Shape Representation and Description

Alexandre Falcão

Institute of Computing - University of Campinas

afalcao@ic.unicamp.br

- A segmented object may contain multiple boundaries.

Introduction

- A segmented object may contain multiple boundaries.
- We will focus on 2D boundaries represented by closed, connected and oriented curves (contours).

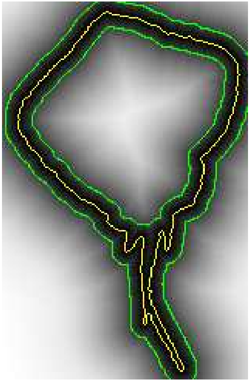
- A segmented object may contain multiple boundaries.
- We will focus on 2D boundaries represented by closed, connected and oriented curves (contours).
- Each contour defines a shape whose properties are very important for image analysis.

- Other shape representations can be derived from a contour and their properties are usually encoded in a more compact representation (i.e., feature vector).

- Other shape representations can be derived from a contour and their properties are usually encoded in a more compact representation (i.e., feature vector).
- Some feature vectors require specific distance functions to compute shape similarities independently of their orientation and size.

- Other shape representations can be derived from a contour and their properties are usually encoded in a more compact representation (i.e., feature vector).
- Some feature vectors require specific distance functions to compute shape similarities independently of their orientation and size.
- The pair, feature extraction function and distance function, is called here a descriptor.

The Euclidean IFT from a contour $\mathcal{S}$ (lecture 2) creates in V multiscale contours (iso-contours) by subsequent exact dilations and erosions of $\mathcal{S}$ [1].



- Each contour is related to its internal and external skeletons (point sets with at least two equidistant pixels in the contour).

- Each contour is related to its internal and external skeletons (point sets with at least two equidistant pixels in the contour).
- The Euclidean IFT can output a labeled map L, which is used to create internal and external multiscale skeletons.

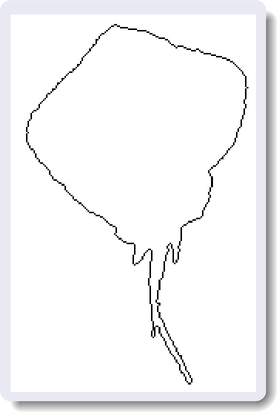
Introduction

- Each contour is related to its internal and external skeletons (point sets with at least two equidistant pixels in the contour).
- The Euclidean IFT can output a labeled map L, which is used to create internal and external multiscale skeletons.
- These skeletons present a highly desirable characteristic of being one-pixel-wide and connected in all scales.

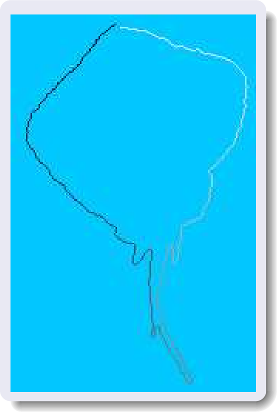
Introduction

- Each contour is related to its internal and external skeletons (point sets with at least two equidistant pixels in the contour).
- The Euclidean IFT can output a labeled map L, which is used to create internal and external multiscale skeletons.
- These skeletons present a highly desirable characteristic of being one-pixel-wide and connected in all scales.
- In the presence of multiple contours, a simple variant computes the skeleton by influence zones (SKIZ — a point set with equidistant pixels in at least two contours).

Introduction



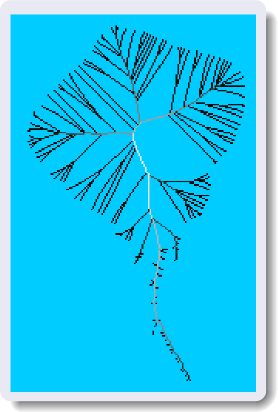
- A given contour $\mathcal{S}$.



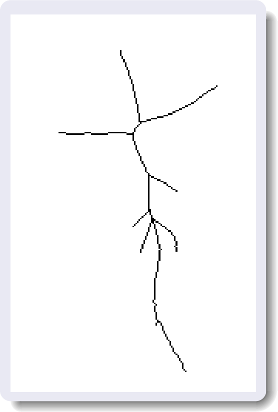
- A given contour $\mathcal{S}$.
- Pixels along $\mathcal{S}$ receive a subsequent label from 1 to $|\mathcal{S}|$.

- A given contour $\mathcal{S}$.
- Pixels along $\mathcal{S}$ receive a subsequent label from 1 to $|\mathcal{S}|$.
- The labels are propagated to form a label map L (discrete Voronoi regions).

