Hierarchical segmentation in a directed graph setting which optimizes a graph cut energy

Krzysztof Chris Ciesielski

Department of Mathematics, West Virginia University
and
MIPG, Department of Radiology, University of Pennsylvania

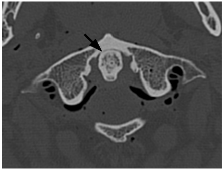
Centre for Image Analysis, Uppsala University, Sweden,
September 10, 2018

Outline

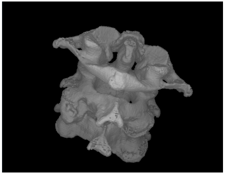
Outline

1. Image segmentation in graph cut setting
2. Dijkstra algorithm in general setting
3. Oriented IFT and graph cut optimization
4. HLOIFT: Hierarchical Layered OIFT algorithm
5. Experimental results for HLOIFT
6. Summary

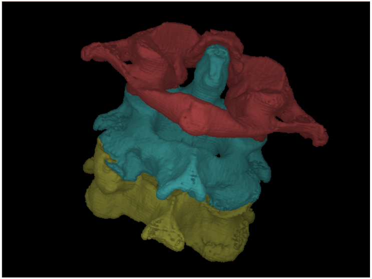
Image segmentation example 1: CT, cervical spine



A slice of an original 3D image

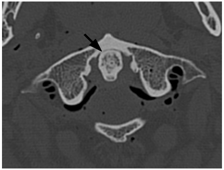


Surface rendition of segmented three vertebrae, together

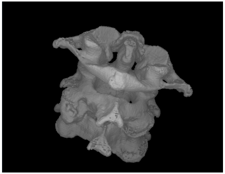


Color surface rendition of the segmented three vertebra

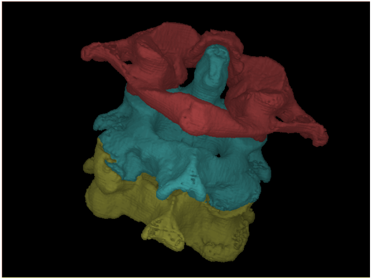
Image segmentation example 1: CT, cervical spine



A slice of an original 3D image

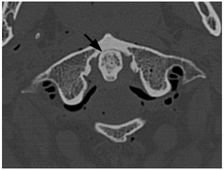


Surface rendition of segmented three vertebrae, together

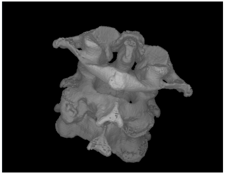


Color surface rendition of the segmented three vertebra

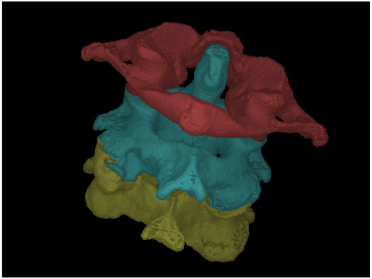
Image segmentation example 1: CT, cervical spine



A slice of an original 3D image



Surface rendition of segmented three vertebrae, together



Color surface rendition of the segmented three vertebra

Example 2: CT, thoracic-abdominal axial cross section

INPUT

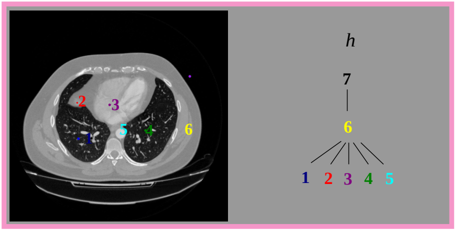


Figure: right lung (O_1), liver (O_2), heart (O_3), left lung (O_4), aorta (O_5) and the thoracic-abdominal region (O_6).

OUTPUT

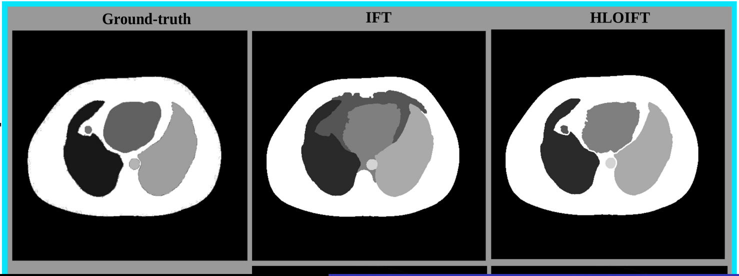


Image segmentation — formal setting

- An *image* is a map f from a set V (of spels) into $\mathbb{R}^k$

  The value $f(c)$ represents image intensity at c, a k-dimensional vector each component of which indicates a measure of some aspect of the signal, like color.

- *Segmentation problem:* Given an image $f\colon V \to \mathbb{R}^k$, find a "desired" family $\{O_1, \ldots, O_M\}$ of subsets of V.

- We will assume the objects are indicated by disjoint sets S_i of seeds, imposing that $S_i \subset O_i$.

Image segmentation — formal setting

- An *image* is a map f from a set V (of spels) into $\mathbb{R}^k$

  The value $f(c)$ represents image intensity at c, a k-dimensional vector each component of which indicates a measure of some aspect of the signal, like color.

- *Segmentation problem:* Given an image $f\colon V \to \mathbb{R}^k$,

  find a "desired" family $\{O_1, \ldots, O_M\}$ of subsets of V.

- We will assume the objects are indicated by disjoint sets S_i of seeds, imposing that $S_i \subset O_i$.

Image segmentation — formal setting

- An *image* is a map f from a set V (of spels) into $\mathbb{R}^k$

  The value $f(c)$ represents image intensity at c, a k-dimensional vector each component of which indicates a measure of some aspect of the signal, like color.

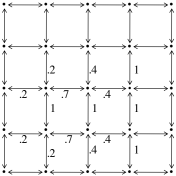
- *Segmentation problem:* Given an image $f\colon V \to \mathbb{R}^k$,

  find a “desired” family $\{O_1, \ldots, O_M\}$ of subsets of V.

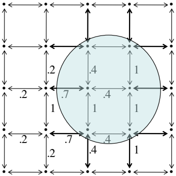
- We will assume the objects are indicated by disjoint sets S_i of seeds, imposing that $S_i \subset O_i$.

Image, its graph, and graph cut

An image, with intensity map $f\colon V \to \mathbb{R}^k$



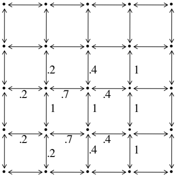
Its graph $G = \langle V, E \rangle$, with some edge weights

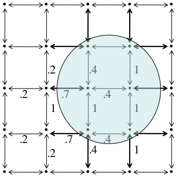


Object O and its graph cut edges $c(O)$ in bold

- Vertices $v \in V$ are image pixels. Direct edges: all $\langle c, d \rangle, \langle d, c \rangle \in E$, with $c, d \in V$ nearby (e.g. 4 adjacency).
- Edge weights: $w(\langle c, d \rangle) =$ some function of $f(c) - f(d)$.
- Graph cut of O: $c(O) = \{\langle c, d \rangle \in E \colon c \in O \ \&\ d \notin O\}$.
  Only in one direction!

Image, its graph, and graph cut





An image, with intensity map $f\colon V \to \mathbb{R}^k$

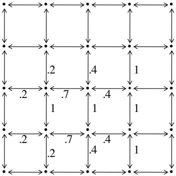
Its graph $G = \langle V, E \rangle$, with some edge weights

Object O and its graph cut edges $c(O)$ in bold

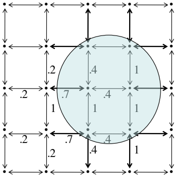
- Vertices $v \in V$ are image pixels. Direct edges: all $\langle c, d \rangle, \langle d, c \rangle \in E$, with $c, d \in V$ nearby (e.g. 4 adjacency).
- Edge weights: $w(\langle c, d \rangle) =$ some function of $f(c) - f(d)$.
- Graph cut of O: $c(O) = \{\langle c, d \rangle \in E \colon c \in O \; \& \; d \notin O\}$.
  Only in one direction!

Image, its graph, and graph cut

An image, with intensity map $f\colon V \to \mathbb{R}^k$



Its graph $G = \langle V, E \rangle$, with some edge weights



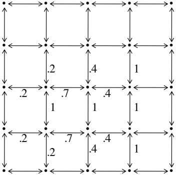
Object O and its graph cut edges $c(O)$ in bold

- Vertices $v \in V$ are image pixels. Direct edges: all $\langle c, d \rangle, \langle d, c \rangle \in E$, with $c, d \in V$ nearby (e.g. 4 adjacency).
- Edge weights: $w(\langle c, d \rangle) =$ some function of $f(c) - f(d)$.
- Graph cut of O: $c(O) = \{\langle c, d \rangle \in E\colon c \in O \;\&\; d \notin O\}$.

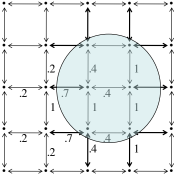
Only in one direction!

Image, its graph, and graph cut

An image, with intensity map $f\colon V \to \mathbb{R}^k$



Its graph $G = \langle V, E\rangle$, with some edge weights



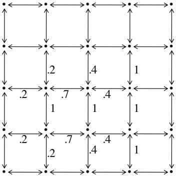
Object O and its graph cut edges $c(O)$ in bold

- Vertices $v \in V$ are image pixels. Direct edges: all $\langle c, d\rangle, \langle d, c\rangle \in E$, with $c, d \in V$ nearby (e.g. 4 adjacency).
- Edge weights: $w(\langle c, d\rangle) =$ some function of $f(c) - f(d)$.
- Graph cut of O: $c(O) = \{\langle c, d\rangle \in E\colon c \in O \,\&\, d \notin O\}$.

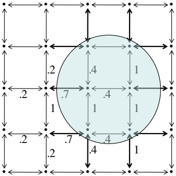
  Only in one direction!

Image, its graph, and graph cut

An image, with intensity map $f\colon V \to \mathbb{R}^k$



Its graph $G = \langle V, E \rangle$, with some edge weights

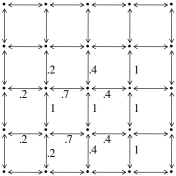


Object O and its graph cut edges $c(O)$ in bold

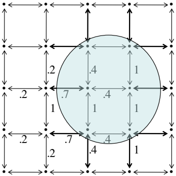
- Vertices $v \in V$ are image pixels. Direct edges: all $\langle c, d \rangle, \langle d, c \rangle \in E$, with $c, d \in V$ nearby (e.g. 4 adjacency).
- Edge weights: $w(\langle c, d \rangle) =$ some function of $f(c) - f(d)$.
- Graph cut of O: $c(O) = \{\langle c, d \rangle \in E \colon c \in O \;\&\; d \notin O\}$.
  Only in one direction!

Image, its graph, and graph cut

An image, with intensity map $f\colon V \to \mathbb{R}^k$



Its graph $G = \langle V, E \rangle$, with some edge weights



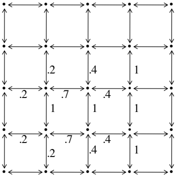
Object O and its graph cut edges $c(O)$ in bold

- Vertices $v \in V$ are image pixels. Direct edges: all $\langle c, d\rangle, \langle d, c\rangle \in E$, with $c, d \in V$ nearby (e.g. 4 adjacency).
- Edge weights: $w(\langle c, d\rangle) =$ some function of $f(c) - f(d)$.
- Graph cut of O: $c(O) = \{\langle c, d\rangle \in E\colon c \in O \;\&\; d \notin O\}$.

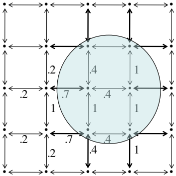
Only in one direction!

Image, its graph, and graph cut

An image, with intensity map $f\colon V \to \mathbb{R}^k$



Its graph $G = \langle V, E \rangle$, with some edge weights



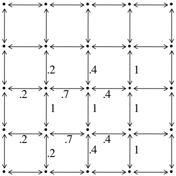
Object O and its graph cut edges $c(O)$ in bold

- Vertices $v \in V$ are image pixels. Direct edges: all $\langle c, d \rangle, \langle d, c \rangle \in E$, with $c, d \in V$ nearby (e.g. 4 adjacency).
- Edge weights: $w(\langle c, d \rangle) =$ some function of $f(c) - f(d)$.
- Graph cut of O: $c(O) = \{\langle c, d \rangle \in E\colon c \in O \;\&\; d \notin O\}$.

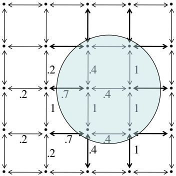
Only in one direction!

Image, its graph, and graph cut

An image, with intensity map $f\colon V \to \mathbb{R}^k$



Its graph $G = \langle V, E \rangle$, with some edge weights



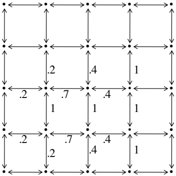
Object O and its graph cut edges $c(O)$ in bold

- Vertices $v \in V$ are image pixels. Direct edges: all $\langle c, d \rangle, \langle d, c \rangle \in E$, with $c, d \in V$ nearby (e.g. 4 adjacency).
- Edge weights: $w(\langle c, d \rangle) =$ some function of $f(c) - f(d)$.
- Graph cut of O: $c(O) = \{\langle c, d \rangle \in E\colon c \in O \,\&\, d \notin O\}$.

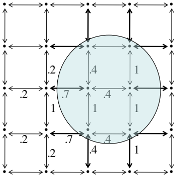
  Only in one direction!

Image, its graph, and graph cut

An image, with intensity map $f\colon V \to \mathbb{R}^k$



Its graph $G = \langle V, E \rangle$, with some edge weights



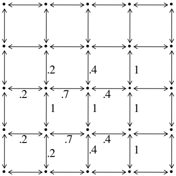
Object O and its graph cut edges $c(O)$ in bold

- Vertices $v \in V$ are image pixels. Direct edges: all $\langle c, d \rangle, \langle d, c \rangle \in E$, with $c, d \in V$ nearby (e.g. 4 adjacency).
- Edge weights: $w(\langle c, d \rangle) =$ some function of $f(c) - f(d)$.
- Graph cut of O: $c(O) = \{\langle c, d \rangle \in E\colon c \in O \;\&\; d \notin O\}$.

