Region-based Image Representation

Alexandre Xavier Falcão

Institute of Computing - UNICAMP

afalcao@ic.unicamp.br

Region-based Image Representation

- An image may be represented by connected regions, named superpixels, with a similar texture (color) pattern.

Region-based Image Representation

- An image may be represented by connected regions, named superpixels, with a similar texture (color) pattern.
- Superpixel segmentation should represent objects by the union of their superpixels.

Region-based Image Representation

- An image may be represented by connected regions, named superpixels, with a similar texture (color) pattern.
- Superpixel segmentation should represent objects by the union of their superpixels.
- The methods may be non-hierarchical and hierarchical, being the latter divided into sparse or dense hierarchies [1].

Region-based Image Representation

- An image may be represented by connected regions, named superpixels, with a similar texture (color) pattern.
- Superpixel segmentation should represent objects by the union of their superpixels.
- The methods may be non-hierarchical and hierarchical, being the latter divided into sparse or dense hierarchies [1].
- This lecture presents a recent non-hierarchical graph-based approach [2], named Dynamic Iterative Spanning Forest (DISF), and discusses its extension to hierarchical segmentation.

Agenda

- Seed-based superpixel segmentation: the traditional pipeline.
- The DISF pipeline and its motivation.
- The DISF algorithm.
- How to extend it to hierarchical segmentation.

Seed-based superpixel segmentation

1. Seed-based methods usually start from a desired number N_f of seed pixels,

1. Seed-based methods usually start from a desired number N_f of seed pixels,
2. apply a superpixel delineation algorithm such that each seed defines one region, and then

Seed-based superpixel segmentation

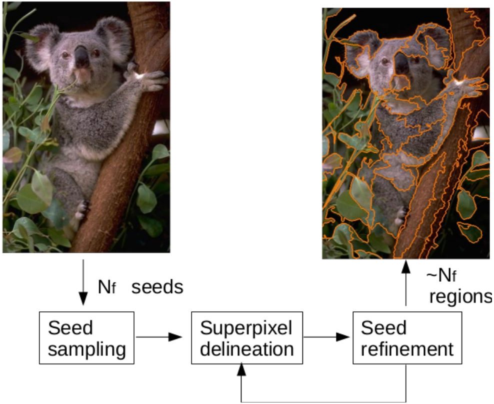
1. Seed-based methods usually start from a desired number N_f of seed pixels,
2. apply a superpixel delineation algorithm such that each seed defines one region, and then
3. refine the seed locations inside its region to improve superpixel delineation by a number of iterations of steps 2 and 3.

Seed-based superpixel segmentation

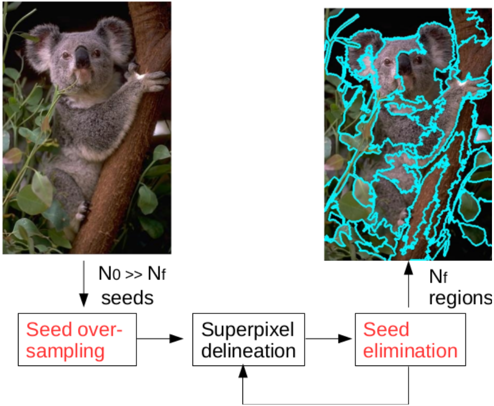
1. Seed-based methods usually start from a desired number N_f of seed pixels,
2. apply a superpixel delineation algorithm such that each seed defines one region, and then
3. refine the seed locations inside its region to improve superpixel delineation by a number of iterations of steps 2 and 3.

They usually do not guarantee the desired number of superpixels and the algorithm for superpixel delineation plays the main role.

Seed-based superpixel segmentation



The Iterative Spanning Forest (ISF) approach [3], for example, relies on the Image Foresting Transform (IFT) algorithm [4] for superpixel delineation.



DISF starts from a much higher number N_0 of seeds, also uses the IFT algorithm for superpixel delineation, and eliminates the number of seeds until the desired number N_f.