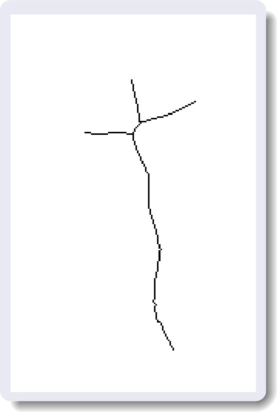
Introduction



- A given contour $\mathcal{S}$.
- Pixels along $\mathcal{S}$ receive a subsequent label from 1 to $|\mathcal{S}|$.
- The labels are propagated to form a label map L (discrete Voronoi regions).
- A multiscale skeleton is created from local differences in L.

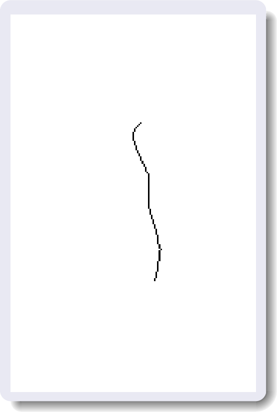


- A given contour $\mathcal{S}$.
- Pixels along $\mathcal{S}$ receive a subsequent label from 1 to $|\mathcal{S}|$.
- The labels are propagated to form a label map L (discrete Voronoi regions).
- A multiscale skeleton is created from local differences in L.
- Skeletons are obtained by thresholding the multiscale skeleton at increasing scales.



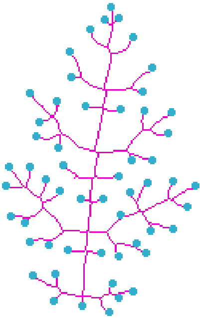
- A given contour $\mathcal{S}$.
- Pixels along $\mathcal{S}$ receive a subsequent label from 1 to $|\mathcal{S}|$.
- The labels are propagated to form a label map L (discrete Voronoi regions).
- A multiscale skeleton is created from local differences in L.
- Skeletons are obtained by thresholding the multiscale skeleton at increasing scales.

Introduction

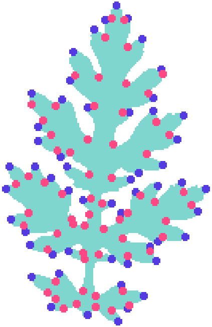


- A given contour $\mathcal{S}$.
- Pixels along $\mathcal{S}$ receive a subsequent label from 1 to $|\mathcal{S}|$.
- The labels are propagated to form a label map L (discrete Voronoi regions).
- A multiscale skeleton is created from local differences in L.
- Skeletons are obtained by thresholding the multiscale skeleton at increasing scales.

- The Euclidean IFT with a small dilation radius from an internal skeleton $\mathcal{S}$ creates a root map R,

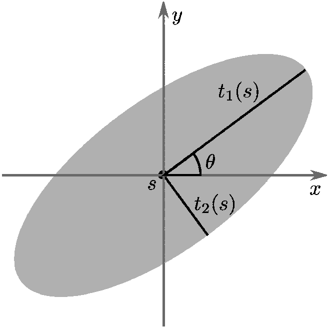


- The Euclidean IFT with a small dilation radius from an internal skeleton $\mathcal{S}$ creates a root map R,
- the aperture angles of the discrete Voronoi regions in R are used to detect salience points of the skeleton,



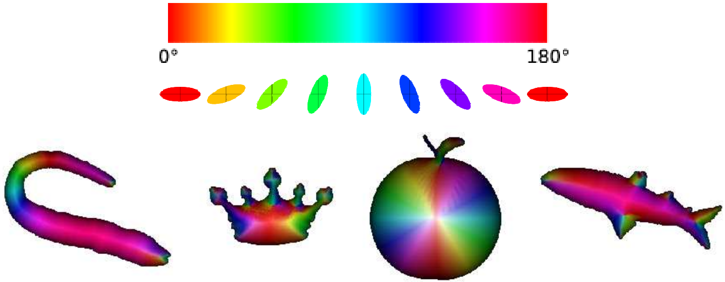
- The Euclidean IFT with a small dilation radius from an internal skeleton $\mathcal{S}$ creates a root map R,
- the aperture angles of the discrete Voronoi regions in R are used to detect salience points of the skeleton,
- from salience points of the internal and external skeletons, we detect convex and concave salience points of the contour.

The Euclidean IFT can also speed up the computation of the largest ellipse (tensor scale) centered at each pixel, creating a region-based shape representation.



- *Orientation*(s) = angle between $t_1(s)$ and the horizontal axis.
- *Anisotropy*$(s) = \sqrt{1 - \frac{|t_2(s)|^2}{|t_1(s)|^2}}$.
- *Thickness*$(s) = |t_2(s)|$.