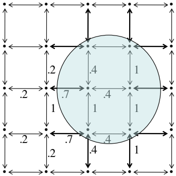
Only in one direction!

Image, its graph, and graph cut

An image, with intensity map $f\colon V \to \mathbb{R}^k$



Its graph $G = \langle V, E\rangle$, with some edge weights



Object O and its graph cut edges $c(O)$ in bold

- Vertices $v \in V$ are image pixels. Direct edges: all $\langle c, d\rangle, \langle d, c\rangle \in E$, with $c, d \in V$ nearby (e.g. 4 adjacency).
- Edge weights: $w(\langle c, d\rangle) =$ some function of $f(c) - f(d)$.
- Graph cut of O: $c(O) = \{\langle c, d\rangle \in E \colon c \in O \,\&\, d \notin O\}$.

  Only in one direction!

Graph cut measures: ℓ_p-norms, $1 \le p \le \infty$

Assuming $\langle c,d\rangle \in E \iff \langle d,c\rangle \in E$ and $w(\langle c,d\rangle) \ge 0$

ℓ_p-norm of $c(O)$ is defined as

$$\varepsilon_p(O) \stackrel{\text{def}}{=} \|w \restriction c(O)\|_p = \begin{cases} \left(\sum_{e\in c(O)} w(e)^p\right)^{1/p} & \text{if } p < \infty \\ \max_{e\in c(O)} w(e) & \text{if } p = \infty. \end{cases}$$

Standard analysis fact: $\|w\|_p \to_{p\to\infty} \|w\|_\infty$ for any map w.

Graph cut measures: ℓ_p-norms, $1 \le p \le \infty$

Assuming $\langle c,d\rangle \in E \iff \langle d,c\rangle \in E$ and $w(\langle c,d\rangle) \ge 0$

ℓ_p-norm of $c(O)$ is defined as

$$\varepsilon_p(O) \stackrel{\text{def}}{=} \|w \restriction c(O)\|_p = \begin{cases} \left(\sum_{e\in c(O)} w(e)^p\right)^{1/p} & \text{if } p < \infty \\ \max_{e\in c(O)} w(e) & \text{if } p = \infty. \end{cases}$$

Standard analysis fact: $\|w\|_p \to_{p\to\infty} \|w\|_\infty$ for any map w.

Graph cut measures: ℓ_p-norms, $1 \le p \le \infty$

Assuming $\langle c, d\rangle \in E \iff \langle d, c\rangle \in E$ and $w(\langle c, d\rangle) \ge 0$

ℓ_p-norm of $c(O)$ is defined as

$$\varepsilon_p(O) \stackrel{\text{def}}{=} \|w \restriction c(O)\|_p = \begin{cases} \left(\sum_{e\in c(O)} w(e)^p\right)^{1/p} & \text{if } p < \infty \\ \max_{e\in c(O)} w(e) & \text{if } p = \infty. \end{cases}$$

Standard analysis fact: $\|w\|_p \to_{p\to\infty} \|w\|_\infty$ for any map w.

Graph cut measures: ℓ_p-norms, $1 \leq p \leq \infty$

Assuming $\langle c, d\rangle \in E \iff \langle d, c\rangle \in E$ and $w(\langle c, d\rangle) \geq 0$

ℓ_p-norm of $c(O)$ is defined as

$$\varepsilon_p(O) \stackrel{\text{def}}{=} \|w \restriction c(O)\|_p = \begin{cases} \left(\sum_{e\in c(O)} w(e)^p\right)^{1/p} & \text{if } p < \infty \\ \max_{e\in c(O)} w(e) & \text{if } p = \infty. \end{cases}$$

Standard analysis fact: $\|w\|_p \to_{p\to\infty} \|w\|_\infty$ for any map w.

Known algorithms minimizing ℓ_p-norms of graph cut

$p = 1$: Minimization solved by classic min-cut/max-flow algorithm.

Graph Cut, GC, delineation algorithm minimizes ε_1.

$p = \infty$: Minimization solved by (versions of) Dijkstra algorithm.

ε_∞ minimized objects are returned by the algorithms:
Power Watershed, PW [C. Couprie *et al*, 2011]
Relative Fuzzy Connectedness, RFC, Iterative RFC, IRFC, Image Foresting Transform, IFT, [Ciesielski, Udupa, Falcão, Miranda, 2012].

$p = 2$: Random Walker, RW, algorithm [Grady, 2006].

Fact: Inclusion-minimal ℓ_p-normed minimized delineations converge, as $p \to \infty$ to ℓ_∞-normed minimized delineation.

This talk's Main Algorithm, HLOIFT, minimizes ℓ_∞-norm of cut.

Known algorithms minimizing ℓ_p-norms of graph cut

$p = 1$: Minimization solved by classic min-cut/max-flow algorithm.

Graph Cut, GC, delineation algorithm minimizes ε_1.

$p = \infty$: Minimization solved by (versions of) Dijkstra algorithm.

ε_∞ minimized objects are returned by the algorithms:
Power Watershed, PW [C. Couprie *et al*, 2011]
Relative Fuzzy Connectedness, RFC, Iterative RFC, IRFC, Image Foresting Transform, IFT, [Ciesielski, Udupa, Falcão, Miranda, 2012].

$p = 2$: Random Walker, RW, algorithm [Grady, 2006].

Fact: Inclusion-minimal ℓ_p-normed minimized delineations converge, as $p \to \infty$ to ℓ_∞-normed minimized delineation.

This talk's Main Algorithm, HLOIFT, minimizes ℓ_∞-norm of cut.

Known algorithms minimizing ℓ_p-norms of graph cut

$p = 1$: Minimization solved by classic min-cut/max-flow algorithm.

Graph Cut, GC, delineation algorithm minimizes ε_1.

$p = \infty$: Minimization solved by (versions of) Dijkstra algorithm.

ε_∞ minimized objects are returned by the algorithms:
Power Watershed, PW [C. Couprie *et al*, 2011]
Relative Fuzzy Connectedness, RFC, Iterative RFC, IRFC,
Image Foresting Transform, IFT, [Ciesielski, Udupa, Falcão, Miranda, 2012].

$p = 2$: Random Walker, RW, algorithm [Grady, 2006].

Fact: Inclusion-minimal ℓ_p-normed minimized delineations converge, as $p \to \infty$ to ℓ_∞-normed minimized delineation.

This talk's Main Algorithm, HLOIFT, minimizes ℓ_∞-norm of cut.

Known algorithms minimizing ℓ_p-norms of graph cut

$p = 1$: Minimization solved by classic min-cut/max-flow algorithm.

Graph Cut, GC, delineation algorithm minimizes ε_1.

$p = \infty$: Minimization solved by (versions of) Dijkstra algorithm.

ε_∞ minimized objects are returned by the algorithms:
Power Watershed, PW [C. Couprie *et al*, 2011]
Relative Fuzzy Connectedness, RFC, Iterative RFC, IRFC, Image Foresting Transform, IFT, [Ciesielski, Udupa, Falcão, Miranda, 2012].

$p = 2$: Random Walker, RW, algorithm [Grady, 2006].

Fact: Inclusion-minimal ℓ_p-normed minimized delineations converge, as $p \to \infty$ to ℓ_∞-normed minimized delineation.

This talk's Main Algorithm, HLOIFT, minimizes ℓ_∞-norm of cut.

Known algorithms minimizing ℓ_p-norms of graph cut

$p = 1$: Minimization solved by classic min-cut/max-flow algorithm.

Graph Cut, GC, delineation algorithm minimizes ε_1.

$p = \infty$: Minimization solved by (versions of) Dijkstra algorithm.

ε_∞ minimized objects are returned by the algorithms:
Power Watershed, PW [C. Couprie *et al*, 2011]
Relative Fuzzy Connectedness, RFC, Iterative RFC, IRFC,
Image Foresting Transform, IFT, [Ciesielski, Udupa, Falcão, Miranda, 2012].

$p = 2$: Random Walker, RW, algorithm [Grady, 2006].

Fact: Inclusion-minimal ℓ_p-normed minimized delineations converge, as $p \to \infty$ to ℓ_∞-normed minimized delineation.

This talk's Main Algorithm, HLOIFT, minimizes ℓ_∞-norm of cut.

Known algorithms minimizing ℓ_p-norms of graph cut

$p = 1$: Minimization solved by classic min-cut/max-flow algorithm.

Graph Cut, GC, delineation algorithm minimizes ε_1.

$p = \infty$: Minimization solved by (versions of) Dijkstra algorithm.

ε_∞ minimized objects are returned by the algorithms:
Power Watershed, PW [C. Couprie *et al*, 2011]
Relative Fuzzy Connectedness, RFC, Iterative RFC, IRFC, Image Foresting Transform, IFT, [Ciesielski, Udupa, Falcão, Miranda, 2012].

$p = 2$: Random Walker, RW, algorithm [Grady, 2006].

Fact: Inclusion-minimal ℓ_p-normed minimized delineations converge, as $p \to \infty$ to ℓ_∞-normed minimized delineation.

This talk's Main Algorithm, HLOIFT, minimizes ℓ_∞-norm of cut.

Known algorithms minimizing ℓ_p-norms of graph cut

$p = 1$: Minimization solved by classic min-cut/max-flow algorithm.

Graph Cut, GC, delineation algorithm minimizes ε_1.

$p = \infty$: Minimization solved by (versions of) Dijkstra algorithm.

ε_∞ minimized objects are returned by the algorithms:
Power Watershed, PW [C. Couprie *et al*, 2011]
Relative Fuzzy Connectedness, RFC, Iterative RFC, IRFC,
Image Foresting Transform, IFT, [Ciesielski, Udupa, Falcão, Miranda, 2012].

$p = 2$: Random Walker, RW, algorithm [Grady, 2006].

Fact: Inclusion-minimal ℓ_p-normed minimized delineations converge, as $p \to \infty$ to ℓ_∞-normed minimized delineation.

This talk's Main Algorithm, HLOIFT, minimizes ℓ_∞-norm of cut.

Known algorithms minimizing ℓ_p-norms of graph cut

$p = 1$: Minimization solved by classic min-cut/max-flow algorithm.

Graph Cut, GC, delineation algorithm minimizes ε_1.

$p = \infty$: Minimization solved by (versions of) Dijkstra algorithm.

ε_∞ minimized objects are returned by the algorithms:
Power Watershed, PW [C. Couprie *et al*, 2011]
Relative Fuzzy Connectedness, RFC, Iterative RFC, IRFC,
Image Foresting Transform, IFT, [Ciesielski, Udupa, Falcão, Miranda, 2012].

$p = 2$: Random Walker, RW, algorithm [Grady, 2006].

Fact: Inclusion-minimal ℓ_p-normed minimized delineations converge, as $p \to \infty$ to ℓ_∞-normed minimized delineation.

This talk's Main Algorithm, HLOIFT, minimizes ℓ_∞-norm of cut.

Known algorithms minimizing ℓ_p-norms of graph cut

$p = 1$: Minimization solved by classic min-cut/max-flow algorithm.

Graph Cut, GC, delineation algorithm minimizes ε_1.

$p = \infty$: Minimization solved by (versions of) Dijkstra algorithm.

ε_∞ minimized objects are returned by the algorithms:
Power Watershed, PW [C. Couprie *et al*, 2011]
Relative Fuzzy Connectedness, RFC, Iterative RFC, IRFC, Image Foresting Transform, IFT, [Ciesielski, Udupa, Falcão, Miranda, 2012].

$p = 2$: Random Walker, RW, algorithm [Grady, 2006].

Fact: Inclusion-minimal ℓ_p-normed minimized delineations converge, as $p \to \infty$ to ℓ_∞-normed minimized delineation.

This talk's Main Algorithm, HLOIFT, minimizes ℓ_∞-norm of cut.

Known algorithms minimizing ℓ_p-norms of graph cut

$p = 1$: Minimization solved by classic min-cut/max-flow algorithm.

Graph Cut, GC, delineation algorithm minimizes ε_1.

$p = \infty$: Minimization solved by (versions of) Dijkstra algorithm.

ε_∞ minimized objects are returned by the algorithms:
Power Watershed, PW [C. Couprie *et al*, 2011]
Relative Fuzzy Connectedness, RFC, Iterative RFC, IRFC,
Image Foresting Transform, IFT, [Ciesielski, Udupa, Falcão, Miranda, 2012].

$p = 2$: Random Walker, RW, algorithm [Grady, 2006].

Fact: Inclusion-minimal ℓ_p-normed minimized delineations converge, as $p \to \infty$ to ℓ_∞-normed minimized delineation.

This talk's Main Algorithm, HLOIFT, minimizes ℓ_∞-norm of cut.

Known algorithms minimizing ℓ_p-norms of graph cut

$p = 1$: Minimization solved by classic min-cut/max-flow algorithm.

Graph Cut, GC, delineation algorithm minimizes ε_1.

$p = \infty$: Minimization solved by (versions of) Dijkstra algorithm.

ε_∞ minimized objects are returned by the algorithms:
Power Watershed, PW [C. Couprie *et al*, 2011]
Relative Fuzzy Connectedness, RFC, Iterative RFC, IRFC,
Image Foresting Transform, IFT, [Ciesielski, Udupa,
Falcão, Miranda, 2012].

$p = 2$: Random Walker, RW, algorithm [Grady, 2006].

Fact: Inclusion-minimal ℓ_p-normed minimized delineations converge, as $p \to \infty$ to ℓ_∞-normed minimized delineation.

This talk's Main Algorithm, HLOIFT, minimizes ℓ_∞-norm of cut.

Outline

Paths and Optimal Path Forest OPF

- Fix directed graph $G = \langle V, E \rangle$ (with edge weight map w)
- *Path (in G)*: $p = \langle v_0, \ldots, v_\ell \rangle$ s.t. $\langle v_j, v_{j+1} \rangle \in E$ for $j < \ell$;
  p is from $S \subset V$ to $v \in V$ when $v_0 \in S$ and $v_\ell = v$;
  $p\hat{\ }w = \langle v_0, \ldots, v_\ell, w \rangle$; Π_G – all paths in G.
- Path cost function: any map $\psi \colon \Pi_G \to \mathbb{R}$.
- A path p (from $S \subset V$) to v is ψ-optimal provided

$$\psi(p) = \max\{\psi(q) \colon q \text{ is a path (from } S) \text{ to } v\}.$$

- Jarník-Prim-Dijkstra algorithm DA for ψ and $S \subset V$ tries to find (S-rooted) forest, OPF, composed of ψ-optimal paths.
- HLOIFT is a DA for appropriate path cost map and graph.