Motivation for DISF

- By starting from a seed oversampling, DISF considerably increases the chances to select relevant seeds for superpixel delineation.

Motivation for DISF

- By starting from a seed oversampling, DISF considerably increases the chances to select relevant seeds for superpixel delineation.
- It uses a connectivity function in the IFT algorithm that guarantees an optimum-path forest – each superpixel is an optimum-path tree rooted at its seed.

Motivation for DISF

- By starting from a seed oversampling, DISF considerably increases the chances to select relevant seeds for superpixel delineation.
- It uses a connectivity function in the IFT algorithm that guarantees an optimum-path forest – each superpixel is an optimum-path tree rooted at its seed.
- One can apply application-dependent criteria to retain relevant seeds at each iteration.

Motivation for DISF

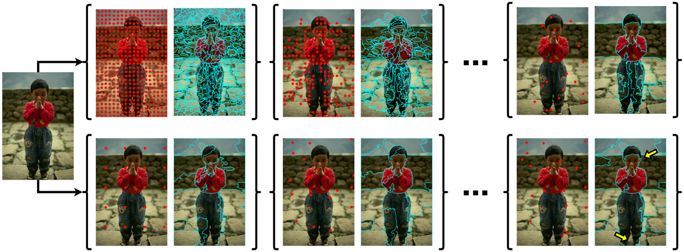
- By starting from a seed oversampling, DISF considerably increases the chances to select relevant seeds for superpixel delineation.

- It uses a connectivity function in the IFT algorithm that guarantees an optimum-path forest – each superpixel is an optimum-path tree rooted at its seed.

- One can apply application-dependent criteria to retain relevant seeds at each iteration.

- It improves superpixel delineation for lower numbers of superpixels.

Motivation for DISF



DISF (above) versus ISF (below) for lower number of superpixels. (Figure from [2].)

The DISF algorithm

- DISF uses grid sampling – a uniform seed distribution – to start the process.

- DISF uses grid sampling – a uniform seed distribution – to start the process.
- The IFT algorithm estimates arc-weights dynamically for the max-arc-weight function $f_{\max}$ based on image properties of the growing trees [5, 6] – this improves boundary adherence.

The DISF algorithm

- DISF uses grid sampling – a uniform seed distribution – to start the process.
- The IFT algorithm estimates arc-weights dynamically for the max-arc-weight function $f_{\max}$ based on image properties of the growing trees [5, 6] – this improves boundary adherence.
- Seed elimination is based on mid-level image properties of the resulting superpixel graph – it can better identify irrelevant superpixels for seed elimination and their relevant borders can be recovered in the next iteration.

The IFT algorithm for dynamic trees

Let $(D_I, \mathcal{A}, \mathbf{I})$ be an image graph and $\mathcal{S} = \mathcal{S}_0$ be the initial seed set with N_0 samples.

The IFT algorithm for dynamic trees

Let $(D_I, \mathcal{A}, \mathbf{I})$ be an image graph and $\mathcal{S} = \mathcal{S}_0$ be the initial seed set with N_0 samples.

DISF uses the version of $f_{\max}$ below as path-cost function:

$$
\begin{aligned}
f_{\max}(\langle q \rangle) &= \begin{cases} 0 & \text{if } q \in \mathcal{S}, \\ +\infty & \text{otherwise.} \end{cases} \\
f_{\max}(\pi_p \cdot \langle p, q \rangle) &= \max\{f_{\max}(\pi_p), \|\mu_{\tau_{R(p)}} - \mathbf{I}(q)\|_2\}, \\
\mu_{\tau_{R(p)}} &= \frac{1}{|\tau_{R(p)}|} \sum_{q \in \tau_{R(p)}} \mathbf{I}(q),
\end{aligned}
$$

where $\tau_{R(p)}$ is the growing tree that contains p and rooted $R(p) \in \mathcal{S}$.

The IFT algorithm for dynamic trees

Let $(D_I, \mathcal{A}, \mathbf{I})$ be an image graph and $\mathcal{S} = \mathcal{S}_0$ be the initial seed set with N_0 samples.