By using the HSI color space, the tensor orientation at each pixel is represented by a distinct color.



The region-based representation stores orientation and anisotropy at each pixel.

- Muliscale skeletonization and SKIZ [1].

Organization of the lecture

- Muliscale skeletonization and SKIZ [1].
- Contour and skeleton saliences [2].

Organization of the lecture

- Muliscale skeletonization and SKIZ [1].
- Contour and skeleton saliences [2].
- Tensor scale computation [3].

Organization of the lecture

- Muliscale skeletonization and SKIZ [1].
- Contour and skeleton saliences [2].
- Tensor scale computation [3].
- Shape description from these representations [4].

Organization of the lecture

- Muliscale skeletonization and SKIZ [1].
- Contour and skeleton saliences [2].
- Tensor scale computation [3].
- Shape description from these representations [4].
- Combining multiple descriptors [5].

Multiscale skeletonization and SKIZ

- Consider a binary image $\hat{I} = (D_I, I)$ with m disjoint contours $\mathcal{S}_i \subset D_I$, $i = 1, 2, \ldots, m$.

- Consider a binary image $\hat{I} = (D_I, I)$ with m disjoint contours $\mathcal{S}_i \subset D_I$, $i = 1, 2, \ldots, m$.
- By circumscribing each contour in a given orientation (clockwise), a function $\lambda_p(t)$ assigns to each pixel $t \in \mathcal{S}_i$ a subsequent integer number from 1 to $|\mathcal{S}_i|$.

Multiscale skeletonization and SKIZ

- Consider a binary image $\hat{I} = (D_I, I)$ with m disjoint contours $\mathcal{S}_i \subset D_I$, $i = 1, 2, \ldots, m$.
- By circumscribing each contour in a given orientation (clockwise), a function $\lambda_p(t)$ assigns to each pixel $t \in \mathcal{S}_i$ a subsequent integer number from 1 to $|\mathcal{S}_i|$.
- Each contour pixel also receives a number $i = 1, 2, \ldots, m$ by a function $\lambda_c(t)$ to identify its contour.

Multiscale skeletonization and SKIZ

- Let $\mathcal{S} = \cup_{i=1}^{c} \mathcal{S}_i$ be the union set of all contour pixels.

Multiscale skeletonization and SKIZ

- Let $\mathcal{S} = \cup_{i=1}^{c} \mathcal{S}_i$ be the union set of all contour pixels.
- The Euclidean IFT propagates contour pixel labels in L_p and contour labels in L_c inside and outside the contours by using $\mathcal{A}_{\sqrt{2}}$ (8-neighbors) and path function f_{euc},

$$
\begin{aligned}
f_{euc}(\langle t \rangle) &= \begin{cases} 0 & \text{if } t \in \mathcal{S}, \\ +\infty & \text{otherwise}, \end{cases} \\
f_{euc}(\pi_s \cdot \langle s, t \rangle) &= \|t - R(\pi_s)\|^2.
\end{aligned}
$$

Multiscale skeletons and SKIZ

Algorithm

– Euclidean IFT with label propagation

For each $t \in D_I \backslash \mathcal{S}$*, set* $V(t) \leftarrow +\infty$ *and* $R(\pi_t) \leftarrow t$.
For each $t \in \mathcal{S}$*, do*
    Set $V(t) \leftarrow 0$, $L_p(t) \leftarrow \lambda_p(t)$*, and* $L_c(t) \leftarrow \lambda_c(t)$.
    Insert t *in* Q.
While Q *is not empty, do*
    Remove from Q *a pixel* s *such that* $V(s)$ *is minimum.*
    For each $t \in \mathcal{A}_{\sqrt{2}}(s)$ *such that* $V(t) > V(s)$*, do*
        Compute $tmp \leftarrow \|t - R(\pi_s)\|^2$.
        If $tmp < V(t)$*, then*
            If $V(t) \neq +\infty$*, remove* t *from* Q.
            Set $V(t) \leftarrow tmp$ *and* $R(\pi_t) \leftarrow R(\pi_s)$.
            Set $L_p(t) \leftarrow L_p(s)$ *and* $L_c(t) \leftarrow L_c(s)$.
            Insert t *in* Q.

Multiscale skeletons and SKIZ

Multiscale skeletons and SKIZ

- Multiscale skeletons and SKIZ are computed from $L_p(s)$ and $L_c(s)$, respectively, creating a difference map $D(s)$.