Paths and Optimal Path Forest OPF

- Fix directed graph $G = \langle V, E \rangle$ (with edge weight map w)
- *Path (in G)*: $p = \langle v_0, \ldots, v_\ell \rangle$ s.t. $\langle v_j, v_{j+1} \rangle \in E$ for $j < \ell$;

  p is from $S \subset V$ to $v \in V$ when $v_0 \in S$ and $v_\ell = v$;

  $p\hat{}w = \langle v_0, \ldots, v_\ell, w \rangle$; Π_G – all paths in G.

- Path cost function: any map $\psi \colon \Pi_G \to \mathbb{R}$.
- A path p (from $S \subset V$) to v is ψ-optimal provided

$$\psi(p) = \max\{\psi(q) \colon q \text{ is a path (from } S\text{) to } v\}.$$

- Jarník-Prim-Dijkstra algorithm DA for ψ and $S \subset V$ tries to find (S-rooted) forest, OPF, composed of ψ-optimal paths.
- HLOIFT is a DA for appropriate path cost map and graph.

Paths and Optimal Path Forest OPF

- Fix directed graph $G = \langle V, E \rangle$ (with edge weight map w)
- *Path (in G)*: $p = \langle v_0, \ldots, v_\ell \rangle$ s.t. $\langle v_j, v_{j+1} \rangle \in E$ for $j < \ell$;

  p is from $S \subset V$ to $v \in V$ when $v_0 \in S$ and $v_\ell = v$;

  $p\hat{}w = \langle v_0, \ldots, v_\ell, w \rangle$; Π_G – all paths in G.

- Path cost function: any map $\psi \colon \Pi_G \to \mathbb{R}$.
- A path p (from $S \subset V$) to v is ψ-optimal provided

$$\psi(p) = \max\{\psi(q) \colon q \text{ is a path (from } S) \text{ to } v\}.$$

- Jarník-Prim-Dijkstra algorithm DA for ψ and $S \subset V$ tries to find (S-rooted) forest, OPF, composed of ψ-optimal paths.
- HLOIFT is a DA for appropriate path cost map and graph.

Paths and Optimal Path Forest OPF

- Fix directed graph $G = \langle V, E \rangle$ (with edge weight map w)
- *Path (in G)*: $p = \langle v_0, \ldots, v_\ell \rangle$ s.t. $\langle v_j, v_{j+1} \rangle \in E$ for $j < \ell$;
  p is from $S \subset V$ to $v \in V$ when $v_0 \in S$ and $v_\ell = v$;
  $p\hat{\ } w = \langle v_0, \ldots, v_\ell, w \rangle$; Π_G – all paths in G.
- Path cost function: any map $\psi \colon \Pi_G \to \mathbb{R}$.
- A path p (from $S \subset V$) to v is ψ-optimal provided

$$\psi(p) = \max\{\psi(q) \colon q \text{ is a path (from } S) \text{ to } v\}.$$

- Jarník-Prim-Dijkstra algorithm DA for ψ and $S \subset V$ tries to find (S-rooted) forest, OPF, composed of ψ-optimal paths.
- HLOIFT is a DA for appropriate path cost map and graph.

Paths and Optimal Path Forest OPF

- Fix directed graph $G = \langle V, E \rangle$ (with edge weight map w)
- *Path (in G)*: $p = \langle v_0, \ldots, v_\ell \rangle$ s.t. $\langle v_j, v_{j+1} \rangle \in E$ for $j < \ell$;
  p is from $S \subset V$ to $v \in V$ when $v_0 \in S$ and $v_\ell = v$;
  $p\hat{}w = \langle v_0, \ldots, v_\ell, w \rangle$; Π_G – all paths in G.
- Path cost function: any map $\psi \colon \Pi_G \to \mathbb{R}$.
- A path p (from $S \subset V$) to v is ψ-optimal provided

$$\psi(p) = \max\{\psi(q) \colon q \text{ is a path (from } S\text{) to } v\}.$$

- Jarník-Prim-Dijkstra algorithm DA for ψ and $S \subset V$ tries to find (S-rooted) forest, OPF, composed of ψ-optimal paths.
- HLOIFT is a DA for appropriate path cost map and graph.

Paths and Optimal Path Forest OPF

- Fix directed graph $G = \langle V, E \rangle$ (with edge weight map w)
- *Path (in G)*: $p = \langle v_0, \ldots, v_\ell \rangle$ s.t. $\langle v_j, v_{j+1} \rangle \in E$ for $j < \ell$;
  p is from $S \subset V$ to $v \in V$ when $v_0 \in S$ and $v_\ell = v$;
  $p\hat{\ } w = \langle v_0, \ldots, v_\ell, w \rangle$; Π_G – all paths in G.
- Path cost function: any map $\psi \colon \Pi_G \to \mathbb{R}$.
- A path p (from $S \subset V$) to v is ψ-optimal provided

$$\psi(p) = \max\{\psi(q) \colon q \text{ is a path (from } S\text{) to } v\}.$$

- Jarník-Prim-Dijkstra algorithm DA for ψ and $S \subset V$ tries to find (S-rooted) forest, OPF, composed of ψ-optimal paths.
- HLOIFT is a DA for appropriate path cost map and graph.

Paths and Optimal Path Forest OPF

- Fix directed graph $G = \langle V, E \rangle$ (with edge weight map w)
- *Path (in G)*: $p = \langle v_0, \ldots, v_\ell \rangle$ s.t. $\langle v_j, v_{j+1} \rangle \in E$ for $j < \ell$;
  p is from $S \subset V$ to $v \in V$ when $v_0 \in S$ and $v_\ell = v$;
  $p\hat{\ } w = \langle v_0, \ldots, v_\ell, w \rangle$; Π_G – all paths in G.
- Path cost function: any map $\psi \colon \Pi_G \to \mathbb{R}$.
- A path p (from $S \subset V$) to v is ψ-optimal provided

$$\psi(p) = \max\{\psi(q) \colon q \text{ is a path (from } S\text{) to } v\}.$$

- Jarník-Prim-Dijkstra algorithm DA for ψ and $S \subset V$ tries to find (S-rooted) forest, OPF, composed of ψ-optimal paths.
- HLOIFT is a DA for appropriate path cost map and graph.

Paths and Optimal Path Forest OPF

- Fix directed graph $G = \langle V, E \rangle$ (with edge weight map w)
- *Path (in G)*: $p = \langle v_0, \ldots, v_\ell \rangle$ s.t. $\langle v_j, v_{j+1} \rangle \in E$ for $j < \ell$;
  p is from $S \subset V$ to $v \in V$ when $v_0 \in S$ and $v_\ell = v$;
  $p \hat{} w = \langle v_0, \ldots, v_\ell, w \rangle$; Π_G – all paths in G.
- Path cost function: any map $\psi \colon \Pi_G \to \mathbb{R}$.
- A path p (from $S \subset V$) to v is ψ-optimal provided

$$\psi(p) = \max\{\psi(q) \colon q \text{ is a path (from } S\text{) to } v\}.$$

- Jarník-Prim-Dijkstra algorithm DA for ψ and $S \subset V$ tries to find (S-rooted) forest, OPF, composed of ψ-optimal paths.
- HLOIFT is a DA for appropriate path cost map and graph.

Paths and Optimal Path Forest OPF

- Fix directed graph $G = \langle V, E \rangle$ (with edge weight map w)
- *Path (in G)*: $p = \langle v_0, \ldots, v_\ell \rangle$ s.t. $\langle v_j, v_{j+1} \rangle \in E$ for $j < \ell$;
  p is from $S \subset V$ to $v \in V$ when $v_0 \in S$ and $v_\ell = v$;
  $p\hat{\ }w = \langle v_0, \ldots, v_\ell, w \rangle$; Π_G – all paths in G.
- Path cost function: any map $\psi \colon \Pi_G \to \mathbb{R}$.
- A path p (from $S \subset V$) to v is ψ-optimal provided

$$\psi(p) = \max\{\psi(q) \colon q \text{ is a path (from } S \text{) to } v\}.$$

- Jarník-Prim-Dijkstra algorithm DA for ψ and $S \subset V$ tries to find (S-rooted) forest, OPF, composed of ψ-optimal paths.
- HLOIFT is a DA for appropriate path cost map and graph.

Dijkstra Algorithm, DA, *aiming* to find ψ-optimal forest

Data: $G = \langle V, E \rangle$ and a path cost map $\psi \colon \Pi_G \to \mathbb{R}$
Result: an array $\pi[\,]$ of paths, aiming for being ψ-optimal

```
foreach v ∈ V do π[v] ← ⟨v⟩
Q ← V
while Q ≠ ∅ do
    remove an element w of max_{u∈Q} ψ(π[u]) from Q
    foreach x such that ⟨w, x⟩ ∈ E do
        if ψ(π[x]) < ψ(π[w]^x) then π[x] ← π[w]^x
```

DA is very efficient: quasi-linear w.r.t. the size of the graph.

Dijkstra Algorithm, DA, *aiming* to find ψ-optimal forest

Data: $G = \langle V, E \rangle$ and a path cost map $\psi \colon \Pi_G \to \mathbb{R}$
Result: an array $\pi[\,]$ of paths, aiming for being ψ-optimal

```
foreach v ∈ V do π[v] ← ⟨v⟩
Q ← V
while Q ≠ ∅ do
    remove an element w of max_{u∈Q} ψ(π[u]) from Q
    foreach x such that ⟨w, x⟩ ∈ E do
        if ψ(π[x]) < ψ(π[w]^x) then π[x] ← π[w]^x
```

DA is very efficient: quasi-linear w.r.t. the size of the graph.

Dijkstra Algorithm, DA, *aiming* to find ψ-optimal forest

Data: $G = \langle V, E \rangle$ and a path cost map $\psi \colon \Pi_G \to \mathbb{R}$
Result: an array $\pi[\,]$ of paths, aiming for being ψ-optimal

foreach $v \in V$ **do** $\pi[v] \leftarrow \langle v \rangle$
$Q \leftarrow V$
while $Q \neq \emptyset$ **do**
  remove an element w of $\max_{u \in Q} \psi(\pi[u])$ **from** Q
  foreach x *such that* $\langle w, x \rangle \in E$ **do**
    if $\psi(\pi[x]) < \psi(\pi[w]\hat{}x)$ **then** $\pi[x] \leftarrow \pi[w]\hat{}x$

DA is very efficient: quasi-linear w.r.t. the size of the graph.

Dijkstra Algorithm, DA, *aiming* to find ψ-optimal forest

Data: $G = \langle V, E \rangle$ and a path cost map $\psi \colon \Pi_G \to \mathbb{R}$
Result: an array $\pi[\,]$ of paths, aiming for being ψ-optimal

foreach $v \in V$ **do** $\pi[v] \leftarrow \langle v \rangle$
$Q \leftarrow V$
while $Q \neq \emptyset$ **do**
  remove an element w of $\max_{u \in Q} \psi(\pi[u])$ **from** Q
  foreach x *such that* $\langle w, x \rangle \in E$ **do**
    if $\psi(\pi[x]) < \psi(\pi[w]\hat{\ }x)$ **then** $\pi[x] \leftarrow \pi[w]\hat{\ }x$

DA is very efficient: quasi-linear w.r.t. the size of the graph.

Dijkstra Algorithm, DA, *aiming* to find ψ-optimal forest

Data: $G = \langle V, E \rangle$ and a path cost map $\psi \colon \Pi_G \to \mathbb{R}$
Result: an array $\pi[\,]$ of paths, aiming for being ψ-optimal

foreach $v \in V$ **do** $\pi[v] \leftarrow \langle v \rangle$
$Q \leftarrow V$
while $Q \neq \emptyset$ **do**
  remove an element w of $\max_{u \in Q} \psi(\pi[u])$ **from** Q
  foreach x *such that* $\langle w, x \rangle \in E$ **do**
    if $\psi(\pi[x]) < \psi(\pi[w]\hat{\ }x)$ **then** $\pi[x] \leftarrow \pi[w]\hat{\ }x$

DA is very efficient: quasi-linear w.r.t. the size of the graph.

For what path cost ψ DA works properly?

Studied in JMIV paper [Ciesielski, Falcão, Miranda, Sept. 2018]

correcting errors of TPAMI paper [Falcão, Stolfi, Lotufo, 2004].

If w is an edge weight map for undirected graph $G = \langle V, E\rangle$, then DA works properly for:

- FC/IFT: $\psi_{\min}(\langle v_0, \ldots, v_\ell\rangle) = \min_{1\le j\le \ell} w(v_{j-1}, v_j)$ for $\ell > 0$

  $\psi_{\min}(\langle v_0\rangle) = \infty$ if $v_0 \in S$, $\psi_{\min}(\langle v_0\rangle) = -\infty$ if $v_0 \notin S$
- $\psi_{\mathrm{sum}}(\langle v_0, \ldots, v_\ell\rangle) = -\sum_{1\le j\le \ell} w(v_{j-1}, v_j)$ for $\ell > 0$

  $\psi_{\mathrm{sum}}(\langle v_0\rangle) = \infty$ if $v_0 \in S$, $\psi_{\mathrm{sum}}(\langle v_0\rangle) = -\infty$ if $v_0 \notin S$
- HLOIFT uses DA with $\psi_{\min}$ and oriented w, a problem!

For what path cost ψ DA works properly?

Studied in JMIV paper [Ciesielski, Falcão, Miranda, Sept. 2018]

correcting errors of TPAMI paper [Falcão, Stolfi, Lotufo, 2004].

