Segmentation

by Dominic Waithe

10th December 2019

IAFIG-RMS - Bioimage Analysis With Python

Cambridge Bioinformatics Training Centre

TODAYS TALK:

- Introduction to segmentation
- Simple numerical segmentation
- Feature finding methods
- Advanced numerical segmentation methods

UK Research and Innovation

RMS Royal Microscopical Society

MRC Weatherall Institute of Molecular Medicine

MRC Medical Research Council

UNIVERSITY OF OXFORD

Introduction to segmentation

"segmentation of nontrivial images is one of the most difficult tasks in image processing. Segmentation accuracy determines the success or failure of computerized analysis procedures."
R. Gonzalez and R. Woods (*Digital Image Processing*)

Types of segmentation.

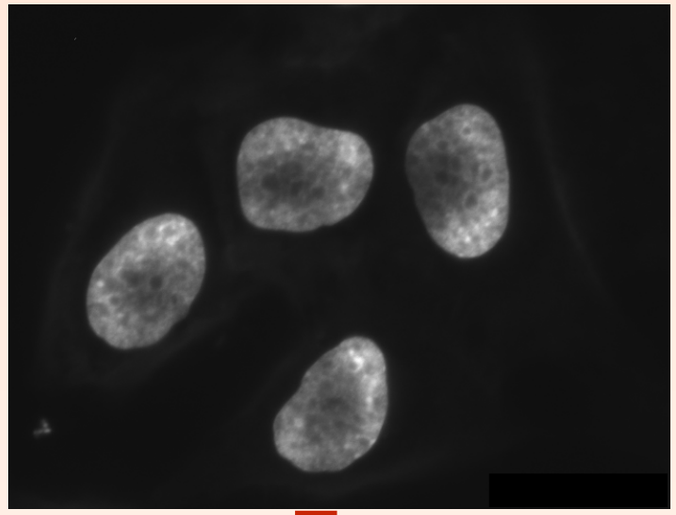
Segmentation is the process of partitioning a digital image into multiple segments (sets of pixels).

There are many forms and variations.

This is usually the first step in an automated computer vision application which will result in some kind of measurement.

Don’t mistake —->

For being the only type of segmentation.

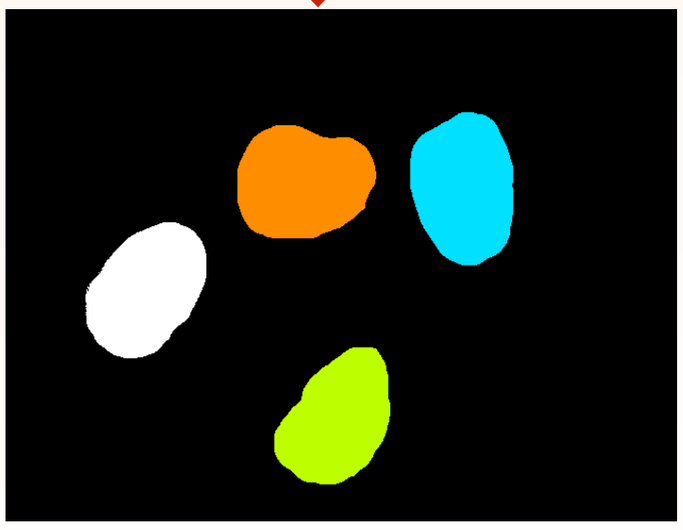


Input

Intensity distribution.

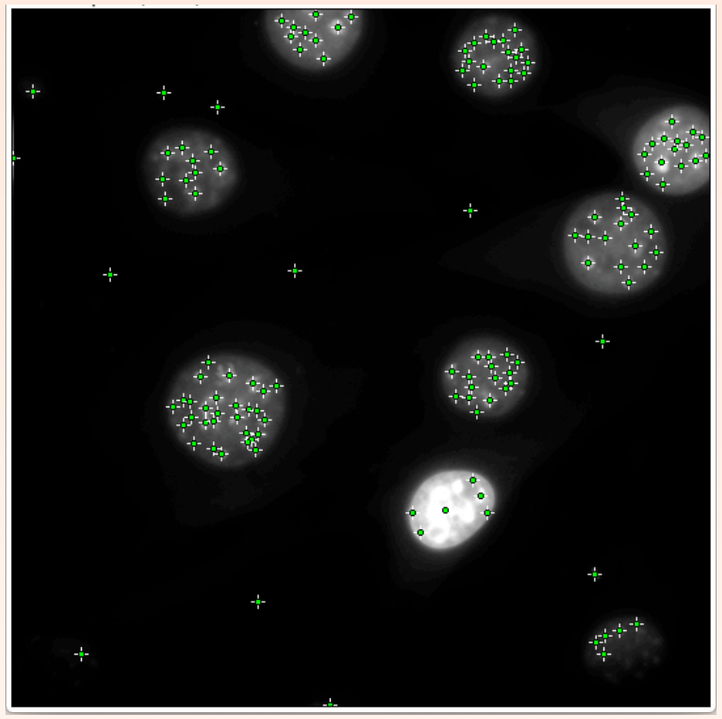
Semantic segmentation

e.g. foreground and background.

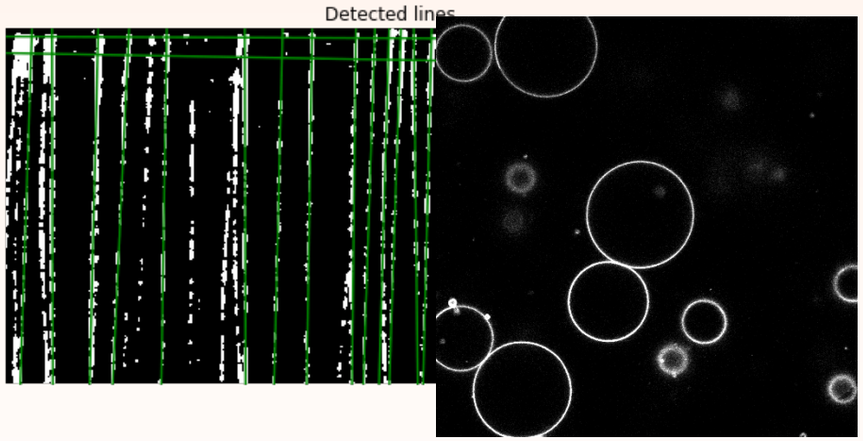


Semantic instance level segmentation. I.e. foreground is also split into different cells (instances)

