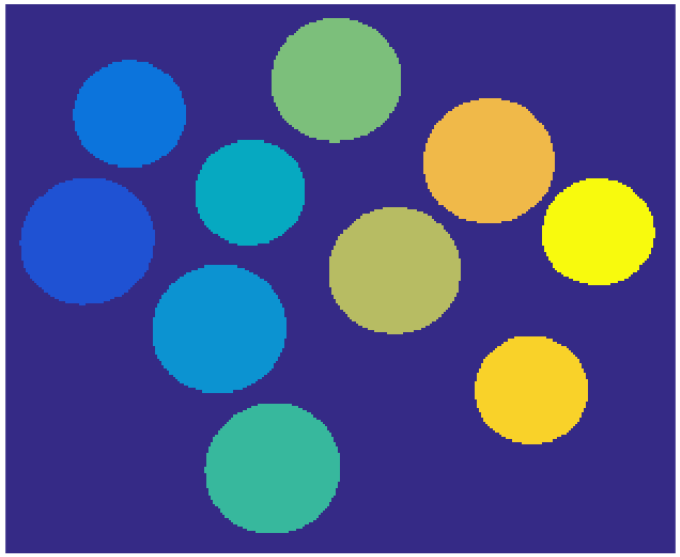
Lecture 6:
Segmentation

Filip Malmberg
Centre for Image Analysis
Uppsala University

What is segmentation?

- Dividing the image into relevant objects and regions.
- Separating objects from background and giving them individual ID numbers (labels).



Why segmentation?

Accurate segmentation of objects of interest in an image greatly facilitates further analysis of these objects. For example, it allows us to:

- Count the number of objects of a certain type.
- Measure geometric properties (e.g., area, perimeter) of objects in the image.
- Study properties of an individual object (intensity, texture, etc.)
- ...

Segmentation is hard!

- Despite decades of active research, there is no ”universal” method for image segmentation.
- Why is it so hard?
  - The human brain is *really* good at recognizing objects around us. This makes us underestimate the difficulty of this task.
  - Segmentation typically requires *high level* knowledge of the objects we want to segment, the imaging process, etc...

There is no universal solution!

Targeted Segmentation

Segmentation is an ill-posed problem...

What is a correct segmentation of this image?

Targeted Segmentation

...unless we specify a segmentation target.

"Segment the orange car from the background"

"Segment all road signs from the background"

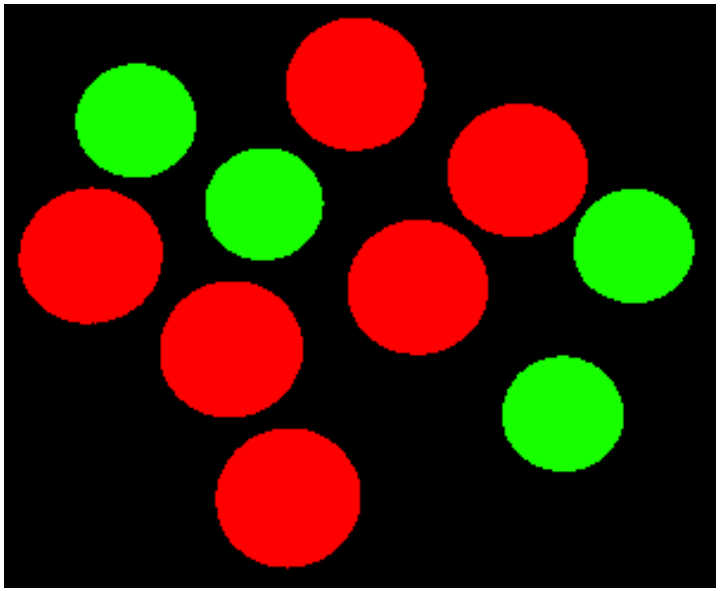
Targeted Segmentation

A segmentation can also be defined as a mapping from the set of pixels to some application dependent target set, e.g.

- {Object, Background}
- {Humans, Other objects}
- {1,2,3,4,...}
- {Healthy tissue, Tumors}

To perform accurate segmentation, we (or our algorithms) need to somehow know how to differentiate between different elements of the target set.

Targeted Segmentation



{background, big coins, small coins}

Segmentation – what does the image data tell us?

Segmentation algorithms are often based on one of the following two basic properties of intensity values:

Similarity
Partitioning an image into regions that are similar according to a set of predefined criteria.

Discontinuity
Detecting boundaries of regions based on local discontinuity in intensity.

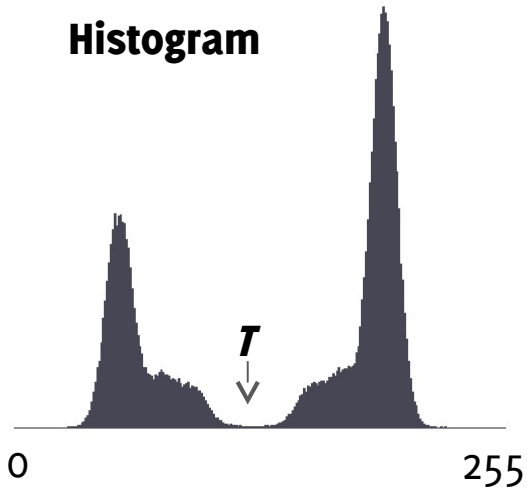
Segmentation by thresholding

Thresholding

Which pixels belong to the object?

A **threshold** ***T***, a gray level intensity, classifies every pixel as belonging to objects (foreground) or background. (Or rather, {dark objects, bright objects}).

Global thesholding



We chose a **threshold *T*** midway between the two gray value distributions.

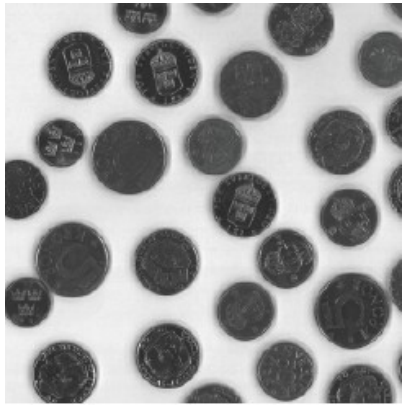
Here: In the tresholded binary image, pixel values below *T* belong to the object (black), pixels above *T* are background (white).

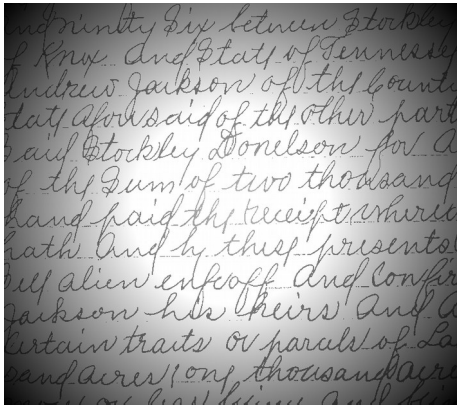
How to find a global threshold

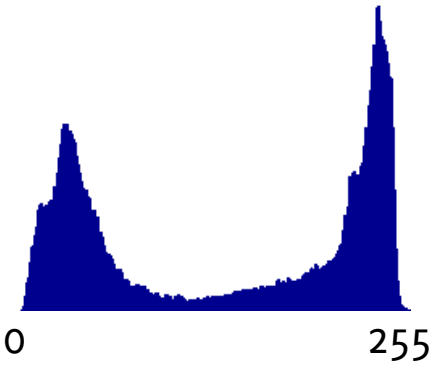
An example method

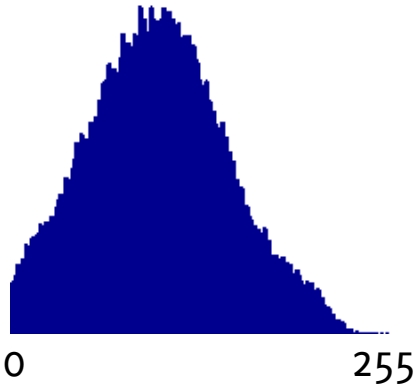
1. Choose initial threshold T_o
2. Define $f(x,y) > T_o$ as background and $f(x,y) < T_o$ as foreground
3. Calculate mean for background μ_{bg} and foreground μ_{fg}
4. Set next threshold $Ti = (\mu_{bg}+\mu_{fg})/2$
5. Repeat 2.-4. until stopping criteria, $Ti = Ti\text{-}1$, is fulfilled

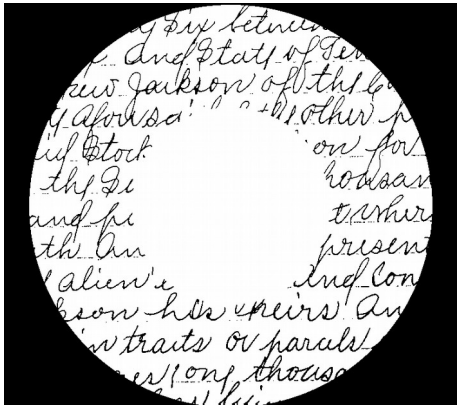
When does intensity based thresholding work?





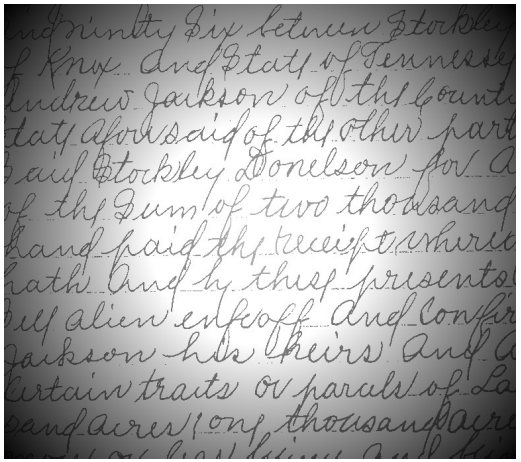


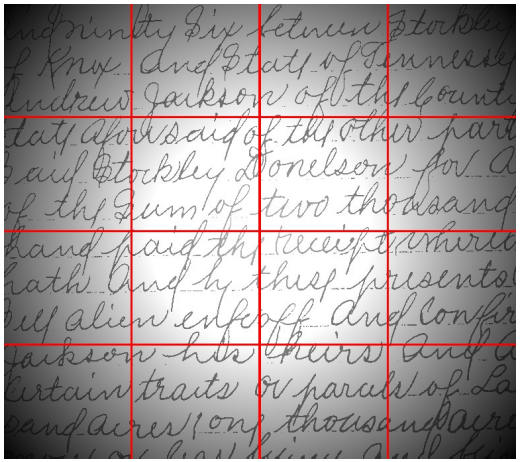


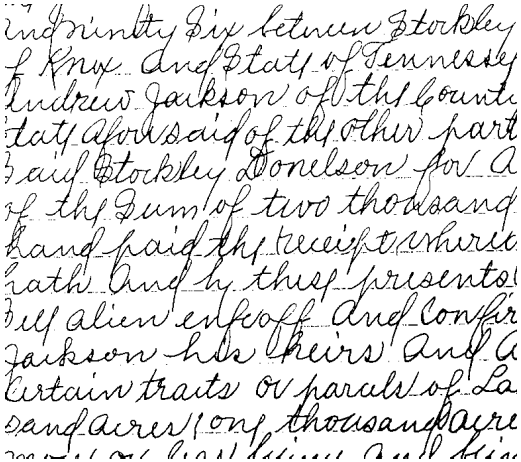


Local thresholding

Determine threshold from local statistics on intensity (i.e. histogram in a local region)







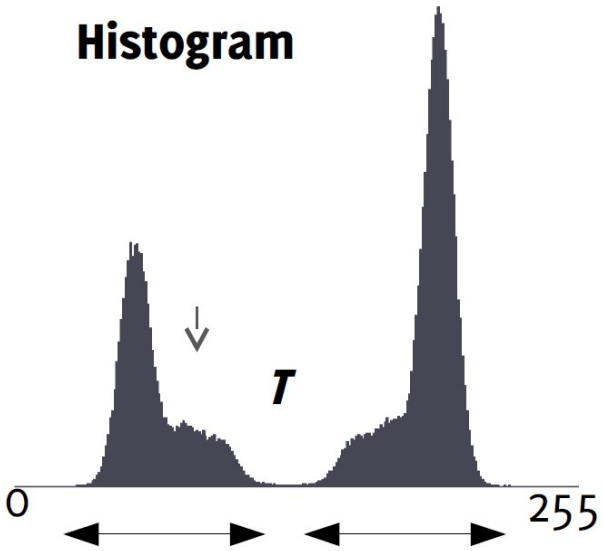
Example method

Subdivide image into non-overlapping rectangles. These rectangles are chosen small enough so that the illumination of each is approximately uniform. Then determine a global threshold for each subimage.

