leq k$.

Auxiliary: Priority queue $Q$, variable $\text{tmp}$, and $C$, $R$, $\kappa$, $\text{size}$, and $\text{cst}$ are maps defined in $D_I$ to store cost and root of each pixel and threshold, wavefront size, and wavefront cost of each seed, respectively.

1. For every pixel $p \in D_I$, do
2. \quad Set $R(p) \leftarrow p$, $\text{size}(p) \leftarrow 0$, $\text{cst}(p) \leftarrow 0$, and $\kappa(p) \leftarrow +\infty$.
3. \quad If $L(p) = -1$, then set $C(p) \leftarrow +\infty$ and $L(p) \leftarrow 0$.
4. \quad Else, set $C(p) \leftarrow 0$ and insert $p$ in $Q$.
5. While $Q \neq \emptyset$, do
6. \quad Remove a pixel $p$ from $Q$ such that $C(p)$ is minimum.
7. \quad If $\kappa(R(p)) = +\infty$ and $L(R(p)) \neq 0$, then
8. \quad \quad If $C(p) \neq \text{cst}(R(p))$, then set $\text{size}(R(p)) \leftarrow 1$ and $\text{cst}(R(p)) \leftarrow C(p)$.
9. \quad \quad Else, set $\text{size}(R(p)) \leftarrow \text{size}(R(p)) + 1$.
10. \quad \quad If $\text{size}(R(p)) > T$, then set $\kappa(R(p)) \leftarrow \max\{\text{cst}(R(p)) - 1, 0\}$.
11. \quad If $C(p) \leq \kappa(R(p))$, then
12. \quad \quad For every $q \in A(p)$, such that $C(q) > C(p)$, do
13. \quad \quad \quad Set $\text{tmp} \leftarrow \max\{C(p), \delta(p, q)\}$.
14. \quad \quad \quad If $\text{tmp} < C(q)$, then
15. \quad \quad \quad \quad If $C(q) \neq +\infty$, then remove $q$ from $Q$.
16. \quad \quad \quad \quad Set $C(q) \leftarrow \text{tmp}$, $R(q) \leftarrow R(p)$, and insert $q$ in $Q$.
17. For every pixel $p \in D_I$, do
18. \quad If $C(p) \leq \kappa(R(p))$, then set $L(p) \leftarrow L(R(p))$.

The priority queue $Q$ can be implemented as described in [24, 25], such that each instance of the IFT will run in time proportional to the number $|D_I|$ of pixels. Note that, the first algorithm above stops propagation when the value $\kappa_s$ of a seed $s$ is found. In the case of
seed competition, the root map is used to find in constant time the root of each pixel in $S$. The influence zone of a seed $s \in S$ is limited either when it meets the influence zone of another seed at the same minimum cost or when the value $\kappa_s$ of $s$ is found.

### 4.2 Interactive computation of $\kappa_s$

A first approach is to compute the IFT for every pixel $p \in D_I$, such that the cost $C(p)$ of the optimum path that reaches $p$ from $S$ is found. In the case of seed competition, the corresponding root $R(p) \in S$ is also propagated to each pixel $p \in D_I$. Then, the user moves the cursor of the mouse over the image, and for each position $q$ of the cursor, the program displays the influence zone of the corresponding root $s = R(q) \in S$ defined by pixels $p \in D_I$, such that $C(p) \leq C(q)$ and $R(p) = R(q)$. This interactive process can be repeated until the user selects a pixel $q$ to confirm the influence zone of $s$ (i.e. $\kappa_s = C(q)$). The user can repeat this interactive process for each seed $s \in S$, in both paradigms.

One drawback of the method above is the abrupt size variations of the wavefronts (Figures 6a and 6b) which makes the selection of pixel $q$ sometimes difficult. We circumvent the problem by exploiting the propagation order $O(p)$ (a number from 1 to $|D_I|$) of each pixel $p$ removed from $Q$ during execution of the IFT. Note that, a pixel $p$ propagates before a pixel $q$ (i.e. $O(p) < O(q)$) when it is reached by an optimum path from $S$, whose cost $C(p)$ is less than the cost $C(q)$ of the optimum path that reaches $q$. When $C(p) = C(q)$, we assume a first-in-first-out (FIFO) tie-breaking policy for $Q$. That is, among all pixels with the same minimum cost in $Q$, the one first reached by an optimum path from $S$ is removed for propagation. Therefore, we also compute the propagation order $O(p)$ of each pixel $p \in D_I$. When the user moves the cursor to a position $q$, the program displays the influence zone of the corresponding root $s = R(q) \in S$ defined by pixels $p \in D_I$, such that $O(p) \leq O(q)$ and $R(p) = R(q)$. The rest of the process is the same. Note that, although $\kappa_s = C(q)$, only the pixels $p$ in the wavefront $W_{C(q)}$ which have $O(p) \leq O(q)$ are selected as belonging to the influence zone of $s$. This provides smoother transitions between consecutive wavefronts (Figure 6c) as compared to the first idea. The algorithms are presented below.

