Skeleton pruning by contour approximation and the integer medial axis transform

uOttawa

University of Ottawa

October 7, 2015

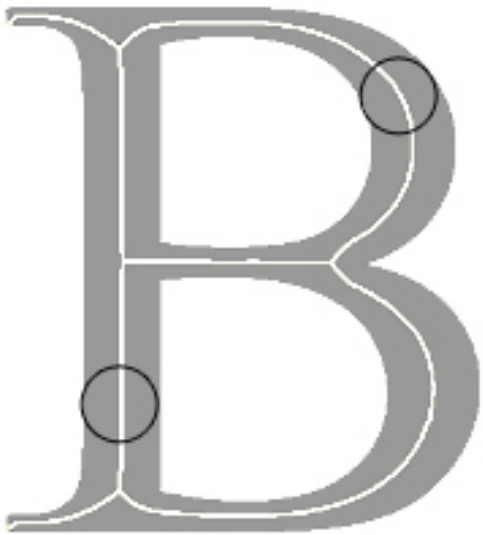
Outline

Medial Axis of a shape (Digital Images)

- The locus of centers of all maximal discs tangent to the boundary in more than one point.
- A set of connected pixels that have at least two closest boundary points.
- Points with more than 2 closest points in the boundary are branching points.
- End points are the extreme pixels of the skeleton.

Blum's definition 1967.

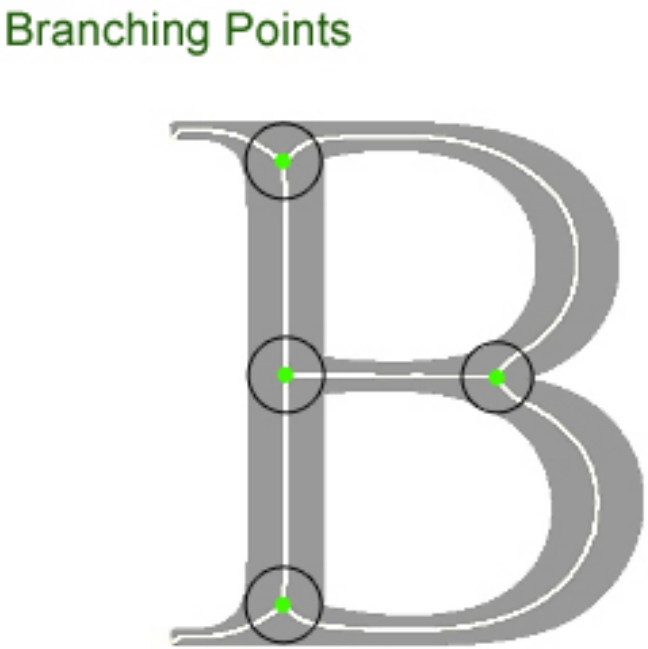
Medial Axis of a shape (Digital Images)



- The locus of centers of all maximal discs tangent to the boundary in more than one point.
- A set of connected pixels that have at least two closest boundary points.
- Points with more than 2 closest points in the boundary are branching points.
- End points are the extreme pixels of the skeleton.

Blum's definition 1967.

Medial Axis of a shape (Digital Images)



- The locus of centers of all maximal discs tangent to the boundary in more than one point.
- A set of connected pixels that have at least two closest boundary points.
- Points with more than 2 closest points in the boundary are branching points.
- End points are the extreme pixels of the skeleton.

Blum's definition 1967.

Medial Axis of a shape (Digital Images)

- The locus of centers of all maximal discs tangent to the boundary in more than one point.
- A set of connected pixels that have at least two closest boundary points.
- Points with more than 2 closest points in the boundary are branching points.
- End points are the extreme pixels of the skeleton.

Blum's definition 1967.

Problem and Objective

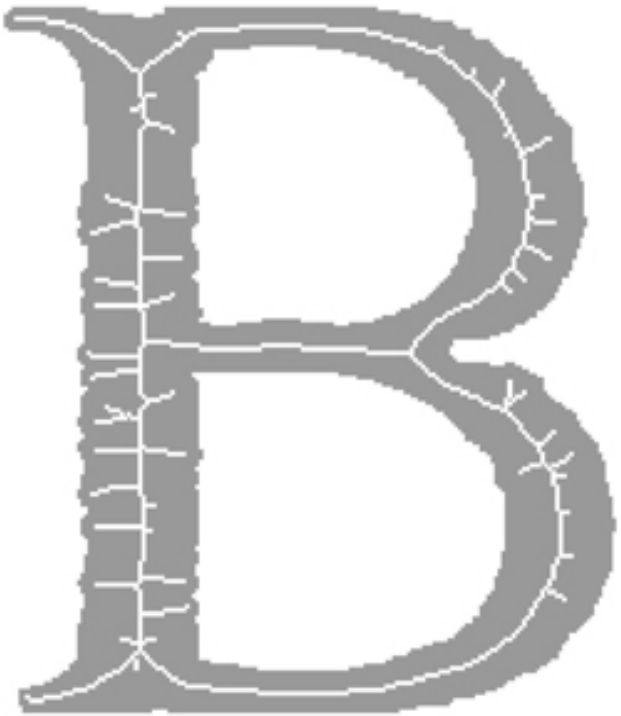
Problem

The digital medium (pixels) and boundary noise create unwanted branches.

Objective

A medial axis subset:

- conserves connectivity.
- removes undesired branches.
- reconstructs the original shape.
- is robust to image transformations.
- computes vectorial medial axis.
- used in real time applications.



Problem and Objective

Problem

The digital medium (pixels) and boundary noise create unwanted branches.

Objective

A medial axis subset:

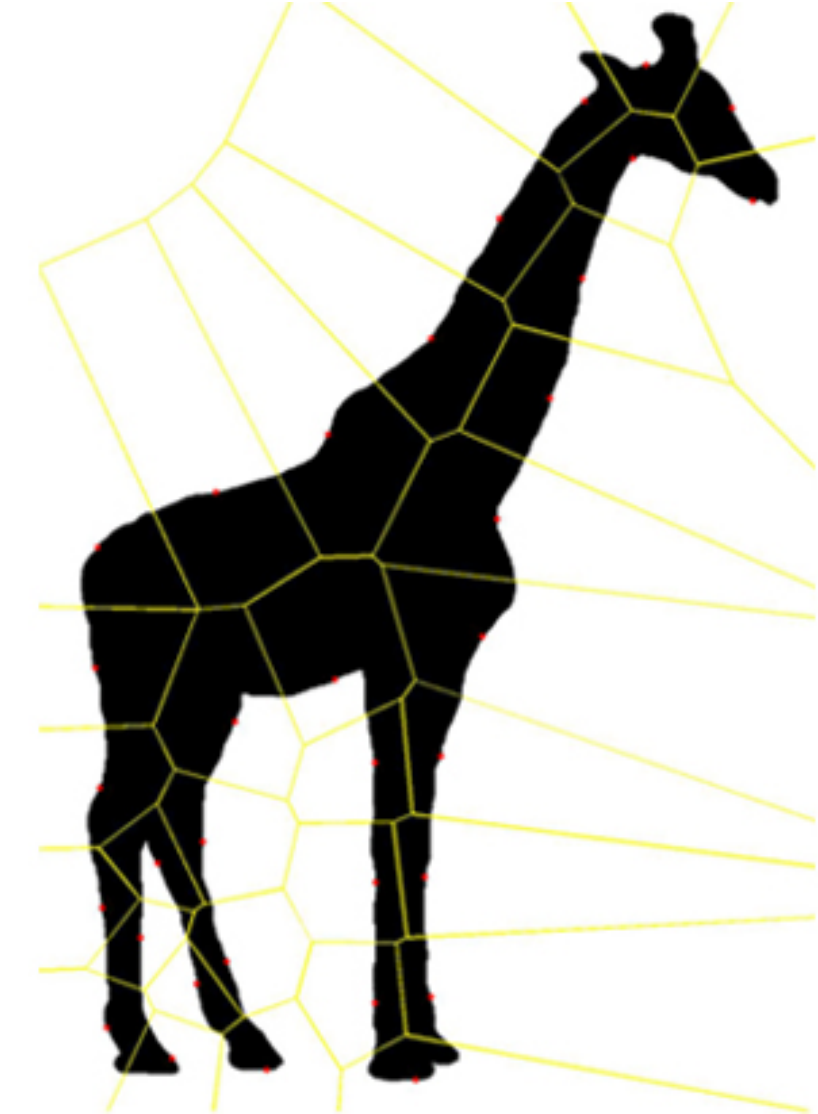
- conserves connectivity.
- removes undesired branches.
- reconstructs the original shape.
- is robust to image transformations.
- computes vectorial medial axis.
- used in real time applications.

Contributions of our solution

Contributions

- Conserves connectivity.
- Outputs a subset of the integer medial axis.
- Fast and easy to implement.

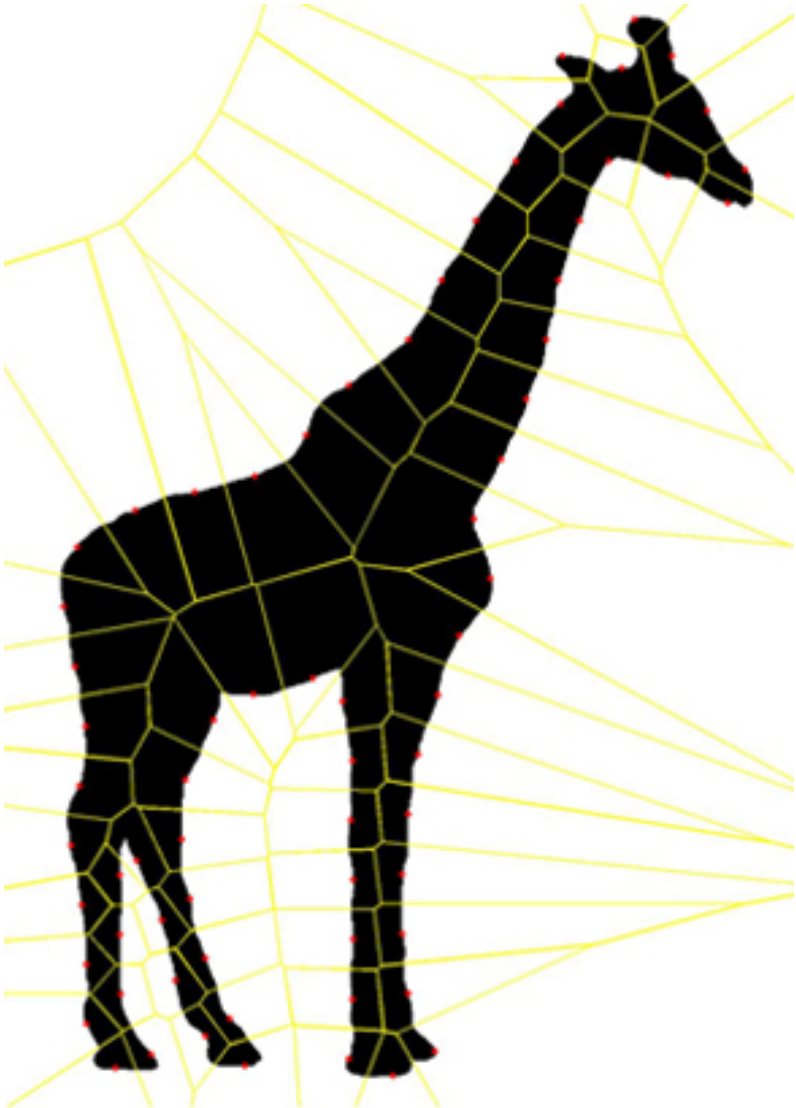
Voronoi diagram.



- Computes voronoi regions of boundary points and pruning.
- As density of boundary points goes to infinity, diagram converges to medial axis.

Orrite-Urunuela C et al. 2004.
X. Bai et. al 2007.
Latecki LJ et. al 2007.
Martnez J et. al 2010.

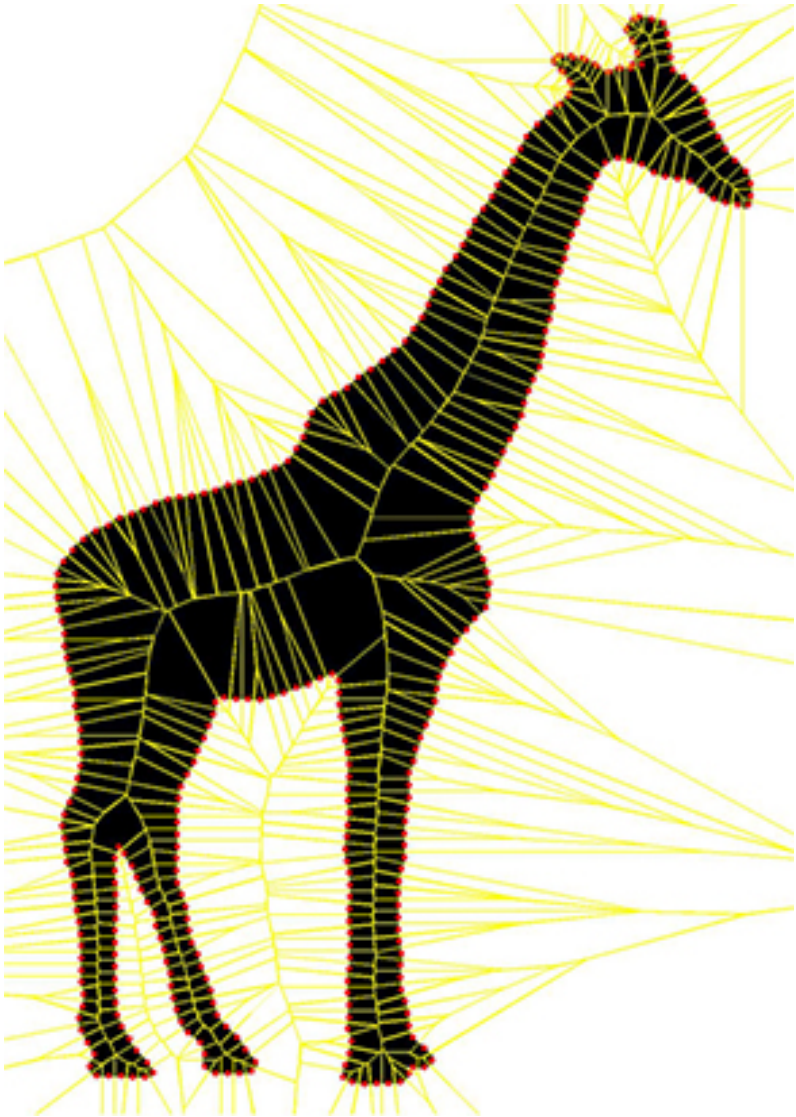
Voronoi diagram.



- Computes voronoi regions of boundary points and pruning.
- As density of boundary points goes to infinity, diagram converges to medial axis.

Orrite-Urunuela C et al. 2004.
X. Bai et. al 2007.
Latecki LJ et. al 2007.
Martnez J et. al 2010.

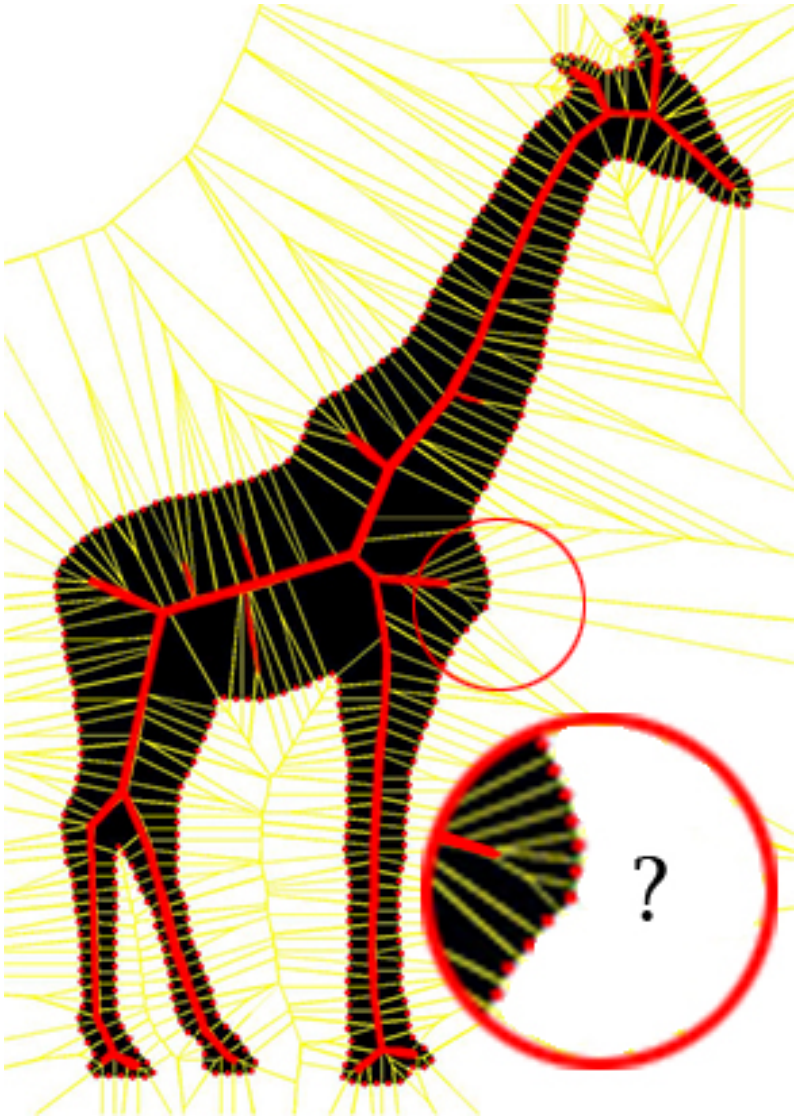
Voronoi diagram.



- Computes voronoi regions of boundary points and pruning.
- As density of boundary points goes to infinity, diagram converges to medial axis.

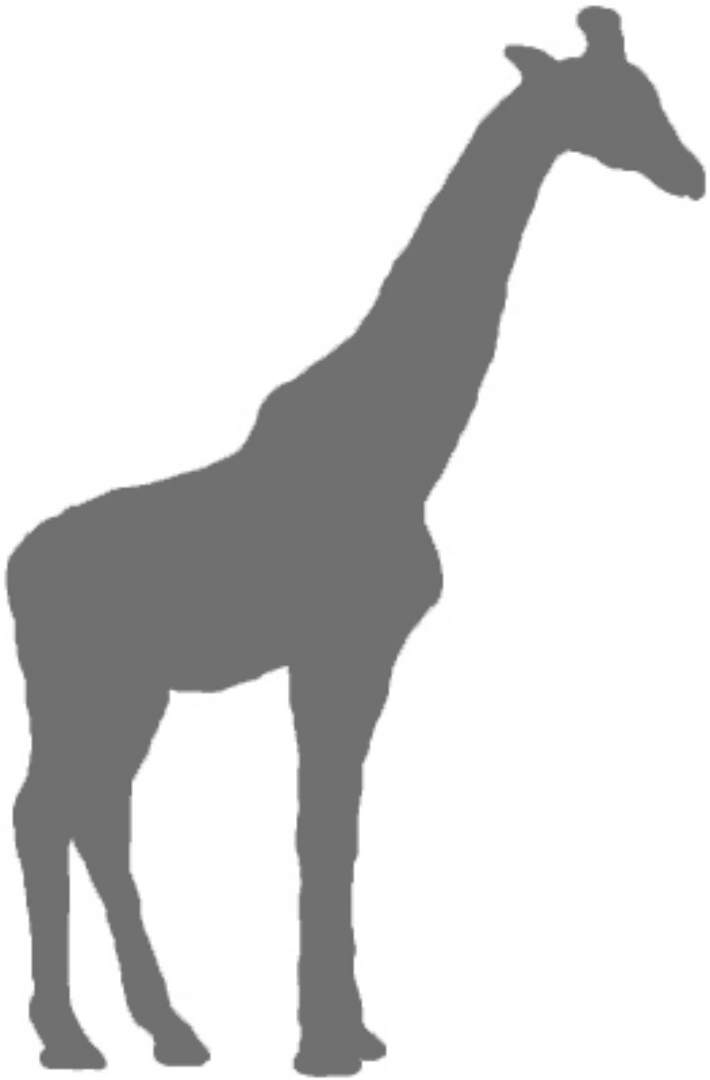
Orrite-Urunuela C et al. 2004.
X. Bai et. al 2007.
Latecki LJ et. al 2007.
Martnez J et. al 2010.

Voronoi's Limitations



- Selecting the correct samples of the boundary pixels.
- Sensitivity to noise.
- Many algorithms in this category are slow and more complicated.
- Needs extra prunning steps.

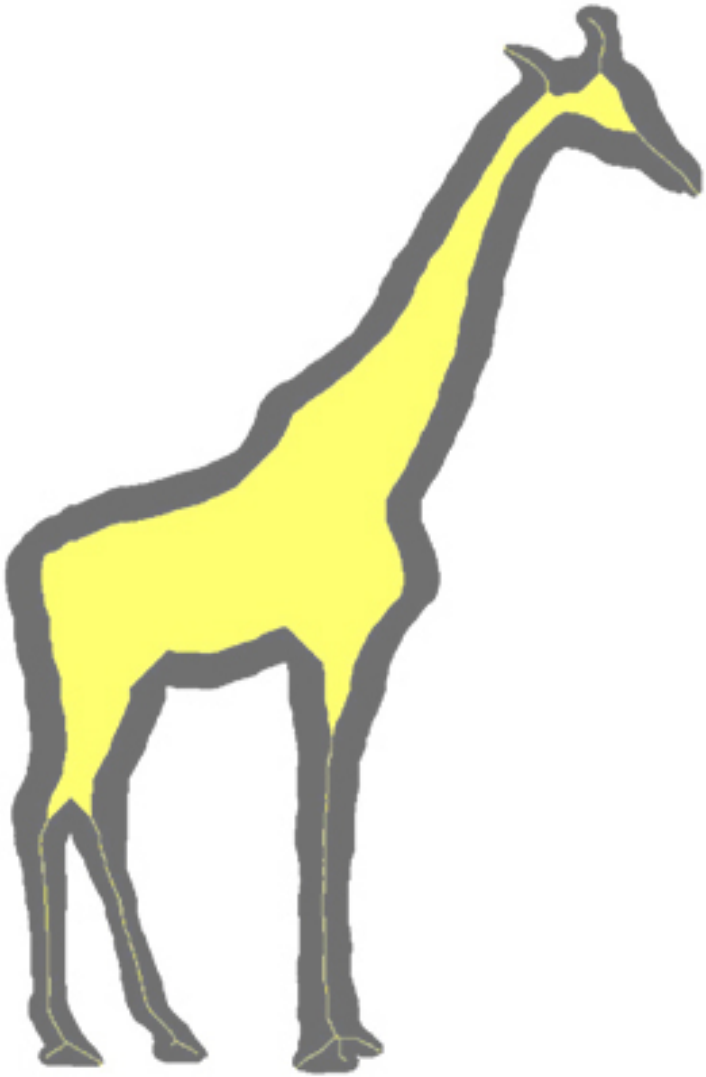
Thinning Solutions



- Peeling pixels from the boundary until no more pixels can be removed.
- Each boundary pixel is checked with their 8 neighbors using a set of rules.

R.M. Haralick et. al 1992.
Lam L et al. 1992.
R. Zhou et. al 1995.
She F et. al 2009.
L. Liu et. al 2011.

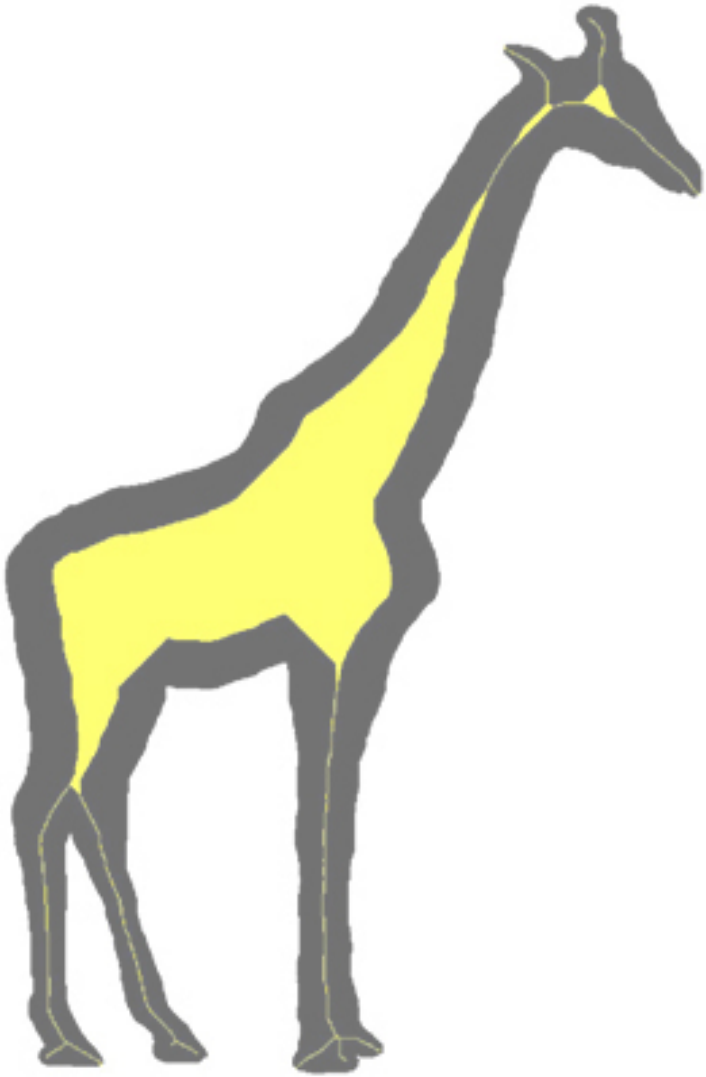
Thinning Solutions



- Peeling pixels from the boundary until no more pixels can be removed.
- Each boundary pixel is checked with their 8 neighbors using a set of rules.

R.M. Haralick et. al 1992.
Lam L et al. 1992.
R. Zhou et. al 1995.
She F et. al 2009.
L. Liu et. al 2011.

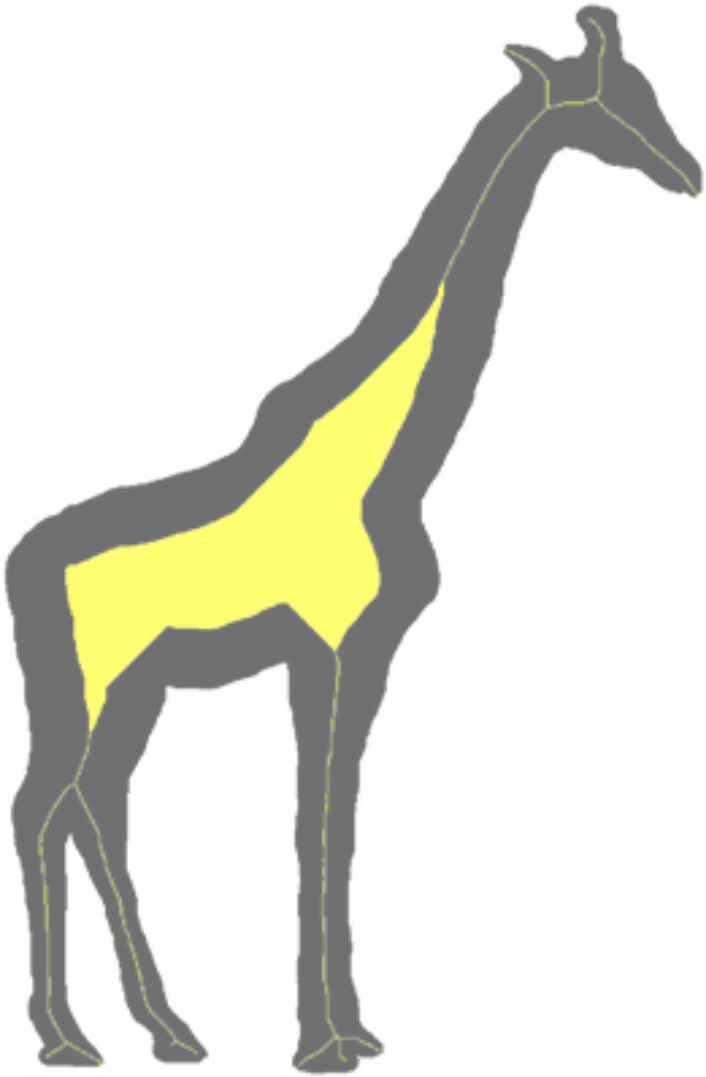
Thinning Solutions



- Peeling pixels from the boundary until no more pixels can be removed.
- Each boundary pixel is checked with their 8 neighbors using a set of rules.

R.M. Haralick et. al 1992.
Lam L et al. 1992.
R. Zhou et. al 1995.
She F et. al 2009.
L. Liu et. al 2011.

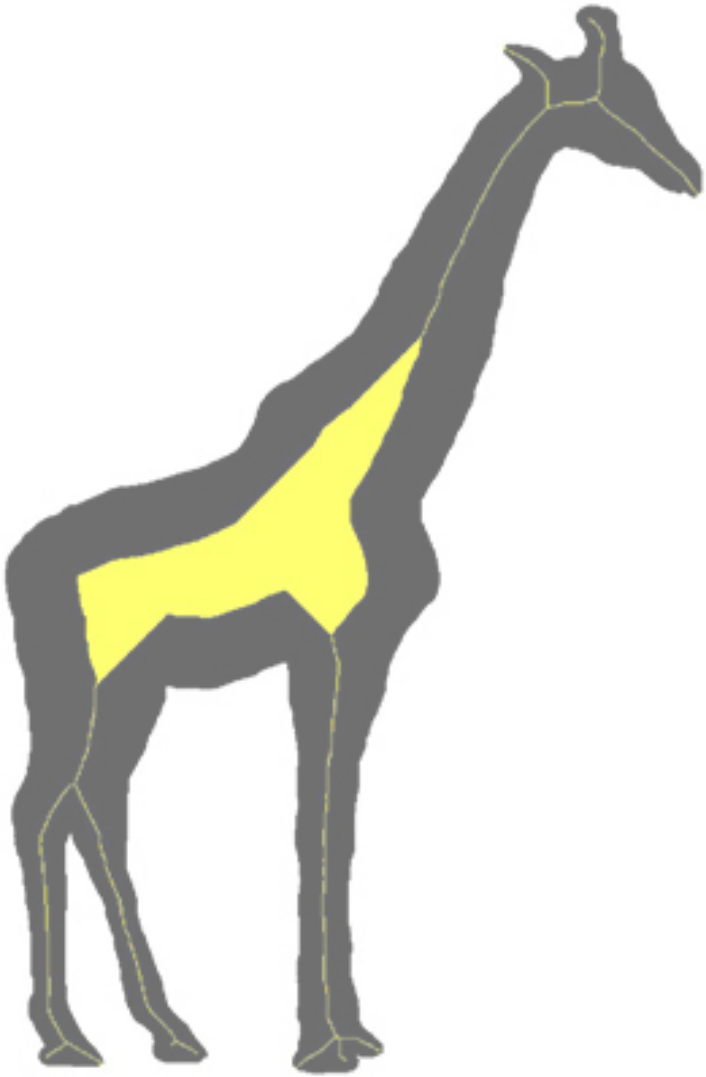
Thinning Solutions



- Peeling pixels from the boundary until no more pixels can be removed.
- Each boundary pixel is checked with their 8 neighbors using a set of rules.

R.M. Haralick et. al 1992.
Lam L et al. 1992.
R. Zhou et. al 1995.
She F et. al 2009.
L. Liu et. al 2011.

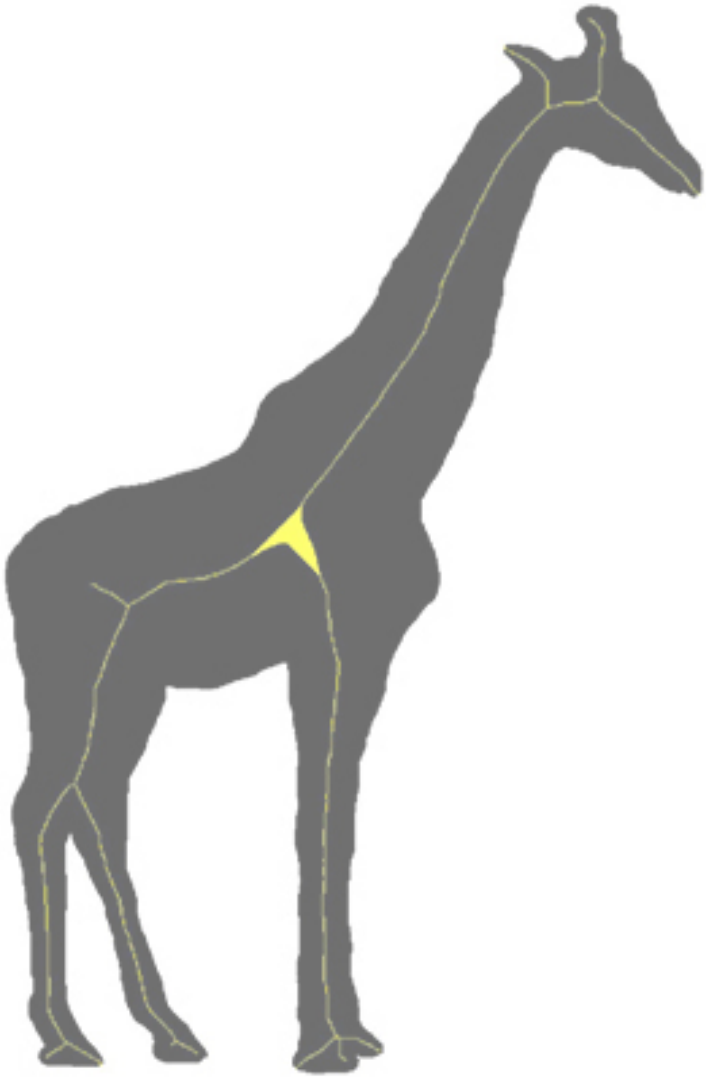
Thinning Solutions



- Peeling pixels from the boundary until no more pixels can be removed.
- Each boundary pixel is checked with their 8 neighbors using a set of rules.

R.M. Haralick et. al 1992.
Lam L et al. 1992.
R. Zhou et. al 1995.
She F et. al 2009.
L. Liu et. al 2011.

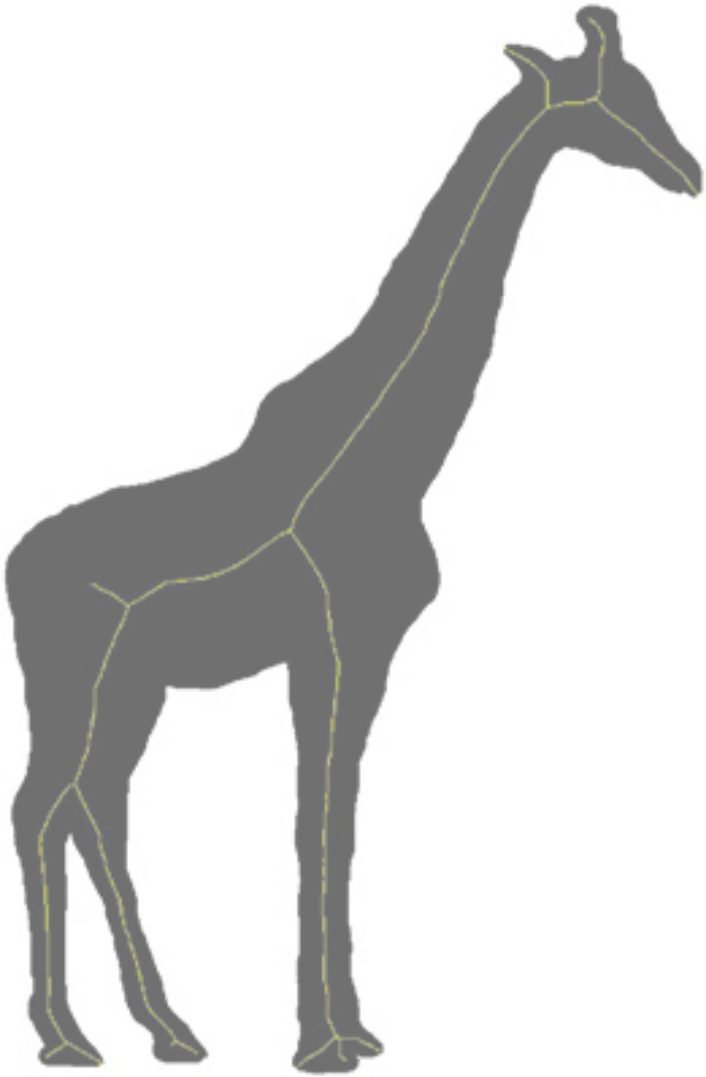
Thinning Solutions



- Peeling pixels from the boundary until no more pixels can be removed.
- Each boundary pixel is checked with their 8 neighbors using a set of rules.

R.M. Haralick et. al 1992.
Lam L et al. 1992.
R. Zhou et. al 1995.
She F et. al 2009.
L. Liu et. al 2011.

Thinning Solutions



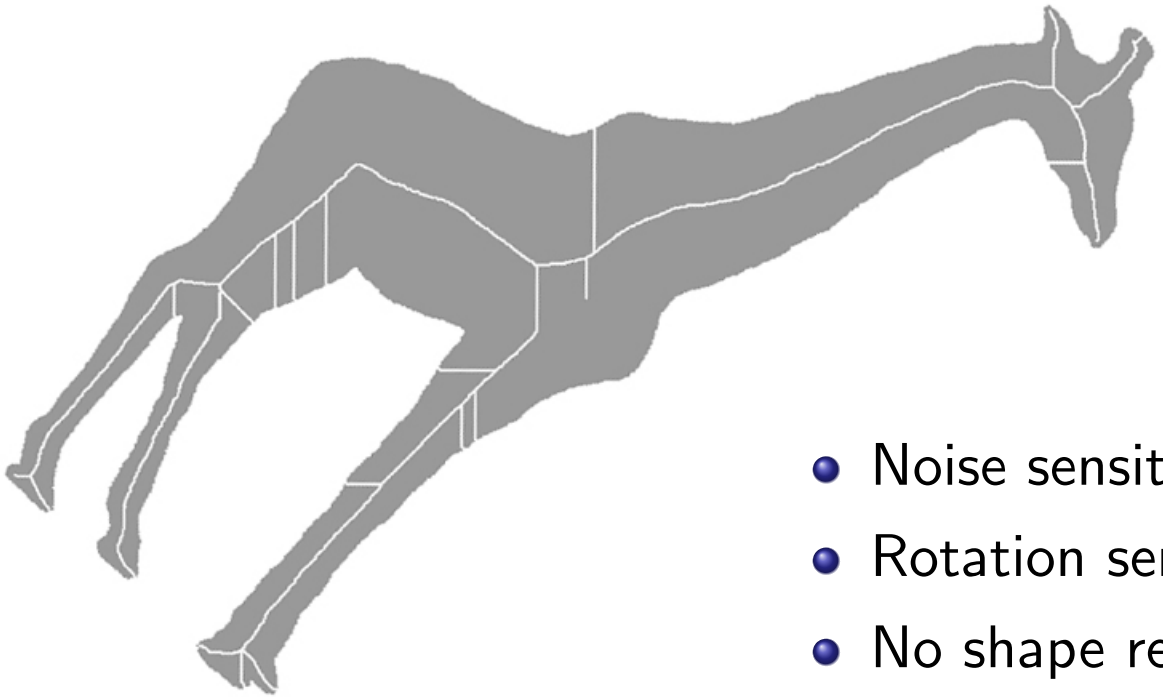
- Peeling pixels from the boundary until no more pixels can be removed.
- Each boundary pixel is checked with their 8 neighbors using a set of rules.

R.M. Haralick et. al 1992.
Lam L et al. 1992.
R. Zhou et. al 1995.
She F et. al 2009.
L. Liu et. al 2011.

Thinning Limitations

- Noise sensitivity.
- Rotation sensitivity.
- No shape reconstruction.
- Fails to compute the medial axis.
- Time consuming.

Thinning Limitations

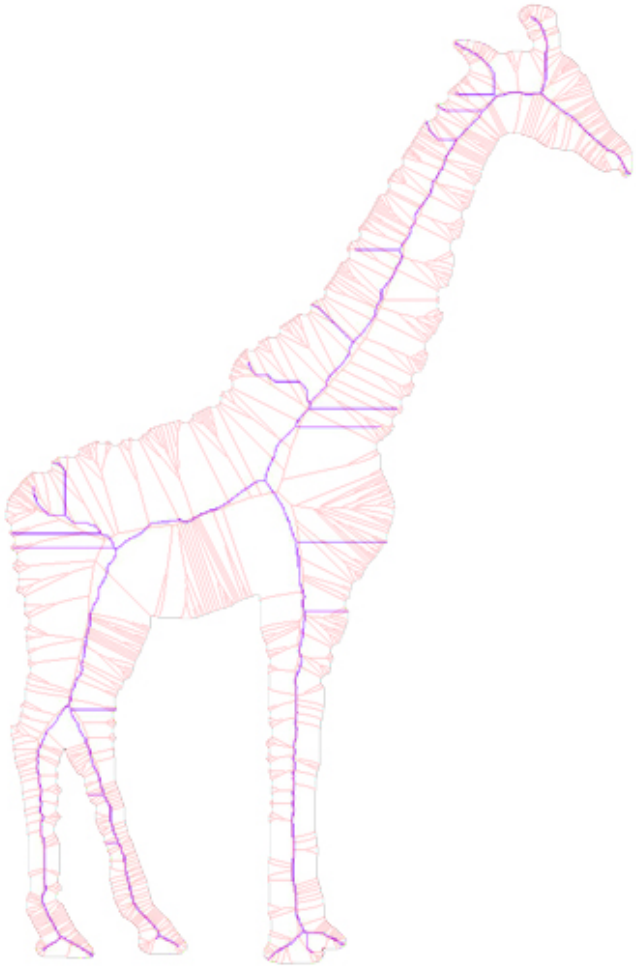


- Noise sensitivity.
- Rotation sensitivity.
- No shape reconstruction.
- Fails to compute the medial axis.
- Time consuming.

Thinning Limitations

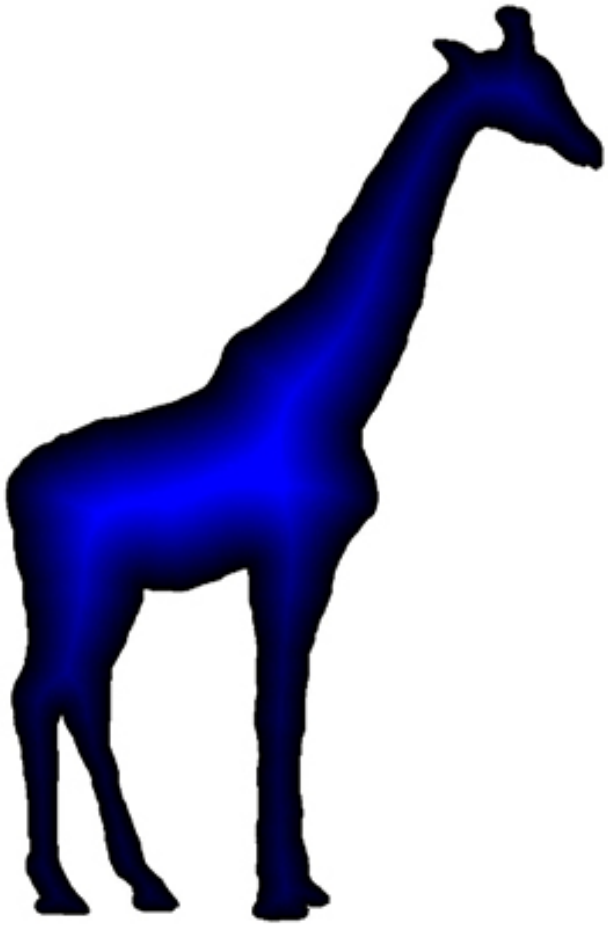
- Noise sensitivity.
- Rotation sensitivity.
- No shape reconstruction.
- Fails to compute the medial axis.
- Time consuming.

Thinning Limitations



- Noise sensitivity.
- Rotation sensitivity.
- No shape reconstruction.
- Fails to compute the medial axis.
- Time consuming.

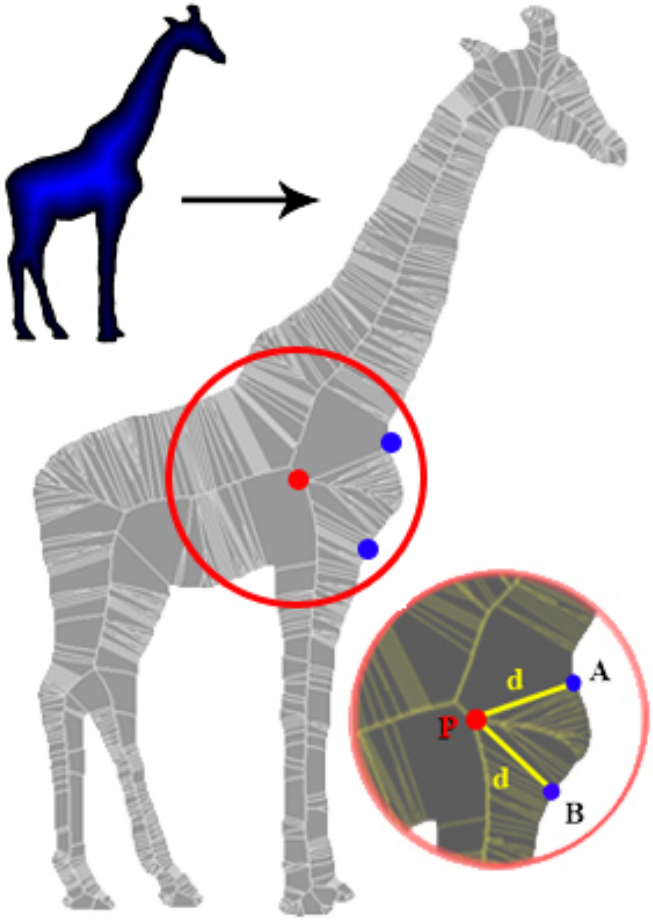
Feature Transform & Distance Transform



- The feature transform computes the closest boundary point for each pixel.
- The distance transform computes the distance to the nearest boundary pixel.
- Points with at least two closest points (same distance to boundary) are selected as medial axis points.

R.M. Haralick and L.G. Shapiro 1992.
C. Arcelli and G.S. di Baja 1993.
G.S. Di Baja and E. Thiel 1996.
X. Bai et al. 2007.
W.H. Hesselink and J.B. Roerdink 2008.

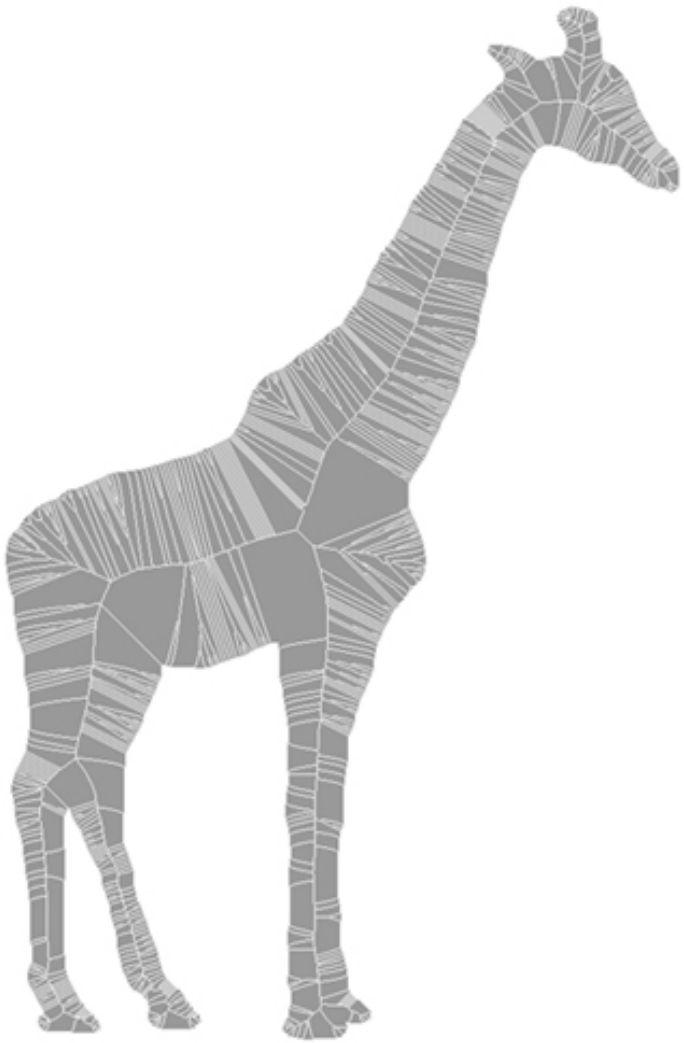
Feature Transform & Distance Transform



- The feature transform computes the closest boundary point for each pixel.
- The distance transform computes the distance to the nearest boundary pixel.
- Points with at least two closest points (same distance to boundary) are selected as medial axis points.

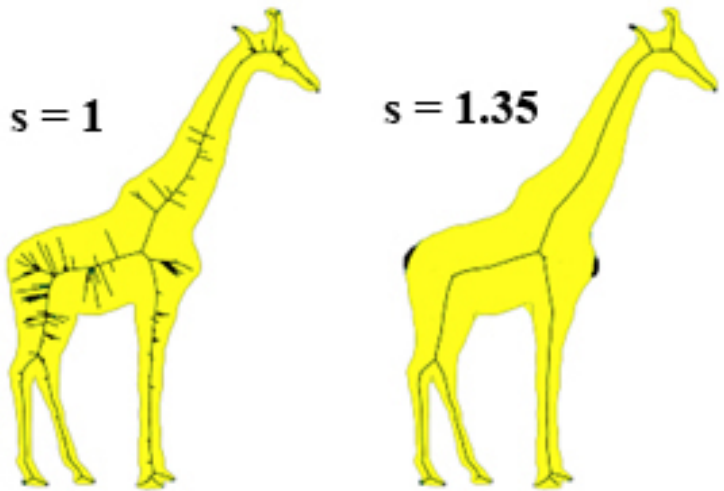
R.M. Haralick and L.G. Shapiro 1992.
C. Arcelli and G.S. di Baja 1993.
G.S. Di Baja and E. Thiel 1996.
X. Bai et al. 2007.
W.H. Hesselink and J.B. Roerdink 2008.

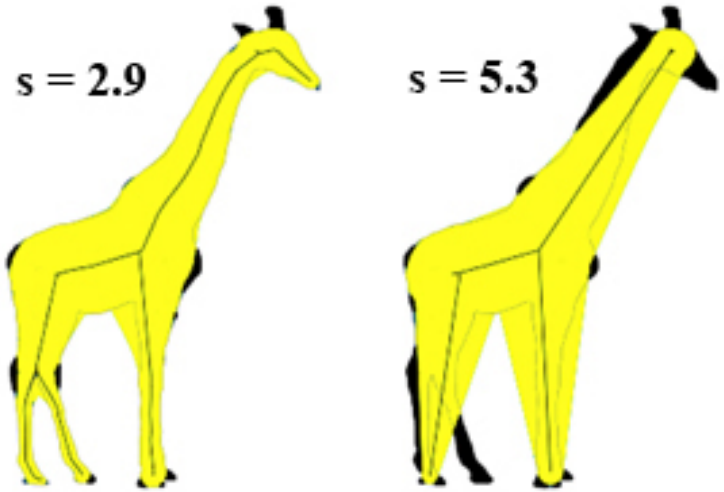
Feature Transform & Distance Transform Limitations



- Noise sensitivity.
- Needs extra pruning.
- Parameter selection associated to the pruning technique.

Pre-processing the shape's boundary



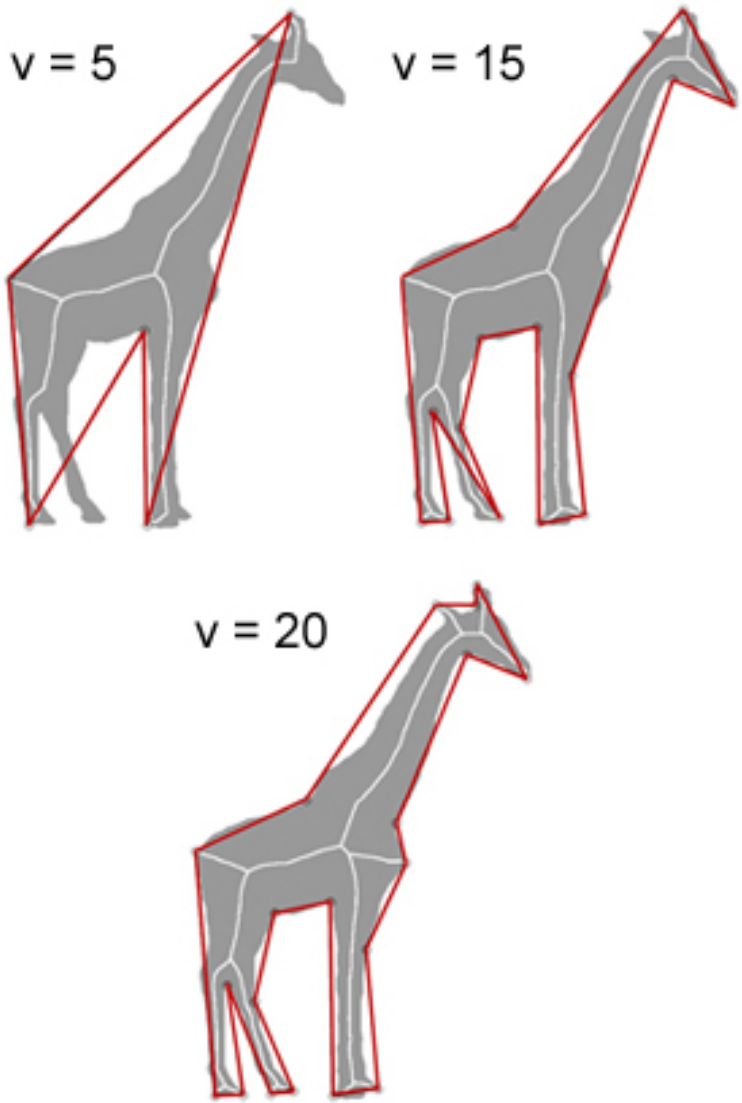


- Uses a noisy pre-computed medial axis.
- Grows the original shape by scaling the radii of the medial disks and then removing branches
- Creates simpler skeleton representation.

