New Results on the Minimum Barrier Distance

Krzysztof Chris Ciesielski

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

Based on a joint work with Robin Strand, Punam K. Saha, and Filip Malmberg

MIPG Seminar, UPenn, April 11, 2014

Outline

Outline

1. Path-induced distance mappings
2. The Minimum Barrier Distance, MBD
3. Fast computation of approximations of MBD
4. Polynomial time algorithm for exact MBD
5. Experiments: comparison of different algorithms for MBD
6. Experiments: segmentations for different distances
7. Conclusions

Image, scene, and the associated graph

Let $f\colon C \to \mathbb{R}^\ell$ be a digital image, where

$C = \mathbb{Z}^k \cap \prod_{i=1}^{k}[a_i, b_i]$ ($a_i, b_i \in \mathbb{R}$) is a digital scene

with $x, y \in C$ ($2k$-)adjacent provided $\sum_i |x_i - y_i| = 1$.

We will treat also this structure as a graph $G = \langle C, E\rangle$,

with vertices C and edges $E = \{\{x, y\}\colon x, y \in C \text{ adjacent}\}$.

(Most theory actually works for arbitrary graphs.)

From path strength to generalized distance

Π — all paths $p = \langle c_0, \ldots, c_k \rangle$ in $G = \langle C, E \rangle$, i.e., $\{c_i, c_{i+1}\} \in E$.

$\Pi_{c,d}$ — all paths from $c \in C$ to $d \in C$.

For a fixed path strength map $\lambda \colon \Pi \to [0, \infty)$

a "distance" is $d_\lambda(c, d) = \min\{\lambda(\pi) \colon \pi \in \Pi_{c,d}\}$.

Example. If $w \colon E \to [0, \infty)$ is an edge weight map on G,

with $w(\{c, d\})$ being a (geodesic) distance from c to d,

then d_Σ is the *geodesic metric*, where

$\Sigma(\langle \pi(0), \pi(1), \ldots, \pi(k) \rangle) = \sum_{i=1}^{k} w(\{\pi(i-1), \pi(i)\})$.

Generalized distance

$d\colon C^2 \to [0,\infty)$ is a *generalized distance mappings* if

it is symmetric and satisfies the triangle inequality.

(We allow possibility that $d(c,c) > 0$ for some $c \in C$.)

Theorem

Assume that for every path $\pi = \langle \pi(0), \pi(1), \ldots, \pi(k)\rangle$

(i) $\lambda(\pi) = \lambda(\langle \pi(k), \pi(k-1), \ldots, \pi(0)\rangle)$, *and*

(ii) $\lambda(\pi) \le \lambda(\langle \pi(0), \ldots, \pi(i)\rangle) + \lambda(\langle \pi(i), \ldots, \pi(k)\rangle)$ *for every* $0 \le i \le k$.

Then d_λ *is a generalized distance.*

All maps d_λ we consider are generalized distances.

Outline

Definition of the Minimum Barrier Distance, MBD

Let $w \colon C \to [0, \infty)$ be vertex weight map, e.g., $w(c) = \|f(c)\|$.

For a path $p = \langle c_i \rangle \in \Pi$ let $\beta_w(p) = \beta_w^+(p) - \beta_w^-(p)$, where

$\beta_w^+(p) = \max_i w(c_i)$ and $\beta_w^-(p) = \min_i w(c_i)$.

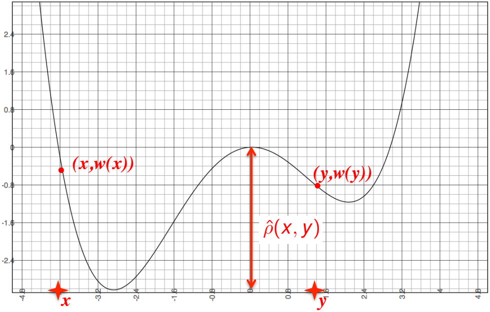
β_w is the barrier cost.

The Minimum Barrier Distance, MBD

between x and y in C

is $d_{\beta_w}(x, y)$, i.e.,

$d_{\beta_w}(x, y) = \min\{\beta_w(p) \colon p \in \Pi_{x,y}\}$.