If w is an edge weight map for undirected graph $G = \langle V, E \rangle$, then DA works properly for:

- FC/IFT: $\psi_{\min}(\langle v_0, \ldots, v_\ell \rangle) = \min_{1 \le j \le \ell} w(v_{j-1}, v_j)$ for $\ell > 0$

  $\psi_{\min}(\langle v_0 \rangle) = \infty$ if $v_0 \in S$, $\psi_{\min}(\langle v_0 \rangle) = -\infty$ if $v_0 \notin S$

- $\psi_{\text{sum}}(\langle v_0, \ldots, v_\ell \rangle) = -\sum_{1 \le j \le \ell} w(v_{j-1}, v_j)$ for $\ell > 0$

  $\psi_{\text{sum}}(\langle v_0 \rangle) = \infty$ if $v_0 \in S$, $\psi_{\text{sum}}(\langle v_0 \rangle) = -\infty$ if $v_0 \notin S$

- HLOIFT uses DA with $\psi_{\min}$ and oriented w, a problem!

For what path cost ψ DA works properly?

Studied in JMIV paper [Ciesielski, Falcão, Miranda, Sept. 2018]

correcting errors of TPAMI paper [Falcão, Stolfi, Lotufo, 2004].

If w is an edge weight map for undirected graph $G = \langle V, E \rangle$,
then DA works properly for:

- FC/IFT: $\psi_{\min}(\langle v_0, \ldots, v_\ell \rangle) = \min_{1 \le j \le \ell} w(v_{j-1}, v_j)$ for $\ell > 0$

  $\psi_{\min}(\langle v_0 \rangle) = \infty$ if $v_0 \in S$, $\psi_{\min}(\langle v_0 \rangle) = -\infty$ if $v_0 \notin S$
- $\psi_{\text{sum}}(\langle v_0, \ldots, v_\ell \rangle) = -\sum_{1 \le j \le \ell} w(v_{j-1}, v_j)$ for $\ell > 0$

  $\psi_{\text{sum}}(\langle v_0 \rangle) = \infty$ if $v_0 \in S$, $\psi_{\text{sum}}(\langle v_0 \rangle) = -\infty$ if $v_0 \notin S$
- HLOIFT uses DA with $\psi_{\min}$ and oriented w, a problem!

For what path cost ψ DA works properly?

Studied in JMIV paper [Ciesielski, Falcão, Miranda, Sept. 2018]

correcting errors of TPAMI paper [Falcão, Stolfi, Lotufo, 2004].

If w is an edge weight map for undirected graph $G = \langle V, E \rangle$, then DA works properly for:

- FC/IFT: $\psi_{\min}(\langle v_0, \ldots, v_\ell \rangle) = \min_{1 \le j \le \ell} w(v_{j-1}, v_j)$ for $\ell > 0$

  $\psi_{\min}(\langle v_0 \rangle) = \infty$ if $v_0 \in S$, $\psi_{\min}(\langle v_0 \rangle) = -\infty$ if $v_0 \notin S$
- $\psi_{\mathrm{sum}}(\langle v_0, \ldots, v_\ell \rangle) = -\sum_{1 \le j \le \ell} w(v_{j-1}, v_j)$ for $\ell > 0$

  $\psi_{\mathrm{sum}}(\langle v_0 \rangle) = \infty$ if $v_0 \in S$, $\psi_{\mathrm{sum}}(\langle v_0 \rangle) = -\infty$ if $v_0 \notin S$
- HLOIFT uses DA with $\psi_{\min}$ and oriented w, a problem!

For what path cost ψ DA works properly?

Studied in JMIV paper [Ciesielski, Falcão, Miranda, Sept. 2018]

correcting errors of TPAMI paper [Falcão, Stolfi, Lotufo, 2004].

If w is an edge weight map for undirected graph $G = \langle V, E \rangle$, then DA works properly for:

- FC/IFT: $\psi_{\min}(\langle v_0, \ldots, v_\ell \rangle) = \min_{1 \le j \le \ell} w(v_{j-1}, v_j)$ for $\ell > 0$

  $\psi_{\min}(\langle v_0 \rangle) = \infty$ if $v_0 \in S$, $\psi_{\min}(\langle v_0 \rangle) = -\infty$ if $v_0 \notin S$

- $\psi_{\text{sum}}(\langle v_0, \ldots, v_\ell \rangle) = -\sum_{1 \le j \le \ell} w(v_{j-1}, v_j)$ for $\ell > 0$

  $\psi_{\text{sum}}(\langle v_0 \rangle) = \infty$ if $v_0 \in S$, $\psi_{\text{sum}}(\langle v_0 \rangle) = -\infty$ if $v_0 \notin S$

- HLOIFT uses DA with $\psi_{\min}$ and oriented w, a problem!

For what path cost ψ DA works properly?

Studied in JMIV paper [Ciesielski, Falcão, Miranda, Sept. 2018]

correcting errors of TPAMI paper [Falcão, Stolfi, Lotufo, 2004].

If w is an edge weight map for undirected graph $G = \langle V, E \rangle$, then DA works properly for:

- FC/IFT: $\psi_{\min}(\langle v_0, \ldots, v_\ell \rangle) = \min_{1 \le j \le \ell} w(v_{j-1}, v_j)$ for $\ell > 0$

  $\psi_{\min}(\langle v_0 \rangle) = \infty$ if $v_0 \in S$, $\quad \psi_{\min}(\langle v_0 \rangle) = -\infty$ if $v_0 \notin S$
- $\psi_{\text{sum}}(\langle v_0, \ldots, v_\ell \rangle) = -\sum_{1 \le j \le \ell} w(v_{j-1}, v_j)$ for $\ell > 0$

  $\psi_{\text{sum}}(\langle v_0 \rangle) = \infty$ if $v_0 \in S$, $\quad \psi_{\text{sum}}(\langle v_0 \rangle) = -\infty$ if $v_0 \notin S$
- HLOIFT uses DA with $\psi_{\min}$ and oriented w, a problem!

For what path cost ψ DA works properly?

Studied in JMIV paper [Ciesielski, Falcão, Miranda, Sept. 2018]

correcting errors of TPAMI paper [Falcão, Stolfi, Lotufo, 2004].

If w is an edge weight map for undirected graph $G = \langle V, E \rangle$, then DA works properly for:

- FC/IFT: $\psi_{\min}(\langle v_0, \ldots, v_\ell \rangle) = \min_{1 \le j \le \ell} w(v_{j-1}, v_j)$ for $\ell > 0$

  $\psi_{\min}(\langle v_0 \rangle) = \infty$ if $v_0 \in S$, $\psi_{\min}(\langle v_0 \rangle) = -\infty$ if $v_0 \notin S$
- $\psi_{\mathrm{sum}}(\langle v_0, \ldots, v_\ell \rangle) = -\sum_{1 \le j \le \ell} w(v_{j-1}, v_j)$ for $\ell > 0$

  $\psi_{\mathrm{sum}}(\langle v_0 \rangle) = \infty$ if $v_0 \in S$, $\psi_{\mathrm{sum}}(\langle v_0 \rangle) = -\infty$ if $v_0 \notin S$
- HLOIFT uses DA with $\psi_{\min}$ and oriented w, a problem!

For what path cost ψ DA works properly?

Studied in JMIV paper [Ciesielski, Falcão, Miranda, Sept. 2018]

correcting errors of TPAMI paper [Falcão, Stolfi, Lotufo, 2004].

If w is an edge weight map for undirected graph $G = \langle V, E \rangle$, then DA works properly for:

- FC/IFT: $\psi_{\min}(\langle v_0, \ldots, v_\ell \rangle) = \min_{1 \le j \le \ell} w(v_{j-1}, v_j)$ for $\ell > 0$
  $\psi_{\min}(\langle v_0 \rangle) = \infty$ if $v_0 \in S$, $\psi_{\min}(\langle v_0 \rangle) = -\infty$ if $v_0 \notin S$
- $\psi_{\text{sum}}(\langle v_0, \ldots, v_\ell \rangle) = -\sum_{1 \le j \le \ell} w(v_{j-1}, v_j)$ for $\ell > 0$
  $\psi_{\text{sum}}(\langle v_0 \rangle) = \infty$ if $v_0 \in S$, $\psi_{\text{sum}}(\langle v_0 \rangle) = -\infty$ if $v_0 \notin S$
- HLOIFT uses DA with $\psi_{\min}$ and oriented w, a problem!

For what path cost ψ DA works properly?

Studied in JMIV paper [Ciesielski, Falcão, Miranda, Sept. 2018]

correcting errors of TPAMI paper [Falcão, Stolfi, Lotufo, 2004].

If w is an edge weight map for undirected graph $G = \langle V, E \rangle$, then DA works properly for:

- FC/IFT: $\psi_{\min}(\langle v_0, \ldots, v_\ell \rangle) = \min_{1 \le j \le \ell} w(v_{j-1}, v_j)$ for $\ell > 0$

  $\psi_{\min}(\langle v_0 \rangle) = \infty$ if $v_0 \in S$, $\psi_{\min}(\langle v_0 \rangle) = -\infty$ if $v_0 \notin S$
- $\psi_{\mathrm{sum}}(\langle v_0, \ldots, v_\ell \rangle) = -\sum_{1 \le j \le \ell} w(v_{j-1}, v_j)$ for $\ell > 0$

  $\psi_{\mathrm{sum}}(\langle v_0 \rangle) = \infty$ if $v_0 \in S$, $\psi_{\mathrm{sum}}(\langle v_0 \rangle) = -\infty$ if $v_0 \notin S$
- HLOIFT uses DA with $\psi_{\min}$ and oriented w, a problem!

For what path cost ψ DA works properly?

Studied in JMIV paper [Ciesielski, Falcão, Miranda, Sept. 2018]

correcting errors of TPAMI paper [Falcão, Stolfi, Lotufo, 2004].

If w is an edge weight map for undirected graph $G = \langle V, E\rangle$, then DA works properly for:

- FC/IFT: $\psi_{\min}(\langle v_0, \ldots, v_\ell\rangle) = \min_{1\le j\le \ell} w(v_{j-1}, v_j)$ for $\ell > 0$

  $\psi_{\min}(\langle v_0\rangle) = \infty$ if $v_0 \in S$, $\quad \psi_{\min}(\langle v_0\rangle) = -\infty$ if $v_0 \notin S$
- $\psi_{\text{sum}}(\langle v_0, \ldots, v_\ell\rangle) = -\sum_{1\le j\le \ell} w(v_{j-1}, v_j)$ for $\ell > 0$

  $\psi_{\text{sum}}(\langle v_0\rangle) = \infty$ if $v_0 \in S$, $\quad \psi_{\text{sum}}(\langle v_0\rangle) = -\infty$ if $v_0 \notin S$
- HLOIFT uses DA with $\psi_{\min}$ and oriented w, a problem!

DA with oriented variant of ψ_{min}

In JMIV paper [Ciesielski, Herman, Kong, 2016]

we studied DA with ith object O_i having its oriented weights w_i and

$\psi^*_{\text{min}}(\langle v_0, \ldots, v_\ell \rangle) = \min_{1 \le j \le \ell} w_i(v_{j-1}, v_j)$ with v_0 a seed of O_i.

Theorem (Ciesielski, Herman, Kong, 2016)

For ψ^*_{min} as above

- *The output of DA is completely robust under (unaffected by) small (within CORE sets) seed changes.*
- *The output of DA has a nice characterization in terms of path strength competition.*

However, for ψ^*_{min}, the forest returned by DA need not be optimal. Also, in general, no minimality of a cut for ψ^*_{min}.

DA with oriented variant of $\psi_{\min}$

In JMIV paper [Ciesielski, Herman, Kong, 2016]

we studied DA with ith object O_i having its oriented weights w_i and

$\psi^*_{\min}(\langle v_0, \ldots, v_\ell \rangle) = \min_{1 \le j \le \ell} w_i(v_{j-1}, v_j)$ with v_0 a seed of O_i.

Theorem (Ciesielski, Herman, Kong, 2016)

For $\psi^*_{\min}$ as above

- *The output of DA is completely robust under (unaffected by) small (within CORE sets) seed changes.*
- *The output of DA has a nice characterization in terms of path strength competition.*

However, for $\psi^*_{\min}$, the forest returned by DA need not be optimal. Also, in general, no minimality of a cut for $\psi^*_{\min}$.

DA with oriented variant of ψ_{min}

In JMIV paper [Ciesielski, Herman, Kong, 2016]

we studied DA with *i*th object O_i having its oriented weights w_i and

$\psi^*_{\text{min}}(\langle v_0, \ldots, v_\ell \rangle) = \min_{1 \le j \le \ell} w_i(v_{j-1}, v_j)$ with v_0 a seed of O_i.

Theorem (Ciesielski, Herman, Kong, 2016)

For ψ^*_{min} as above

- *The output of DA is completely robust under (unaffected by) small (within CORE sets) seed changes.*
- *The output of DA has a nice characterization in terms of path strength competition.*

However, for ψ^*_{min}, the forest returned by DA need not be optimal. Also, in general, no minimality of a cut for ψ^*_{min}.

DA with oriented variant of $\psi_{\min}$

In JMIV paper [Ciesielski, Herman, Kong, 2016]

we studied DA with *i*th object O_i having its oriented weights w_i and

$\psi^*_{\min}(\langle v_0, \ldots, v_\ell \rangle) = \min_{1 \le j \le \ell} w_i(v_{j-1}, v_j)$ with v_0 a seed of O_i.

Theorem (Ciesielski, Herman, Kong, 2016)

For $\psi^*_{\min}$ as above

- *The output of DA is completely robust under (unaffected by) small (within CORE sets) seed changes.*
- *The output of DA has a nice characterization in terms of path strength competition.*

However, for $\psi^*_{\min}$, the forest returned by DA need not be optimal. Also, in general, no minimality of a cut for $\psi^*_{\min}$.

DA with oriented variant of $\psi_{\min}$

In JMIV paper [Ciesielski, Herman, Kong, 2016]

we studied DA with ith object O_i having its oriented weights w_i and

$\psi^*_{\min}(\langle v_0, \ldots, v_\ell \rangle) = \min_{1 \le j \le \ell} w_i(v_{j-1}, v_j)$ with v_0 a seed of O_i.