- Parameter selection?
- Shift skeleton from the medial axis.
- Shape holes are occluded when the shape is grown.

The scale axis transform. Giesen J. et al. 2008.

Pre-processing the shape's boundary

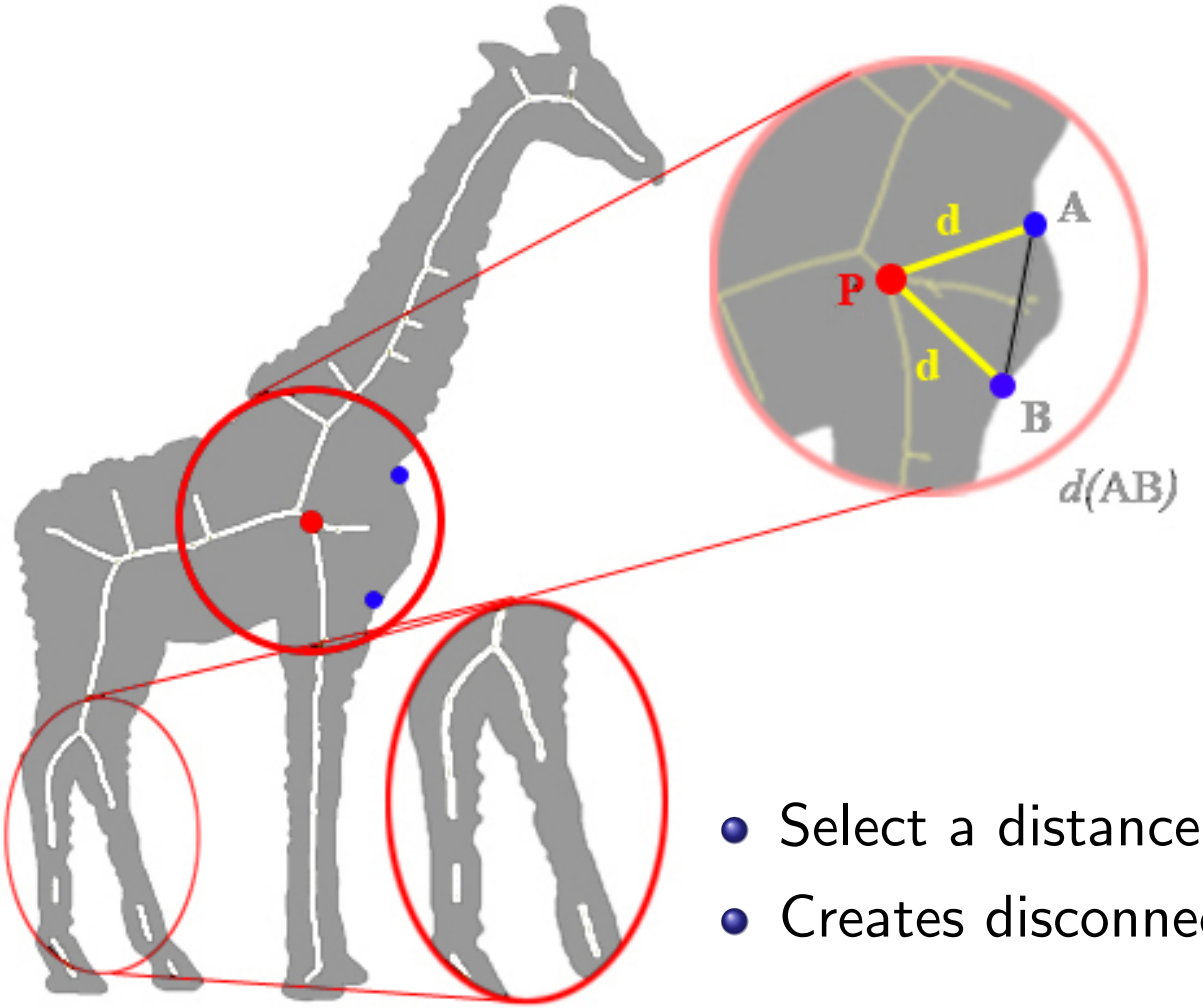


- Simplify the shape's boundary using a Discrete Evolution Approach.
- Uses the approximation to prune pre-computed medial axis.
- Uses the intuition of shape sides.

- Parameter selection?
- A priori knowledge of the shape to prune.
- It is slow.

Skeleton pruning with discrete curve evolution. X. Bai. et al. 2007.

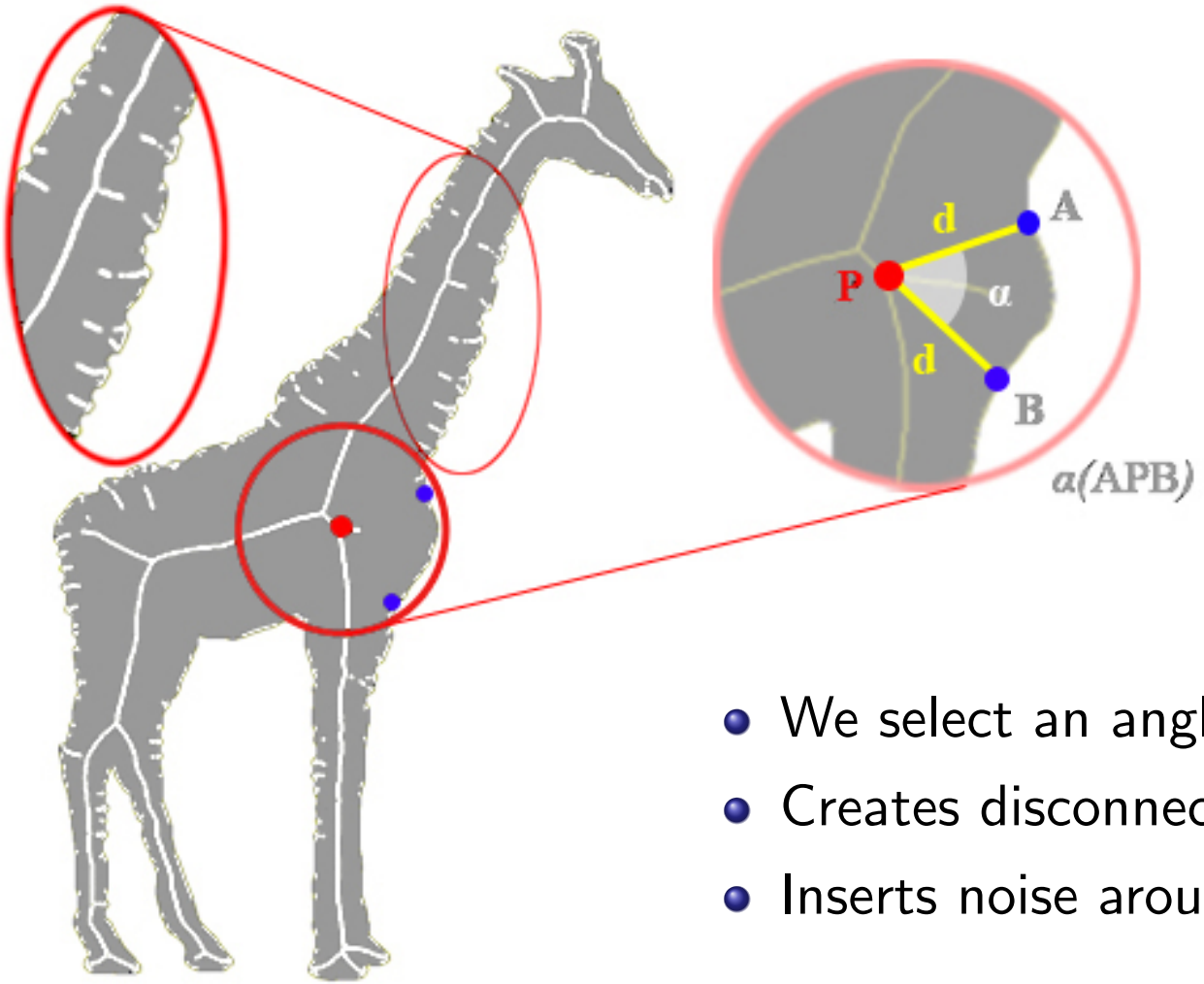
Remove unwanted points based on classification criteria



- Select a distance threshold value T. ???
- Creates disconnectivity.

Distance Prunning.
W.H. Hesselink and J.B. Roerdink 2007.

Remove unwanted points based on classification criteria



- We select an angle threshold value α. ???
- Creates disconnectivity.
- Inserts noise around the boundary.

Angular Pruning.
W.H. Hesselink and J.B. Roerdink 2007.

Algorithm Outline

Four Steps Algorithm

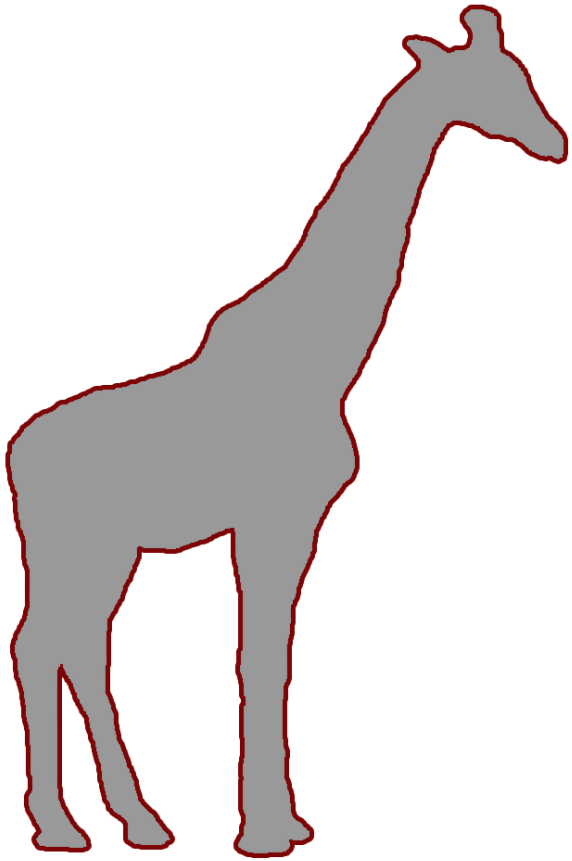
- Contour Approximation (e.g.: piecewise linear/cubic).
- Integer Medial Axis.
- Pruning.
- Skeleton vectorization.

Time Complexity

$$O(kn+N)$$

k: number of knots, n: contour length, N: image size

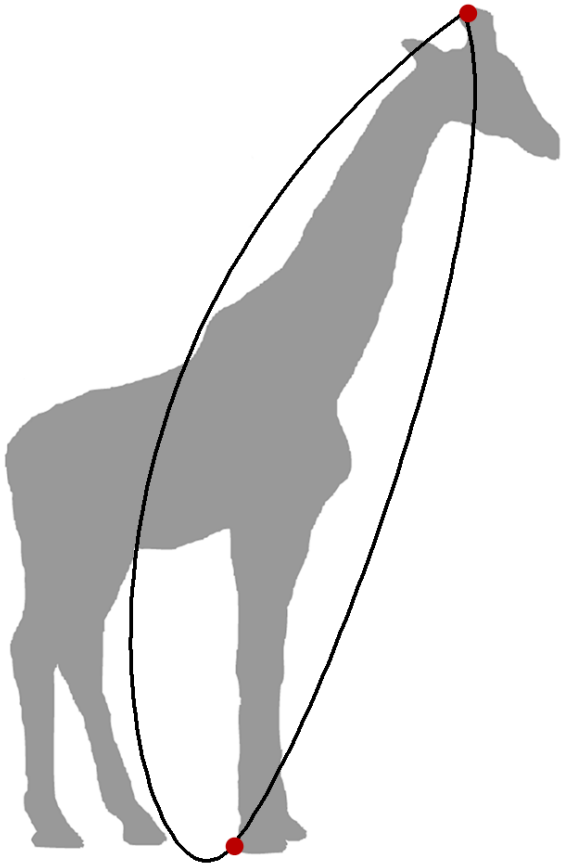
1. Contour Approximation of the boundary



- Select a threshold value T (distance from boundary to curve).
- Contour approximation using simple curves (e.g.: cubic or line segments).
- Assigns each boundary pixel to its curve.

An iterative procedure for the polygonal approximation of plane curves.
U. Ramer 1972
Time complexity: O(kn)
k: number of knots
n: contour length

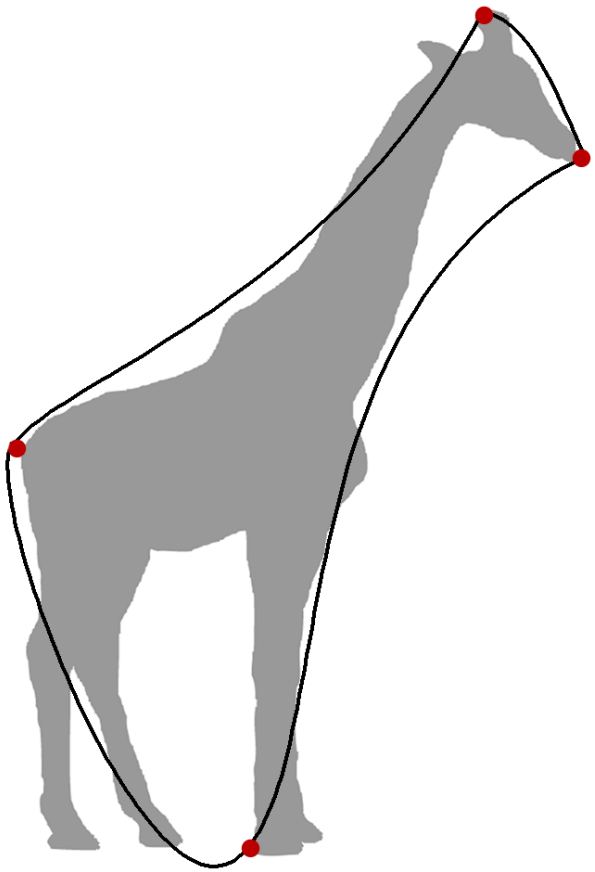
1. Contour Approximation of the boundary



- Select a threshold value T (distance from boundary to curve).
- Contour approximation using simple curves (e.g.: cubic or line segments).
- Assigns each boundary pixel to its curve.

An iterative procedure for the polygonal approximation of plane curves.
U. Ramer 1972
Time complexity: O(kn)
k: number of knots
n: contour length

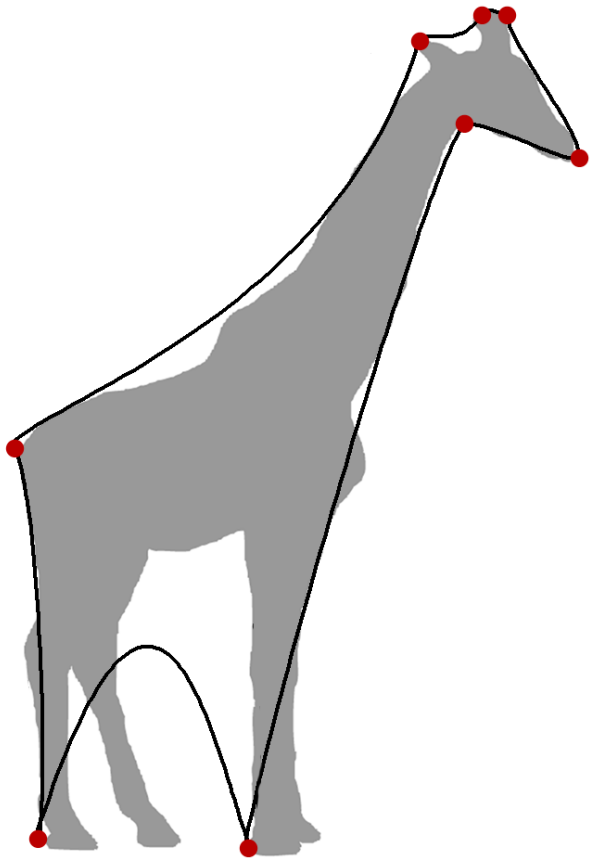
1. Contour Approximation of the boundary



- Select a threshold value T (distance from boundary to curve).
- Contour approximation using simple curves (e.g.: cubic or line segments).
- Assigns each boundary pixel to its curve.

An iterative procedure for the polygonal approximation of plane curves.
U. Ramer 1972
Time complexity: O(kn)
k: number of knots
n: contour length

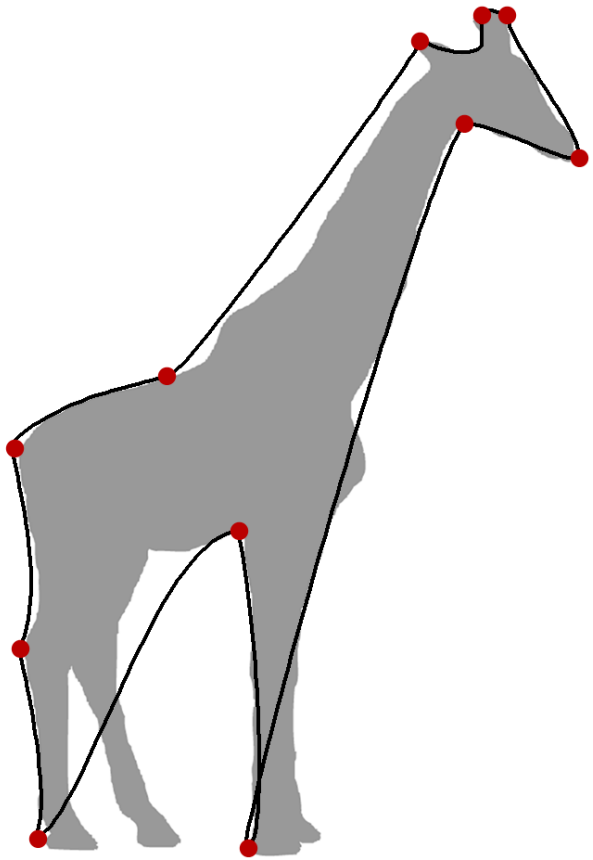
1. Contour Approximation of the boundary



- Select a threshold value T (distance from boundary to curve).
- Contour approximation using simple curves (e.g.: cubic or line segments).
- Assigns each boundary pixel to its curve.

An iterative procedure for the polygonal approximation of plane curves.
U. Ramer 1972
Time complexity: O(kn)
k: number of knots
n: contour length

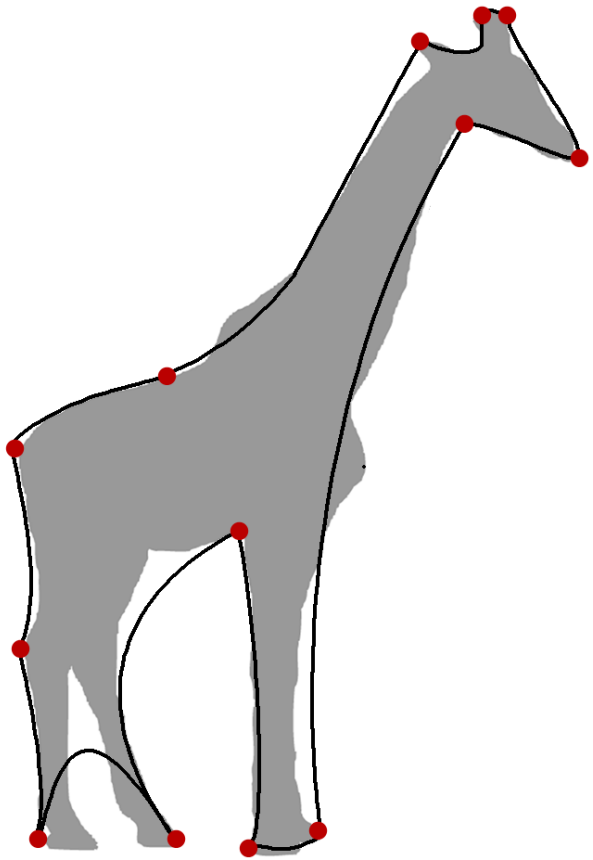
1. Contour Approximation of the boundary



- Select a threshold value T (distance from boundary to curve).
- Contour approximation using simple curves (e.g.: cubic or line segments).
- Assigns each boundary pixel to its curve.

An iterative procedure for the polygonal approximation of plane curves.
U. Ramer 1972
Time complexity: O(kn)
k: number of knots
n: contour length

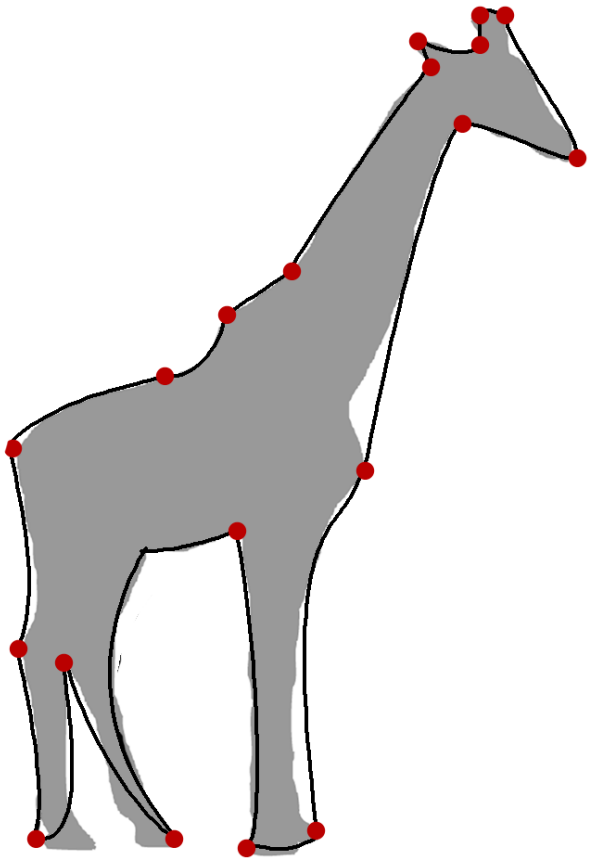
1. Contour Approximation of the boundary



- Select a threshold value T (distance from boundary to curve).
- Contour approximation using simple curves (e.g.: cubic or line segments).
- Assigns each boundary pixel to its curve.

An iterative procedure for the polygonal approximation of plane curves.
U. Ramer 1972
Time complexity: O(kn)
k: number of knots
n: contour length

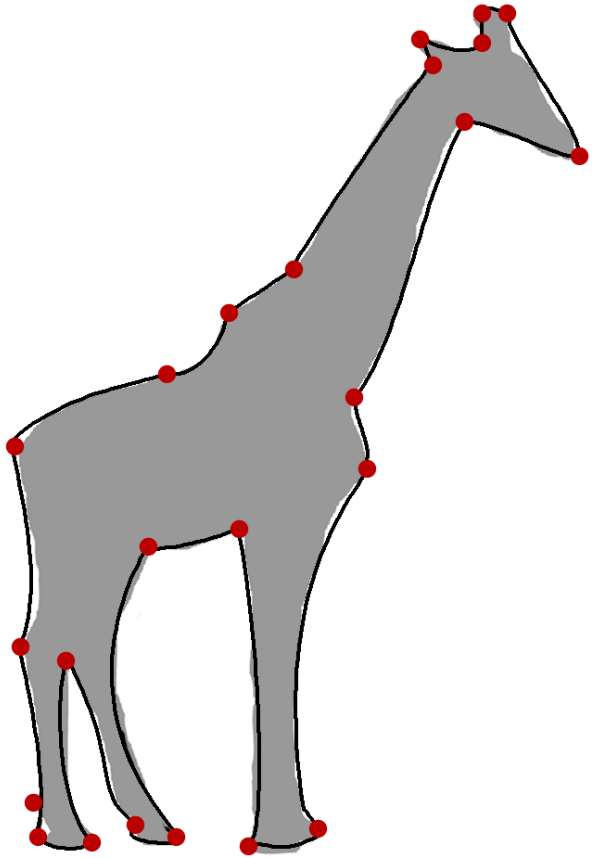
1. Contour Approximation of the boundary



- Select a threshold value T (distance from boundary to curve).
- Contour approximation using simple curves (e.g.: cubic or line segments).
- Assigns each boundary pixel to its curve.

An iterative procedure for the polygonal approximation of plane curves.
U. Ramer 1972
Time complexity: O(kn)
k: number of knots
n: contour length

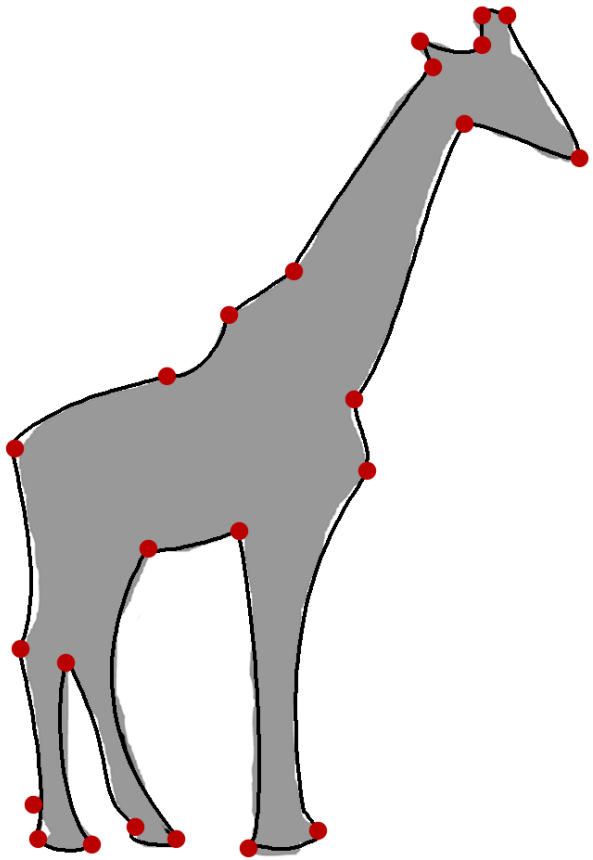
1. Contour Approximation of the boundary



- Select a threshold value T (distance from boundary to curve).
- Contour approximation using simple curves (e.g.: cubic or line segments).
- Assigns each boundary pixel to its curve.

An iterative procedure for the polygonal approximation of plane curves.
U. Ramer 1972
Time complexity: O(kn)
k: number of knots
n: contour length

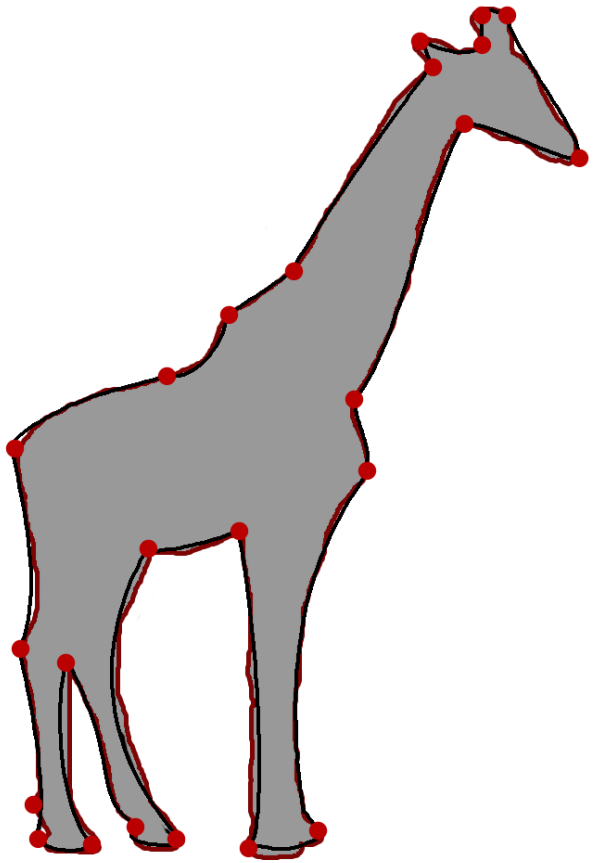
1. Contour Approximation of the boundary



- Select a threshold value T (distance from boundary to curve).
- Contour approximation using simple curves (e.g.: cubic or line segments).
- Assigns each boundary pixel to its curve.

An iterative procedure for the polygonal approximation of plane curves.
U. Ramer 1972
Time complexity: O(kn)
k: number of knots
n: contour length

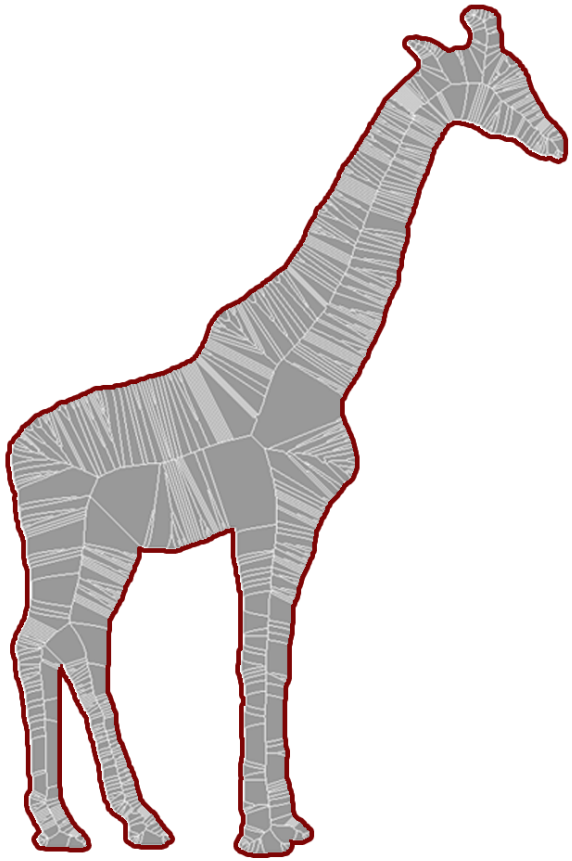
1. Contour Approximation of the boundary



- Select a threshold value T (distance from boundary to curve).
- Contour approximation using simple curves (e.g.: cubic or line segments).
- Assigns each boundary pixel to its curve.

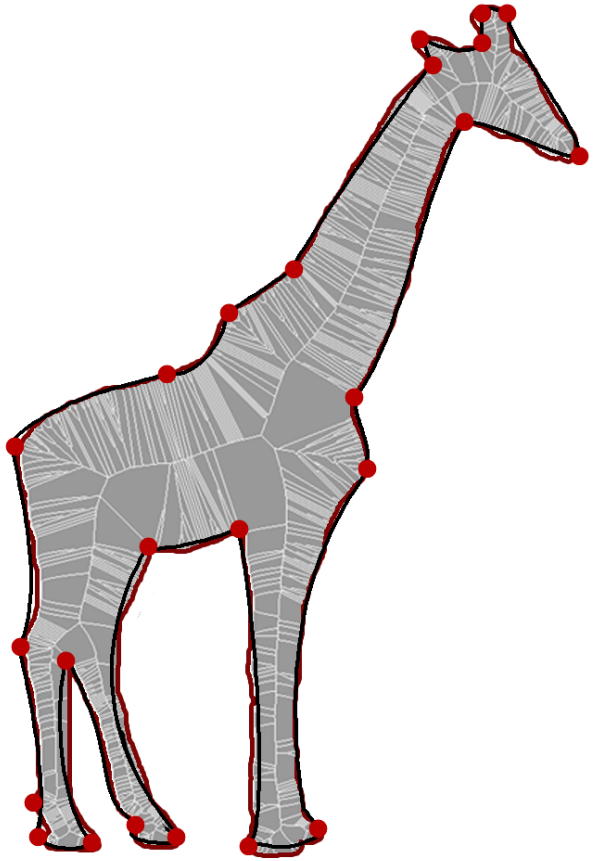
An iterative procedure for the polygonal approximation of plane curves.
U. Ramer 1972
Time complexity: O(kn)
k: number of knots
n: contour length

2. Compute and prune Integer Medial Axis



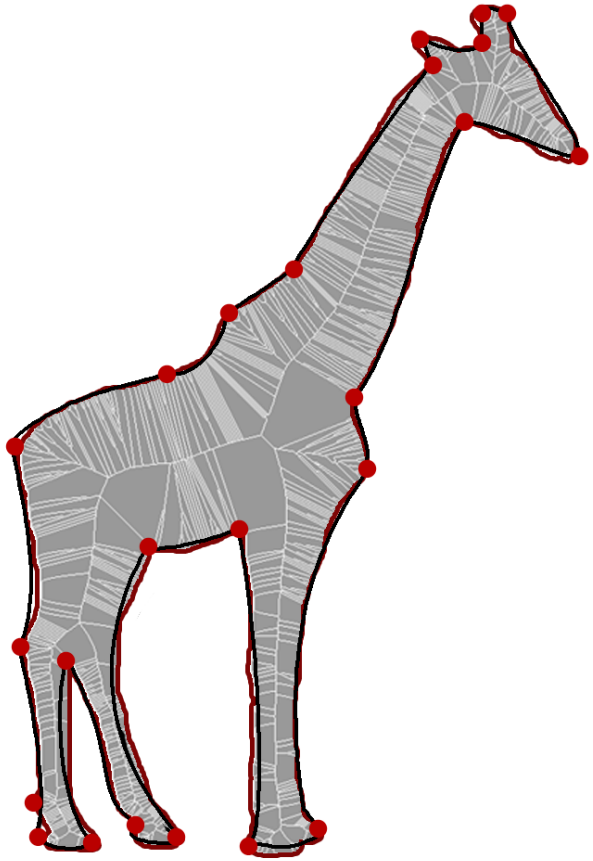
Euclidean skeletons of digital image and volume data in linear time by the integer medial axis transform
W.H. Hesselink and J.B. Roerdink 2007
Time complexity: O(N)
N: image size

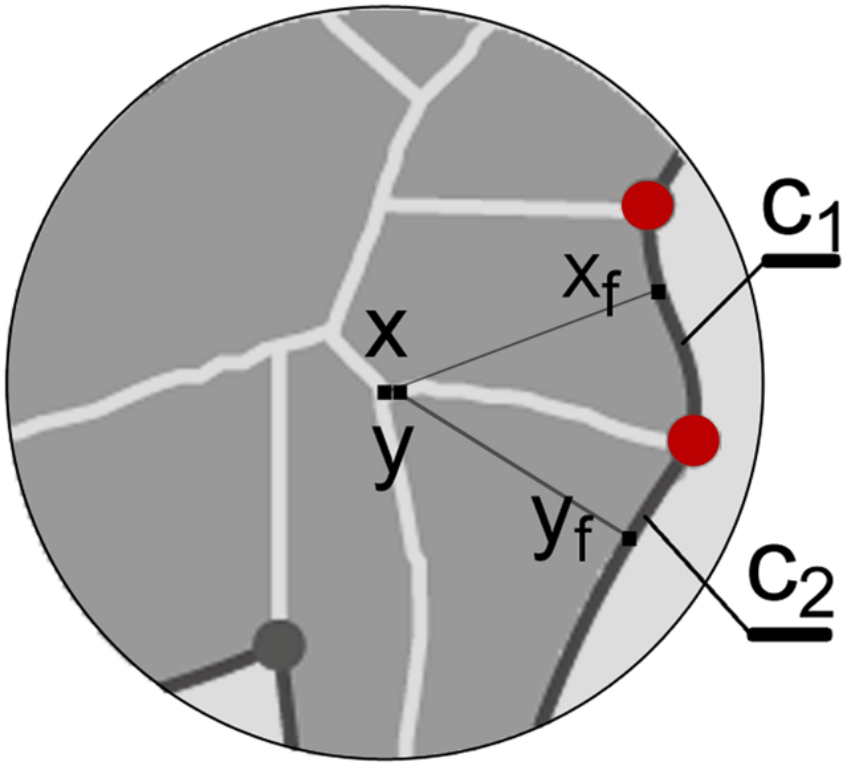
2. Compute and prune Integer Medial Axis



Euclidean skeletons of digital image and volume data in linear time by the integer medial axis transform
W.H. Hesselink and J.B. Roerdink 2007
Time complexity: O(N)
N: image size

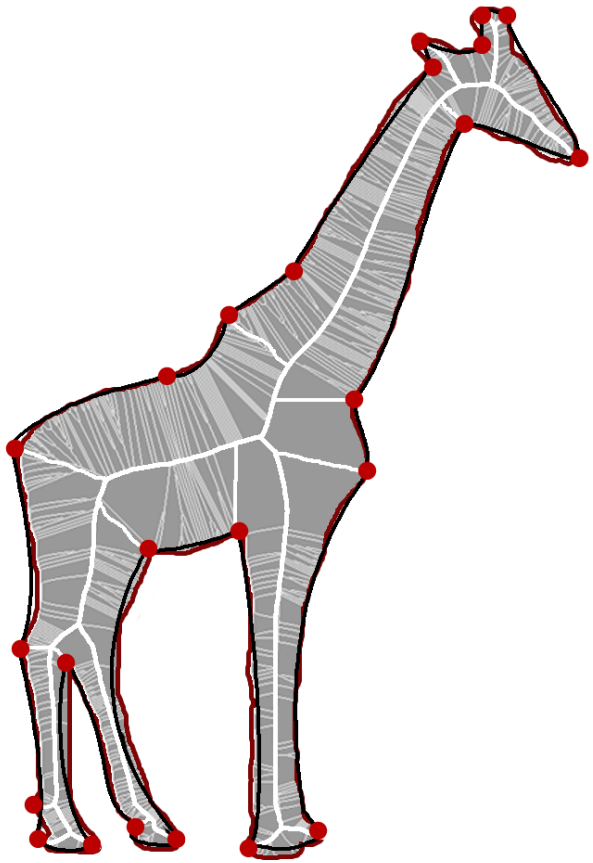
2. Compute and prune Integer Medial Axis

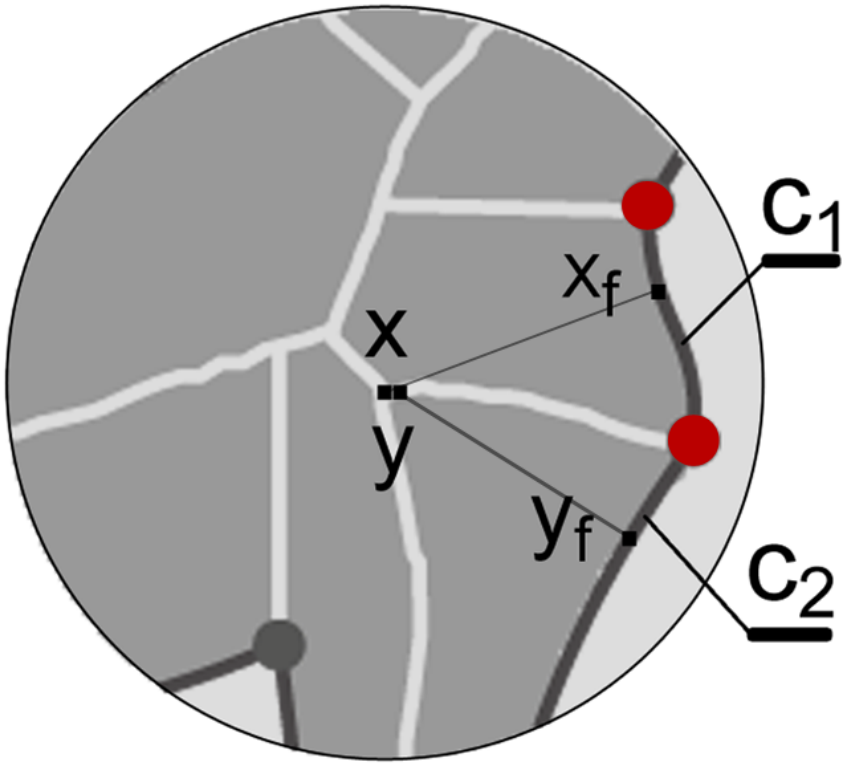




Time complexity: O(N)
N: image size

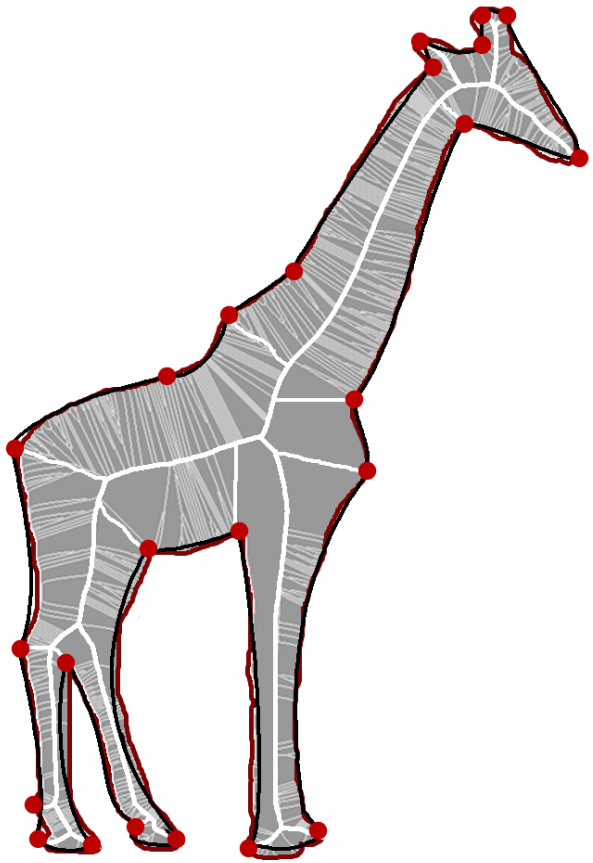
2. Compute and prune Integer Medial Axis

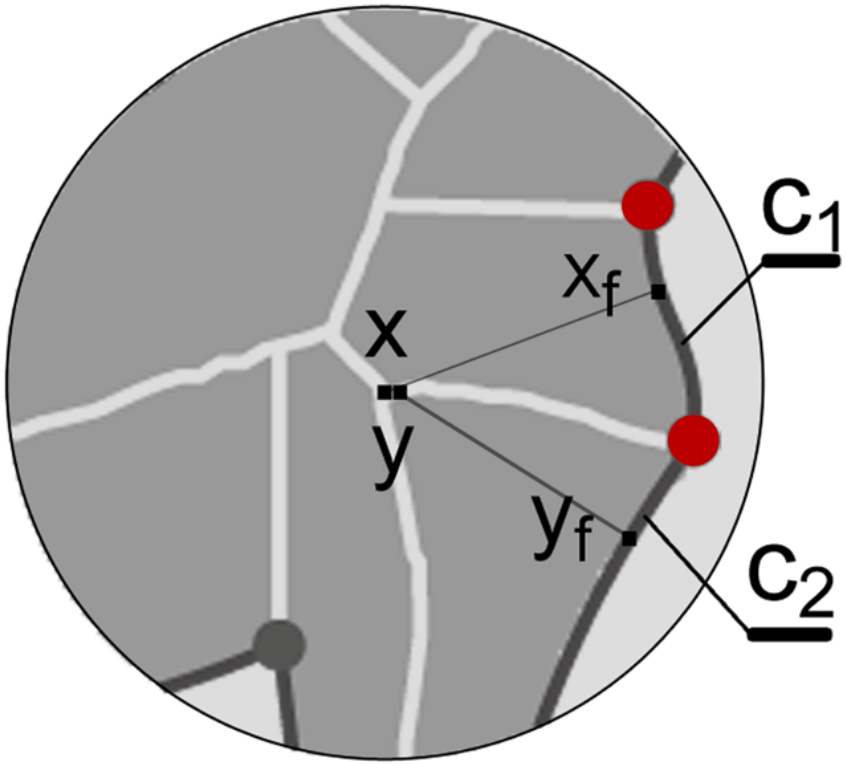




Time complexity: O(N)
N: image size

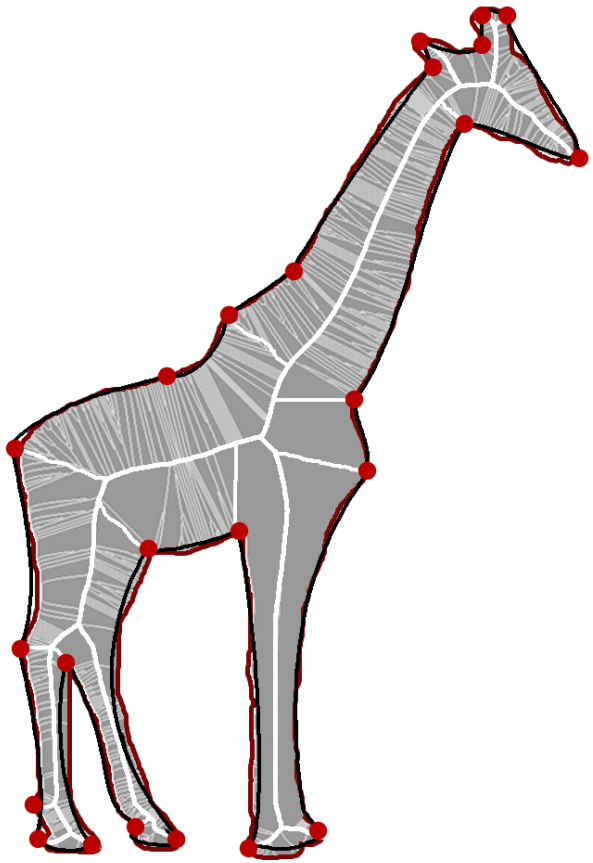
2. Compute and prune Integer Medial Axis

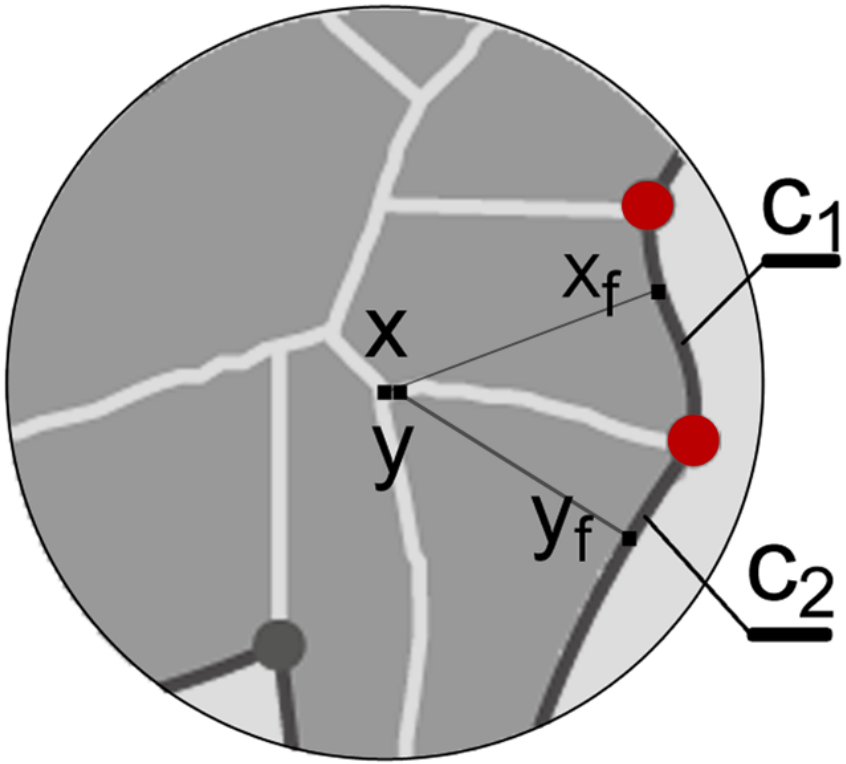




Time complexity: O(N)
N: image size

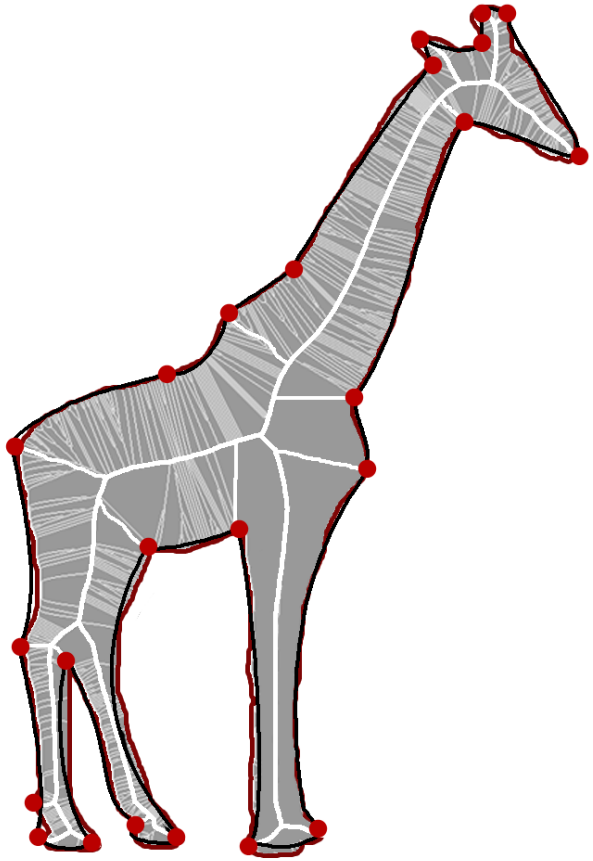
2. Compute and prune Integer Medial Axis

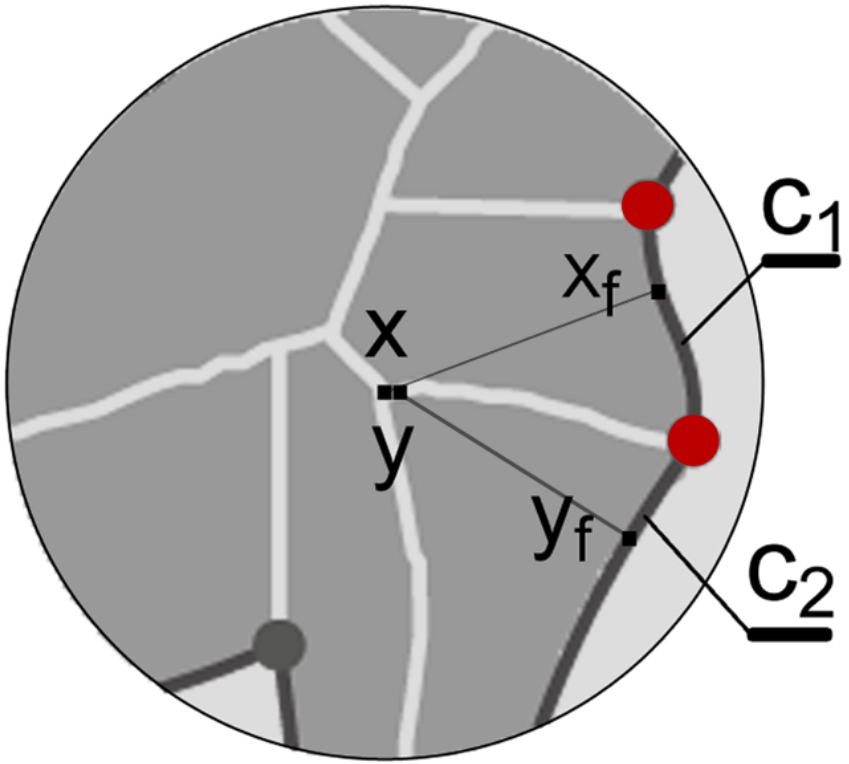




Time complexity: O(N)
N: image size

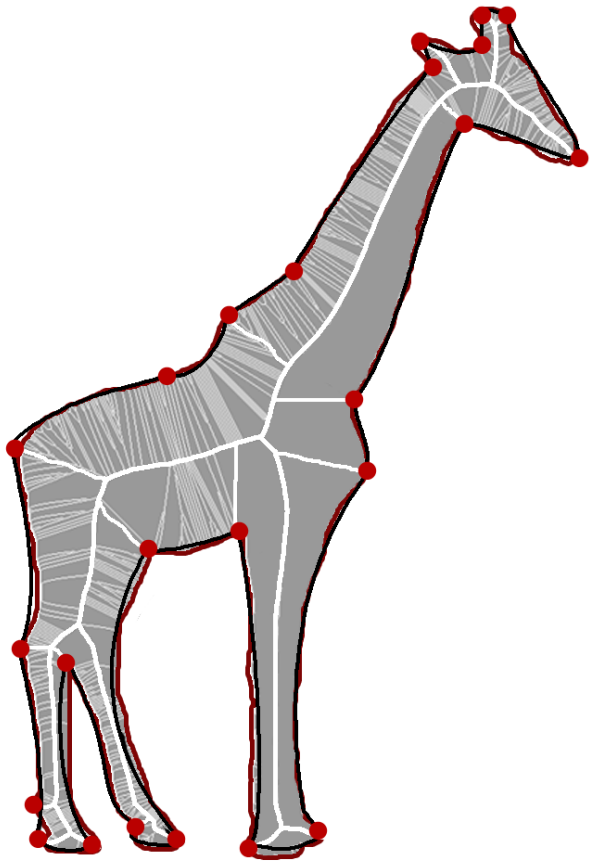
2. Compute and prune Integer Medial Axis