$\hat{\rho} = d_{\beta_w}$

MBD vs geodesic distance

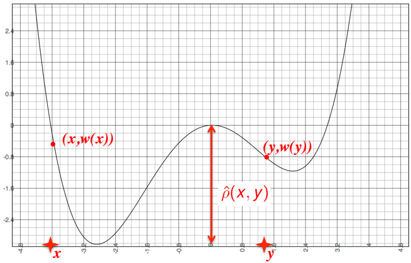
$d_{\beta_w}(x,y) = \min\{c_b(p)\colon p \text{ is a path in } G \text{ from } x \text{ to } y\}$

$d_{\beta_w}(x,y)$ is, in a way,

a vertical component of

the geodesic distance d_Σ

between x and y.



d_{β_w} is a pseudo-metric: it is symmetric,

satisfies the triangle inequality, and $d_{\beta_w}(x,x) = 0$.

(However, $d_{\beta_w}(x,y)$ can be equal 0 for $x \neq y$.)

Generalized distances used in imaging

- Geodesic Distance, d_Σ, including the Euclidean Distance
- Fuzzy Connectedness, FC: if μ is FC connectivity strength for affinity $\kappa\colon E \to [0, M]$ and weight $w(e) = M - \kappa(e)$, then $d_\lambda(c, d) = M - \mu(c, d)$, where $\lambda(\langle c_i \rangle) = \max_i w(\{c_{i-1}, c_i\})$.
- Our new Minimum Barrier Distance, d_{β_w}
- Fuzzy Distance, FD: it is $d_{\hat{\Sigma}}$, where for $w\colon C \to [0, \infty)$ $\hat{w}(c, d) = \frac{w(c)+w(d)}{2}$ and $\hat{\Sigma}(\langle c_i \rangle) = \sum_i \hat{w}(\{c_{i-1}, c_i\})$
- Watershed: it is $d_{\beta_w^+}$ $(\beta_w^+(\langle c_i \rangle) = \max_i w(c_i))$

For distance d and seed sets $S, T \subset C$, define RFC-like object:

$$P(S, T) = \{c \in C\colon d(c, S) < D(c, T)\}.$$

We experimentally compared these for d_Σ, FC, MBD, FD.

Outline

Standard Dijkstra algorithm, DA, for cost function λ

Algorithm 1 Dijkstra algorithm $DA(\lambda, R)$

Input: Path cost function λ on $G = \langle C, E \rangle$, non-empty $R \subset C$.
Output: For every $c \in C$, a path π_c from an $r \in R$ to c.
Auxiliary: Queue Q: if c precedes d in Q, then $\lambda(\pi_c) \leq \lambda(\pi_d)$.
begin

Init: $p_r = \langle r \rangle$ for $r \in R$, $p_c = \emptyset$ for $c \notin R$, push all $r \in R$ to Q;
while Q is not empty **do**
    Pop d from Q;
    for every $c \in C$ connected by an edge to d **do**
        if $\lambda(\pi_d \hat{\ } c) < \lambda(\pi_c)$ **then**
            Put $\pi_c = \pi_d \hat{\ } c$, place c into a proprer place in Q;
        end if
    end for
end while

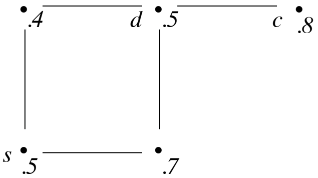
end

Can Dijkstra Algorithm, DA, find (exact) MBD?

DA returns correctly distances: Geodesic, FC, FD, Watershed,

as their paths strengths are *smooth* in sense of Falcão et al.

DA does not work properly for MBD:



Example: MBD value $d_{\beta_w}(s, c) = .8 - .5$ for the indicated w.

$DA(\beta_w, \{s\})$ returns suboptimal π_c, with $\beta_w(\pi_c) = .8 - .4$.

Fast algorithms approximating MBD

Algorithm 2 $A^{appr}_{MBD}(\{s\})$

Input: A vertex weight map w on a graph $G = \langle C, E \rangle$, an $s \in C$.
Output: A map $\varphi(\cdot, \{s\}))$.
begin

1: Run $DA(\beta^+_w, \{s\}))$; record $d_{\beta^+_w}(c, \{s\})) = \beta^+_w(\pi_c)$ for $c \in C$;
2: Run $DA(\beta^+_v, \{s\}))$, where $v = M - w$ and $M = \max_{c \in C} w(c)$, and record $d_{\beta^-_w}(c, \{s\})) = M - \beta^+_v(\pi_c)$ for every $c \in C$;
3: Return $\varphi(\cdot, \{s\})) = d_{\beta^+_w}(c, \{s\})) - d_{\beta^-_w}(c, \{s\}))$ for $c \in C$;

end

The output of $A^{appr}_{MBD}(\{s\})$ approximates MBD $d_{\beta_w}(\cdot, \{s\}))$:

$\varphi(\cdot, \{s\})) \approx d_{\beta_w}(\cdot, \{s\}))$

$G = \langle C, E, w \rangle$ — graph of a rectangular k-D image f, $w = \|f\|$,

$\varepsilon = \max\{|w(x) - w(y)| \colon x, y \in C$ are $(2^k - 1)$-adjacent$\}$.