Other types of segmentation

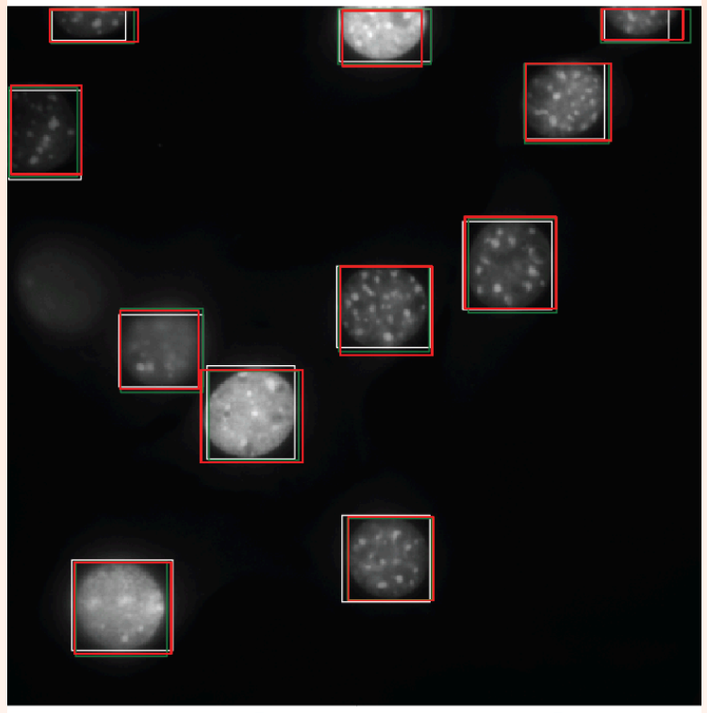


Point detection (maxima finding)
detection of points in the image.

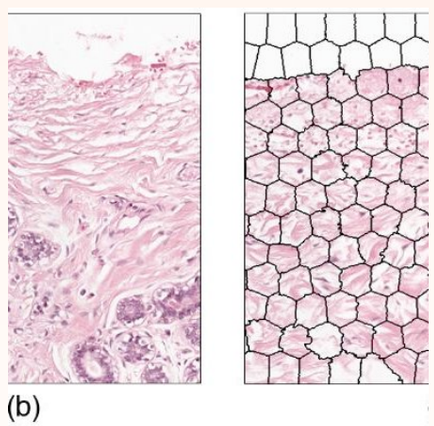


Shape detection
detection of straight lines in images.

Source: https://mlwhiz.com/tags/object-detection/

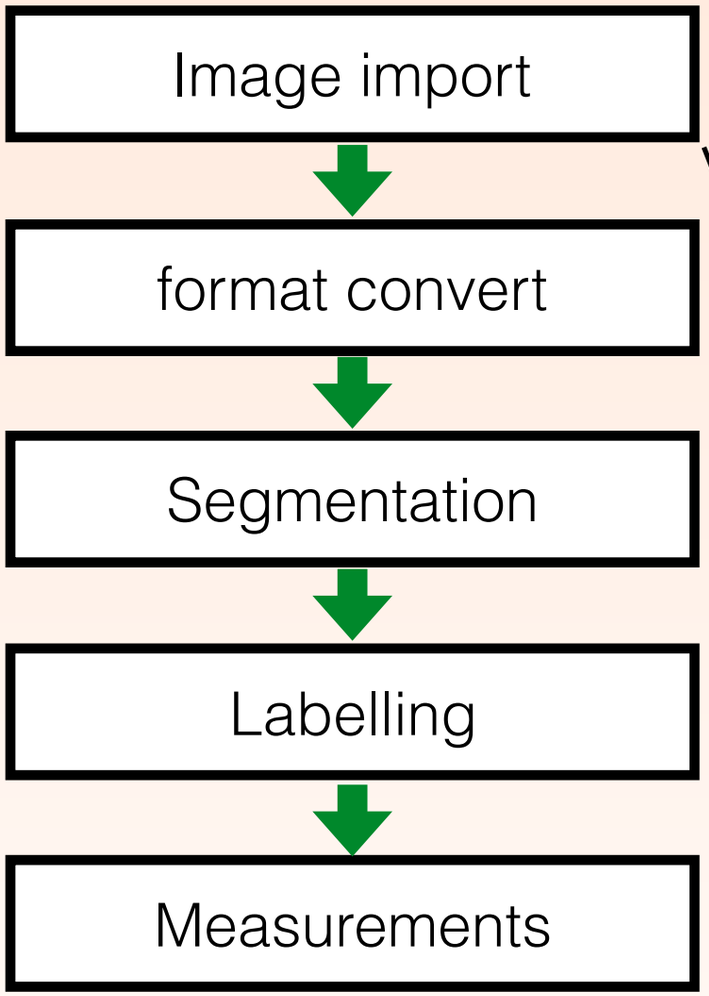


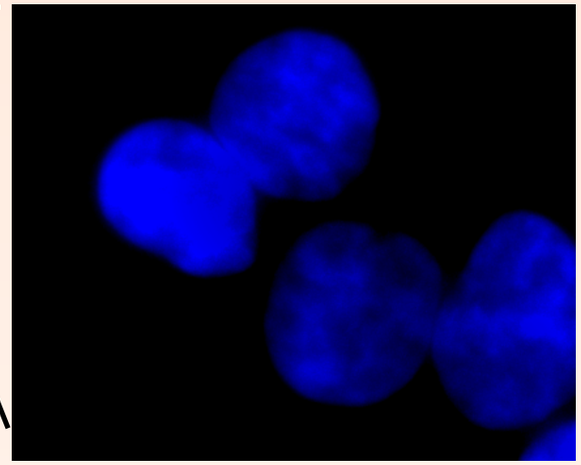
object detection
bounding boxes for instances
potentially more than one class



