Transformada Imagem-Floresta

Alexandre Xavier Falcão

Instituto de Computação - UNICAMP

afalcao@ic.unicamp.br

Introduction

Image transformations can be based on

Introduction

Image transformations can be based on

- pixels (e.g., thresholding).

Introduction

Image transformations can be based on

- pixels (e.g., thresholding).
- adjacency relations: pixels and their neighbors (e.g., linear filtering).

Introduction

Image transformations can be based on

- pixels (e.g., thresholding).
- adjacency relations: pixels and their neighbors (e.g., linear filtering).
- connectivity relations: sequences of adjacent pixels (e.g., component labeling).

Motivation

- The interpretation of an image as a graph provides a more general topology to the design of image transformations.

Motivation

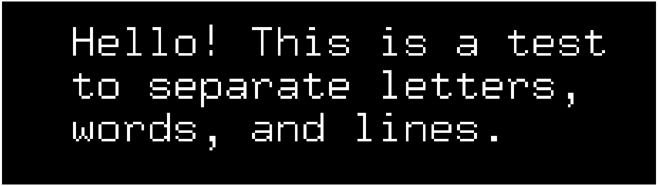
- The interpretation of an image as a graph provides a more general topology to the design of image transformations.
- The graph nodes may be pixels, edges, regions, and the arcs will result from a given adjacency relation.

Motivation

- The interpretation of an image as a graph provides a more general topology to the design of image transformations.
- The graph nodes may be pixels, edges, regions, and the arcs will result from a given adjacency relation.
- This strategy counts with several algorithms and their proof of correctness.

Motivation

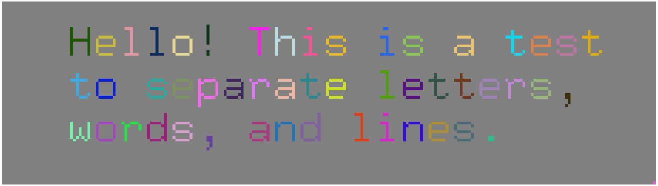
The same algorithm with distinct adjacency relations, for example, can label letters, words and lines.



Hello! This is a test
to separate letters,
words, and lines.

Motivation

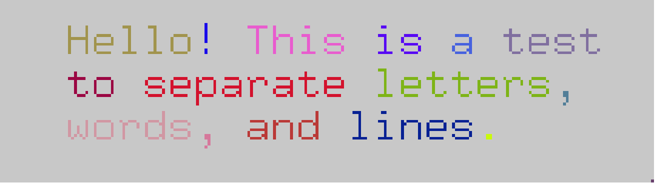
The same algorithm with distinct adjacency relations, for example, can label letters, words and lines.



Hello! This is a test
to separate letters,
words, and lines.

Motivation

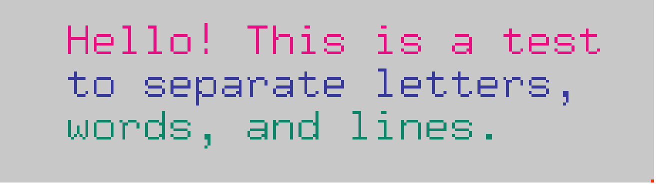
The same algorithm with distinct adjacency relations, for example, can label letters, words and lines.



Hello! This is a test
to separate letters,
words, and lines.

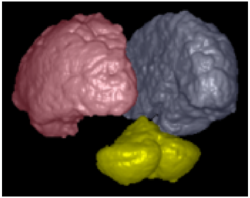
Motivation

The same algorithm with distinct adjacency relations, for example, can label letters, words and lines.



Hello! This is a test
to separate letters,
words, and lines.

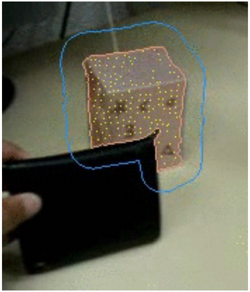
Objective



This course presents the Image Foresting Transform (IFT) [1] — a tool to the design of image operations based on optimum connectivity relations, and some of its applications. Examples are

- segmentation,

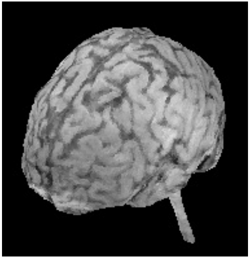
Objective



This course presents the Image Foresting Transform (IFT) [1] — a tool to the design of image operations based on optimum connectivity relations, and some of its applications. Examples are

- segmentation,
- object tracking,

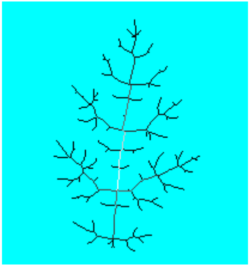
Objective



This course presents the Image Foresting Transform (IFT) [1] — a tool to the design of image operations based on optimum connectivity relations, and some of its applications. Examples are

- segmentation,
- object tracking,
- visualization

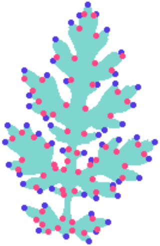
Objective



This course presents the Image Foresting Transform (IFT) [1] — a tool to the design of image operations based on optimum connectivity relations, and some of its applications. Examples are

- segmentation,
- object tracking,
- visualization
- multiscale skeletonization,

Objective



This course presents the Image Foresting Transform (IFT) [1] — a tool to the design of image operations based on optimum connectivity relations, and some of its applications. Examples are

- segmentation,
- object tracking,
- visualization
- multiscale skeletonization,
- salience detection,

filtering, clustering, classification, etc.

Advantages

The IFT provides

Advantages

The IFT provides

- efficient algorithms with linear execution times [2, 3, 4] in the worst case for most applications.

Advantages

The IFT provides

- efficient algorithms with linear execution times [2, 3, 4] in the worst case for most applications.
- a unified framework that favors a better understanding of the relation among methods [5, 6, 7] and hardware-based implementations [8], and

Advantages

The IFT provides

- efficient algorithms with linear execution times [2, 3, 4] in the worst case for most applications.
- a unified framework that favors a better understanding of the relation among methods [5, 6, 7] and hardware-based implementations [8], and
- effective solutions to image processing and analysis problems from the specification of a few parameters.

Organization of the course

- First: the IFT framework.

Organization of the course

- First: the IFT framework.
- Second: connected filters.

Organization of the course

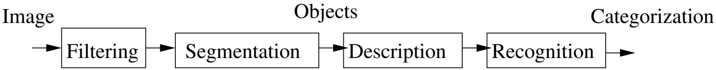
- First: the IFT framework.
- Second: connected filters.
- Third: interactive and automatic segmentation methods.

Organization of the course

- First: the IFT framework.
- Second: connected filters.
- Third: interactive and automatic segmentation methods.
- Fourth: shape representation and description.