Theorem ($\varphi(c, s) \leq d_{\beta_w}(c, s) \leq \varphi(c, s) + 2\varepsilon$)

Proof is based on deep result on continuous equivalent of MBD:

For f being continuous on a simple connected domain,

continuous-$\varphi(c, d)$ = continuous-$d_{\beta_w}(c, d)$.

Proof of Thm:
(1) Extend f to continuous $\hat{f}$ via k-linear interpolation.
(2) Find continuous path $p \in \Pi_{x,y}$ with $\beta_w(p) \approx \varphi(x, y)$.
(3) Digitize p.

$A^{appr}_{MBD}(S)$ and $DA(\beta_w, S)$: pros and cons

- Both fast, in order between $O(n)$ and $O(n \ln n)$, $n = |C|$.
- $A^{appr}_{MBD}(S)$ underestimates MBD, with known error rate ε;
  needs to run "simple" DA $|S|$-many times, slowing for large S.
- $DA(\beta_w, S)$ overestimates MBD with unknown error bound;
  complexity is (essentially) independent of the size of S;

Conjecture

The error of $DA(\beta_w, S)$ does not exceed 2ε, maybe even ε.

So far, no theoretical proof for this.

Outline

Simple algorithm for exact MBD

Algorithm 3 $A_{MBD}^{simple}(S)$

Input: A vertex weight w on $G = \langle C, E \rangle$, non-empty $S \subset C$.
Output: The paths p_c from S to c with $\beta_w(p_c) = d_{\beta_w}(c, S)$.

```
begin
  Init: U = max{w(s): s ∈ S} and p_c = ∅ for every c ∈ C;
  Push all numbers from {w(c) ≤ U: c ∈ C} to a queue Q;
  while Q is not empty do
      Pop a from Q, run DA(β_v^+, S) with v = w_a, return π_c's;
            (w_a(c) = w(c) if w(c) ≥ a, w_a(c) = ∞ otherwise)
      for every c ∈ C do
          if β_v(π_c) < β_w(p_c) then
              Put p_c = π_c;
          end if
      end for
  end while
end
```

Faster algorithm for exact MBD

Algorithm 4 $A_{MBD}(S)$

Auxiliary: β_w^--optimal π_c from S to c; a queue Q: if $c \preceq d$ then $\beta_w^+(\pi_c) < \beta_w^+(\pi_d)$ or $\beta_w^+(\pi_c) = \beta_w^+(\pi_d)$ and $\beta_w^-(\pi_c) > \beta_w^-(\pi_d)$.

begin

Init: $p_s = \pi_s = \langle s \rangle$ for $s \in S$ and $p_c = \pi_c = \emptyset$ for $c \in C \setminus S$;
Push all $s \in S$ to Q;
while Q is not empty **do**
    Pop c from Q;
    for every $d \in C$ connected by an edge to c **do**
        if $\beta_w^-(\pi_c \hat{}\, d) > \beta_w^-(\pi_d)$ **then**
            Set $\pi_d \leftarrow \pi_c \hat{}\, d$ and place d into Q;
            if $\beta_w(\pi_d) < \beta_w(p_d)$ **then**
                Set $p_d \leftarrow \pi_d$;
            end if
        end if
        End everything;

Correctness of the algorithms for exact MBD

Theorem

Let n be the size of the graph and m be the size of a fix set Z, containing $W = \{w(c) \colon c \in C\}$. The algorithm computational complexity is either

(BH) $O(m\, n \ln n)$, *if we use binary heap as Q, or*

(LS) $O(m(n+m))$, *if we use as Q a list structure.*

After $A_{MBD}(S)$ terminates, we indeed have $\beta_w(p_c) = d_w(c, S)$ for all $c \in C$. The same is true for $A_{MBD}^{simple}(S)$.

