# Sub-pixel Segmentation with the Image Foresting Transform

Filip Malmberg, Joakim Lindblad, and Ingela Nyström

Centre for Image Analysis
Uppsala University and Swedish University of Agricultural Sciences
Sweden
http://www.cb.uu.se

**Abstract.** The Image Foresting Transform (IFT) is a framework for image partitioning, commonly used for interactive segmentation. Given an image where a subset of the image elements (seed-points) have been assigned user-defined labels, the IFT completes the labeling by computing minimal cost paths from all image elements to the seed-points. Each image element is then given the same label as the closest seed-point. In its original form, the IFT produces crisp segmentations, i.e., each image element is assigned the label of exactly one seed-point. Here, we propose a modified version of the IFT that computes region boundaries with sub-pixel precision by allowing mixed labels at region boundaries. We demonstrate that the proposed sub-pixel IFT allows properties of the segmented object to be measured with higher precision.

**Key words:** Image foresting transform, Interactive image segmentation, Sub-pixel precision.

## 1 Introduction

Image segmentation, i.e., the partitioning of an image into relevant regions, is a fundamental problem in image analysis. Accurate segmentation of objects of interest is often required before further analysis can be performed. Despite years of active research, fully automatic segmentation of arbitrary images is still seen as an unsolved problem. Semi-automatic, interactive segmentation methods [12] use human expert knowledge as additional input, thereby making the segmentation problem more tractable.

The segmentation process can be divided into two tasks: *recognition* and *delineation* [6]. Recognition is the task of roughly determining where in the image an object is located, while delineation consists of determining the exact extent of the object. Human users outperform computers in most recognition tasks, while computers are often better at delineation. A successful semi-automatic method combines these abilities to minimize user interaction time, while maintaining tight user control to guarantee the correctness of the result.

One popular paradigm for interactive segmentation is *seeded region segmentation*, where the user assigns labels (e.g., object and background) to a small

subset of the image elements (known as *seed-points*). An automatic algorithm then completes the labeling for all image elements. If the result is not satisfactory, the user can add or remove seed-points until a desired segmentation has been obtained. Many different algorithms have been proposed for performing the label completion, see, e.g, [1, 7]. Here, we will focus on one such algorithm, the Image Foresting Transform (IFT) [4, 5].

The IFT belongs to a family of graph-based methods, where the image is interpreted as a graph. Each image element corresponds to a node in the graph, and adjacent image elements are connected by edges. For each node in the graph, the minimum cost path from the node to the set of seed-points is computed. The cost of a path typically depends on local image features. By choosing an appropriate path cost function, popular image segmentation methods such as relative fuzzy-connectedness [3] and watersheds [11] can be implemented.

The IFT can be computed efficiently using Dijkstra's algorithm, slightly modified to allow multiple seed-points [5]. In interactive segmentation applications, a user often adds or removes seed-points to refine an existing segmentation. In [4], it was shown that seed-points can be added to, or removed from, an existing IFT solution, without recomputing the entire solution. This modified algorithm is called the *differential* IFT, and has been shown to give a significant reduction of the total time required for interactive segmentation.

In the original IFT, the resulting labels are *crisp*, i.e., each image element is assigned the label of exactly one seed-point. However, due to the finite resolution of digital images, an image element may be partially covered by more than one (continuous) object. By allowing mixed labels, it is possible to obtain segmentations with sub-pixel precision. Numerous studies have confirmed that *pixel coverage segmentation* [14] outperforms crisp segmentation for subsequent measuring of object properties such as length and area/volume, see, e.g., [13, 15]. In [9], it is shown that consequently misplacing the tissue borders, in a brain volume having voxels of size 1 $mm^3$, by one voxel resulted in volume errors of approximately 30%, 40% and 60% for white matter, grey matter and cerebrospinal fluid, respectively. Segmentation methods with sub-pixel precision can also produce more visually pleasing results than their crisp counterparts. Surface extraction algorithms such as Marching Cubes [10] can utilize sub-pixel precision to produce visually smoother surfaces. In the context of image compositing, sub-pixel segmentation is necessary to avoid aliasing artifacts [2].