Organization of the course

- First: the IFT framework.
- Second: connected filters.
- Third: interactive and automatic segmentation methods.
- Fourth: shape representation and description.
- Fifth: clustering and classification.



Organization of this lecture

- Basic definitions.

Organization of this lecture

- Basic definitions.
- Images as graphs.

Organization of this lecture

- Basic definitions.
- Images as graphs.
- Connectivity functions.

Organization of this lecture

- Basic definitions.
- Images as graphs.
- Connectivity functions.
- Image foresting transform.

Organization of this lecture

- Basic definitions.
- Images as graphs.
- Connectivity functions.
- Image foresting transform.
- General algorithm, some variants and implementation issues.

General image definition

A digital image $\hat{I} = (D_I, \vec{I})$ is a pair, where

General image definition

A digital image $\hat{I} = (D_I, \vec{I})$ is a pair, where

- $D_I \subset \mathcal{Z}^n$ is the image domain (a set of spels — space elements), and

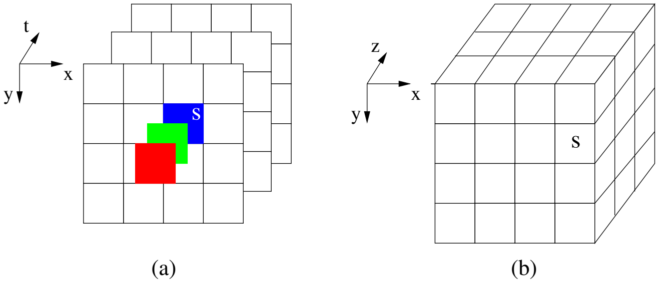
General image definition

A digital image $\hat{I} = (D_I, \vec{I})$ is a pair, where

- $D_I \subset \mathcal{Z}^n$ is the image domain (a set of spels — space elements), and
- $\vec{I}(s) = (I_1(s), I_2(s), \ldots, I_m(s)) \in \mathcal{Z}^m$ is a vectorial mapping, which assigns a set of values to each $s \in D_I$.

For $m = 1$, we use $\hat{I} = (D_I, I)$.

General image definition



(a) A RGB video $\hat{I}$, $n = 3$ and $m = 3$. (b) A CT image $\hat{I}$, $n = 3$ and $m = 1$.

Image domain and feature space

A spel $s \in D_I$ is a point $\vec{I}(s) \in \mathcal{Z}^m$ in the feature space.

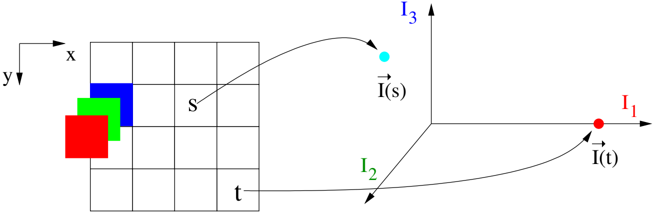
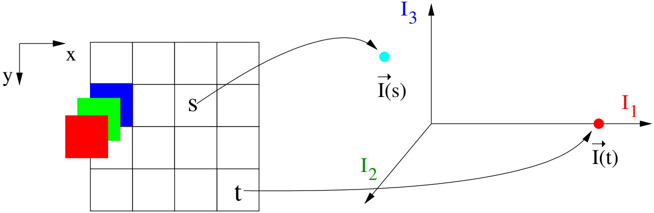


Image domain and feature space

A spel $s \in D_I$ is a point $\vec{I}(s) \in \mathcal{Z}^m$ in the feature space.



New features may result from any image transformation $\Psi(\hat{I})$, creating a real image $\hat{F} = (D_F, \vec{F})$ where $D_F = D_I$ and $\vec{F}(s) = (F_1, F_2, \ldots, F_{m'}) \in \Re^{m'}$.

Images as graphs

An image can be interpreted as a graph $(\mathcal{N}, \mathcal{A})$, whose

Images as graphs

An image can be interpreted as a graph $(\mathcal{N}, \mathcal{A})$, whose

- nodes form a subset $\mathcal{N} \subseteq D_I$ and

Images as graphs

An image can be interpreted as a graph $(\mathcal{N}, \mathcal{A})$, whose

- nodes form a subset $\mathcal{N} \subseteq D_I$ and
- arcs are defined by an adjacency relation $\mathcal{A} \subset D_I \times D_I$.

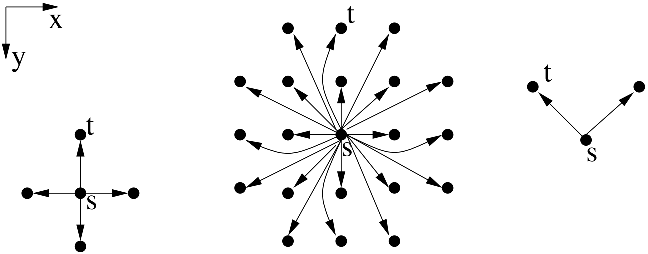
Images as graphs

An image can be interpreted as a graph $(\mathcal{N}, \mathcal{A})$, whose

- nodes form a subset $\mathcal{N} \subseteq D_I$ and
- arcs are defined by an adjacency relation $\mathcal{A} \subset D_I \times D_I$.

We will indicate that a spel t is adjacent to spel s either by $(s, t) \in \mathcal{A}$ or $t \in \mathcal{A}(s)$.

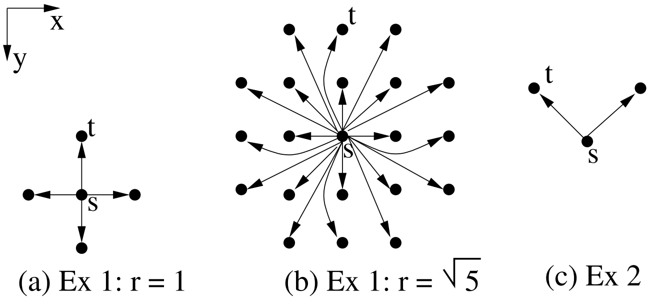
Images as graphs



(a) Ex 1: r = 1 (b) Ex 1: r = $\sqrt{5}$ (c) Ex 2

- Example 1: $(s,t) \in \mathcal{A}$ when $\|t - s\|^2 \leq r^2$, for $r \geq 1$.