Region detection
detection of regions of similarity

Universal aspects of a segmentation pipeline

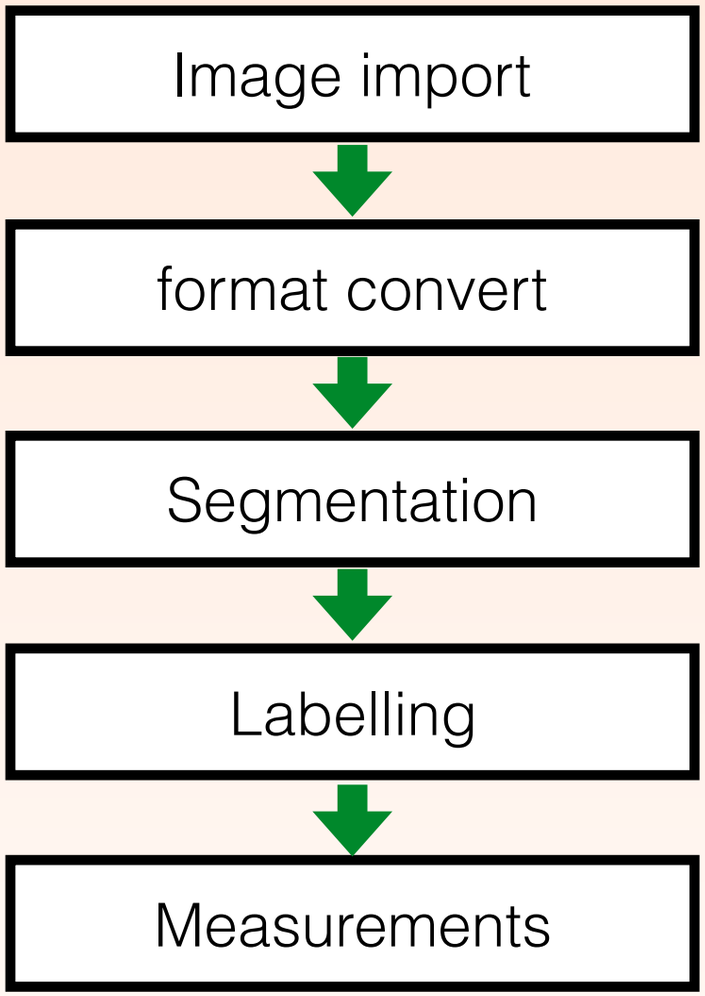


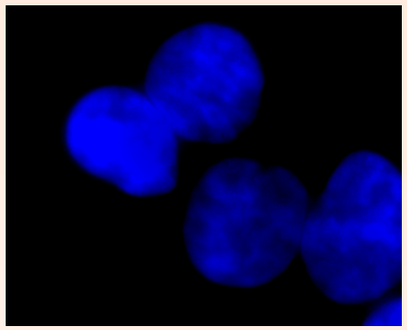


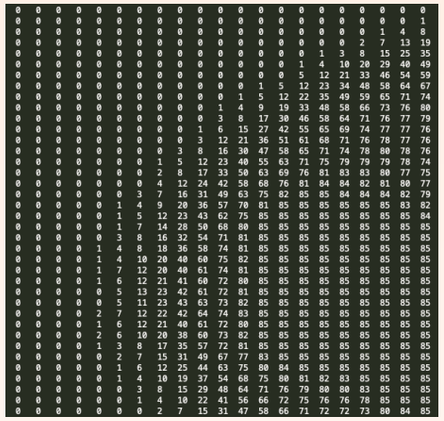
Importing the image into the Python environment so we can access the underlying data. You have seen this in previous talks.

Source:

Universal aspects of a segmentation pipeline







convert to floating point format
e.g 32-bit or 64-bit.

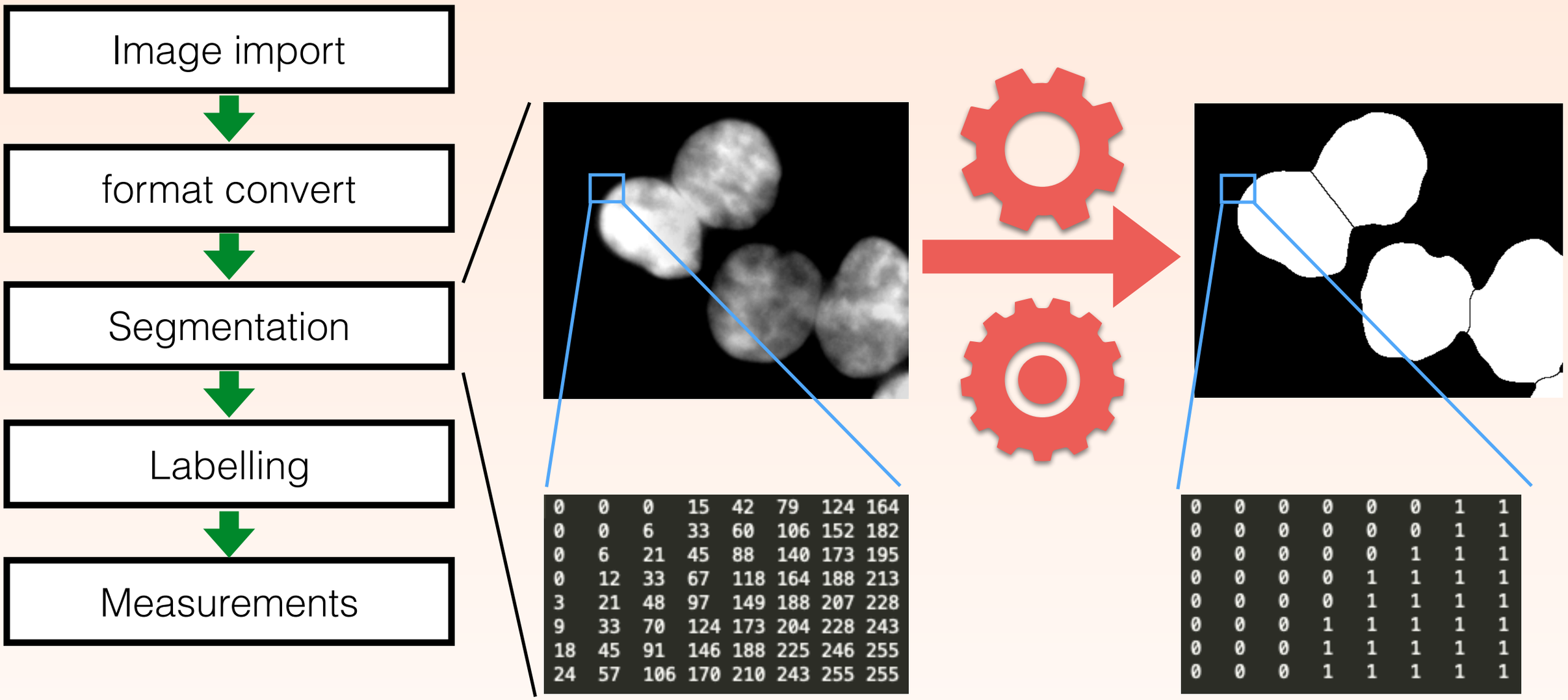
3,4,5 —> 3.0,4.0,5.0

e.g. img = img.astype(np.float64)

We may also need to index the imported data to be able to isolate the channels. We usually need to convert the data to ensure that it is floating point.