Theorem (Ciesielski, Herman, Kong, 2016)

For $\psi^*_{\min}$ as above

- *The output of DA is completely robust under (unaffected by) small (within CORE sets) seed changes.*
- *The output of DA has a nice characterization in terms of path strength competition.*

However, for $\psi^*_{\min}$, the forest returned by DA need not be optimal. Also, in general, no minimality of a cut for $\psi^*_{\min}$.

DA with oriented variant of $\psi_{\min}$

In JMIV paper [Ciesielski, Herman, Kong, 2016]

we studied DA with *i*th object O_i having its oriented weights w_i and

$\psi^*_{\min}(\langle v_0, \ldots, v_\ell \rangle) = \min_{1 \le j \le \ell} w_i(v_{j-1}, v_j)$ with v_0 a seed of O_i.

Theorem (Ciesielski, Herman, Kong, 2016)

For $\psi^*_{\min}$ as above

- *The output of DA is completely robust under (unaffected by) small (within CORE sets) seed changes.*
- *The output of DA has a nice characterization in terms of path strength competition.*

However, for $\psi^*_{\min}$, the forest returned by DA need not be optimal. Also, in general, no minimality of a cut for $\psi^*_{\min}$.

DA with oriented variant of $\psi_{\min}$

In JMIV paper [Ciesielski, Herman, Kong, 2016]

we studied DA with *i*th object O_i having its oriented weights w_i and

$\psi^*_{\min}(\langle v_0, \ldots, v_\ell \rangle) = \min_{1 \le j \le \ell} w_i(v_{j-1}, v_j)$ with v_0 a seed of O_i.

Theorem (Ciesielski, Herman, Kong, 2016)

For $\psi^*_{\min}$ as above

- *The output of DA is completely robust under (unaffected by) small (within CORE sets) seed changes.*
- *The output of DA has a nice characterization in terms of path strength competition.*

However, for $\psi^*_{\min}$, the forest returned by DA need not be optimal. Also, in general, no minimality of a cut for $\psi^*_{\min}$.

DA with oriented variant of $\psi_{\min}$

In JMIV paper [Ciesielski, Herman, Kong, 2016]

we studied DA with *i*th object O_i having its oriented weights w_i and

$\psi^*_{\min}(\langle v_0, \ldots, v_\ell \rangle) = \min_{1 \le j \le \ell} w_i(v_{j-1}, v_j)$ with v_0 a seed of O_i.

Theorem (Ciesielski, Herman, Kong, 2016)

For $\psi^*_{\min}$ as above

- *The output of DA is completely robust under (unaffected by) small (within CORE sets) seed changes.*
- *The output of DA has a nice characterization in terms of path strength competition.*

However, for $\psi^*_{\min}$, the forest returned by DA need not be optimal. Also, in general, no minimality of a cut for $\psi^*_{\min}$.

Outline

$\psi^*_{\min}$ for which DA returns delineation with optimal cut

Let $\psi^\star_{\min}$ denotes $\psi^*_{\min}$ in object/background setting such that

$w_1(c,d) = w_0(d,c)$ for all $\langle c,d\rangle \in E$.

Theorem (preliminary; & Leon, Ciesielski, Miranda, submitted)

If object O is an output of DA run with $\psi^\star_{\min}$, then the graph cut

$$c(O) = \{\langle c,d\rangle \in E\colon c \in O \;\&\; d \notin O\}$$

minimizes the ℓ_∞ norm $\varepsilon_\infty(O) \stackrel{\text{def}}{=} \max_{\langle c,d\rangle \in c(O)} w_1(c,d)$ among all objects satisfying the constrains.

Assumption $w_1(c,d) = w_0(d,c)$ is needed to ensure that incorporating $\langle c,d\rangle$ in a path from either object or background influences the path strength the same way.

$\psi^*_{\min}$ for which DA returns delineation with optimal cut

Let $\psi^\star_{\min}$ denotes $\psi^*_{\min}$ in object/background setting such that

$w_1(c,d) = w_0(d,c)$ for all $\langle c,d\rangle \in E$.

Theorem (preliminary; & Leon, Ciesielski, Miranda, submitted)

If object O is an output of DA run with $\psi^\star_{\min}$, then the graph cut

$$c(O) = \{\langle c,d\rangle \in E \colon c \in O \ \&\ d \notin O\}$$

minimizes the ℓ_∞ norm $\varepsilon_\infty(O) \stackrel{\text{def}}{=} \max_{\langle c,d\rangle \in c(O)} w_1(c,d)$ among all objects satisfying the constrains.

Assumption $w_1(c,d) = w_0(d,c)$ is needed to ensure that incorporating $\langle c,d\rangle$ in a path from either object or background influences the path strength the same way.

$\psi^*_{\min}$ for which DA returns delineation with optimal cut

Let $\psi^\star_{\min}$ denotes $\psi^*_{\min}$ in object/background setting such that

$w_1(c,d) = w_0(d,c)$ for all $\langle c,d\rangle \in E$.

Theorem (preliminary; & Leon, Ciesielski, Miranda, submitted)

If object O is an output of DA run with $\psi^\star_{\min}$, then the graph cut

$$c(O) = \{\langle c,d\rangle \in E \colon c \in O \;\&\; d \notin O\}$$

minimizes the ℓ_∞ norm $\varepsilon_\infty(O) \stackrel{\text{def}}{=} \max_{\langle c,d\rangle \in c(O)} w_1(c,d)$ among all objects satisfying the constrains.

Assumption $w_1(c,d) = w_0(d,c)$ is needed to ensure that incorporating $\langle c,d\rangle$ in a path from either object or background influences the path strength the same way.

$\psi^*_{\min}$ for which DA returns delineation with optimal cut

Let $\psi^\star_{\min}$ denotes $\psi^*_{\min}$ in object/background setting such that

$w_1(c,d) = w_0(d,c)$ for all $\langle c,d \rangle \in E$.

Theorem (preliminary; & Leon, Ciesielski, Miranda, submitted)

If object O is an output of DA run with $\psi^\star_{\min}$, then the graph cut

$$c(O) = \{\langle c,d \rangle \in E \colon c \in O \;\&\; d \notin O\}$$

minimizes the ℓ_∞ norm $\varepsilon_\infty(O) \stackrel{\text{def}}{=} \max_{\langle c,d \rangle \in c(O)} w_1(c,d)$ among all objects satisfying the constrains.

Assumption $w_1(c,d) = w_0(d,c)$ is needed to ensure that incorporating $\langle c,d \rangle$ in a path from either object or background influences the path strength the same way.

$\psi^*_{\min}$ for which DA returns delineation with optimal cut

Let $\psi^\star_{\min}$ denotes $\psi^*_{\min}$ in object/background setting such that

$w_1(c,d) = w_0(d,c)$ for all $\langle c,d\rangle \in E$.

Theorem (preliminary; & Leon, Ciesielski, Miranda, submitted)

If object O is an output of DA run with $\psi^\star_{\min}$, then the graph cut

$$c(O) = \{\langle c,d\rangle \in E \colon c \in O \;\&\; d \notin O\}$$

minimizes the ℓ_∞ norm $\varepsilon_\infty(O) \stackrel{\text{def}}{=} \max_{\langle c,d\rangle \in c(O)} w_1(c,d)$ among all objects satisfying the constrains.

Assumption $w_1(c,d) = w_0(d,c)$ is needed to ensure that incorporating $\langle c,d\rangle$ in a path from either object or background influences the path strength the same way.

Oriented Image Foresting Transform algorithm OIFT

Is OIFT a DA run with $\psi^*_{\min}$? Close, but formally not.

Assume that $w_1(c,d) = w_0(d,c)$ for all $\langle c,d\rangle \in E$ and let

$\psi_{\text{last}}(\langle v_0,\ldots,v_\ell\rangle) = w_i(v_{\ell-1},v_\ell)$ when $\ell > 0$ and v_0 a seed of O_i.

$\psi_{\text{last}}(\langle v_0\rangle) = \infty$ when v_0 a seed and $\psi_{\text{last}}(\langle v_0\rangle) = -\infty$ otherwise.

Definition

OIFT is a DA run with ψ_{last} as above.

Theorem (preliminary result: OIFT as DA with $\psi^*_{\min}$)

Any output of OIFT is an output of a particular implementation of DA with $\psi^*_{\min}$.
Thus, a graph cut of any object returned by OIFT minimizes the ℓ_∞ norm among all objects satisfying the constrains.

Oriented Image Foresting Transform algorithm OIFT

Is OIFT a DA run with $\psi^*_{\min}$? Close, but formally not.

Assume that $w_1(c,d) = w_0(d,c)$ for all $\langle c,d\rangle \in E$ and let

$\psi_{\text{last}}(\langle v_0,\ldots,v_\ell\rangle) = w_i(v_{\ell-1},v_\ell)$ when $\ell > 0$ and v_0 a seed of O_i.

$\psi_{\text{last}}(\langle v_0\rangle) = \infty$ when v_0 a seed and $\psi_{\text{last}}(\langle v_0\rangle) = -\infty$ otherwise.

Definition

OIFT is a DA run with ψ_{last} as above.

Theorem (preliminary result: OIFT as DA with $\psi^*_{\min}$)

Any output of OIFT is an output of a particular implementation of DA with $\psi^*_{\min}$.
Thus, a graph cut of any object returned by OIFT minimizes the ℓ_∞ norm among all objects satisfying the constrains.

Oriented Image Foresting Transform algorithm OIFT

Is OIFT a DA run with ψ^*_{min}? Close, but formally not.

Assume that $w_1(c,d) = w_0(d,c)$ for all $\langle c,d\rangle \in E$ and let

$\psi_{\text{last}}(\langle v_0,\ldots,v_\ell\rangle) = w_i(v_{\ell-1},v_\ell)$ when $\ell > 0$ and v_0 a seed of O_i.

$\psi_{\text{last}}(\langle v_0\rangle) = \infty$ when v_0 a seed and $\psi_{\text{last}}(\langle v_0\rangle) = -\infty$ otherwise.

Definition

OIFT is a DA run with ψ_{last} as above.

Theorem (preliminary result: OIFT as DA with ψ^*_{min})

Any output of OIFT is an output of a particular implementation of DA with ψ^*_{min}.
Thus, a graph cut of any object returned by OIFT minimizes the ℓ_∞ norm among all objects satisfying the constrains.

Oriented Image Foresting Transform algorithm OIFT

Is OIFT a DA run with ψ^*_{min}? Close, but formally not.

Assume that $w_1(c,d) = w_0(d,c)$ for all $\langle c,d\rangle \in E$ and let

$\psi_{\text{last}}(\langle v_0,\ldots,v_\ell\rangle) = w_i(v_{\ell-1},v_\ell)$ when $\ell > 0$ and v_0 a seed of O_i.

$\psi_{\text{last}}(\langle v_0\rangle) = \infty$ when v_0 a seed and $\psi_{\text{last}}(\langle v_0\rangle) = -\infty$ otherwise.

Definition

OIFT is a DA run with ψ_{last} as above.

Theorem (preliminary result: OIFT as DA with ψ^*_{min})

Any output of OIFT is an output of a particular implementation of DA with ψ^*_{min}.
Thus, a graph cut of any object returned by OIFT minimizes the ℓ_∞ norm among all objects satisfying the constrains.

Oriented Image Foresting Transform algorithm OIFT

Is OIFT a DA run with ψ^*_{min}? Close, but formally not.

Assume that $w_1(c,d) = w_0(d,c)$ for all $\langle c,d\rangle \in E$ and let

$\psi_{\text{last}}(\langle v_0,\ldots,v_\ell\rangle) = w_i(v_{\ell-1},v_\ell)$ when $\ell > 0$ and v_0 a seed of O_i.

$\psi_{\text{last}}(\langle v_0\rangle) = \infty$ when v_0 a seed and $\psi_{\text{last}}(\langle v_0\rangle) = -\infty$ otherwise.

Definition

OIFT is a DA run with ψ_{last} as above.

Theorem (preliminary result: OIFT as DA with ψ^*_{min})

Any output of OIFT is an output of a particular implementation of DA with ψ^*_{min}.
Thus, a graph cut of any object returned by OIFT minimizes the ℓ_∞ norm among all objects satisfying the constrains.

Oriented Image Foresting Transform algorithm OIFT

Is OIFT a DA run with ψ^*_{min}? Close, but formally not.

Assume that $w_1(c,d) = w_0(d,c)$ for all $\langle c,d\rangle \in E$ and let

$\psi_{\text{last}}(\langle v_0,\ldots,v_\ell\rangle) = w_i(v_{\ell-1},v_\ell)$ when $\ell > 0$ and v_0 a seed of O_i.

$\psi_{\text{last}}(\langle v_0\rangle) = \infty$ when v_0 a seed and $\psi_{\text{last}}(\langle v_0\rangle) = -\infty$ otherwise.

Definition

OIFT is a DA run with ψ_{last} as above.

Theorem (preliminary result: OIFT as DA with ψ^*_{min})

Any output of OIFT is an output of a particular implementation of DA with ψ^*_{min}.

Thus, a graph cut of any object returned by OIFT minimizes the ℓ_∞ norm among all objects satisfying the constrains.

Oriented Image Foresting Transform algorithm OIFT

Is OIFT a DA run with $\psi^*_{\min}$? Close, but formally not.

Assume that $w_1(c,d) = w_0(d,c)$ for all $\langle c,d\rangle \in E$ and let

$\psi_{\mathrm{last}}(\langle v_0,\ldots,v_\ell\rangle) = w_i(v_{\ell-1},v_\ell)$ when $\ell > 0$ and v_0 a seed of O_i.

$\psi_{\mathrm{last}}(\langle v_0\rangle) = \infty$ when v_0 a seed and $\psi_{\mathrm{last}}(\langle v_0\rangle) = -\infty$ otherwise.

Definition

OIFT is a DA run with ψ_{last} as above.

Theorem (preliminary result: OIFT as DA with $\psi^*_{\min}$)

Any output of OIFT is an output of a particular implementation of DA with $\psi^*_{\min}$.
Thus, a graph cut of any object returned by OIFT minimizes the ℓ_∞ norm among all objects satisfying the constrains.