Thresholding in matlab

```
> im = imread('coins.png');
> T = graythresh(im); % Gives value in [0,1]
> BW = im>(T*max(im(:)));
> imagesc(BW)
```

Graythresh uses Otsu's method for finding the threshold. Otsu's method minimizes the intraclass variance.



Hysteresis thresholding

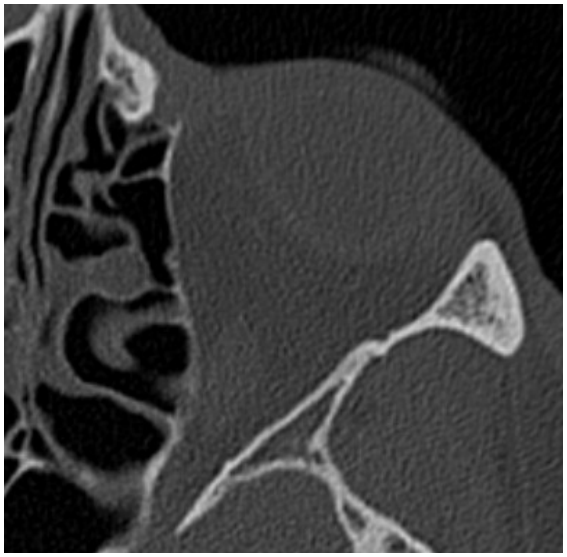
- In the thresholding methods described so far each pixel is labeled individually, without taking into account the labels of neighboring pixels.
- Hysteresis thresholding tries to improve the segmentation by capturing the ”hanging-togetherness” of objects.

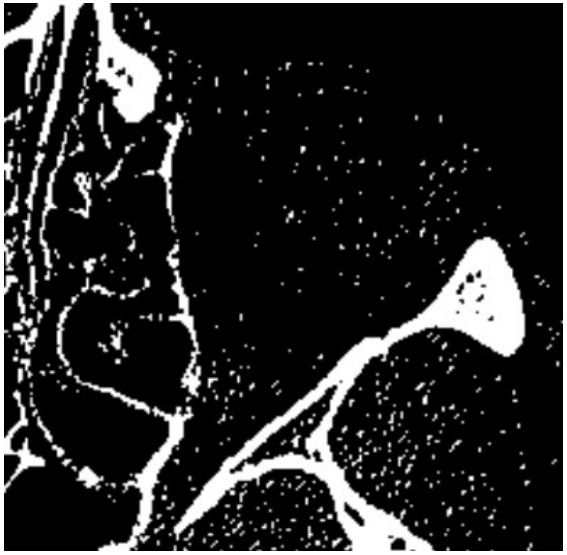
Hysteresis thresholding

- Specify two thresholds: $\mathbf{T}_{high}$ and $\mathbf{T}_{low}$.
- Pixels brighter than $\mathbf{T}_{high}$ are considered to be ”definitely object” and pixels darker than $\mathbf{T}_{low}$ are considered to be ”definitely background”.
- Pixels with intensities between $\mathbf{T}_{high}$ and $\mathbf{T}_{low}$ are considered to be ”uncertain”.
- In a second step, ”uncertain” pixels are labeled as ”object” if they are connected to a pixel with label ”definitely object”. Otherwise, they are labeled as ”background”.

Hysteresis thresholding, example

Hysteresis thresholding, example





Global thresholding

Hysteresis thresholding

Distance transforms

Distance transforms

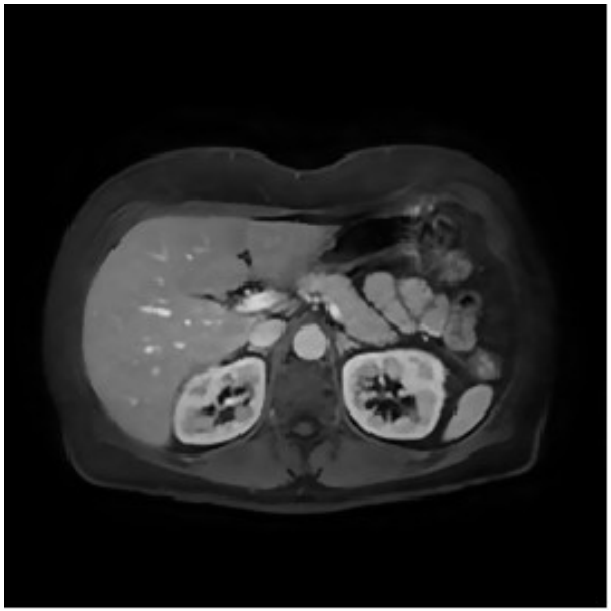
Input: Binary image
Output: In each object (or background) pixel, write the *distance* to the closest background (or object) pixel.

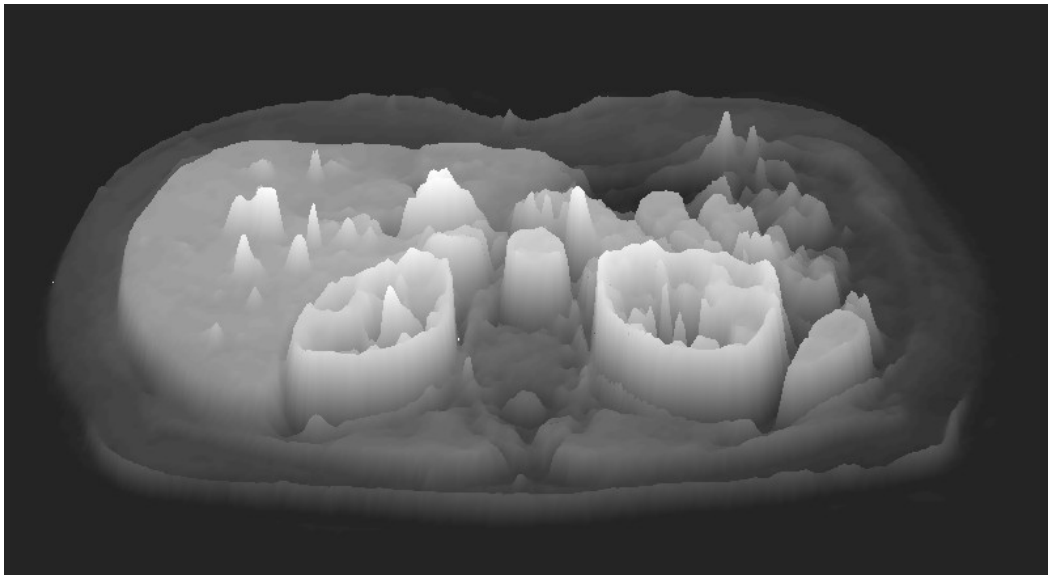
Distance transform algorithm

Input: Binary image
Output: In each object (or background) pixel, write the *distance* to the closest background (or object) pixel.

Watershed segmentation

2D Images as 3D ”intensity landscapes”





Watershed -flooding analogy

Think of the gray level image as a **landscape**.
Let **water rise** from the bottom of each valley (the water from the valley it given its own label).
As soon as the water from two valleys meet, build a dam, or a **watershed**.
These watersheds will define the **borders** between different regions in the image.

The watershed algorithm can be used directly on the image, on an edge enhanced image or on a distance transformed image.

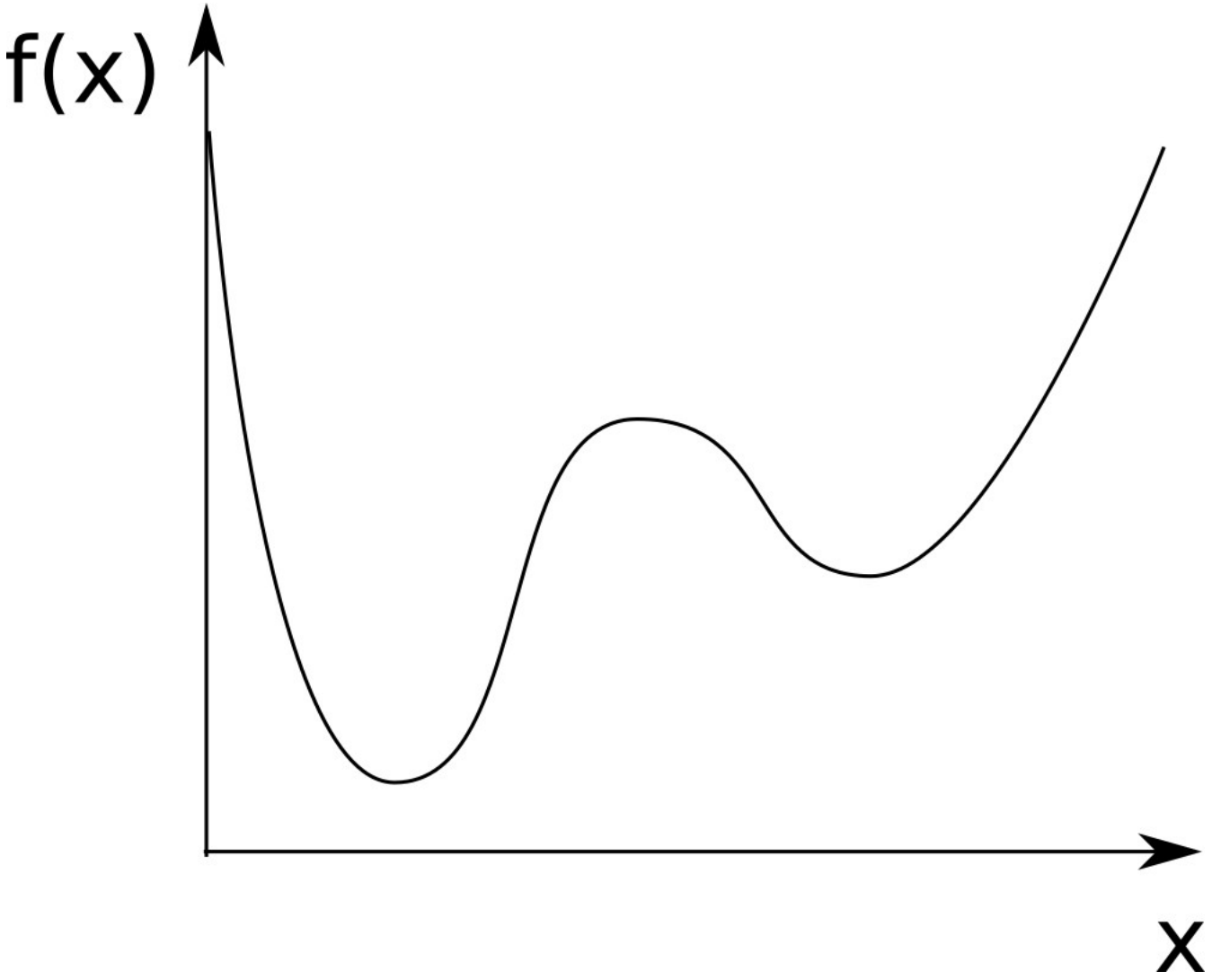
Watershed -drop of water analogy

Think of the gray level image as a **landscape**. A drop of water landing at any point in the landscape will flow down to a local minimum in the landscape.