Source:

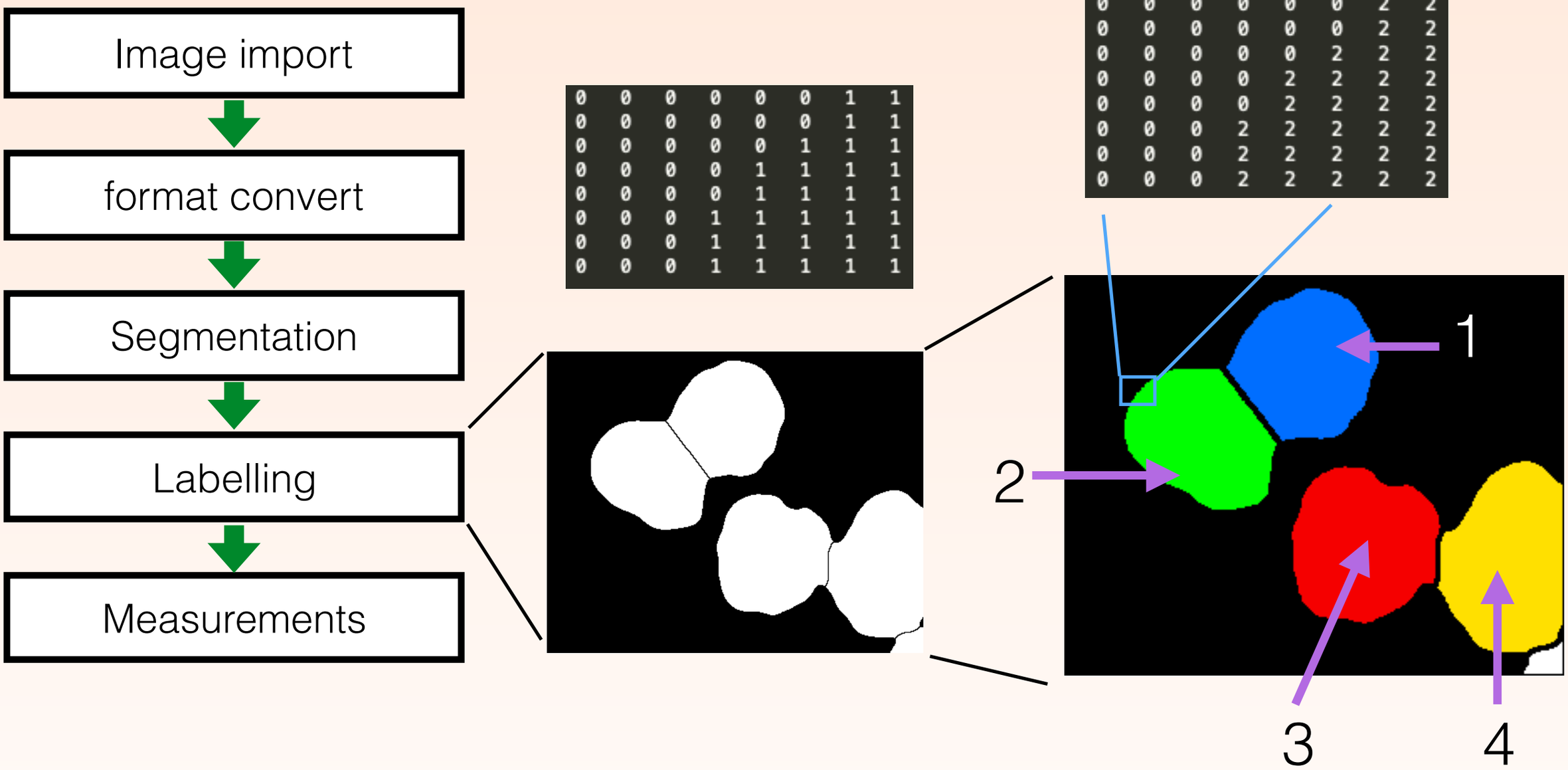
Universal aspects of a segmentation pipeline



There are many strategies for segmentation. We consider various methods later in this talk. What we want is a binary representation (1 and 0s) with each cell or structure separated.

Source:

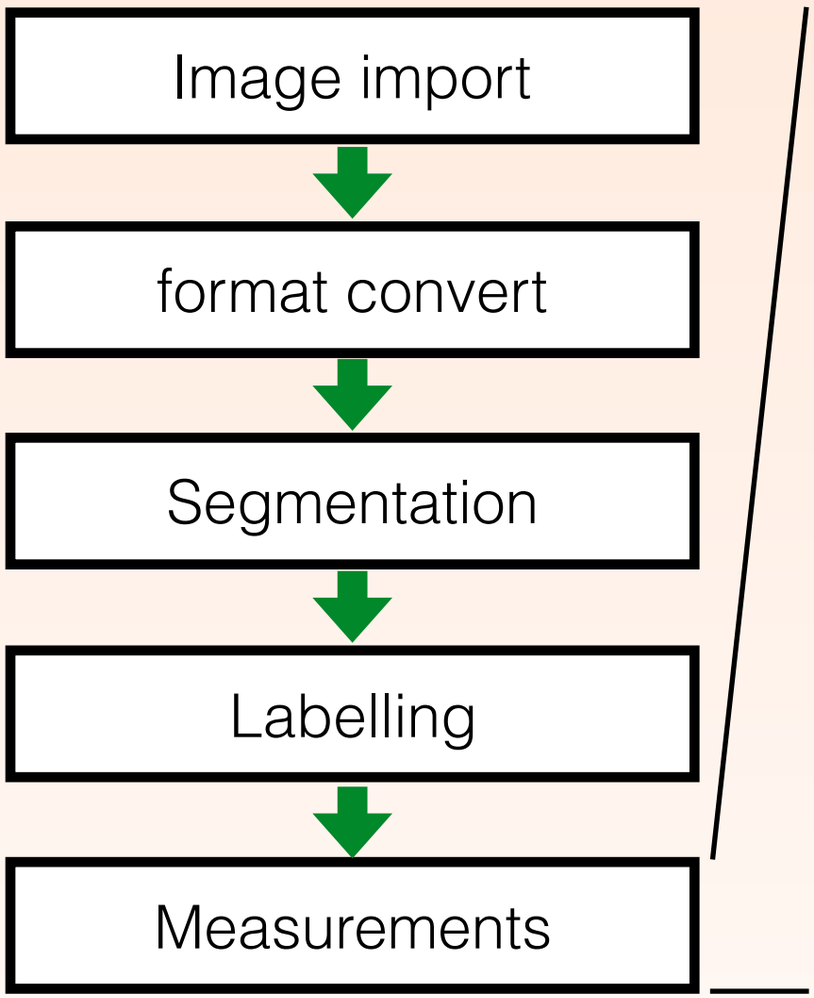
Universal aspects of a segmentation pipeline