Some properties of OIFT

- Can incorporate image brightness increase/decrease in weight function. If we like to favor transitions from bright to dark pixels when passing from object to the background, we can define, for some $\alpha \in (0, 1)$,

$$w_1(c, d) = \begin{cases} (1 - \alpha)e^{-\|f(c)-f(d)\|} & \text{if } \|f(c)\| > \|f(d)\| \\ (1 + \alpha)e^{-\|f(c)-f(d)\|} & \text{otherwise.} \end{cases}$$

- Can incorporate shape constraints like geodesic star convexity [Mansilla, Jackowski, Miranda, 2013], geodesic band constraints [Braz, Miranda, 2014], Hedgehog Shape Prior, and other to be explored.

Some properties of OIFT

- Can incorporate image brightness increase/decrease in weight function. If we like to favor transitions from bright to dark pixels when passing from object to the background, we can define, for some $\alpha \in (0, 1)$,

$$w_1(c,d) = \begin{cases} (1-\alpha)e^{-\|f(c)-f(d)\|} & \text{if } \|f(c)\| > \|f(d)\| \\ (1+\alpha)e^{-\|f(c)-f(d)\|} & \text{otherwise.} \end{cases}$$

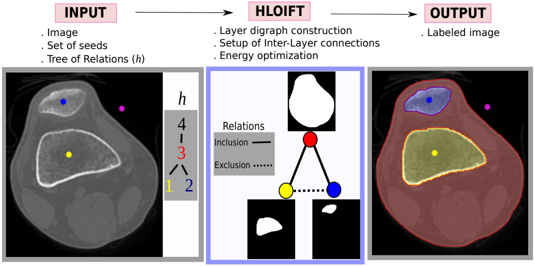
- Can incorporate shape constraints like geodesic star convexity [Mansilla, Jackowski, Miranda, 2013], geodesic band constraints [Braz, Miranda, 2014], Hedgehog Shape Prior, and other to be explored.

Outline

HLOIFT: input and output

HLOIFT is, essentially, OIFT algorithm run on a modified graph.

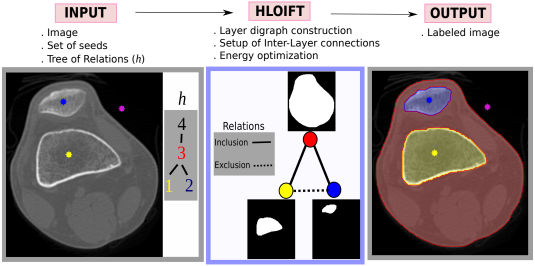
Input: Image, a tree representing inclusion/exclusion relations between the objects we seek, seeds representing the objects; $\rho \geq 0$ giving minimal distance between boundaries of objects.



HLOIFT: input and output

HLOIFT is, essentially, OIFT algorithm run on a modified graph.

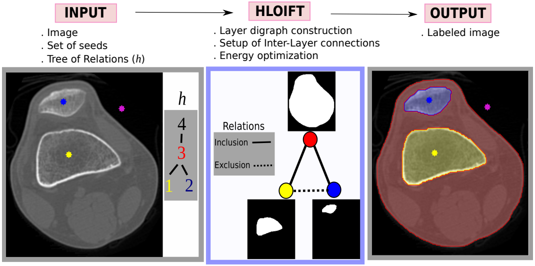
Input: Image, a tree representing inclusion/exclusion relations between the objects we seek, seeds representing the objects; $\rho \geq 0$ giving minimal distance between boundaries of objects.



HLOIFT: input and output

HLOIFT is, essentially, OIFT algorithm run on a modified graph.

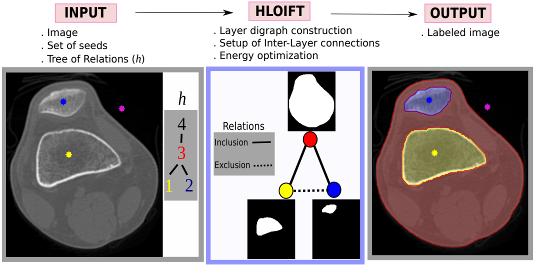
Input: Image, a tree representing inclusion/exclusion relations between the objects we seek, seeds representing the objects; $\rho \geq 0$ giving minimal distance between boundaries of objects.



HLOIFT: input and output

HLOIFT is, essentially, OIFT algorithm run on a modified graph.

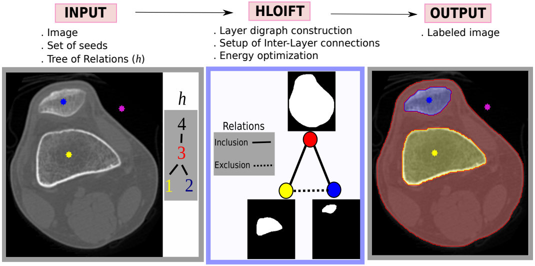
Input: Image, a tree representing inclusion/exclusion relations between the objects we seek, seeds representing the objects; $\rho \geq 0$ giving minimal distance between boundaries of objects.



HLOIFT: input and output

HLOIFT is, essentially, OIFT algorithm run on a modified graph.

Input: Image, a tree representing inclusion/exclusion relations between the objects we seek, seeds representing the objects; $\rho \geq 0$ giving minimal distance between boundaries of objects.



Forming HLOIFT's graph

Let $f\colon V \to \mathbb{R}^k$ be an (n-dimensional) image containing objects $O_1, \ldots, O_m, O_{m+1} = V$. A hierarchy tree is indicated by a parent map h, with $h(i) = j$ meaning that O_j is a parent of O_i.

For every $i \in \mathcal{L} = \{1, \ldots, m\}$ let $\langle V, E_i, w_i \rangle$ be an edge weighted graph associated with image f and object O_i. The edges and weights can include other constrains, like shape.

HLOIFT weighted digraph is defined as $\langle \mathcal{L} \times V, E, w \rangle$, where its restriction to ith object layer, $\langle \{i\} \times V, E^i, w^i \rangle$, is an isomorphic copy of $\langle V, E_i, w_i \rangle$.

We still need to define inter-layer edges and their weights on the HLOIFT graph $\mathcal{N} = \mathcal{L} \times V$.

Let $p\colon \mathcal{N} \to V$ be a projection, $p(i, c) = c$.

Forming HLOIFT's graph

Let $f\colon V \to \mathbb{R}^k$ be an (n-dimensional) image containing objects $O_1, \ldots, O_m, O_{m+1} = V$. A hierarchy tree is indicated by a parent map h, with $h(i) = j$ meaning that O_j is a parent of O_i.

For every $i \in \mathcal{L} = \{1, \ldots, m\}$ let $\langle V, E_i, w_i \rangle$ be an edge weighted graph associated with image f and object O_i. The edges and weights can include other constrains, like shape.

HLOIFT weighted digraph is defined as $\langle \mathcal{L} \times V, E, w \rangle$, where its restriction to ith object layer, $\langle \{i\} \times V, E^i, w^i \rangle$, is an isomorphic copy of $\langle V, E_i, w_i \rangle$.

We still need to define inter-layer edges and their weights on the HLOIFT graph $\mathcal{N} = \mathcal{L} \times V$.

Let $p\colon \mathcal{N} \to V$ be a projection, $p(i, c) = c$.

Forming HLOIFT's graph

Let $f\colon V \to \mathbb{R}^k$ be an (n-dimensional) image containing objects $O_1, \ldots, O_m, O_{m+1} = V$. A hierarchy tree is indicated by a parent map h, with $h(i) = j$ meaning that O_j is a parent of O_i.

For every $i \in \mathcal{L} = \{1, \ldots, m\}$ let $\langle V, E_i, w_i \rangle$ be an edge weighted graph associated with image f and object O_i. The edges and weights can include other constrains, like shape.

HLOIFT weighted digraph is defined as $\langle \mathcal{L} \times V, E, w \rangle$, where its restriction to ith object layer, $\langle \{i\} \times V, E^i, w^i \rangle$, is an isomorphic copy of $\langle V, E_i, w_i \rangle$.

We still need to define inter-layer edges and their weights on the HLOIFT graph $\mathcal{N} = \mathcal{L} \times V$.

Let $p\colon \mathcal{N} \to V$ be a projection, $p(i, c) = c$.

Forming HLOIFT's graph

Let $f\colon V \to \mathbb{R}^k$ be an (n-dimensional) image containing objects $O_1, \ldots, O_m, O_{m+1} = V$. A hierarchy tree is indicated by a parent map h, with $h(i) = j$ meaning that O_j is a parent of O_i.

For every $i \in \mathcal{L} = \{1, \ldots, m\}$ let $\langle V, E_i, w_i \rangle$ be an edge weighted graph associated with image f and object O_i. The edges and weights can include other constrains, like shape.

HLOIFT weighted digraph is defined as $\langle \mathcal{L} \times V, E, w \rangle$, where its restriction to ith object layer, $\langle \{i\} \times V, E^i, w^i \rangle$, is an isomorphic copy of $\langle V, E_i, w_i \rangle$.

We still need to define inter-layer edges and their weights on the HLOIFT graph $\mathcal{N} = \mathcal{L} \times V$.

Let $p\colon \mathcal{N} \to V$ be a projection, $p(i, c) = c$.

Forming HLOIFT's graph

Let $f\colon V \to \mathbb{R}^k$ be an (n-dimensional) image containing objects $O_1, \ldots, O_m, O_{m+1} = V$. A hierarchy tree is indicated by a parent map h, with $h(i) = j$ meaning that O_j is a parent of O_i.

For every $i \in \mathcal{L} = \{1, \ldots, m\}$ let $\langle V, E_i, w_i \rangle$ be an edge weighted graph associated with image f and object O_i. The edges and weights can include other constrains, like shape.

HLOIFT weighted digraph is defined as $\langle \mathcal{L} \times V, E, w \rangle$, where its restriction to ith object layer, $\langle \{i\} \times V, E^i, w^i \rangle$, is an isomorphic copy of $\langle V, E_i, w_i \rangle$.

We still need to define inter-layer edges and their weights on the HLOIFT graph $\mathcal{N} = \mathcal{L} \times V$.

Let $p\colon \mathcal{N} \to V$ be a projection, $p(i, c) = c$.

Forming HLOIFT's graph

Let $f \colon V \to \mathbb{R}^k$ be an (n-dimensional) image containing objects $O_1, \ldots, O_m, O_{m+1} = V$. A hierarchy tree is indicated by a parent map h, with $h(i) = j$ meaning that O_j is a parent of O_i.

For every $i \in \mathcal{L} = \{1, \ldots, m\}$ let $\langle V, E_i, w_i \rangle$ be an edge weighted graph associated with image f and object O_i. The edges and weights can include other constrains, like shape.

HLOIFT weighted digraph is defined as $\langle \mathcal{L} \times V, E, w \rangle$, where its restriction to ith object layer, $\langle \{i\} \times V, E^i, w^i \rangle$, is an isomorphic copy of $\langle V, E_i, w_i \rangle$.

We still need to define inter-layer edges and their weights on the HLOIFT graph $\mathcal{N} = \mathcal{L} \times V$.

Let $p \colon \mathcal{N} \to V$ be a projection, $p(i, c) = c$.

Forming HLOIFT's graph

Let $f\colon V \to \mathbb{R}^k$ be an (n-dimensional) image containing objects $O_1, \ldots, O_m, O_{m+1} = V$. A hierarchy tree is indicated by a parent map h, with $h(i) = j$ meaning that O_j is a parent of O_i.

For every $i \in \mathcal{L} = \{1, \ldots, m\}$ let $\langle V, E_i, w_i \rangle$ be an edge weighted graph associated with image f and object O_i. The edges and weights can include other constrains, like shape.

HLOIFT weighted digraph is defined as $\langle \mathcal{L} \times V, E, w \rangle$, where its restriction to ith object layer, $\langle \{i\} \times V, E^i, w^i \rangle$, is an isomorphic copy of $\langle V, E_i, w_i \rangle$.

We still need to define inter-layer edges and their weights on the HLOIFT graph $\mathcal{N} = \mathcal{L} \times V$.

Let $p\colon \mathcal{N} \to V$ be a projection, $p(i, c) = c$.

Labeling of objects

HLOIFT, being essentially OIFT run on $\mathcal{N}$, returns a single object $O \subset \mathcal{N}$.

It encodes the objects and the background as

$$O_i = \{t \in V : (i,t) \in O\} = p[O \cap (\{i\} \times V)] \ \& \ O_0 = V \setminus \bigcup_{i \in \mathcal{L}} O_i.$$

This indicates how to define inter-layer edges and their weights to ensure tree-indicated relations.

If seed sets $\langle \mathcal{S}_0, \ldots, \mathcal{S}_m \rangle$ in V indicate objects $\langle O_0, \ldots, O_m \rangle$, then $\bar{\mathcal{S}}_1 = \bigcup_{i \in \mathcal{L}} \{i\} \times \mathcal{S}_i$ indicates object O in $\mathcal{N}$, while $\bar{\mathcal{S}}_0 = \mathcal{L} \times \mathcal{S}_0$ indicatess its complement in $\mathcal{N}$.

Sets $\bar{\mathcal{S}}_0$ and $\bar{\mathcal{S}}_1$ are used to define ψ_{last} in $\mathcal{N}$.

Labeling of objects

HLOIFT, being essentially OIFT run on $\mathcal{N}$, returns a single object $O \subset \mathcal{N}$.

It encodes the objects and the background as

$$O_i = \{t \in V \colon (i,t) \in O\} = p[O \cap (\{i\} \times V)] \text{ \& } O_0 = V \setminus \bigcup_{i \in \mathcal{L}} O_i.$$

This indicates how to define inter-layer edges and their weights to ensure tree-indicated relations.

If seed sets $\langle \mathcal{S}_0, \ldots, \mathcal{S}_m \rangle$ in V indicate objects $\langle O_0, \ldots, O_m \rangle$, then $\bar{\mathcal{S}}_1 = \bigcup_{i \in \mathcal{L}} \{i\} \times \mathcal{S}_i$ indicates object O in $\mathcal{N}$, while $\bar{\mathcal{S}}_0 = \mathcal{L} \times \mathcal{S}_0$ indicatess its complement in $\mathcal{N}$.