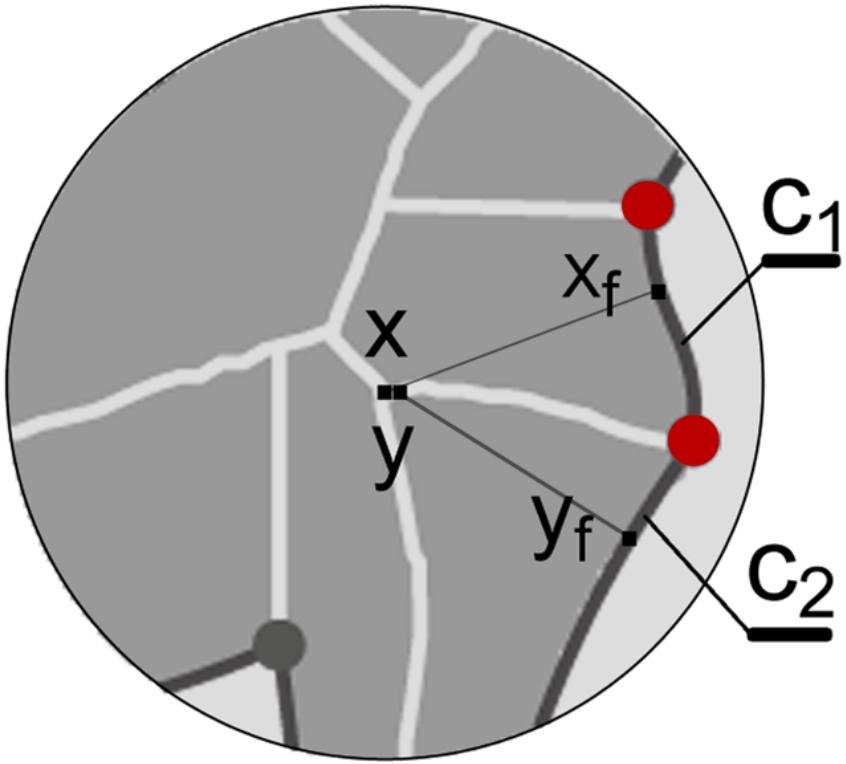




Time complexity: O(N)
N: image size

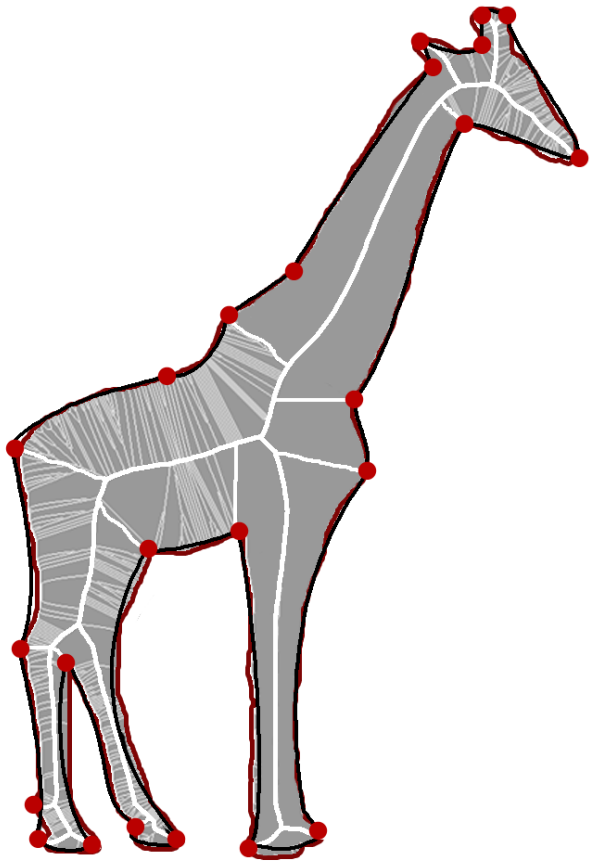
2. Compute and prune Integer Medial Axis

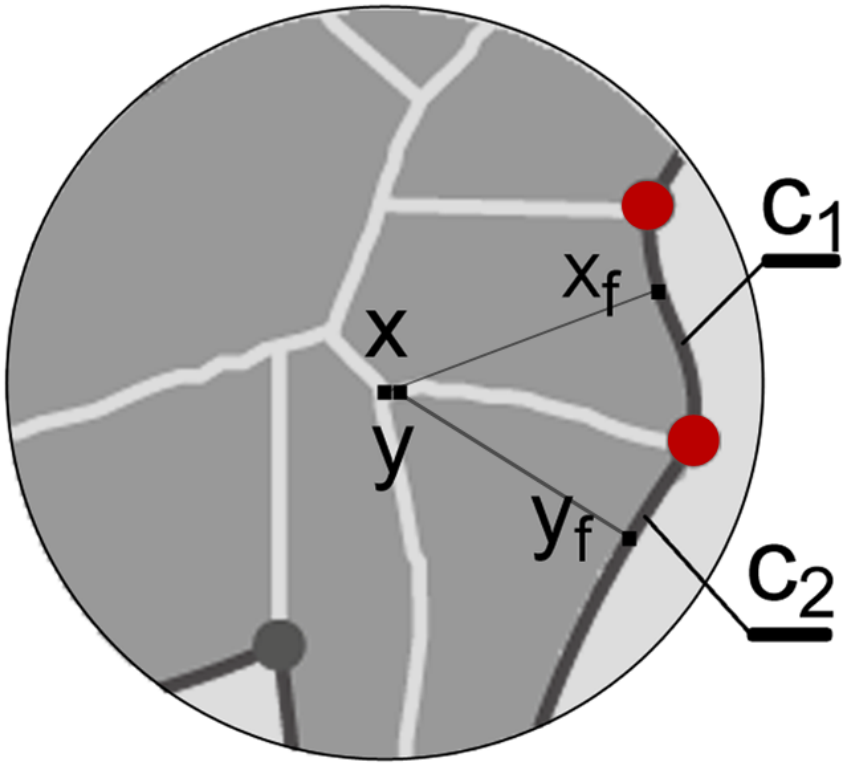




Time complexity: O(N)
N: image size

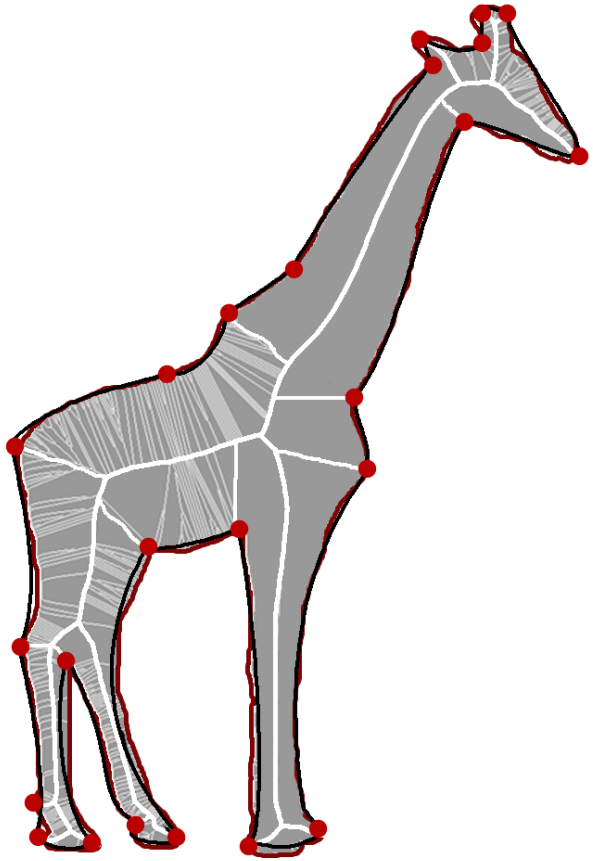
2. Compute and prune Integer Medial Axis

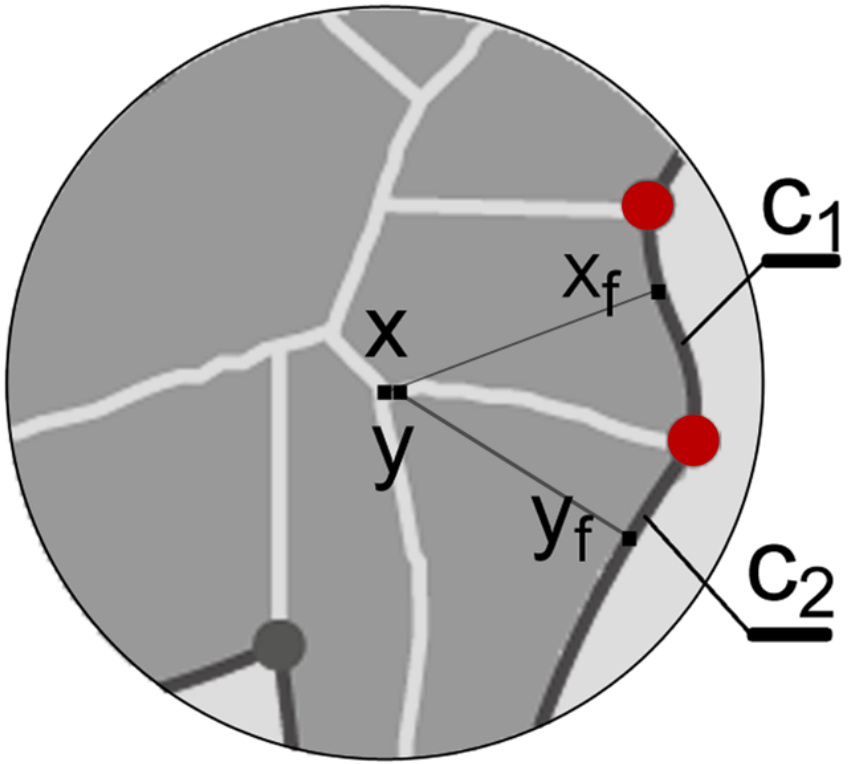




Time complexity: O(N)
N: image size

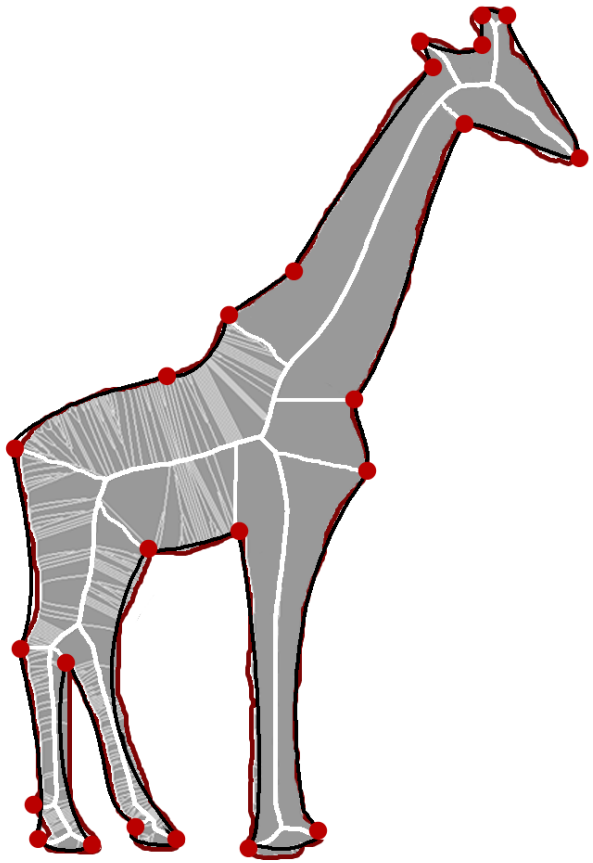
2. Compute and prune Integer Medial Axis

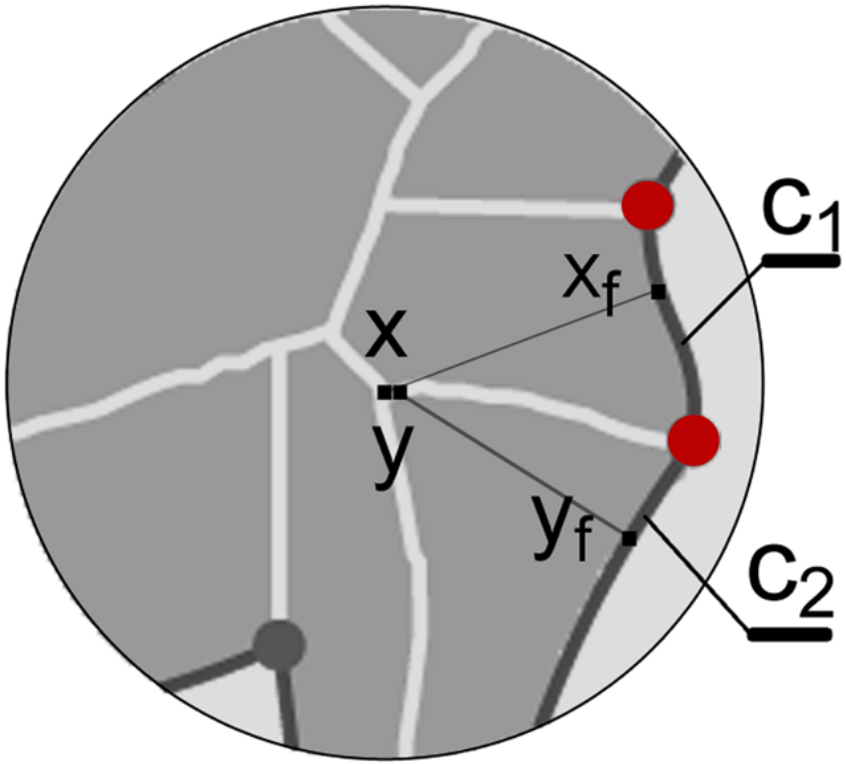




Time complexity: O(N)
N: image size

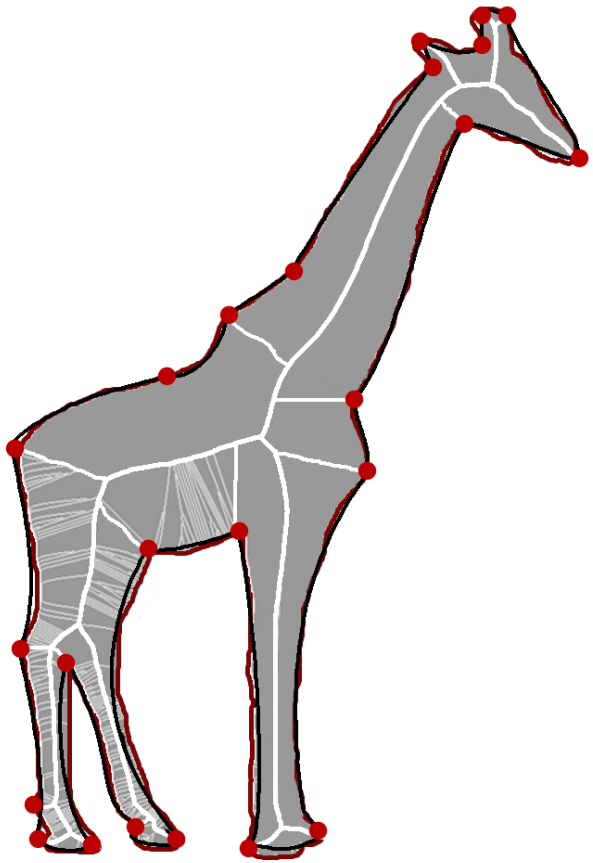
2. Compute and prune Integer Medial Axis

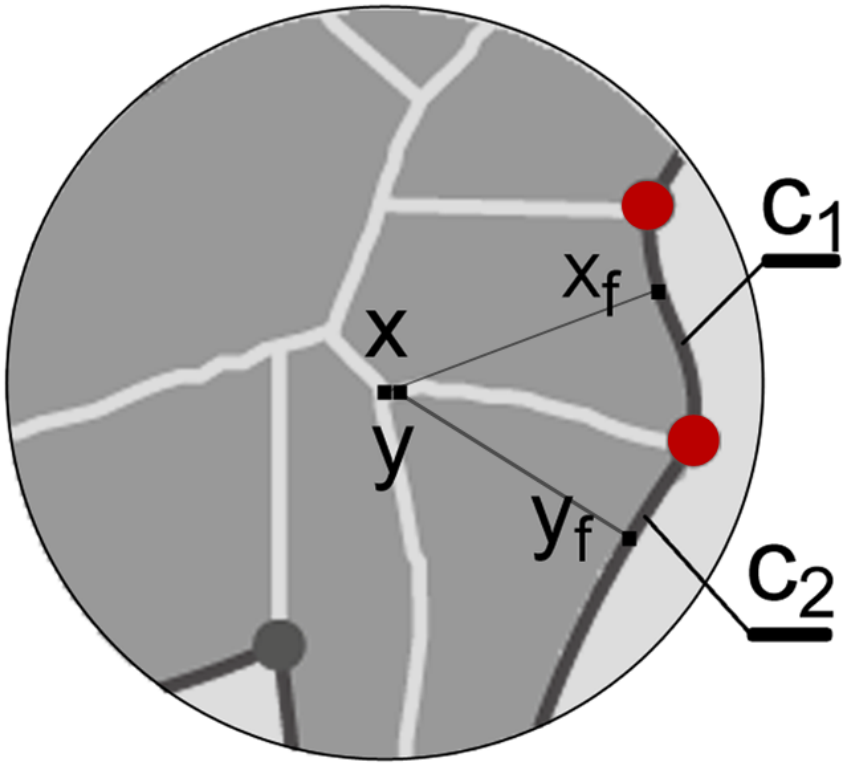




Time complexity: O(N)
N: image size

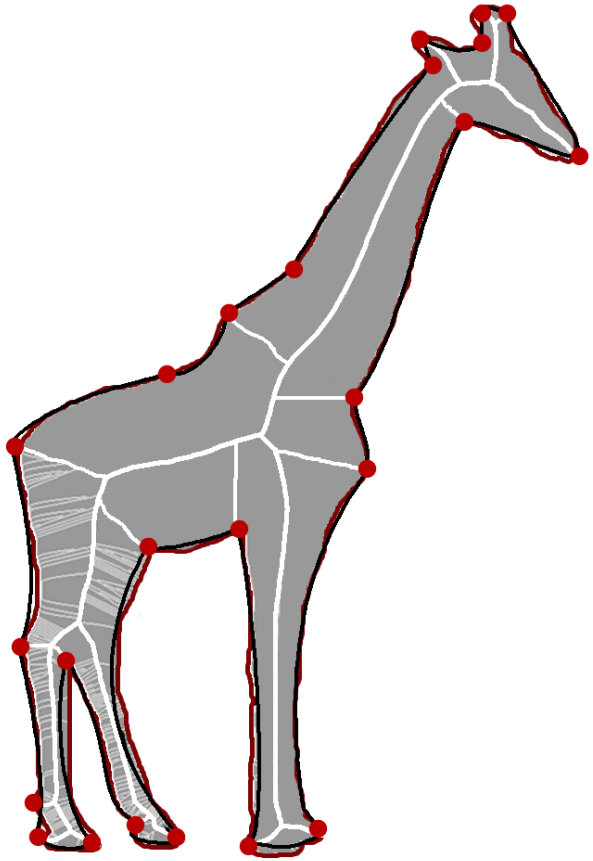
2. Compute and prune Integer Medial Axis

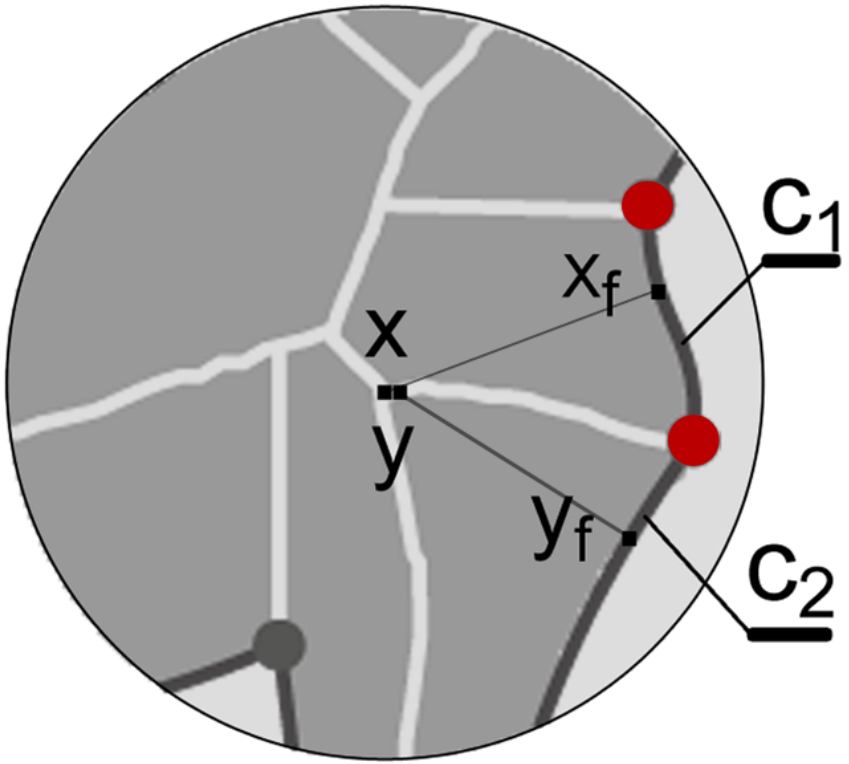




Time complexity: O(N)
N: image size

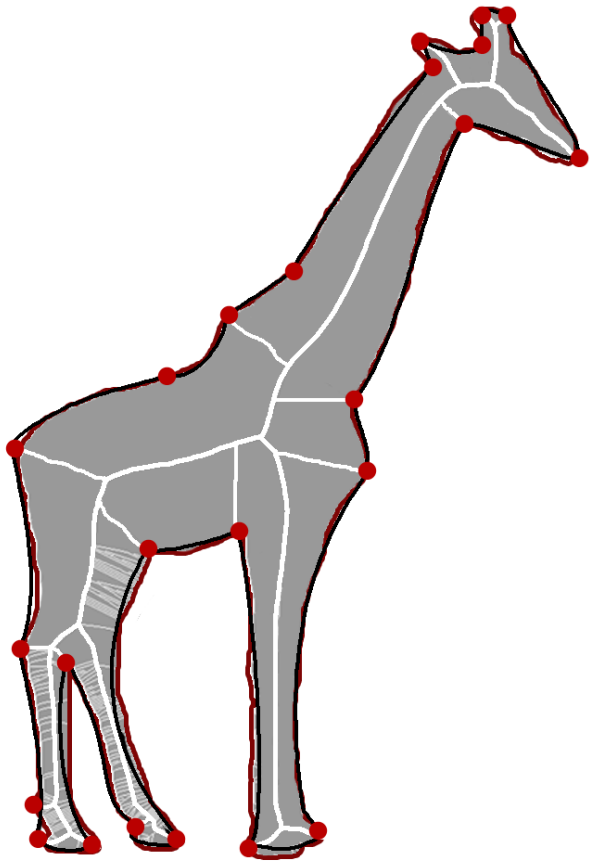
2. Compute and prune Integer Medial Axis

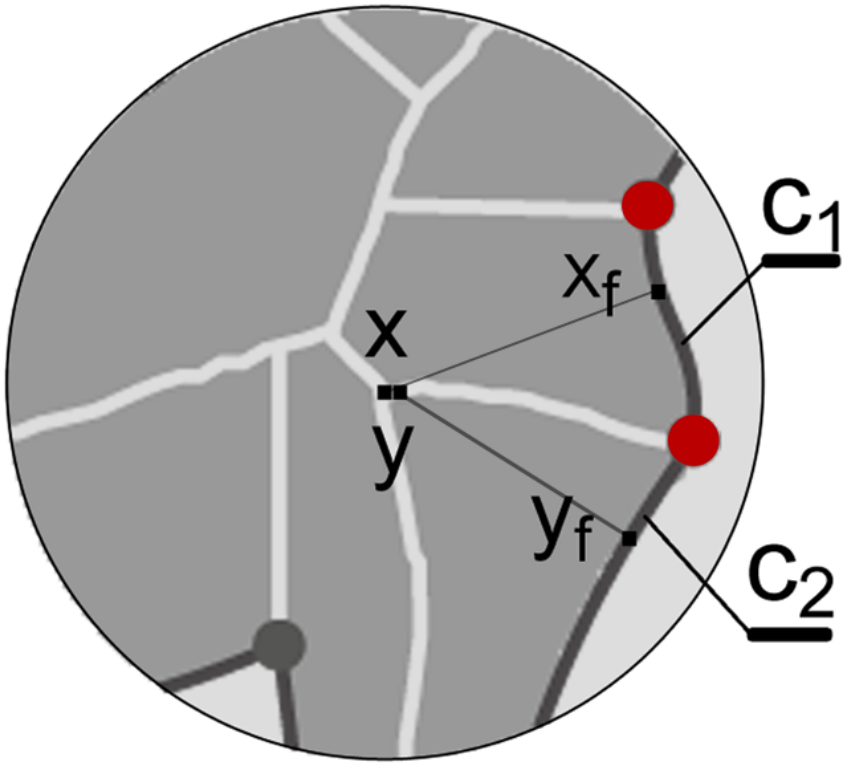




Time complexity: O(N)
N: image size

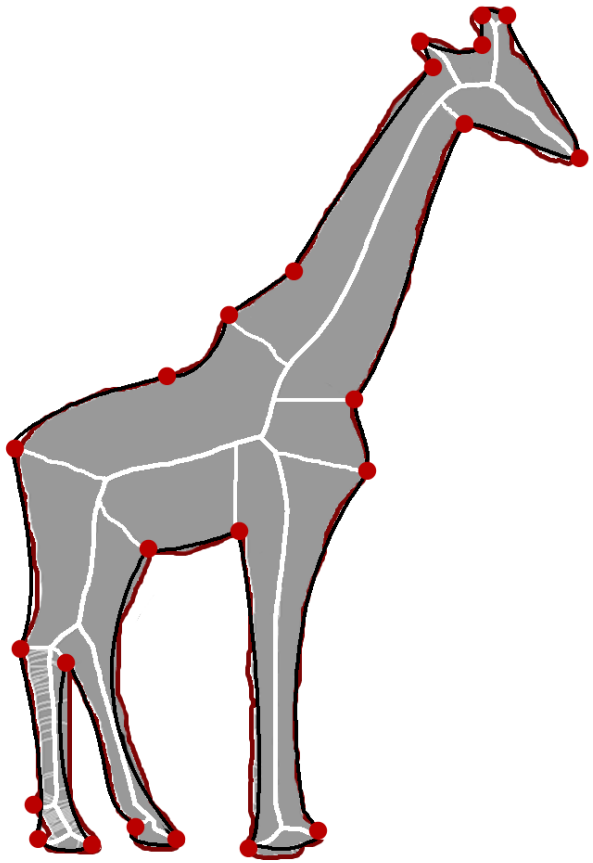
2. Compute and prune Integer Medial Axis

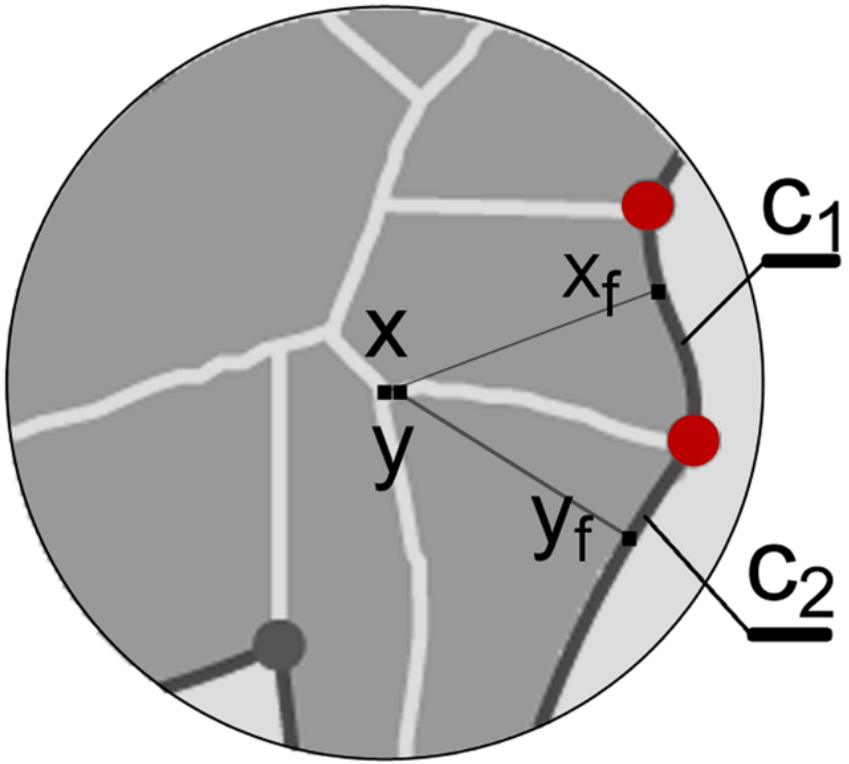




Time complexity: O(N)
N: image size

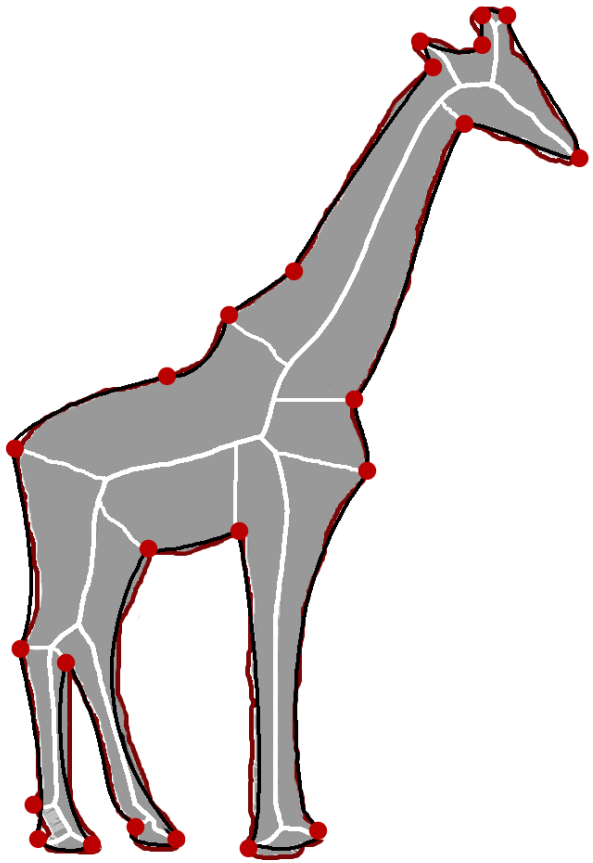
2. Compute and prune Integer Medial Axis

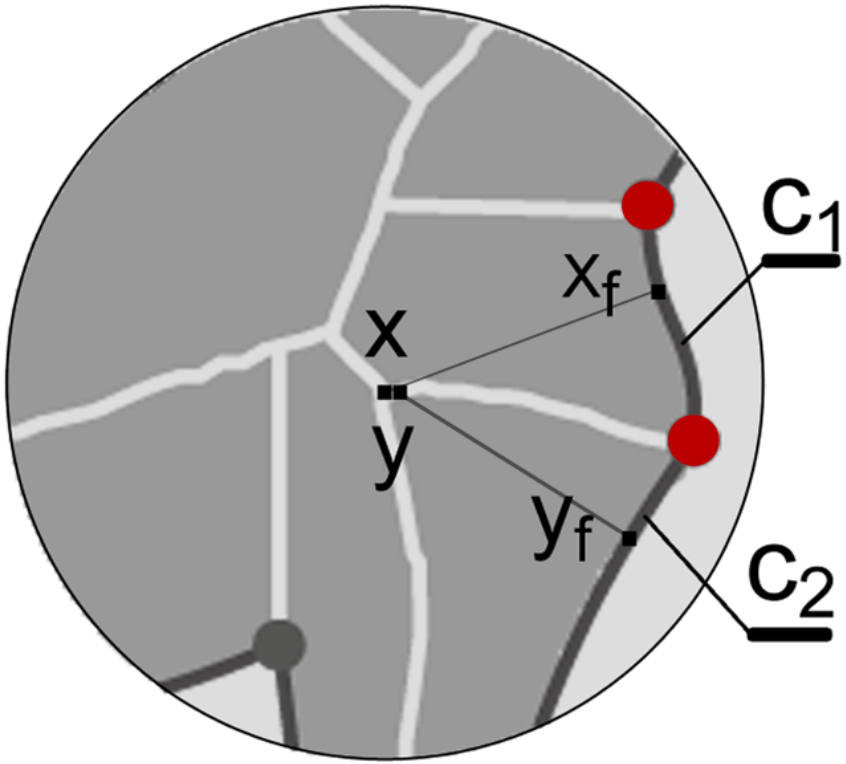




Time complexity: O(N)
N: image size

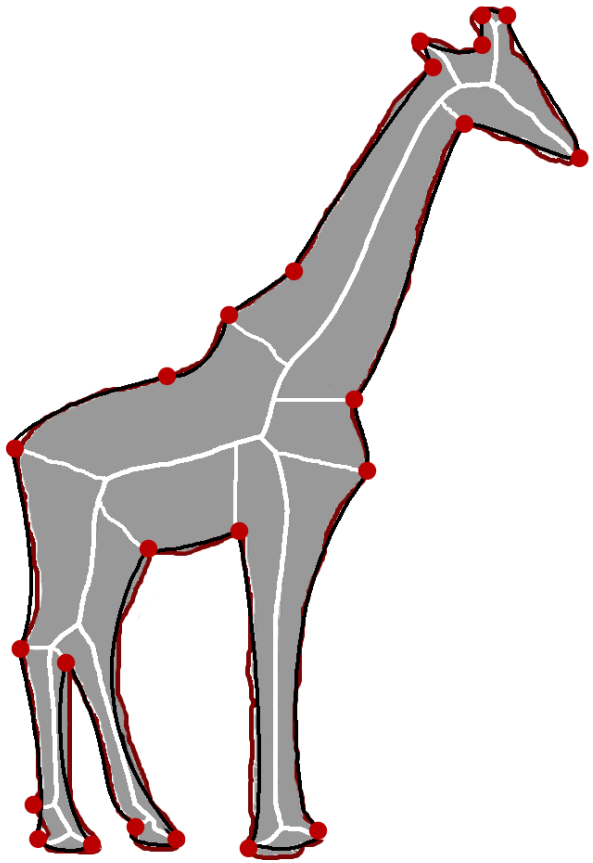
2. Compute and prune Integer Medial Axis

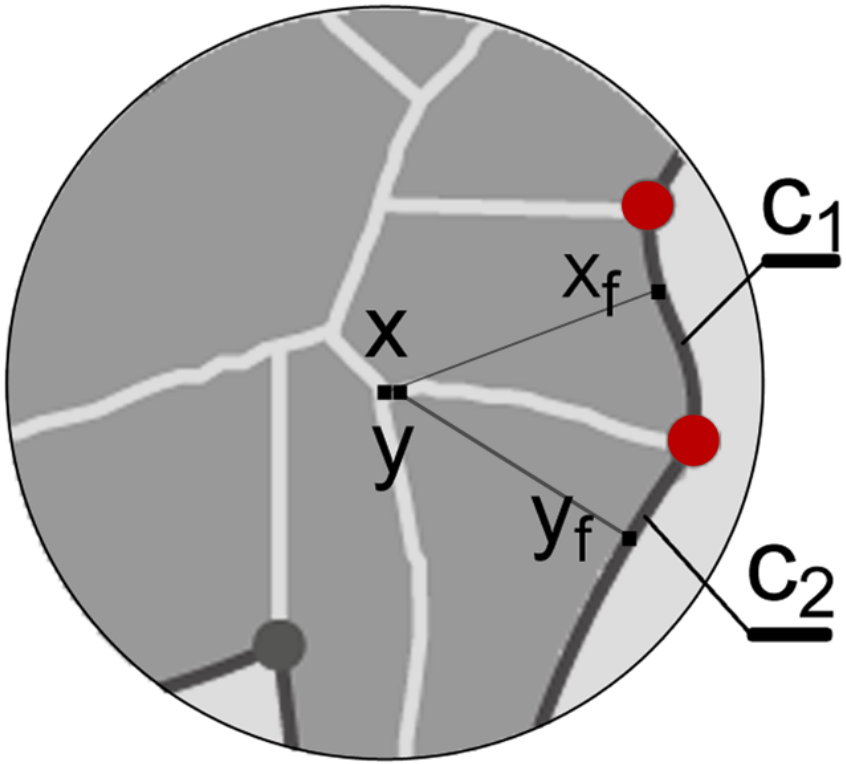




Time complexity: O(N)
N: image size

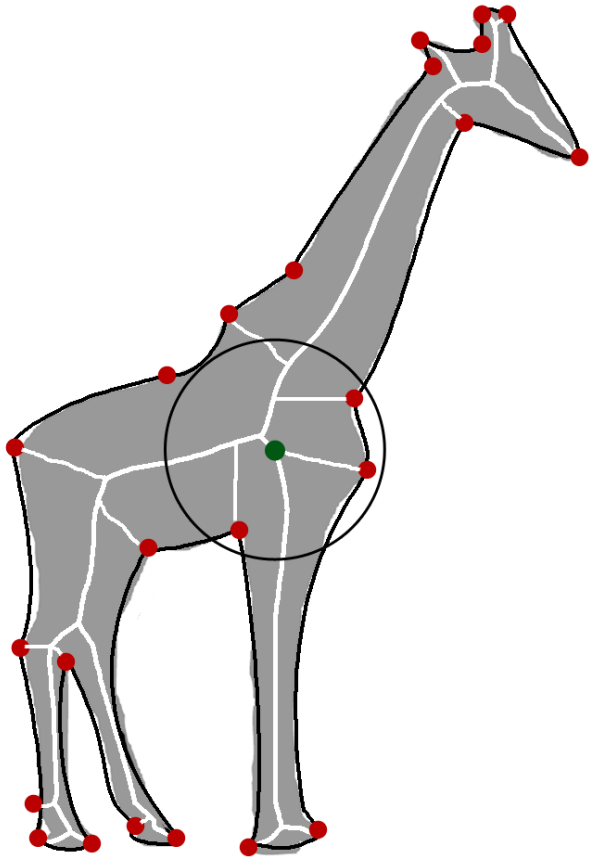
2. Compute and prune Integer Medial Axis