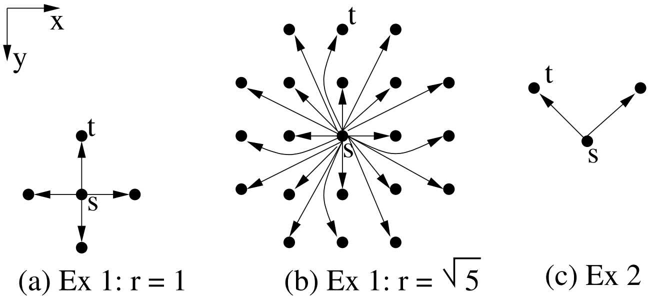
Images as graphs



(a) Ex 1: r = 1 (b) Ex 1: r = $\sqrt{5}$ (c) Ex 2

- Example 1: $(s,t) \in \mathcal{A}$ when $\|t-s\|^2 \leq r^2$, for $r \geq 1$.
- Example 2: $(s,t) \in \mathcal{A}$ when $t-s \in \{(-1,-1),(1,-1)\}$.

Images as graphs



(a) Ex 1: r = 1 (b) Ex 1: r = $\sqrt{5}$ (c) Ex 2

- Example 1: $(s, t) \in \mathcal{A}$ when $\|t - s\|^2 \leq r^2$, for $r \geq 1$.
- Example 2: $(s, t) \in \mathcal{A}$ when $t - s \in \{(-1, -1), (1, -1)\}$.
- Example 3: $(s, t) \in \mathcal{A}$ when t is a k-nearest neighbor of s in the feature space, for $k \geq 1$.

Images as graphs

Position invariant relations $\mathcal{A}$ can be represented by a vector of relative displacements

$$t - s \quad \in \quad \{(dx_1, dy_1), (dx_2, dy_2), \ldots, (dx_d, dy_d)\},$$

and fixed size $d = |\mathcal{A}(s)|\ \forall s$, leading to an implicit graph representation.

$$(x_t, y_t) \quad = \quad (x_s, y_s) + (dx_i, dy_i), i = 1, 2, \ldots, d,$$

where $t = (x_t, y_t)$ and $s = (x_s, y_s)$.

Images as graphs

The formalism of discrete mathematics naturally leads to the algorithm and its implementation. For example,

Images as graphs

The formalism of discrete mathematics naturally leads to the algorithm and its implementation. For example,

- A convolution kernel $\mathbf{K} = (\mathcal{A}, w)$ will consist of the adjacency relation $\mathcal{A}$ and a mapping $w(t-s) \in \{w_1, w_2, \ldots, w_d\}$ of fixed arc weights.

Images as graphs

The formalism of discrete mathematics naturally leads to the algorithm and its implementation. For example,

- A convolution kernel $\mathbf{K} = (\mathcal{A}, w)$ will consist of the adjacency relation $\mathcal{A}$ and a mapping $w(t-s) \in \{w_1, w_2, \ldots, w_d\}$ of fixed arc weights.
- The reflection $\mathbf{K}'$ of $\mathbf{K}$ around its origin is simply $(\mathcal{A}', w')$, where $\mathcal{A}' = \{(-dx_1, -dy_1), (-dx_2, -dy_2), \ldots, (-dx_d, -dy_d)\}$ and $w'(s-t) \in \{w_1, w_2, \ldots, w_d\}$.

Images as graphs

Then, the convolution $\hat{I} * \mathbf{K}$ creates an image $\hat{F} = (D_F, F)$, by

$$F(s) \quad = \quad \sum_{\forall t \in \mathcal{A}'(s)} I(t) w'(s-t),$$

for every $s \in D_I$. An example is a Gaussian filter:

Images as graphs

Then, the convolution $\hat{I} * \mathbf{K}$ creates an image $\hat{F} = (D_F, F)$, by

$$F(s) = \sum_{\forall t \in \mathcal{A}'(s)} I(t) w'(s-t),$$

for every $s \in D_I$. An example is a Gaussian filter:

Images as graphs

It is possible, for example, to reduce edge blurring in linear filtering by defining $(s,t) \in \mathcal{A}$ when $\|t-s\|^2 \leq r^2$, $r > 0$, and t is a k-nearest neighbor of s in the feature space.

- Original image.

Images as graphs

It is possible, for example, to reduce edge blurring in linear filtering by defining $(s, t) \in \mathcal{A}$ when $\|t - s\|^2 \leq r^2$, $r > 0$, and t is a k-nearest neighbor of s in the feature space.

- Original image.
- Gaussian filtering with $r = 7$.

Images as graphs

It is possible, for example, to reduce edge blurring in linear filtering by defining $(s,t) \in \mathcal{A}$ when $\|t-s\|^2 \leq r^2$, $r > 0$, and t is a k-nearest neighbor of s in the feature space.

- Original image.
- Gaussian filtering with $r = 7$.
- Gaussian filtering with the above adjacency, $r = 10$ and $k = 154$.

Connectivity functions

Connectivity-based operators use paths in $(\mathcal{N}, \mathcal{A})$.

Connectivity functions

Connectivity-based operators use paths in $(\mathcal{N}, \mathcal{A})$.

- A path π_t is a sequence of distinct adjacent nodes with terminus at node t.

Connectivity functions

Connectivity-based operators use paths in $(\mathcal{N}, \mathcal{A})$.

- A path π_t is a sequence of distinct adjacent nodes with terminus at node t.
- A spel t is said connected to a spel r if there exists a path π_t from r to t, in which case the root $R(\pi_t) = r$.

Connectivity functions

Connectivity-based operators use paths in $(\mathcal{N}, \mathcal{A})$.

- A path π_t is a sequence of distinct adjacent nodes with terminus at node t.
- A spel t is said connected to a spel r if there exists a path π_t from r to t, in which case the root $R(\pi_t) = r$.
- A connectivity function $f(\pi_t)$ assigns a value to any path.

Connectivity functions

Connectivity-based operators use paths in $(\mathcal{N}, \mathcal{A})$.

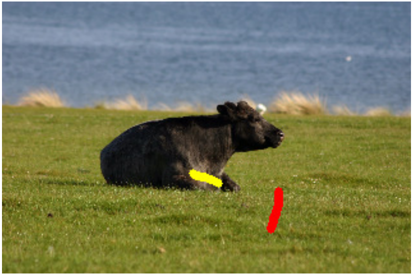
- A path π_t is a sequence of distinct adjacent nodes with terminus at node t.
- A spel t is said connected to a spel r if there exists a path π_t from r to t, in which case the root $R(\pi_t) = r$.
- A connectivity function $f(\pi_t)$ assigns a value to any path.
- A path π_t is optimum if $f(\pi_t) \leq f(\tau_t)$ for any other τ_t (minimum).

Connectivity functions

Connectivity-based operators use paths in $(\mathcal{N}, \mathcal{A})$.