DISF uses the version of $f_{\max}$ below as path-cost function:

$$
\begin{aligned}
f_{\max}(\langle q \rangle) &= \begin{cases} 0 & \text{if } q \in \mathcal{S}, \\ +\infty & \text{otherwise.} \end{cases} \\
f_{\max}(\pi_p \cdot \langle p, q \rangle) &= \max\{f_{\max}(\pi_p), \|\mu_{\tau_{R(p)}} - \mathbf{I}(q)\|_2\}, \\
\mu_{\tau_{R(p)}} &= \frac{1}{|\tau_{R(p)}|} \sum_{q \in \tau_{R(p)}} \mathbf{I}(q),
\end{aligned}
$$

where $\tau_{R(p)}$ is the growing tree that contains p and rooted $R(p) \in \mathcal{S}$.

We call it segmentation by dynamic trees and other variants can be found in [5, 6].

The IFT algorithm for dynamic trees

For each $q \in D_I$, do

  Set $V(q) \leftarrow +\infty$, $R(q) \leftarrow q$, and $P(q) \leftarrow nil$.

  If $q \in \mathcal{S}$ then $V(q) \leftarrow 0$.

  Set $S_{\tau_q} \leftarrow 0$, $N_{\tau_q} \leftarrow 0$, and insert q in Q.

While $Q \neq \emptyset$ do

  Remove from Q the node $p = arg \min_{q \in Q}\{V(q)\}$.

  Set $S_{\tau_{R(p)}} \leftarrow S_{\tau_{R(p)}} + \frac{\mathbf{I}(p) - S_{\tau_{R(p)}}}{N_{\tau_{R(p)}} + 1}$ and $N_{\tau_{R(p)}} \leftarrow N_{\tau_{R(p)}} + 1$.

  Set $\mu_{\tau_{R(p)}} \leftarrow \frac{S_{\tau_{R(p)}}}{N_{\tau_{R(p)}}}$.

  For each $q \in \mathcal{A}(p)$, $q \in Q$, do

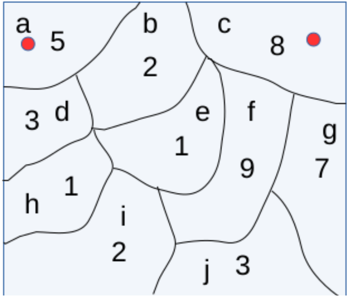
    If $V(q) > \max\{V(p), \|\mu_{\tau_{R(p)}} - \mathbf{I}(q)\|_2\}$, then

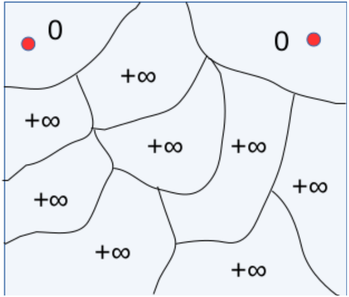
      Set $V(q) \leftarrow \max\{V(p), \|\mu_{\tau_{R(p)}} - \mathbf{I}(q)\|_2\}$,

      $R(q) \leftarrow R(p)$, and $P(q) \leftarrow p$.

Example

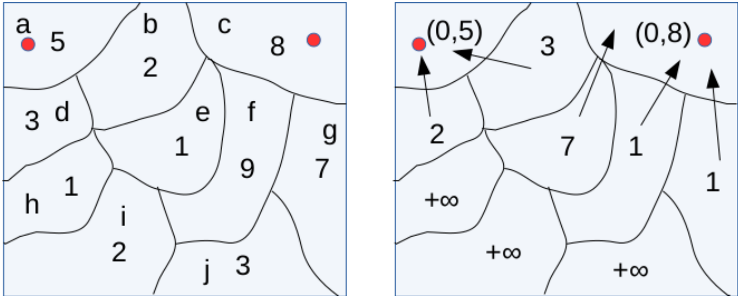
This example applies dynamic trees on an implicit region adjacency graph whose letters indicate nodes and numbers indicate node intensity on the left.