Multiscale skeletons and SKIZ

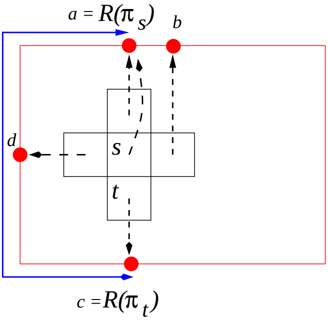
- Multiscale skeletons and SKIZ are computed from $L_p(s)$ and $L_c(s)$, respectively, creating a difference map $D(s)$.
- SKIZ and one-pixel wide and connected skeletons are then obtained by thresholding $D(s)$. Higher the threshold, more simplified become the skeletons.

Multiscale skeletons and SKIZ

- Multiscale skeletons and SKIZ are computed from $L_p(s)$ and $L_c(s)$, respectively, creating a difference map $D(s)$.
- SKIZ and one-pixel wide and connected skeletons are then obtained by thresholding $D(s)$. Higher the threshold, more simplified become the skeletons.
- Multiscale skeletons and SKIZ are computed as follows.

Multiscale skeletons and SKIZ

Each pair of contour points in $\mathcal{S}_i$ "equidistant" to a pixel $s \notin \mathcal{S}_i$ defines two segments between them. Among the shortest segments from each pair, the length of the longest one (blue line) is assigned to $D(s)$.



This condition is relaxed by computing segment lengths between root points (a, b, c, and d) related to s and its 4-neighbors.

Multiscale skeletons and SKIZ

- If $L_c(s) = L_c(t) = i$ for all $t \in \mathcal{A}_1(s)$, then

$$\begin{aligned}\Delta(s,t) &= L_p(t) - L_p(s) \\ D(s) &= \max_{\forall (s,t) \in \mathcal{A}_1} \{\min\{\Delta(s,t), |\mathcal{S}_i| - \Delta(s,t)\}\}.\end{aligned}$$

  Note that, for clockwise contour labeling, $L(a) < L(b) < L(c) < L(d)$, and the FIFO tie-breaking policy will favor the root with lowest label.

Multiscale skeletons and SKIZ

- If $L_c(s) = L_c(t) = i$ for all $t \in \mathcal{A}_1(s)$, then

$$\begin{array}{rcl} \Delta(s,t) & = & L_p(t) - L_p(s) \\ D(s) & = & \max_{\forall (s,t) \in \mathcal{A}_1} \{\min\{\Delta(s,t), |\mathcal{S}_i| - \Delta(s,t)\}\}. \end{array}$$

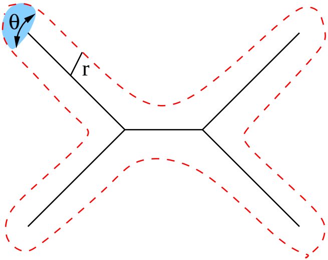
  Note that, for clockwise contour labeling, $L(a) < L(b) < L(c) < L(d)$, and the FIFO tie-breaking policy will favor the root with lowest label.

- When $L_c(s) \neq L_c(t)$ for some $t \in A_1(s)$, then the SKIZ is in between pixels s and t. Since the SKIZ is never filtered by thresholding, for $L_c(t) > L_c(s)$, $D(t) = +\infty$ and $D(s) = 0$, and for $L_c(t) < L_c(s)$, $D(s) = +\infty$ and $D(t) = 0$.

Show demo program.

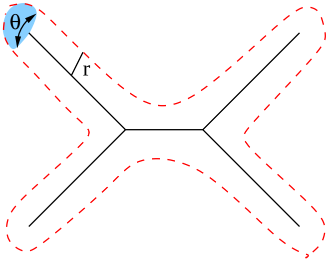
Skeleton saliences

How do we compute skeleton saliences?



Skeleton saliences

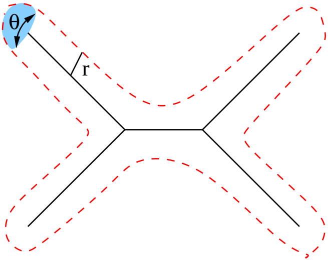
How do we compute skeleton saliences?



- The IFT dilation of the skeletons up to a small radius r (e.g., 10) produces a small influence zone for each point.

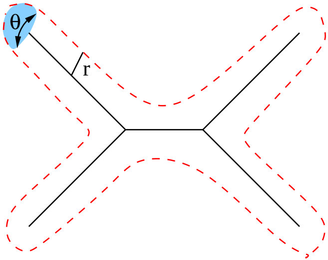
Skeleton saliences

How do we compute skeleton saliences?



- The IFT dilation of the skeletons up to a small radius r (e.g., 10) produces a small influence zone for each point.
- The area $A = \frac{\theta r^2}{2}$ of each influence zone is related to its aperture angle θ at each point.