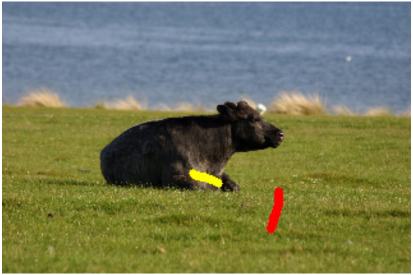
- A path π_t is a sequence of distinct adjacent nodes with terminus at node t.
- A spel t is said connected to a spel r if there exists a path π_t from r to t, in which case the root $R(\pi_t) = r$.
- A connectivity function $f(\pi_t)$ assigns a value to any path.
- A path π_t is optimum if $f(\pi_t) \leq f(\tau_t)$ for any other τ_t (minimum).
- The dual definition $f(\pi_t) \geq f(\tau_t)$ (maximum) is also valid.

Connectivity functions



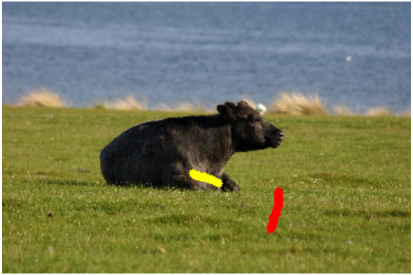
- Consider for example a segmentation problem, where internal $\mathcal{S}_i$ (yellow) and external $\mathcal{S}_e$ (red) sets of seed spels are given.

Connectivity functions



- Consider for example a segmentation problem, where internal $\mathcal{S}_i$ (yellow) and external $\mathcal{S}_e$ (red) sets of seed spels are given.
- The seeds may compete among themselves by offering optimum paths to every spel in the image.

Connectivity functions



- Consider for example a segmentation problem, where internal $\mathcal{S}_i$ (yellow) and external $\mathcal{S}_e$ (red) sets of seed spels are given.
- The seeds may compete among themselves by offering optimum paths to every spel in the image.
- The object can be defined by spels t whose optimum paths π_t have roots $R(\pi_t)$ in $\mathcal{S}_i$.

Connectivity functions

We may

- interpret the image as an 8-neighborhood graph,

Connectivity functions

We may

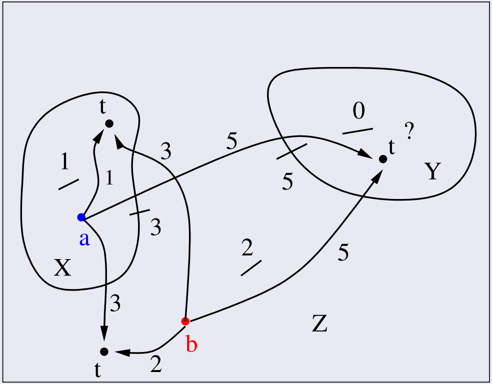
- interpret the image as an 8-neighborhood graph,
- assign higher arc weights $w(s,t)$ across the object's boundary than inside and outside it, and

Connectivity functions

We may

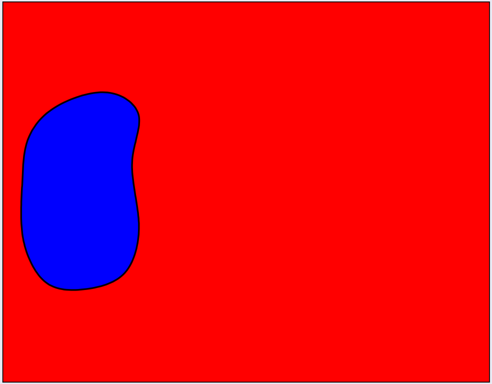
- interpret the image as an 8-neighborhood graph,
- assign higher arc weights $w(s, t)$ across the object's boundary than inside and outside it, and
- define the value of a path to be the maximum arc weight along it, such that any path that crosses the boundary will be penalized.

The IFT computation



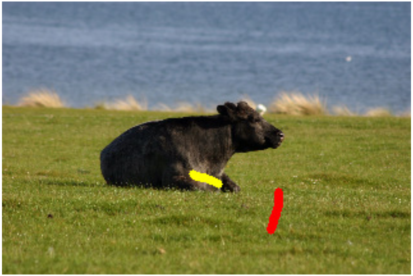
Seeds *a* and *b* compete between them, but *b* conquers all spels *t* in component *Y* because it will be surrounded by optimum paths rooted at *b*.

The IFT computation



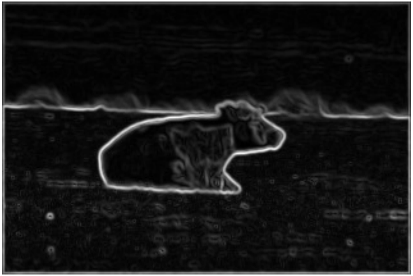
Seeds a and b compete between them, but b conquers all spels t in component Y because it will be surrounded by optimum paths rooted at b.

Connectivity functions



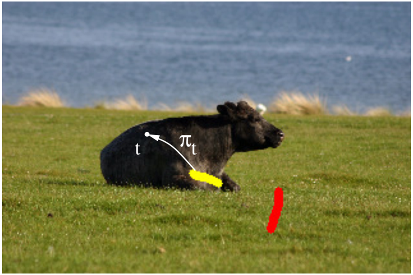
- Image with internal and external markers.

Connectivity functions



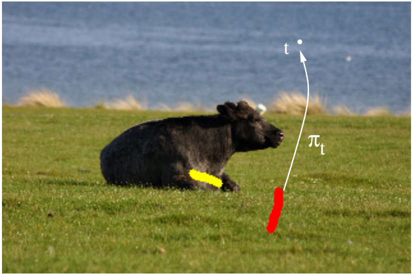
- Image with internal and external markers.
- Arc-weight image.

Connectivity functions



- Image with internal and external markers.
- Arc-weight image.
- Optimum-paths to foreground pixels.

Connectivity functions



- Image with internal and external markers.
- Arc-weight image.
- Optimum-paths to foreground pixels.
- Optimum-paths to background pixels.

Connectivity functions

- Image with internal and external markers.
- Arc-weight image.
- Optimum-paths to foreground pixels.
- Optimum-paths to background pixels.
- Segmentation result.

Connectivity functions

show video-iftsc.gif

- Image with internal and external markers.
- Arc-weight image.
- Optimum-paths to foreground pixels.
- Optimum-paths to background pixels.
- Segmentation result.