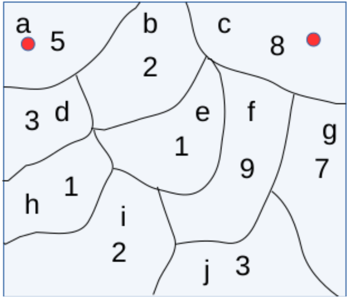
Trivial trees with initial costs on the right, forced to be zero on two root nodes, *a* and *c* (red).

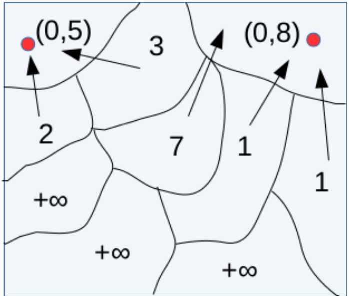
Example



- After two IFT iterations on the right, when *a* and *c* are removed from *Q*, and path costs (numbers) and predecessors (arrows) of its adjacent nodes change.

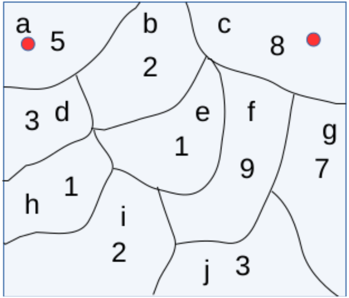
Example

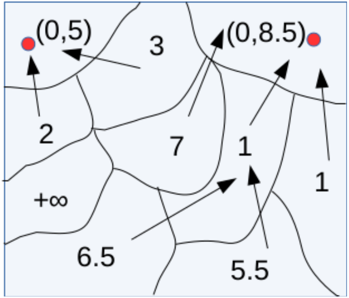




- After two IFT iterations on the right, when a and c are removed from Q, and path costs (numbers) and predecessors (arrows) of its adjacent nodes change.
- The notation (x, y) indicates cost $V(r) = x$ and mean $\mu_{\tau_r} = y$ for nodes in the growing tree τ_r rooted on node r.

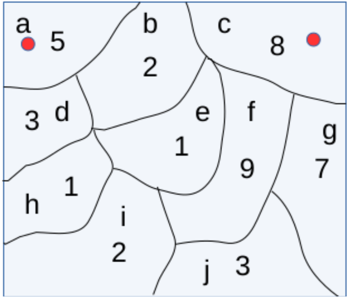
Example

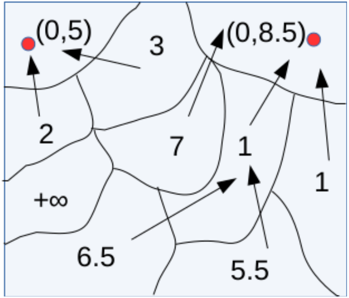




- When f is removed from Q (right, third IFT iteration), the mean $\mu_{\tau_{R(f)}}$ changes to $\frac{I(c)+I(f)}{2} = 8.5$.

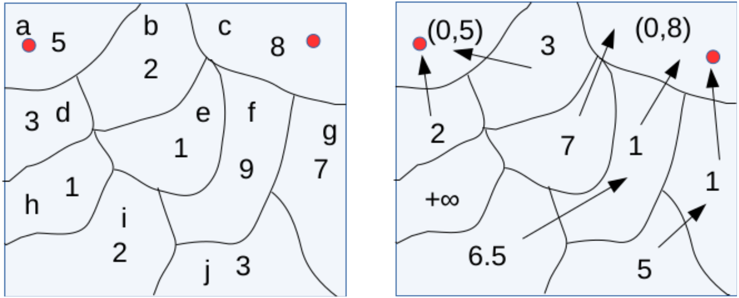
Example





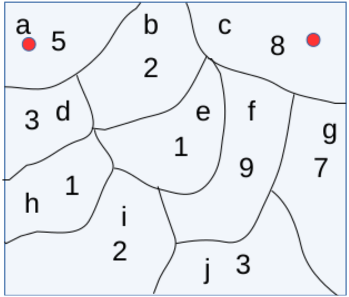
- When f is removed from Q (right, third IFT iteration), the mean $\mu_{\tau_{R(f)}}$ changes to $\frac{I(c)+I(f)}{2} = 8.5$.
- It then conquers nodes i and j by changing predecessors and costs to $P(i) = f$, $V(i) = 6.5$, $P(j) = f$, and $V(j) = 5.5$.