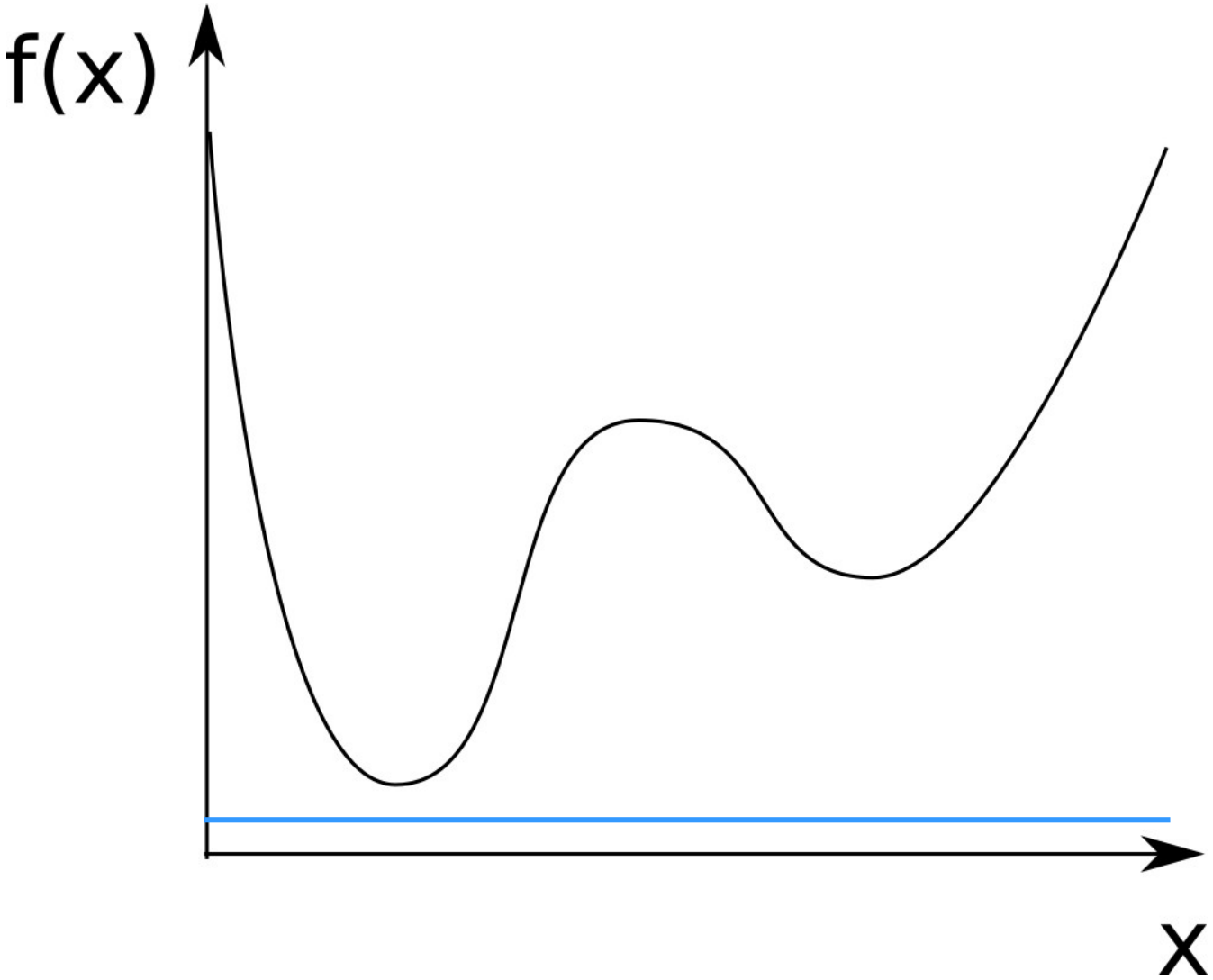
For any local minimum in the landscape, there is a set of points, called the *catchment basin,* from which a drop of water will flow to that given minimum.

The boundaries between adjacent catchment basins form the watershed.