Connectivity functions

More formally,

Connectivity functions

More formally,

- a path $\pi_t = \pi_s \cdot \langle s, t \rangle$ is the extension of a path π_s by an arc (s, t), being $\pi_t = \langle t \rangle$ a trivial path, and

Connectivity functions

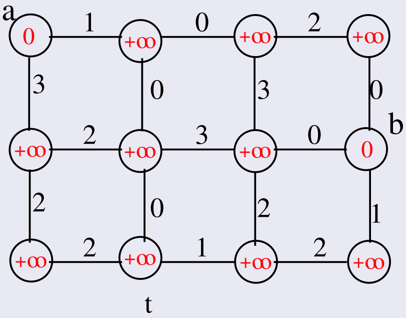
More formally,

- a path $\pi_t = \pi_s \cdot \langle s, t \rangle$ is the extension of a path π_s by an arc (s, t), being $\pi_t = \langle t \rangle$ a trivial path, and
- the max-arc path function is defined as

$$\begin{aligned} f_{\max}(\langle t \rangle) &= \begin{cases} 0 & \text{if } t \in \mathcal{S} = \mathcal{S}_i \cup \mathcal{S}_e \\ +\infty & \text{otherwise} \end{cases} \\ f_{\max}(\pi_s \cdot \langle s, t \rangle) &= \max\{f_{\max}(\pi_s), w(s, t)\}. \end{aligned}$$

Connectivity functions

Consider, for example, a 4-neighborhood graph, path function $f_{\max}$ and two seeds $\mathcal{S}_i = \{a\}$ and $\mathcal{S}_e = \{b\}$.

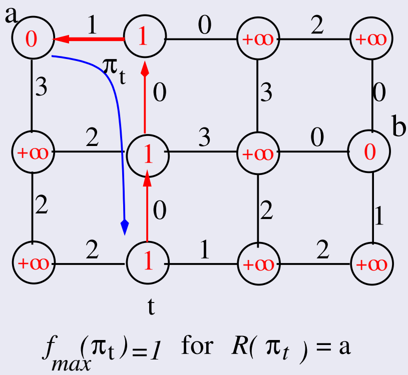


4−neighborhood graph

Paths are represented in backwards.

Connectivity functions

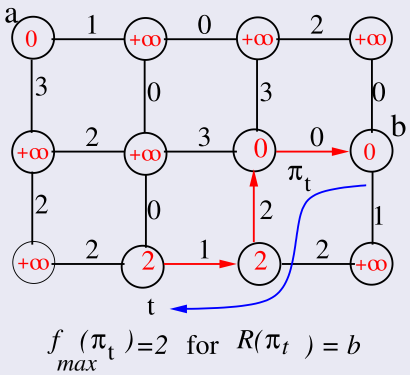
Consider, for example, a 4-neighborhood graph, path function $f_{\max}$ and two seeds $\mathcal{S}_i = \{a\}$ and $\mathcal{S}_e = \{b\}$.



Paths are represented in backwards.

Connectivity functions

Consider, for example, a 4-neighborhood graph, path function $f_{\max}$ and two seeds $\mathcal{S}_i = \{a\}$ and $\mathcal{S}_e = \{b\}$.



Paths are represented in backwards.

Connectivity functions

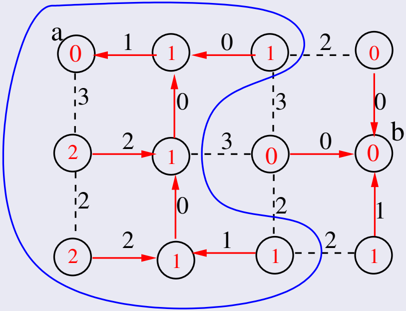
The segmentation essentially minimizes a connectivity map

$$V(t) \quad = \quad \min_{\forall \pi_t \in \Pi(\mathcal{N},\mathcal{A},t)} \{f_{\max}(\pi_t)\}$$

by considering the set $\Pi(\mathcal{N}, \mathcal{A}, t)$ of all paths with terminus t and function $f_{\max}$.

Connectivity function

The IFT algorithm solves this problem by computing an optimum-path forest P in $(\mathcal{N}, \mathcal{A})$ — a predecessor map with no cycles, containing all optimum paths from a root set $\mathcal{R}$, which in this case is $\mathcal{S} = \mathcal{S}_i \cup \mathcal{S}_e$.



The object is defined by the optimum forest for $f_{\max}$ rooted in $\mathcal{S}_i$.

Connectivity function

An optimum-path forest for $f_{\max}$ also provides the graph cut whose minimum arc weight

$$\min_{\forall (s,t) \in \mathcal{A}, R(\pi_s)=a, R(\pi_t)=b} w(s,t)$$

is maximum, considering all possible cuts between a and b [6].

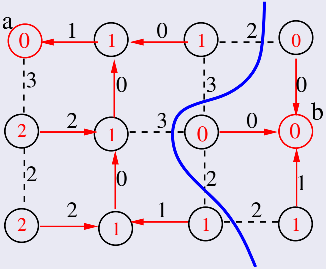


Image Foresting Transform

The IFT essentially reduces a given image processing problem into the computation of an optimum-path forest followed by a local processing of its attributes:

Image Foresting Transform

The IFT essentially reduces a given image processing problem into the computation of an optimum-path forest followed by a local processing of its attributes:

- the connectivity map $V(t)$,

Image Foresting Transform

The IFT essentially reduces a given image processing problem into the computation of an optimum-path forest followed by a local processing of its attributes:

- the connectivity map $V(t)$,
- the optimum paths in $P(t)$, and

Image Foresting Transform

The IFT essentially reduces a given image processing problem into the computation of an optimum-path forest followed by a local processing of its attributes:

- the connectivity map $V(t)$,
- the optimum paths in $P(t)$, and
- a root label $L(t)$.

The IFT computation

The IFT computation

- First, all nodes $t \in \mathcal{N}$ are trivial paths with initial connectivity values $V_0(t) = f(\langle t \rangle)$.

The IFT computation

- First, all nodes $t \in \mathcal{N}$ are trivial paths with initial connectivity values $V_0(t) = f(\langle t \rangle)$.
- The initial roots are identified at the global minima of $V_0(t)$.

The IFT computation

- First, all nodes $t \in \mathcal{N}$ are trivial paths with initial connectivity values $V_0(t) = f(\langle t \rangle)$.
- The initial roots are identified at the global minima of $V_0(t)$.
- They may conquer their adjacent nodes by offering them better paths.