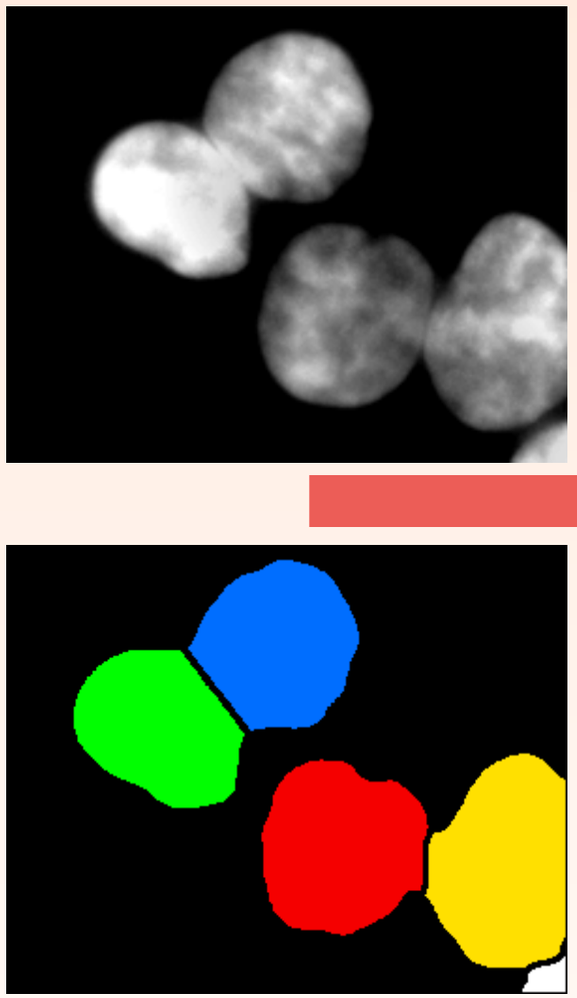


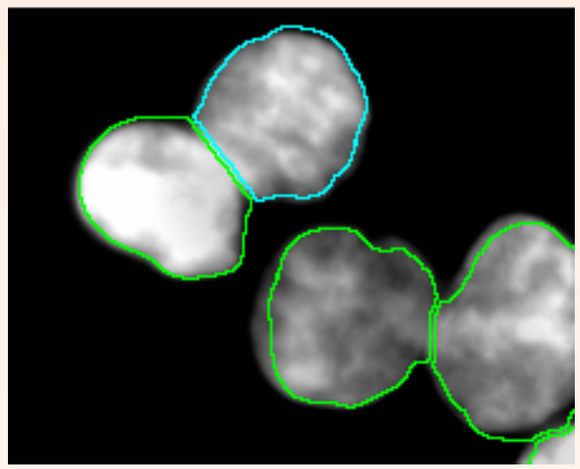
This is how we isolate each of the individual cells. We give the pixels in each cell and unique number. We can then search for and isolate these pixels.

Source:

Universal aspects of a segmentation pipeline





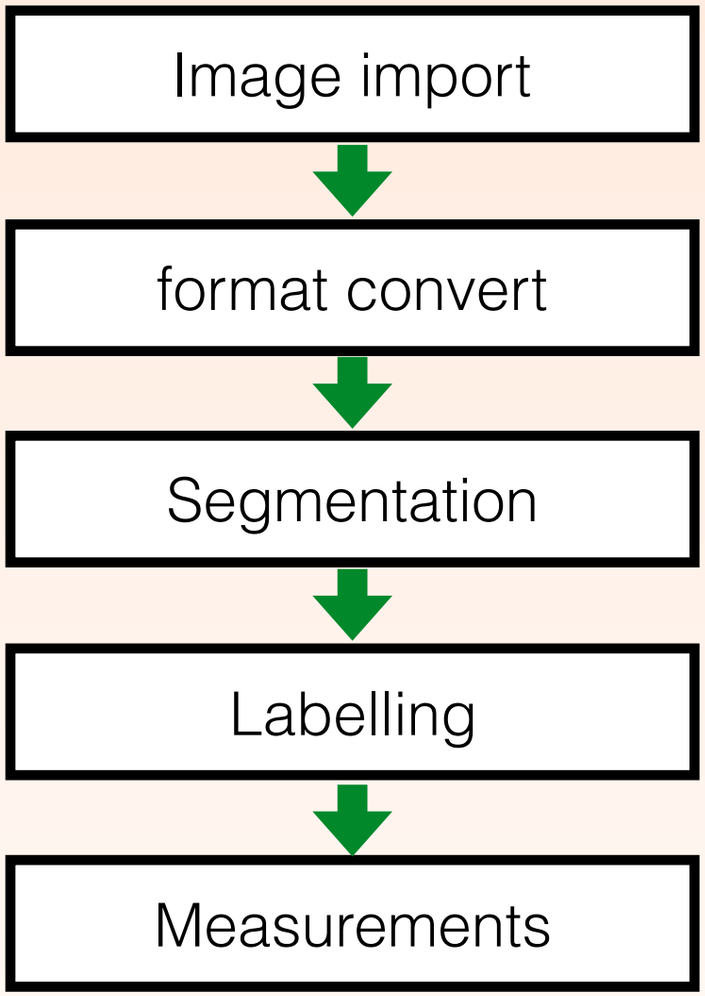


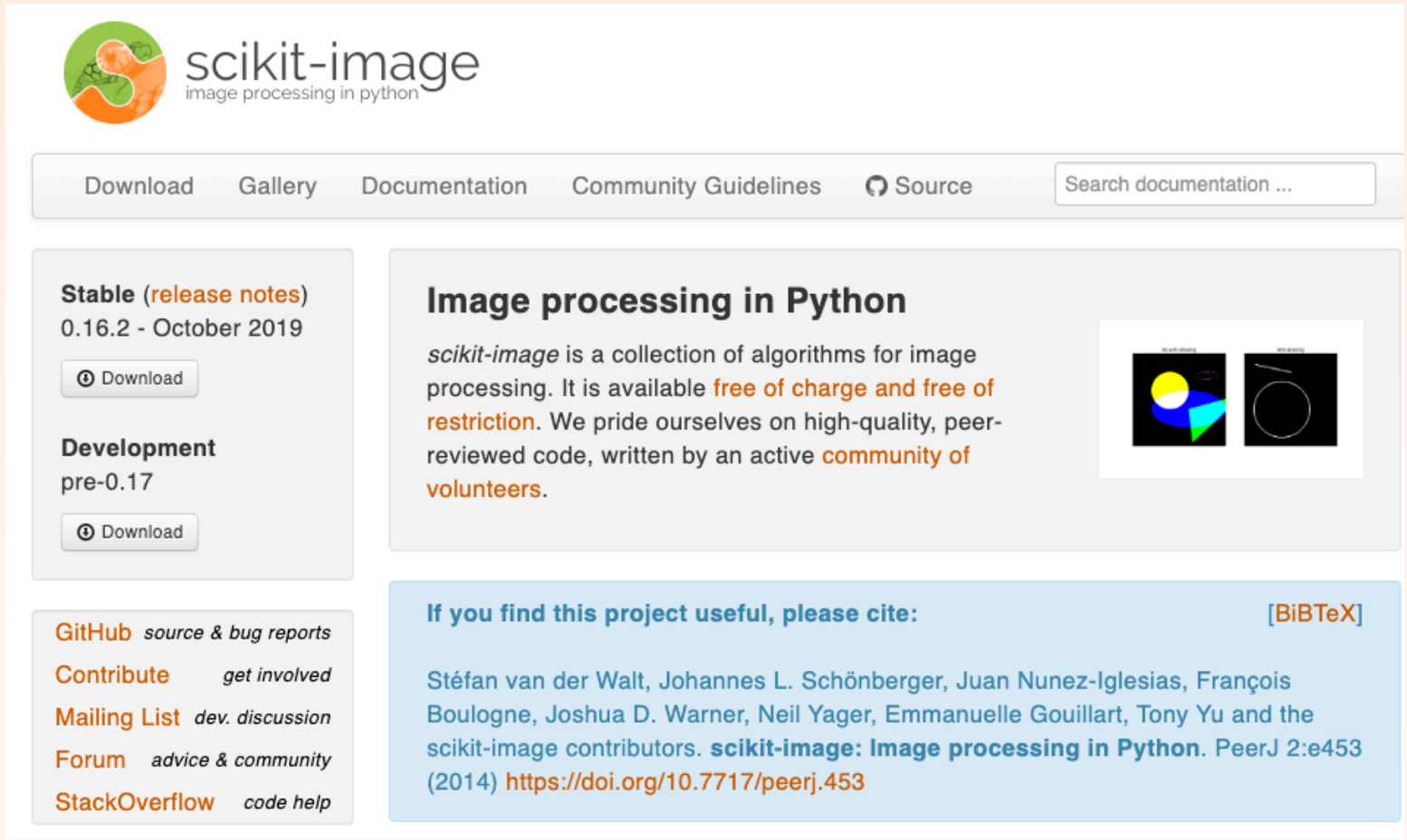
Cell 0: 232.3 33 0.1
Cell 1: 343.3 22 0.2
Cell 2: 233.0 13 0.6
Cell 3: 123.0 32 0.5
Cell 4: 323.0 78 0.2
....
....
Cell N: M1 M2 M3

Once we have our instances we can then make measurements on the individual cells.

Source:

Universal aspects of a segmentation pipeline





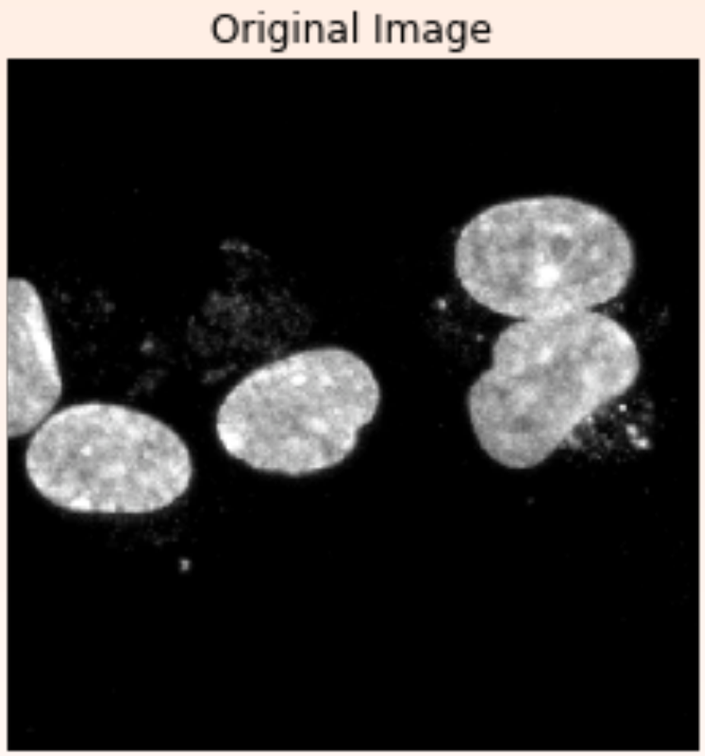
Scikit-image is a library for image processing in Python. It has many powerful modules for segmentation, labelling and measurements. In the practical you will see how to use some of the key functionality.

Source:https://scikit-image.org/

Simple numerical segmentation

Image Thresholding

Image thresholding involves selecting an intensity value which separates the image pixels into a foreground set and a background set of pixels.