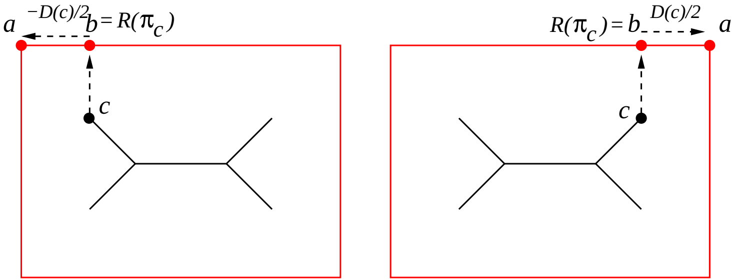
How do we compute skeleton saliences?



- The IFT dilation of the skeletons up to a small radius r (e.g., 10) produces a small influence zone for each point.
- The area $A = \frac{\theta r^2}{2}$ of each influence zone is related to its aperture angle θ at each point.
- Salience points are then obtained by thresholding θ.

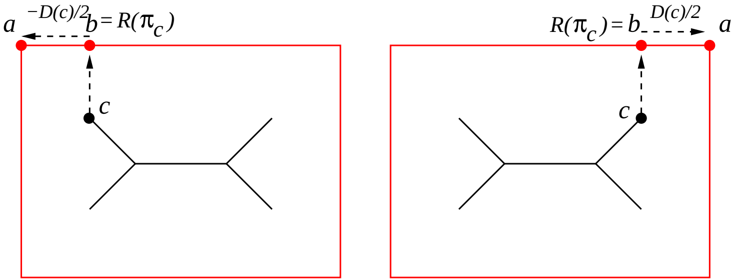
How do we compute contour saliences?

For clockwise contour labeling, a contour salience a is detected from a skeleton salience c by skipping $\frac{D(c)}{2}$ pixels in either anti-clockwise or clockwise from the root $R(\pi_c)$.



How do we compute contour saliences?

For clockwise contour labeling, a contour salience a is detected from a skeleton salience c by skipping $\frac{D(c)}{2}$ pixels in either anti-clockwise or clockwise from the root $R(\pi_c)$.



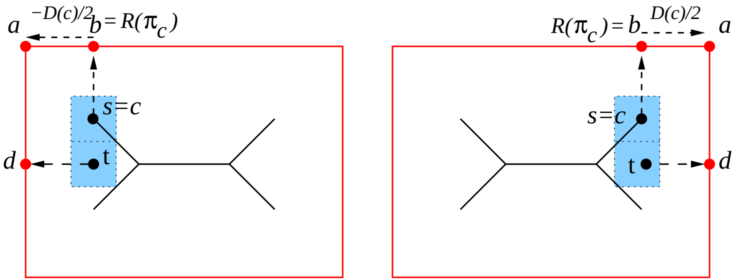
However, how do we know which orientation to go?

Contour and skeleton saliences

Let $\Delta^*(s,t) = L_p(t) - L_p(s)$ be the one which satisfies

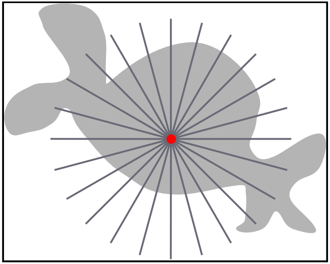
$$D(s) = \max_{\forall(s,t)\in\mathcal{A}_1} \{\min\{\Delta(s,t), |\mathcal{S}_i| - \Delta(s,t)\}\}.$$

We go anti-clockwise, when $\Delta^*(s,t) > |\mathcal{S}_i| - \Delta^*(s,t)$, and clockwise in the opposite case.

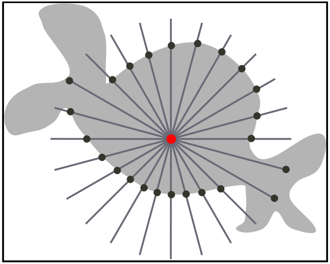


Tensor scale computation

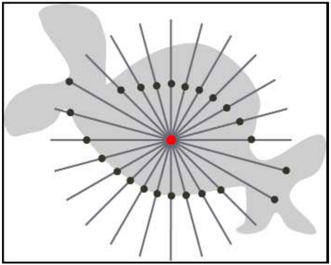
As proposed by Saha [6], the tensor scale at s may be computed by tracing sample lines, finding edge points in each line, and fitting the largest ellipse through these points.



As proposed by Saha [6], the tensor scale at s may be computed by tracing sample lines, finding edge points in each line, and fitting the largest ellipse through these points.