**Single object definition without seed competition:**

Input: Image $\hat{I} = (D_I, I)$, adjacency $A$, internal seeds $S$, and path-cost function $f_{\mathit{max}}$.

Output: Binary image $\hat{L} = (D_I, L)$, where $L(p) = 1$, if $p$ belongs to the object, and $L(p) = 0$ otherwise.

Auxiliary: Priority queue $Q$, variables $\mathit{tmp}$, $\mathit{ord}$, and cost map $C$ and propagation order map $O$ defined in $D_I$.

1. For every pixel $p \in D_I$, set $L(p) \leftarrow 0$.
2. While $S \neq \emptyset$, do
3. For every pixel $p \in D_I$, set $C(p) \leftarrow +\infty$.
4. Remove a seed $s$ from $S$.
5. Set $C(s) \leftarrow 0$, $\mathit{ord} \leftarrow 0$, and insert $s$ in $Q$. 


Object definition by $\kappa$-connected components

1. Set $ord \leftarrow 0$.
2. For every pixel $p \in D_I$, do
3. Set $R(p) \leftarrow p$.
4. If $L(p) = -1$, then set $C(p) \leftarrow +\infty$ and $L(p) \leftarrow 0$.
5. Else, set $C(p) \leftarrow 0$ and insert $p$ in $Q$.
6. While $Q \neq \emptyset$, do
7. Remove a pixel $p$ from $Q$ such that $C(p)$ is minimum.
8. Set $O(p) \leftarrow ord + 1$ and $ord \leftarrow ord + 1$.
9. For every $q \in A(p)$, such that $C(q) > C(p)$, do
10. Set $tmp \leftarrow \max\{C(p), \delta(p,q)\}$.
11. If $tmp < C(q)$, then
12. If $C(q) \neq +\infty$, then remove $q$ from $Q$.
13. Set $C(q) \leftarrow tmp$ and insert $q$ in $Q$.
14. While the user is not satisfied.
15. The user can select a pixel $q$ on the image.
16. For every pixel $p \in D_I$, do
17. If $O(p) \leq O(q)$ and $R(p) = R(q)$, then set $L(p) \leftarrow L(R(p))$.

Multiple object definition with seed competition:

Input: Image $\hat{I} = (D_I, I)$, adjacency $A$, path-cost function $f_{\text{max}}$, and a labeled image $\hat{L} = (D_I, L)$, where $L(p) = i$, $0 \leq i \leq k$, if $p$ is a seed pixel selected inside object $i > 0$ among $k$ objects, being $i = 0$ reserved for seeds in the background, and $L(p) = -1$ otherwise.

Output: A labeled image $\hat{L} = (D_I, L)$, where $L(p) = i$, $0 \leq i \leq k$.

Auxiliary: Priority queue $Q$, variables $tmp$ and $ord$, and $C$, $R$, $O$ are maps defined in $D_I$ to store cost, root and propagation order of each pixel, respectively.

1. Set $ord \leftarrow 0$.
2. For every pixel $p \in D_I$, do
3. Set $R(p) \leftarrow p$.
4. If $L(p) = -1$, then set $C(p) \leftarrow +\infty$ and $L(p) \leftarrow 0$.
5. Else, set $C(p) \leftarrow 0$ and insert $p$ in $Q$.
6. While $Q \neq \emptyset$, do
7. Remove a pixel $p$ from $Q$ such that $C(p)$ is minimum.
8. Set $O(p) \leftarrow ord + 1$ and $ord \leftarrow ord + 1$.
9. For every $q \in A(p)$, such that $C(q) > C(p)$, do
10. Set $tmp \leftarrow \max\{C(p), \delta(p,q)\}$.
11. If $tmp < C(q)$, then
12. If $C(q) \neq +\infty$, then remove $q$ from $Q$.
13. Set $C(q) \leftarrow tmp$, $R(q) \leftarrow R(p)$, and insert $q$ in $Q$.
14. While the user is not satisfied.
15. The user can select a pixel $q$ on the image.
16. For every pixel $p \in D_I$, do
17. If $O(p) \leq O(q)$ and $R(p) = R(q)$, then set $L(p) \leftarrow L(R(p))$.