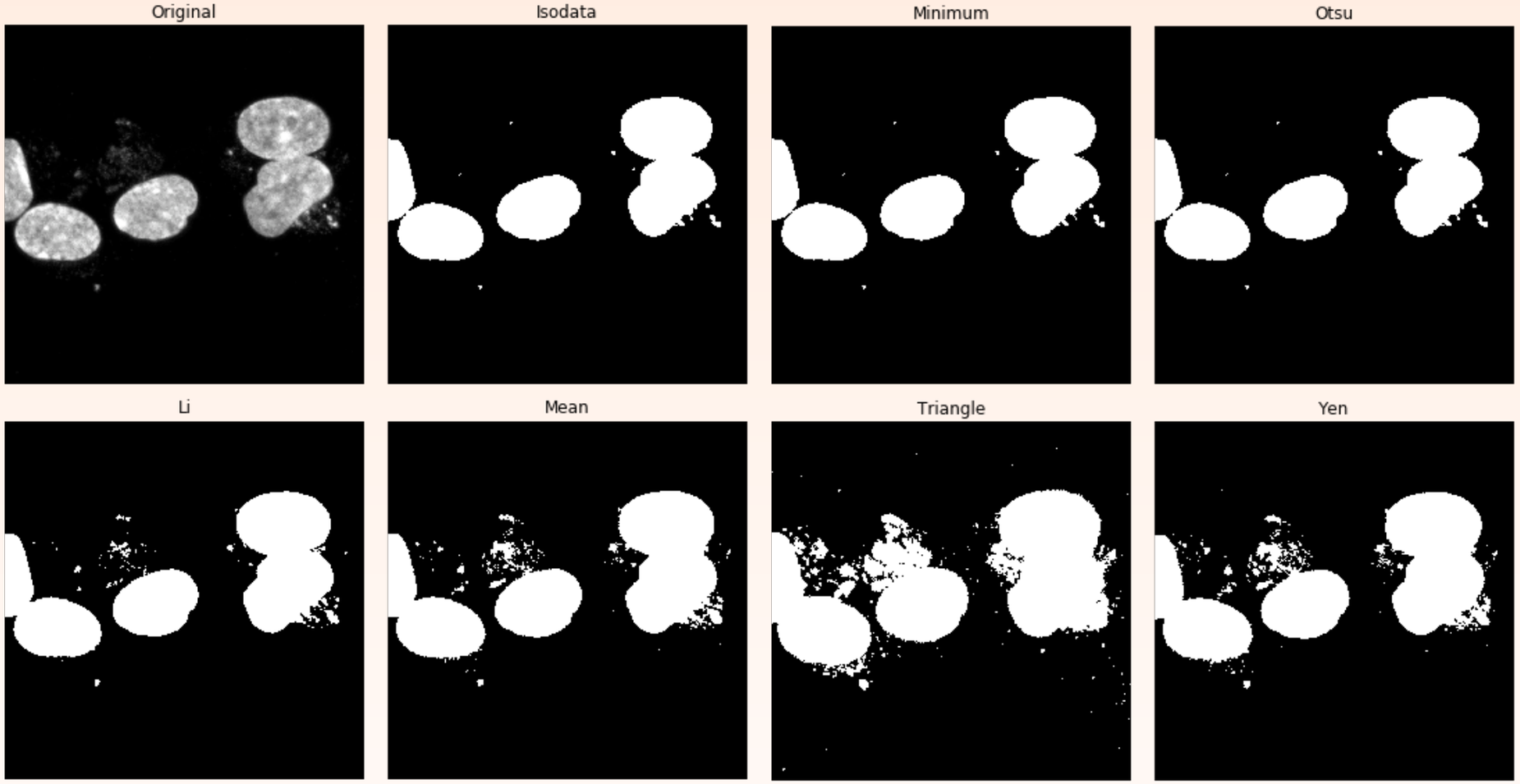


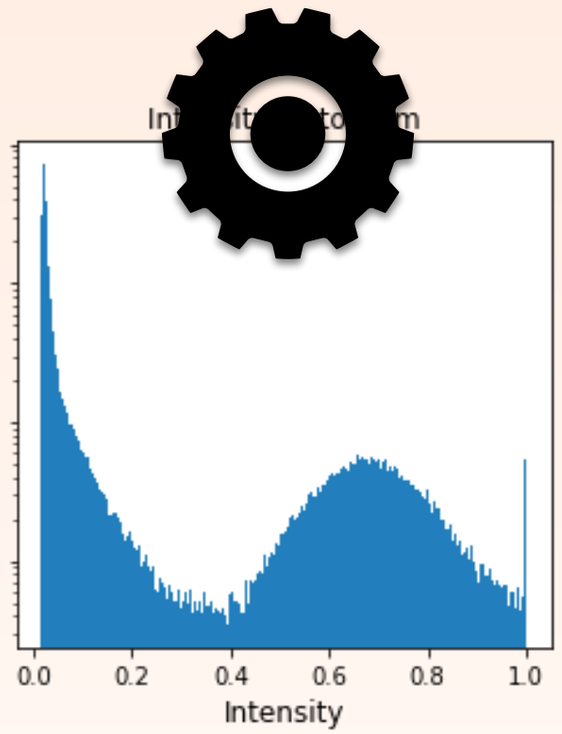
If a manually selected value can be used, this is great, but invariably the variation in the experiment means we need to do this more intelligently.

Source:

Image Thresholding

Image thresholding involves selecting an intensity value which separates the image pixels into a foreground set and a background set.

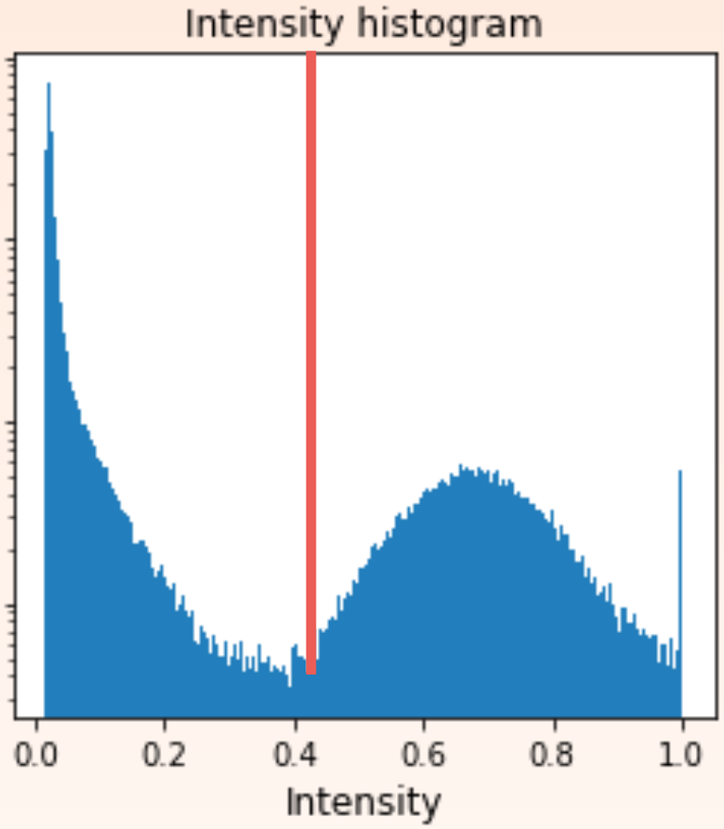




If a manually selected value can be used this is great, but invariably the variation in the experiment means we need to do this more intelligently.