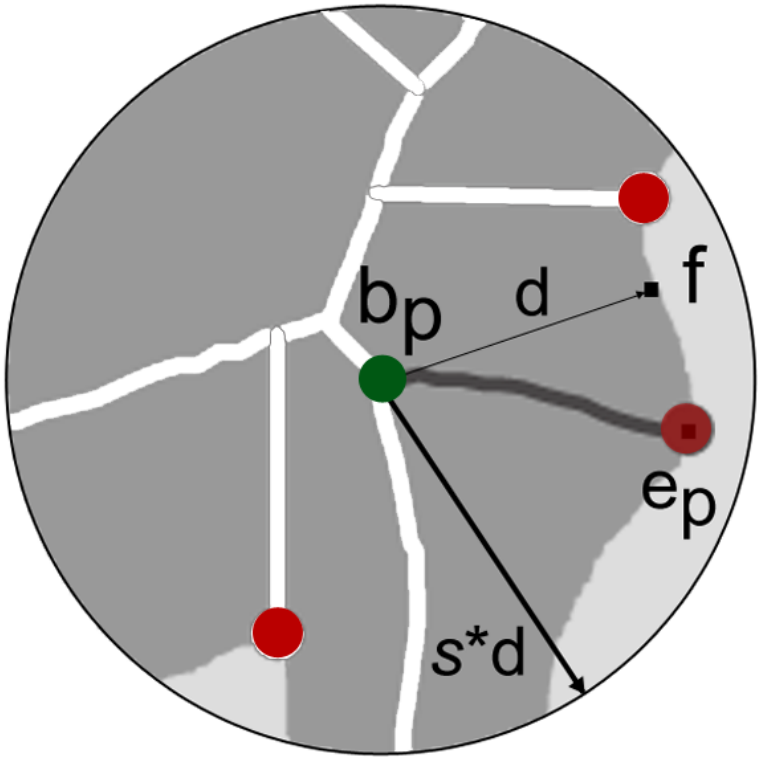




Time complexity: O(N)
N: image size

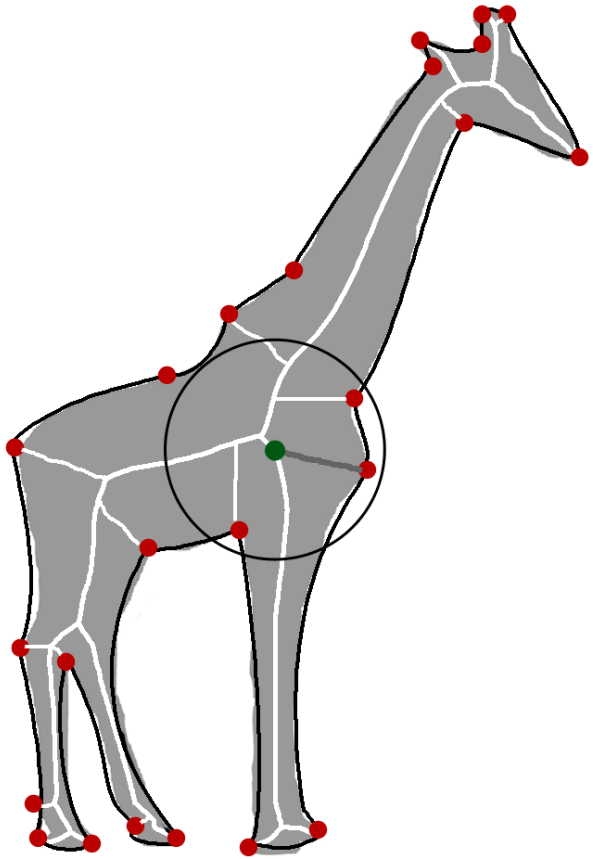
3. Check every branch chain for removal

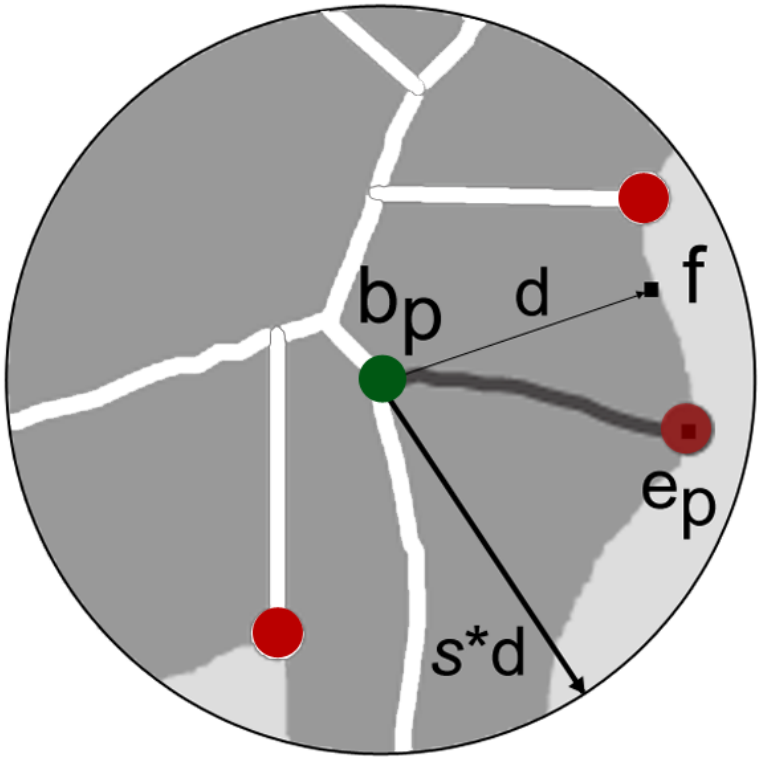




Time complexity: O(n)
n: skeleton pixels

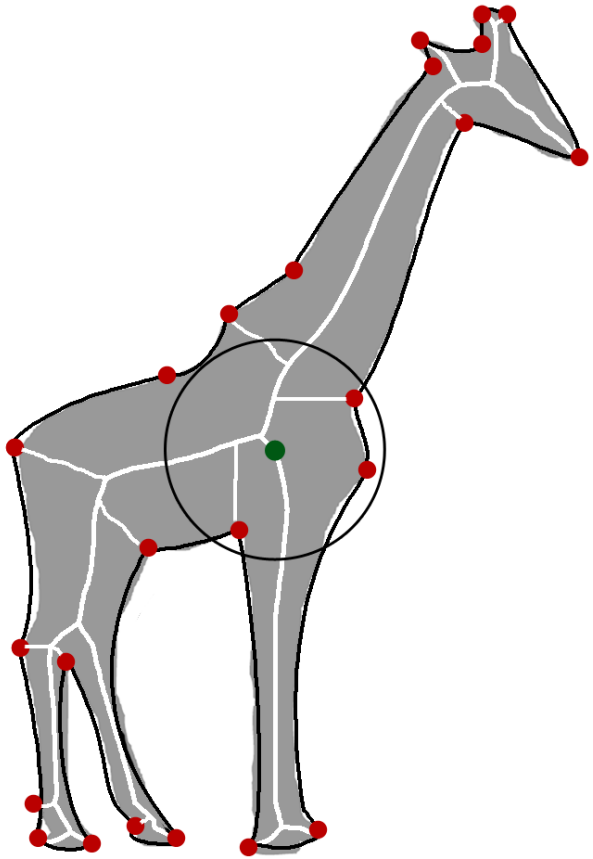
3. Check every branch chain for removal

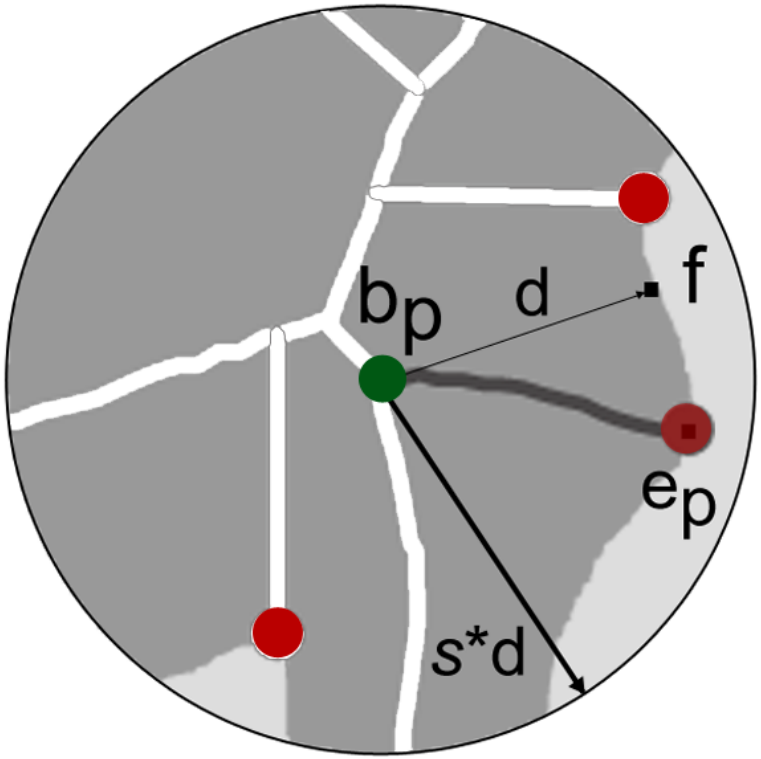




Time complexity: O(n)
n: skeleton pixels

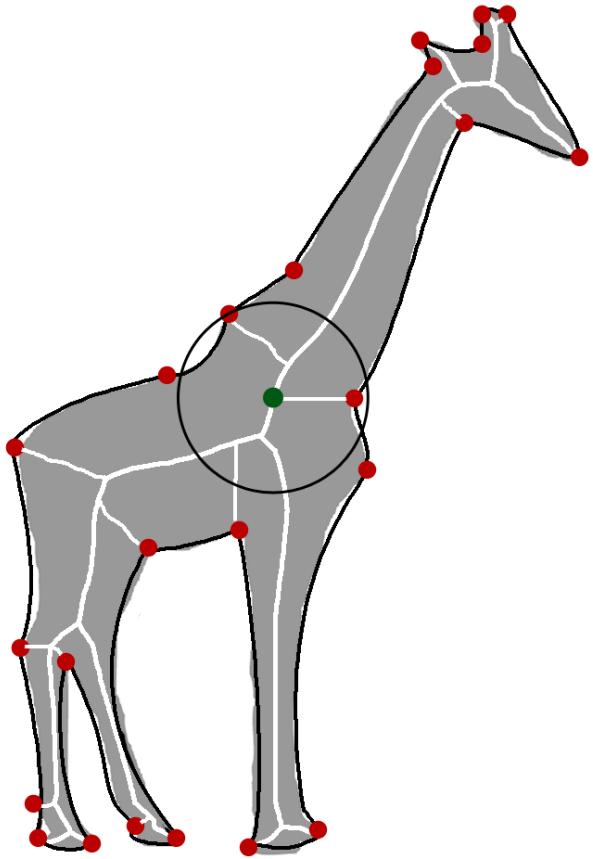
3. Check every branch chain for removal

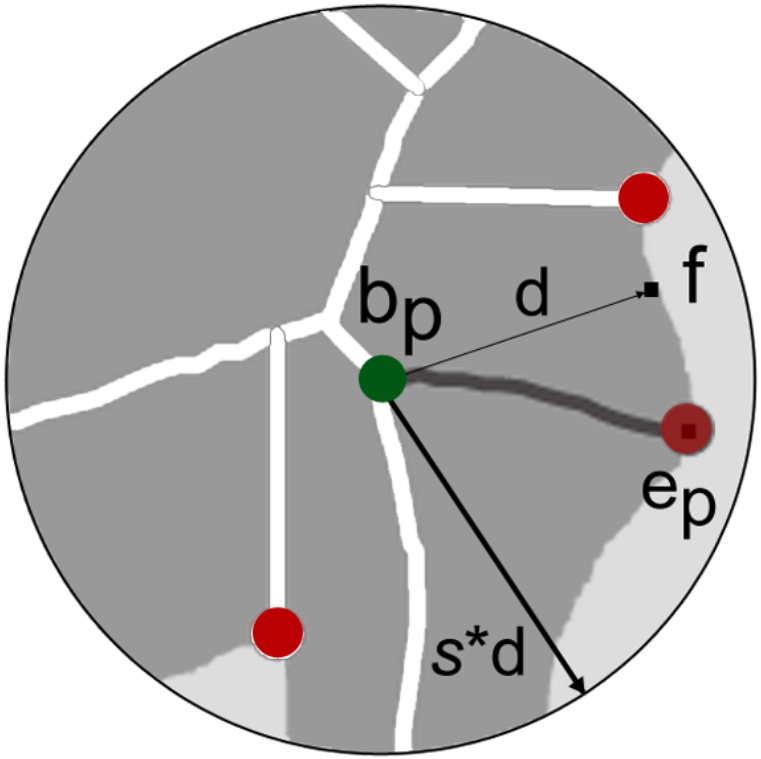




Time complexity: O(n)
n: skeleton pixels

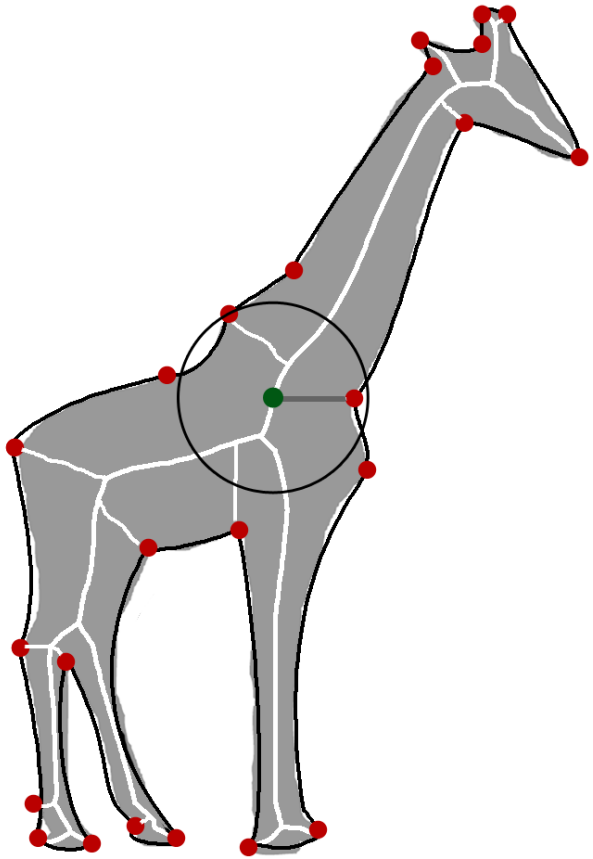
3. Check every branch chain for removal

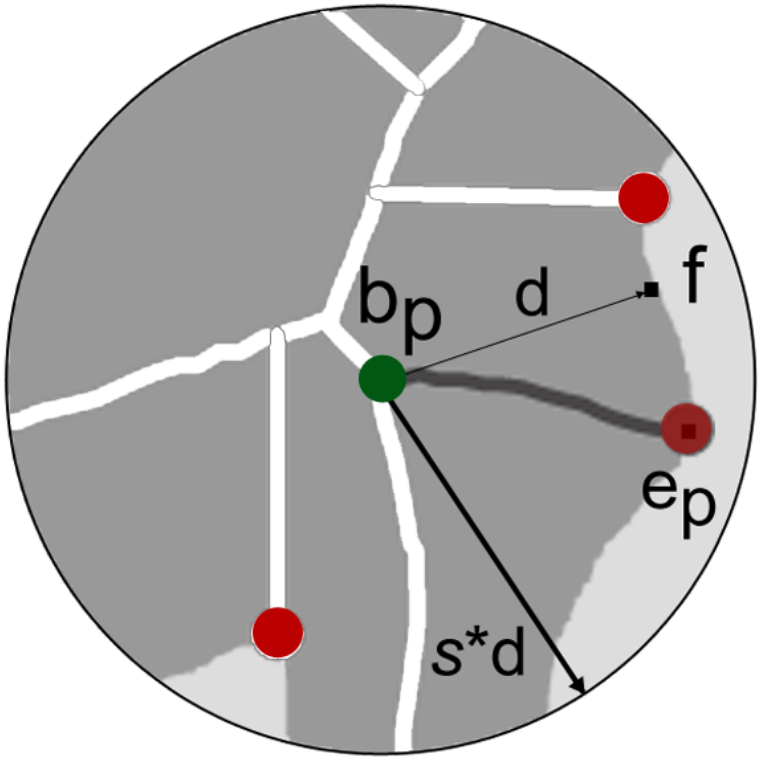




Time complexity: O(n)
n: skeleton pixels

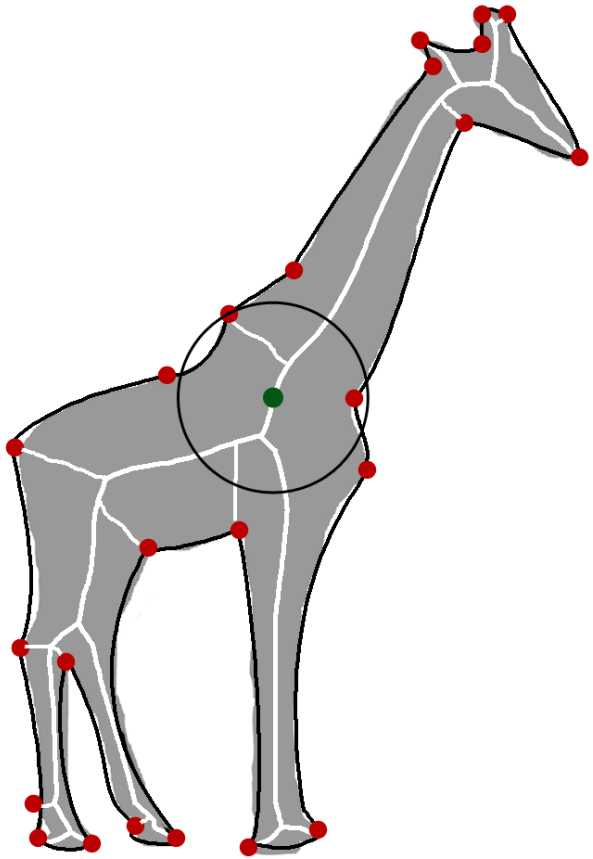
3. Check every branch chain for removal

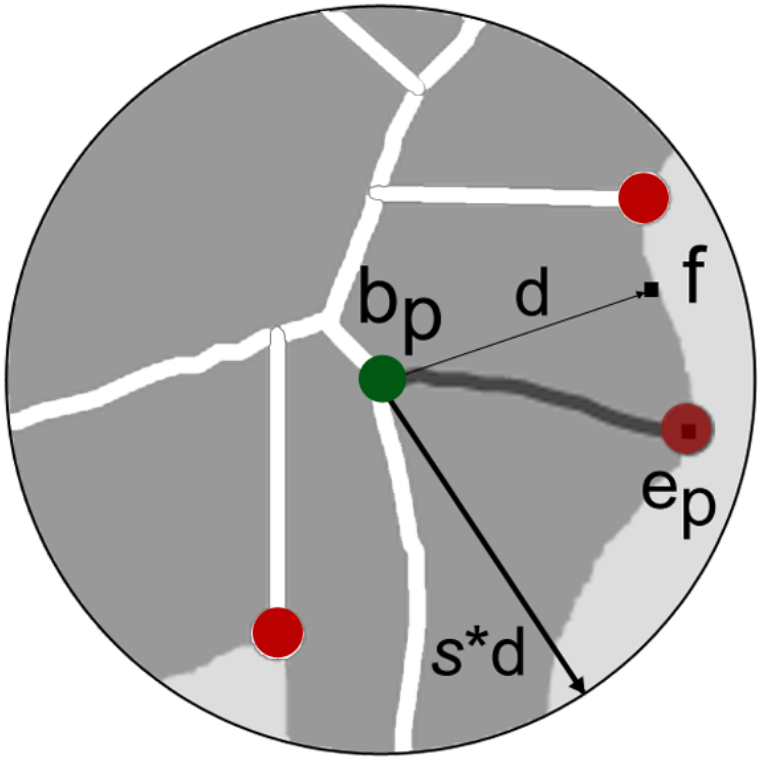




Time complexity: O(n)
n: skeleton pixels

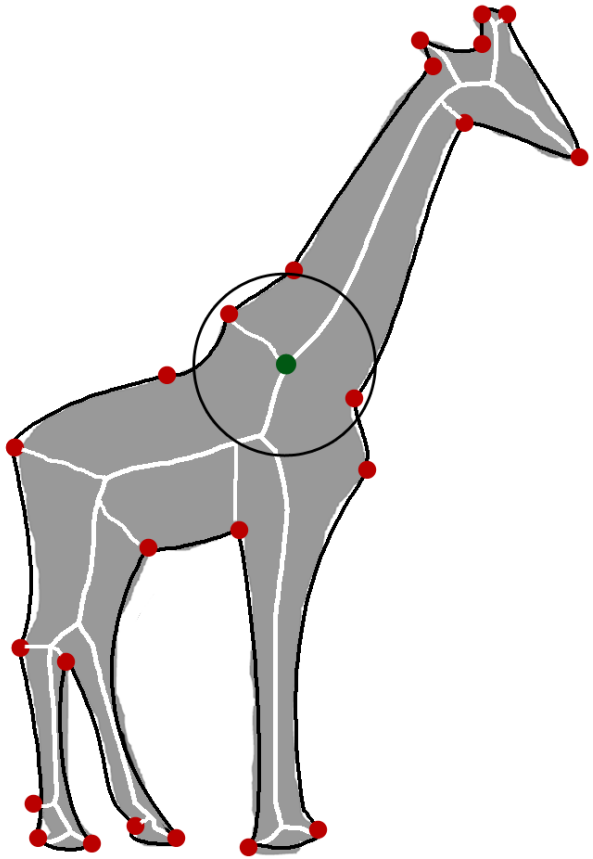
3. Check every branch chain for removal

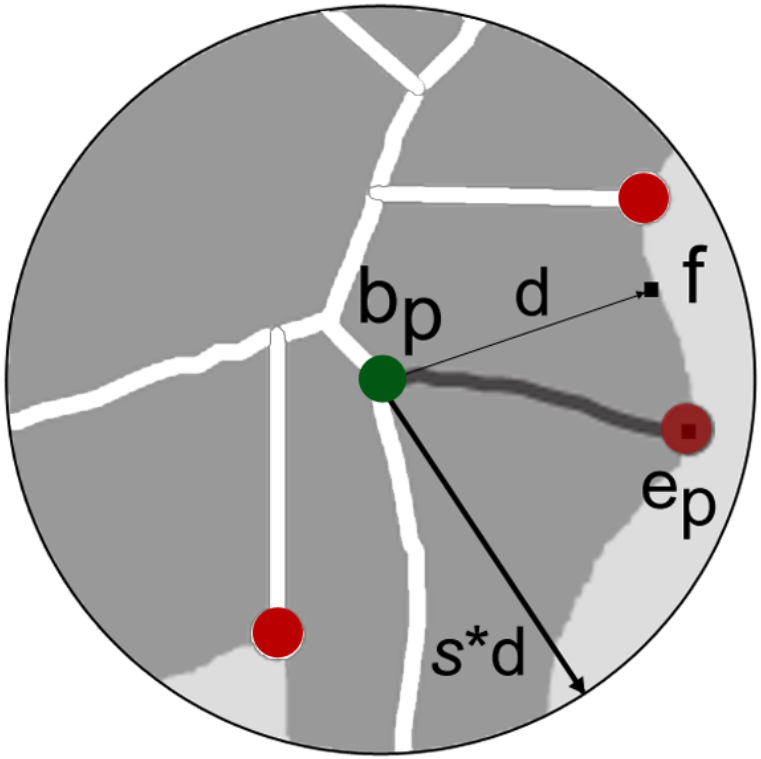




Time complexity: O(n)
n: skeleton pixels

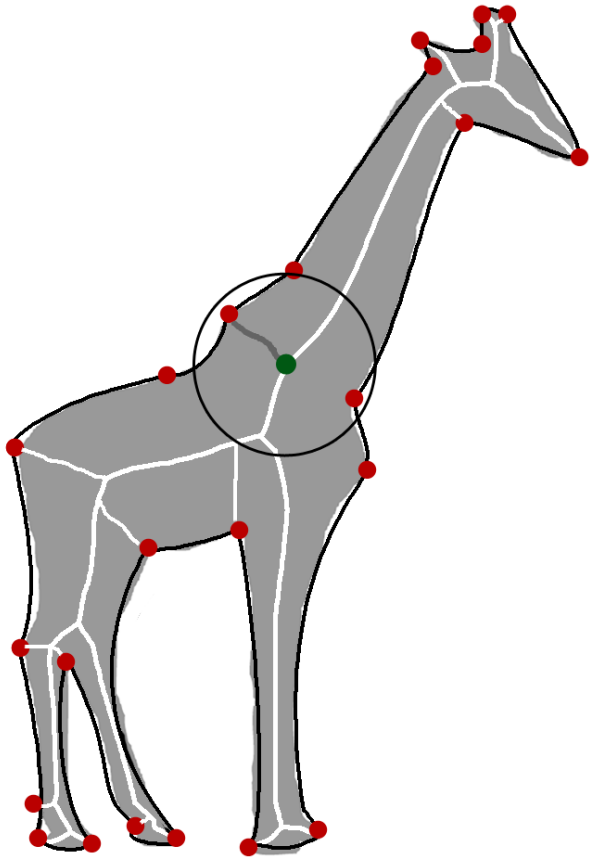
3. Check every branch chain for removal

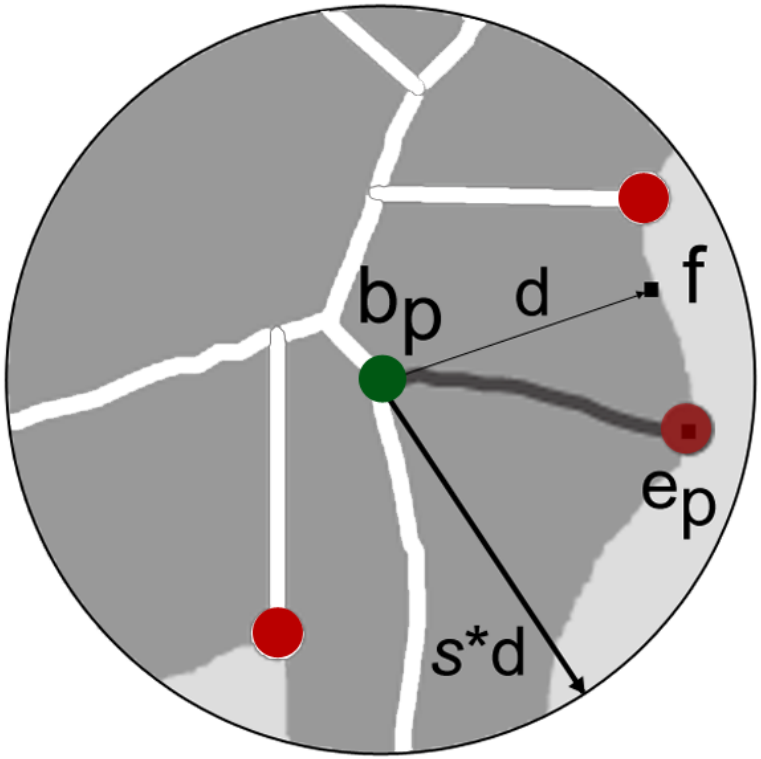




Time complexity: O(n)
n: skeleton pixels

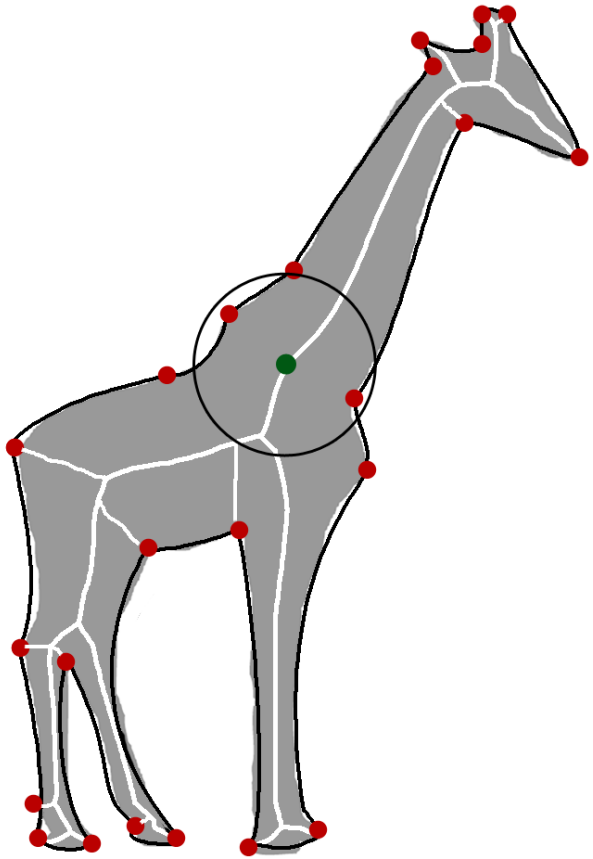
3. Check every branch chain for removal

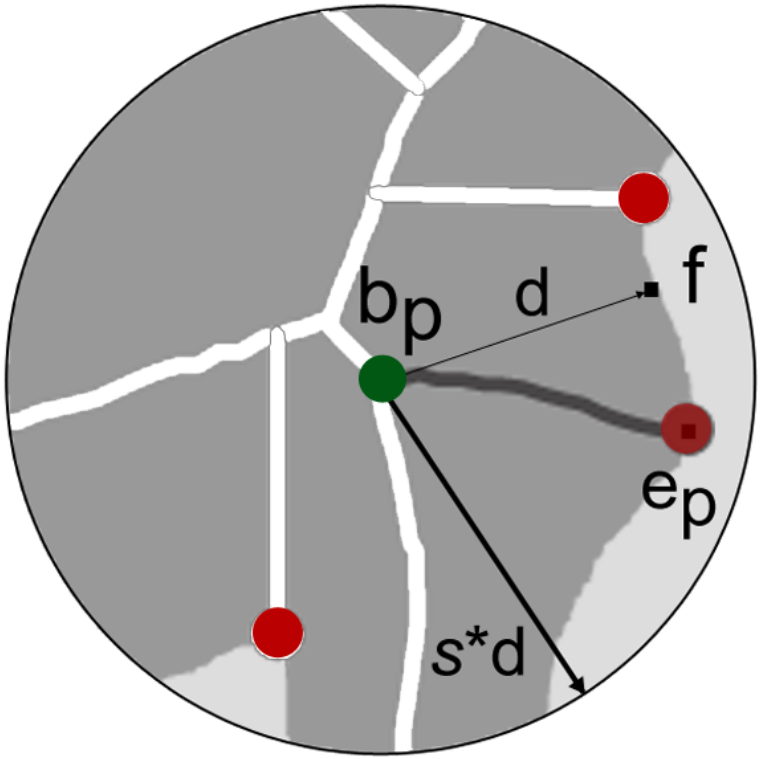




Time complexity: O(n)
n: skeleton pixels

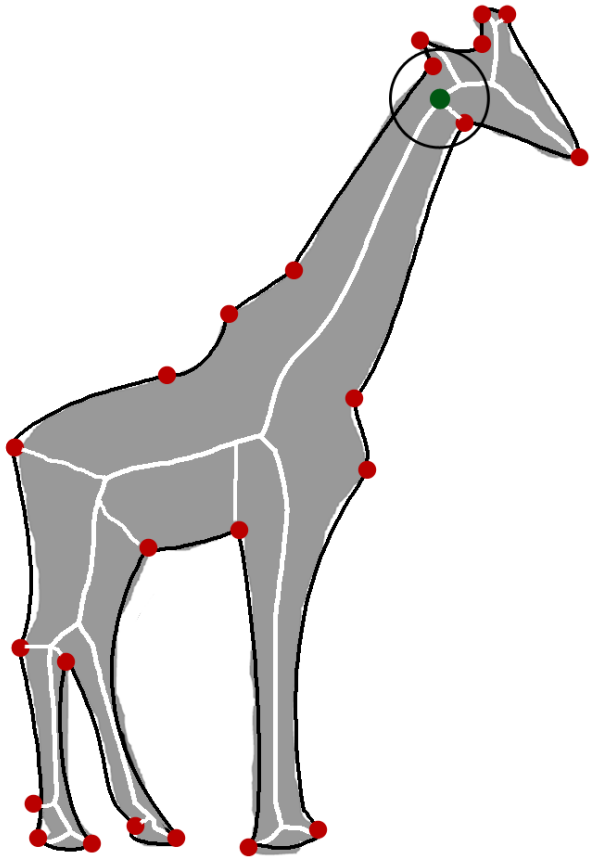
3. Check every branch chain for removal

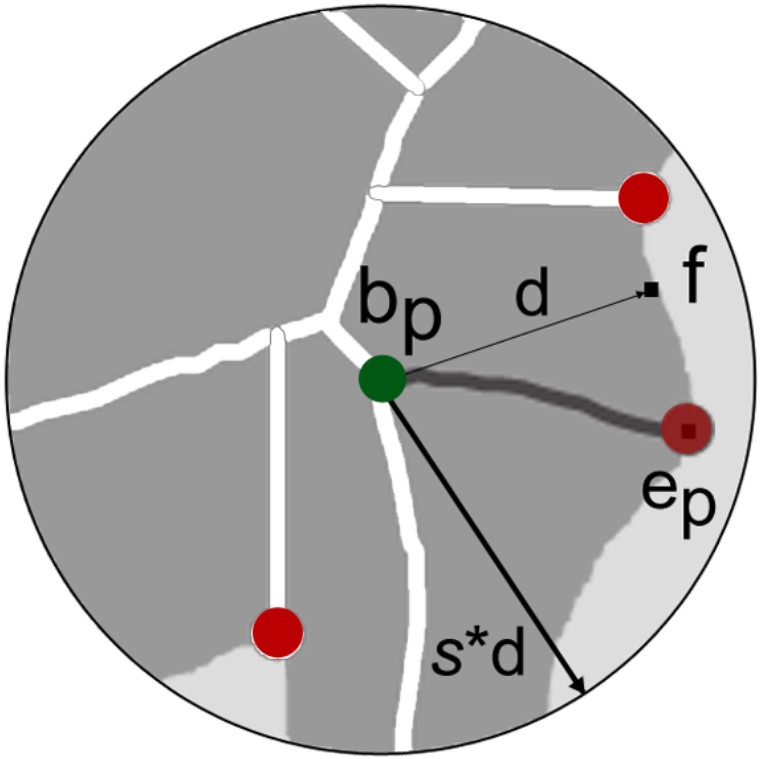




Time complexity: O(n)
n: skeleton pixels

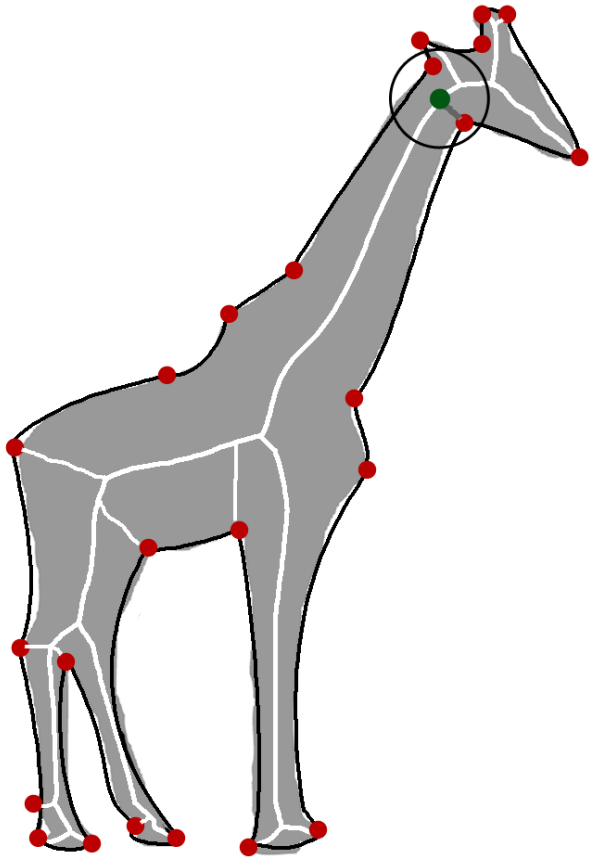
3. Check every branch chain for removal

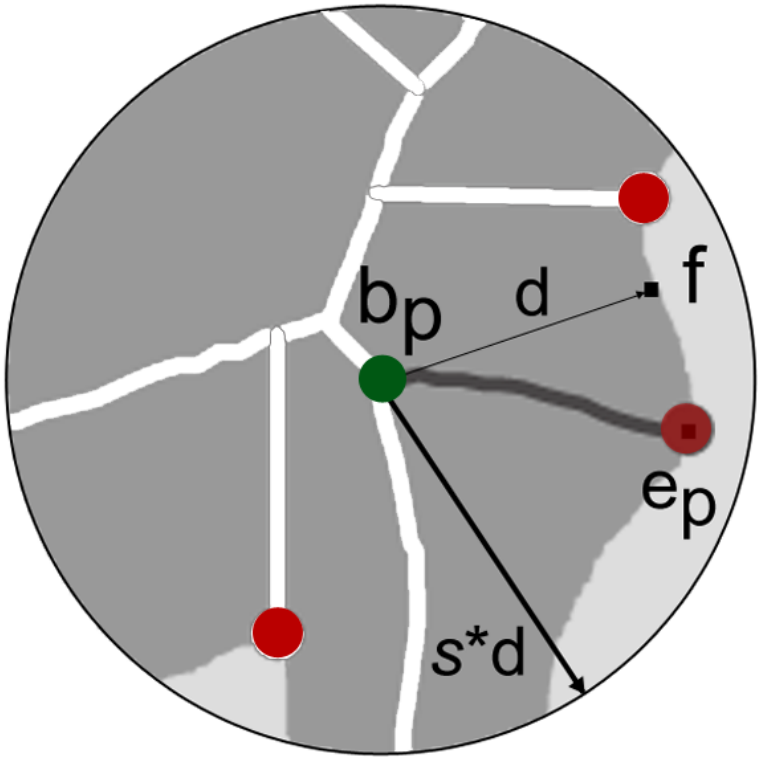




Time complexity: O(n)
n: skeleton pixels

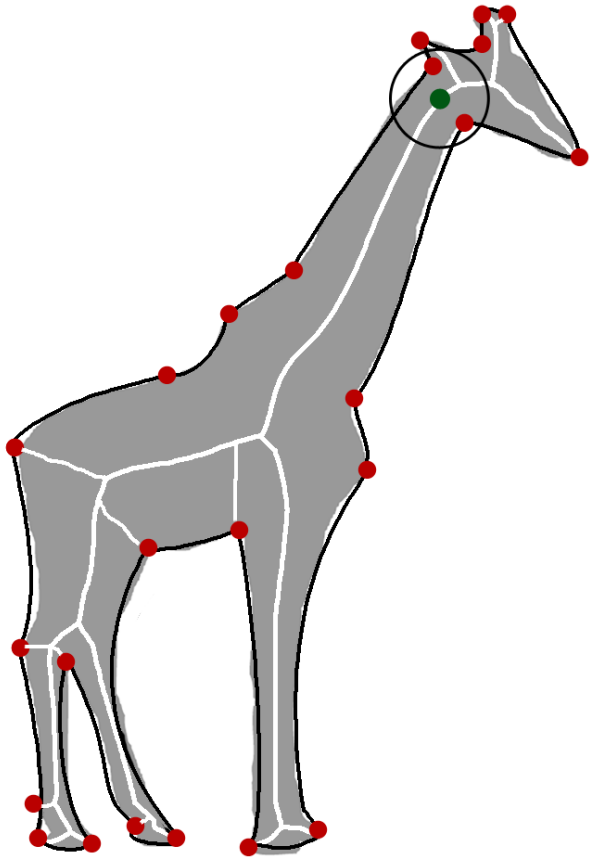
3. Check every branch chain for removal

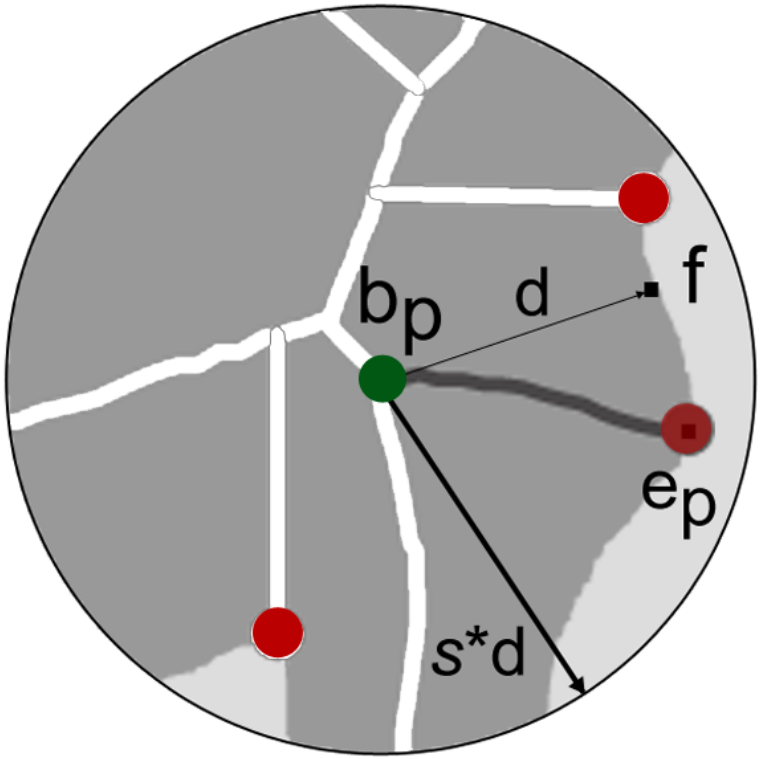




Time complexity: O(n)
n: skeleton pixels

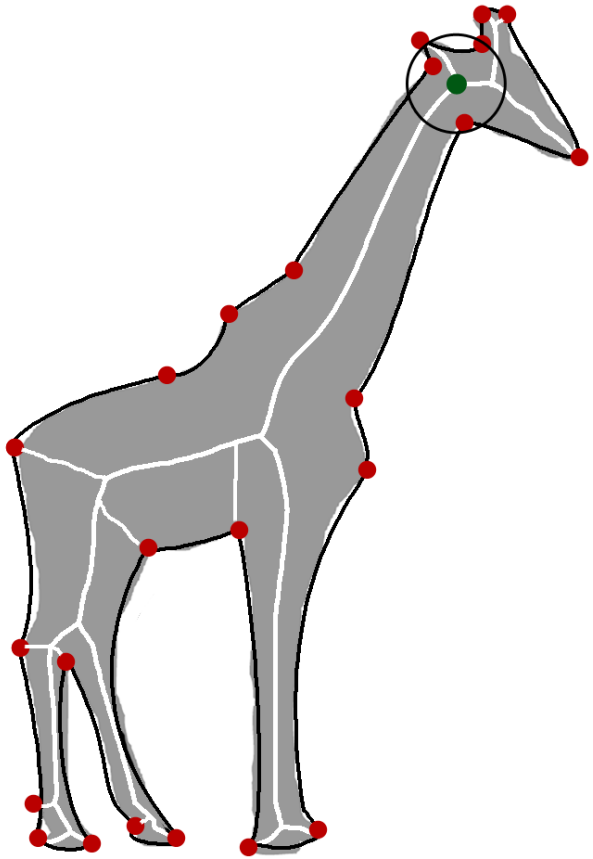
3. Check every branch chain for removal

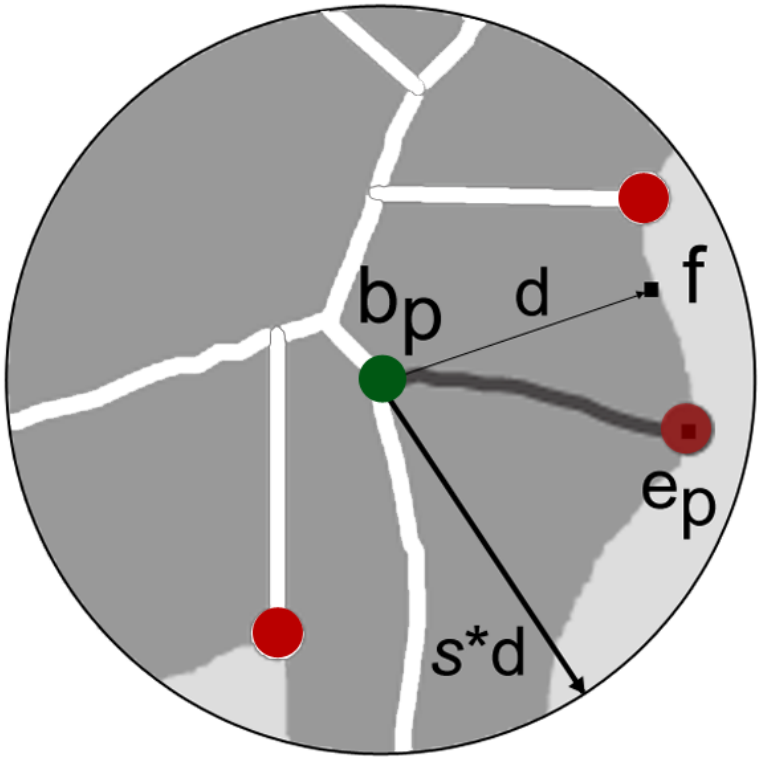




Time complexity: O(n)
n: skeleton pixels

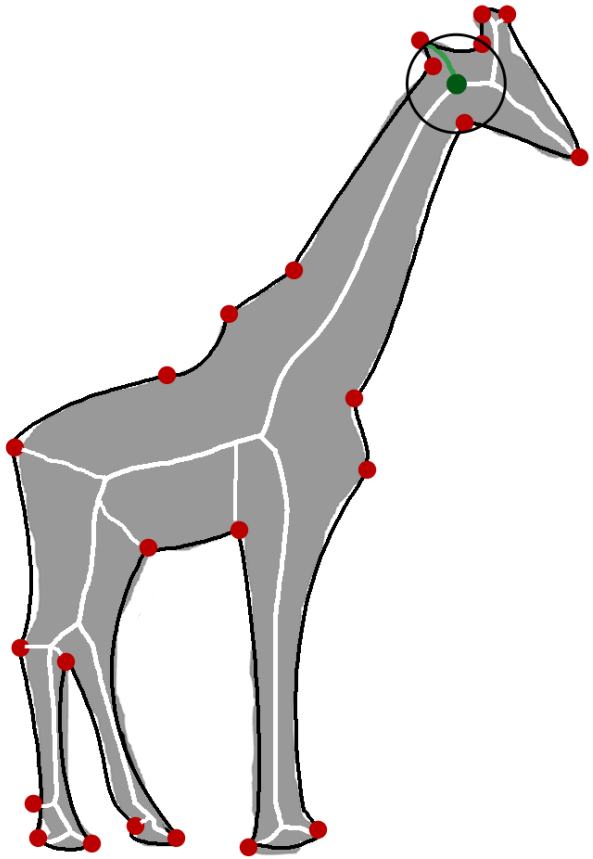
3. Check every branch chain for removal

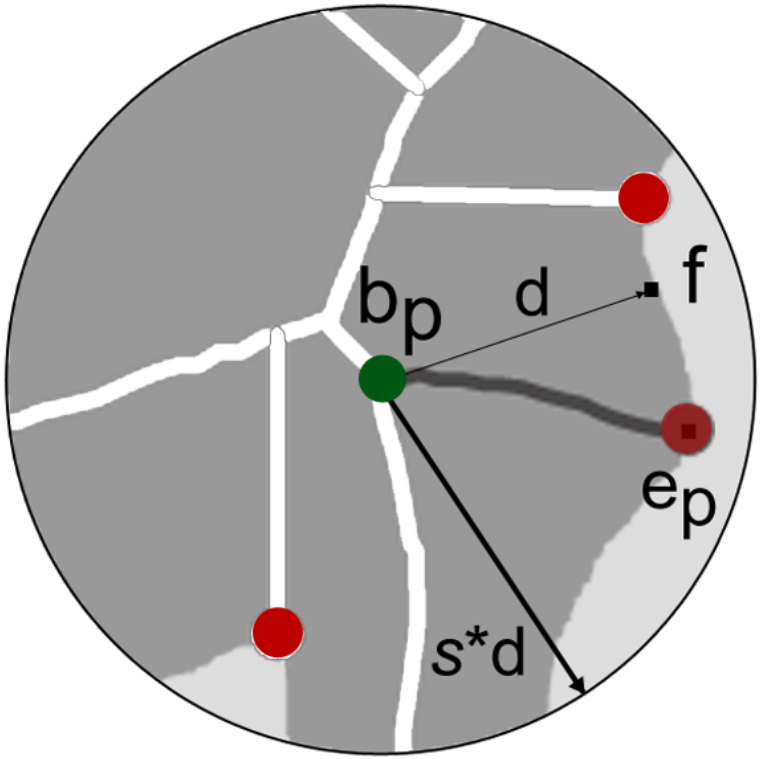




Time complexity: O(n)
n: skeleton pixels

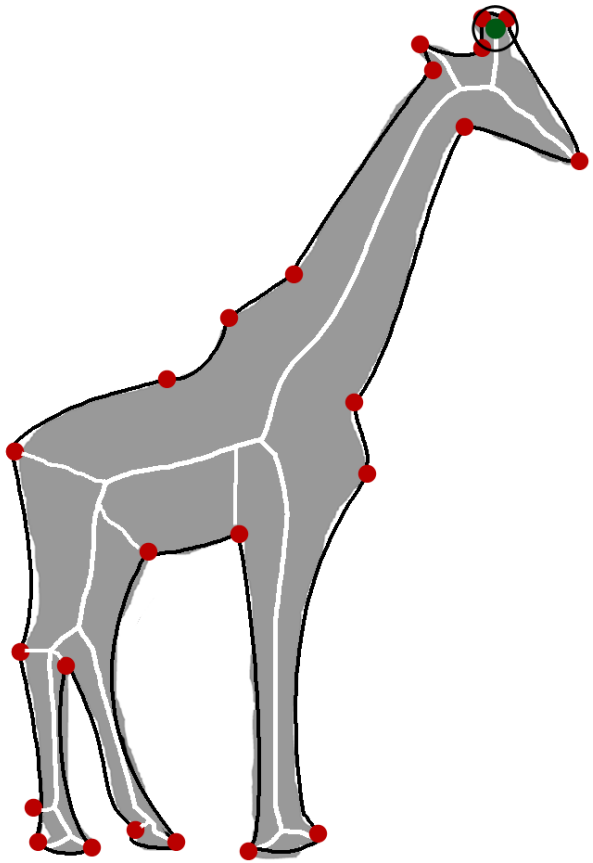
3. Check every branch chain for removal

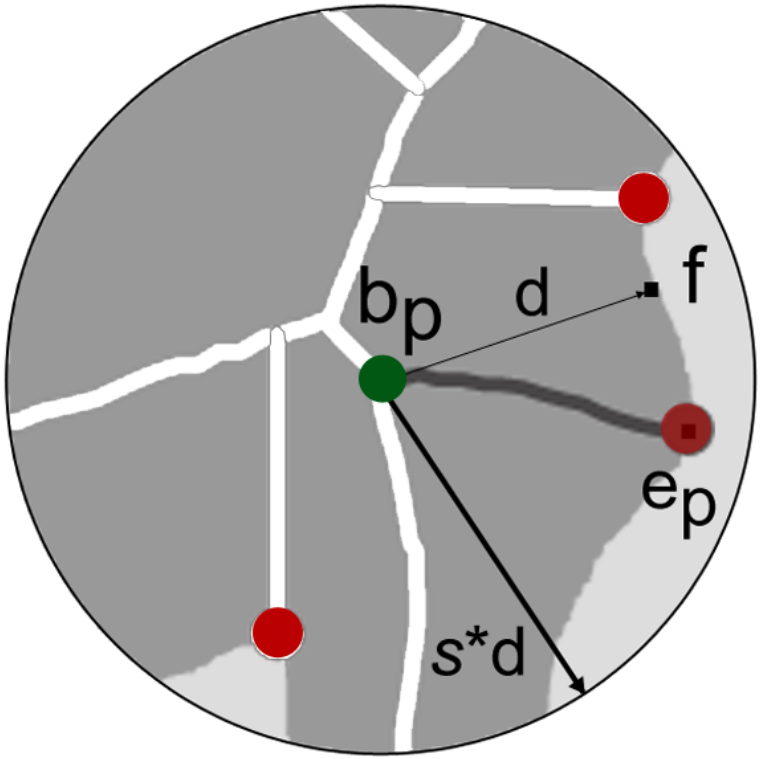




Time complexity: O(n)
n: skeleton pixels

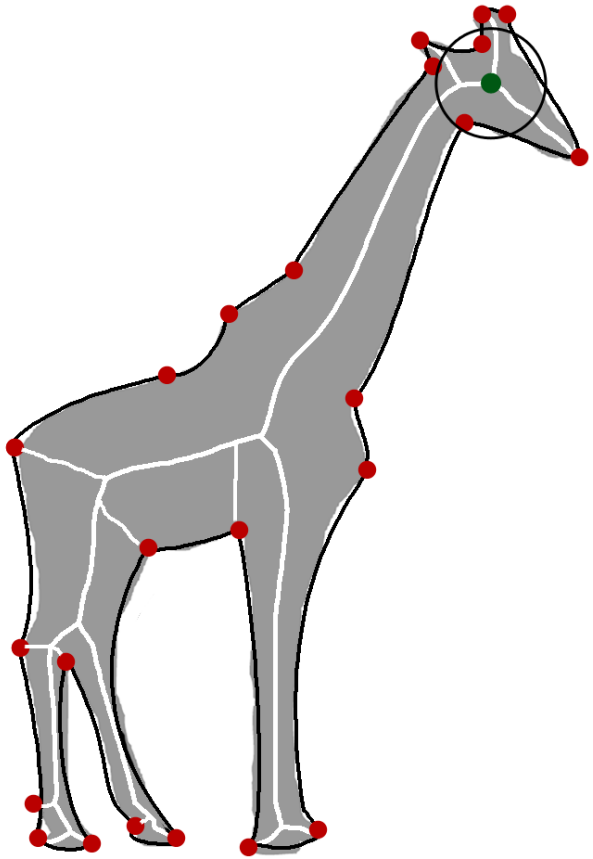
3. Check every branch chain for removal

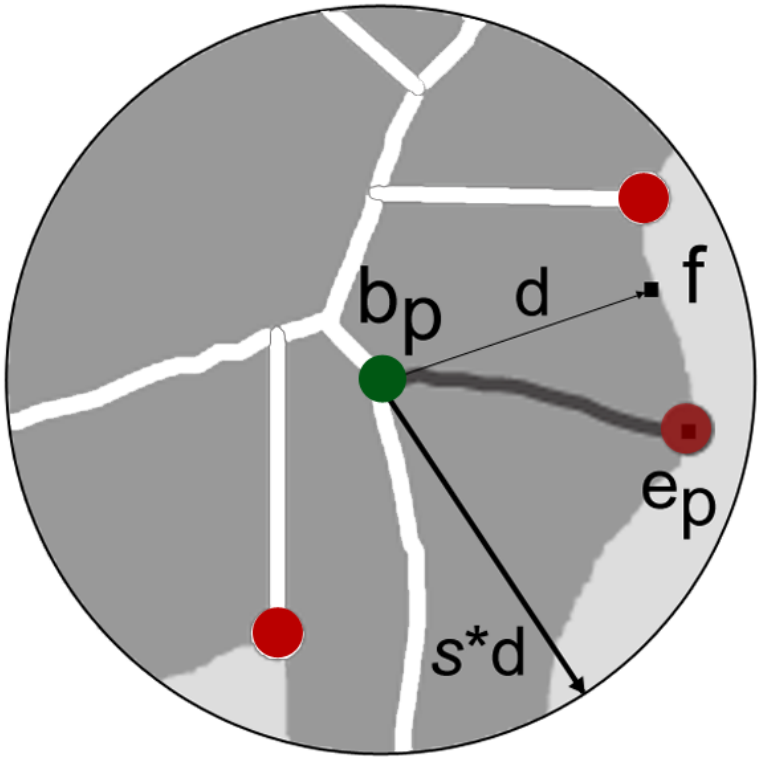




Time complexity: O(n)
n: skeleton pixels

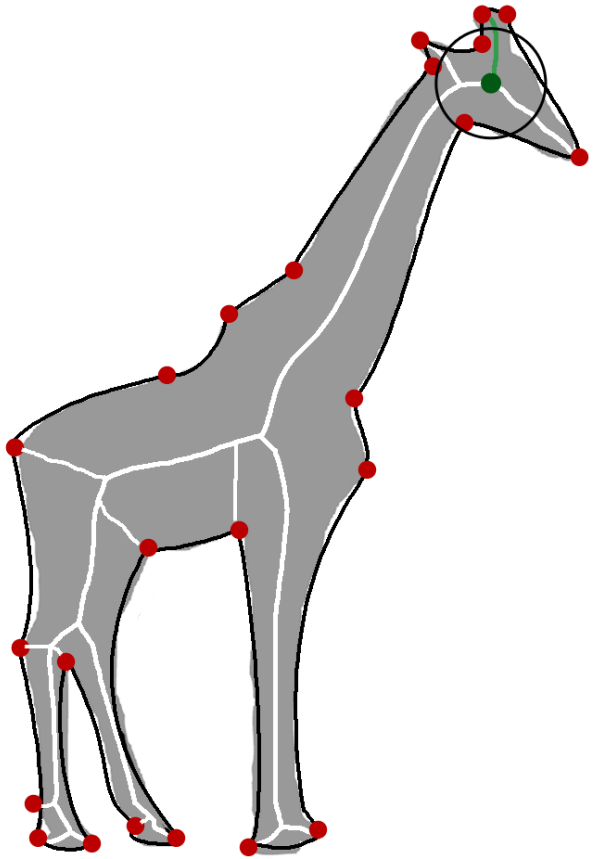
3. Check every branch chain for removal

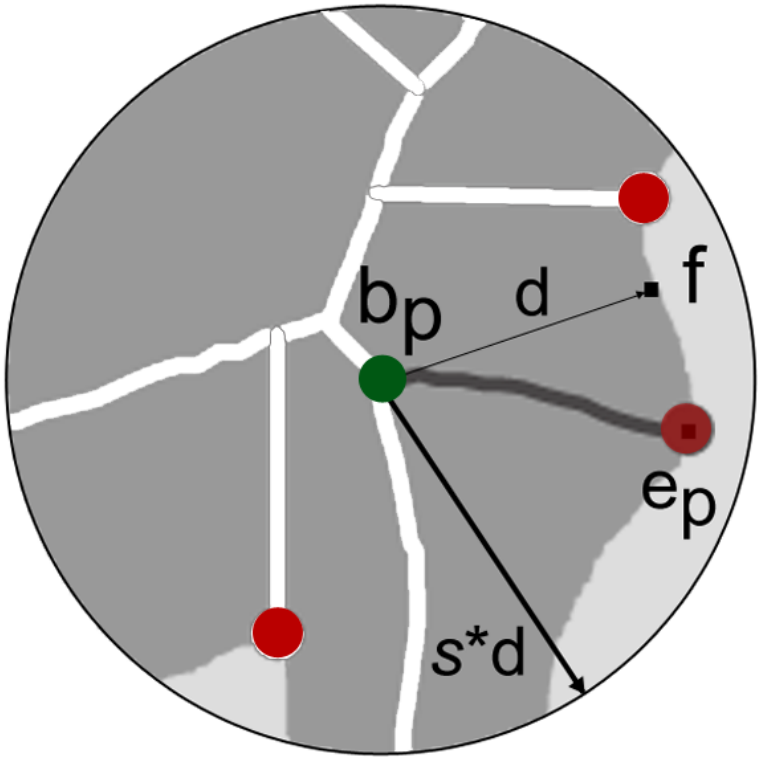




Time complexity: O(n)
n: skeleton pixels

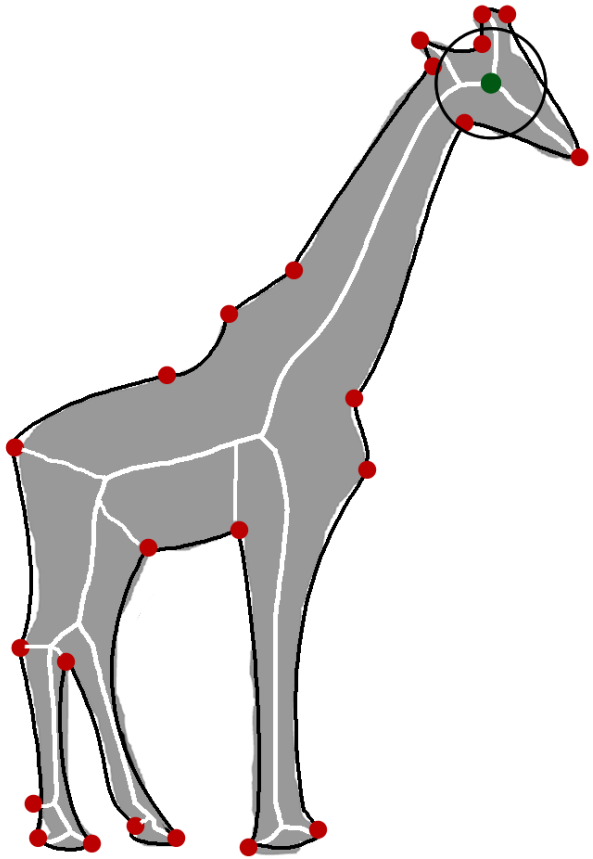
3. Check every branch chain for removal