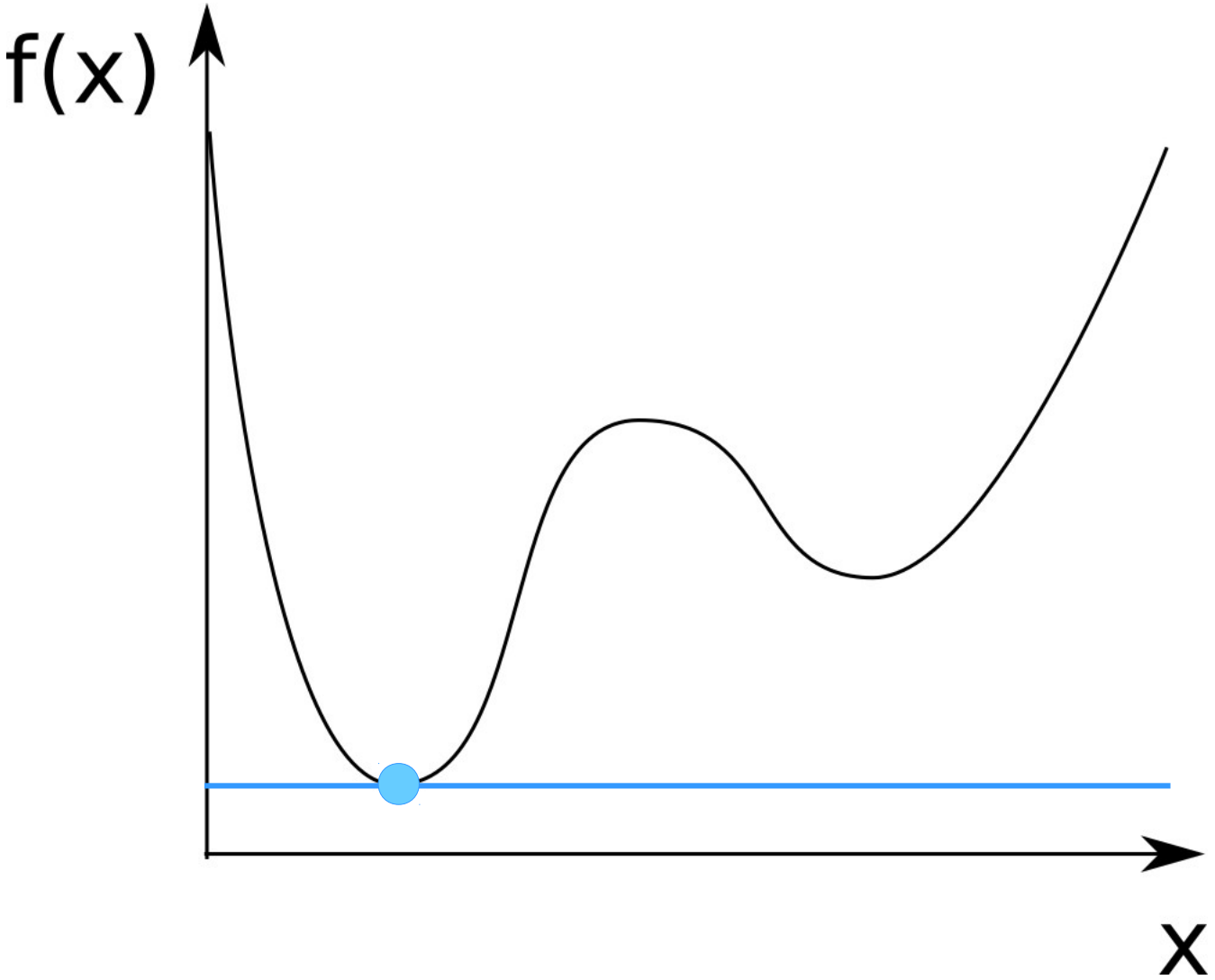
Watershed process



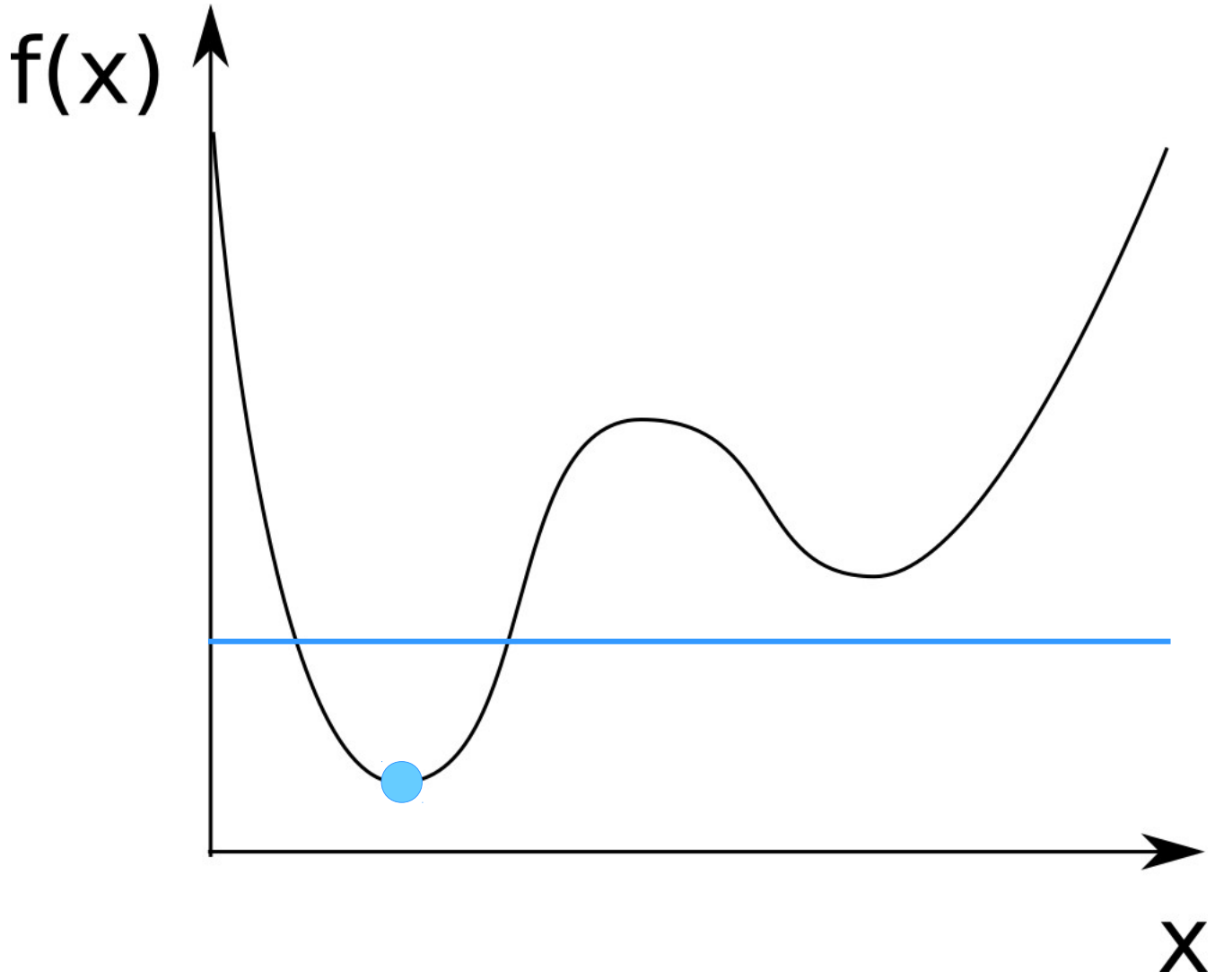
Watershed process



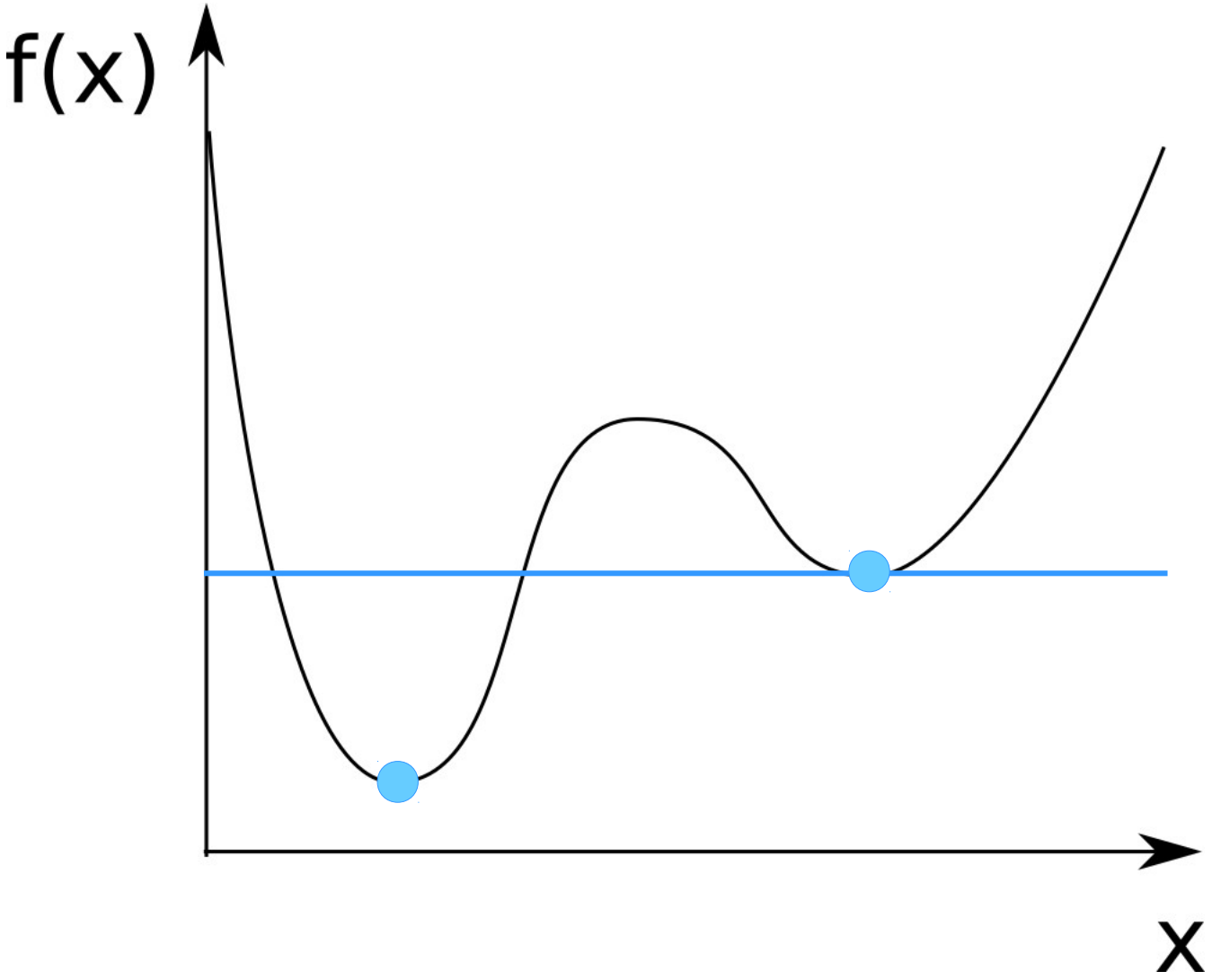
Watershed process



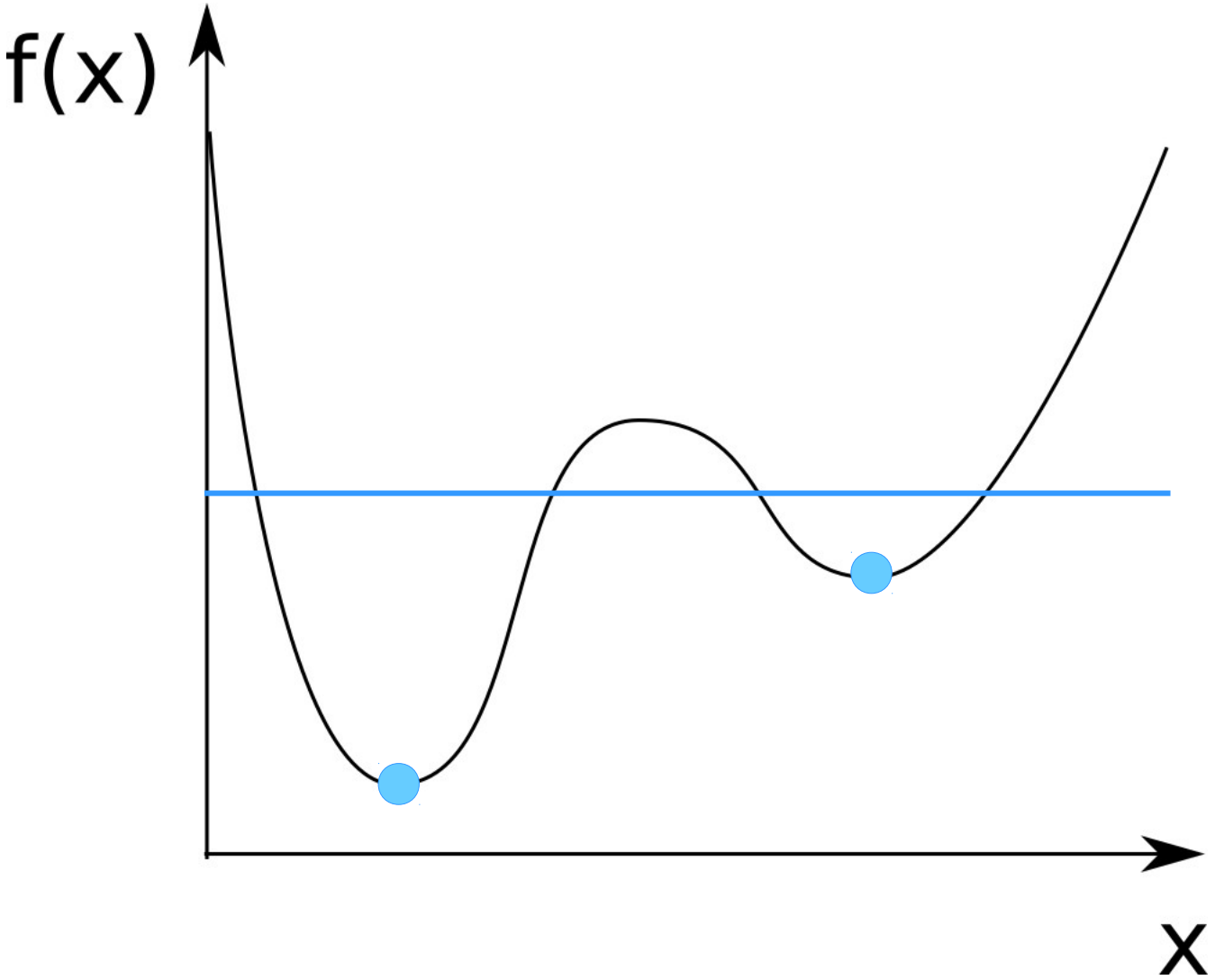
Watershed process



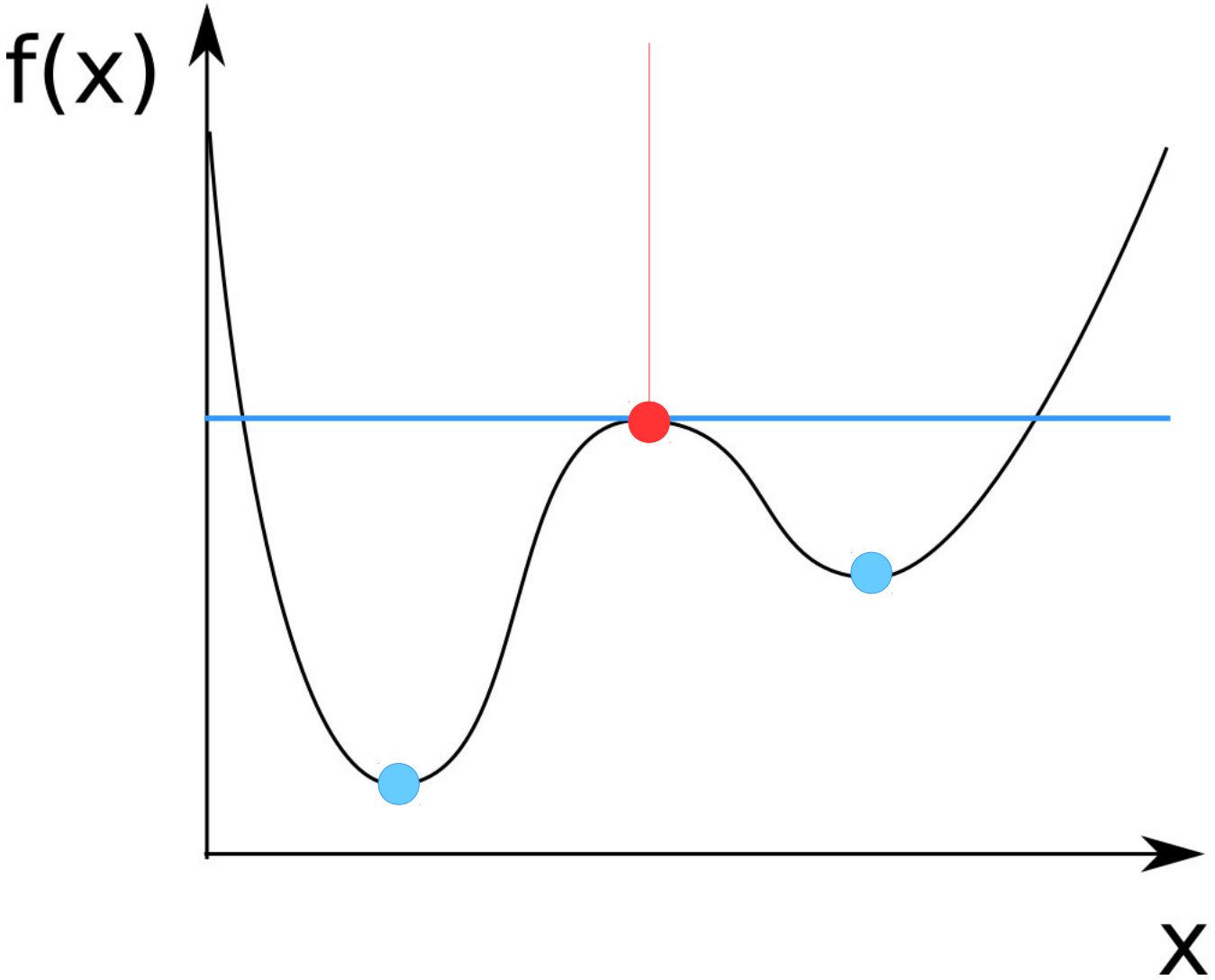
Watershed process



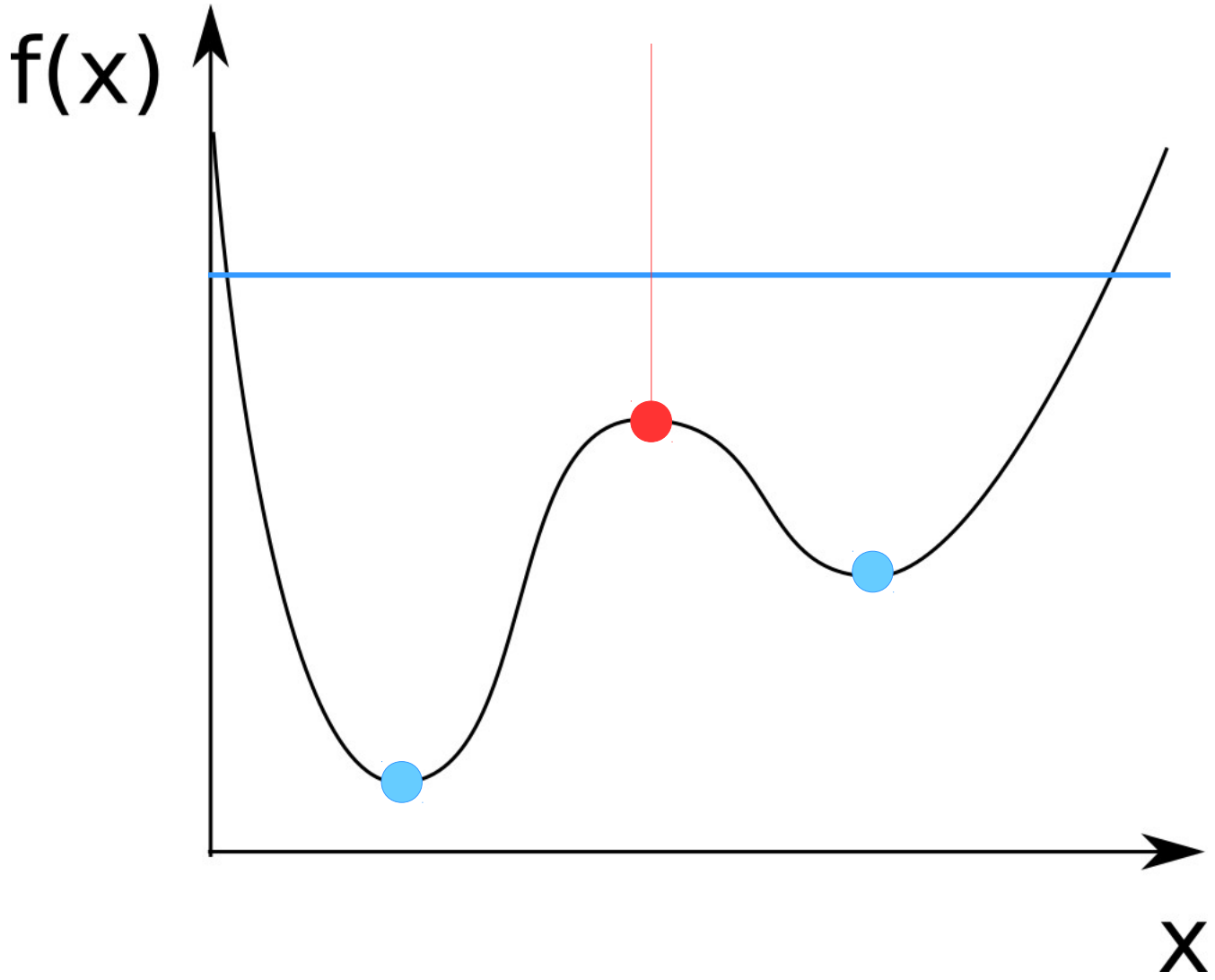
Watershed process



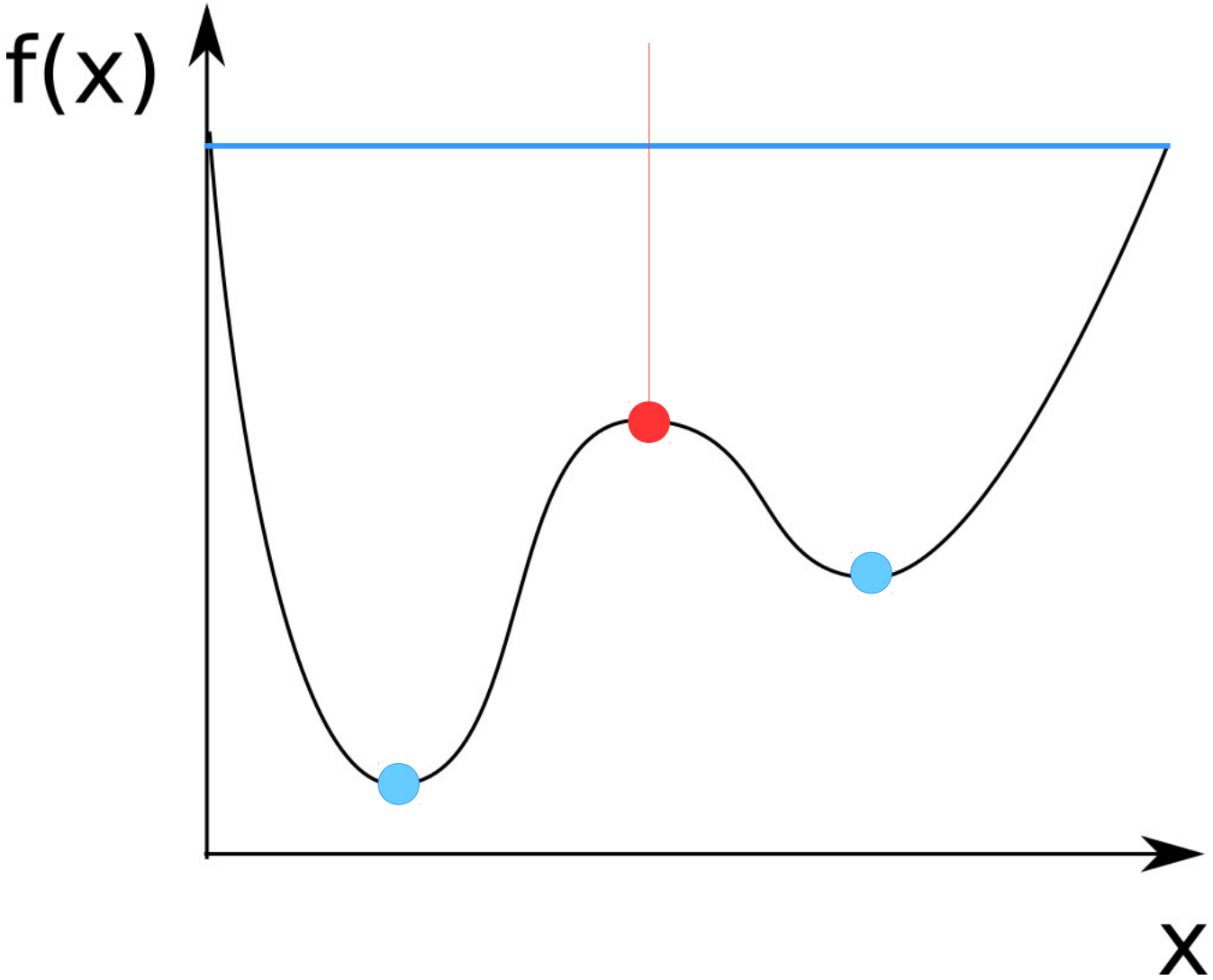
Watershed process



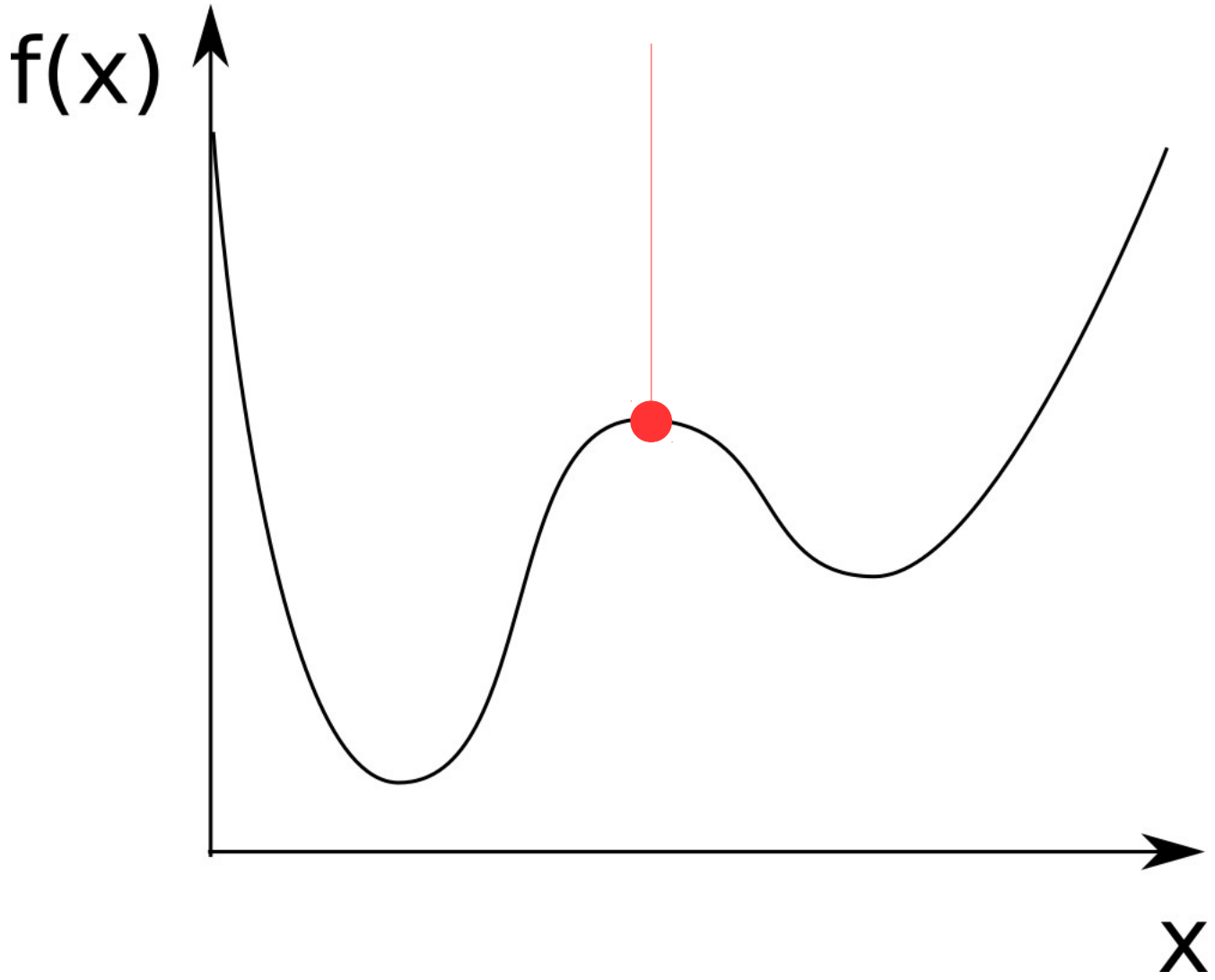