Here we propose a modified version of the IFT, that computes labels with sub-pixel precision. In the following, we will refer to the original IFT method as *crisp IFT*, and the proposed method as *sub-pixel IFT*. Like the crisp IFT, the proposed method is defined on general graphs. Therefore, the method can be applied to higher-dimensional data without modification. Our method does not rely on any assumptions about the *shape* of the image elements. This makes the method more general, but also means that the output of the method is not strictly a pixel coverage segmentation. Instead, we see the previously demonstrated advantages of pixel coverage segmentation as a motivation for including

sub-pixel information in the IFT. We demonstrate that similar improvements in feature estimation can be achieved with the proposed sub-pixel IFT.

## 2 Notation and definitions

### 2.1 Images and graphs

An *image* $\mathbf{I}$ is a pair $(\mathcal{I}, I)$ consisting of a set $\mathcal{I}$ of image elements and a mapping $I$ that assigns to each image element $p \in \mathcal{I}$ a value in some arbitrary set (e.g., $\mathcal{I} \subset \mathbb{Z}^n$ and $I : \mathcal{I} \to [0, 255]$).

Over the image elements, we define an *adjacency* function $\mathcal{N}$ that maps each image element $p \in \mathcal{I}$ to a set $\mathcal{N}(p) \subset \mathcal{I}$ of *adjacent* nodes. Once the adjacency function has been fixed, the image $\mathbf{I}$ can be interpreted as a directed graph, whose nodes are the image elements and whose edges are all ordered pairs of image elements $p, q \in \mathcal{I}$ such that $q \in \mathcal{N}(p)$.

For each ordered pair of adjacent nodes $p$ and $q$, we assign a real valued *edge weight* $w(p, q)$. This weight typically depends on local image features such as intensity or gradient magnitude. A thorough discussion on how the choice of adjacency function and edge weights affect the segmentation results can be found in [8].

### 2.2 Paths and path costs

A *path* $\pi = \langle p_1, p_2, \ldots, p_k \rangle$ of length $|\pi| = k$ is a sequence $p_1, p_2, \ldots, p_k$ such that $p_{i+1} \in \mathcal{N}(p_i)$. We denote the *origin* $p_1$ and the *destination* $p_k$ of $\pi$ by $org(\pi)$ and $dst(\pi)$, respectively. If $\pi$ and $\tau$ are paths such that $dst(\pi) = org(\tau)$, we denote by $\pi \cdot \tau$ the concatenation of the two paths. Given a set $\Pi$ of paths such that $dst(\pi) = p$ for all $\pi \in \Pi$, and a path $\tau$ such that $org(\tau) = p$, we define the set $\Pi \cdot \tau$ as

$$\{\pi \cdot \tau \mid \pi \in \Pi\} . \tag{1}$$

The *cost* of a path is denoted $f(\pi)$. This cost is typically a function of the edge weights along the path, e.g., the sum of all the edge weights along the path or the maximum edge weight along the path. We require $f(\pi)$ to be strictly increasing with respect to $|\pi|$, i.e., for two non-empty paths $\pi, \tau$ such that $dst(\pi) = org(\tau)$

$$f(\pi) < f(\pi \cdot \tau) \text{ if } |\pi| < |\pi \cdot \tau| . \tag{2}$$

This requirement is slightly more strict than the corresponding requirement in [5], where the path cost function was only required to be monotonically increasing. The stricter requirement is necessary to guarantee the existence of the sub-pixel region boundaries in Section 3.2. In practice, this additional restriction is not very limiting. For a given monotonically increasing path cost function $f$, a corresponding strictly increasing function $g$ can be defined as

$$g(\pi) = f(\pi) + \epsilon|\pi| , \tag{3}$$

where $\epsilon$ is a small positive number.