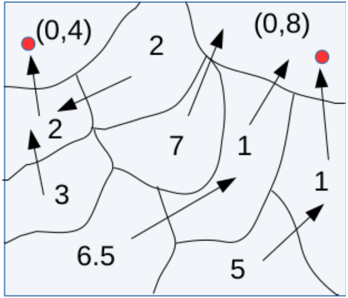
Example



When g is removed from Q (right, fourth IFT iteration), the mean $\mu_{\tau_{R(g)}}$ changes to $\frac{I(c)+I(f)+I(g)}{3} = 8$ and it conquers j with cost 5.

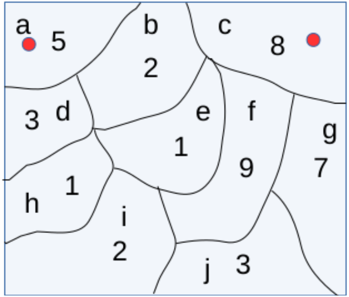
Example

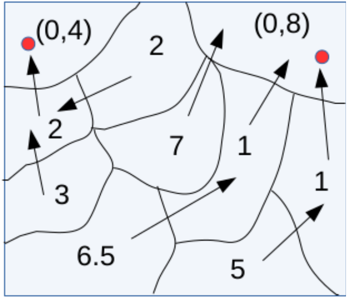




- When d is removed from Q (right, fifth IFT iteration), the mean $\mu_{\tau_{R(d)}}$ changes to $\frac{I(a)+I(d)}{2} = 4$.

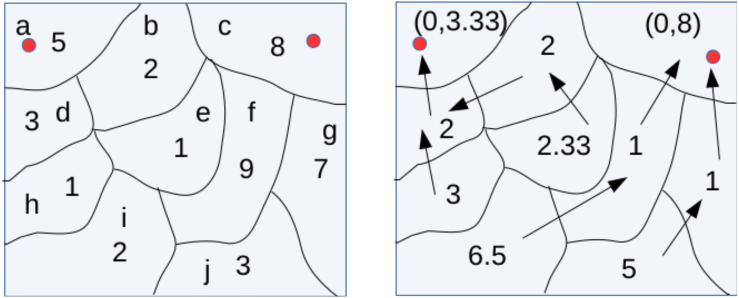
Example





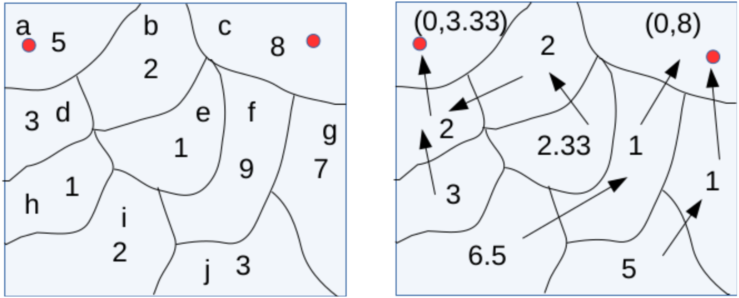
- When d is removed from Q (right, fifth IFT iteration), the mean $\mu_{\tau_{R(d)}}$ changes to $\frac{I(a)+I(d)}{2} = 4$.
- It conquers b and h by changing predecessors and costs to $P(b) = d$, $V(b) = 2$, $P(h) = d$, and $V(h) = 3$.