The IFT computation

- First, all nodes $t \in \mathcal{N}$ are trivial paths with initial connectivity values $V_0(t) = f(\langle t \rangle)$.
- The initial roots are identified at the global minima of $V_0(t)$.
- They may conquer their adjacent nodes by offering them better paths.
- The process continues from the adjacent nodes in a non-decreasing order of path values.

$$\text{if } f(\pi_s \cdot \langle s, t \rangle) < f(\pi_t) \quad \text{then } \pi_t \leftarrow \pi_s \cdot \langle s, t \rangle.$$

The IFT computation

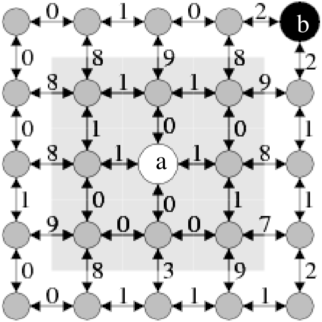
- First, all nodes $t \in \mathcal{N}$ are trivial paths with initial connectivity values $V_0(t) = f(\langle t \rangle)$.
- The initial roots are identified at the global minima of $V_0(t)$.
- They may conquer their adjacent nodes by offering them better paths.
- The process continues from the adjacent nodes in a non-decreasing order of path values.

$$\text{if } f(\pi_s \cdot \langle s, t \rangle) < f(\pi_t) \quad \text{then } \pi_t \leftarrow \pi_s \cdot \langle s, t \rangle.$$

- Essentially the minima in $V_0(t)$ compete among themselves and some of them become roots in $\mathcal{R}$, being also minima in $V(t)$. show video-iftsc-computation.gif

Path propagation

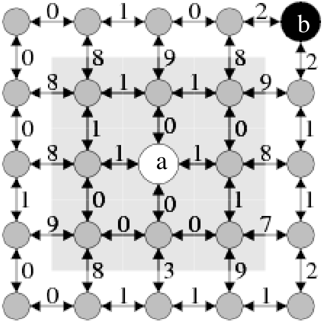
Consider the optimum path propagation for $f_{\max}$ from $\mathcal{S} = \{a, b\}$ in the 4-neighborhood graph below.

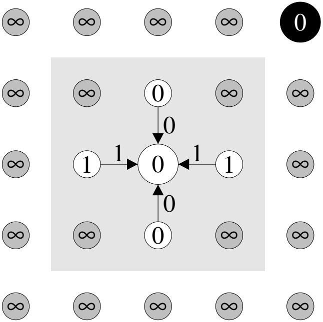


Object and background are separated by the arcs with higher weights.

Path propagation

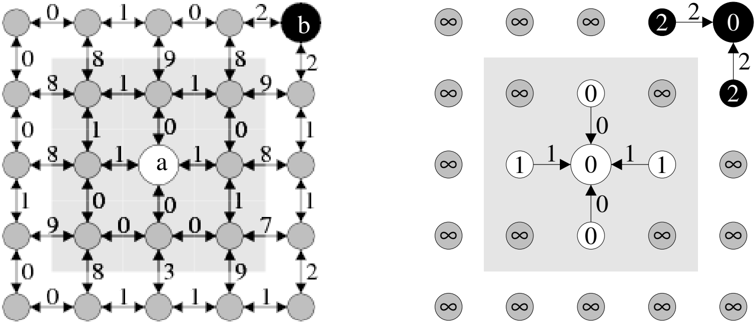
From iteration 1 to 5, iteration 12, 20, and 25.





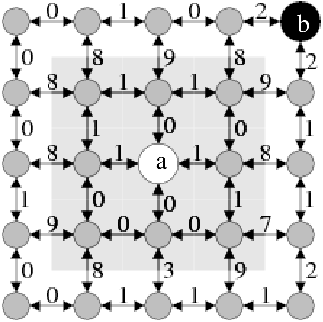
Path propagation

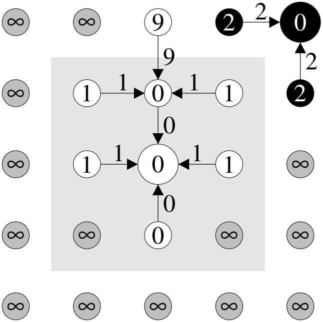
From iteration 1 to 5, iteration 12, 20, and 25.



Path propagation

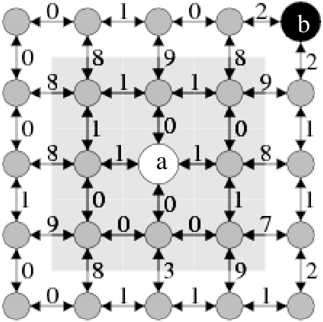
From iteration 1 to 5, iteration 12, 20, and 25.

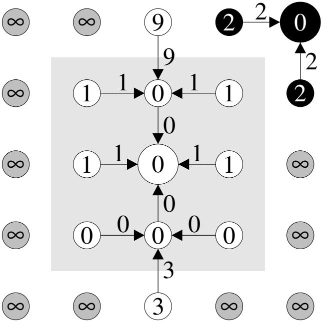




Path propagation

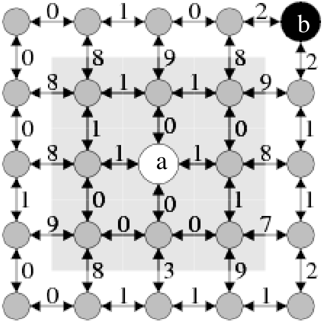
From iteration 1 to 5, iteration 12, 20, and 25.

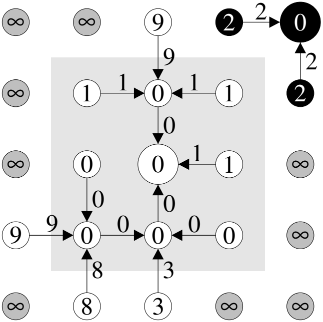




Path propagation

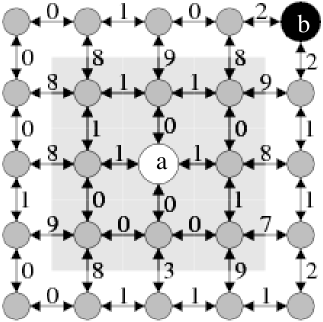
From iteration 1 to 5, iteration 12, 20, and 25.

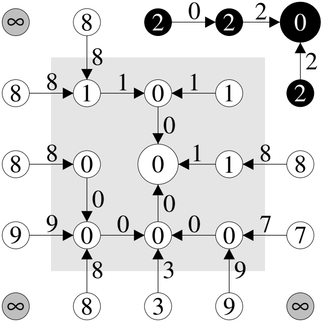




Path propagation

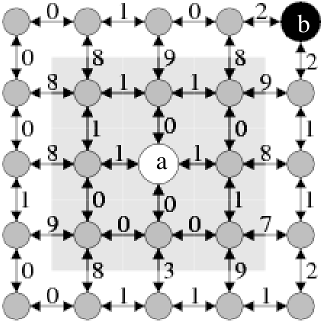
From iteration 1 to 5, iteration 12, 20, and 25.

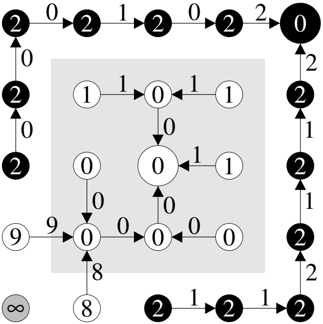




Path propagation

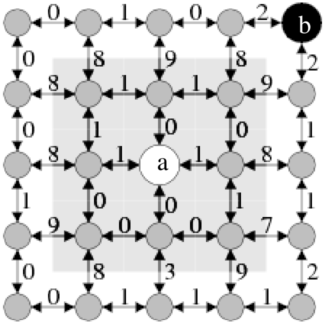
From iteration 1 to 5, iteration 12, 20, and 25.