Given three paths $\pi$, $\tau$ and $\upsilon$ such that $dst(\pi) = dst(\tau) = org(\upsilon)$ and$f(\pi) = f(\tau)$, we also require of the path cost function that

$$f(\pi \cdot \upsilon) = f(\tau \cdot \upsilon) \ . \tag{4}$$

A path $\pi$ is a *minimum cost path* if $f(\pi) \leq f(\tau)$ for any other path $\tau$ with $org(\tau) = org(\pi)$ and $dst(\tau) = dst(\pi)$. Note that in general, the minimum cost path is not unique. The set of minimum cost paths between two nodes $p$ and $q$ is denoted $\pi_{min}(p, q)$. For a node $p$ and a set $A \subset \mathcal{I}$, the set of minimal cost paths between $p$ and $A$ is defined as

$$\pi_{min}(p, A) = \bigcup_q \pi_{min}(p, q) \ , \tag{5}$$

for all $q \in \underset{r \in A}{\operatorname{argmin}}(f(\pi_{min}(p, r)))$.

### 2.3 Spanning Forests

A *predecessor map* is a function $P$ that assigns to each image element $p \in \mathcal{I}$ a (possibly empty) subset of $\mathcal{N}(p)$. Note that in contrast to [5], we here allow $|P(p)|$ to be greater than one, i.e. a node can have more than on predecessor. For any node $p \in \mathcal{I}$, a predecessor map $P$ defines a set $P^*(p)$ of one or more paths recursively as

$$P^*(p) = \begin{cases} \{\langle p \rangle\} & \text{if } P(p) = \emptyset \\ \bigcup_{q \in P(p)} (P^*(q) \cdot \langle q, p \rangle) & \text{otherwise} \end{cases} \ . \tag{6}$$

A *spanning forest* is a predecessor map that contains no cycles, i.e., $|\pi|$ is finite for all paths $\pi \in P^*(p)$.

### 2.4 The Image Foresting Transform

The IFT takes an image $\mathbf{I}$, a path cost function $f$, an adjacency function $\mathcal{N}$ and a set of seed-points $S \in \mathcal{I}$, and returns a spanning forest $P$ such that $P^*(p) = \pi_{min}(p, S)$ for all nodes $p \in \mathcal{I}$.

During this process, a *cost map* $C$ and a label map $\mathcal{L}$ is built. The cost map contains the cost of the minimum cost path from each pixel to $S$, i.e., $C(p) = f(\pi_{min}(p, S))$. The label map assigns to each node a *label vector* $\mathcal{L}(p) = (l_1, l_2, ..., l_k)$ where $l_i \in [0, 1]$ (For the original, crisp IFT, $l_i \in \{0, 1\}$). Each element in the label vector indicates the *belongingness* of the node to a certain class (such as object or background). The labels $l_i$ sum up to 1, i.e.

$$\sum_{i=0}^{k} l_i = 1 \ . \tag{7}$$

For all nodes $p \in \mathcal{I}$, $\mathcal{L}(p)$ should represent the label of the seed-point *closest* to $p$. As discussed in previous sections, assigning a single element of the set $\mathcal{L}_S = \{L(q) \,|\, q \in S\}$ to $\mathcal{L}(p)$ is problematic, due to both the ambiguity of minimal cost paths in the graph and the limited resolution of the image. We therefore calculate $\mathcal{L}(p)$ as a weighted average of the label vectors in $\mathcal{L}_S$. The exact procedure for calculating this average is described in Section 3. To differentiate between crisp and sub-pixel labels, label maps that have been computed with sub-pixel precision are denoted $\mathcal{L}_{sub}$.

## 3 Method

In this section, we present the proposed sub-pixel IFT method. The method consists of three steps. First, the IFT is computed, using a new policy for handling cases where the minimum cost path is not unique. Pseudo-code for this modified IFT is given in Section 3.1. Secondly, region boundaries between the nodes are estimated with sub-pixel precision using a linear model. Finally, the sub-pixel label of each node is computed by integrating the labels over the graph edges connected to the node. In Section 3.2, we show that this integral can be evaluated analytically.

### 3.1 Handling ties

As observed in Section 2.2, the minimum cost path between a node and the set of seed-points is not necessarily unique. Therefore, a strategy is needed for assigning labels in cases where the minimal cost path is ambiguous. In previous literature on the IFT, such a strategy is usually referred to as a *tie-breaking policy*. In [5], two strategies were suggested: the *first-in-first-out* (FIFO) strategy and *last-in-first-out* (LIFO) strategy. With these strategies, each node is assigned the label corresponding to the minimum cost path found first and last, respectively. Both these strategies are somewhat ad-hoc and have the effect that the output of the algorithm depends on the order in which we process the seed-points. A better strategy is therefore desirable.