One advantage of the presented algorithms as compared to classical segmentation methods based on seed competition occurs when the object contains several background parts.
Table 1: Description, imaging modality and number of slices for each object used in the experiments.

<table>
<thead>
<tr>
<th>Object</th>
<th>Description</th>
<th>Imaging Modality</th>
<th>Number of Slices</th>
</tr>
</thead>
<tbody>
<tr>
<td>O1</td>
<td>left eye ball</td>
<td>CT-orbit</td>
<td>15</td>
</tr>
<tr>
<td>O2</td>
<td>left caudate nucleus</td>
<td>MR-brain</td>
<td>15</td>
</tr>
<tr>
<td>O3</td>
<td>lateral ventricles</td>
<td>MR-brain</td>
<td>15</td>
</tr>
<tr>
<td>O4</td>
<td>corpus callosum</td>
<td>MR-brain</td>
<td>10</td>
</tr>
<tr>
<td>O5</td>
<td>patella</td>
<td>CT-knee</td>
<td>15</td>
</tr>
<tr>
<td>O6</td>
<td>femur</td>
<td>CT-knee</td>
<td>15</td>
</tr>
<tr>
<td>O7</td>
<td>white matter</td>
<td>MR-brain</td>
<td>15</td>
</tr>
</tbody>
</table>

(holes) inside it. In this case, the use of $\kappa_s$ usually eliminates the need for at least one background seed at each hole. One the other hand, some small noisy parts of the object may not be conquered by the internal seeds due to the use of $\kappa_s$. The labeled image can be post-processed, such that holes with area below a threshold are closed [26, 27]. The area closing operator has shown to be a very effective complement for the presented algorithms. In many situations, the objects do not have holes and high area thresholds can be used to reduce the number of internal seeds. These algorithms are compared to the classical segmentation approach based on seed competition in the next section.

5 Evaluation

We have selected 100 images from Magnetic Resonance (MR) and Computerized Tomography (CT) data sets of 7 objects for evaluation (see Table 1 and Figure 7). Each object consists of some slices that represent different degrees of challenge for segmentation. The original images have been pre-processed to increase the similarities between pixels inside the objects and the contrast between object and background. Each of four users have performed segmentation over the 100 images using each of three methods:

M1. Object detection without seed competition and automatic/interactive computation of $\kappa_s$;

M2. Object detection with seed competition and automatic $\kappa_s$ computation;

M3. Object detection with seed competition and without $\kappa_s$ computation.

The experiments aimed to compare these methods with respect to the number of user interactions required to complete segmentation. Although interactive $\kappa_s$ detection may reduce the number of external seeds in M2, we decided to avoid it in order to evaluate the combination of seed competition and automatic $\kappa_s$ detection with respect to M3.

In order to show the robustness of these approaches, we have chosen the best dissimilarity function for each situation and fixed the parameters of segmentation. We used the 8-neighborhood as adjacency relation $A$. The size threshold $T$ was set to 1% of the image
Figure 7: (a)-(g) Results of slice segmentation of the objects from 1 to 7, respectively, overlaid with the pre-processed images.
Table 2: The dissimilarity functions used for each combination of object and method.

<table>
<thead>
<tr>
<th>Object</th>
<th>M1</th>
<th>M2</th>
<th>M3</th>
</tr>
</thead>
<tbody>
<tr>
<td>O1</td>
<td>$\delta_3$</td>
<td>$\delta_2$</td>
<td>$\delta_2$</td>
</tr>
<tr>
<td>O2</td>
<td>$\delta_3$</td>
<td>$\delta_4$</td>
<td>$\delta_2$</td>
</tr>
<tr>
<td>O3</td>
<td>$\delta_3$</td>
<td>$\delta_3$</td>
<td>$\delta_3$</td>
</tr>
<tr>
<td>O4</td>
<td>$\delta_3$</td>
<td>$\delta_4$</td>
<td>$\delta_2$</td>
</tr>
<tr>
<td>O5</td>
<td>$\delta_3$</td>
<td>$\delta_4$</td>
<td>$\delta_4$</td>
</tr>
<tr>
<td>O6</td>
<td>$\delta_3$</td>
<td>$\delta_3$</td>
<td>$\delta_2$</td>
</tr>
<tr>
<td>O7</td>
<td>$\delta_3$</td>
<td>$\delta_3$</td>
<td>$\delta_3$</td>
</tr>
</tbody>
</table>

In Medical Image Analysis, it is common to use as ground truth the results of manual segmentation performed by an user expert. This methodology is questionable, because the experts usually make mistakes when they delineate the same object twice. In most cases, the results look like the same but there are small differences along the object boundaries. The small differences, however, seem to be acceptable in many applications. This similarity measure was defined as follows.