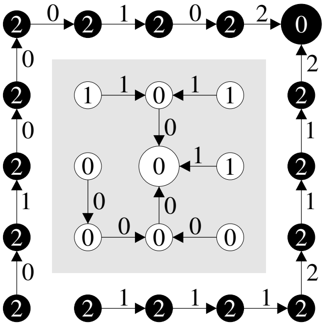




Path propagation

From iteration 1 to 5, iteration 12, 20, and 25.





Information propagation

Information propagation

- The IFT requires a priority queue Q for path propagation (modified Dijkstra's algorithm).

Information propagation

- The IFT requires a priority queue Q for path propagation (modified Dijkstra's algorithm).
- It can propagate other informations to each node:

Information propagation

- The IFT requires a priority queue Q for path propagation (modified Dijkstra's algorithm).
- It can propagate other informations to each node:
    - its propagation order [9],

Information propagation

- The IFT requires a priority queue Q for path propagation (modified Dijkstra's algorithm).
- It can propagate other informations to each node:
  - its propagation order [9],
  - a graph-cut measure [10], etc.

Information propagation

- The IFT requires a priority queue Q for path propagation (modified Dijkstra's algorithm).
- It can propagate other informations to each node:
    - its propagation order [9],
    - a graph-cut measure [10], etc.
- For its correctness, verify `www.math.wvu.edu/~kcies/SubmittedPapers/SS17DijkstraCharacterization.pdf`

The general IFT algorithm

Algorithm

– GENERAL IFT ALGORITHM

```
For each $t \in \mathcal{N}$, do
    Set $P(t) \leftarrow nil$, $L(t) \leftarrow t$ and $V(t) \leftarrow f(\langle t \rangle)$.
    If $V(t) \neq +\infty$, then insert $t$ in $Q$.
While $Q$ is not empty, do
    Remove from $Q$ a spel $s$ such that $V(s)$ is minimum.
    For each $t \in \mathcal{A}(s)$ such that $V(t) > V(s)$, do
        Compute $tmp \leftarrow f(\pi_s \cdot \langle s, t \rangle)$.
        If $tmp < V(t)$, then
            If $V(t) \neq +\infty$, remove $t$ from $Q$.
            Set $P(t) \leftarrow s$, $V(t) \leftarrow tmp$, $L(t) \leftarrow L(s)$.
            Insert $t$ in $Q$.
```

Important remarks and variants

- The queue Q is then a wavefront whose propagation from each root follows a non-decreasing order of path values.

Important remarks and variants

- The queue Q is then a wavefront whose propagation from each root follows a non-decreasing order of path values.
- Ties are broken in Q, by following the FIFO policy. A variant of the algorithm with LIFO tie-breaking policy is given in [1].

Important remarks and variants

- The queue Q is then a wavefront whose propagation from each root follows a non-decreasing order of path values.
- Ties are broken in Q, by following the FIFO policy. A variant of the algorithm with LIFO tie-breaking policy is given in [1].
- Other root information in a function $\lambda(t)$ can be propagated by setting $L(t) \leftarrow \lambda(t)$ in Line 2.

Important remarks and variants

- The queue Q is then a wavefront whose propagation from each root follows a non-decreasing order of path values.
- Ties are broken in Q, by following the FIFO policy. A variant of the algorithm with LIFO tie-breaking policy is given in [1].
- Other root information in a function $\lambda(t)$ can be propagated by setting $L(t) \leftarrow \lambda(t)$ in Line 2.
- Line 7 can be simplified to $tmp \leftarrow \max\{V(s), w(s,t)\}$ in the case of $f_{\max}$.

Important remarks and variants

- The queue Q is then a wavefront whose propagation from each root follows a non-decreasing order of path values.
- Ties are broken in Q, by following the FIFO policy. A variant of the algorithm with LIFO tie-breaking policy is given in [1].
- Other root information in a function $\lambda(t)$ can be propagated by setting $L(t) \leftarrow \lambda(t)$ in Line 2.
- Line 7 can be simplified to $tmp \leftarrow \max\{V(s), w(s,t)\}$ in the case of $f_{\max}$.
- The dual operation $V(t) = \max_{\forall \pi_t \in \Pi(\mathcal{N},\mathcal{A},t)}\{f_{\min}(\pi_t)\}$ requires: $V(t) \neq -\infty$ in Lines 3 and 9, $V(s)$ is maximum in Line 5, $V(t) < V(s)$ in Line 6, and $tmp > V(t)$ in Line 8.

Important remarks and variants

When s is removed from Q in Line 5:

Important remarks and variants

When s is removed from Q in Line 5:

- Path π_s is optimum, it can be obtained from P, which also contains all optimum paths π_r with values $V(r) < V(s)$.

Important remarks and variants

When s is removed from Q in Line 5:

- Path π_s is optimum, it can be obtained from P, which also contains all optimum paths π_r with values $V(r) < V(s)$.
- Besides that, if $P(s) = nil$ then s is a root node of $\mathcal{R}$, found on-the-fly.

Important remarks and variants

When s is removed from Q in Line 5:

- Path π_s is optimum, it can be obtained from P, which also contains all optimum paths π_r with values $V(r) < V(s)$.
- Besides that, if $P(s) = nil$ then s is a root node of $\mathcal{R}$, found on-the-fly.
- Early termination is possible whenever s is a destination node [2] or when $V(s)$ is greater than a given threshold [9].

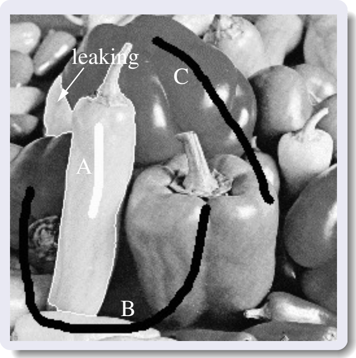
Important remarks and variants

When s is removed from Q in Line 5:

- Path π_s is optimum, it can be obtained from P, which also contains all optimum paths π_r with values $V(r) < V(s)$.
- Besides that, if $P(s) = nil$ then s is a root node of $\mathcal{R}$, found on-the-fly.
- Early termination is possible whenever s is a destination node [2] or when $V(s)$ is greater than a given threshold [9].
- Later, whenever necessary, the remaining optimum paths π_t with $V(t) \geq V(s)$ can be obtained incrementally from the nodes in Q.