Example



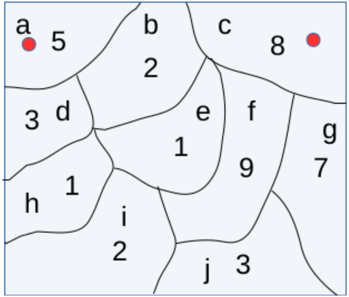
- When b is removed from Q (right, sixth IFT iteration), the mean $\mu_{\tau_{R(b)}}$ changes to $\frac{I(a)+I(d)+I(b)}{3} = 3.33$.

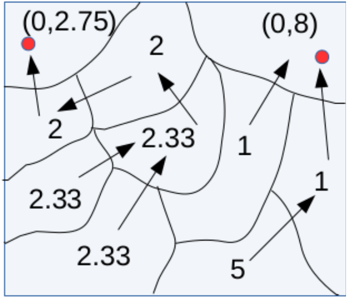
Example



- When b is removed from Q (right, sixth IFT iteration), the mean $\mu_{\tau_{R(b)}}$ changes to $\frac{I(a)+I(d)+I(b)}{3} = 3.33$.
- It conquers e by changing predecessor and cost to $P(e) = b$ and $V(e) = 2.33$.

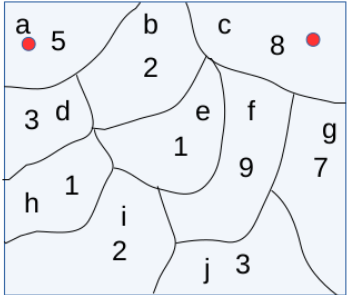
Example

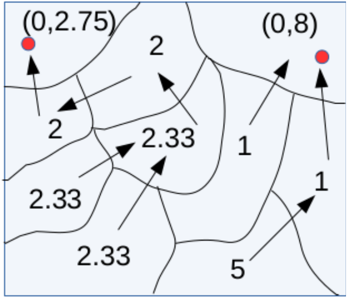




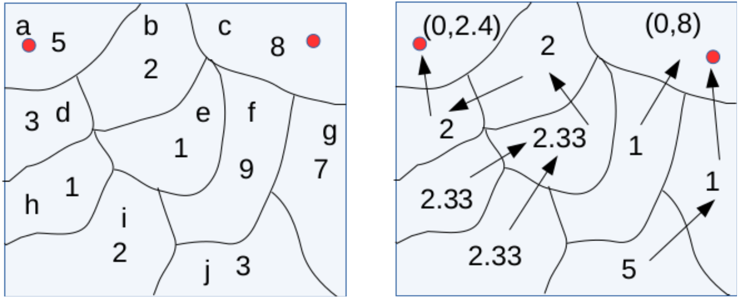
- When e is removed from Q (right, seventh IFT iteration), the mean $\mu_{\tau_{R(e)}}$ changes to $\frac{I(a)+I(d)+I(b)+I(e)}{4} = 2.75$.

Example



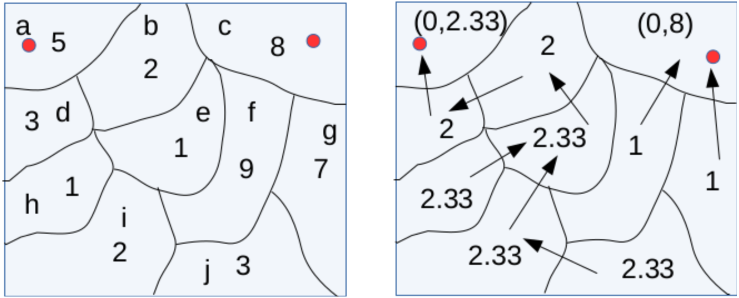


- When e is removed from Q (right, seventh IFT iteration), the mean $\mu_{\tau_{R(e)}}$ changes to $\frac{I(a)+I(d)+I(b)+I(e)}{4} = 2.75$.
- It conquers h and i by changing predecessors and costs to $P(h) = e$, $V(h) = 2.33$, $P(i) = e$, and $V(i) = 2.33$.