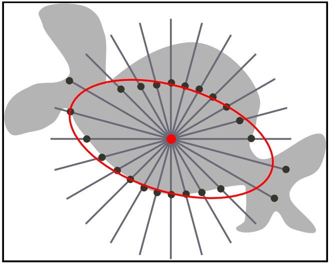


As proposed by Saha [6], the tensor scale at s may be computed by tracing sample lines, finding edge points in each line, and fitting the largest ellipse through these points.



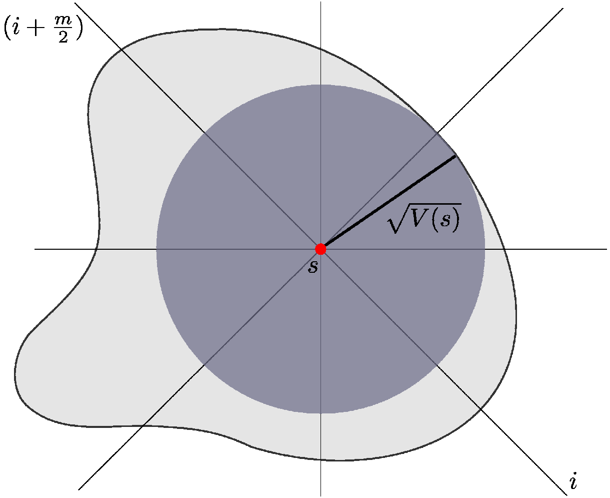
Tensor scale computation

As proposed by Saha [6], the tensor scale at s may be computed by tracing sample lines, finding edge points in each line, and fitting the largest ellipse through these points.



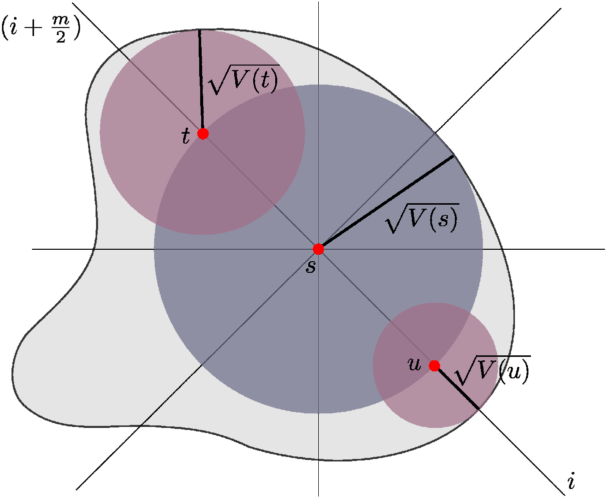
Tensor scale computation

The Euclidean IFT speeds up the search for each pair of edge points by exploiting the values in $V(s)$ [3].



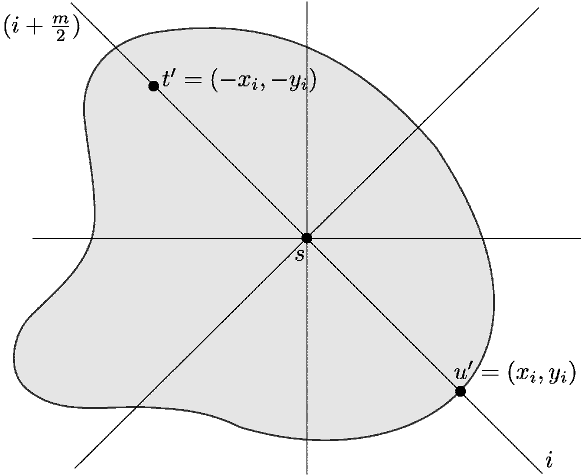
Tensor scale computation

The Euclidean IFT speeds up the search for each pair of edge points by exploiting the values in $V(s)$ [3].



Tensor scale computation

The Euclidean IFT speeds up the search for each pair of edge points by exploiting the values in $V(s)$ [3].