Each object was represented by a set of $l$ binary slices $\hat{L}_i = (D_I, L_i)$, $i = 1, 2, \ldots, l$, where $L_i(p) = 1$ for object pixels and 0 otherwise. Let $\hat{L}_i$ and $\hat{L}'_i$ be the binary images resulting from the segmentation of a same object slice using different methods. The similarity between these results was measured by:

$$1.0 - \frac{\sum_{i=1}^{l} \sum_{\forall p \in D_I} L_i(p) \oplus L'_i(p)}{\sum_{i=1}^{l} \sum_{\forall p \in D_I} L_i(p) + \sum_{i=1}^{l} \sum_{\forall p \in D_I} L'_i(p)},$$

where $\oplus$ is the “exclusive or” operation (i.e. $L_i(p) \oplus L'_i(p) = 1$, if $L_i(p) \neq L'_i(p)$, and 0 otherwise). In the case of manual segmentation by an user expert, it has been shown that the similarity values are around 0.96 [28]. Since none of the users is an expert, we required from them acceptable results from the expert’s point of view with similarity values around 0.90 between distinct segmentations of a same object, using different methods.

Method M3 represents the classical approach based on relative fuzzy connectedness/watershed transform. The number of user interactions in M3 is the total number of seeds selected inside (NIS - Number of Internal Seeds) and outside (NES - Number of External Seeds) the object. Methods M1 and M2 are the proposed variants of absolute fuzzy connectedness and relative fuzzy connectedness, respectively. In M1, the amount of user interaction is represented by the total number of interactive $\kappa_s$ detections (IKD) and
Object definition by $\kappa$-connected components

<table>
<thead>
<tr>
<th></th>
<th>M1</th>
<th>PDK</th>
<th>M2</th>
<th>M3</th>
<th>M1,M2</th>
<th>M1,M3</th>
<th>M2,M3</th>
</tr>
</thead>
<tbody>
<tr>
<td>O1</td>
<td>35.0</td>
<td>83.4%</td>
<td>29.5</td>
<td>77.6</td>
<td>0.965</td>
<td>0.969</td>
<td>0.962</td>
</tr>
<tr>
<td>O2</td>
<td>38.5</td>
<td>92.4%</td>
<td>29.3</td>
<td>38.8</td>
<td>0.904</td>
<td>0.890</td>
<td>0.915</td>
</tr>
<tr>
<td>O3</td>
<td>31.8</td>
<td>0.0%</td>
<td>31.3</td>
<td>61.3</td>
<td>0.992</td>
<td>0.932</td>
<td>0.935</td>
</tr>
<tr>
<td>O4</td>
<td>29.7</td>
<td>57.4%</td>
<td>27.5</td>
<td>46.8</td>
<td>0.922</td>
<td>0.914</td>
<td>0.918</td>
</tr>
<tr>
<td>O5</td>
<td>15.0</td>
<td>—</td>
<td>15.0</td>
<td>61.0</td>
<td>0.973</td>
<td>0.954</td>
<td>0.946</td>
</tr>
<tr>
<td>O6</td>
<td>26.3</td>
<td>27.1%</td>
<td>26.3</td>
<td>37.8</td>
<td>0.992</td>
<td>0.982</td>
<td>0.981</td>
</tr>
<tr>
<td>O7</td>
<td>47.5</td>
<td>6.8%</td>
<td>46.3</td>
<td>284.8</td>
<td>0.973</td>
<td>0.931</td>
<td>0.930</td>
</tr>
</tbody>
</table>

Table 3: The percentages of different $\kappa$ values (PDK) in M1, the average numbers of user interactions for each object and method, and the average similarity values between different methods for a same object.

NIS. The automatic $\kappa$ detections (AKD) are chosen as much as possible in order to reduce the number of user interactions. In M2, the number of user interactions is computed as in M3, but the number of seeds is expected to be much less due to automatic $\kappa$ detection.