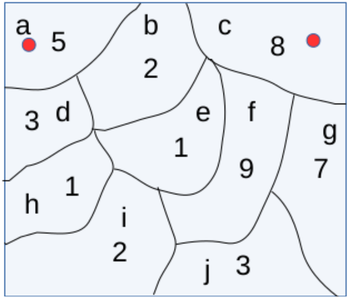
When h is removed from Q (right, eighth IFT iteration), it cannot conquer any node but the mean $\mu_{\tau_{R(h)}}$ changes to $\frac{I(a)+I(d)+I(b)+I(e)+I(h)}{5} = 2.4.$

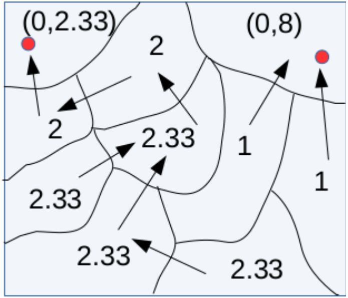
Example



- When i is removed from Q (right, ninth IFT iteration), the mean $\mu_{\tau_{R(i)}}$ changes to $\frac{I(a)+I(d)+I(b)+I(e)+I(h)+I(i)}{6} = 2.33$.

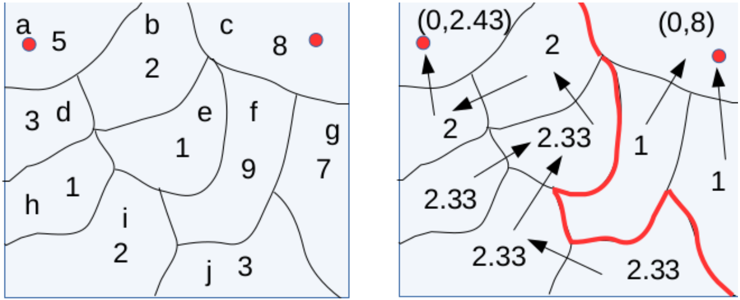
Example





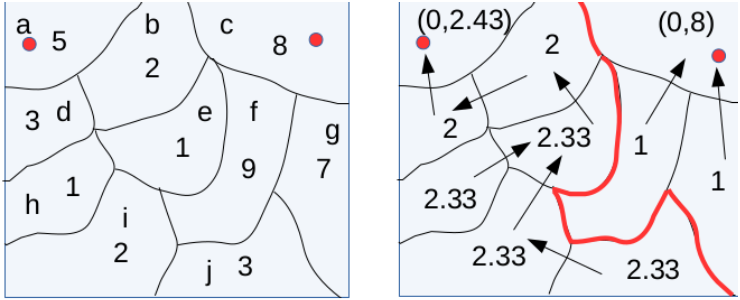
- When i is removed from Q (right, ninth IFT iteration), the mean $\mu_{\tau_{R(i)}}$ changes to $\frac{I(a)+I(d)+I(b)+I(e)+I(h)+I(i)}{6} = 2.33$.
- It conquers j by changing predecessor and cost to $P(j) = i$ and $V(j) = 2.33$.

Example



- When j is removed from Q (right, tenth IFT iteration), the mean $\mu_{\tau_{R(j)}}$ changes to $\frac{I(a)+I(d)+I(b)+I(e)+I(h)+I(i)+I(j)}{7} = 2.43$.

Example



- When j is removed from Q (right, tenth IFT iteration), the mean $\mu_{\tau_{R(j)}}$ changes to $\frac{I(a)+I(d)+I(b)+I(e)+I(h)+I(i)+I(j)}{7} = 2.43$.

- The process terminates with two optimum path trees.

The DISF algorithm

For a desired number N_f of superpixels.

Use grid sampling to get $\mathcal{S}$ with $|\mathcal{S}| = N_0 \gg N_f$ seeds.
Do
    Compute $(P, R, V) \leftarrow$ IFT-Algorithm$(D_I, \mathbf{I}, \mathcal{A}, \mathcal{S})$.
    Update $\mathcal{S}$ by eliminating seeds from irrelevant superpixels.
While $|\mathcal{S}| \neq N_f$.
Set $i \leftarrow 1$
For each $p \in D_I$ do
    If $R(p) = p$ then set $L(p) \leftarrow i$ and $i \leftarrow i + 1$.
For each $p \in D_I$ do
    Set $L(p) \leftarrow L(R(p))$.
Return segmentation in the label map L.