Watershed process



Watershed process

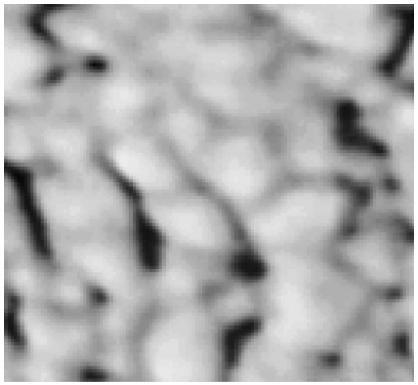


Watershed process

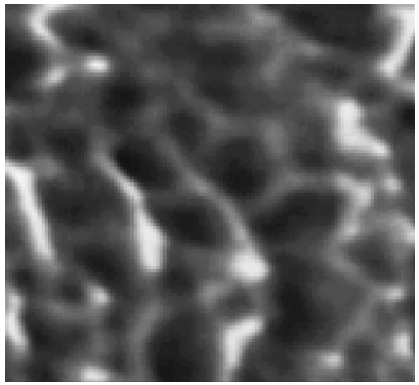


Watershed

Example of watershed directly applied on gray level image:

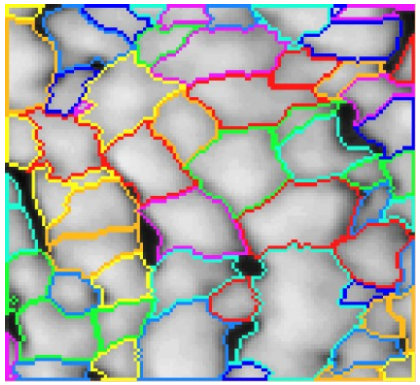


Original image



Inverted image

(starting points are in valleys)

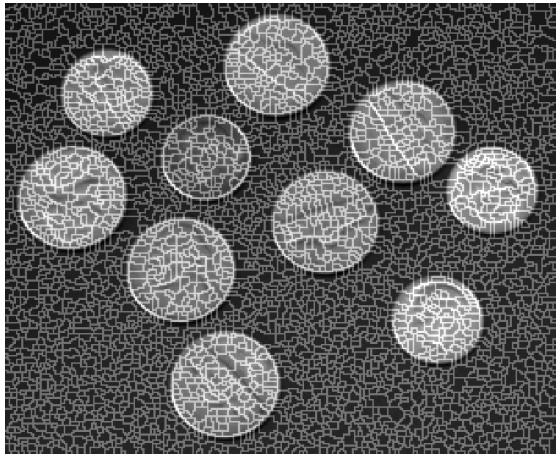


Segmentation result

Watershed

Example of watershed directly applied on gray level image:

Original image

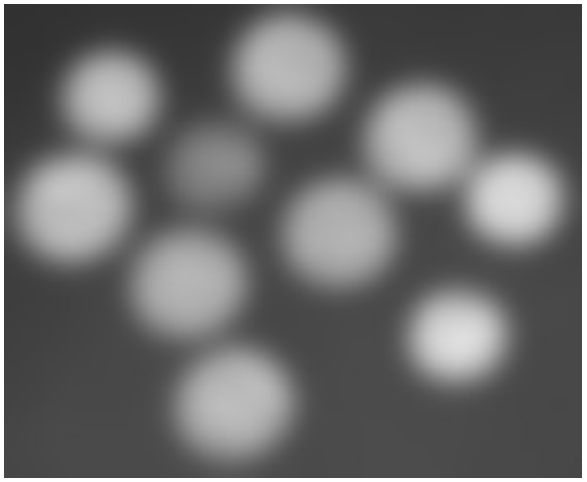


Segmentation result

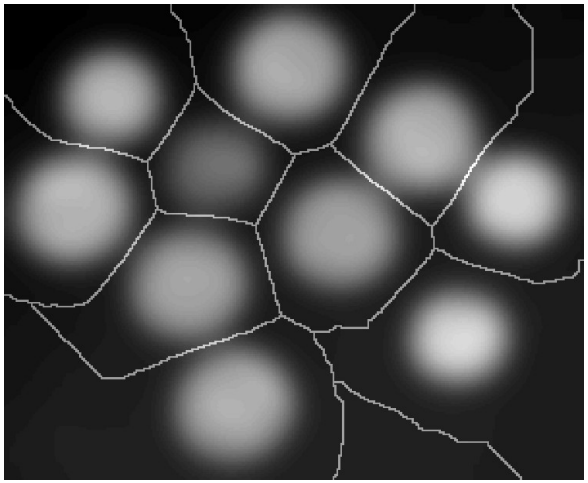
```
>> image=imread('coins.png');
>> ws=watershed(image);
>> imagesc(ws);
>>
```

Watershed

Example of watershed directly applied on gray level image:



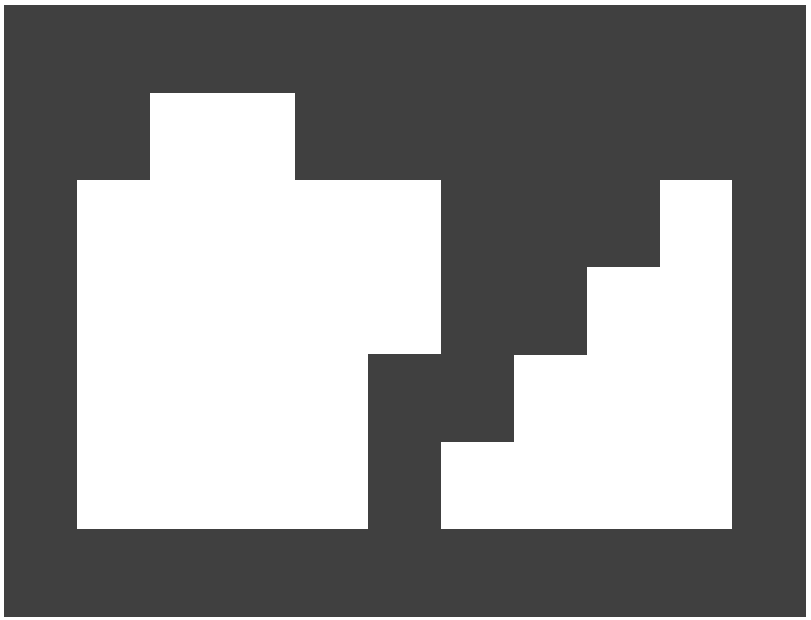
Blurred image



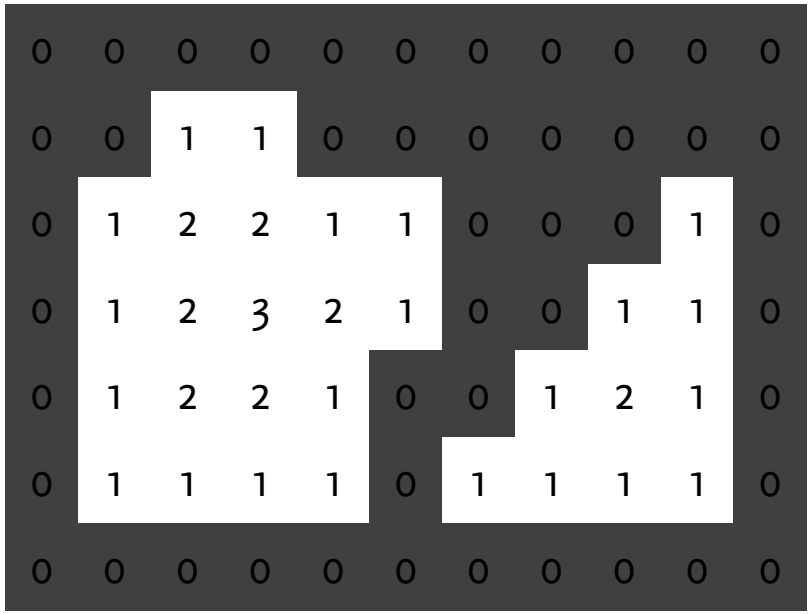
Segmentation result

Distance transforms

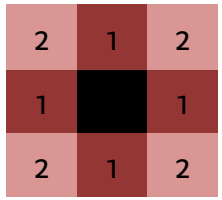
Input: A binary image.
Output: An image where every pixel is labeled with the distance to the nearest background (or object) pixel.



Original image

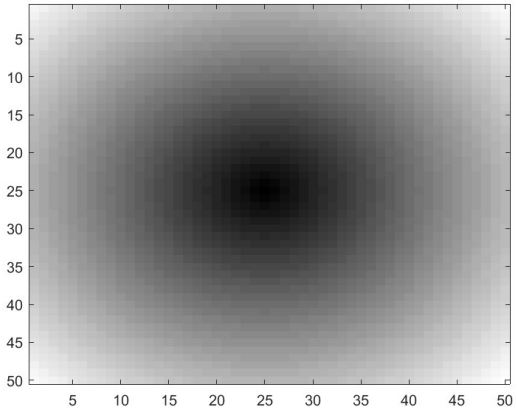


Distance transform

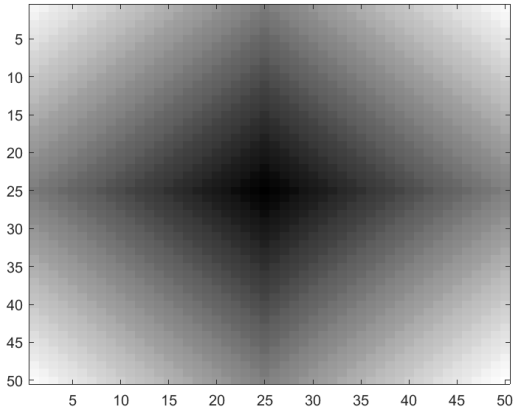


Distance measure

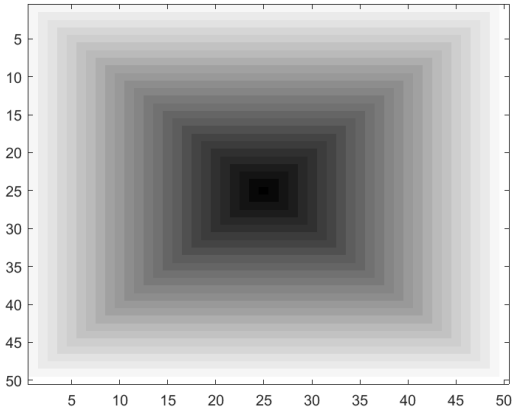
Distance transforms



Euclidean



Cityblock



Chessboard

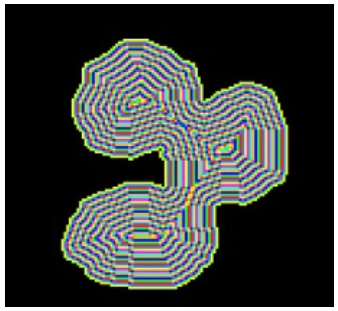
```
binary_image=zeros(50,50);
binary_image(25,25)=1;

figure;
imagesc(bwdist(binary_image, 'euclidean'));
figure;
imagesc(bwdist(binary_image, 'cityblock'));
figure;
imagesc(bwdist(binary_image, 'chessboard'));
```

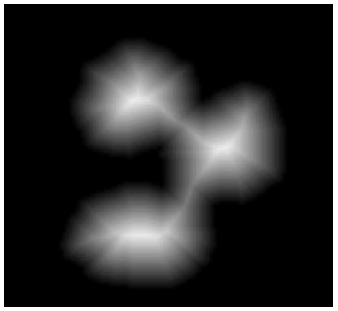
Watershed

Example of watershed on distance transformed image:

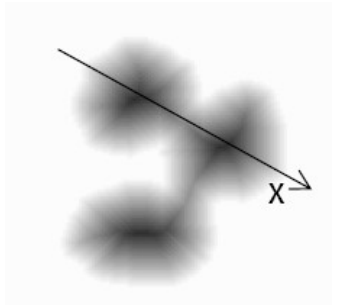
Original image



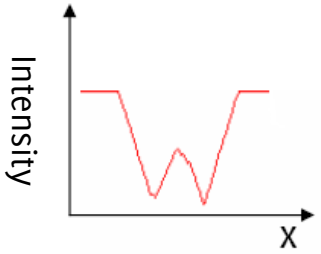
Distance transform



DT as intensity



DT inverse

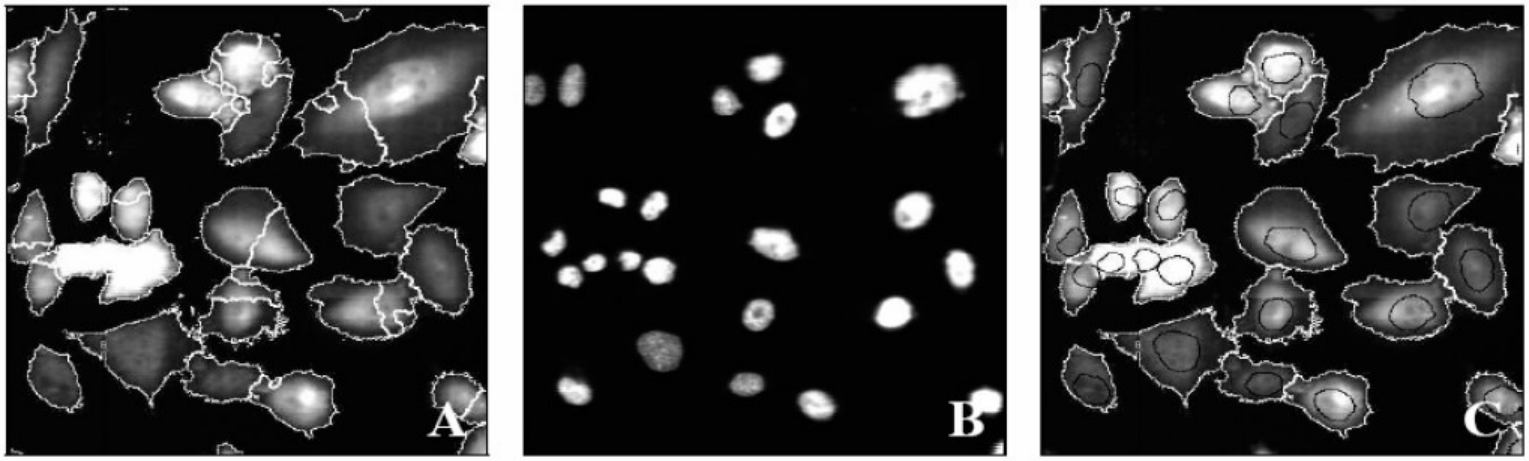


“Intensity landscape“

Segmentation result

Seeded watershed

Instead of starting from all local intensity minima, start flooding from a predefined set of *seedpoints* (pixels).