Tensor scale computation

The ellipse orientation is obtained from the value of γ that minimizes function g below.

$$g(\gamma) = \sum_{i=1,2,\ldots,m} [x_{i_\gamma}^2 - y_{i_\gamma}^2]$$

where

The ellipse orientation is obtained from the value of γ that minimizes function g below.

$$g(\gamma) = \sum_{i=1,2,\ldots,m} [x_{i_\gamma}^2 - y_{i_\gamma}^2]$$

where

- m is the number of sample lines,

Tensor scale computation

The ellipse orientation is obtained from the value of γ that minimizes function g below.

$$g(\gamma) = \sum_{i=1,2,\ldots,m} [x_{i_\gamma}^2 - y_{i_\gamma}^2]$$

where

- m is the number of sample lines,
- $(x_{i_\gamma}, y_{i_\gamma})$ are obtained by rotation using angle γ on the relative coordinates (x_i, y_i) of the edge points with respect $s = (x_s, y_s)$.

$$\begin{aligned} x_{i_\gamma} &= x_i \cos(\gamma) - y_i \sin(\gamma) \\ y_{i_\gamma} &= x_i \sin(\gamma) + y_i \cos(\gamma) \end{aligned}$$

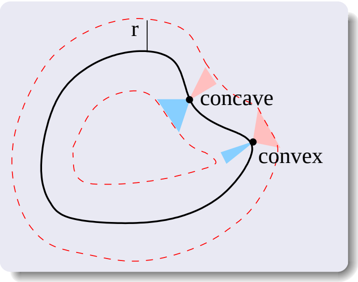
Organization of the lecture

- Muliscale skeletonization and SKIZ.
- Contour and skeleton saliences.
- Tensor scale computation.
- Shape description from these representations.
- Combining multiple descriptors.

Shape description

A descriptor is a pair (v, d), where

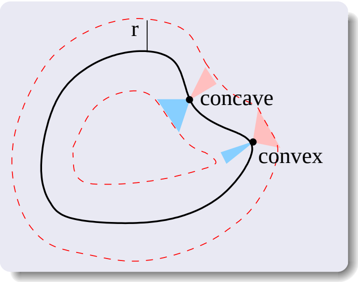
- v is a feature extraction function, which assigns a vector $\vec{s}$ to any sample s (shape, image, spel), and
- d is a distance function between samples s and t in the feature space (e.g., $d(s,t) = \|\vec{t} - \vec{s}\|$).



Feature vectors may represent

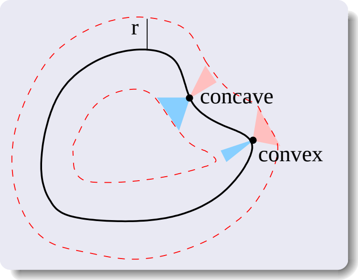
- a multiscale fractal dimension [2] computed from the distance map V.

Shape description



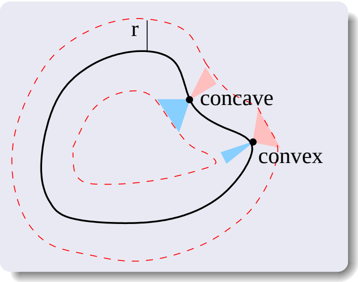
Feature vectors may represent

- a multiscale fractal dimension [2] computed from the distance map V.
- area (salience value) of the largest influence zone [2] for each convex and concave point obtained from the label map L_p.



Feature vectors may represent

- a multiscale fractal dimension [2] computed from the distance map V.
- area (salience value) of the largest influence zone [2] for each convex and concave point obtained from the label map L_p.
- salience values [4] of contour segments obtained from the label map L_p.



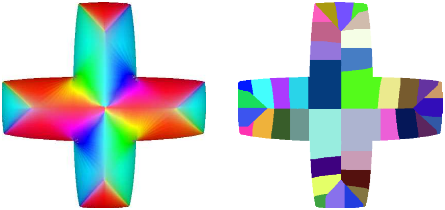
Feature vectors may represent

- a multiscale fractal dimension [2] computed from the distance map V.
- area (salience value) of the largest influence zone [2] for each convex and concave point obtained from the label map L_p.
- salience values [4] of contour segments obtained from the label map L_p.

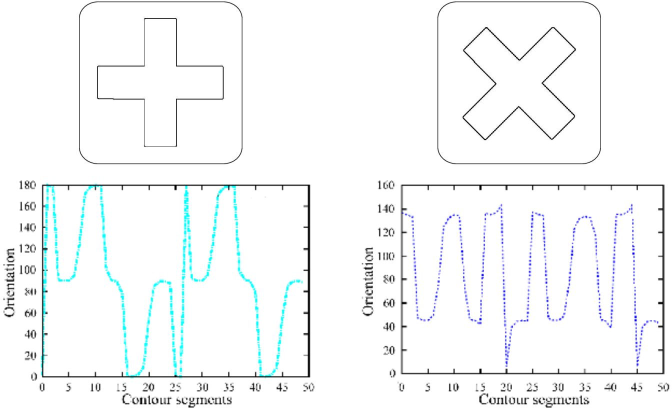
In most cases, a specific distance function is required to take into account possible shape rotation and scaling.