Sets $\bar{\mathcal{S}}_0$ and $\bar{\mathcal{S}}_1$ are used to define ψ_{last} in $\mathcal{N}$.

Labeling of objects

HLOIFT, being essentially OIFT run on $\mathcal{N}$, returns a single object $O \subset \mathcal{N}$.

It encodes the objects and the background as

$$O_i = \{t \in V \colon (i,t) \in O\} = p[O \cap (\{i\} \times V)] \;\&\; O_0 = V \setminus \bigcup_{i \in \mathcal{L}} O_i.$$

This indicates how to define inter-layer edges and their weights to ensure tree-indicated relations.

If seed sets $\langle \mathcal{S}_0, \ldots, \mathcal{S}_m \rangle$ in V indicate objects $\langle O_0, \ldots, O_m \rangle$, then $\bar{\mathcal{S}}_1 = \bigcup_{i \in \mathcal{L}} \{i\} \times \mathcal{S}_i$ indicates object O in $\mathcal{N}$, while $\bar{\mathcal{S}}_0 = \mathcal{L} \times \mathcal{S}_0$ indicatess its complement in $\mathcal{N}$.

Sets $\bar{\mathcal{S}}_0$ and $\bar{\mathcal{S}}_1$ are used to define ψ_{last} in $\mathcal{N}$.

Labeling of objects

HLOIFT, being essentially OIFT run on $\mathcal{N}$, returns a single object $O \subset \mathcal{N}$.

It encodes the objects and the background as

$$O_i = \{t \in V\colon (i,t) \in O\} = p[O \cap (\{i\} \times V)] \;\&\; O_0 = V \setminus \bigcup_{i \in \mathcal{L}} O_i.$$

This indicates how to define inter-layer edges and their weights to ensure tree-indicated relations.

If seed sets $\langle \mathcal{S}_0, \ldots, \mathcal{S}_m \rangle$ in V indicate objects $\langle O_0, \ldots, O_m \rangle$, then $\bar{\mathcal{S}}_1 = \bigcup_{i \in \mathcal{L}} \{i\} \times \mathcal{S}_i$ indicates object O in $\mathcal{N}$, while $\bar{\mathcal{S}}_0 = \mathcal{L} \times \mathcal{S}_0$ indicatess its complement in $\mathcal{N}$.

Sets $\bar{\mathcal{S}}_0$ and $\bar{\mathcal{S}}_1$ are used to define ψ_{last} in $\mathcal{N}$.

Labeling of objects

HLOIFT, being essentially OIFT run on $\mathcal{N}$, returns a single object $O \subset \mathcal{N}$.

It encodes the objects and the background as

$$O_i = \{t \in V \colon (i,t) \in O\} = p[O \cap (\{i\} \times V)] \ \& \ O_0 = V \setminus \bigcup_{i \in \mathcal{L}} O_i.$$

This indicates how to define inter-layer edges and their weights to ensure tree-indicated relations.

If seed sets $\langle \mathcal{S}_0, \ldots, \mathcal{S}_m \rangle$ in V indicate objects $\langle O_0, \ldots, O_m \rangle$, then $\bar{\mathcal{S}}_1 = \bigcup_{i \in \mathcal{L}} \{i\} \times \mathcal{S}_i$ indicates object O in $\mathcal{N}$, while $\bar{\mathcal{S}}_0 = \mathcal{L} \times \mathcal{S}_0$ indicatess its complement in $\mathcal{N}$.

Sets $\bar{\mathcal{S}}_0$ and $\bar{\mathcal{S}}_1$ are used to define ψ_{last} in $\mathcal{N}$.

Labeling of objects

HLOIFT, being essentially OIFT run on $\mathcal{N}$, returns a single object $O \subset \mathcal{N}$.

It encodes the objects and the background as

$$O_i = \{t \in V\colon (i,t) \in O\} = p[O \cap (\{i\} \times V)] \ \& \ O_0 = V \setminus \bigcup_{i \in \mathcal{L}} O_i.$$

This indicates how to define inter-layer edges and their weights to ensure tree-indicated relations.

If seed sets $\langle \mathcal{S}_0, \ldots, \mathcal{S}_m \rangle$ in V indicate objects $\langle O_0, \ldots, O_m \rangle$, then $\bar{\mathcal{S}}_1 = \bigcup_{i \in \mathcal{L}} \{i\} \times \mathcal{S}_i$ indicates object O in $\mathcal{N}$, while $\bar{\mathcal{S}}_0 = \mathcal{L} \times \mathcal{S}_0$ indicatess its complement in $\mathcal{N}$.

Sets $\bar{\mathcal{S}}_0$ and $\bar{\mathcal{S}}_1$ are used to define ψ_{last} in $\mathcal{N}$.

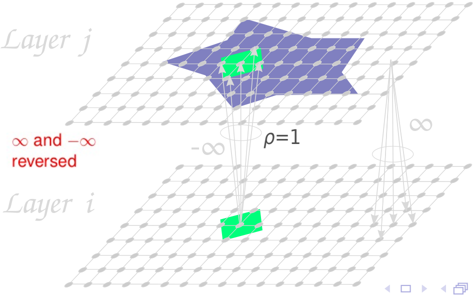
Inter-layer edges indicating inclusions

If O_j is the parent of O_i (i.e., $h(i) = j$),

we add all edges $\langle (i,c),(j,d) \rangle$ with $\|c - d\| \leq \rho$.

For $s = (i,c)$ and $t = (j,d)$ we define

$w_1(s,t) = w_0(t,s) = \infty$ and $w_0(s,t) = w_1(t,s) = -\infty$.



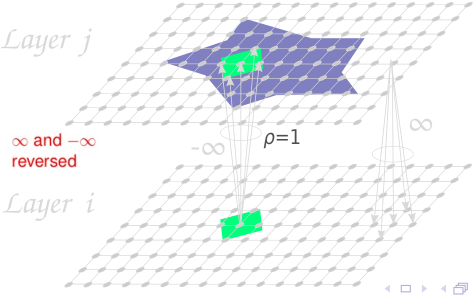
Inter-layer edges indicating inclusions

If O_j is the parent of O_i (i.e., $h(i) = j$),

we add all edges $\langle (i, c), (j, d) \rangle$ with $\|c - d\| \leq \rho$.

For $s = (i, c)$ and $t = (j, d)$ we define

$w_1(s, t) = w_0(t, s) = \infty$ and $w_0(s, t) = w_1(t, s) = -\infty$.



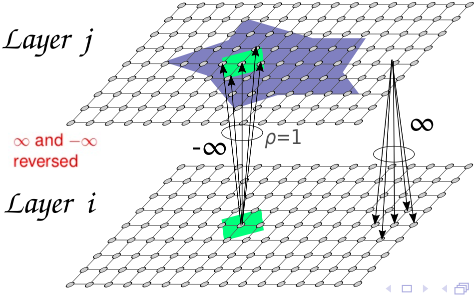
Inter-layer edges indicating inclusions

If O_j is the parent of O_i (i.e., $h(i) = j$),

we add all edges $\langle (i,c),(j,d)\rangle$ with $\|c-d\| \le \rho$.

For $s=(i,c)$ and $t=(j,d)$ we define

$w_1(s,t) = w_0(t,s) = \infty$ and $w_0(s,t) = w_1(t,s) = -\infty$.



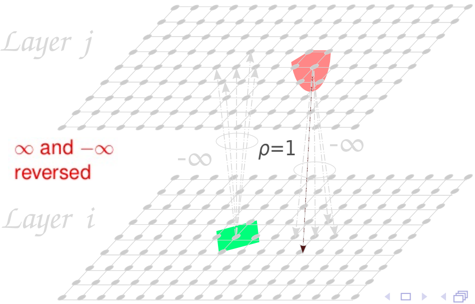
Inter-layer edges indicating exclusions

If O_i and O_j are siblings (i.e., $h(i) = h(j)$ and $i \neq j$),

we add all edges $\langle (i,c),(j,d) \rangle$ with $\|c - d\| \leq \rho$.

For $s = (i,c)$ and $t = (j,d)$ we define

$w_1(s,t) = w_0(t,s) = w_0(s,t) = w_1(t,s) = \infty$.



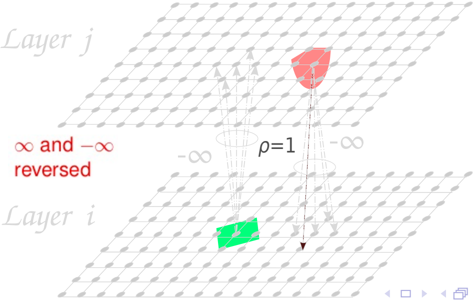
Inter-layer edges indicating exclusions

If O_i and O_j are siblings (i.e., $h(i) = h(j)$ and $i \neq j$),

we add all edges $\langle (i, c), (j, d) \rangle$ with $\|c - d\| \leq \rho$.

For $s = (i, c)$ and $t = (j, d)$ we define

$$w_1(s, t) = w_0(t, s) = w_0(s, t) = w_1(t, s) = \infty.$$



Inter-layer edges indicating exclusions

If O_i and O_j are siblings (i.e., $h(i) = h(j)$ and $i \neq j$),

we add all edges $\langle (i, c), (j, d) \rangle$ with $\|c - d\| \leq \rho$.

For $s = (i, c)$ and $t = (j, d)$ we define

$w_1(s, t) = w_0(t, s) = w_0(s, t) = w_1(t, s) = \infty$.

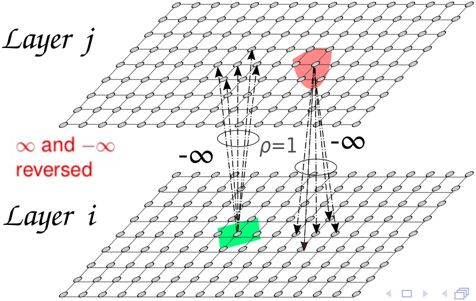


Illustration of the inter-layer arc construction

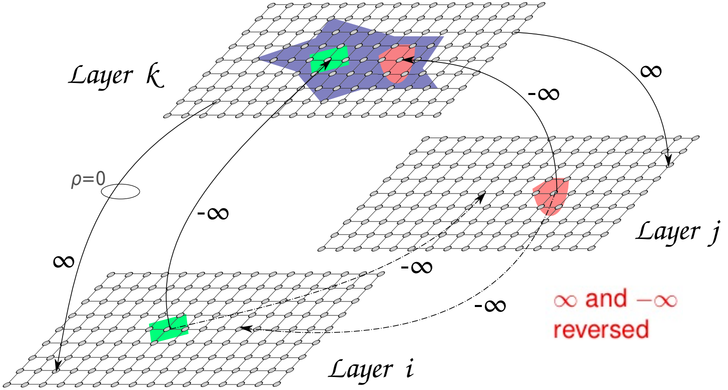


Figure: Illustration of the inter-layer arc construction, involving three objects O_i, O_j, and O_k, where O_k is the parent of two sibling objects, O_i and O_j, i.e., $h(i) = h(j) = k$.

HLOIFT Algorithm

Data: Weighted digraph $\mathcal{N}$; ψ_{last} from image and sets $\bar{\mathcal{S}}_0$, $\bar{\mathcal{S}}_1$
Result: Array $\pi[\,]$ of paths, $\pi[t]$ being a path from a seed to t

foreach $t \in \mathcal{N}$ **do** $\pi[t] \leftarrow \langle t \rangle$ and $S(t) \leftarrow 0$;
$Q \leftarrow \bar{\mathcal{S}}_0 \cup \bar{\mathcal{S}}_1$
while $Q \neq \emptyset$ **do**
    remove an element s of $\max_{t \in Q} \psi_{\text{last}}(\pi[t])$ **from** Q
    $S(s) \leftarrow 1$
    foreach x *such that* $\langle s, x \rangle \in E$ *and* $S(x) = 0$ **do**
        if $\psi_{\text{last}}(\pi[x]) < \psi_{\text{last}}(\pi[s]\hat{\ }x)$ *and*
            [$\pi[s]$ is from $\bar{\mathcal{S}}_1$ or s and x are not siblings] **then**
            $\pi[x] \leftarrow \pi[s]\hat{\ }x$
            if $x \notin Q$ **then** insert t in Q

HLOIFT Algorithm

Data: Weighted digraph $\mathcal{N}$; ψ_{last} from image and sets $\bar{\mathcal{S}}_0$, $\bar{\mathcal{S}}_1$
Result: Array $\pi[\,]$ of paths, $\pi[t]$ being a path from a seed to t

```
foreach t ∈ N do π[t] ← ⟨t⟩ and S(t) ← 0;
Q ← S̄0 ∪ S̄1
while Q ≠ ∅ do
    remove an element s of max_{t∈Q} ψlast(π[t]) from Q
    S(s) ← 1
    foreach x such that ⟨s, x⟩ ∈ E and S(x) = 0 do
        if ψlast(π[x]) < ψlast(π[s]^x) and
                [π[s] is from S̄1 or s and x are not siblings] then
            π[x] ← π[s]^x
            if x ∉ Q then insert t in Q
```

HLOIFT Algorithm

Data: Weighted digraph $\mathcal{N}$; ψ_{last} from image and sets $\bar{\mathcal{S}}_0$, $\bar{\mathcal{S}}_1$
Result: Array $\pi[\,]$ of paths, $\pi[t]$ being a path from a seed to t

```
foreach t ∈ N do π[t] ← ⟨t⟩ and S(t) ← 0;
Q ← S̄_0 ∪ S̄_1
while Q ≠ ∅ do
    remove an element s of max_{t∈Q} ψ_last(π[t])  from Q
    S(s) ← 1
    foreach x such that ⟨s, x⟩ ∈ E and S(x) = 0 do
        if ψ_last(π[x]) < ψ_last(π[s]^x) and
              [π[s] is from S̄_1 or s and x are not siblings] then
            π[x] ← π[s]^x
            if x ∉ Q then insert t in Q
```