Oversegmentation

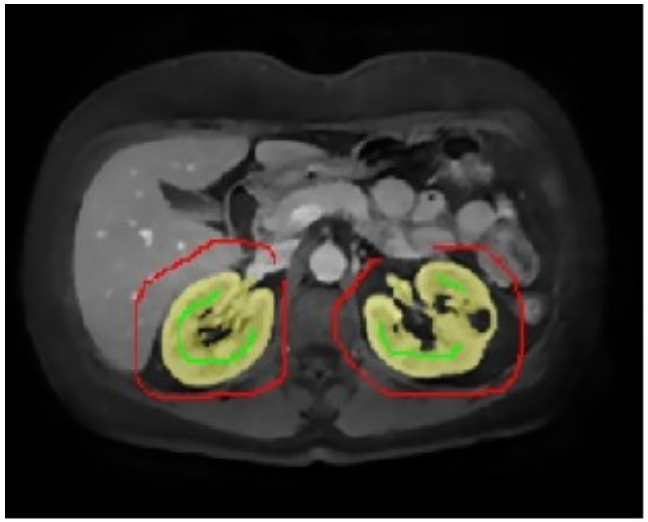
Seeds (nuclei)

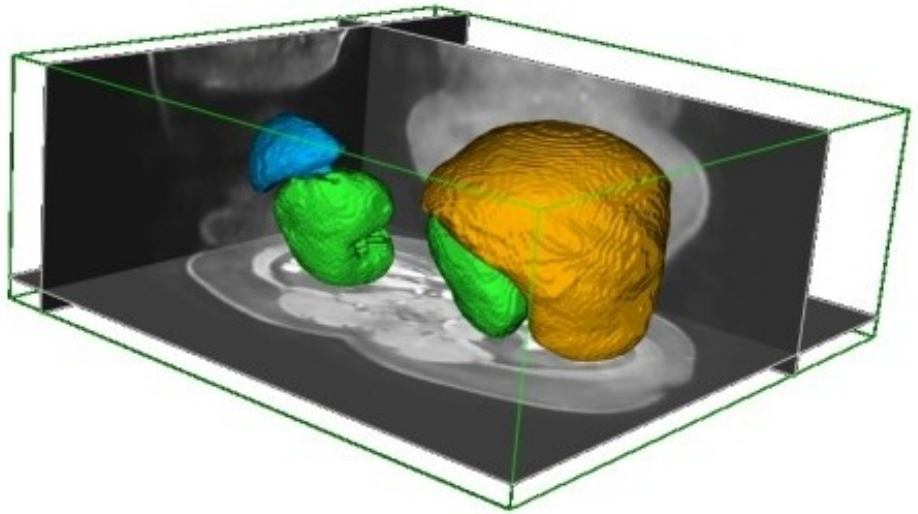
Seeded watershed result

Example for seeded watershed:

Every cell has a cell nucleus, which can be detected by thresholding and watershed segmentation. Using these nuclei as seeds, the cytoplasms are easy to find.

Seeded watershed - demo





Computing watersheds

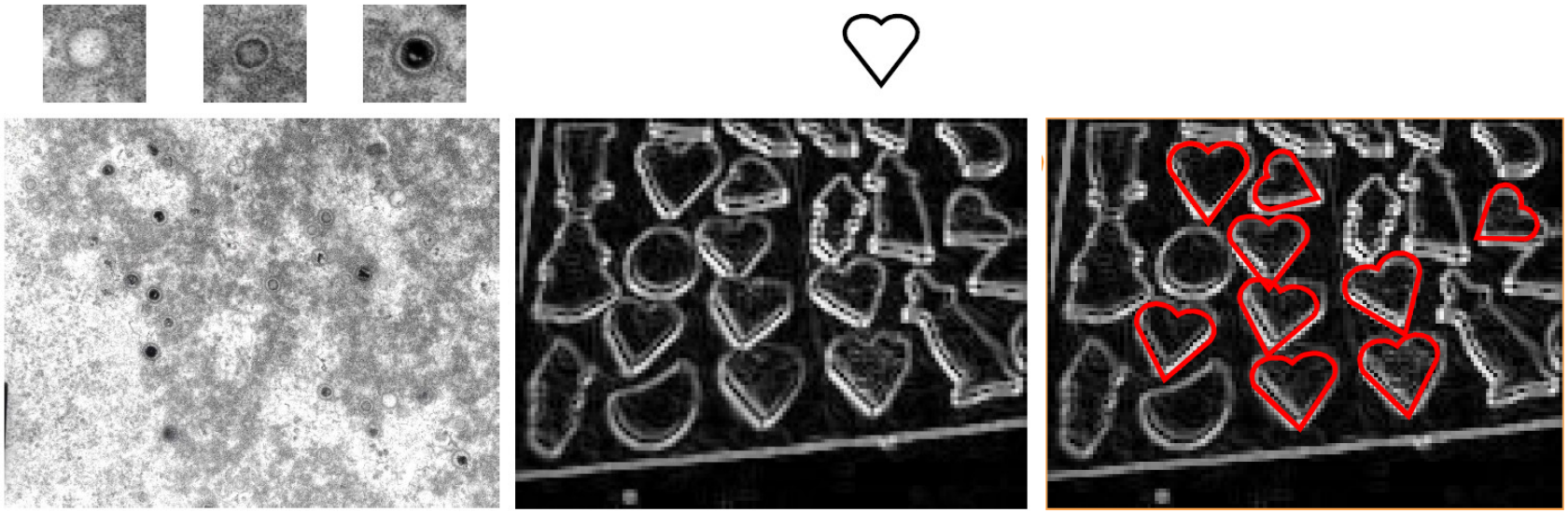
Algorithm by F. Meyer (early 90's)

1. A set of markers, pixels where the flooding shall start, are chosen. Each is given a different label.
2. The neighboring pixels of each marked area are inserted into a priority queue with a priority level corresponding to the gray level of the pixel.
3. The pixel with the highest priority level is extracted from the priority queue. If the neighbors of the extracted pixel that have already been labeled all have the same label, then the pixel is labeled with their label. All non-marked neighbors that are not yet in the priority queue are put into the priority queue.
4. Redo step 3 until the priority queue is empty.

The non-labeled pixels are the watershed lines.

Match-based segmentation

Compare a **template** to the underlying image to find objects with a certain intensity distribution or shape.

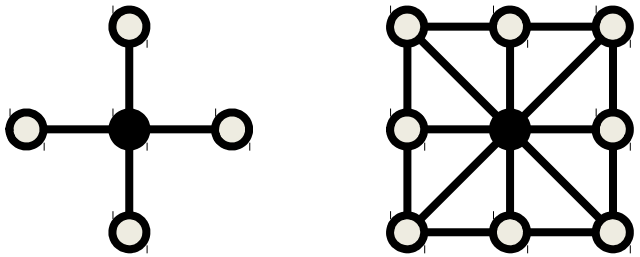


- Can, e.g., be implemented using correlation filters! Strong filter responses correspond to possible matches.
- Computational problem: Testing all relevant variations (e.g, rotation, scaling) of the template.

Mathematical morphology (Filtering/postprocessing of segmentation results)

Mathematical morphology

- Manipulation, or “filtering”, of objects in images, represented as binary masks. (0=background, 1=object)
- Structuring Element (SE): small set or structuring element SE to probe the image under study
- For each SE, define origo
- Shape and size must be adapted to geometric properties for the objects



Mathematical morhphology

Four basic operations:

Erosion $\ominus$

Dilation $\oplus$

Opening $\circ$

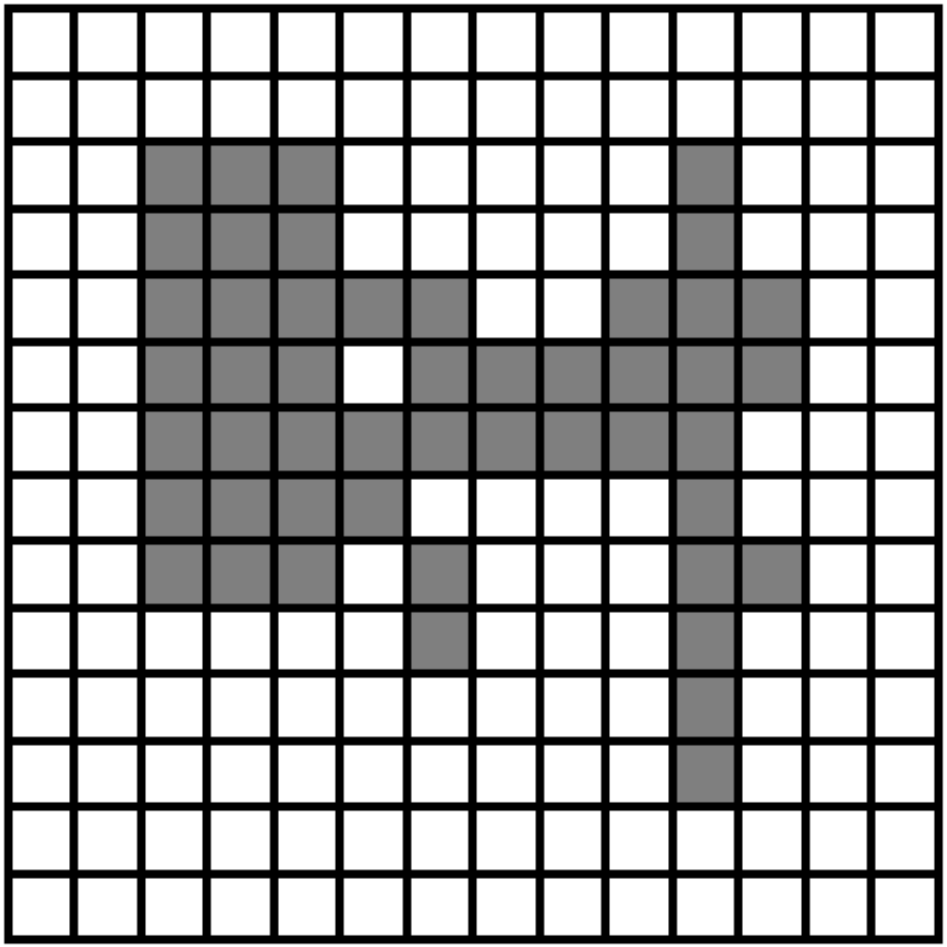
Closing $\bullet$

Erosion (shrinking)

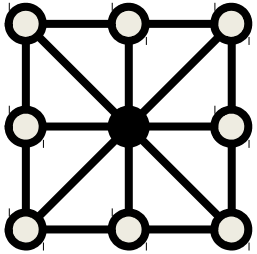
- Does the structuring element fit inside the object?
- Keep only the object pixels corresponding to the origo of the SE when the SE fits entirely inside the object.