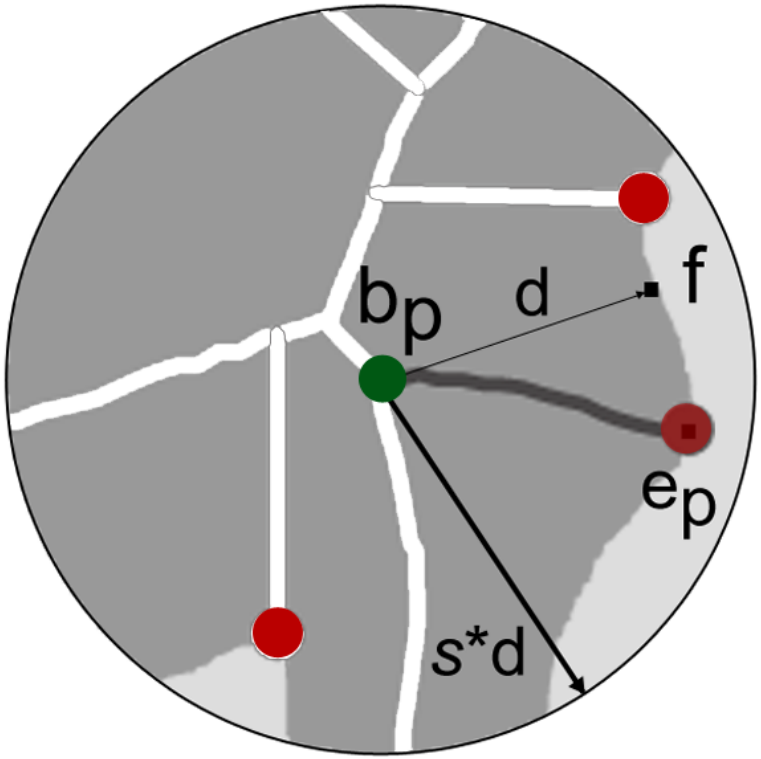




Time complexity: O(n)
n: skeleton pixels

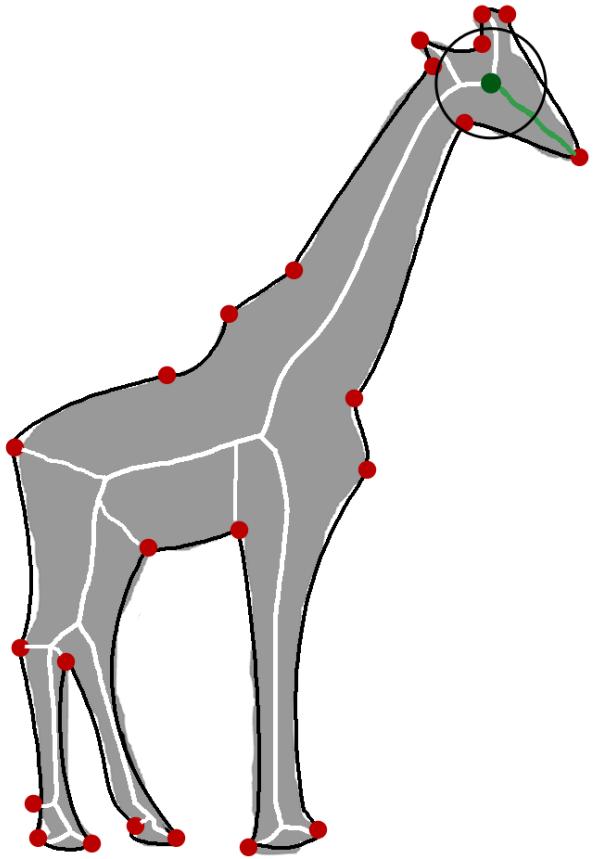
3. Check every branch chain for removal

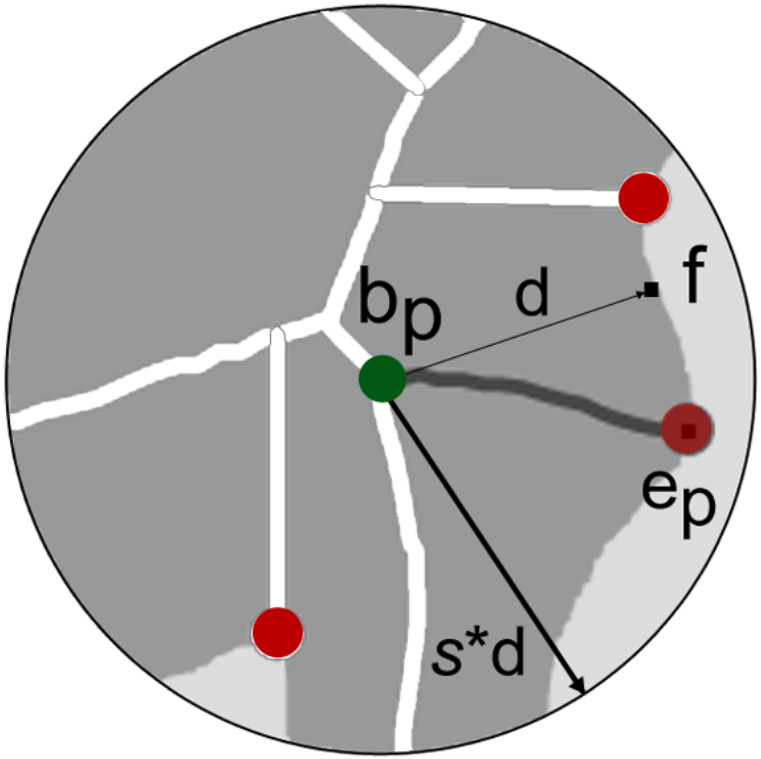




Time complexity: O(n)
n: skeleton pixels

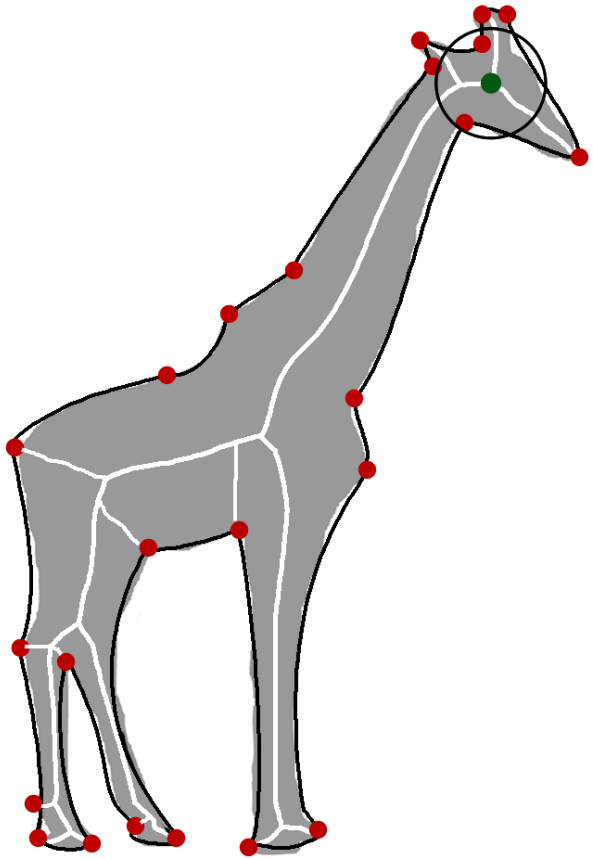
3. Check every branch chain for removal

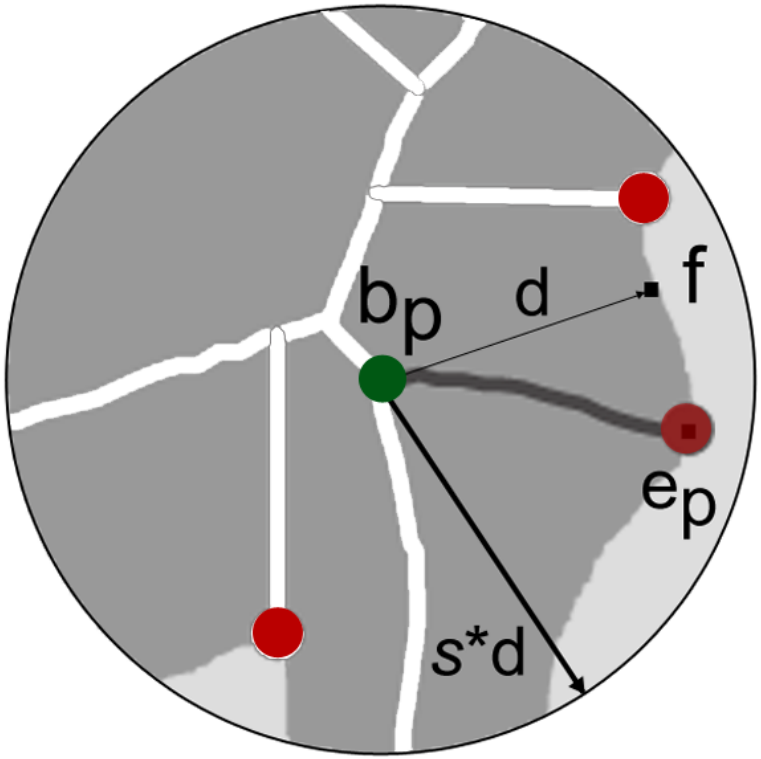




Time complexity: O(n)
n: skeleton pixels

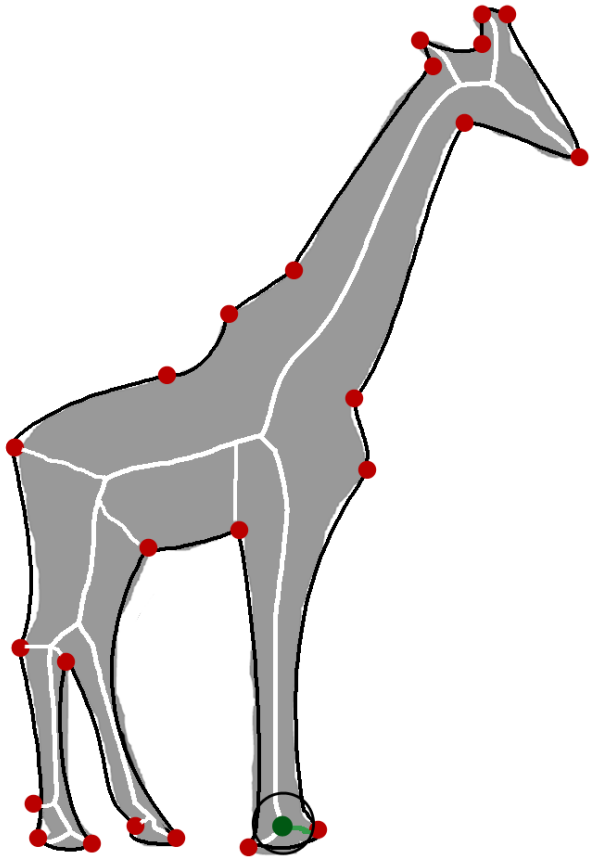
3. Check every branch chain for removal

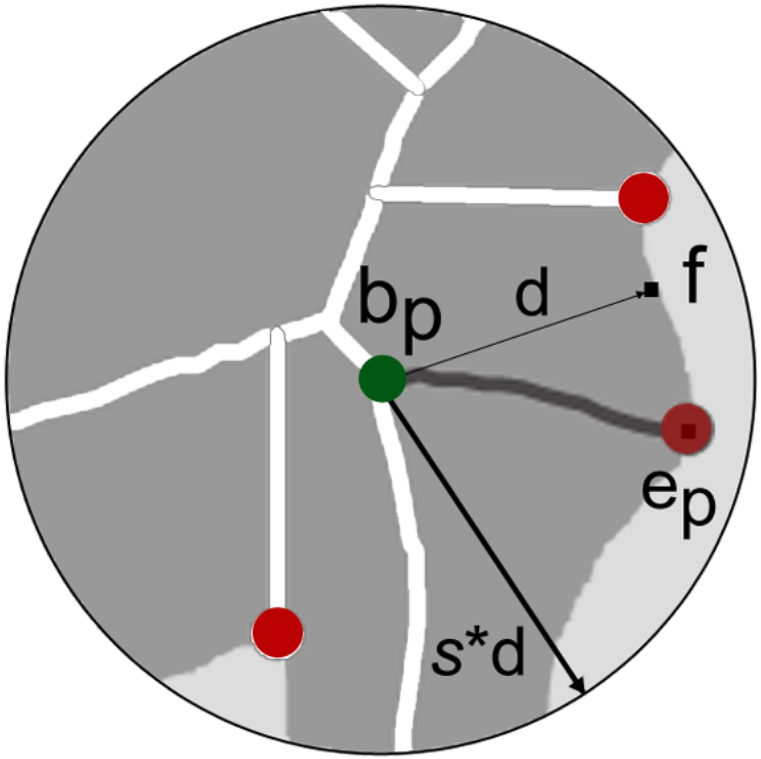




Time complexity: O(n)
n: skeleton pixels

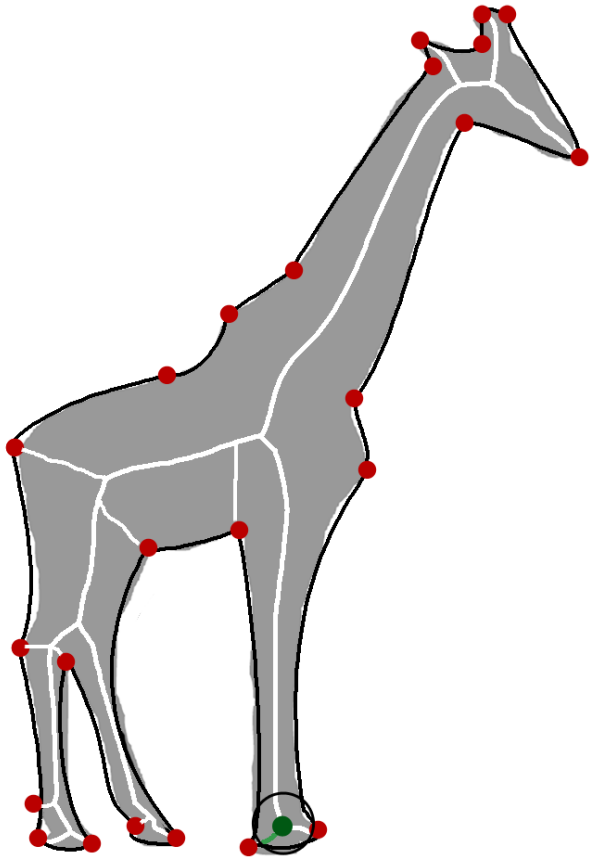
3. Check every branch chain for removal

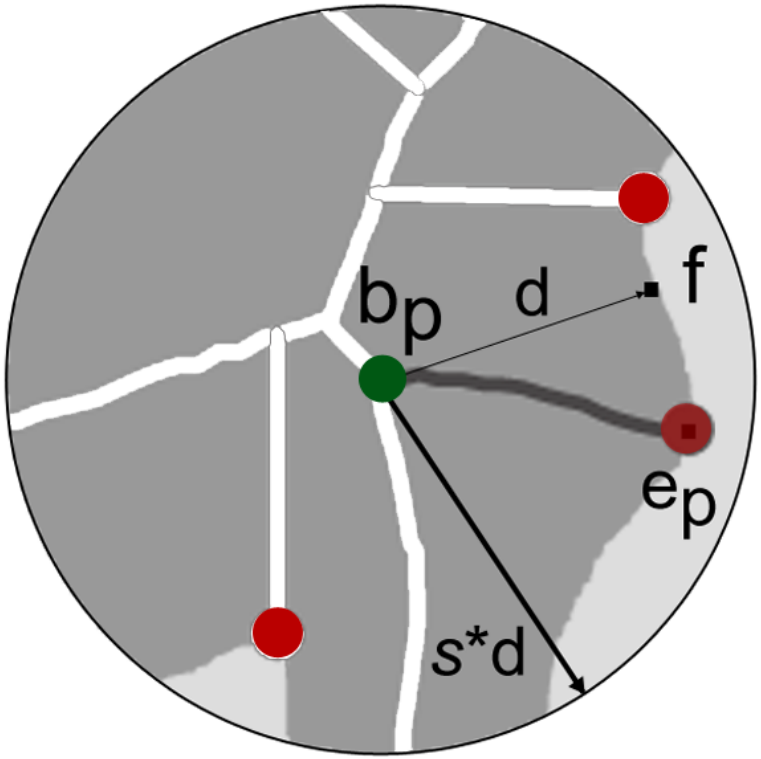




Time complexity: O(n)
n: skeleton pixels

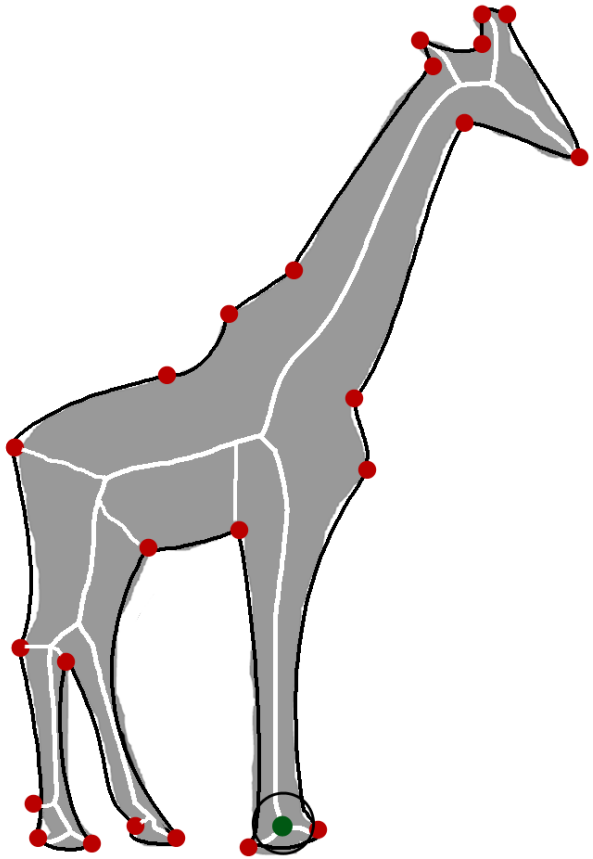
3. Check every branch chain for removal

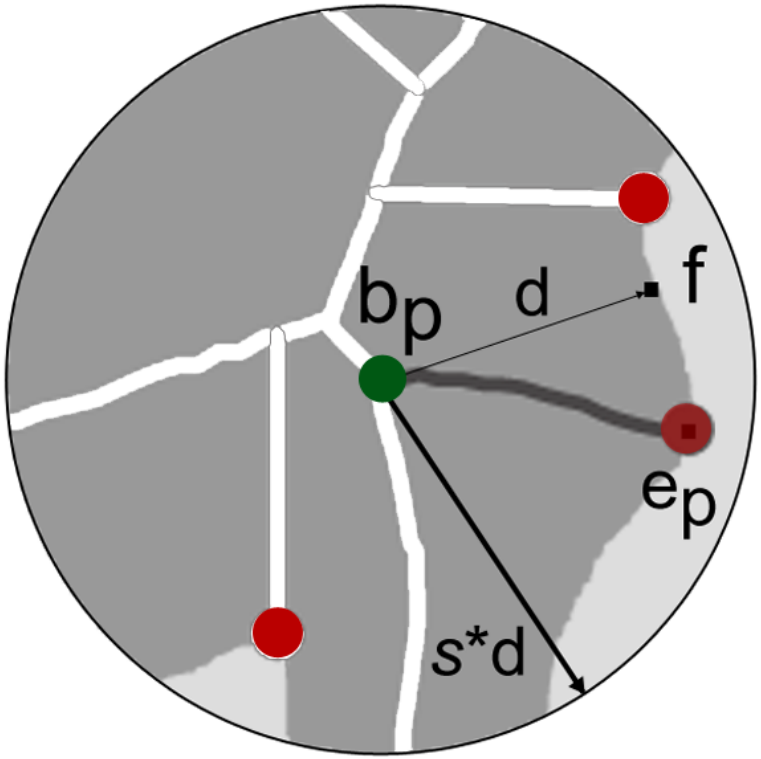




Time complexity: O(n)
n: skeleton pixels

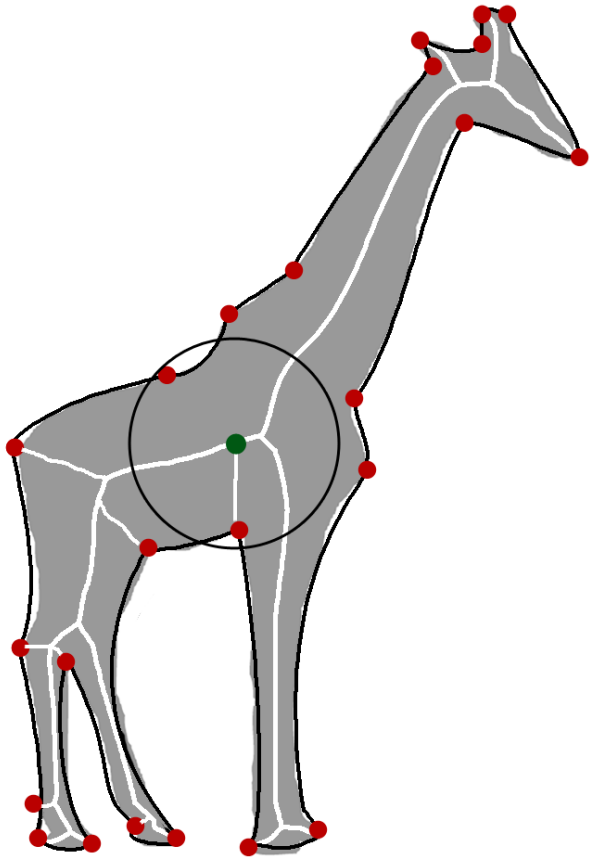
3. Check every branch chain for removal

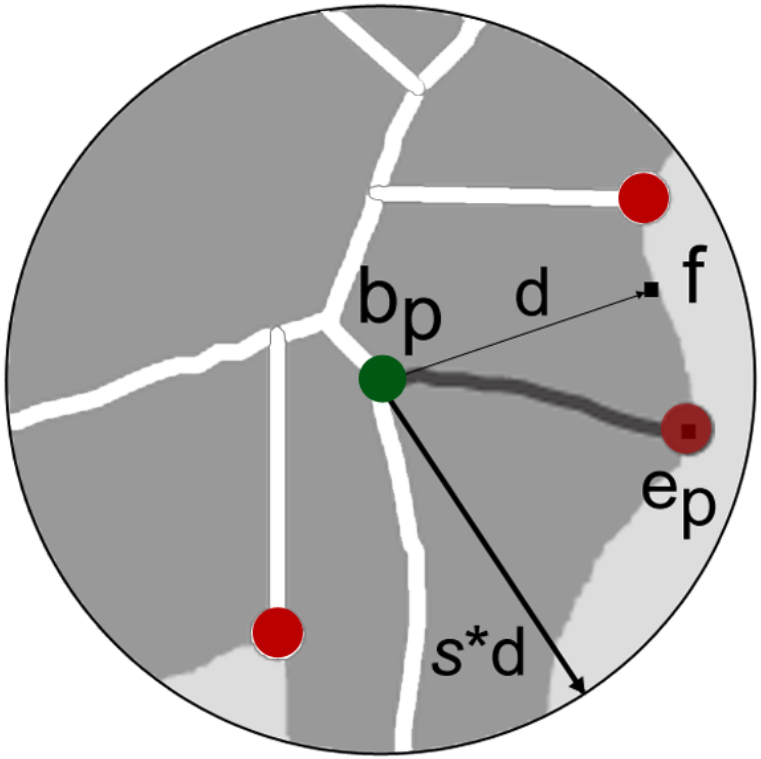




Time complexity: O(n)
n: skeleton pixels

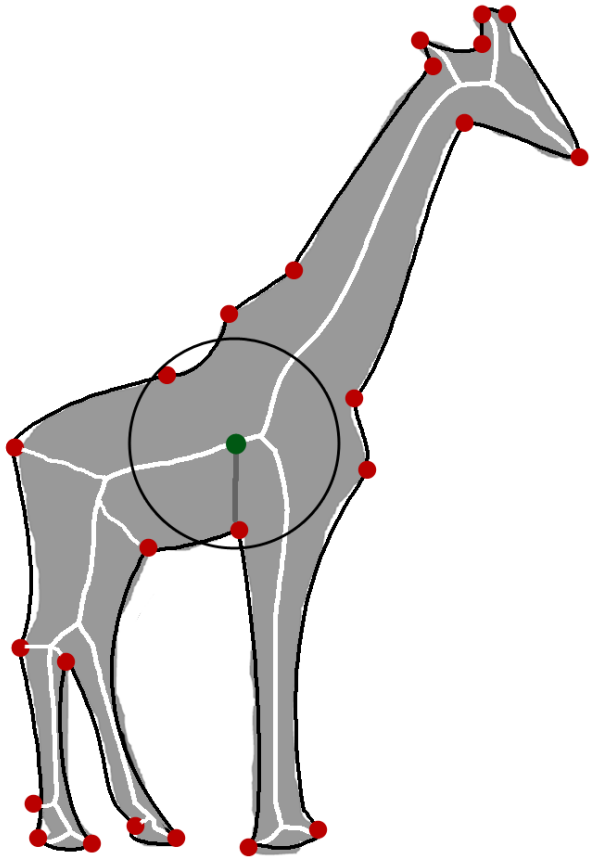
3. Check every branch chain for removal

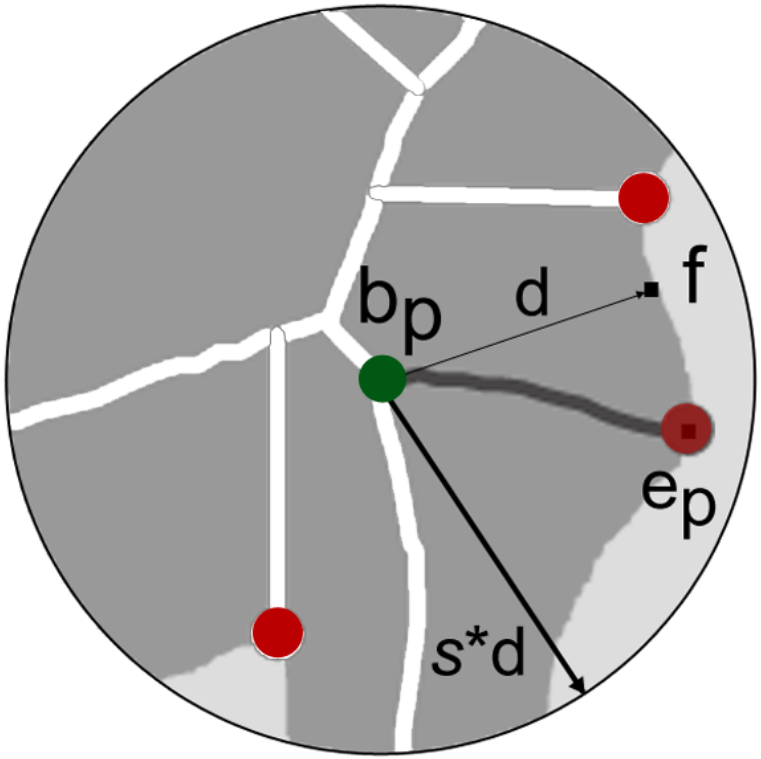




Time complexity: O(n)
n: skeleton pixels

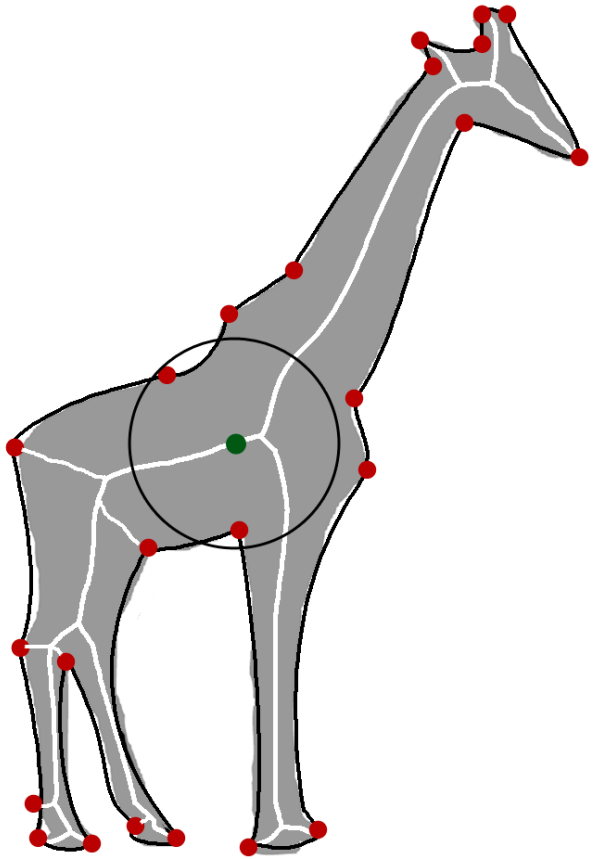
3. Check every branch chain for removal

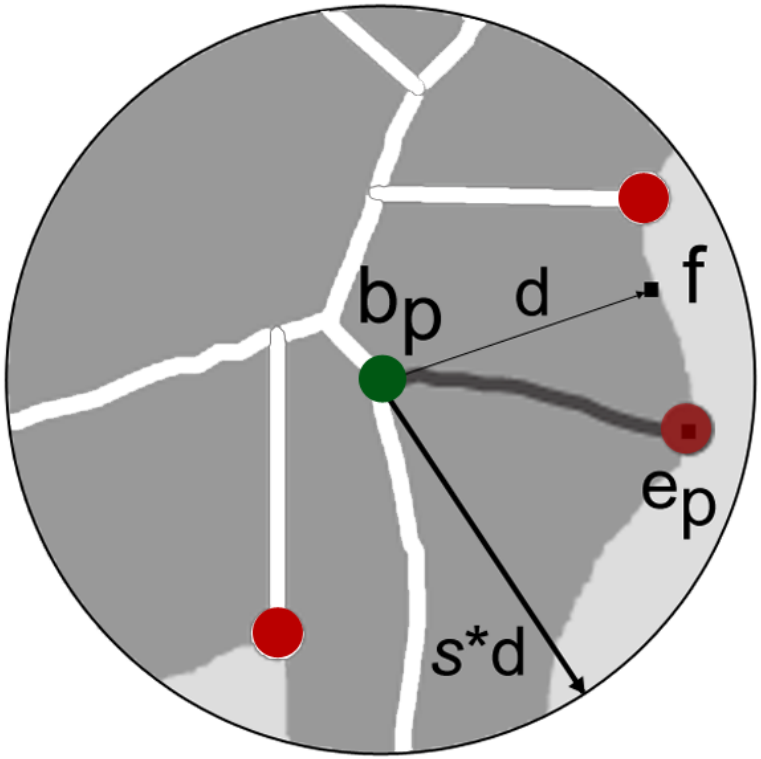




Time complexity: O(n)
n: skeleton pixels

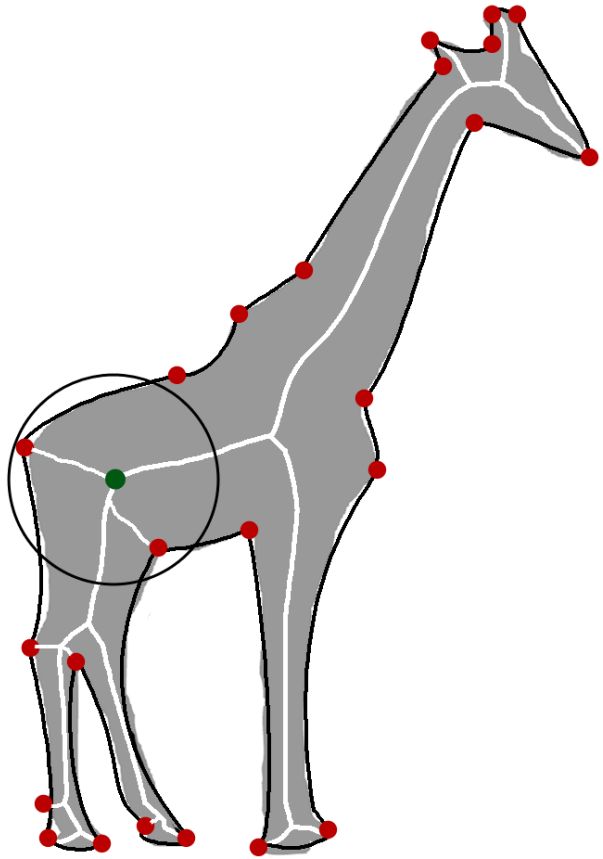
3. Check every branch chain for removal

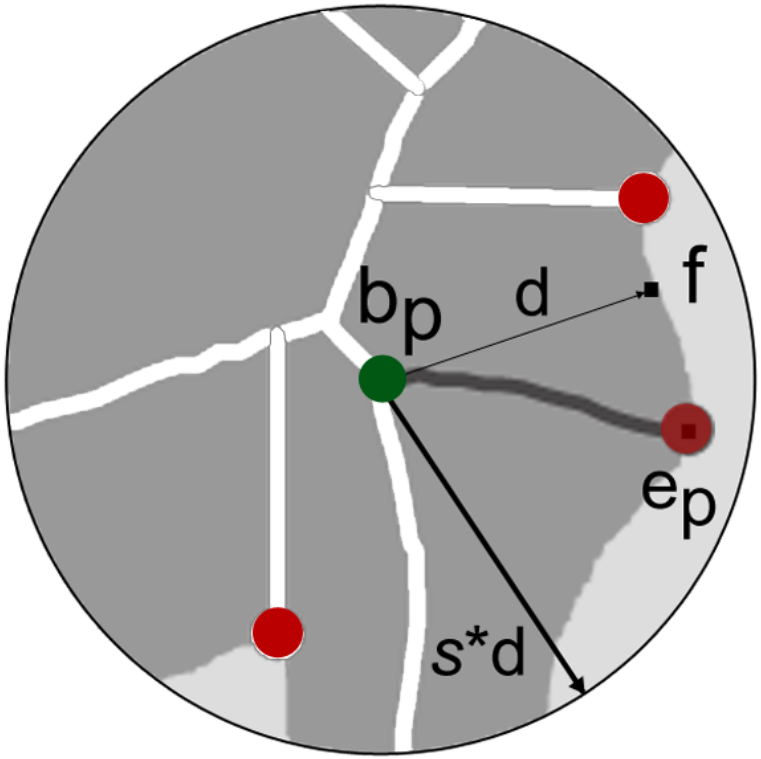




Time complexity: O(n)
n: skeleton pixels

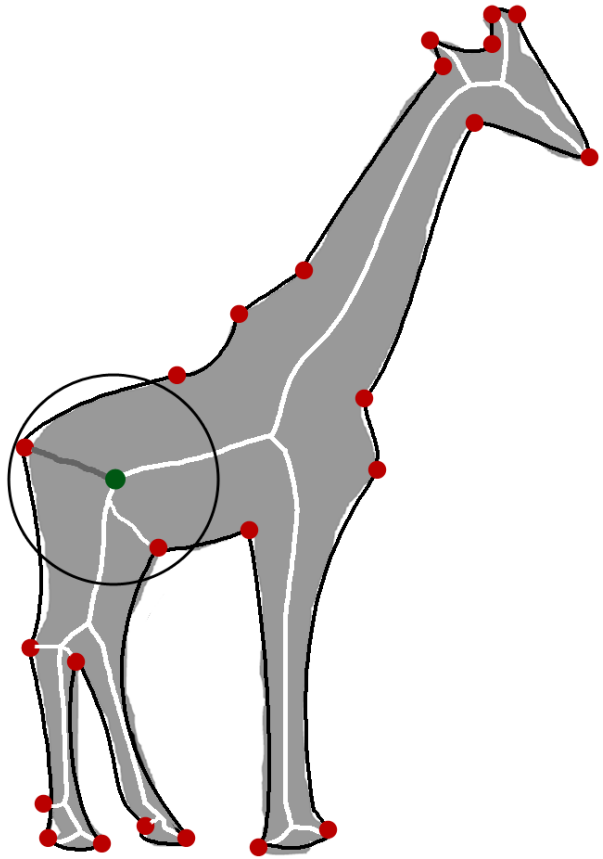
3. Check every branch chain for removal

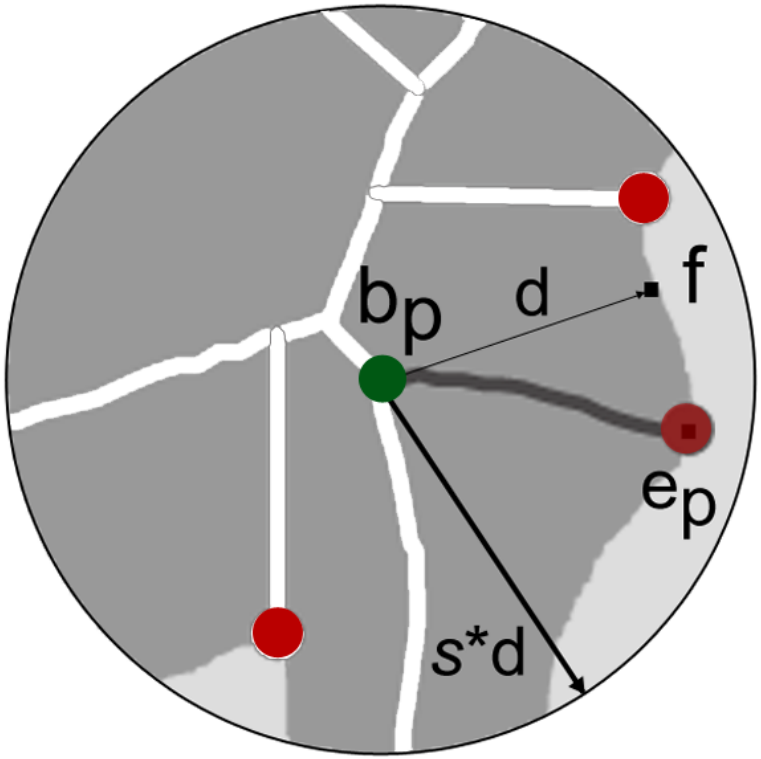




Time complexity: O(n)
n: skeleton pixels

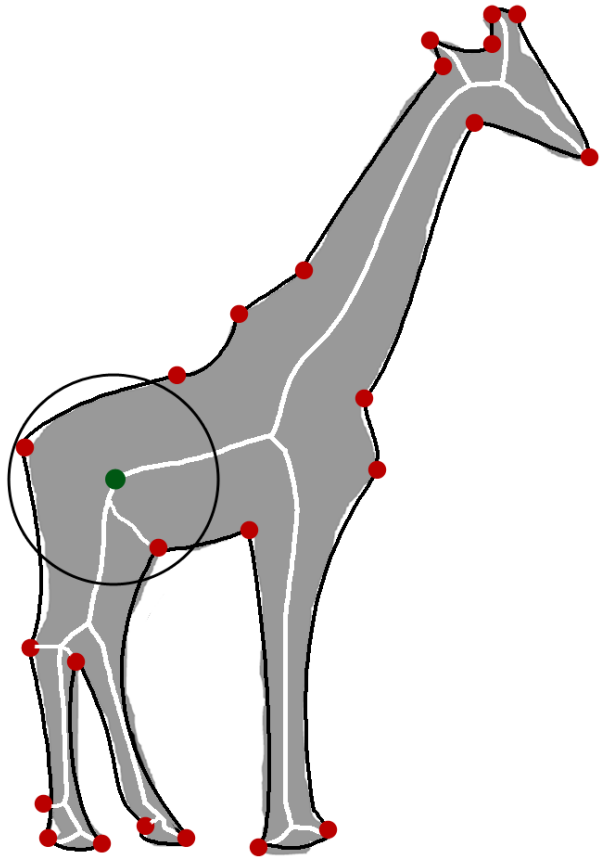
3. Check every branch chain for removal

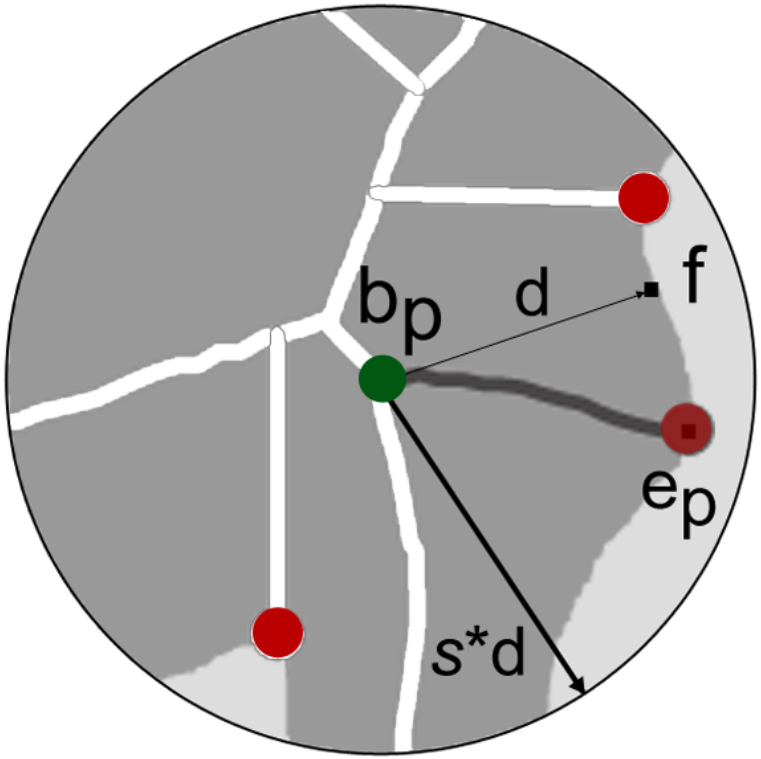




Time complexity: O(n)
n: skeleton pixels

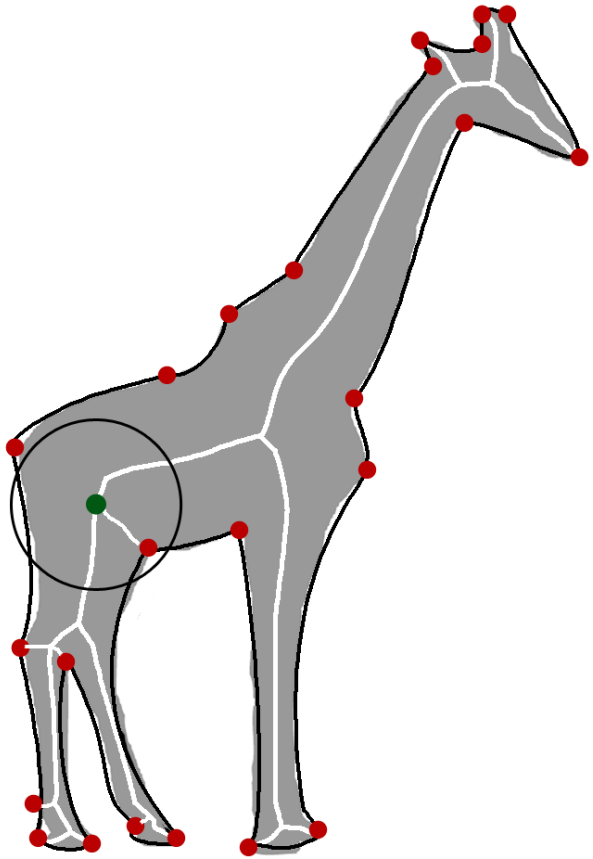
3. Check every branch chain for removal

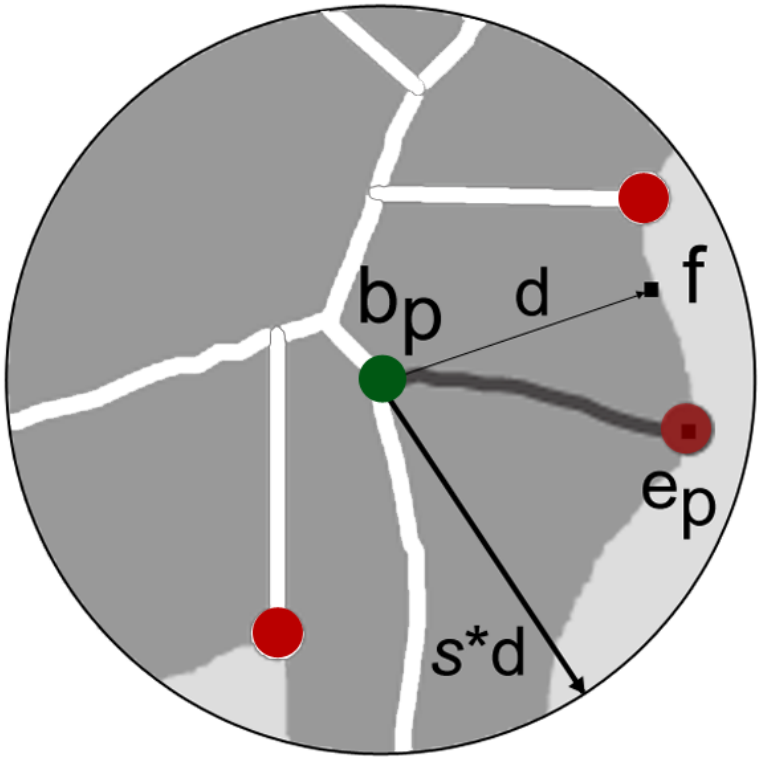




Time complexity: O(n)
n: skeleton pixels

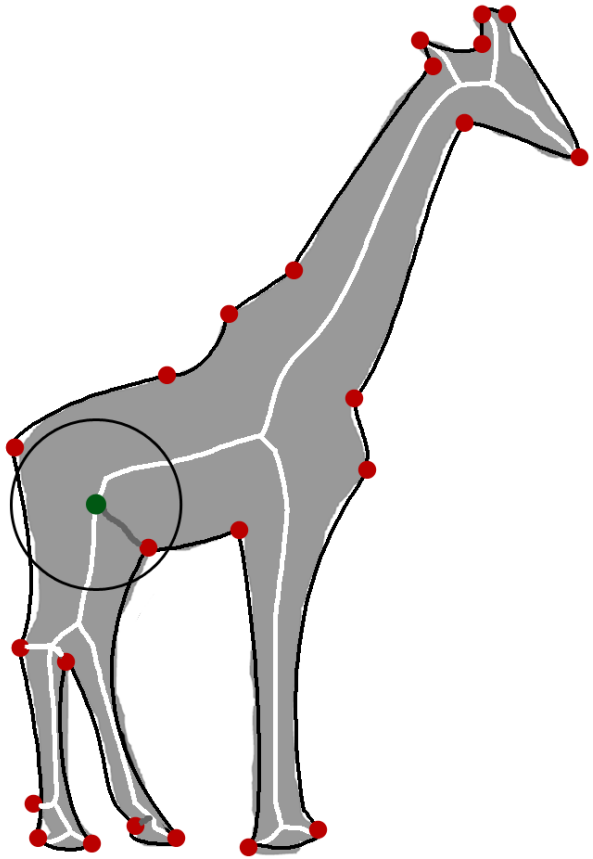
3. Check every branch chain for removal

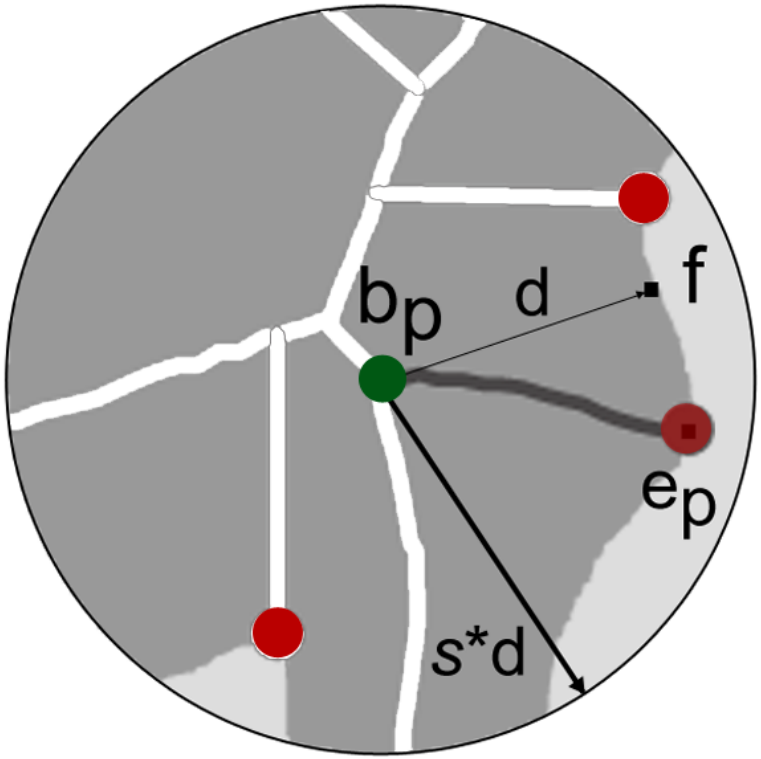




Time complexity: O(n)
n: skeleton pixels

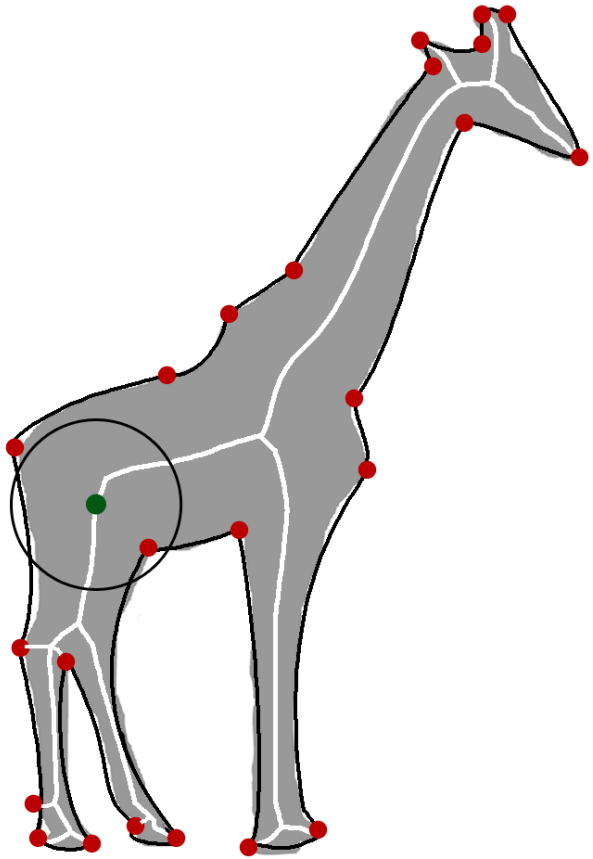
3. Check every branch chain for removal

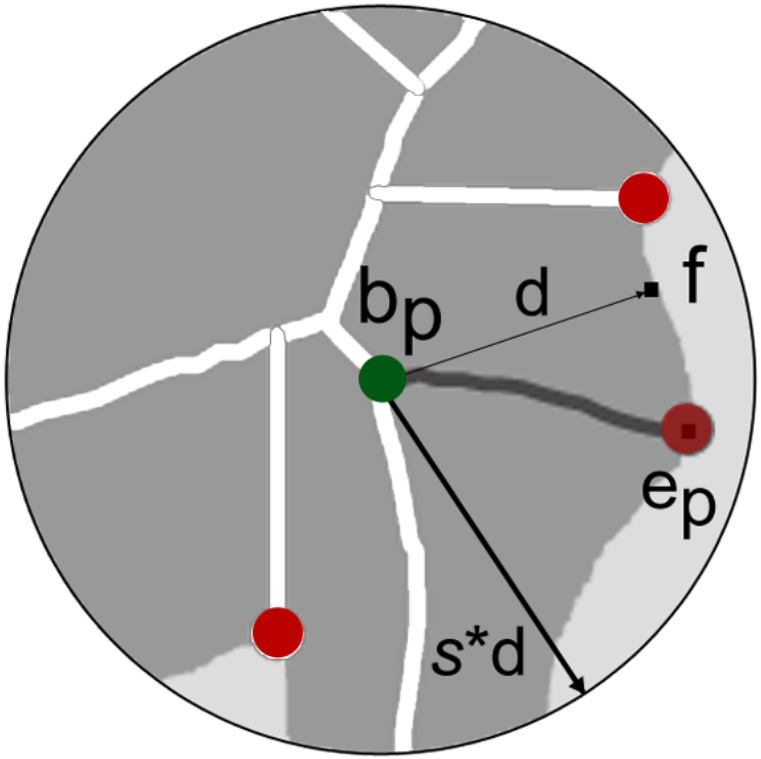




Time complexity: O(n)
n: skeleton pixels

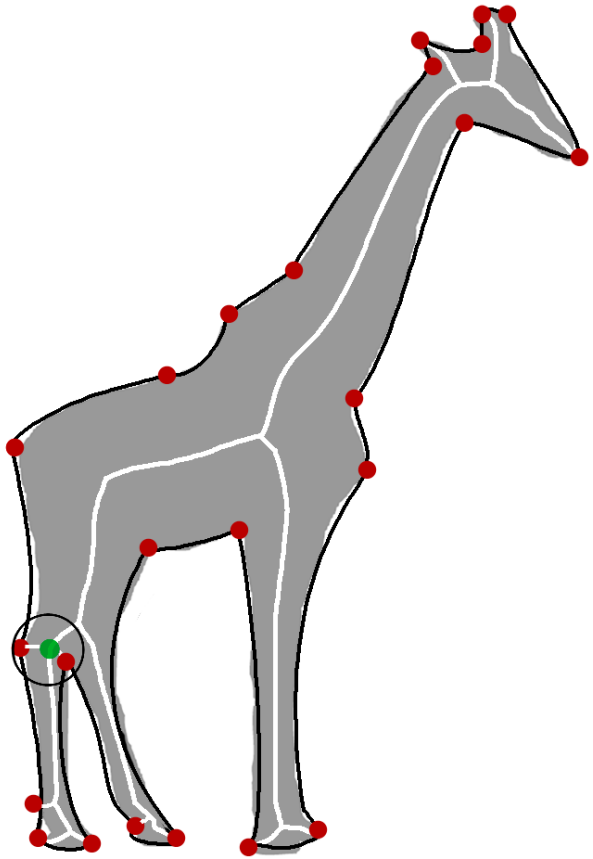
3. Check every branch chain for removal

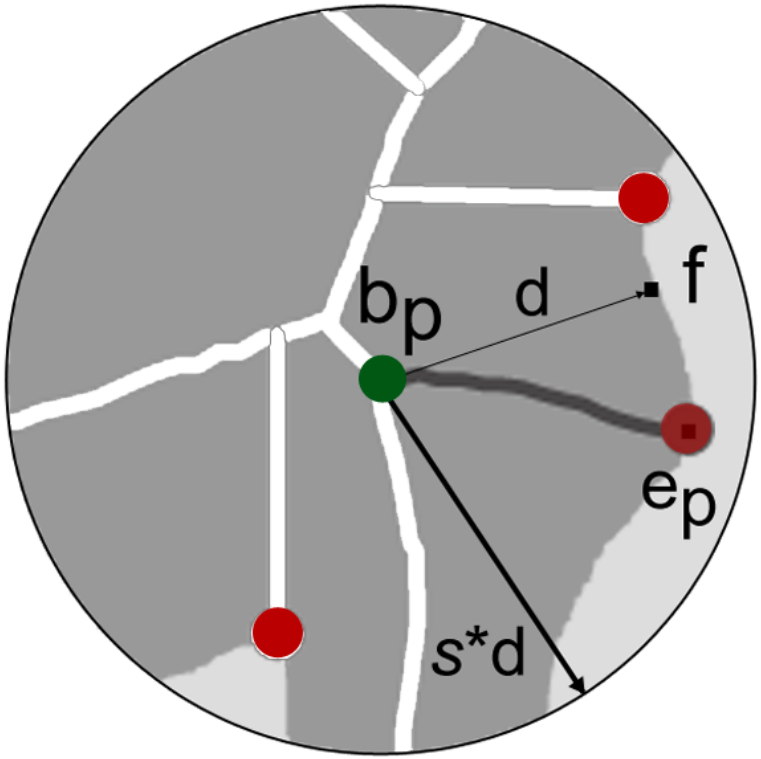




Time complexity: O(n)
n: skeleton pixels

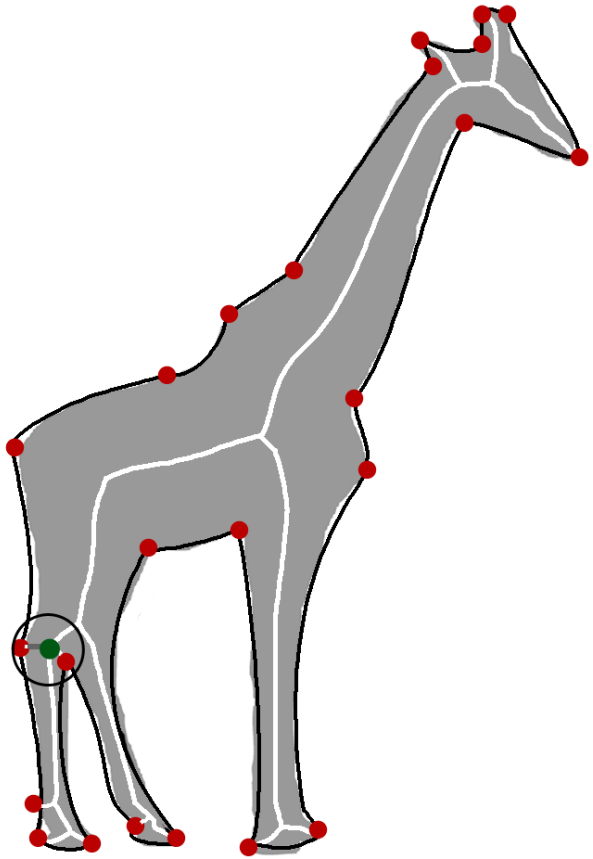
3. Check every branch chain for removal

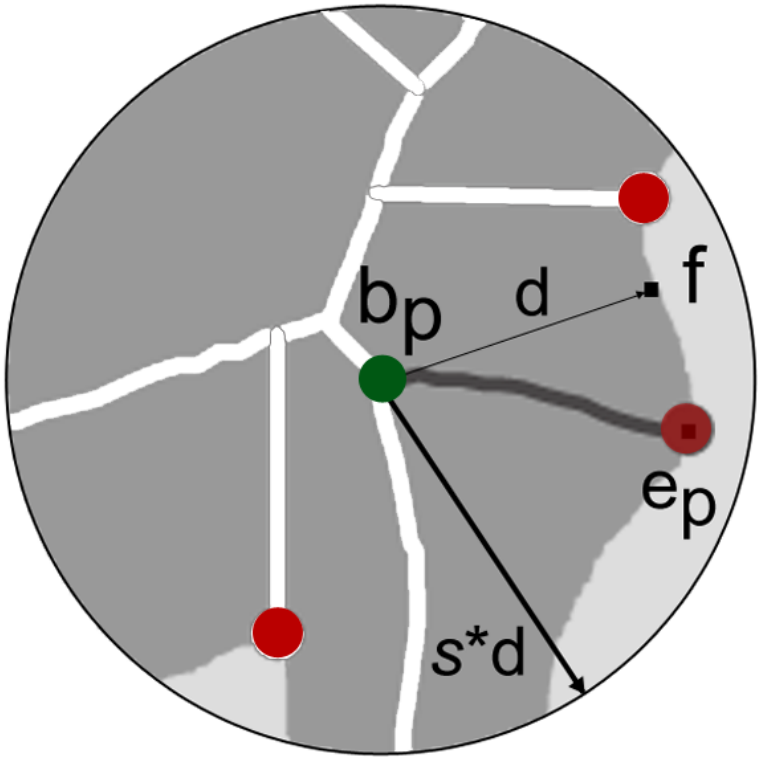




Time complexity: O(n)
n: skeleton pixels

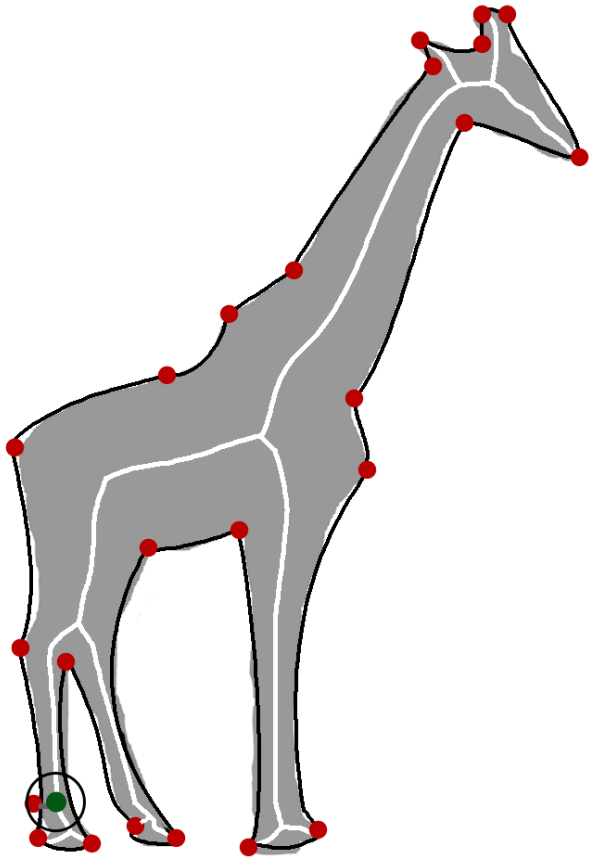
3. Check every branch chain for removal