The problem with handling ties in the crisp IFT framework is that a single label must be determined for each node. For the sub-pixel IFT this requirement is lifted, and we therefore have more flexibility for handling ties. Here, we have used a *mean* tie-breaking scheme, where each node $p$ is assigned the mean of the labels of the predecessors $P(p)$ along the minimal cost paths from $p$ to $S$. Pseudo-code for computing the IFT with mean tie-breaking is given in Algorithm 3.1. Note that due to Equation (4), $f(P^*(p) \cdot \langle p, q \rangle)$ is well defined even if $|P^*(p)| > 1$.

### 3.2 Sub-pixel estimation of region boundaries

In the crisp IFT, labels are defined for all nodes. To obtain sub-pixel precision, we define the labels over the graph edges as a piecewise constant function. Given

**Algorithm 1** Computing the IFT with mean tie-breaking

**Input:** An image $\mathbf{I}$, a path-cost function $f$, an adjacency function $\mathcal{N}$, and a set of seed-points $S \subset \mathcal{I}$.
**Output:** A spanning forest $P$ such that $P(p) = \pi_{min}(p, S)$ for all $p \in S$, a cost map $C$ such that $C(p) = f(\pi_{min}(p, S))$ for all $p \in S$, and a label map $\mathcal{L}$.
**Auxiliary data structures:** A list $Q$ of active nodes.
**Initialization:** Set $C(p) \leftarrow 0$ for source nodes $p \in S$ and $C(p) \leftarrow \infty$ for remaining nodes. Assign appropriate labels $\mathcal{L}(p)$ to all source nodes. Insert all source nodes in $Q$. Set $P(p) = \emptyset$ for all nodes.

```
while Q is not empty
  Remove from Q a node p such that C(p) is minimal.
  forall nodes q ∈ N(p)
    if f(P*(p) · ⟨p, q⟩) < C(q)
      Set C(q) ← f(P*(p) · ⟨p, q⟩)
      Set L(q) ← L(p)
      Set P(q) ← p
      Insert q in Q
    elseif f(P*(p) · ⟨p, q⟩) = C(q)
      Set L(q) ← (|P|L(q)+L(p)) / (|P|+1)
      Set P(q) ← P(p) ∪ {p}
    endif
  endfor
endwhile
```

two adjacent nodes $p$ and $q$ with corresponding labels, the label along the edge $p, q$ is given by

$$\mathcal{L}_{p,q}(t) = \begin{cases} \mathcal{L}(p) \text{ if } t \leq {t'}_{p,q} \\ \mathcal{L}(q) \text{ if } t > {t'}_{p,q} \end{cases} , \tag{8}$$

where the parameter $t \in \{0, 1\}$ determines the position along the edge and ${t'}_{p,q} \in \{0, 1\}$ is the point along the edge where the label changes.

To approximate ${t'}_{p,q}$, we compare the cost of a minimum path to $p$ with the cost of a minimal path to $q$ appended by the path $\langle q, p \rangle$, and vice versa. We denote these four scalar values $a$, $b$, $c$ and $d$:

$$\begin{aligned} a &= f(\pi_{min}(p, S)) \\ b &= f(\pi_{min}(p, S) \cdot \langle p, q \rangle) \\ c &= f(\pi_{min}(q, S)) \\ d &= f(\pi_{min}(q, S) \cdot \langle q, p \rangle) \ . \end{aligned} \tag{9}$$

We assume that the path costs vary linearly between nodes, a natural assumption if we consider the edge weights to be constant along each edge. Thus we can solve a linear equation to find ${t'}_{p,q}$. See Figure 1. For the intersection point ${t'}_{p,q}$, we obtain the equation