For example, we have divided a contour into a fixed number of segments and assigned to each segment the weighted angular mean of the orientation θ_i at each pixel s in the influence zone of that segment [3]. The anisotropy α_i of s is the weight.

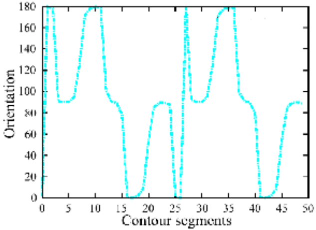
$$\bar{\theta} = arctan\left(\frac{\sum_{i=1}^{n} \alpha_i * sin(2\theta_i)}{\sum_{i=1}^{n} \alpha_i * cos(2\theta_i)}\right)$$

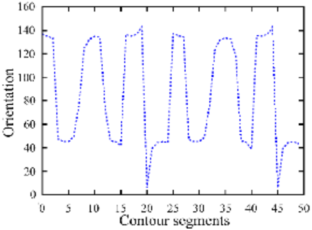


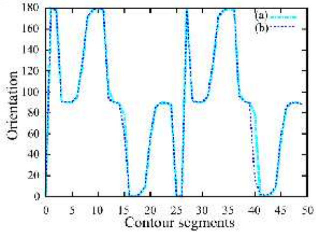
Feature vectors for a shape in different positions.



Matching between the feature vectors for distance computation.





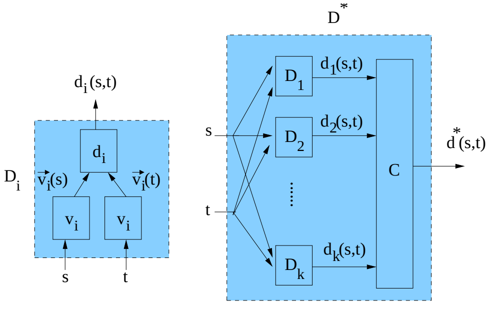


Combining multiple descriptors

- Let $\mathbf{\Delta} = \{D_1, D_2, \ldots, D_k\}$ be a collection of descriptors $D_i = (v_i, d_i)$, $i = 1, 2, \ldots, k$, needed to handle different shape, color and texture properties.

Combining multiple descriptors

- Let $\mathbf{\Delta} = \{D_1, D_2, \ldots, D_k\}$ be a collection of descriptors $D_i = (v_i, d_i)$, $i = 1, 2, \ldots, k$, needed to handle different shape, color and texture properties.
- The combination C of their distance functions is an application-dependent optimization problem which creates a composite descriptor $D^* = (\mathbf{\Delta}, C)$.



(a) (b)

We have found C by genetic programming [5].

- The Euclidean IFT was exploited to derive several shape representations.

Conclusion

- The Euclidean IFT was exploited to derive several shape representations.
- These representations involved multiscale skeletons, salience points, and tensor scale.

Conclusion

- The Euclidean IFT was exploited to derive several shape representations.
- These representations involved multiscale skeletons, salience points, and tensor scale.
- Given that contour saliences are estimated from skeleton saliences, the multiscale skeletons can also obtain contour saliences in different scales.

Conclusion

- The Euclidean IFT was exploited to derive several shape representations.
- These representations involved multiscale skeletons, salience points, and tensor scale.
- Given that contour saliences are estimated from skeleton saliences, the multiscale skeletons can also obtain contour saliences in different scales.
- There are many ways to create shape descriptors from those representations and combine their distance functions into a composite descriptor.

[1] A.X. Falcão, L.F. Costa, and B.S. da Cunha.
Multiscale skeletons by image foresting transform and its applications to neuromorphometry.
Pattern Recognition, 35(7):1571–1582, Apr 2002.

[2] R.S. Torres, A.X. Falcão, and L.F. Costa.
A graph-based approach for multiscale shape analysis.
Pattern Recognition, 37(6):1163–1174, 2004.

[3] F.A. Andaló, P.A.V. Miranda, R. da S. Torres, and A.X.Falcão.
Shape feature extraction and description based on tensor scale.
Pattern Recognition, 43(1):26–36, Jan 2010.

[4] R.S. Torres and A.X. Falcão.
Contour salience descriptors for effective image retrieval and analysis.
Image and Vision Computing, 25(1):3–13, Jan 2007.

[5] R.S. Torres, A.X. Falcão, M.A. Gonçalves, J.P. Papa, B. Zhang, W. Fan, and E.A. Fox.

A genetic programming framework for content-based image retrieval.
Pattern Recognition, 42:217–312, Feb 2009.

[6] P.K. Saha.
Tensor Scale: A local morphometric parameter with applications to computer vision and image processing.
Computer Vision and Image Understanding, 99:384–413, 2005.