Proof for $A_{MBD}(S)$ is quite intricate; for $A_{MBD}^{simple}(S)$ is quite easy.

However, $A_{MBD}(S)$ executes the main *while* loop considerably fewer times than $A_{MBD}^{simple}(S)$ does.

Outline

What is compared?

- the exact MBD algorithm $A_{MBD}(S)$;
- the interval algorithm $DA(\beta_w, S)$ overestimating MBD;
- $A^{appr}_{MBD}(S)$ executed ones for each seed point;
  it underestimates MBD, with an error $\leq 2\varepsilon$;
- $A^{\star appr}_{MBD}(S)$ executed only ones even for multiple seeds.

Experiments were conducted on a computer: HP Proliant ML350 G6 with 2 Intel X5650 6-core processors (2.67Hz) and 104GGB memory.

The used 2D images, from the grabcut dataset, came with the true segmentations. Their sizes range from 113032 pixels (for 284×398 image) to 307200 (for 640×480 image).

2D images from the grabcut dataset

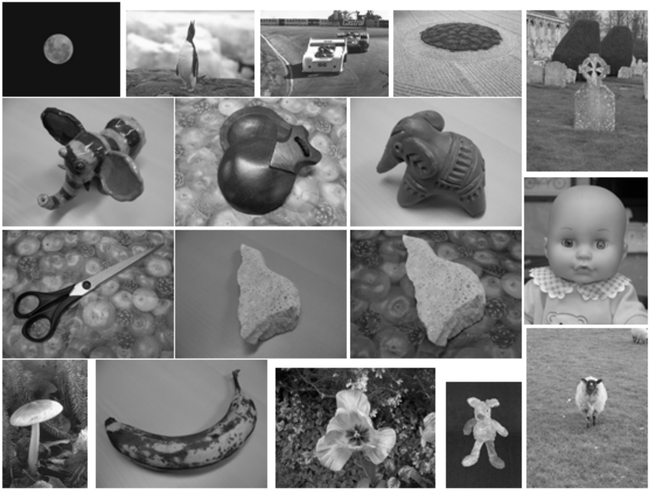
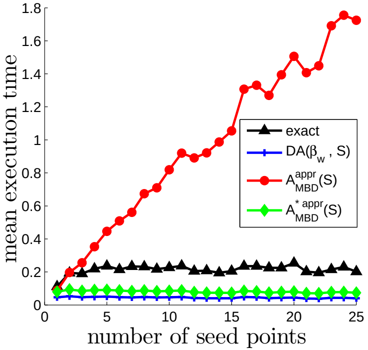
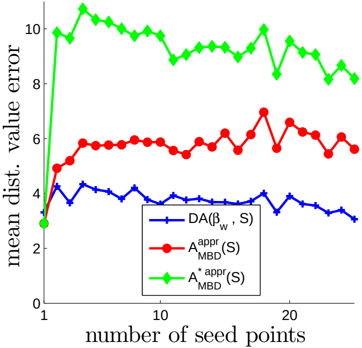


Figure: Images from the grabcut dataset used in the experiments.

Results

For each $s = 1, \ldots, 25$, the following was repeated 100 times:
(1) extract a random image from the database;
(2) generate randomly the set S of s seed points in the image;
(3) run each algorithm on this image with the chosen set S.
Graphs display averages.





More results and conclusions

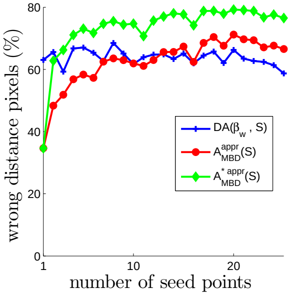


Figure: The mean number pixels with incorrect value of MBD

We declared as "winners," used in the segmentation experiments:

$A_{MBD}(S)$ as it is exact and reasonably fast;

$DA(\beta_w, S)$ as it is the fastest and has the smallest error from approximations.

Outline

Algorithms used in the segmentation valuation

For gray-scale digital images $f\colon C \to [0,\infty)$:

- The *exact MBD* computed with $A_{MBD}(S)$, where $w(c) = f(c)$.
- An *approximate MBD* computed with $DA(\beta_w, S)$, where $w(c) = f(c)$.
- The *geodesic distance* computed with $DA(\Sigma, S)$, where, for adjacent $c, d \in C$, $w(c,d) = |f(c) - f(d)|$.
- The *fuzzy distance* computed with $DA(\hat{\Sigma}, S)$, where $w(c) = f(c)$.
- The *fuzzy connectedness* computed with $DA(w, S)$, where, for adjacent $c, d \in C$, $w(c,d) = M - \kappa(c,d) = |f(c) - f(d)|$.