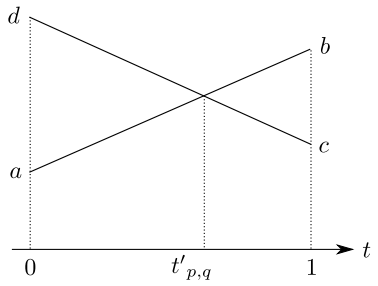


**Fig. 1.** Finding the intersection point $t'_{p,q}$ between two adjacent nodes. The scalars $a$,$b$,$c$ and $d$ are defined in the text.

$$t'_{p,q}a + (1 - t'_{p,q})b = (1 - t'_{p,q})c + t'_{p,q}d \; . \tag{10}$$

Solving Equation (10) for $t'_{p,q}$, we obtain

$$t'_{p,q} = \frac{b - c}{(b - a) + (d - c)} \; . \tag{11}$$

Since the path cost function $f$ is required to be strictly increasing with respect to path length, $(b-a) > 0$ and $(d-c) > 0$. Thus, the denominator of Equation (11) is non-zero.

Once the intersection points are determined, we calculate a sub-pixel label for each node by integrating the labels over edges connected to the node. For each edge, the domain of integration is the half of the edge that is closest to the node. This concept is illustrated in Figure 2. Formally, for each node $p \in G$, the sub-pixel label $\mathcal{L}_{sub}(p)$ is determined as

$$\mathcal{L}_{sub}(p) = \frac{2 \sum_{q \in \mathcal{N}(p)} \int_0^{\frac{1}{2}} \mathcal{L}_{p,q}(t) \, dt}{|\mathcal{N}(p)|} \; . \tag{12}$$

The integral in the numerator of Equation (12) can be written in closed form as

$$2 \int_0^{\frac{1}{2}} \mathcal{L}_{p,q}(t) \, dt = \begin{cases} s\mathcal{L}(p) + (1 - s)\mathcal{L}(q) & \text{if } s < 1 \\ \mathcal{L}(p) & \text{otherwise} \end{cases} \; , \tag{13}$$

where $s = 2t'_{p,q}$.

### 3.3 Implementation details

We have implemented the sub-pixel IFT in an interactive segmentation application. In our implementation, the sub-pixel boundaries are computed in a post-processing step. For efficiency, sub-pixel labels are only calculated for nodes that are adjacent to at least one node having different label. The time required for

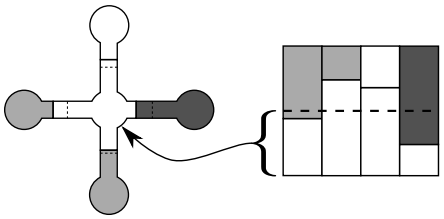

**Fig. 2.** Determining the sub-pixel label for a node with four neighbors. The sub-pixel label for the middle node is calculated by integrating over the closest half of all edges connected to the node.

computing the sub-pixel boundaries is small compared to the computation time of the IFT itself.

The proposed sub-pixel model can also be implemented differentially. For each node $p$ whose label is changed due to insertion or removal of a seed-point, only the sub-pixel labels of $p$ and $\mathcal{N}(p)$ need to be updated, since the sub-pixel label only depends on the labels at adjacent nodes.

## 4 Evaluation

All interactive segmentation methods are subject to variations in user input. A labeling method used for seeded region segmentation should therefore be insensitive to small variations in seed placement. To compare the crisp IFT and sub-pixel IFT with respect to this property, we perform an experiment where we segment the spleen in a slice from a CT volume of a human abdomen. The spleen is selected because it is a non-trivial object from a real application, yet it can be accurately segmented using a limited number of seed-points.

For this experiment, an additive path cost functions is used, i.e.,

$$f(\pi) = \sum_{i=2}^{|\pi|} w(\pi_i - 1, \pi_i) \; . \tag{14}$$

The edge cost is

$$w(p, q) = |I(q) - I(p)| + \epsilon \; , \tag{15}$$

where $\epsilon$ is a small positive number.