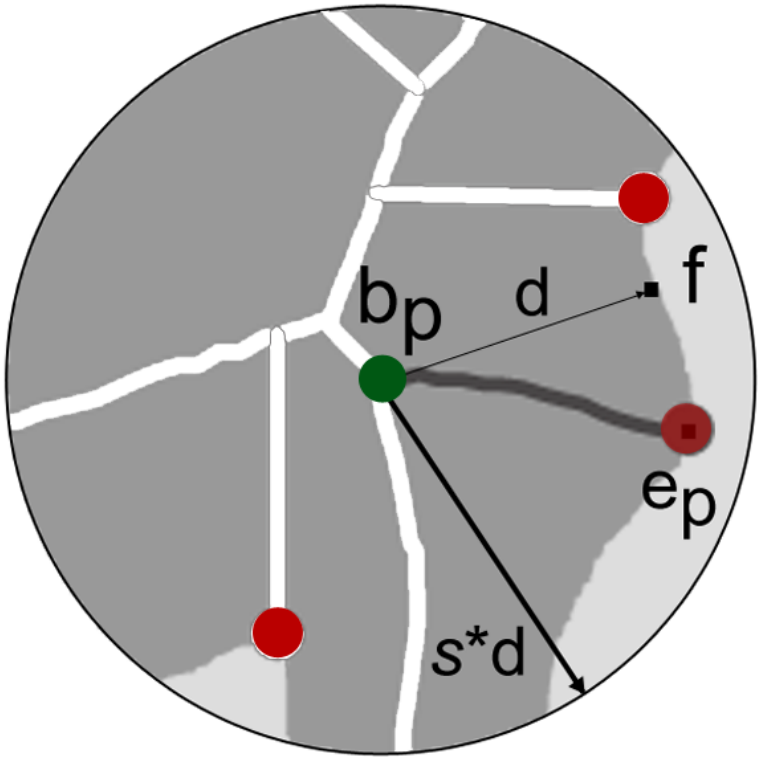




Time complexity: O(n)
n: skeleton pixels

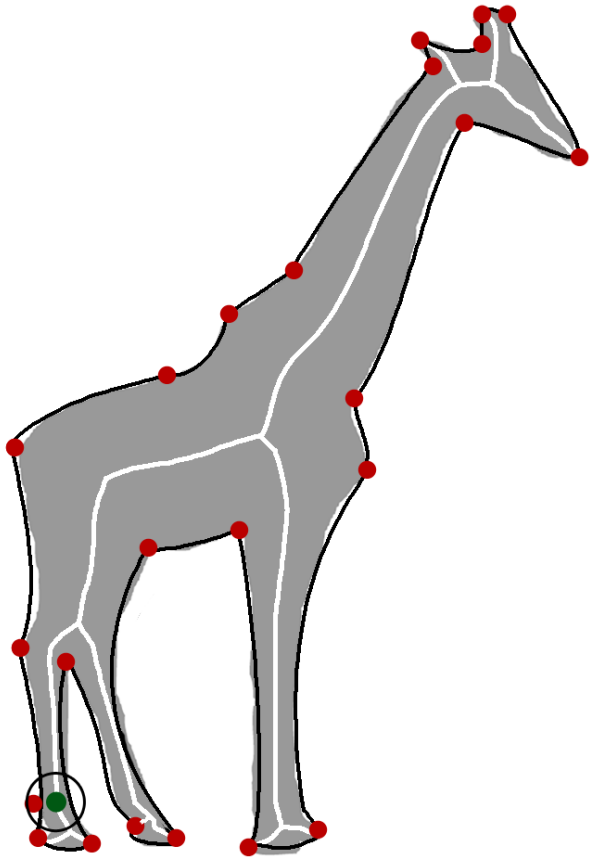
3. Check every branch chain for removal

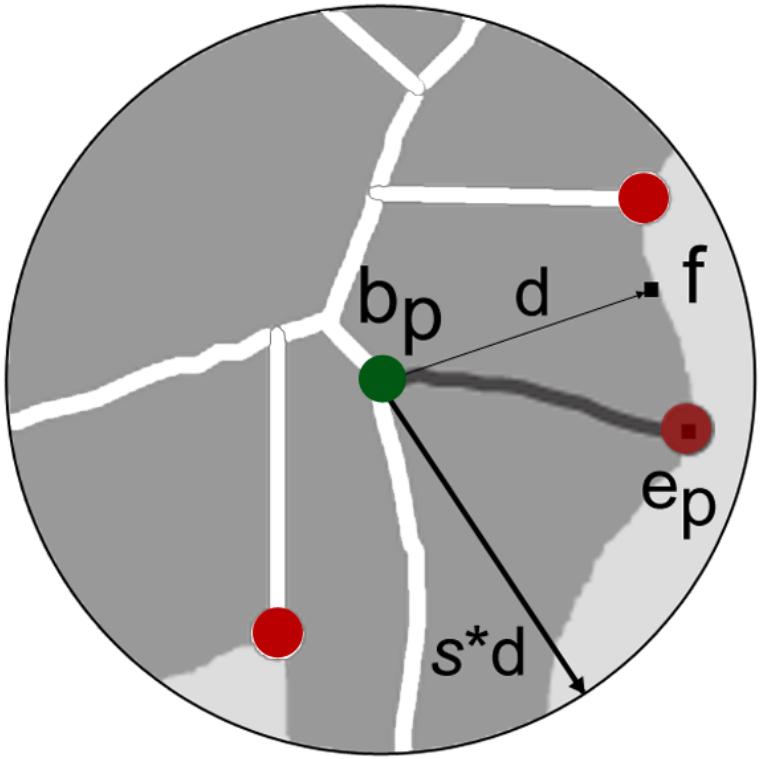




Time complexity: O(n)
n: skeleton pixels

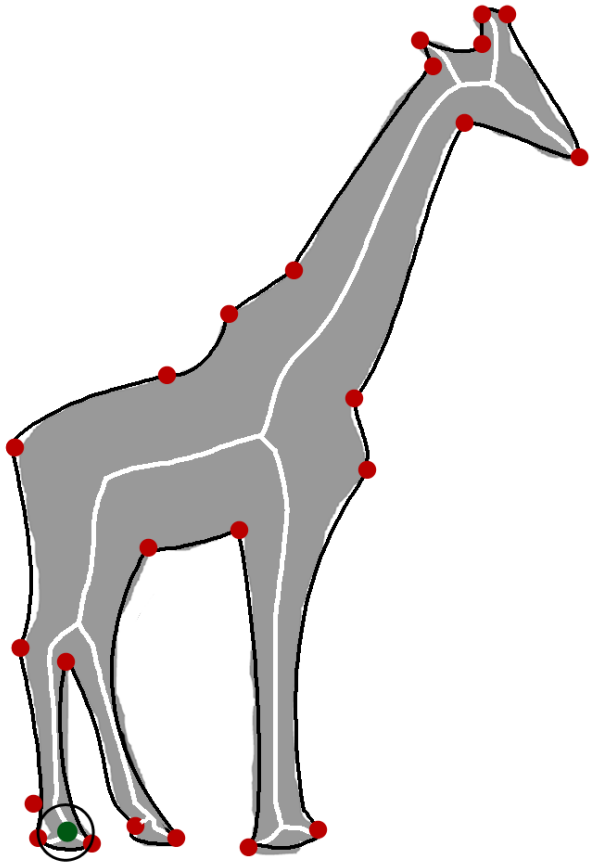
3. Check every branch chain for removal

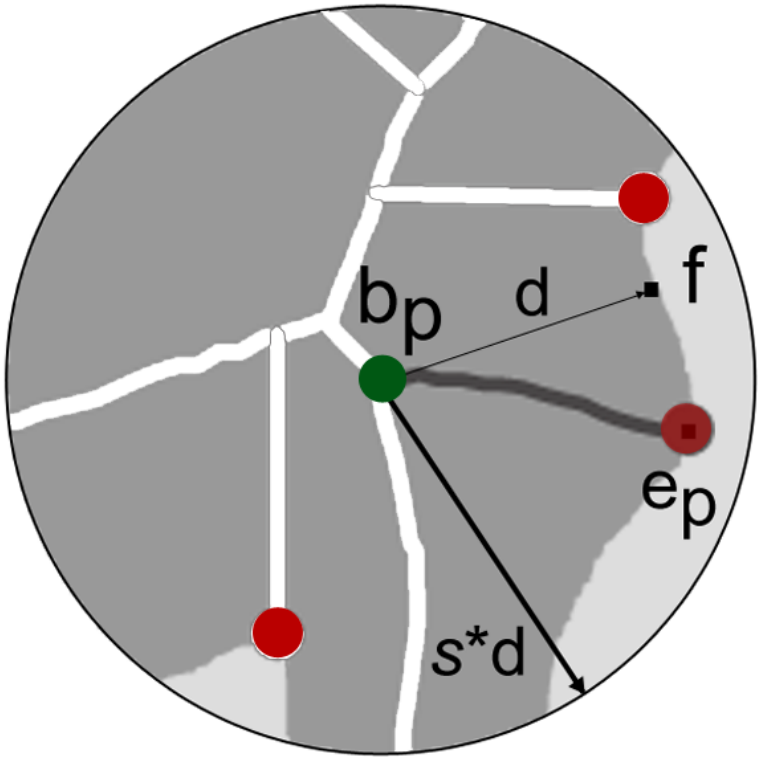




Time complexity: O(n)
n: skeleton pixels

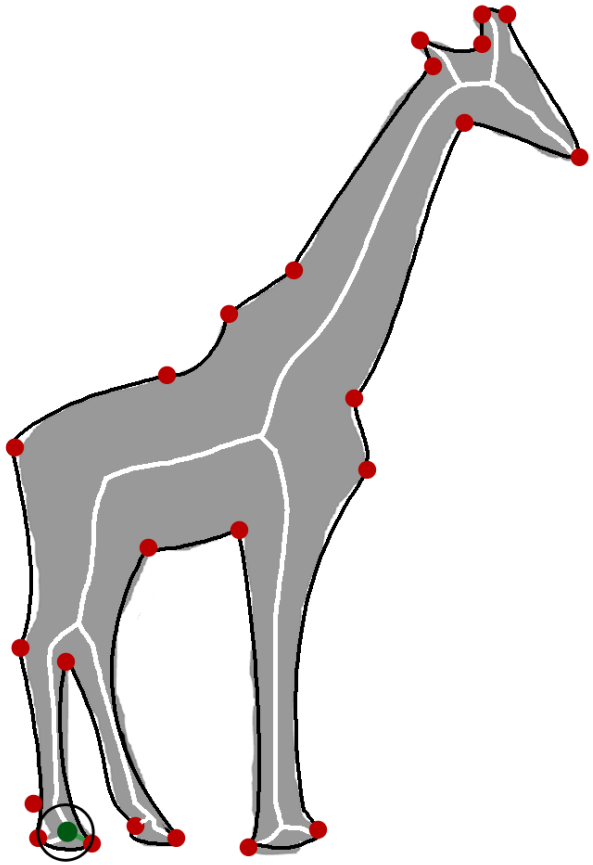
3. Check every branch chain for removal

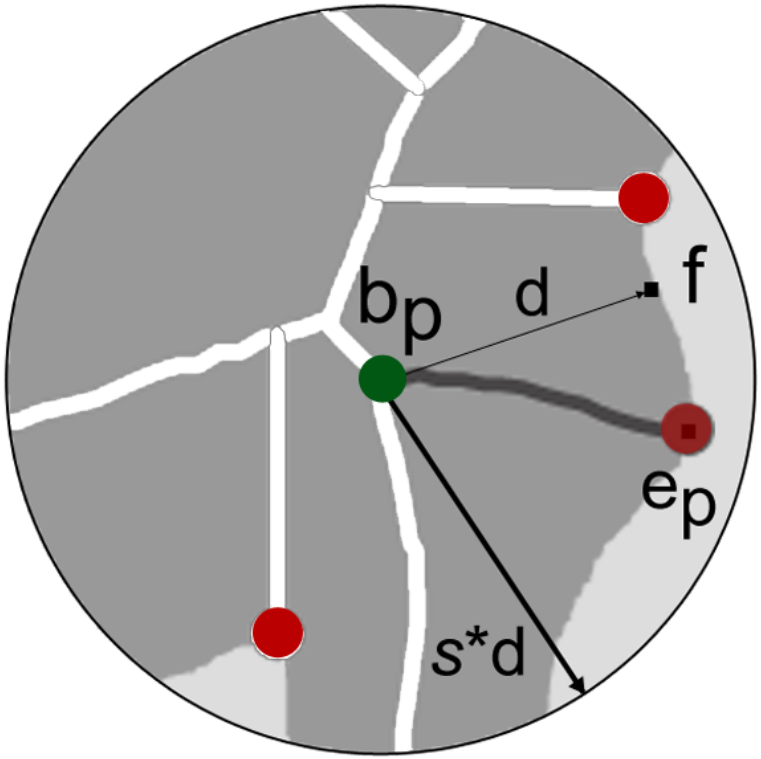




Time complexity: O(n)
n: skeleton pixels

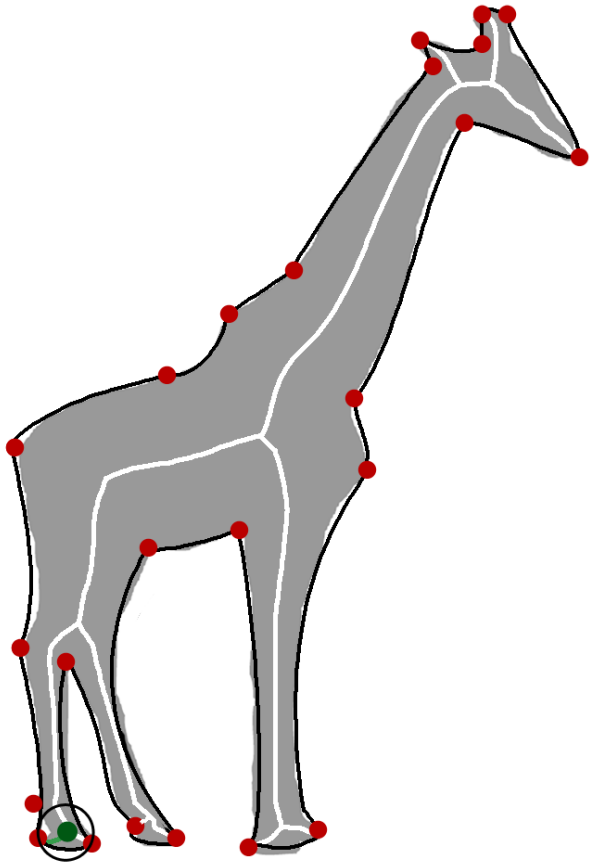
3. Check every branch chain for removal

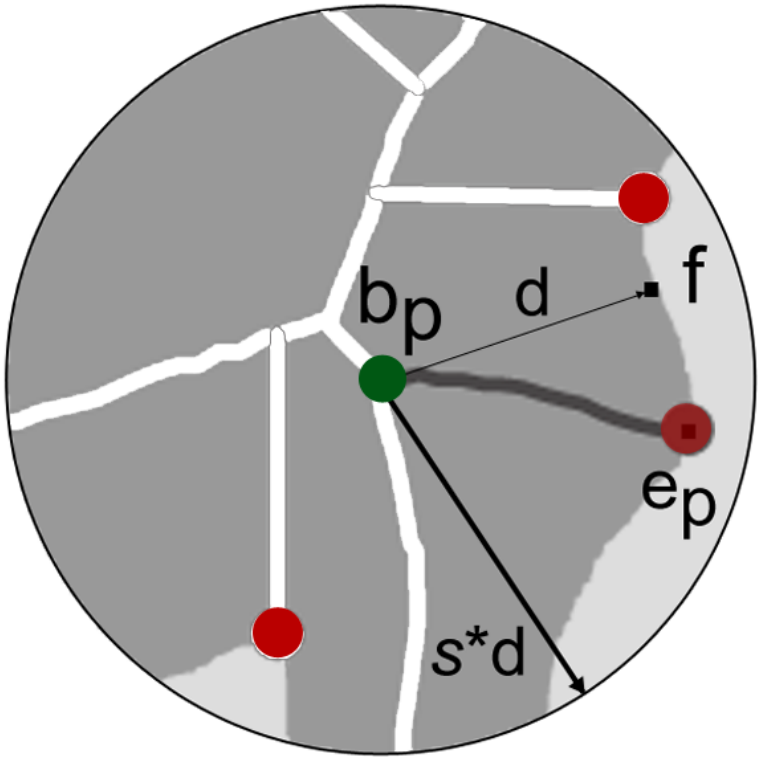




Time complexity: O(n)
n: skeleton pixels

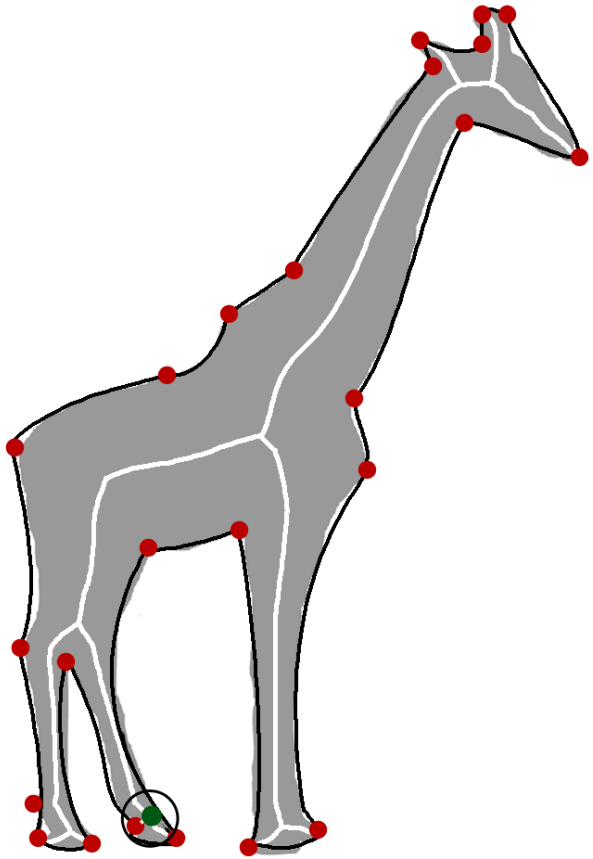
3. Check every branch chain for removal

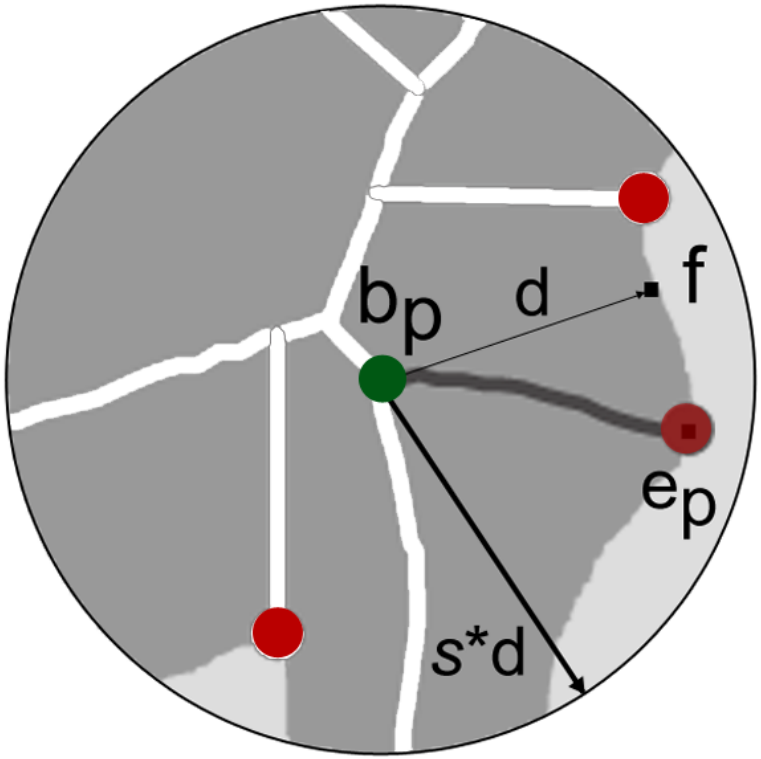




Time complexity: O(n)
n: skeleton pixels

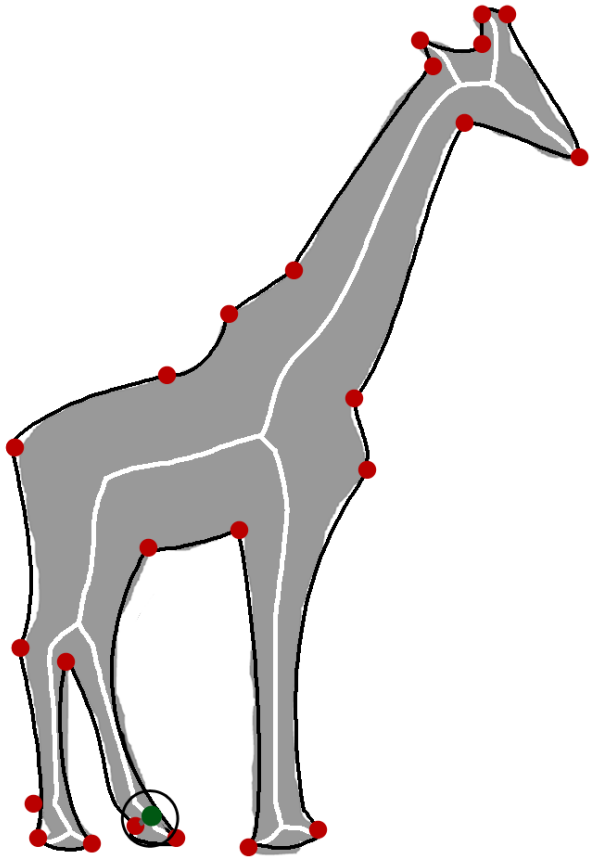
3. Check every branch chain for removal

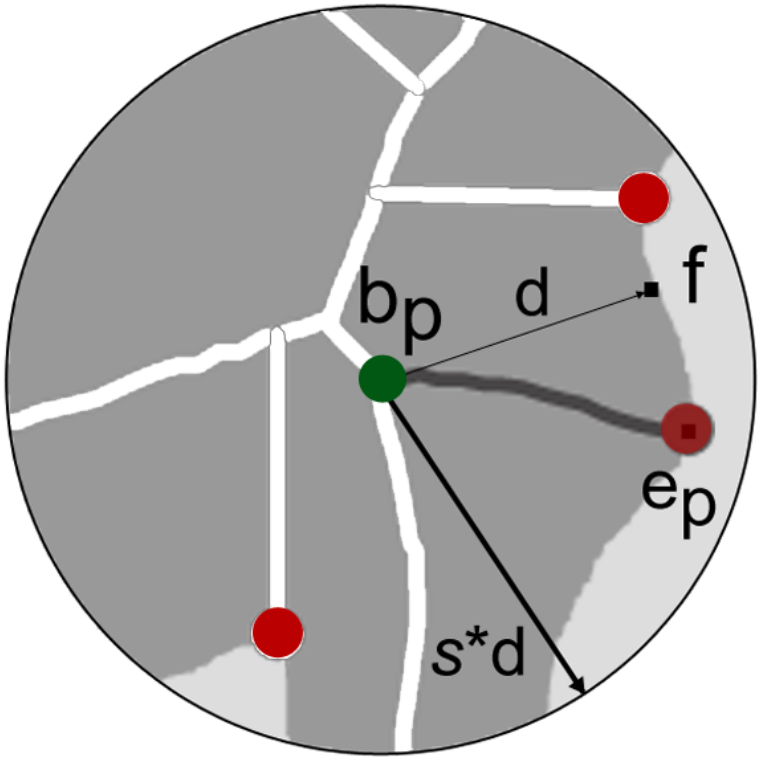




Time complexity: O(n)
n: skeleton pixels

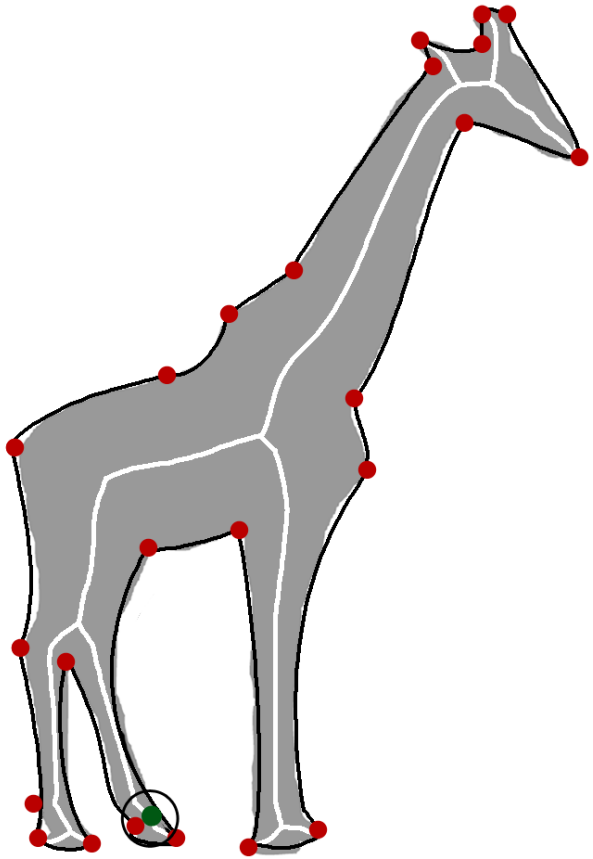
3. Check every branch chain for removal

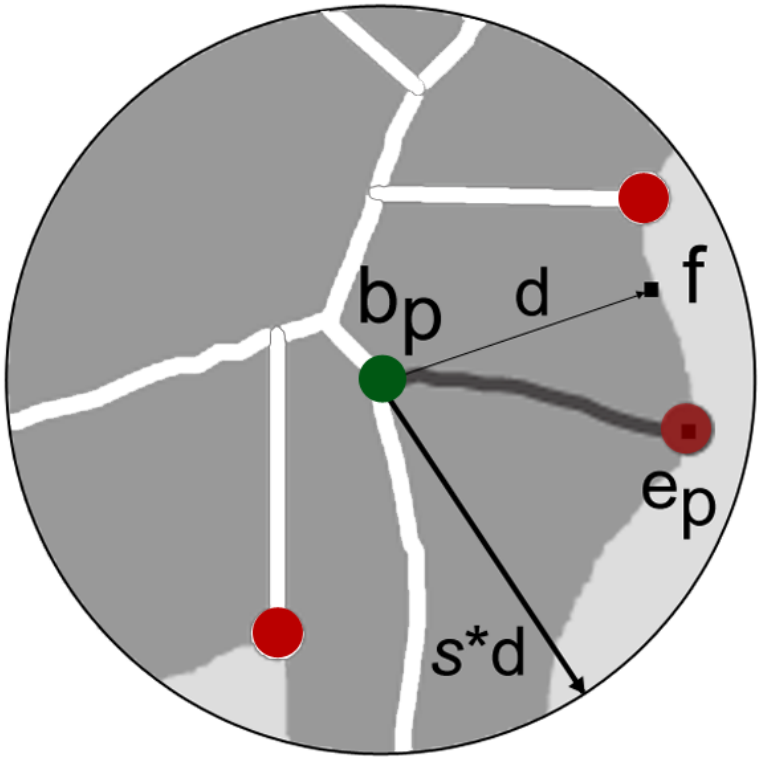




Time complexity: O(n)
n: skeleton pixels

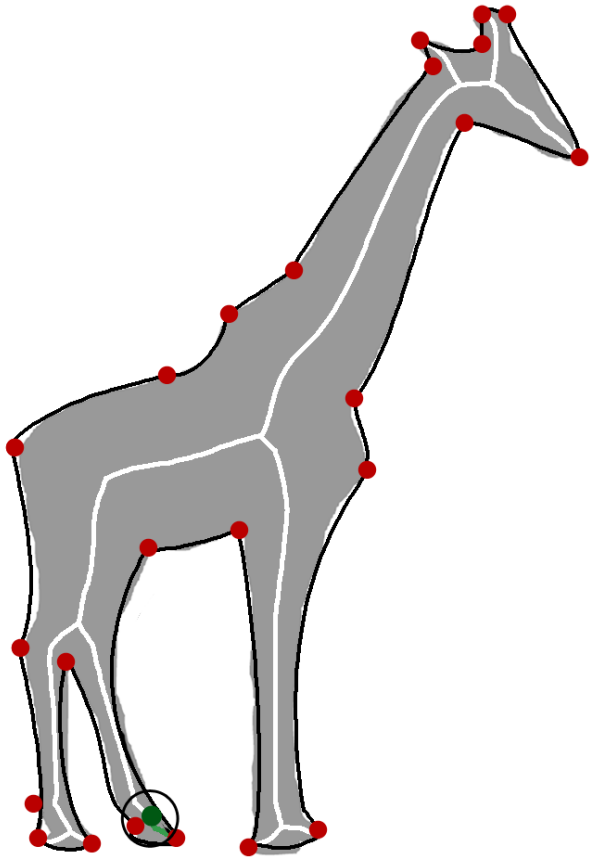
3. Check every branch chain for removal

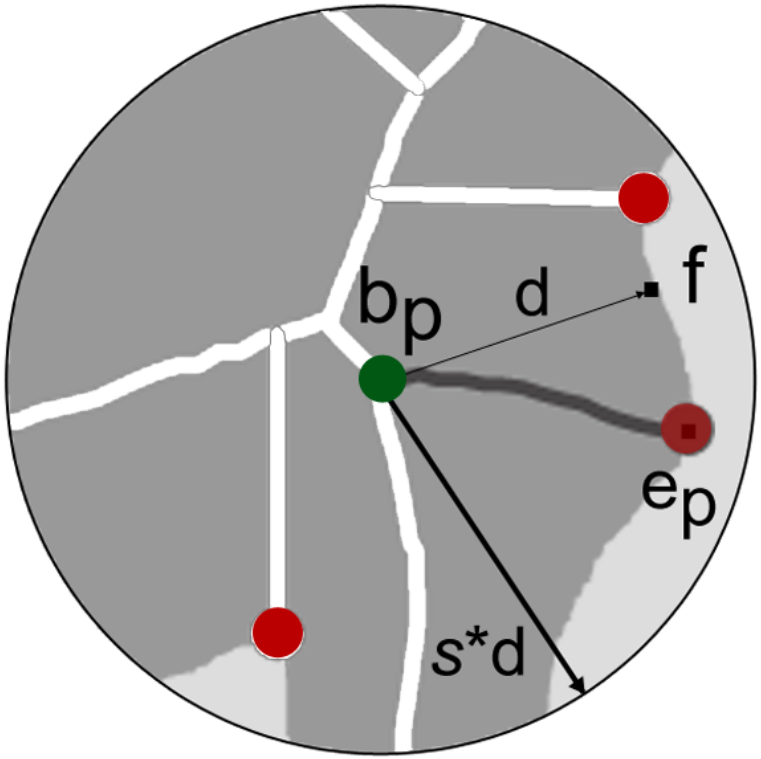




Time complexity: O(n)
n: skeleton pixels

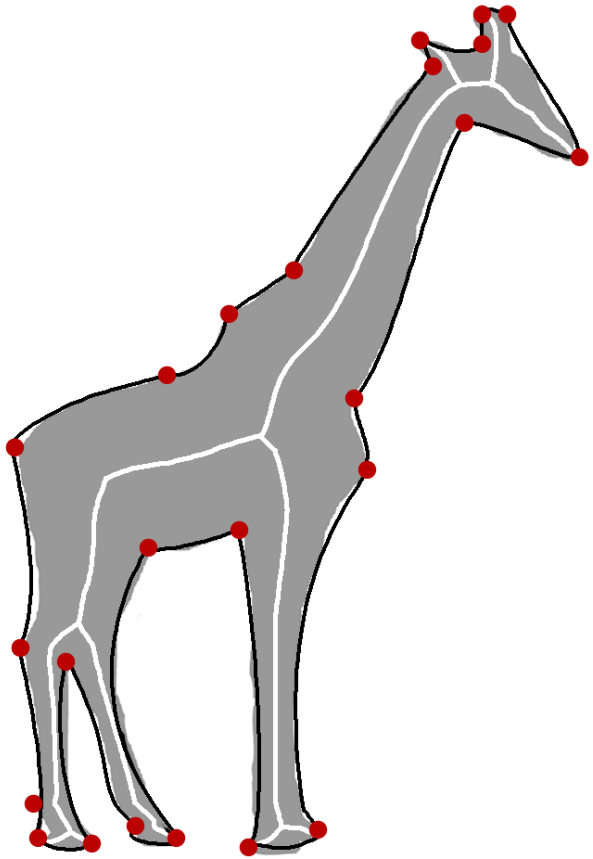
3. Check every branch chain for removal

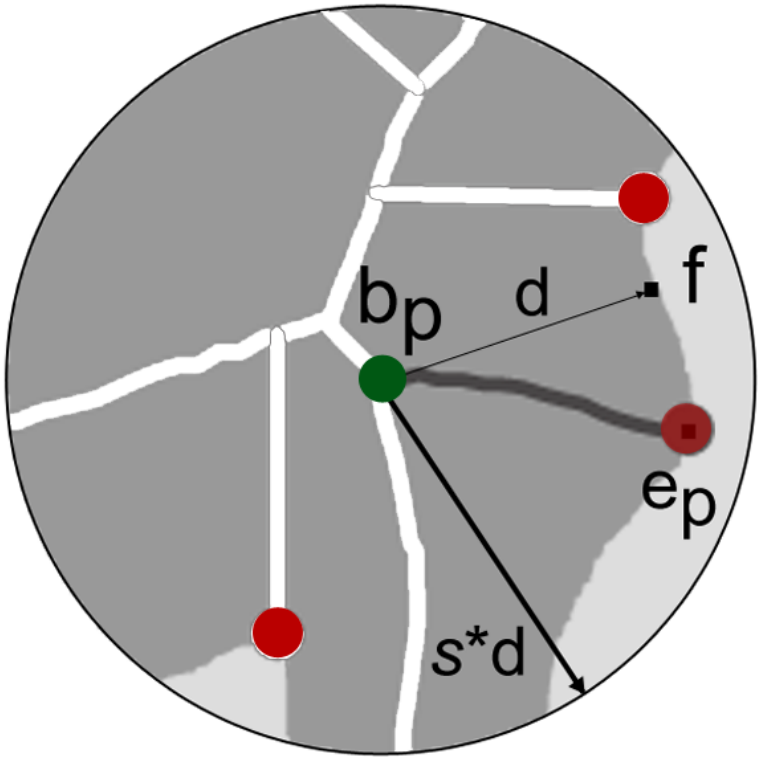




Time complexity: O(n)
n: skeleton pixels

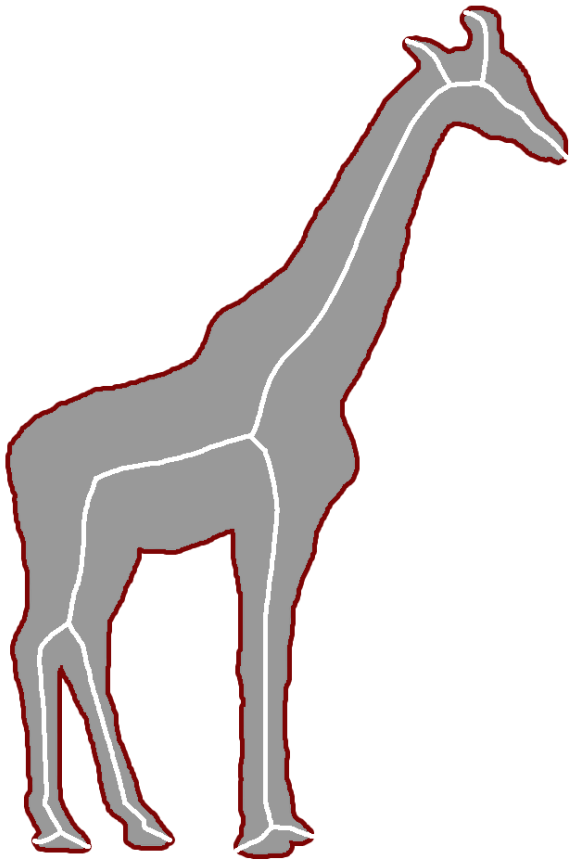
3. Check every branch chain for removal

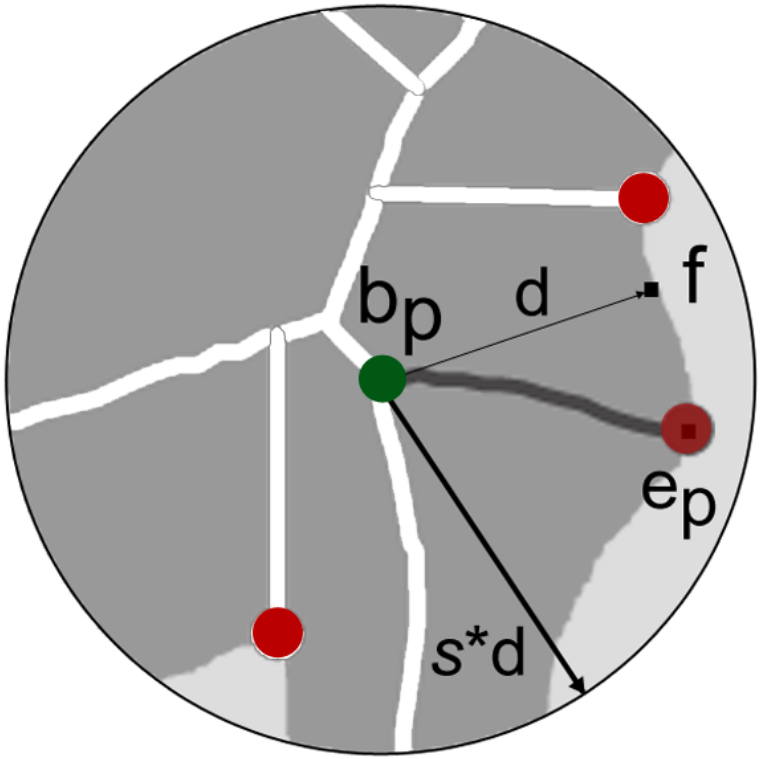




Time complexity: O(n)
n: skeleton pixels

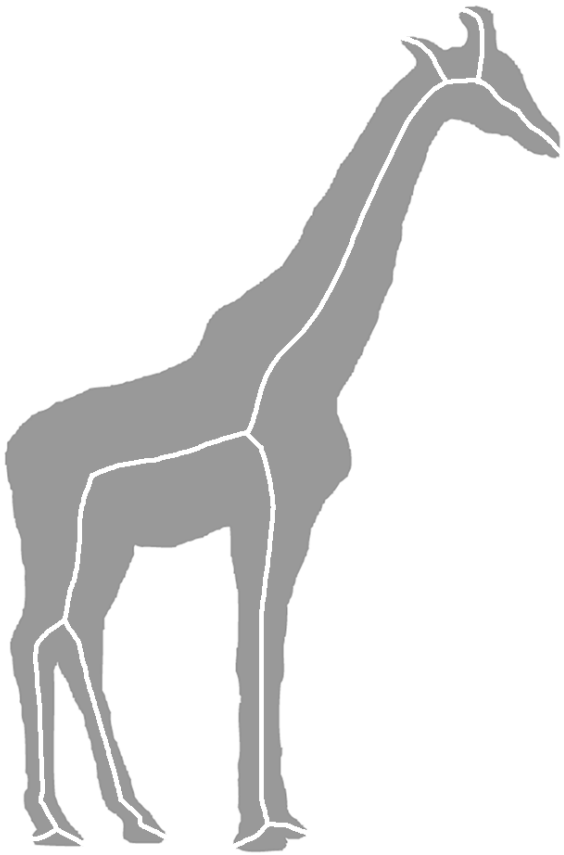
3. Check every branch chain for removal

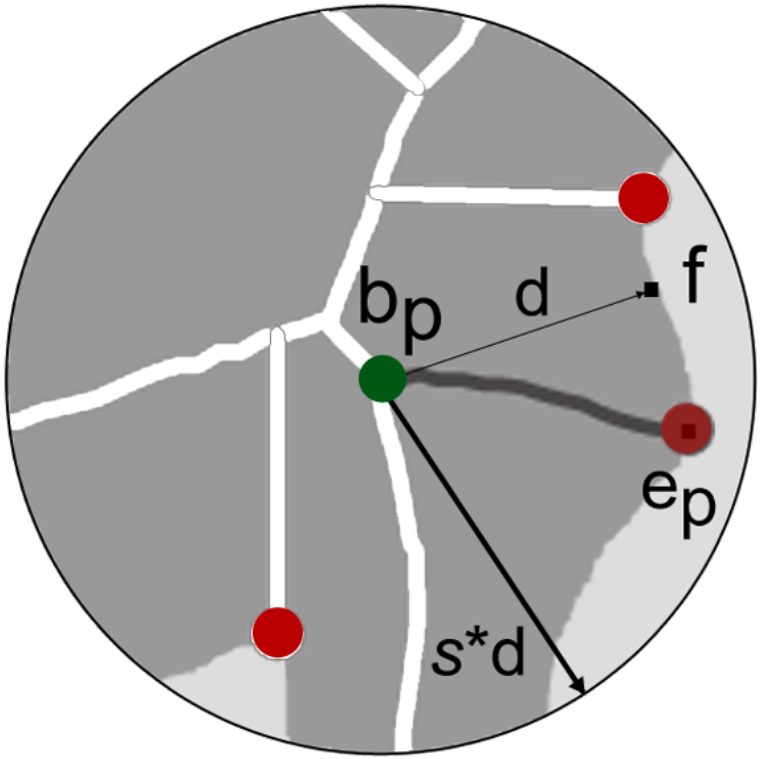




Time complexity: O(n)
n: skeleton pixels

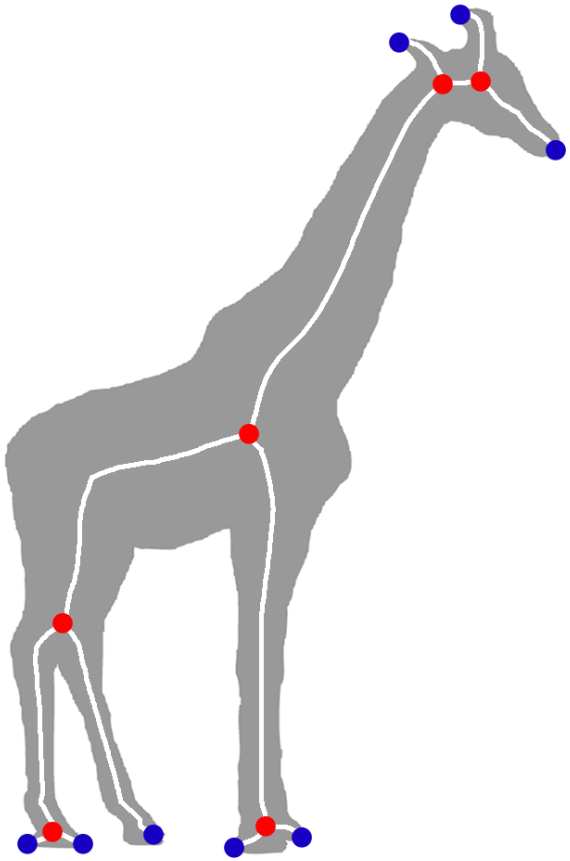
3. Check every branch chain for removal

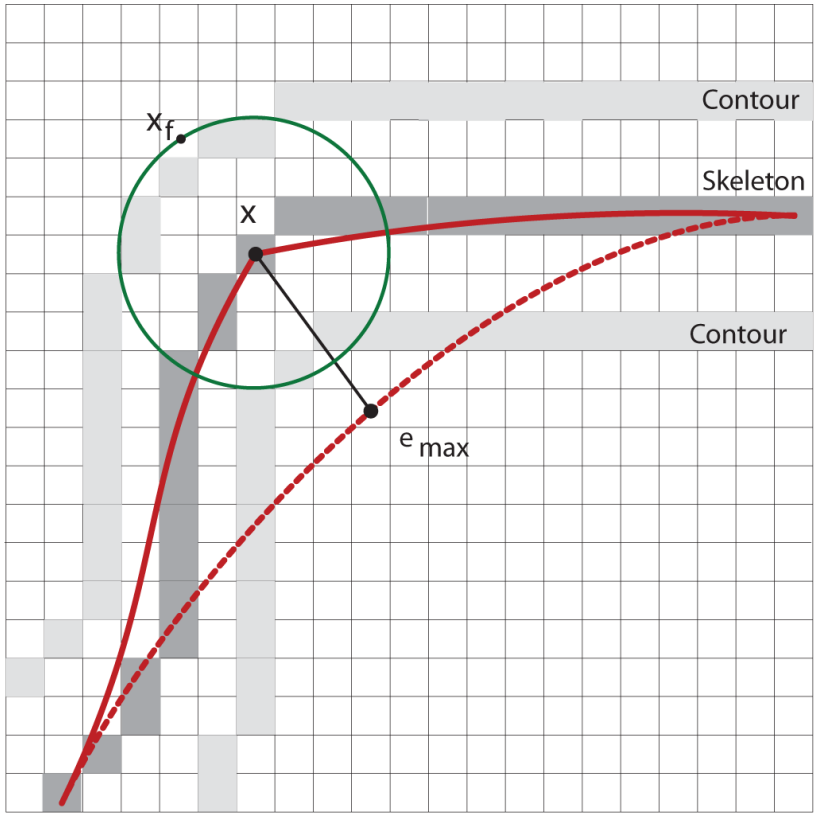




Time complexity: O(n)
n: skeleton pixels

4. Vectorize skeleton

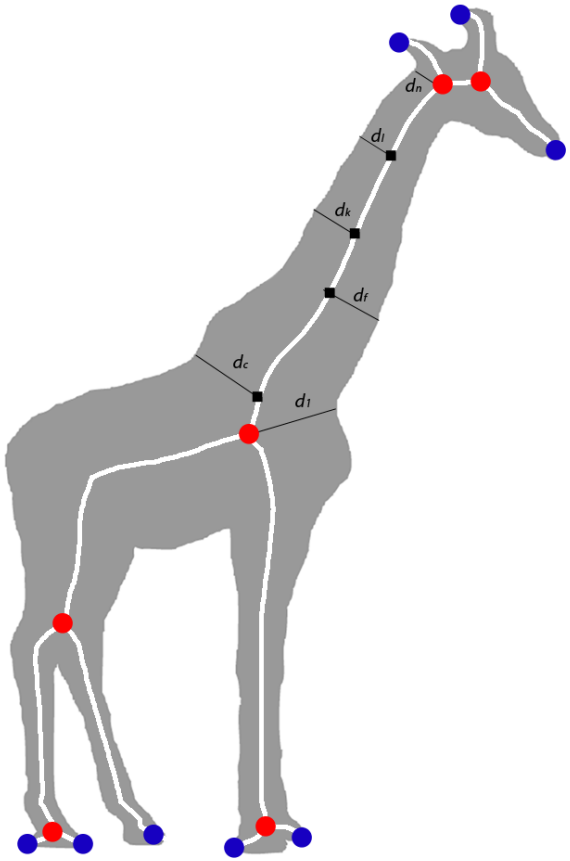


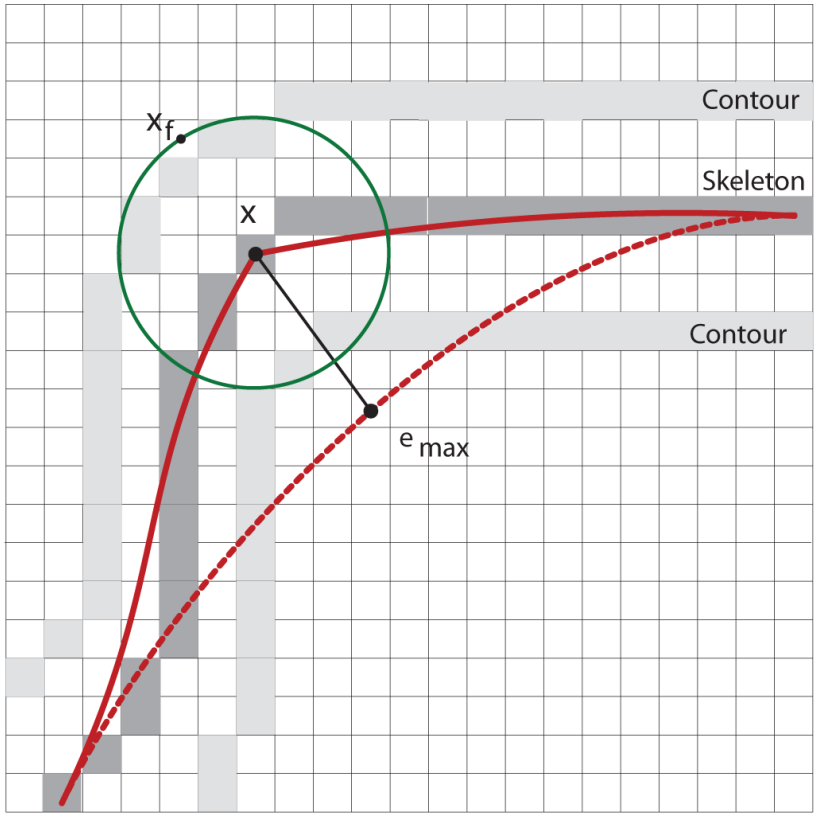


$$\| x_f - x \| < e_{max} < T$$

Time complexity: O(kn)
k: number of knots
n: skeleton pixels

4. Vectorize skeleton

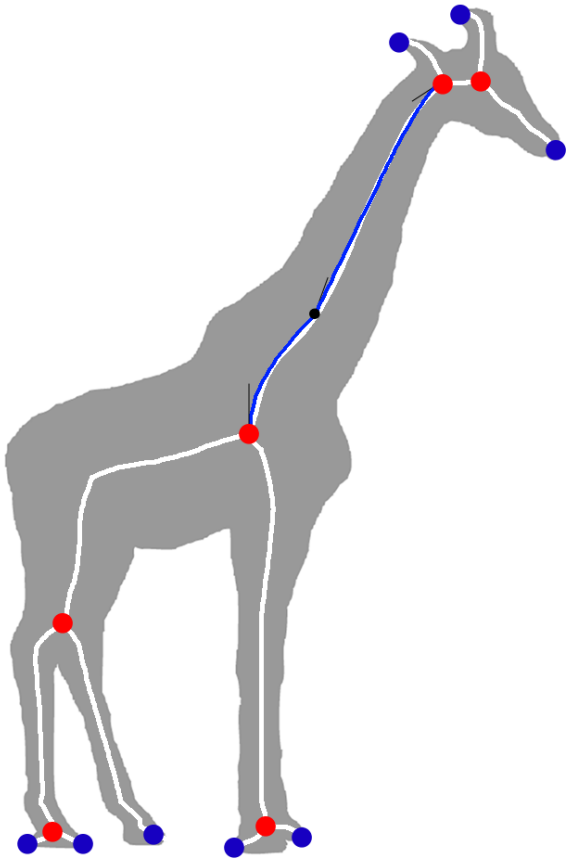


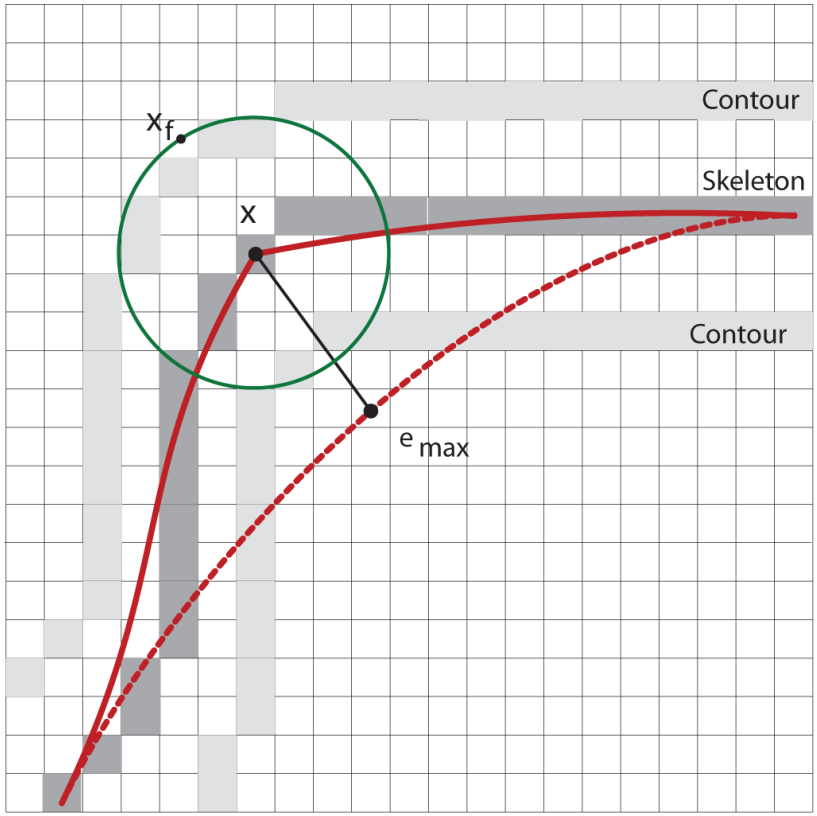


$$\| x_f - x \| < e_{max} < T$$

Time complexity: O(kn)
k: number of knots
n: skeleton pixels

4. Vectorize skeleton

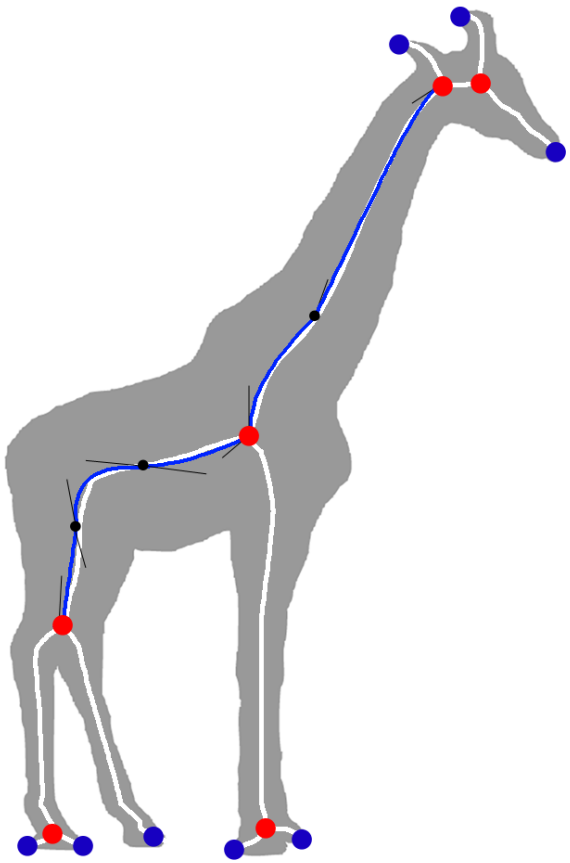


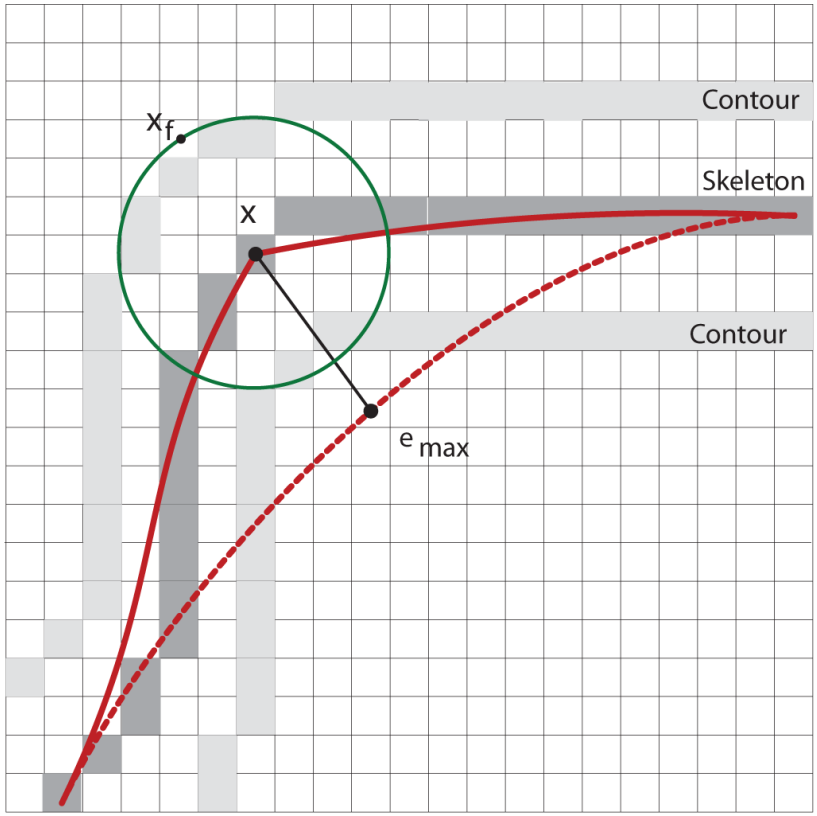


$$\| x_f - x \| < e_{max} < T$$

Time complexity: O(kn)
k: number of knots
n: skeleton pixels

4. Vectorize skeleton

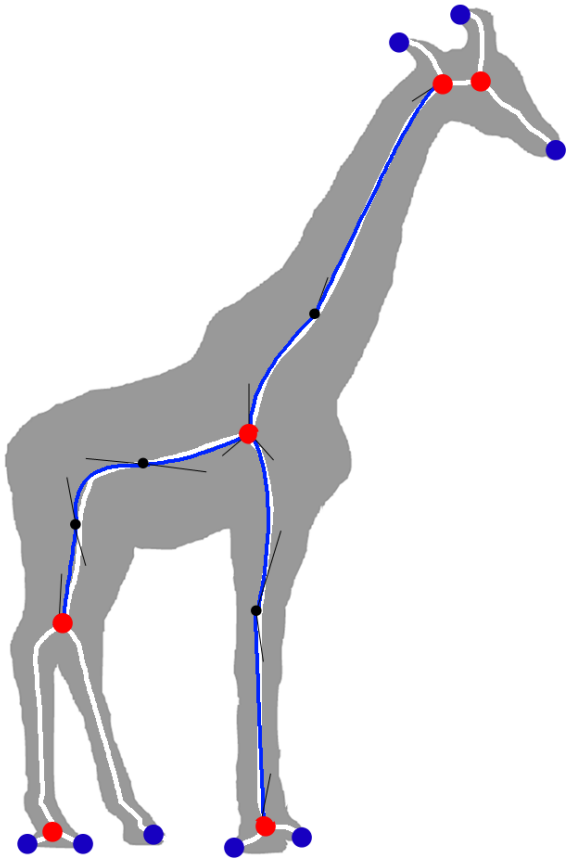


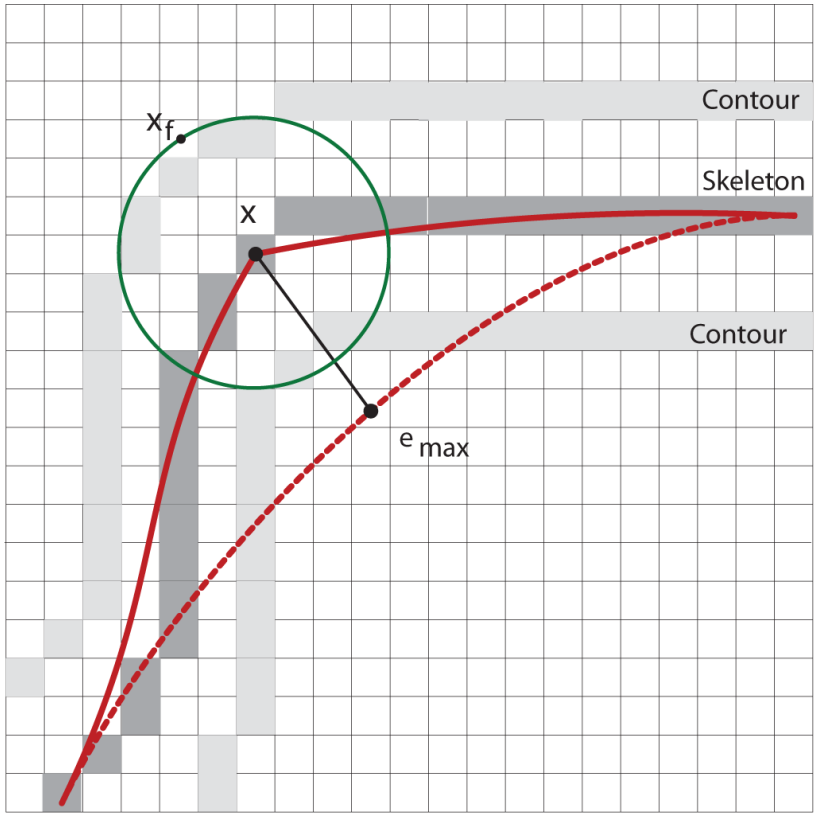


$$\| x_f - x \| < e_{max} < T$$