(min filter with binary inputs, with mirrored SE)

Example, erosion



SE =

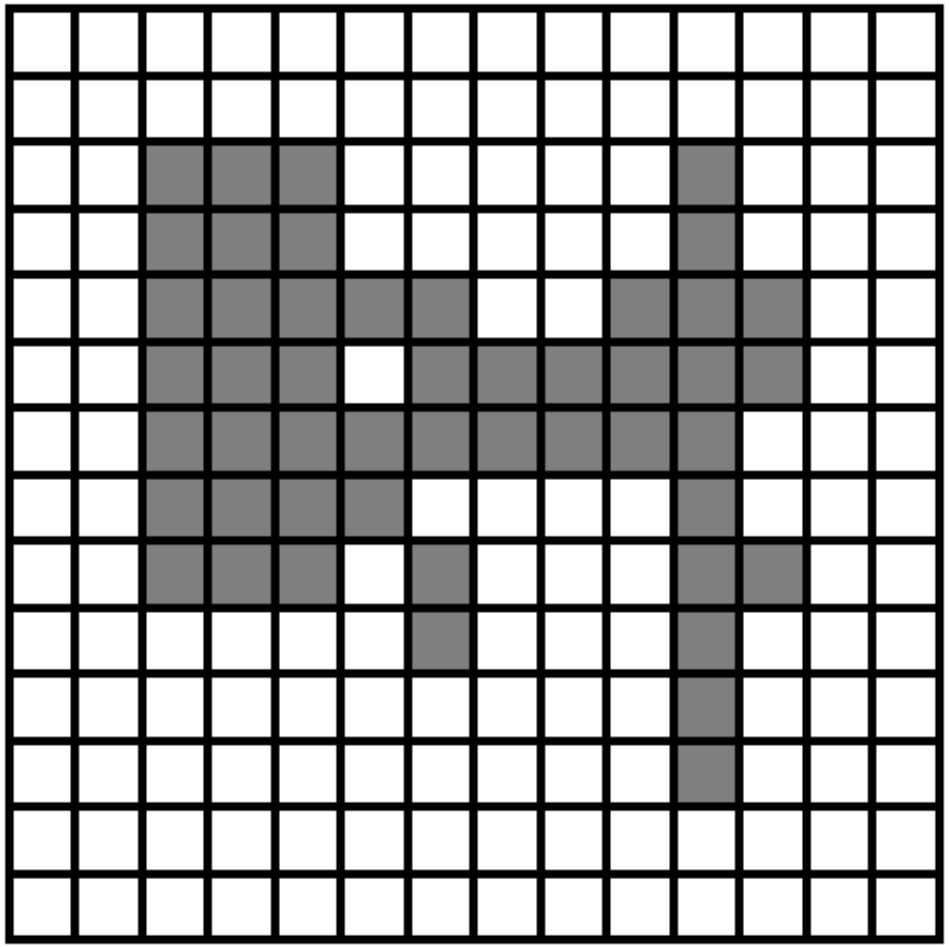


Dilation (growing)

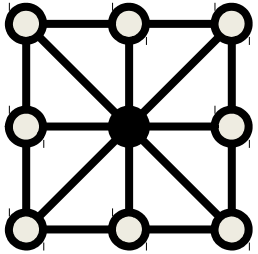
- Grow the object with the SE
- Expand your object with all pixels in the SE, when the origo of the SE hits the object

(max filter with binary inputs, with mirrored SE)

Example: Dilation



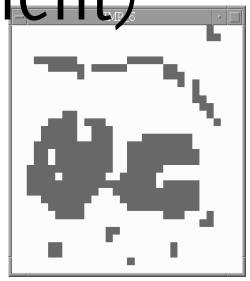
SE=

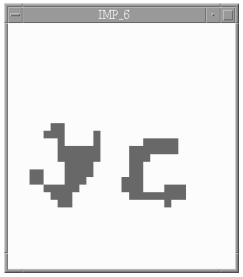


Effects of erosion and dilation

- erosion
  - removal of structures of certain shape and size, given by SE (structure element)

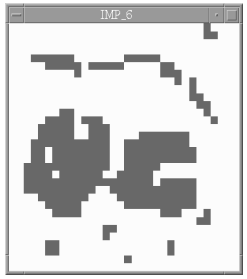
Example
3x3 SE

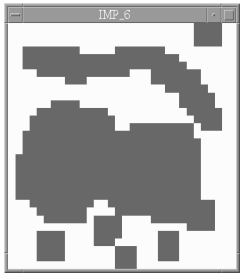




- dilation
  - filling of holes of certain shape and size, given by SE

Example
3x3 SE





Combining erosion and dilation

WANTED:
remove structures / fill holes without affecting remaining parts (overall size of objects)

SOLUTION:
combine erosion and dilation (using same SE)

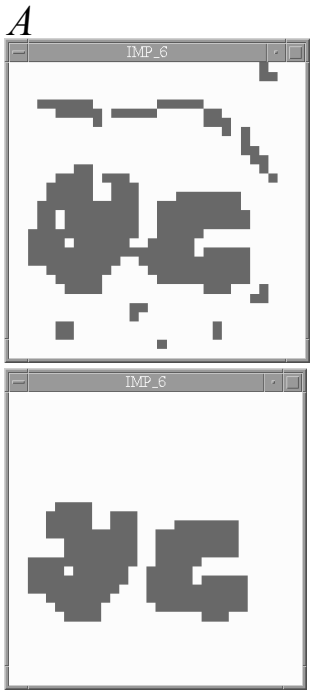
Opening $\circ$

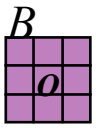
Closing $\bullet$

Opening

erosion followed by dilation, denoted $\circ$

$$A \circ B = \left(A \Theta B\right) \oplus B$$





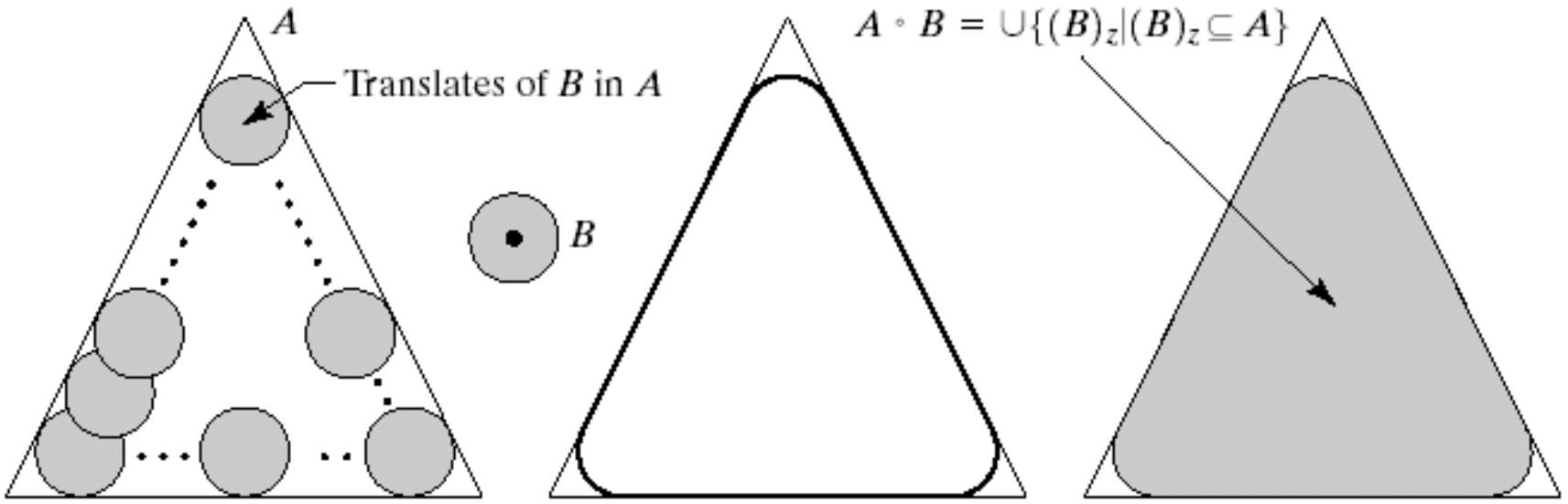
- eliminates protrusions
- breaks necks
- smooths contour

Rolling ball analogy

opening: roll ball(=SE) inside object

see SE as a ”rolling ball”

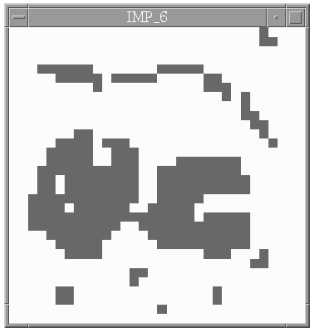
boundary of A∘B = points in B that reaches closest to A boundary when B is rolled *inside* A

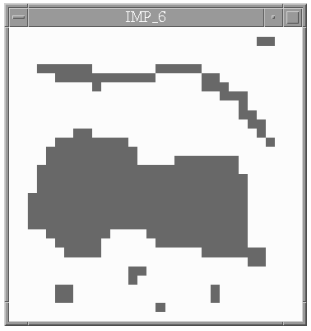


Closing

dilation followed by erosion, denoted •

A





$$A \bullet B = \left(A \oplus B\right) \ominus B$$

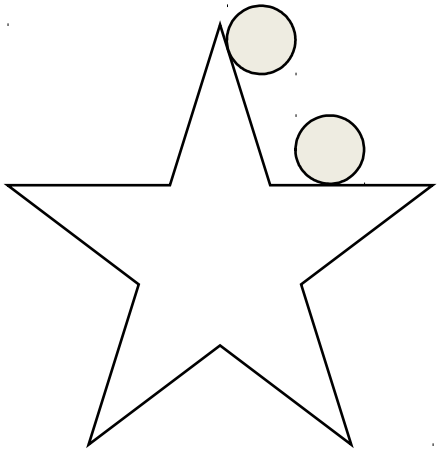
B

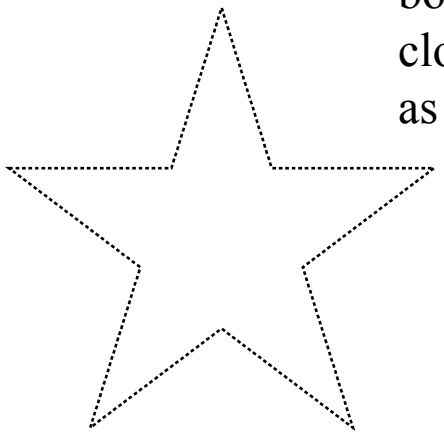
- smooth contour
- fuse narrow breaks and long thin gulfs
- eliminate small holes
- fill gaps in the contour

Rolling ball analogy

Closing: roll ball(=SE) outside object

boundary of $A \circ B$ = points in B that reaches closest to A boundary when B is rolled *outside* A





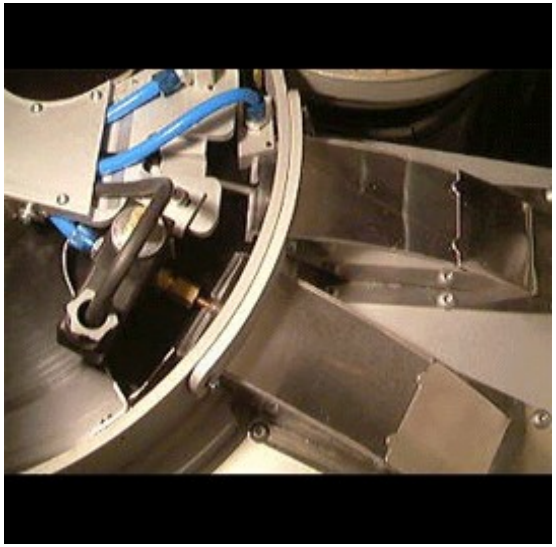
Fill in true border after closing with ball as SE

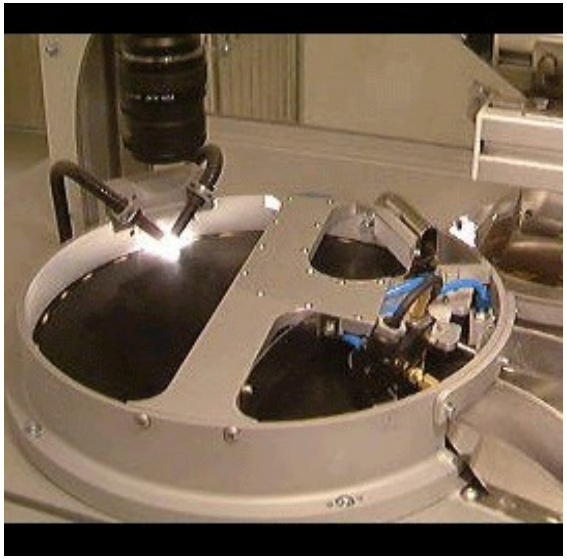
Simplify segmentation by experimental design

When possible:

- Make sure the illumination is even
- Avoid shading
- Have a uniform background (in a different color)
- Avoid reflection (glittering)
- Use a standardized position (industry)

Simplify segmentation by experimental design





Lecture 6:
Segmentation

Filip Malmberg
Centre for Image Analysis
Uppsala University