To simulate realistic variations in user input, we select regions that are determined to be inside and outside the spleen, respectively. We then compute both the crisp IFT and the sub-pixel IFT, using a single pixel from the inside region and the complete outside region as seed-points. See Figure 3. We thus obtain 41 crisp and 41 sub-pixel segmentations, one for each pixel in the inside region. Visually, all (crisp and sub-pixel) segmentations correctly delineate the spleen. Each segmentation was computed in less than a second on a standard

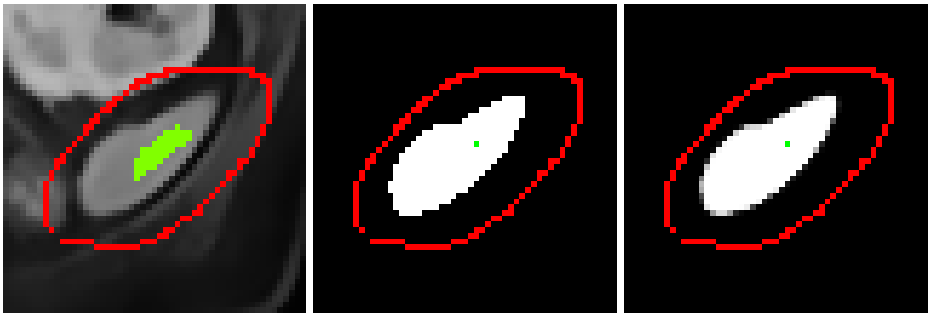

**Fig. 3.** Segmentation of the spleen in a slice from a CT volume.(Left) Seed-point regions used in the experiment. The green pixels define all object seeds, while the red pixels define background seeds. Single pixels from the green region were used to define object seeds. (Middle) Example result of crisp IFT. (Right) Example result of the proposed sub-pixel IFT.

**Table 1.** Statistics on the measured area for the 41 segmentations in the experiment. (Areas are given in number of pixels.)

| Method | Mean area | Min area | Max area | $\sigma$ |
|---|---|---|---|---|
| Crisp IFT | 266.5 | 265 | 269 | 0.98 |
| Sub-Pixel IFT | 266.2 | 265.4 | 266.7 | 0.40 |

PC (3.6 GHz, 3 GB RAM). The area of each segmented object is measured by summing all pixel values of the segmented image. The results are plotted in Table 1. The difference between the maximum and the minimum area as well as the standard deviation of the area measurements is smaller for the sub-pixel IFT than for the crisp IFT. The measured area for all individual segmentations are shown in Figure 4. For all segmentations, the area of the sub-pixel segmentation deviates less from the mean area than the area of the crisp segmentation. This is true regardless of whether the area is larger or smaller than the mean area.

The results of the experiment indicate that the sub-pixel IFT is less sensitive to variations in seed placement than the crisp IFT, for the purpose of estimating area/volume of a segmented object.

## 5 Conclusions

We have presented a modified version of the IFT algorithm, that computes labels with sub-pixel precision. The sub-pixel IFT is straightforward to implement in an existing IFT implementation, and preserves the advantages of the crisp IFT. It can be computed efficiently, and can be implemented differentially to allow fast editing. Like the crisp IFT, the sub-pixel IFT is defined for general graphs, and can therefore be applied to images of any dimension. In addition to 2D segmentation, an example of volume image segmentation with the sub-pixel IFT is shown in Figure 5.

Here, we have assumed that all edges that connect to a node affect the sub-pixel label of the node equally. For some graphs, this assumption may be invalid. In such cases, the uniform average in Equation (12) could be replaced by a weighted average that better reflects the influence of each edge.