Time complexity: O(kn)
k: number of knots
n: skeleton pixels

4. Vectorize skeleton

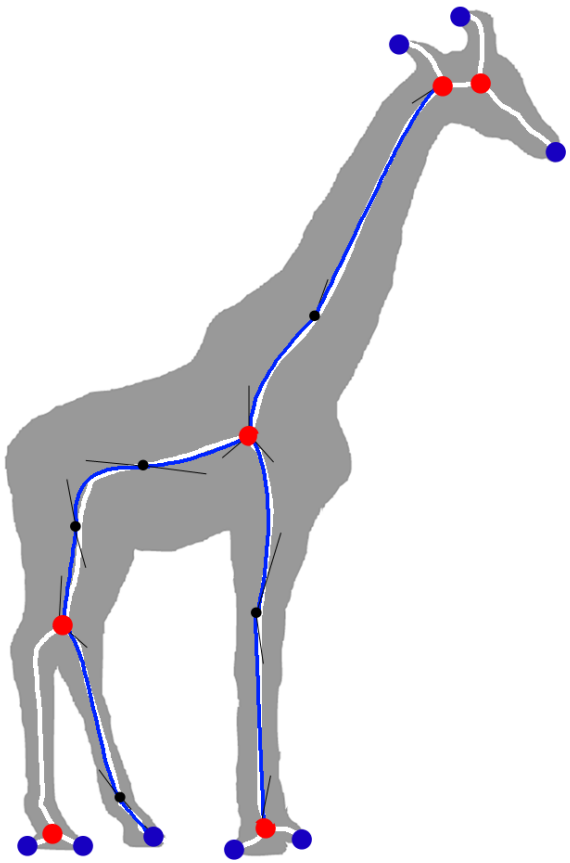


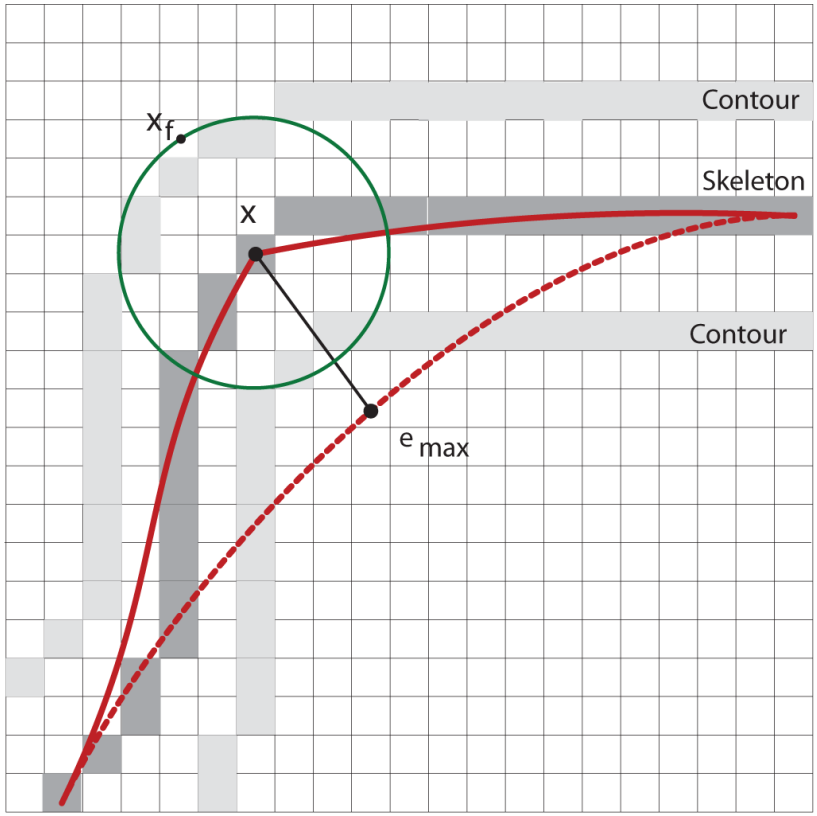


$$\| x_f - x \| < e_{max} < T$$

Time complexity: O(kn)
k: number of knots
n: skeleton pixels

4. Vectorize skeleton

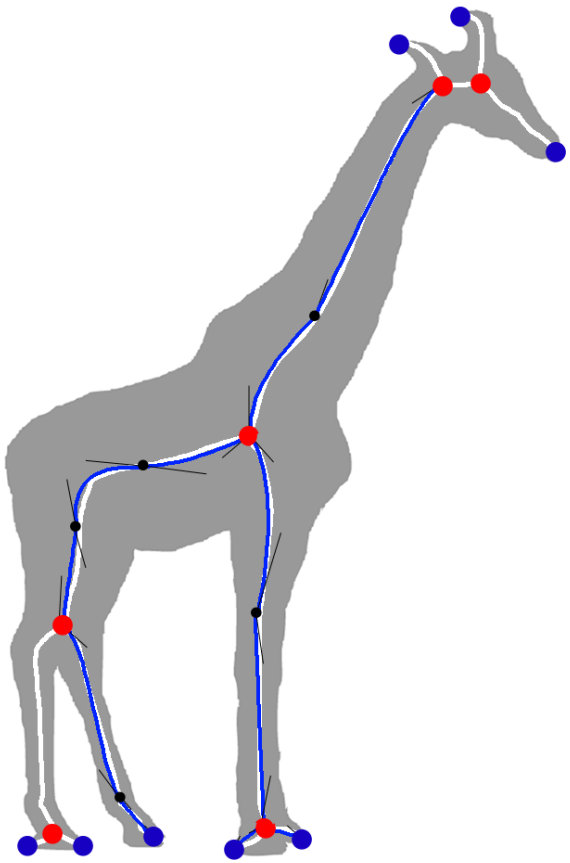


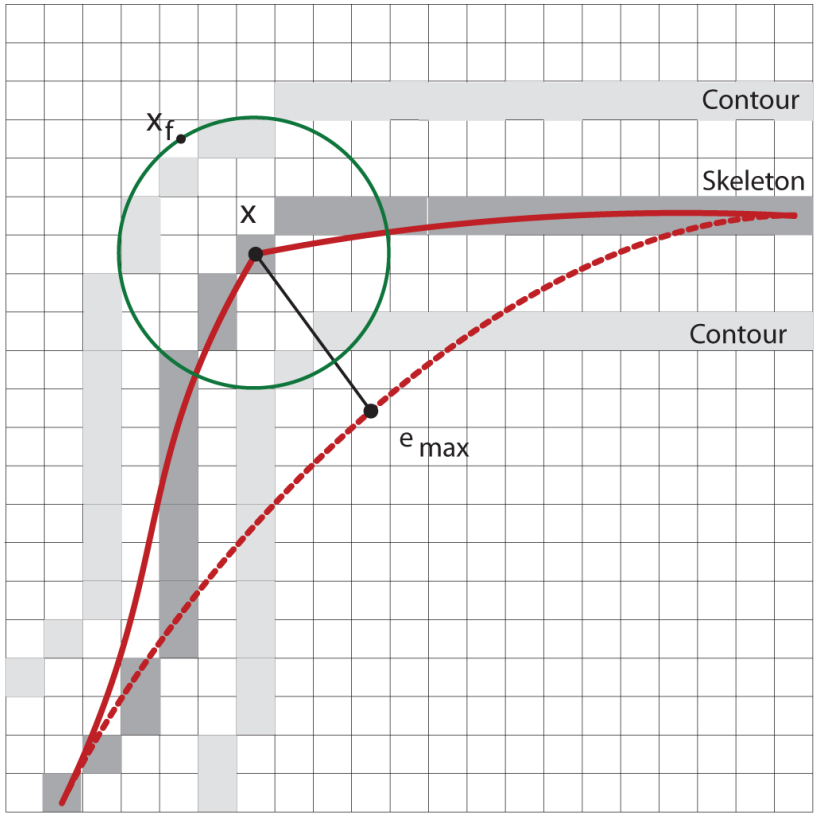


$$\| x_f - x \| < e_{max} < T$$

Time complexity: O(kn)
k: number of knots
n: skeleton pixels

4. Vectorize skeleton

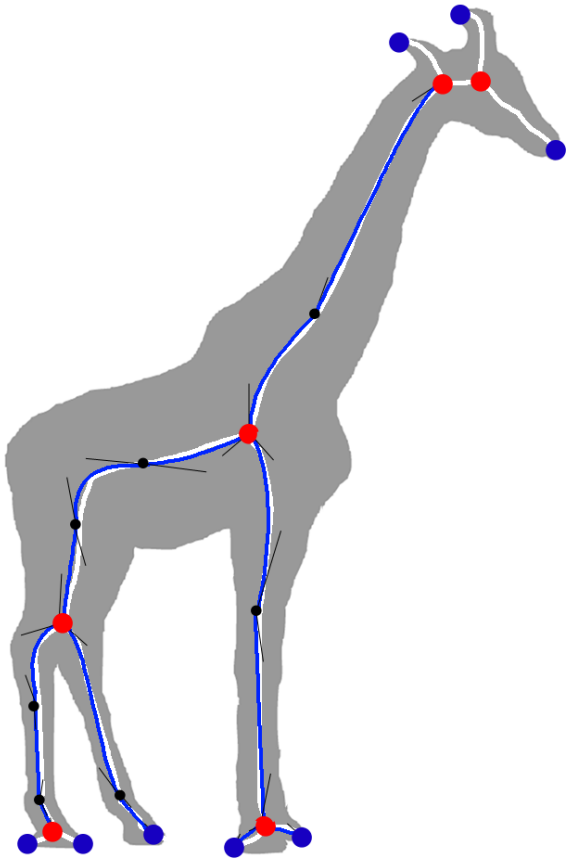


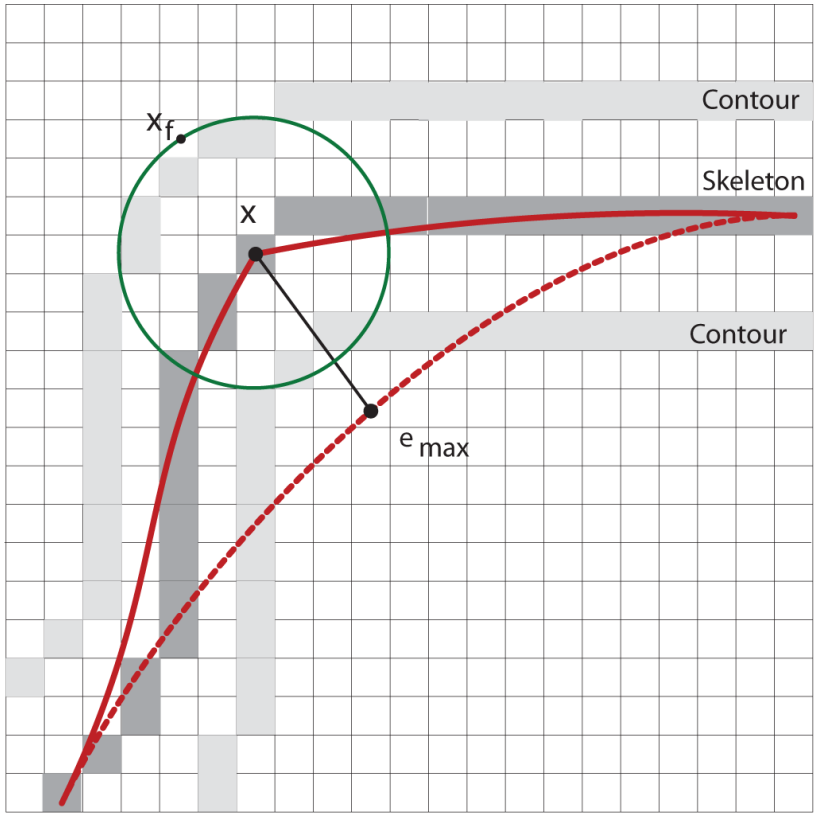


$$\| x_f - x \| < e_{max} < T$$

Time complexity: O(kn)
k: number of knots
n: skeleton pixels

4. Vectorize skeleton

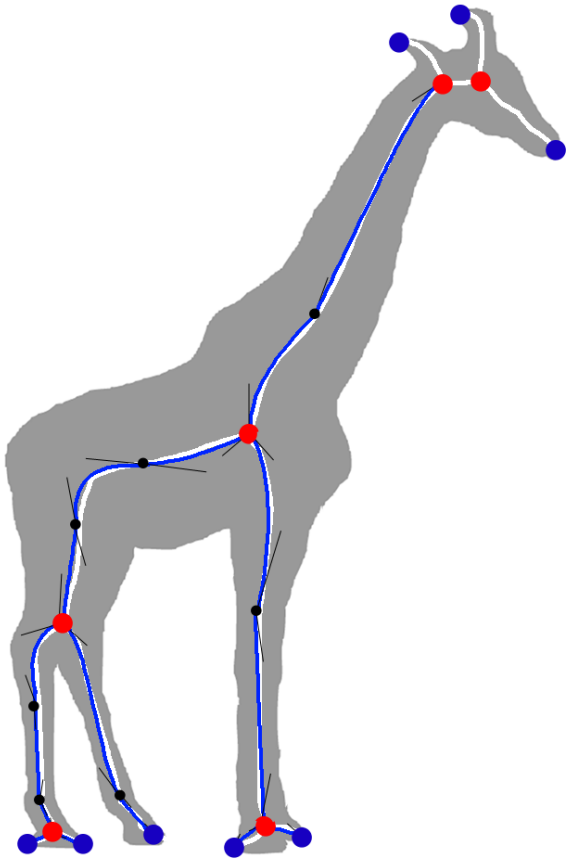


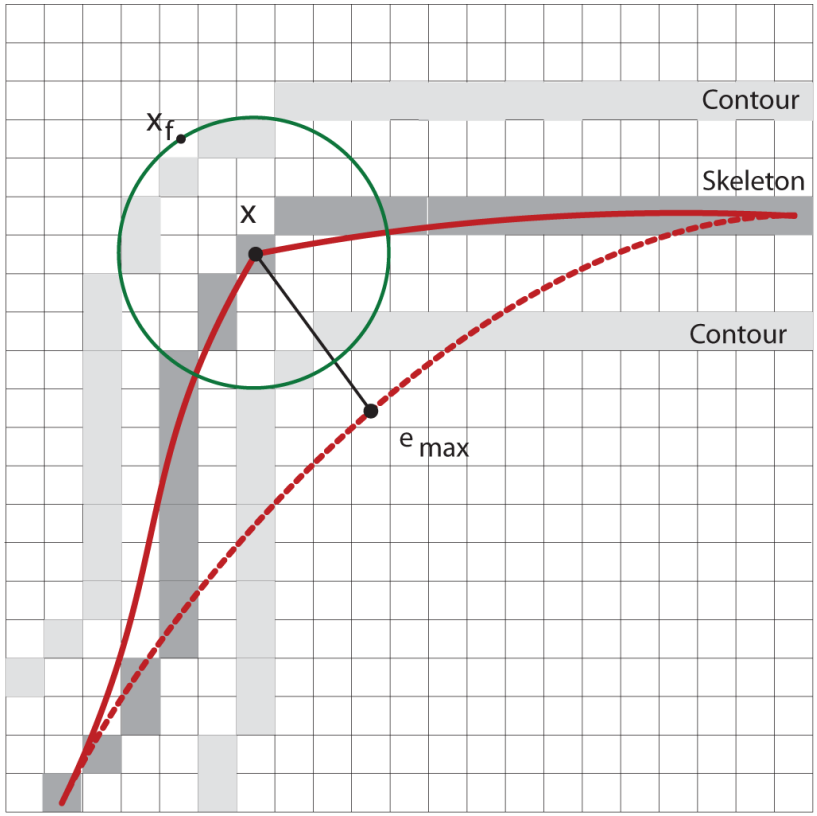


$$\| x_f - x \| < e_{max} < T$$

Time complexity: O(kn)
k: number of knots
n: skeleton pixels

4. Vectorize skeleton

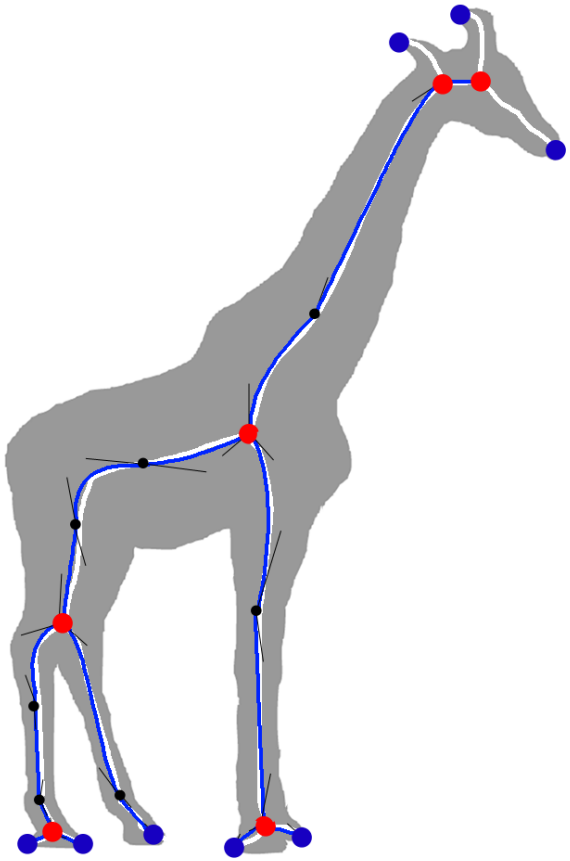


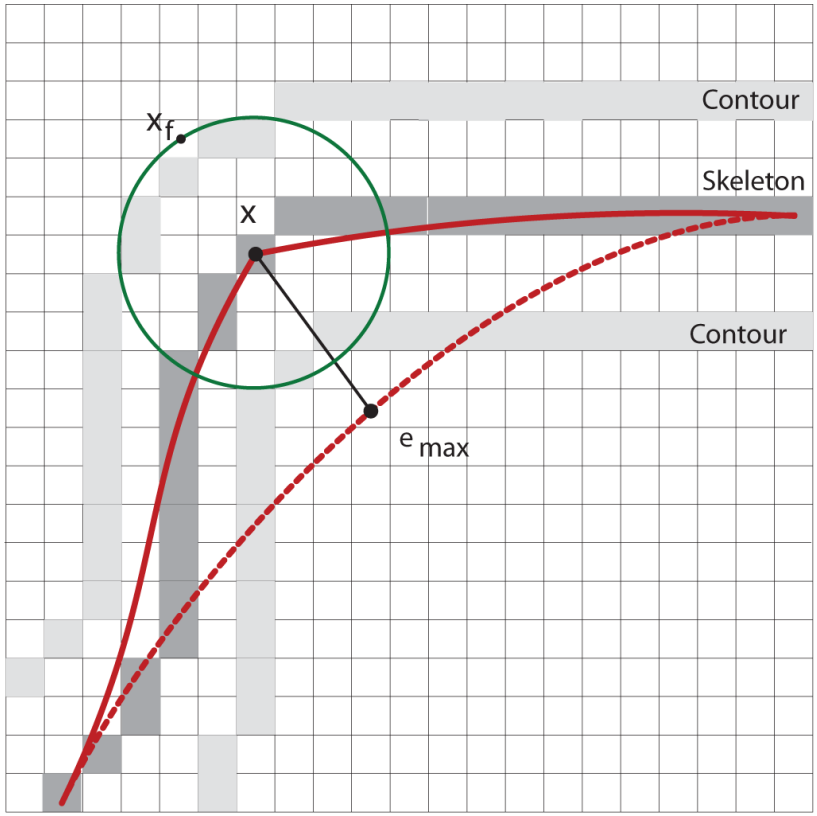


$$\| x_f - x \| < e_{max} < T$$

Time complexity: O(kn)
k: number of knots
n: skeleton pixels

4. Vectorize skeleton

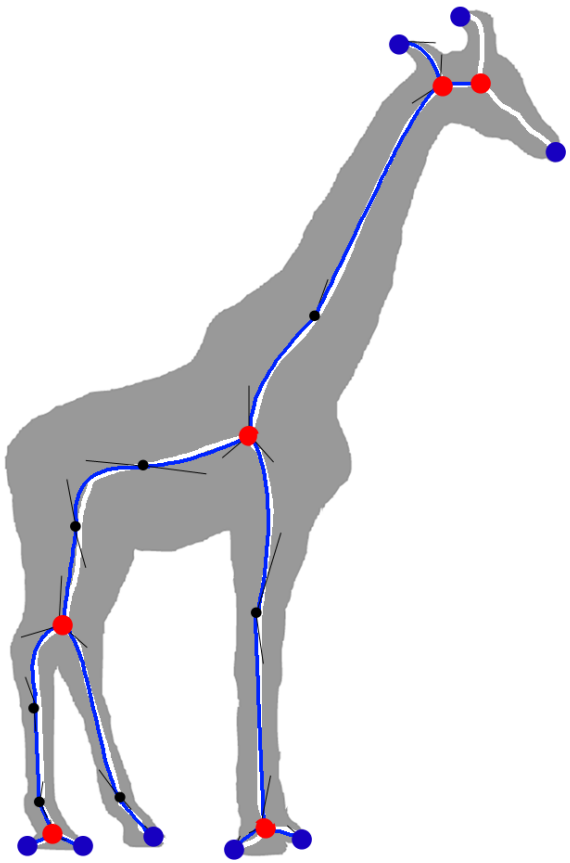


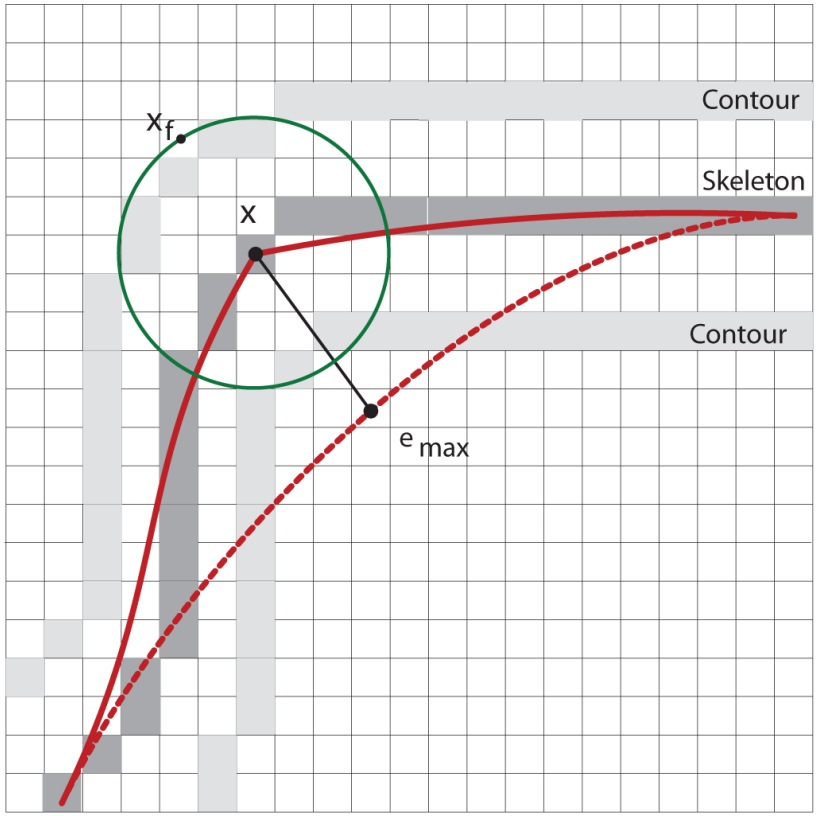


$$\| x_f - x \| < e_{max} < T$$

Time complexity: O(kn)
k: number of knots
n: skeleton pixels

4. Vectorize skeleton

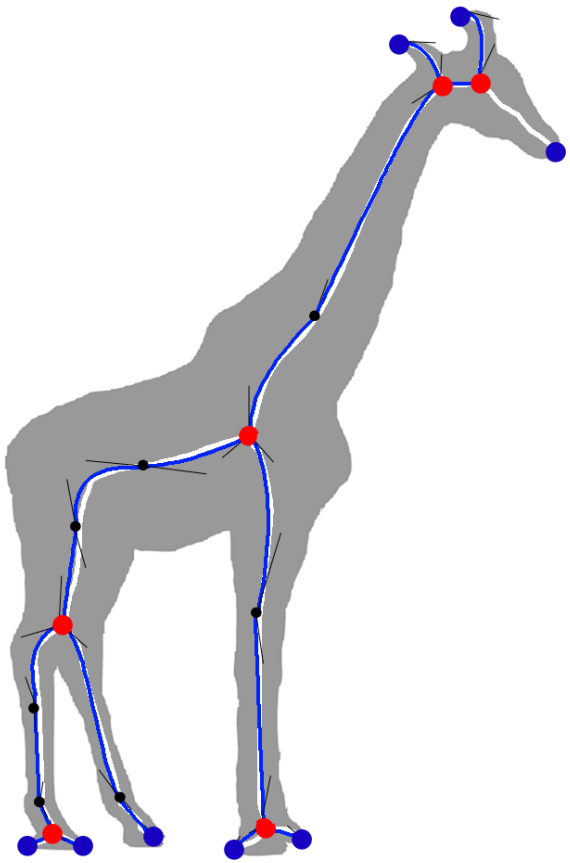


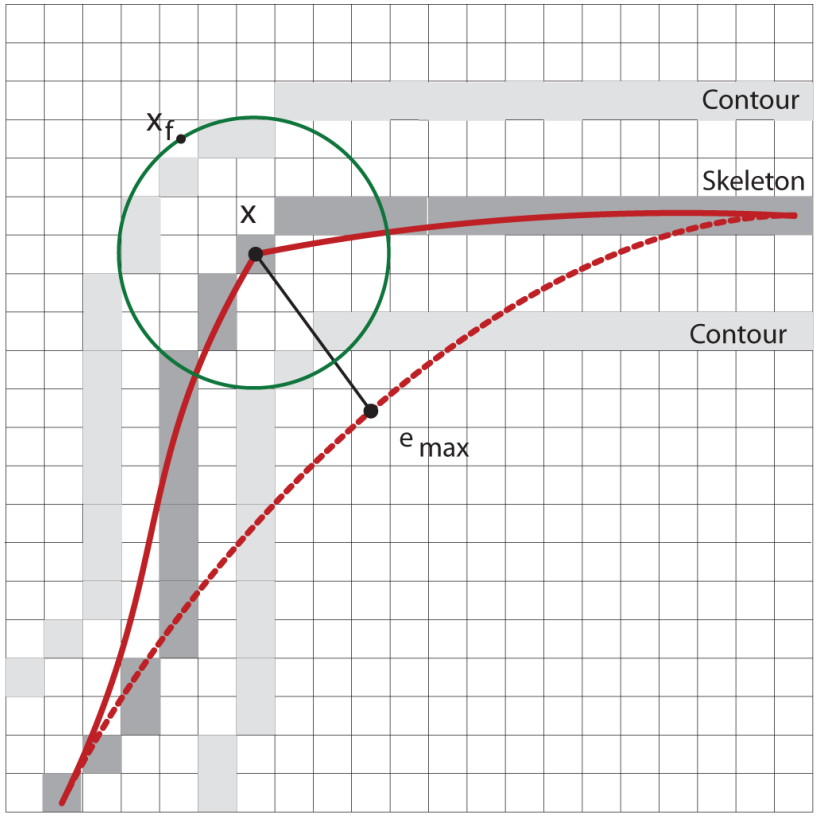


$$\| x_f - x \| < e_{max} < T$$

Time complexity: O(kn)
k: number of knots
n: skeleton pixels

4. Vectorize skeleton

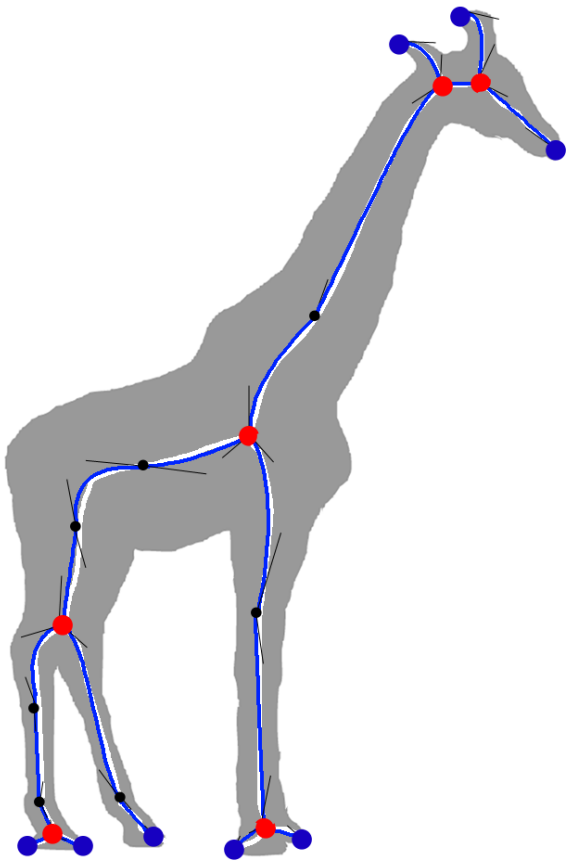


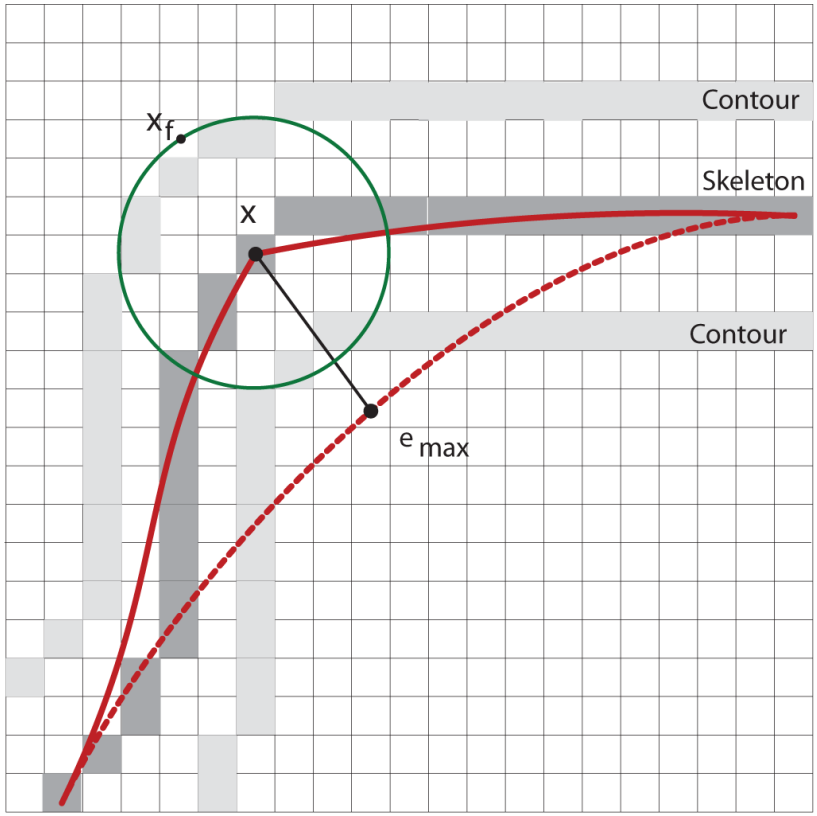


$$\| x_f - x \| < e_{max} < T$$

Time complexity: O(kn)
k: number of knots
n: skeleton pixels

4. Vectorize skeleton

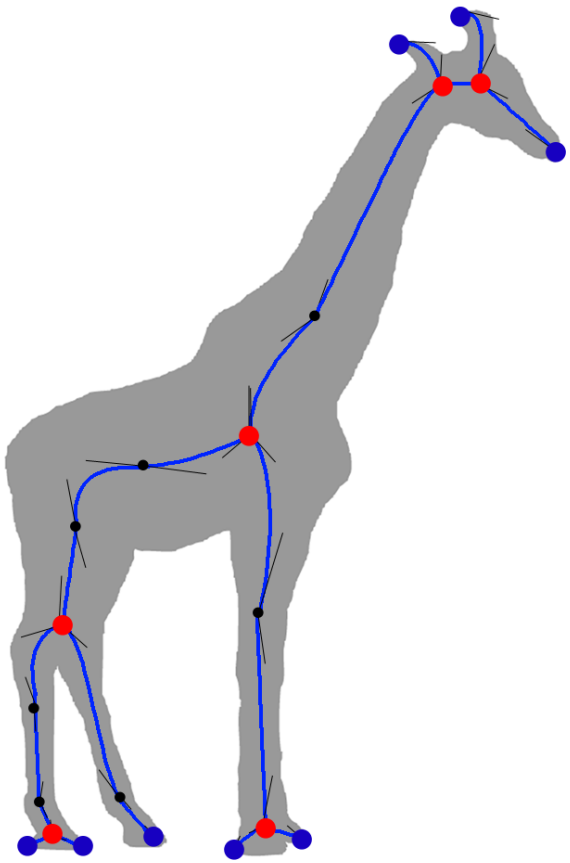


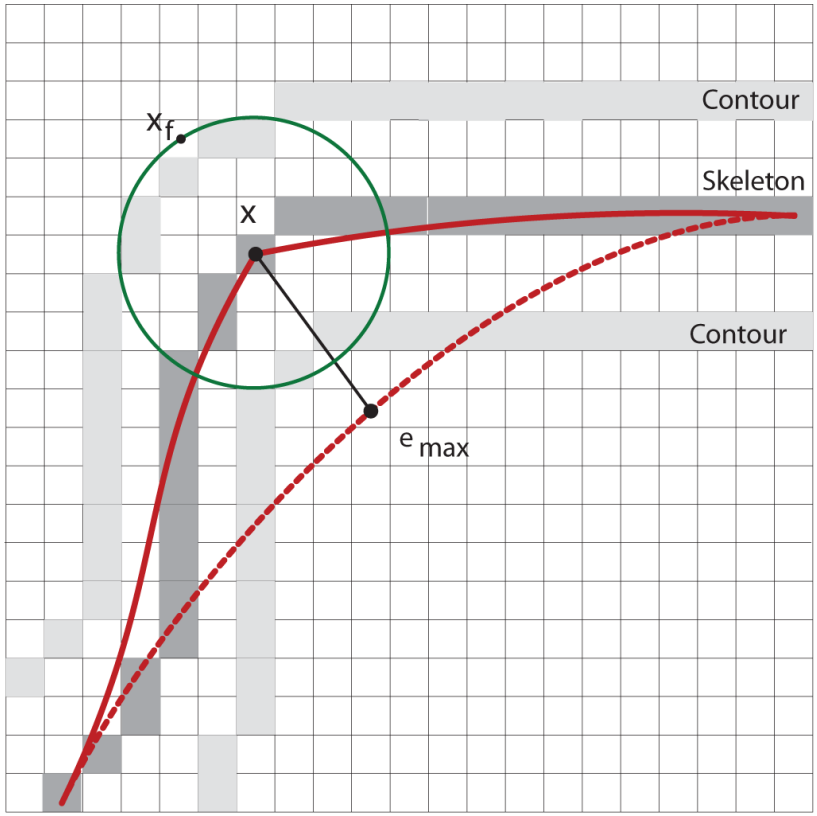


$$\| x_f - x \| < e_{max} < T$$

Time complexity: O(kn)
k: number of knots
n: skeleton pixels

4. Vectorize skeleton





$$\| x_f - x \| < e_{max} < T$$

Time complexity: O(kn)
k: number of knots
n: skeleton pixels

Experiment

Dataset

Experiments were done on the mpeg-7 dataset (1401 images), testing for accuracy, running time, connectivity, shape reconstruction, rotation invariance, noise sensitivity, and parameter selection.

Resolution of Images

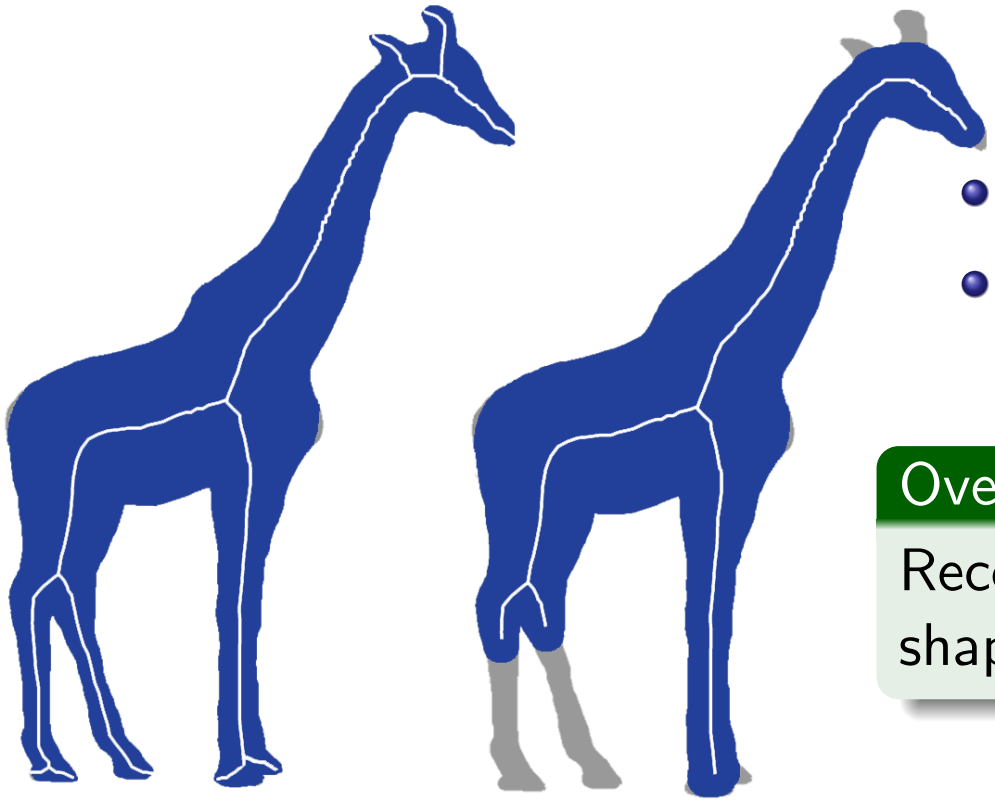
Image resolutions ranged from [256x256] to [950x800]

Experimental Results

Running Time

Algorithm	95% Confidence Interval
IMA+D,A	0.08 ± 0.01 sec
Linear	0.09 ± 0.004 sec
Cubic	0.10 ± 0.009 sec
Scale Axis Transform	0.17 ± 0.013 sec
Thinning	0.35 ± 0.018 sec
Voronoi	0.68 ± 0.021 sec
DCE	5.13 ± 0.518 sec

Experimental Results — Shape reconstruction

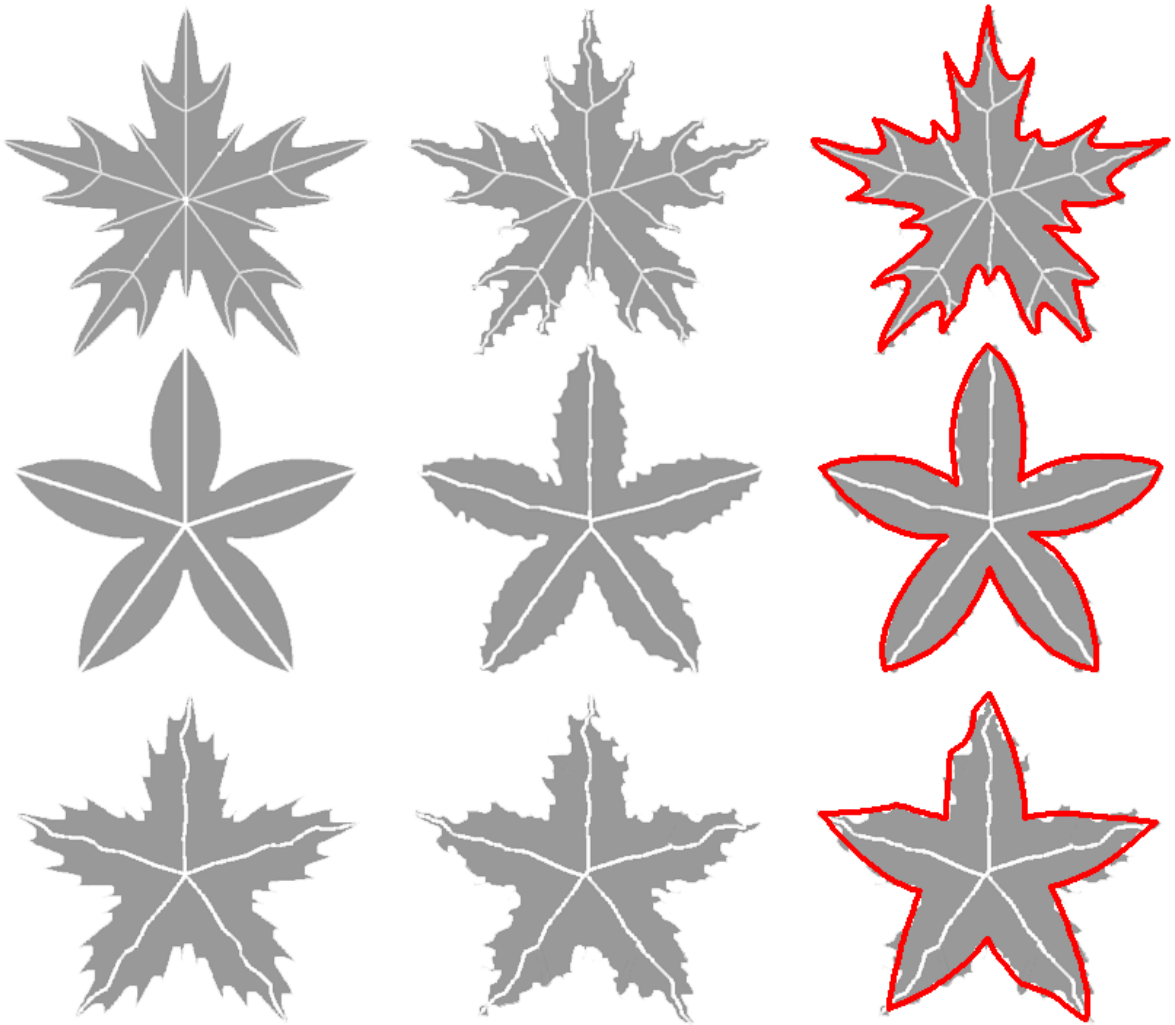


- Left: Our solution.
- Right: IMA + Angular Pruning.

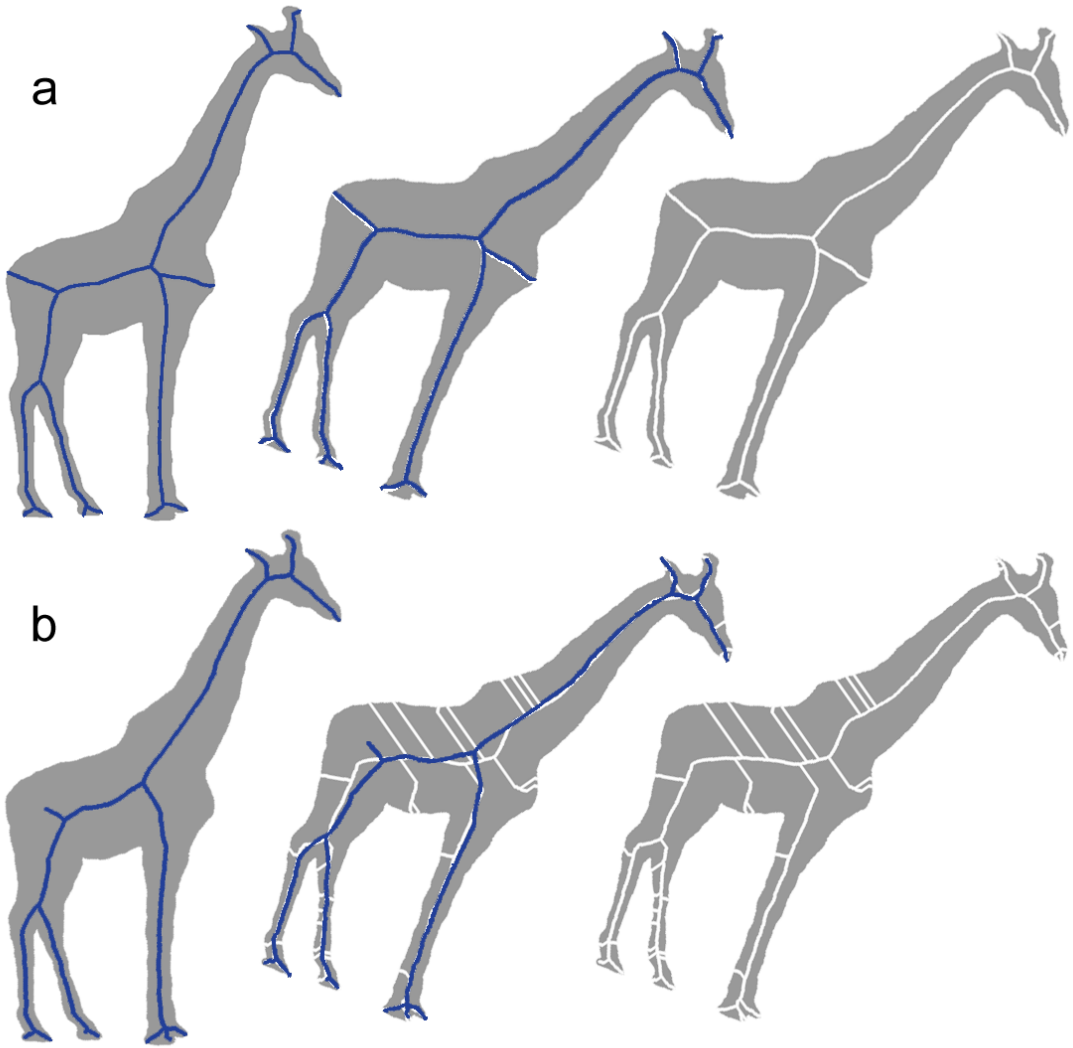
Overall

Reconstruction of the original shape 98 % (average) in area.