Seed elimination

The seed set $\mathcal{S}_j$ for a given iteration j of loop 2-5 is defined by the $M_j = \max\{N_0 \exp^{-j}, N_f\}$ seeds from the previous set $\mathcal{S}_{j-1}$ with the highest values $v(s)$, such that

The seed set $\mathcal{S}_j$ for a given iteration j of loop 2-5 is defined by the $M_j = \max\{N_0 \exp^{-j}, N_f\}$ seeds from the previous set $\mathcal{S}_{j-1}$ with the highest values $v(s)$, such that

$$v(s) = \frac{|\tau_s|}{|D_I|} \min_{(\tau_s,\tau_t)\in\mathcal{B}} \{\|\mu_{\tau_t} - \mu_{\tau_s}\|_2\}$$

is the relevance of a superpixel rooted at seed $s \in \mathcal{S}_{j-1}$,

$$\mathcal{B} = \{(\tau_s, \tau_t) \in \mathcal{T} \times \mathcal{T} \mid \exists (p,q) \in \mathcal{A}, p \in \tau_s, q \in \tau_t, s \neq t\}$$

is a tree-adjacency relation, and $\mathcal{T}$ is the set of optimum-path trees generated by the IFT algorithm.

How to extend DISF for hierarchical segmentation

- Except for the first iteration, the IFT algorithm must execute on a region adjacency graph (RAG) created from the result of its previous execution.

How to extend DISF for hierarchical segmentation

- Except for the first iteration, the IFT algorithm must execute on a region adjacency graph (RAG) created from the result of its previous execution.

- Each RAG can be created from the root map R, by inserting a step between Lines 3 and 4.

How to extend DISF for hierarchical segmentation

- Except for the first iteration, the IFT algorithm must execute on a region adjacency graph (RAG) created from the result of its previous execution.

- Each RAG can be created from the root map R, by inserting a step between Lines 3 and 4.

- By that, the subsequent RAGs compose a hierarchical segmentation – a tree of connected regions where each node contains the regions of the previous segmentation.

How to extend DISF for hierarchical segmentation

- Except for the first iteration, the IFT algorithm must execute on a region adjacency graph (RAG) created from the result of its previous execution.
- Each RAG can be created from the root map R, by inserting a step between Lines 3 and 4.
- By that, the subsequent RAGs compose a hierarchical segmentation – a tree of connected regions where each node contains the regions of the previous segmentation.
- How does it compare with the original DISF algorithm and the recursive ISF algorithm in [1]?

[1] F.L. Galvão, S.J.F. Guimarães, and A.X. Falcão.
Image segmentation using dense and sparse hierarchies of superpixels.
Pattern Recognition, 108:107532, 2020.

[2] F. Belem, S. Guimaraes, and A. X. Falcão.
Superpixel segmentation using dynamic and iterative spanning forest.
IEEE Signal Processing Letters, 2020.
arXiv:2007.04257.

[3] J.E. Vargas-Muñoz, A.S. Chowdhury, E.B. Alexandre, F.L. Galvão, P.A.V. Miranda, and A.X. Falcão.
An iterative spanning forest framework for superpixel segmentation.
IEEE Transactions on Image Processing, 28(7):3477–3489, 2019.

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

[5] Jordão Bragantini, Samuel Botter Martins, Cesar Castelo-Fernandez, and Alexandre Xavier Falcão.

Graph-based image segmentation using dynamic trees.

In *Iberoamerican Congress on Pattern Recognition*, pages 470–478, 2018.

[6] Alexandre Xavier Falcão and Jordão Bragantini.

The role of optimum connectivity in image segmentation: Can the algorithm learn object information during the process?

In *Intl. Conf. on Discrete Geometry for Computer Imagery*, pages 180–194, 2019.