HLOIFT Algorithm

Data: Weighted digraph $\mathcal{N}$; $\psi_{\rm last}$ from image and sets $\bar{\mathcal{S}}_0$, $\bar{\mathcal{S}}_1$
Result: Array $\pi[\,]$ of paths, $\pi[t]$ being a path from a seed to t

foreach $t \in \mathcal{N}$ **do** $\pi[t] \leftarrow \langle t \rangle$ and $S(t) \leftarrow 0$;
$Q \leftarrow \bar{\mathcal{S}}_0 \cup \bar{\mathcal{S}}_1$
while $Q \neq \emptyset$ **do**
  remove an element s of $\max_{t \in Q} \psi_{\rm last}(\pi[t])$ **from** Q
  $S(s) \leftarrow 1$
  foreach x *such that* $\langle s, x \rangle \in E$ *and* $S(x) = 0$ **do**
    if $\psi_{\rm last}(\pi[x]) < \psi_{\rm last}(\pi[s]\hat{\ }x)$ *and*
      [$\pi[s]$ is from $\bar{\mathcal{S}}_1$ or s and x are not siblings] **then**
      $\pi[x] \leftarrow \pi[s]\hat{\ }x$
      if $x \notin Q$ **then** insert t in Q

HLOIFT Algorithm

Data: Weighted digraph $\mathcal{N}$; ψ_{last} from image and sets $\bar{\mathcal{S}}_0$, $\bar{\mathcal{S}}_1$

Result: Array $\pi[\,]$ of paths, $\pi[t]$ being a path from a seed to t

foreach $t \in \mathcal{N}$ **do** $\pi[t] \leftarrow \langle t \rangle$ and $S(t) \leftarrow 0$;

$Q \leftarrow \bar{\mathcal{S}}_0 \cup \bar{\mathcal{S}}_1$

while $Q \neq \emptyset$ **do**

  remove an element s of $\max_{t \in Q} \psi_{\text{last}}(\pi[t])$ **from** Q

  $S(s) \leftarrow 1$

  foreach x *such that* $\langle s, x \rangle \in E$ *and* $S(x) = 0$ **do**

    if $\psi_{\text{last}}(\pi[x]) < \psi_{\text{last}}(\pi[s]\hat{\ }x)$ *and*

      [$\pi[s]$ *is from* $\bar{\mathcal{S}}_1$ *or* s *and* x *are not siblings]* **then**

      $\pi[x] \leftarrow \pi[s]\hat{\ }x$

      if $x \notin Q$ **then** insert t in Q

HLOIFT Algorithm

Data: Weighted digraph $\mathcal{N}$; ψ_{last} from image and sets $\bar{\mathcal{S}}_0$, $\bar{\mathcal{S}}_1$

Result: Array $\pi[\,]$ of paths, $\pi[t]$ being a path from a seed to t

foreach $t \in \mathcal{N}$ **do** $\pi[t] \leftarrow \langle t \rangle$ and $S(t) \leftarrow 0$;

$Q \leftarrow \bar{\mathcal{S}}_0 \cup \bar{\mathcal{S}}_1$

while $Q \neq \emptyset$ **do**

    remove an element s of $\max_{t \in Q} \psi_{\text{last}}(\pi[t])$ **from** Q

    $S(s) \leftarrow 1$

    foreach x *such that* $\langle s, x \rangle \in E$ *and* $S(x) = 0$ **do**

        if $\psi_{\text{last}}(\pi[x]) < \psi_{\text{last}}(\pi[s]\hat{}x)$ *and*

            [$\pi[s]$ is from $\bar{\mathcal{S}}_1$ or s and x are not siblings] **then**

            $\pi[x] \leftarrow \pi[s]\hat{}x$

            if $x \notin Q$ **then** insert t in Q

Correctness of HLOIFT

Theorem (Leon, Ciesielski, Miranda, submitted)

An object O returned by HLOIFT generates objects $\langle O_0, \ldots, O_m\rangle$ which are consistent with the seeds $\langle \mathcal{S}_0, \ldots, \mathcal{S}_m\rangle$ and the hierarchy indicated by h.

Moreover, the graph cut $c(O)$ associated with O minimizes its ℓ_∞ norm among all such objects, where

$$c(O) = \{\langle s,t\rangle \in E \colon s \in O \,\&\, t \notin O \,\&\, s \text{ and } t \text{ are not siblings}\} \cup \{\langle s,t\rangle \in E \colon s,t \in O \,\&\, s \text{ and } t \text{ are siblings}\}.$$

Correctness of HLOIFT

Theorem (Leon, Ciesielski, Miranda, submitted)

An object O returned by HLOIFT generates objects $\langle O_0, \ldots, O_m \rangle$ which are consistent with the seeds $\langle \mathcal{S}_0, \ldots, \mathcal{S}_m \rangle$ and the hierarchy indicated by h.
Moreover, the graph cut $c(O)$ associated with O minimizes its ℓ_∞ norm among all such objects, where

$$\begin{aligned} c(O) &= \{\langle s,t\rangle \in E\colon s\in O\ \&\ t\notin O\ \&\ s \text{ and } t \text{ are not siblings}\} \\ &\cup \{\langle s,t\rangle \in E\colon s,t\in O\ \&\ s \text{ and } t \text{ are siblings}\}. \end{aligned}$$

Correctness of HLOIFT

Theorem (Leon, Ciesielski, Miranda, submitted)

An object O returned by HLOIFT generates objects $\langle O_0, \ldots, O_m \rangle$ which are consistent with the seeds $\langle \mathcal{S}_0, \ldots, \mathcal{S}_m \rangle$ and the hierarchy indicated by h.
Moreover, the graph cut $c(O)$ associated with O minimizes its ℓ_∞ norm among all such objects, where

$$\begin{aligned} c(O) &= \{\langle s,t \rangle \in E \colon s \in O \ \&\ t \notin O \ \&\ s \text{ and } t \text{ are not siblings}\} \\ &\cup \{\langle s,t \rangle \in E \colon s,t \in O \ \&\ s \text{ and } t \text{ are siblings}\}. \end{aligned}$$

Outline

Experiment #1

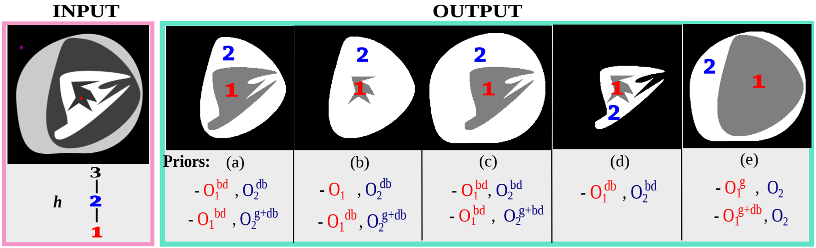
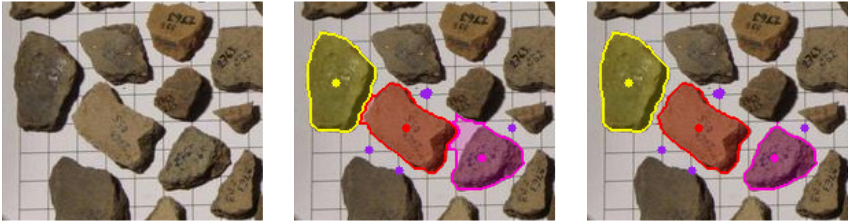


Figure: Example of two object segmentation by HLOIFT, where O_2 is parent of O_1. Each object has different high-level priors –db: polarity from dark to bright pixels, bd: polarity from bright to dark pixels and g: geodesic star convexity prior. We used $\rho = 1.5$. Only two seeds.

Experiment #2



Input image $\rho = 0$ $\rho = 2$

Figure: Example showing how changing the ρ value from 0 to 2 can improve the archaeological fragment segmentation by HLOIFT, avoiding a result with touching objects.

Experiment #3

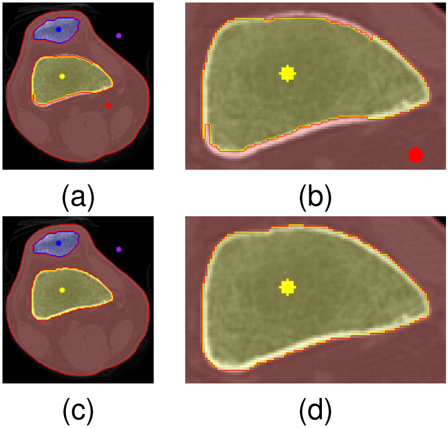


Figure: Knee segmentation composed of three objects in a CT image. (a-b) Result by IFT where the O_1 is mixing bright & dark boundaries. (c-d) An improved result is obtained by HLOIFT with boundary polarity from bright to dark pixels, requiring fewer seeds.

Experiment #4

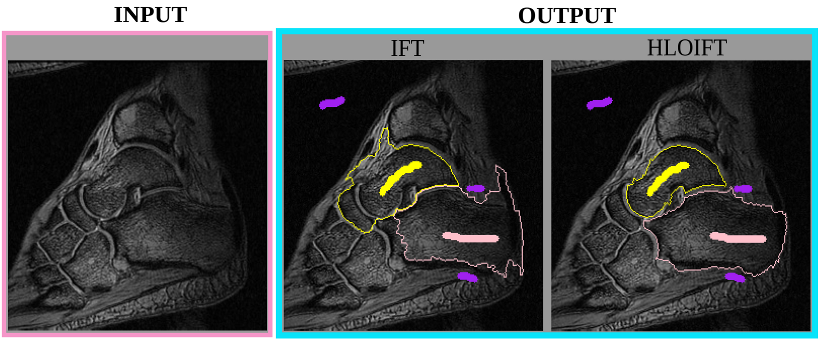


Figure: Talus (O_1) and calcaneus (O_2) segmentation. The two objects are sibling objects. For HLOIFT, we used $\rho = 0$, the geodesic star convexity and boundary polarity ($\alpha = -0.75$).

Exper. #5: CT, thoracic-abdominal axial cross section

INPUT

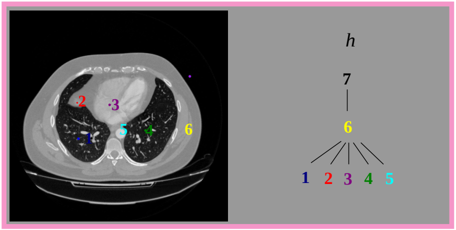
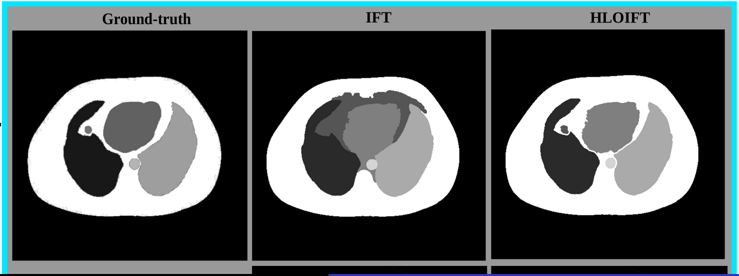


Figure: right lung (O_1), liver (O_2), heart (O_3), left lung (O_4), aorta (O_5) and the thoracic-abdominal region (O_6).

OUTPUT



Experiment #6

(a) (b) (c)

Figure: Flower segmentation in two objects, the central part in cyan and the petals in yellow, using the inclusion relation. (a) The input image. (b) Result by the min-cut/max-flow algorithm in layered graphs. (c) Result by HLOIFT.

Efficiency: HLOIFT versus min-cut/max-flow

Image size (pixels)	Time of HLOIFT (ms)	Time of min-cut/max-flow (ms)
380×320	114.65	323.61
760×640	488.62	1,798.91
1520×1280	1,823.55	19,021.71

The running times for the flower segmentation by HLOIFT and the min-cut/max-flow algorithm in layered graphs using different image sizes.

Outline

Summary

- We described efficient multi-object segmentation algorithm HLOIFT, which can use orientation, hierarchical relations between objects, and high-level priors for each object.
- We placed HLOIFT within a general framework of FC/IFT, which allows us to conclude its provable robustness on seed placements.
- We proved that the objects returned by HLOIFT are consistent with seeds placement and given hierarchy.
- We proved that the output of HLOIFT minimizes appropriate graph cut energy.

Summary

- We described efficient multi-object segmentation algorithm HLOIFT, which can use orientation, hierarchical relations between objects, and high-level priors for each object.
- We placed HLOIFT within a general framework of FC/IFT, which allows us to conclude its provable robustness on seed placements.
- We proved that the objects returned by HLOIFT are consistent with seeds placement and given hierarchy.
- We proved that the output of HLOIFT minimizes appropriate graph cut energy.

Summary

- We described efficient multi-object segmentation algorithm HLOIFT, which can use orientation, hierarchical relations between objects, and high-level priors for each object.
- We placed HLOIFT within a general framework of FC/IFT, which allows us to conclude its provable robustness on seed placements.
- We proved that the objects returned by HLOIFT are consistent with seeds placement and given hierarchy.
- We proved that the output of HLOIFT minimizes appropriate graph cut energy.

Summary

- We described efficient multi-object segmentation algorithm HLOIFT, which can use orientation, hierarchical relations between objects, and high-level priors for each object.
- We placed HLOIFT within a general framework of FC/IFT, which allows us to conclude its provable robustness on seed placements.
- We proved that the objects returned by HLOIFT are consistent with seeds placement and given hierarchy.
- We proved that the output of HLOIFT minimizes appropriate graph cut energy.

Credits

- K.C. Ciesielski, J.K. Udupa, A.X. Falcão, P.A.V. Miranda, "Fuzzy Connectedness image segmentation in Graph Cut formulation," J. Math. Imaging Vision 44(3) (2012), 375-398
- K.C. Ciesielski, A.X. Falcão, P.A.V. Miranda, "Path-value functions for which Dijkstra's algorithm returns optimal mapping," J. Math. Imaging Vision 60(7) (2018), 1025-1036
- K.C. Ciesielski, Gabor T. Herman, T. Yung Kong, "General Theory of Fuzzy Connectedness Segmentations," J. Math. Imaging Visionn 55(3) (2016), 304-342;
- L.M.C. Leon, K.C. Ciesielski, P.A.V. Miranda, "Efficient Hierarchical Multi-Object Segmentation in Layered Graph," (2018), submitted.

Thank you for your attention!