## References


1. Boykov, Y., Funka-Lea, G.: Graph cuts and efficient N-D image segmentation. *International Journal of Computer Vision*, 70(2):109–131, 2006.
2. Braquelaire, J.-P., Vialard, A.: A new antialiasing approach for image compositing. *Visual Computer*, 13(5):218–227, 1997.
3. Ciesielski, K., Udupa, J., Saha, P., Zhuge, Y.: Iterative relative fuzzy connectedness for multiple objects with multiple seeds. *Computer Vision and Image Understanding*, 107(3):160–182, 2007.
4. Falcão, A. X., Bergo, F. P. G.: Interactive volume segmentation with differential image foresting transforms. *IEEE Transactions on Medical Imaging*, 23(9):1100–1108, 2004.
5. Falcão, A. X., Stolfi, J., Lotufo, R. A.: The image foresting transform: Theory, algorithms, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(1):19–29, 2004.
6. Falcão, A. X., Udupa, J. K., Samarasekera, S., Sharma, S., Hirsch, B. E., Lotufo, R. A.: User-steered image segmentation paradigms: Live wire and Live lane. *Graphical Models and Image Processing*, 60(4):233–260, 1998.
7. Grady, L.: Random walks for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(11):1768–1783, 2006.
8. Grady, L., Jolly, M.-P.: Weights and topology: A study of the effects of graph construction on 3D image segmentation. In D. Metaxas et al., editors, *MICCAI 2008*, volume 5241 of *LNCS*, pages 153–161, 2008.
9. Leemput, K. V., Maes, F., Vandermeulen, D., Suetens, P.: A unifying framework for partial volume segmentation of brain MR images. *IEEE Transactions on Medical Imaging*, 22(1):105–119, 2003.
10. Lorensen, W. E., Cline, H. E.: Marching cubes: A high resolution 3D surface construction algorithm. *Computer Graphics*, 21(4):163–169, 1987.
11. Lotufo, R. A., Falcão, A. X., Zampirolli, F.: IFT–Watershed from gray-scale marker. In *XV Brazilian Symposium on Computer Graphics and Image Processing*, pages 146–152. IEEE, 2002.
12. Olabarriaga, S. D., Smeulders, A. M.: Interaction in the segmentation of medical images: A survey. *Medical Image Analysis*, 5(2):127–142, 2001.
13. Sladoje, N., Lindblad, J.: High-precision boundary length estimation by utilizing gray-level information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):357–363, 2009.
14. Sladoje, N., Lindblad, J.: Pixel coverage segmentation for improved feature estimation. In P. Foggia et al., editors, *ICIAP 2009*, volume 5716 of *LNCS*, 2009. In press.
15. Sladoje, N., Nyström, I., Saha, P. K.: Measurements of digitized objects with fuzzy borders in 2D and 3D. *Image and Vision Computing*, 23(2):123–132, 2005.

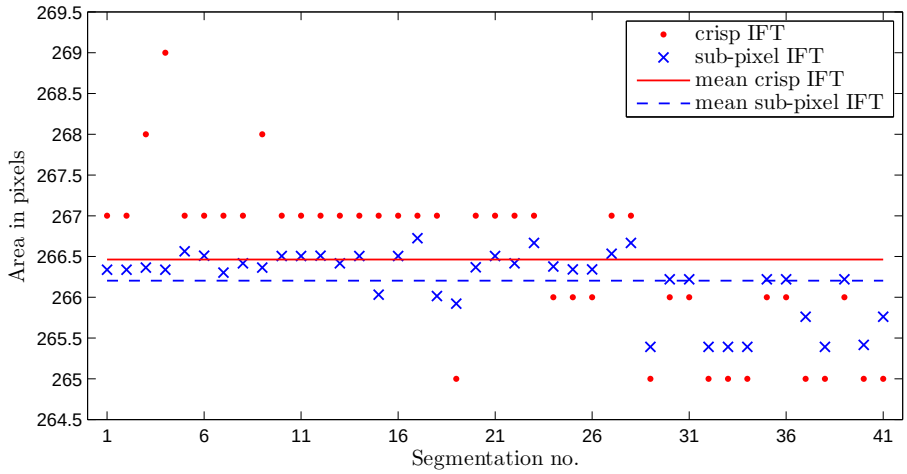


**Fig. 4.** Area of the segmented object for the 41 different segmentations in the experiment, sorted by the position of the seed-point inside the object. For all segmentations, the area of the sub-pixel segmentation deviates less from the mean area than the area of the crisp segmentation.

**Fig. 5.** Lateral ventricles of a human brain, segmented from an MR volume image using 20 single-voxel seed-points. A polygonal surface was extracted from the segmented volume using the Marching Cubes algorithm, which takes sub-pixel information into account. Both segmentations were produced using the same seed-points and path-cost function. (Left) Surface extracted from crisp IFT segmentation. (Right) Surface extracted from sub-pixel IFT segmentation.