Important remarks and variants

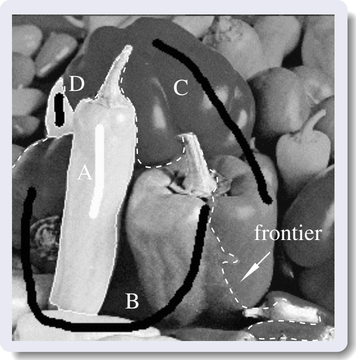
For a fixed path function f, the optimum forest can be updated in a differential way whenever we add/remove root nodes [3].



- Internal (A) and external (B and C) markers are selected, but a "leaking" occurs.

Important remarks and variants

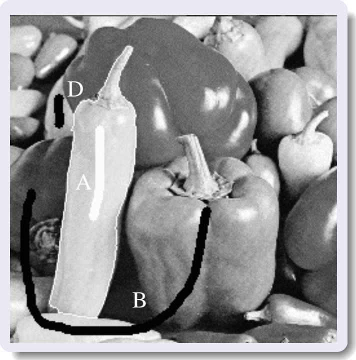
For a fixed path function f, the optimum forest can be updated in a differential way whenever we add/remove root nodes [3].



- Internal (A) and external (B and C) markers are selected, but a "leaking" occurs.
- We add an external marker D and select marker C for removal. The competition involves D and frontier spels (dashed line) of the forests of A and B.

Important remarks and variants

For a fixed path function f, the optimum forest can be updated in a differential way whenever we add/remove root nodes [3].



- Internal (A) and external (B and C) markers are selected, but a "leaking" occurs.
- We add an external marker D and select marker C for removal. The competition involves D and frontier spels (dashed line) of the forests of A and B.
- Result of segmentation.

Priority queue

Priority queue

- The IFT algorithm runs in $O(|\mathcal{A}| + |\mathcal{N}|^2)$.

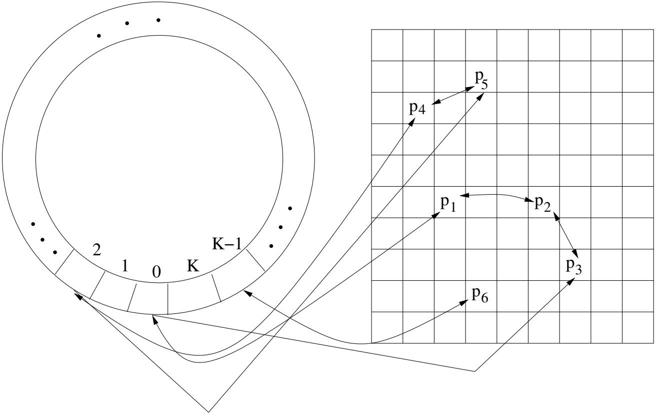
Priority queue

- The IFT algorithm runs in $O(|\mathcal{A}| + |\mathcal{N}|^2)$.
- Its running time is $O(|\mathcal{N}| \log(|\mathcal{N}|))$, when Q is a binary heap and the graph is sparse $(|\mathcal{A}| \ll |\mathcal{N}|^2)$.

Priority queue

- The IFT algorithm runs in $O(|\mathcal{A}| + |\mathcal{N}|^2)$.
- Its running time is $O(|\mathcal{N}| \log(|\mathcal{N}|))$, when Q is a binary heap and the graph is sparse ($|\mathcal{A}| \ll |\mathcal{N}|^2$).
- Its running time is $O(|\mathcal{N}|)$, when $f(\pi_s \cdot \langle s, t \rangle) - f(\pi_s) \in [0..K]$, $K \ll |\mathcal{N}|$, are integers, the graph is sparse, and Q uses bucket sort [2].

Priority queue



Nodes t are inserted in bucket $V(t)\%(K+1)$ (left), forming $K+1$ lists (right). The property $f(\pi_s \cdot \langle s,t\rangle) - f(\pi_s) \in [0..K]$ guarantees that nodes with different values are never in a same bucket.

Conclusion

Conclusion

- It should be clear the advantages of interpreting images as graphs for the design of adjacency-based and connectivity-based image transformations.

Conclusion

- It should be clear the advantages of interpreting images as graphs for the design of adjacency-based and connectivity-based image transformations.
- The IFT plays an important role in this scenario, given that it unifies image transformations based on optimum connectivity.

Conclusion

- It should be clear the advantages of interpreting images as graphs for the design of adjacency-based and connectivity-based image transformations.
- The IFT plays an important role in this scenario, given that it unifies image transformations based on optimum connectivity.
- It can be efficiently implemented in time proportional to the number of nodes in $\mathcal{N}$ for most image transformations.

[1] 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.

[2] A.X. Falcão, J.K. Udupa, and F.K. Miyazawa.

An ultra-fast user-steered image segmentation paradigm: Live-wire-on-the-fly.

IEEE Trans. on Medical Imaging, 19(1):55–62, Jan 2000.

[3] A. X. Falcão and F. P. G. Bergo.

Interactive volume segmentation with differential image foresting transforms.

IEEE Trans. on Medical Imaging, 23(9):1100–1108, 2004.

[4] F.P.G. Bergo and A.X. Falcão.

A partitioned algorithm for the image foresting transform.

In *Mathematical Morphology and its Applications to Signal and Image Processing (ISMM)*, pages 425–436, Rio de Janeiro, RJ, Oct 2007. MCT/INPE.

[5] A.X. Falcão, B. S. da Cunha, and R. A. Lotufo.

Design of connected operators using the image foresting transform.

In *SPIE on Medical Imaging*, volume 4322, pages 468–479, Feb 2001.

[6] P.A.V. Miranda and A.X. Falcão.

Links between image segmentation based on optimum-path forest and minimum cut in graph.

Journal of Mathematical Imaging and Vision, 35(2):128–142, Oct 2009.

doi:10.1007/s10851-009-0159-9.

[7] R. Audigier and R.A. Lotufo.

Watershed by image foresting transform, tie-zone, and theoretical relationship with other watershed definitions.

In *Mathematical Morphology and its Applications to Signal and Image Processing (ISMM)*, pages 277–288, Rio de Janeiro, RJ, Oct 2007. MCT/INPE.

[8] F. Cappabianco, G. Araujo, R. Azevedo, and A. Falcão.
A general image processing architecture for fpga.
In *V Southern Programmable Logic Conference*, pages 27–32, São Carlos, SP, 2009. Rima.

[9] P.A.V. Miranda, A.X. Falcão, A. Rocha, and F.P.G. Bergo.
Object delineation by κ-connected components.
EURASIP Journal on Advances in Signal Processing, 2008.

[10] A.X. Falcão, P.A.V. Miranda, and A. Rocha.
A linear-time approach for image segmentation using graph-cut measures.
In *8th Intl. Conf. on Advanced Concepts for Intelligent Vision Systems (ACIVS)*, volume LNCS 4179, pages 138–149, Antwerp, Belgium, 2006. Springer.