In Section 1, we showed that the use of a unique $\kappa$ for all seeds may increase the number of seeds (user interactions). Instead of quantifying the number of user interactions for a fixed value of $\kappa$, we decided to quantify the number of different $\kappa$ values found in M1 for cases of multiple seeds. The percentages of different $\kappa$ values (PDK) are presented in Table 3, together with the average number of interactions and similarity values among all users. Note that, O3 was detected with the same value of $\kappa$, but the other objects required from 6.8% to 92.4% of different $\kappa$ values. O5 did not count because it was segmented with only one seed per slice. On average, M3 and M1 required 2.8 and 1.1 more user interactions than M2, respectively. The advantages of M1 and M2 over M3 increase in more complex situations, such as in the detection of O7 where M3 required 6 times more user interactions than M1 and M2. Although the performances of M1 and M2 have been equivalent, M2 is preferable because it is more general than M1 and M3.

Table 4 shows in detail the average values of NIS, NES, IKD, and AKD for each object and method. Note that, AKD varied from 59% to 100% of NIS, being on average 90% of NIS in M1 and 88% of NIS in M2. This demonstrates the effectiveness of the proposed approach for automatic $\kappa$ detection and explains the reduction of user interactions in M1 and M2 with respect to M3. Note also that, the number of external seeds was considerably reduced in comparison to M3. This is an important result for future automation, because seed competition is sensitive to the location of the external seeds due to the heterogeneity of the background.

6 Conclusions

We have presented four IFT-based algorithms for object detection based on $\kappa$-connected components with and without seed competition. They differ from the previous approaches in the following aspects: computation of different values of $\kappa$ for each seed, effective au-
Table 4: Average numbers of internal seeds (NIS), interactive $\kappa_s$ detections (IKD), external seeds (NES), and automatic $\kappa_s$ detections (AKD).

<table>
<thead>
<tr>
<th></th>
<th>M1</th>
<th></th>
<th>M2</th>
<th></th>
<th>M3</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NIS</td>
<td>IKD</td>
<td>AKD</td>
<td>NIS</td>
<td>NES</td>
<td>AKD</td>
</tr>
<tr>
<td>O1</td>
<td>30.7</td>
<td>4.3</td>
<td>26.4</td>
<td>18.0</td>
<td>11.5</td>
<td>13.0</td>
</tr>
<tr>
<td>O2</td>
<td>30.0</td>
<td>8.5</td>
<td>21.5</td>
<td>25.3</td>
<td>4.0</td>
<td>24.3</td>
</tr>
<tr>
<td>O3</td>
<td>30.0</td>
<td>1.8</td>
<td>28.2</td>
<td>30.3</td>
<td>1.0</td>
<td>30.3</td>
</tr>
<tr>
<td>O4</td>
<td>24.5</td>
<td>5.2</td>
<td>19.3</td>
<td>22.3</td>
<td>5.2</td>
<td>19.8</td>
</tr>
<tr>
<td>O5</td>
<td>15.0</td>
<td>0.0</td>
<td>15.0</td>
<td>15.0</td>
<td>0.0</td>
<td>15.0</td>
</tr>
<tr>
<td>O6</td>
<td>26.3</td>
<td>0.0</td>
<td>26.3</td>
<td>26.3</td>
<td>0.0</td>
<td>15.0</td>
</tr>
<tr>
<td>O7</td>
<td>47.5</td>
<td>0.0</td>
<td>47.5</td>
<td>46.0</td>
<td>0.3</td>
<td>46.0</td>
</tr>
</tbody>
</table>

Automatic $\kappa_s$ detection, and user-friendly $\kappa_s$ computation, where the user moves the cursor of the mouse to indicate the pixel whose propagation order defines the object. The use of propagation order rather than the pixel cost is important to create smoother transitions between possible objects, facilitating the user's work. The new methods have considerably reduced the number of user interactions in medical image segmentation with respect to the previous approaches. We believe that these results are extensive to other image types by suitable choice of pre-processing and dissimilarity function.

The interactive $\kappa_s$ detection counted with real time response for every position of the cursor, but this may not be feasible in 3D segmentation involving several slices. In this sense, the algorithms based on interactive $\kappa_s$ detection are more adequate for 2D/3D segmentation in a slice by slice fashion, where seeds may be automatically propagated along the slices. In such a case, the interactive $\kappa_s$ detection can be used to correct segmentation when the automatic detection of $\kappa_s$ fails.

We are currently investigating two approaches for 3D segmentation of medical images: (i) automatic segmentation with only internal seeds and automatic $\kappa_s$ detection, and (ii) interactive segmentation with automatic $\kappa_s$ detection, where the user can add/remove internal and external seeds, and subsequent IFTs are executed in a differential way [17].

Acknowledgments

The authors thank CNPq (Proc. 302427/04-0), FAPESP (Proc. 03/09793-1), and CAPES for the financial support.

References

Object definition by $\kappa$-connected components