Experimental Results — Noise invariant

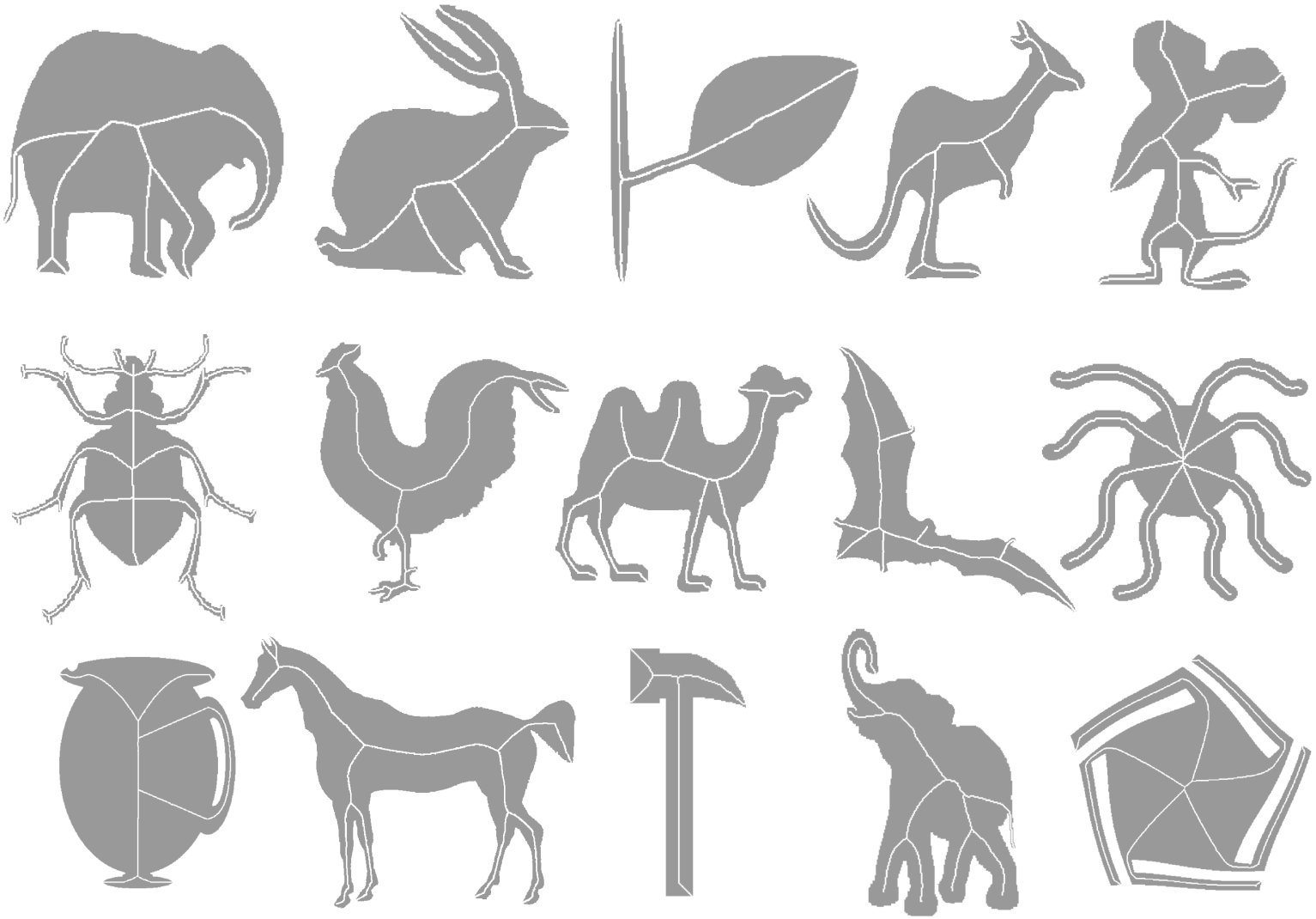


Experimental Results — Rotation Invariance

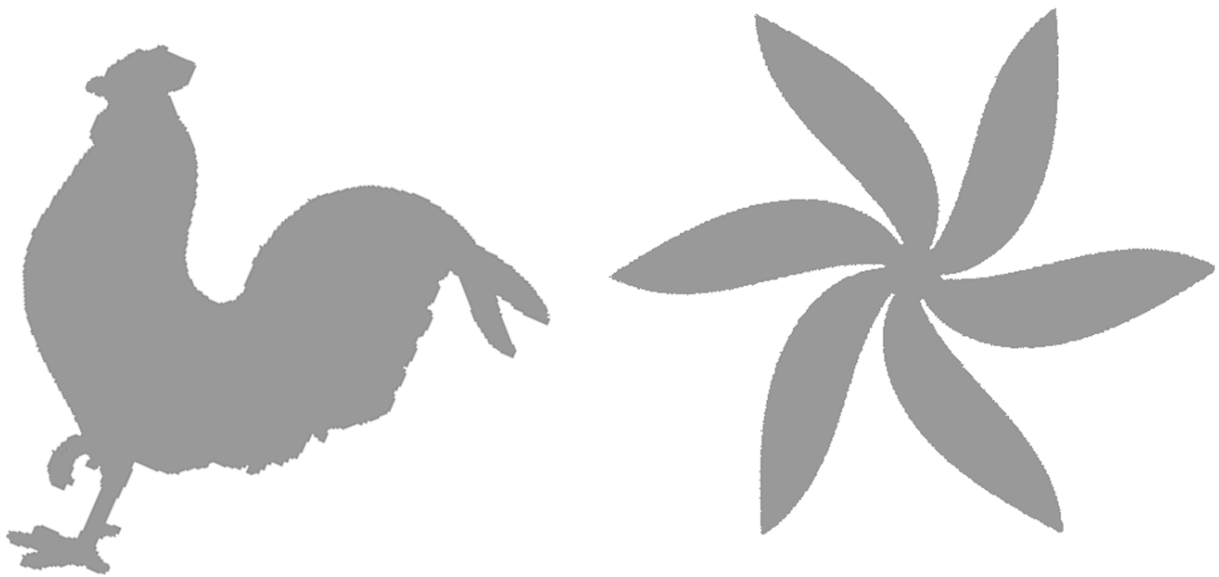


Experimental Results — Complex shapes

Experimental Results — Parameter Selection

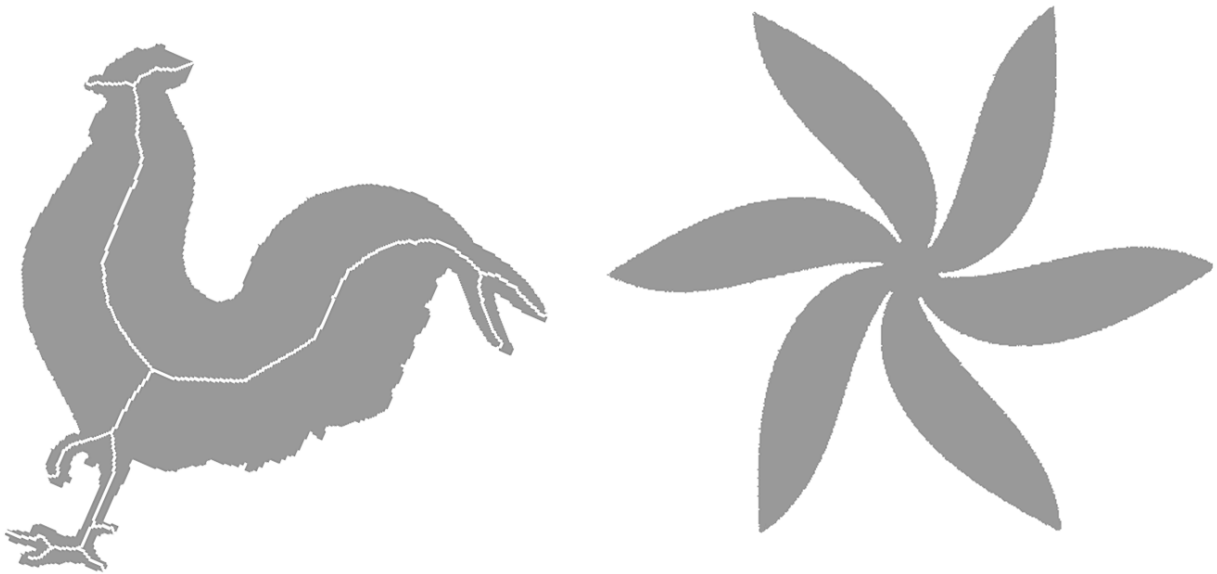


More Examples



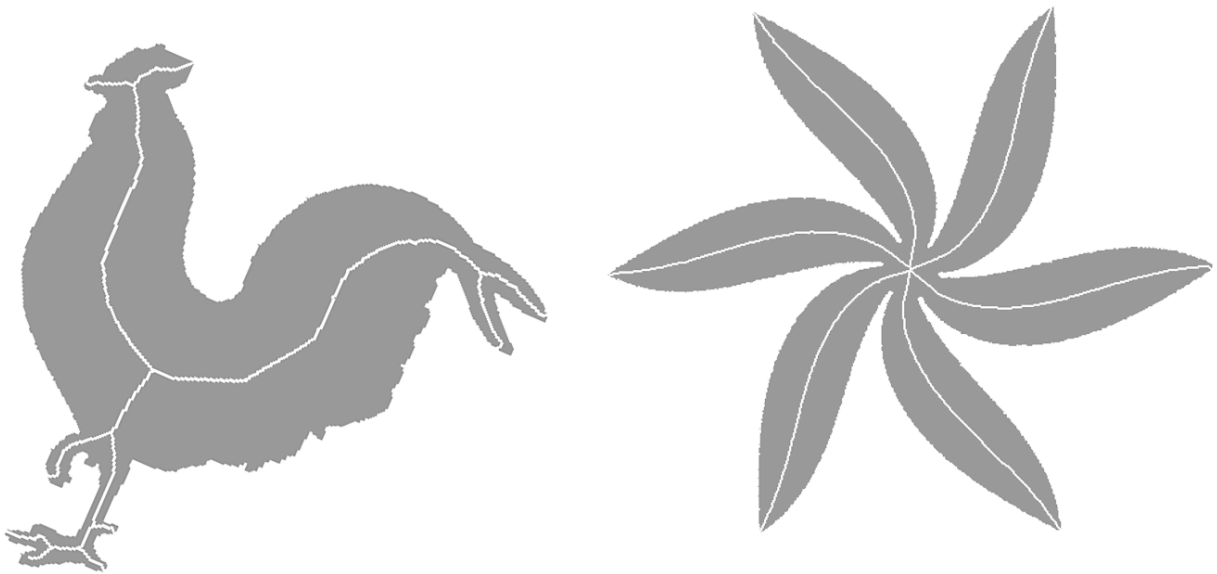
The selected input parameters are the same

More Examples



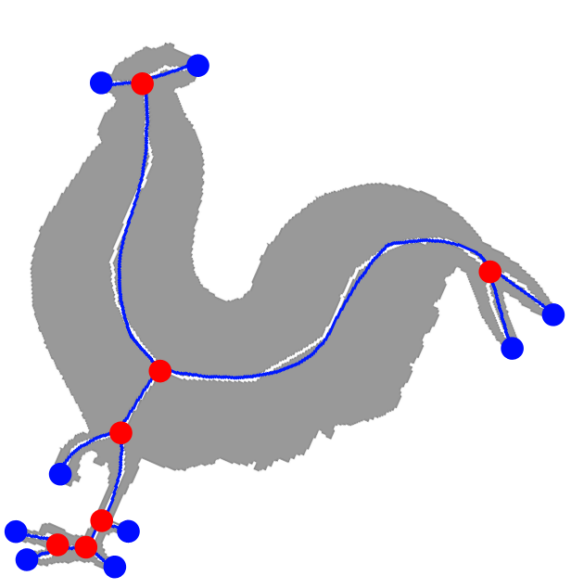
The selected input parameters are the same

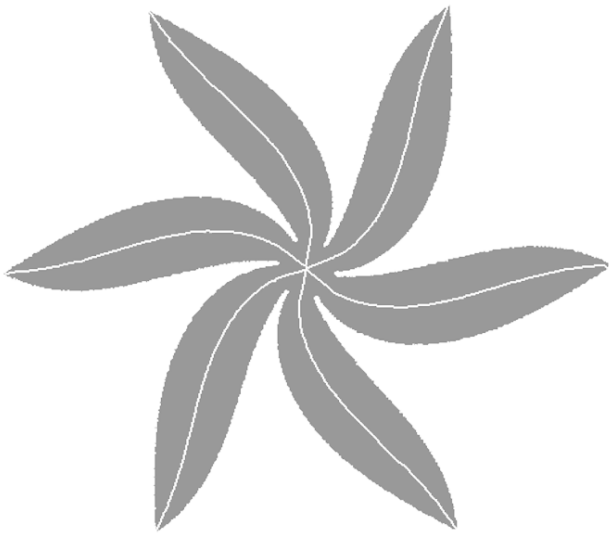
More Examples



The selected input parameters are the same

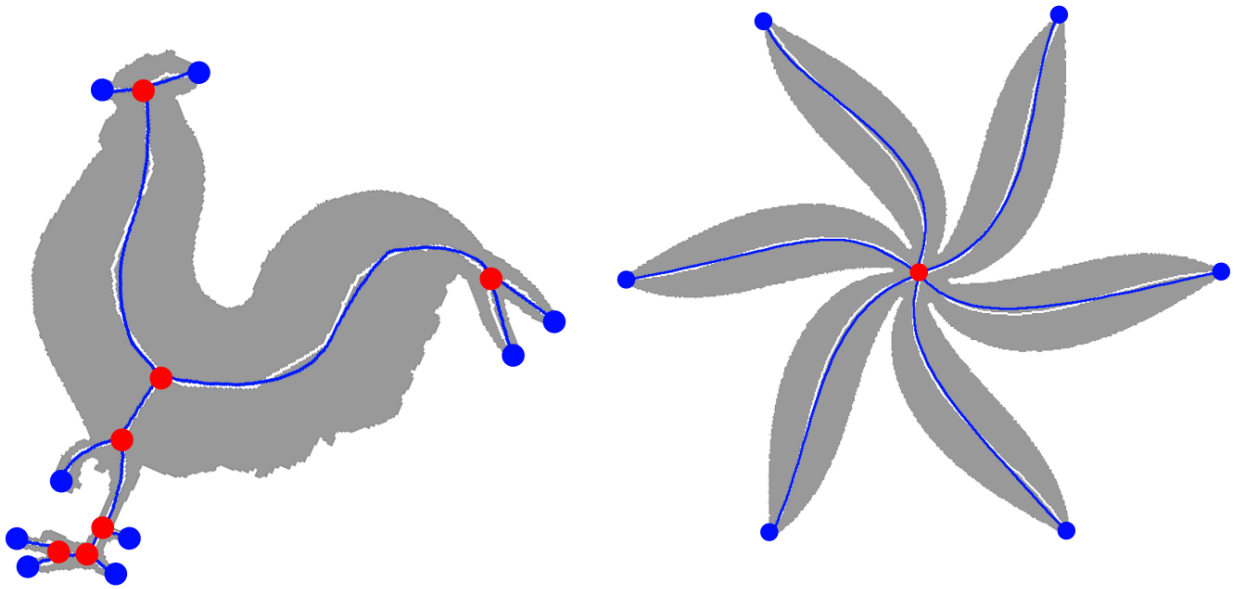
More Examples





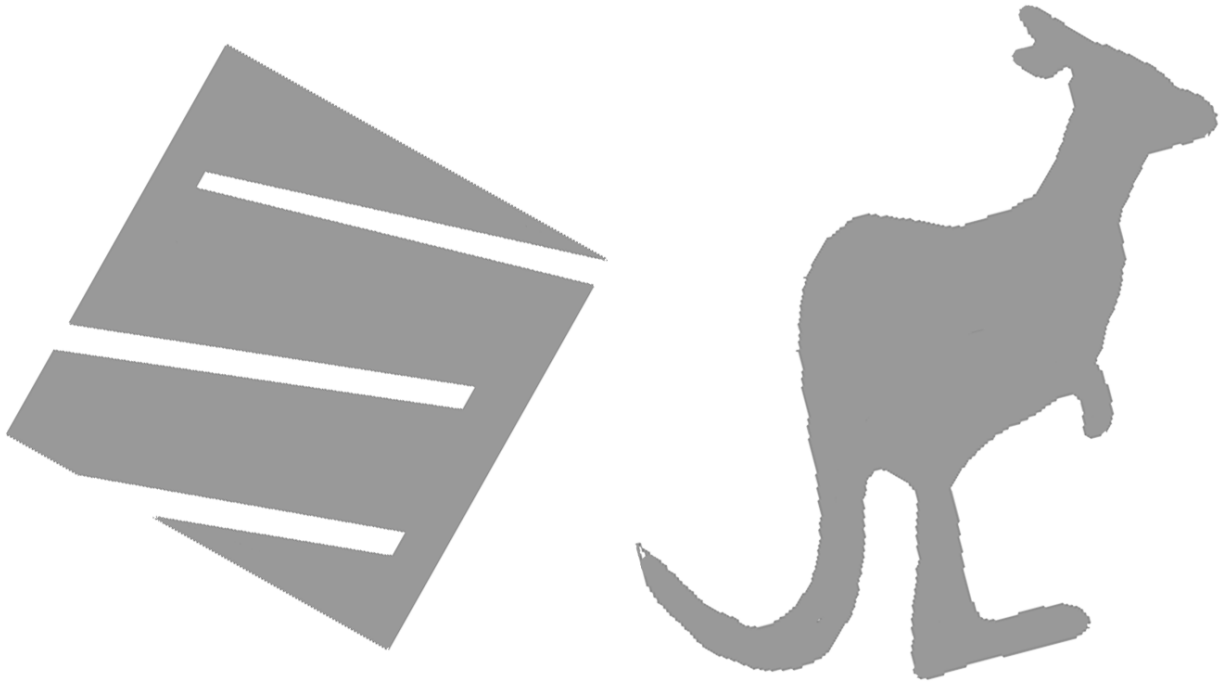
The selected input parameters are the same

More Examples



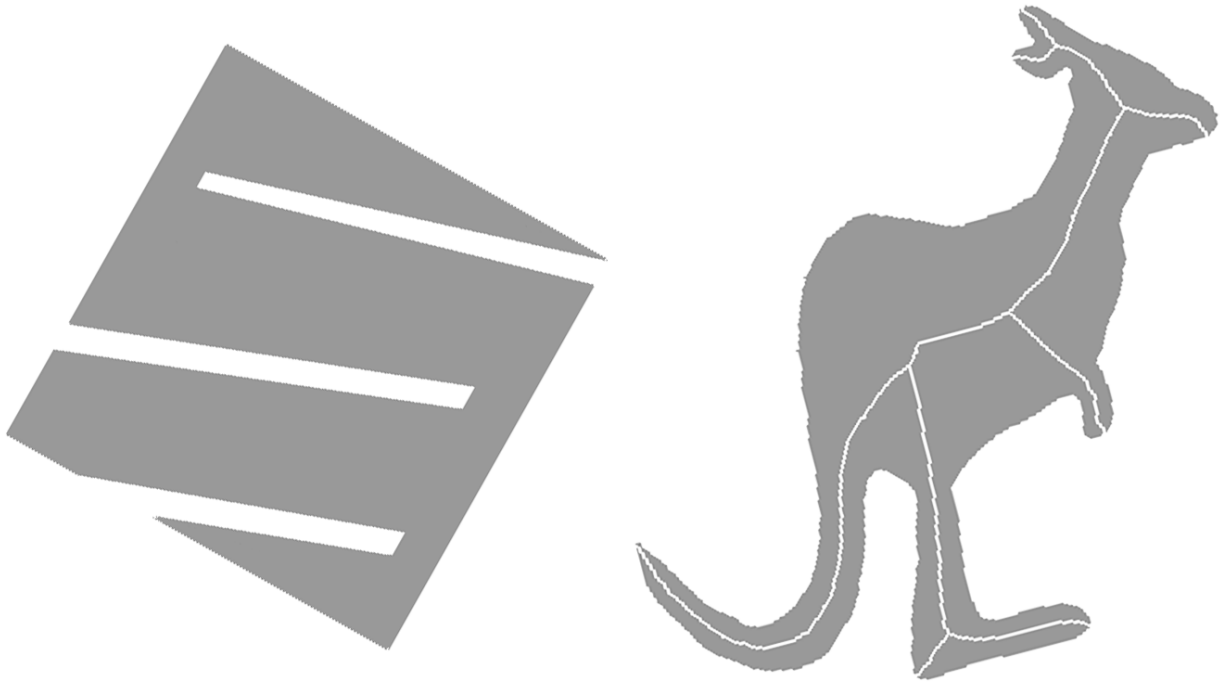
The selected input parameters are the same

More Examples



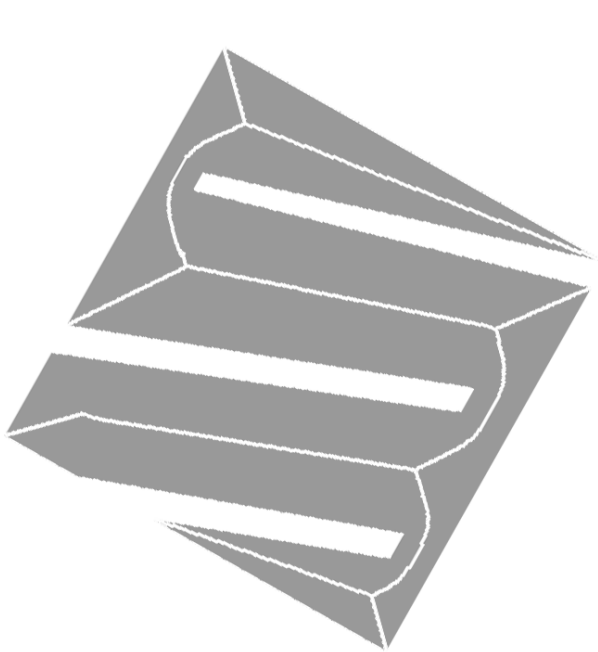
The selected input parameters are the same

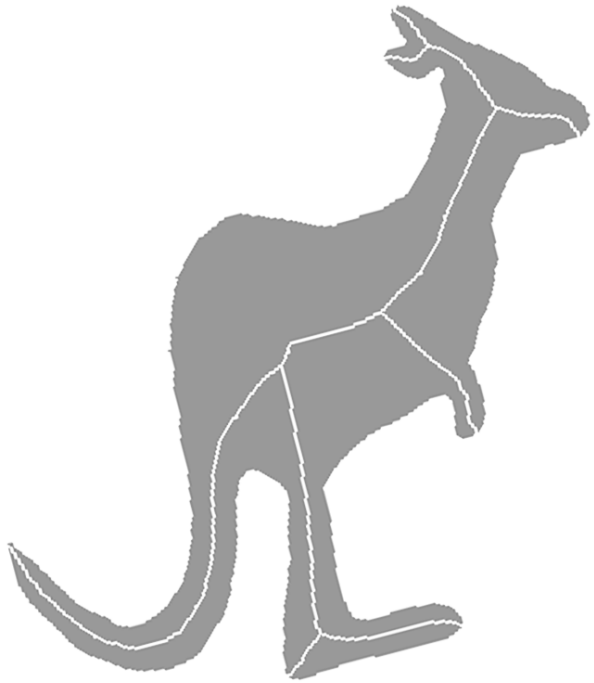
More Examples



The selected input parameters are the same

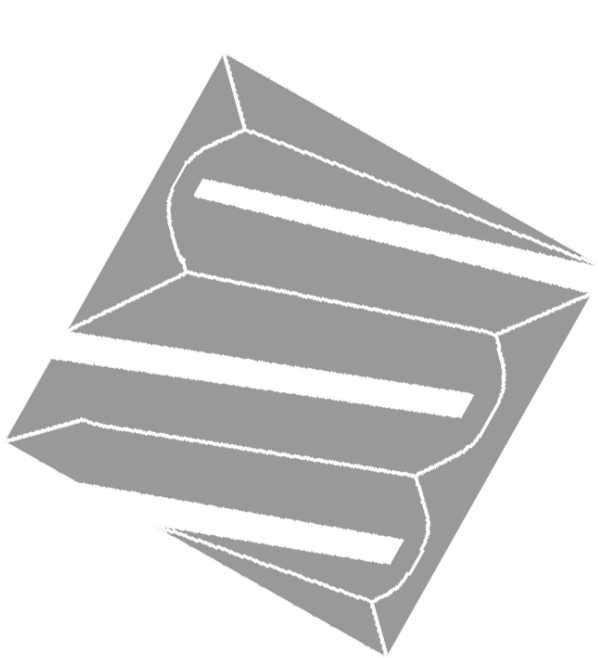
More Examples

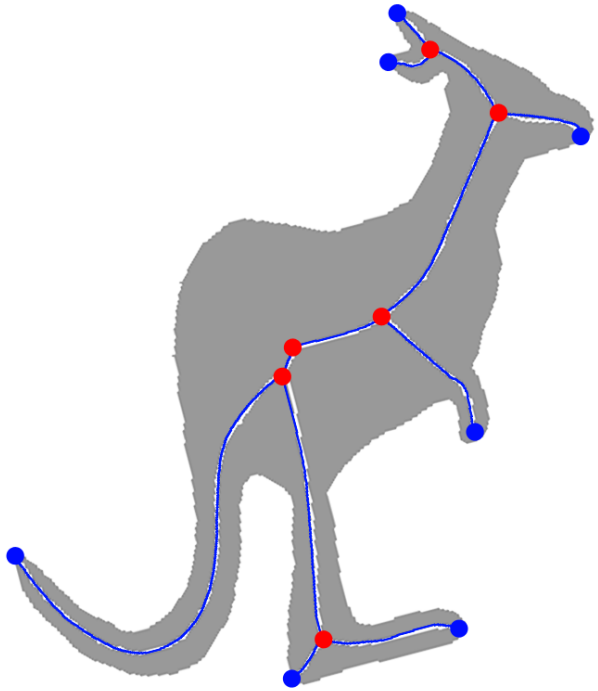




The selected input parameters are the same

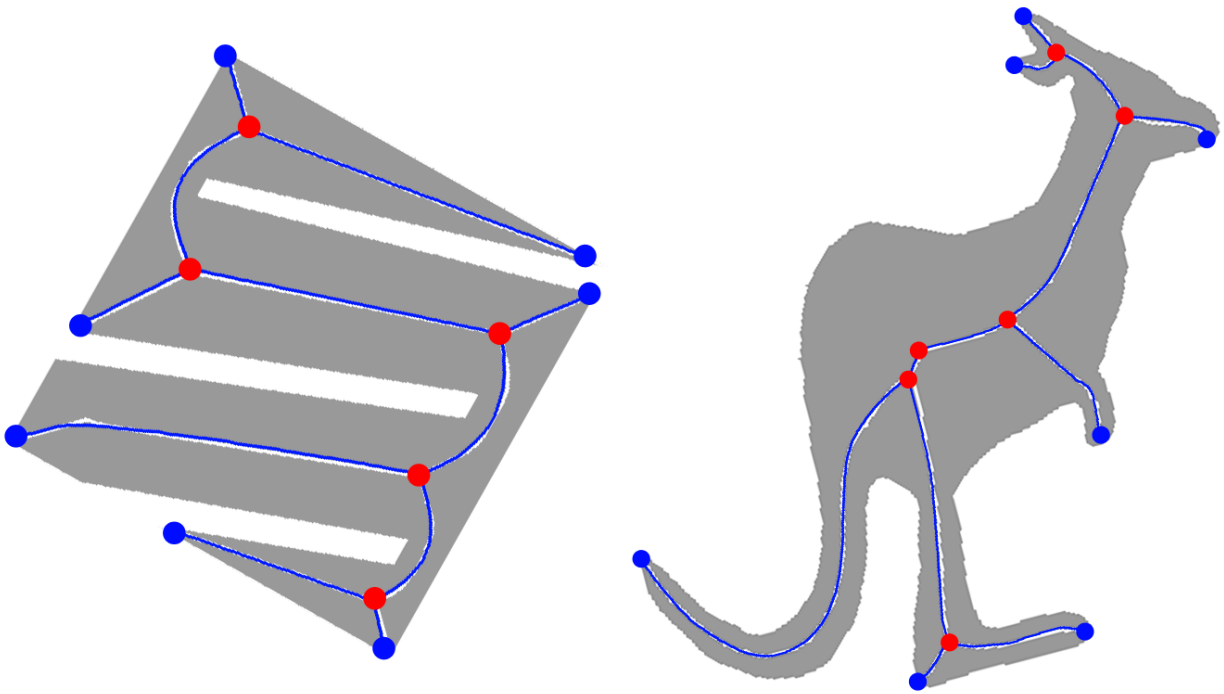
More Examples





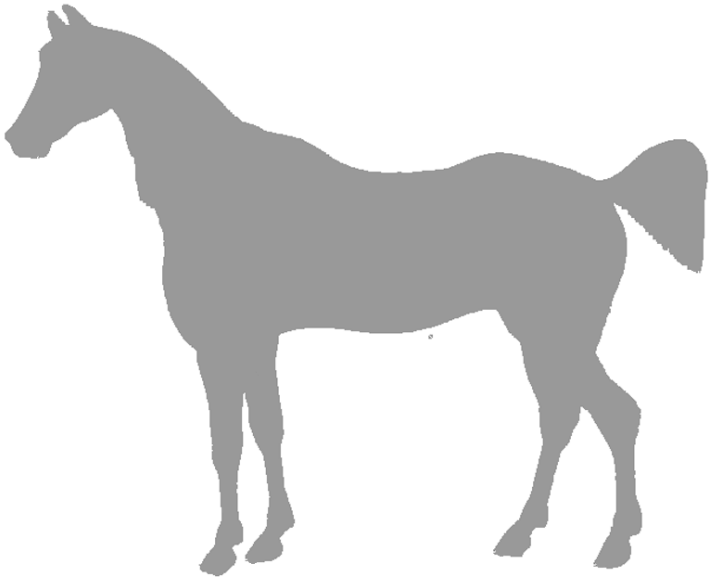
The selected input parameters are the same

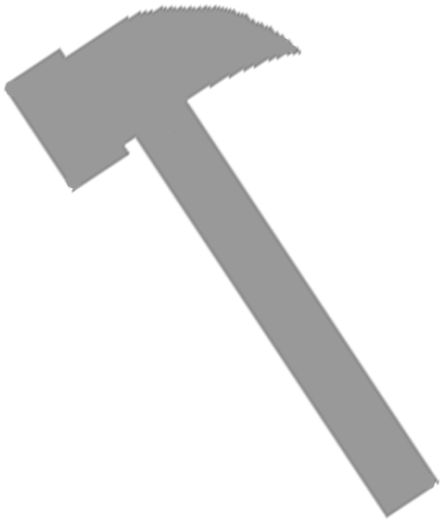
More Examples



The selected input parameters are the same

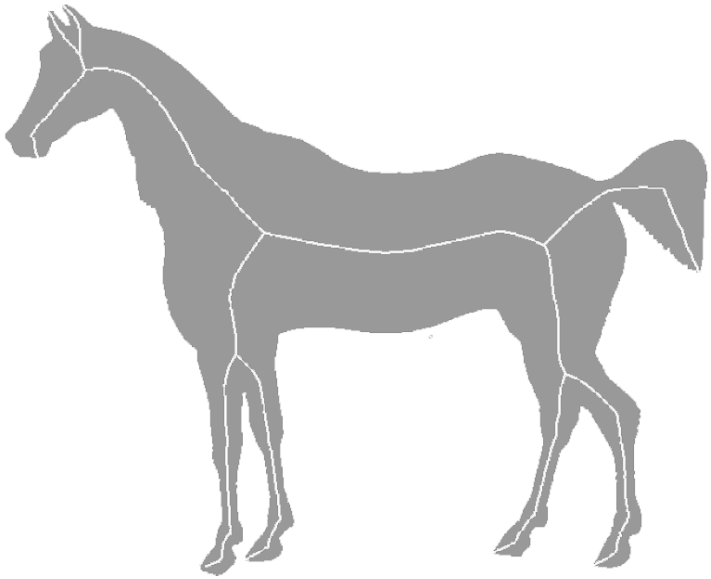
More Examples

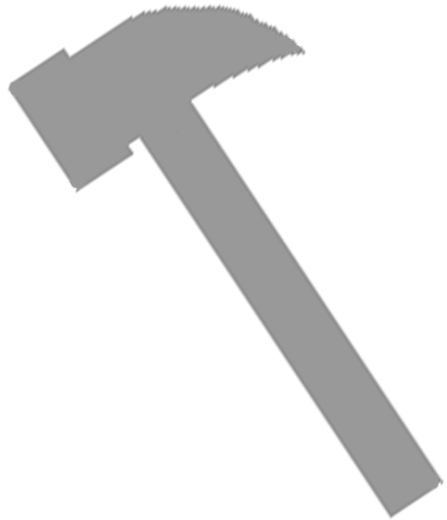




The selected input parameters are the same

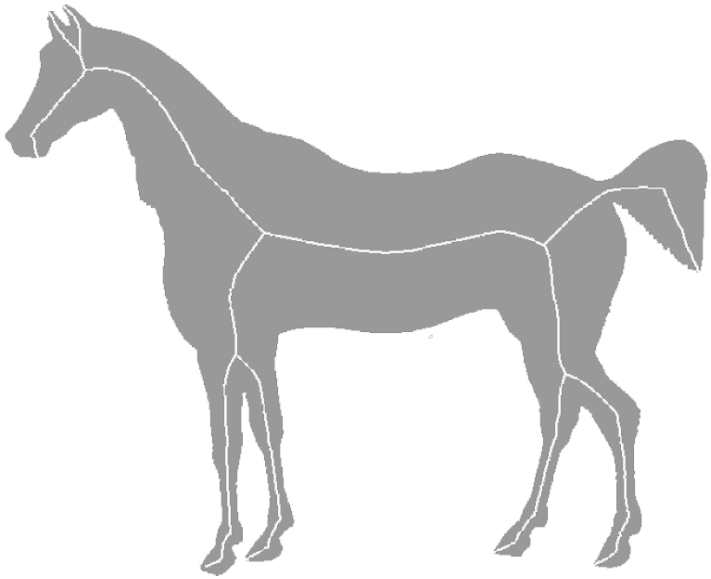
More Examples

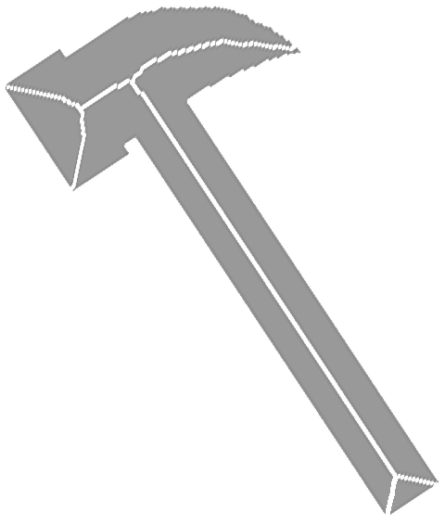




The selected input parameters are the same

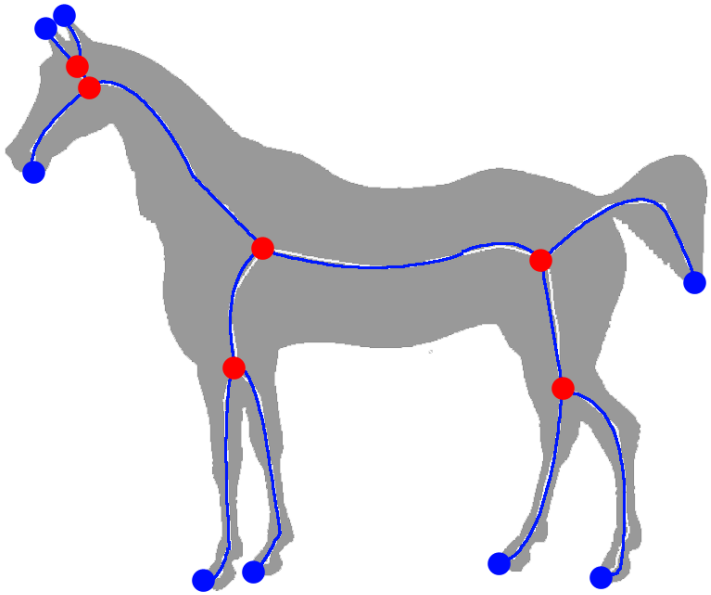
More Examples

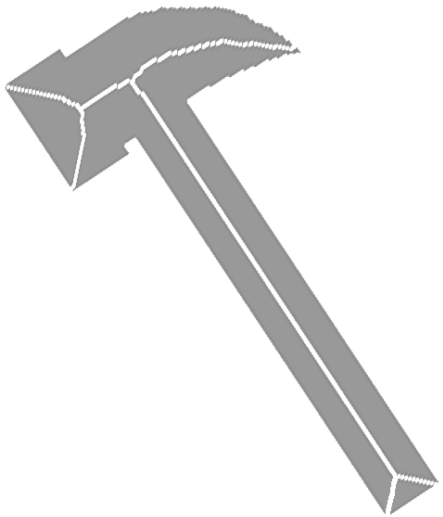




The selected input parameters are the same

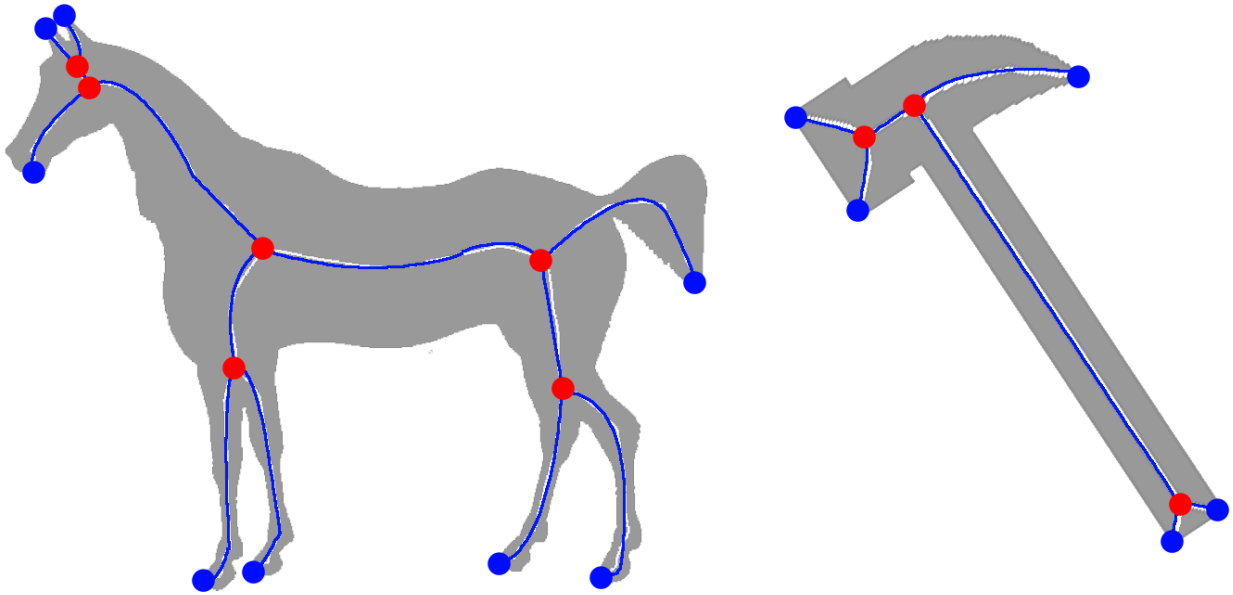
More Examples





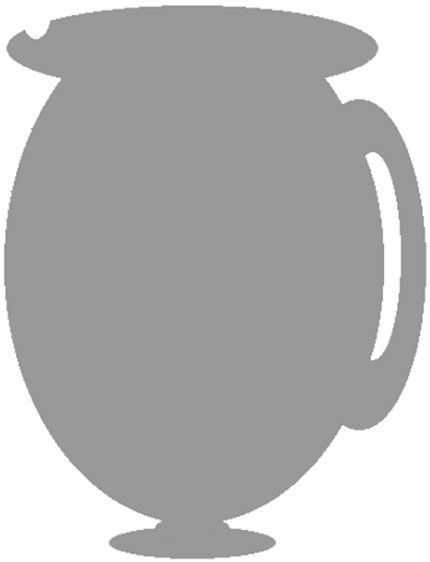
The selected input parameters are the same

More Examples



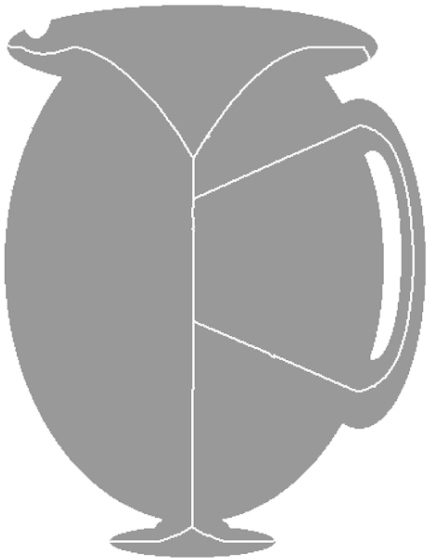
The selected input parameters are the same

More Examples



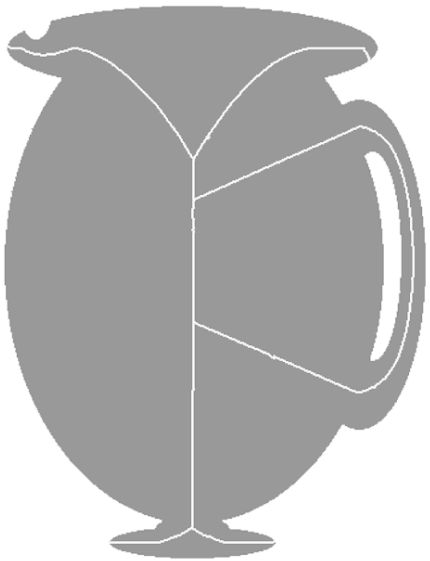
The selected input parameters are the same

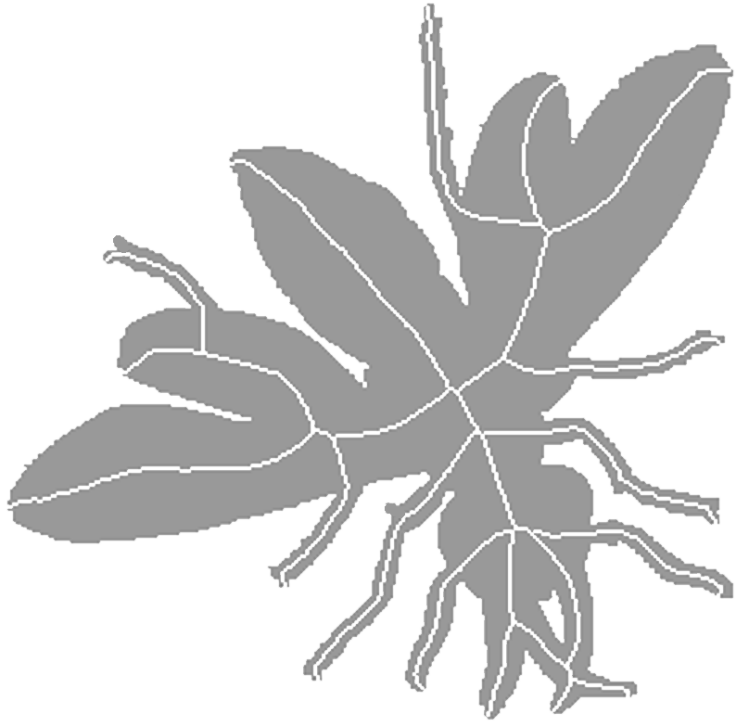
More Examples



The selected input parameters are the same

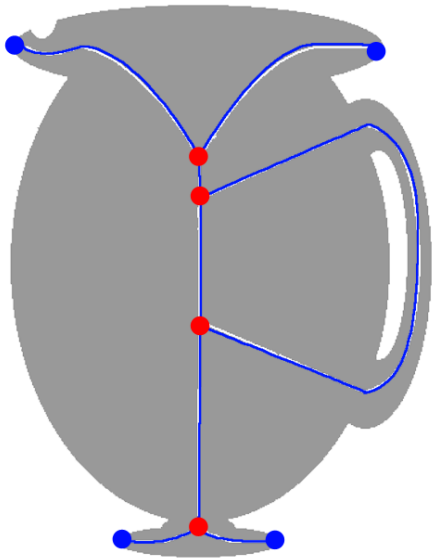
More Examples

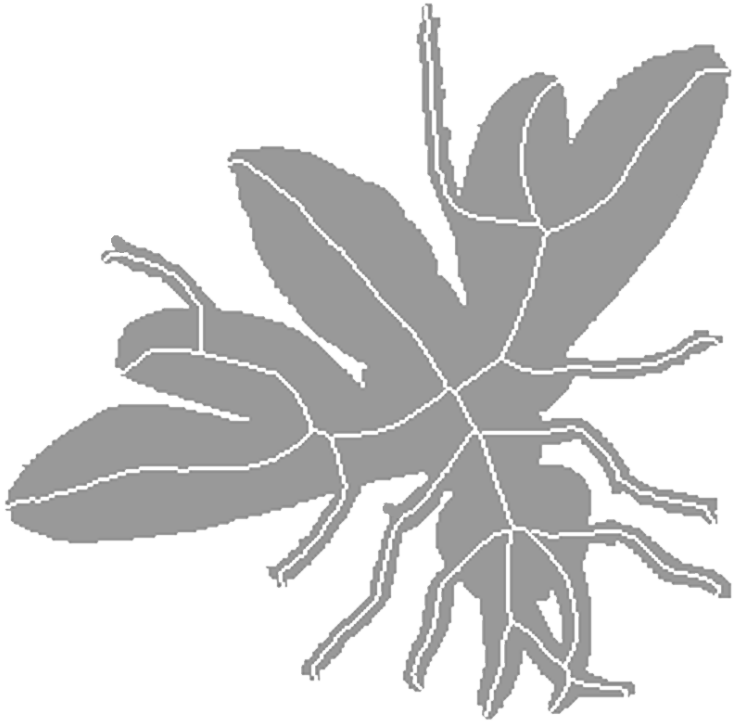




The selected input parameters are the same

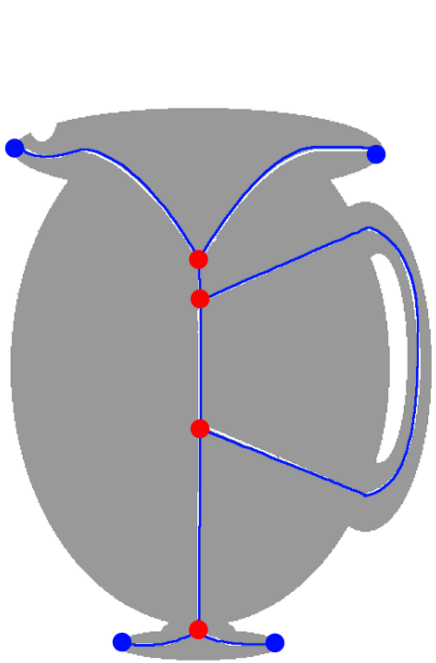
More Examples

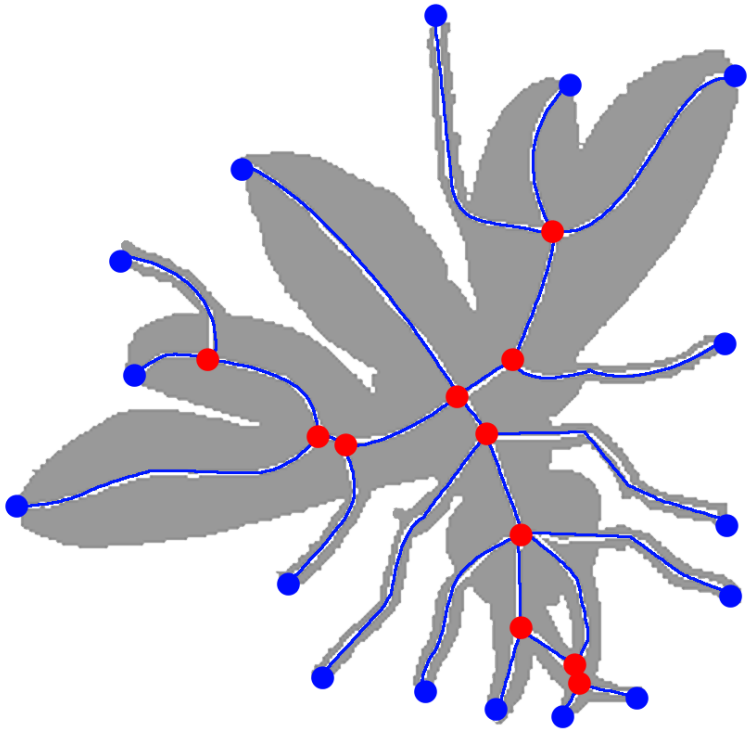




The selected input parameters are the same

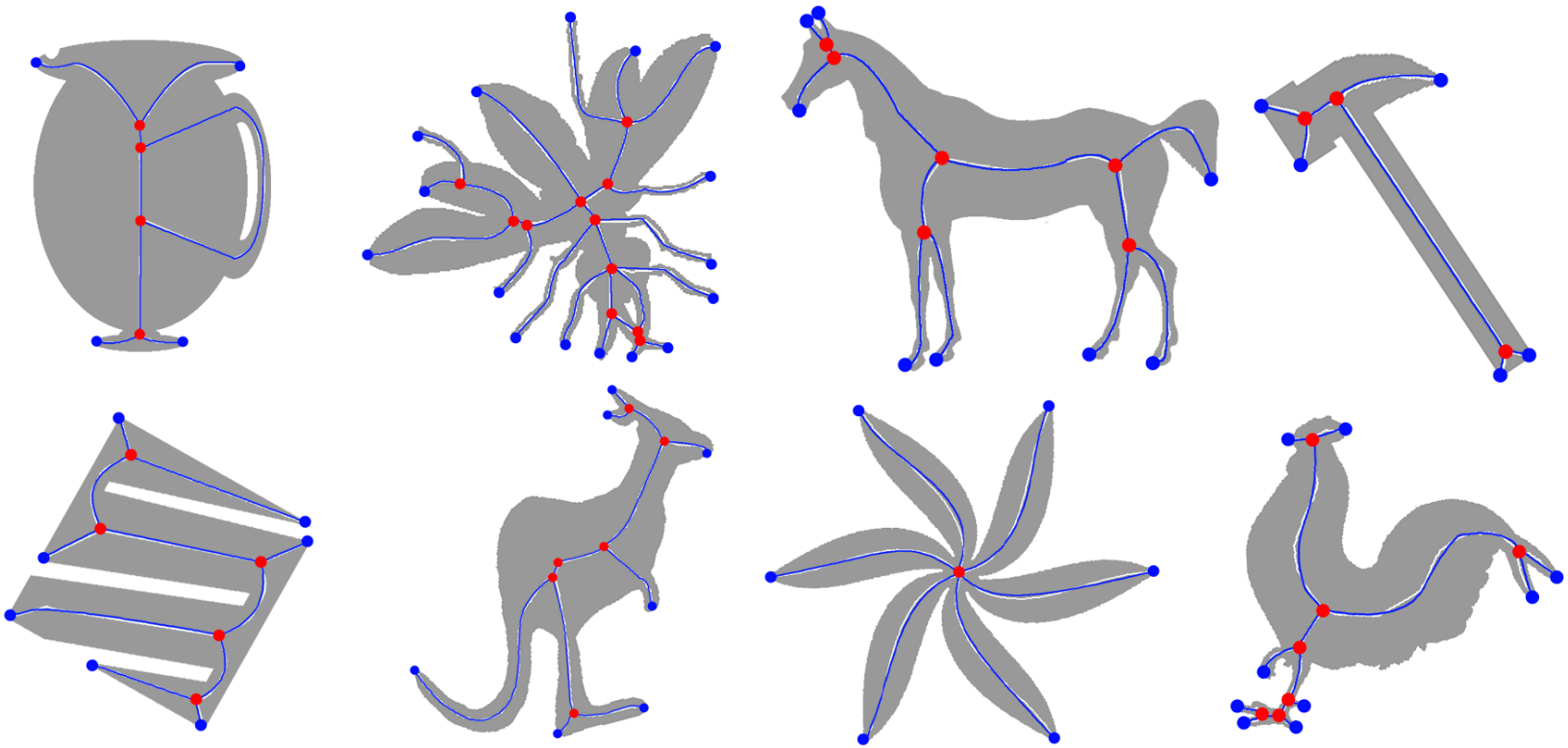
More Examples





The selected input parameters are the same

More Examples



The selected input parameters are the same

Conclusion

- The pruning algorithm presented is fast and gives good results.
- It proved to be very stable with the parameter selection and always preserved connectivity.
- Reconstruction of the original shape area was 98% in average.
- It returns a subset of the medial axis, robust to image rotations and boundary noise.
- It handles complex shapes (i.e. holes).

Thanks

Questions?