We start with the 2D grabcut images.

Speed w.r.t. image size

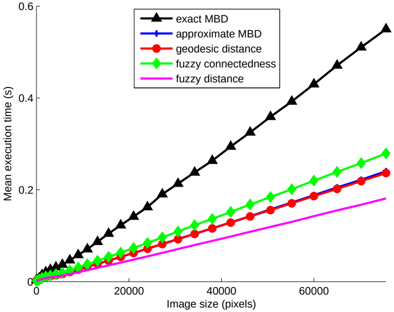
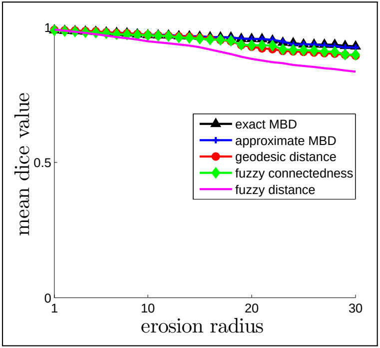


Figure: Mean execution time on small images obtained by cutting out grabcut images. A single seed point is used for each image.

The actual execution time of $A_{MBD}(S)$ depends on the image size in a linear manner, rather than in the (worst case scenario proven) quadratic manner.

Seeds chosen by erosion, no noise or blur



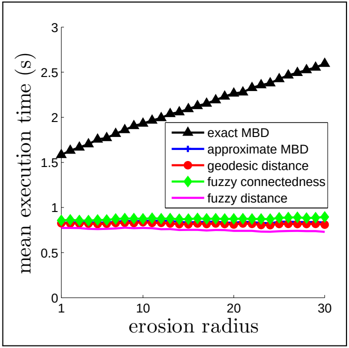


Figure: The value for each algorithm for the seeds chosen for indicated erosion radius represent average over the 17 images.

All algorithms performed well, with just a slight better accuracy for MBD algorithms.

Seeds chosen by the users, no noise or blur

Figure: Example of seed points, users 1–4, respectively.

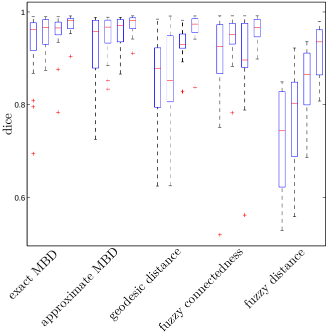


Figure: Boxplots of Dice coefficient, seeds from users 1–4.

Seeds chosen by the users, smoothing added

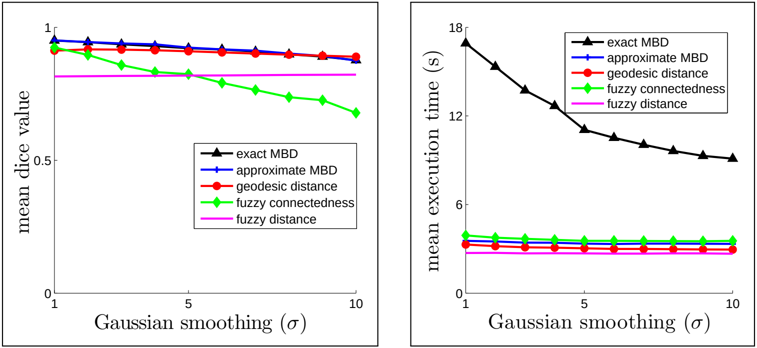


Figure: The performance of the five algorithms as a function of smoothing the images.

MBD algorithms handled smoothing a lot better than FC and FD

Smoothing improves execution time for exact MBD algorithm

Seeds chosen by the users, noise added

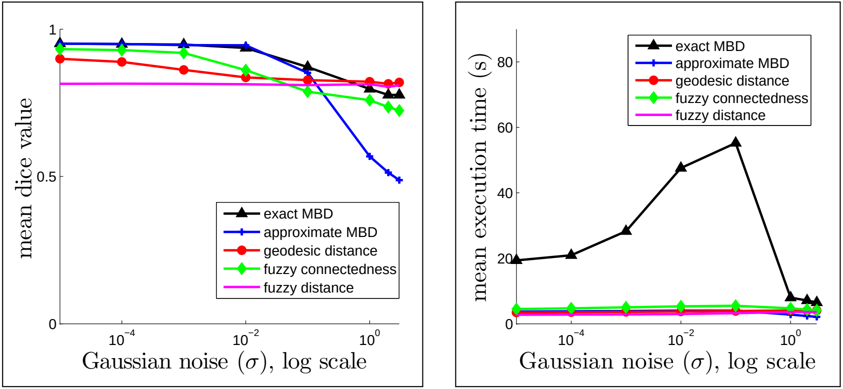


Figure: The performance of the five algorithms as a function of adding noise to the images.

MBD algorithms handled noise better than other algorithms for not very noisy images

Blur added to the images with fixed level of noise

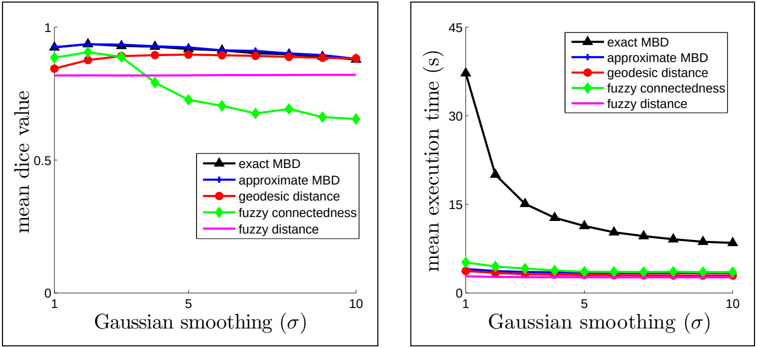


Figure: The performance of the five algorithms as a function of smoothing, applied to the images with added fixed level of noise.

Noise added to the smoothed images

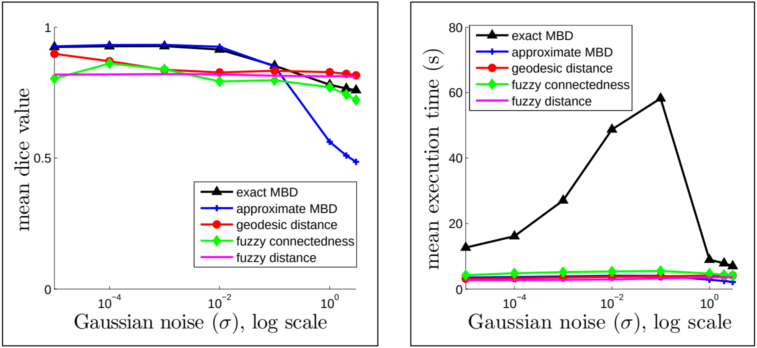


Figure: The performance of the five algorithms as a function of adding noise, applied to the smoothed images.

3D experiments: the image

(a) (b) (c)

Figure: The 3D T1-weighted MRI image of the brain, smoothed by Gaussian blur with sigma value 0.5. (a) three perpendicular slices; (b) reference segmentation of the same slices; (c) surface rendering of the reference segmentation.

3D experiments: the results

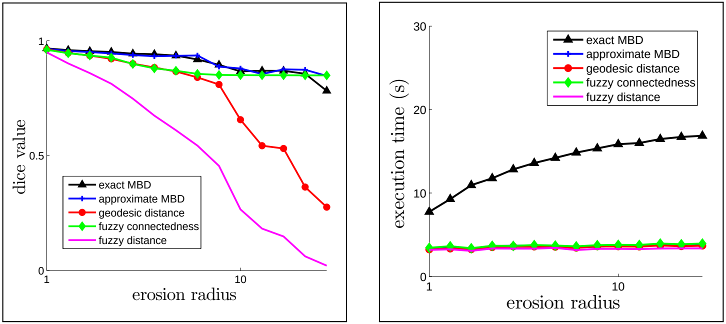


Figure: The performance of the five algorithms on the image for the asymmetrically chosen seeds at the indicated erosion radius.

MBD algorithms compare favorably with the other algorithms

Outline

Summary

Minimum Barrier Distance:

- Can be efficiently computed: (a) exactly; (b) approximately.
- The segmentations associated with MBD compare favorably with those associates with: geodesic distance (GD), fuzzy distance (FD), and relative fuzzy connectedness (RFC).
- The segmentations associated with MBD are more robust to smoothing and to noise than GD, FD, and RFC.

Thank you for your attention!