Source:

How the famous Otsu method works.



For each possible threshold value (e.g. 0-255) we calculate:

$$\text{Within Class Variance} \quad \sigma_W^2 = W_b\,\sigma_b^2 + W_f\,\sigma_f^2$$

Where σ_b is the variance in background intensities.
Where σ_f is the variance in foreground intensities. W_f is the weight of foreground (num. of f pixels over total) and W_b is the weight of background pixels.

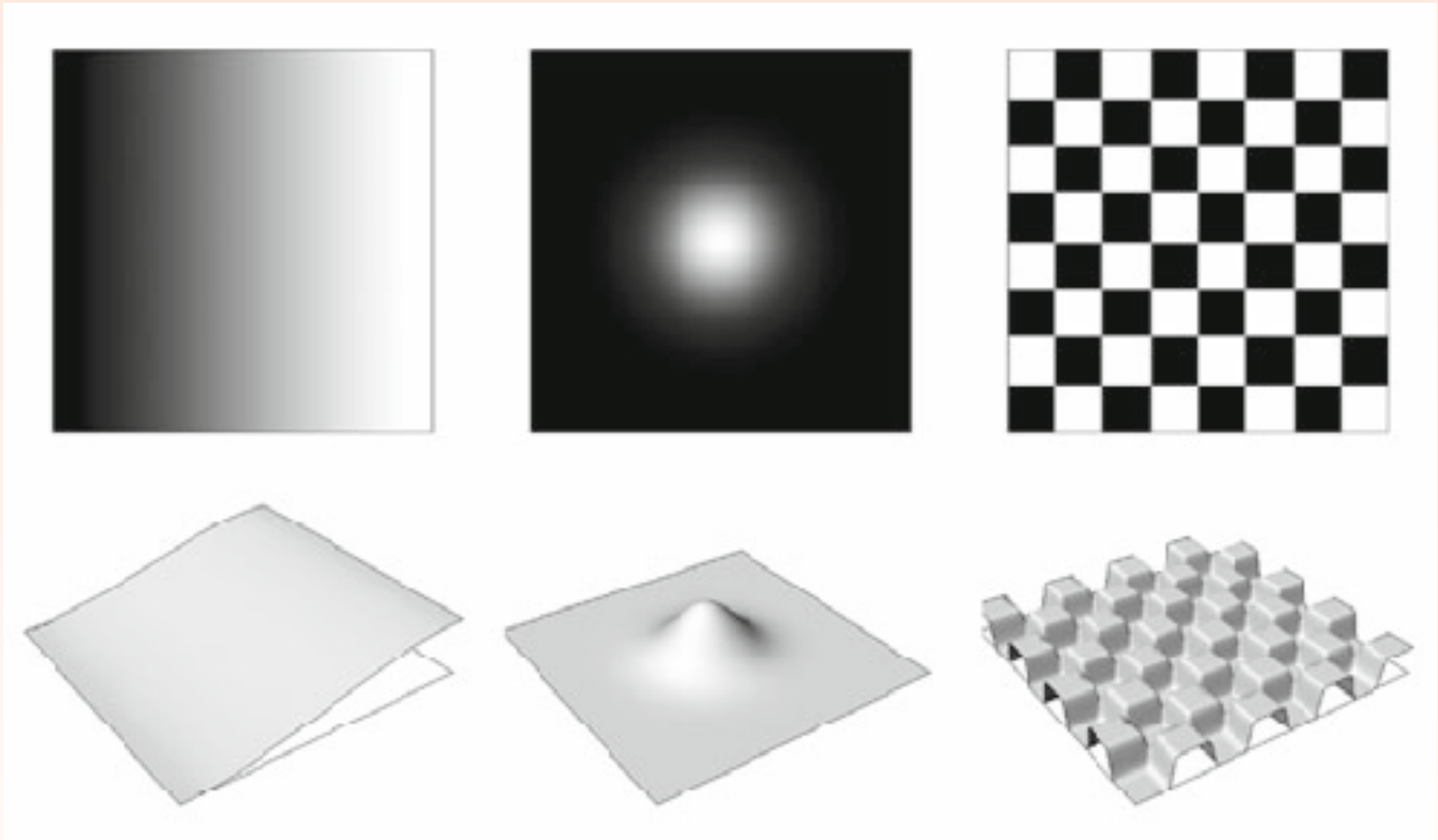
We retain the threshold which gives the smallest σ^2 across all the thresholds.

Otsu's method, named after its inventor Nobuyuki Otsu is a famous example, but one of many. Best approach is try and see how it goes.

Source: http://www.labbookpages.co.uk/software/imgProc/otsuThreshold.html

Thinking topographically about the image.

Change how you think about an image.

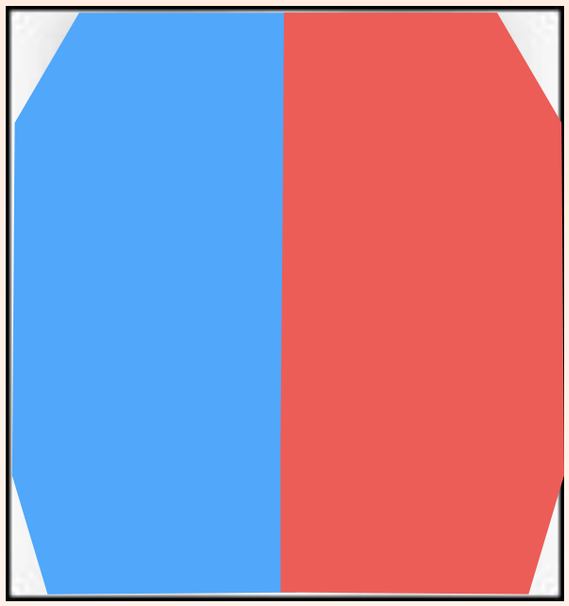


It can be useful to not just think of your image as pixels, but as a 3-D surface, where the bright parts represent peaks.

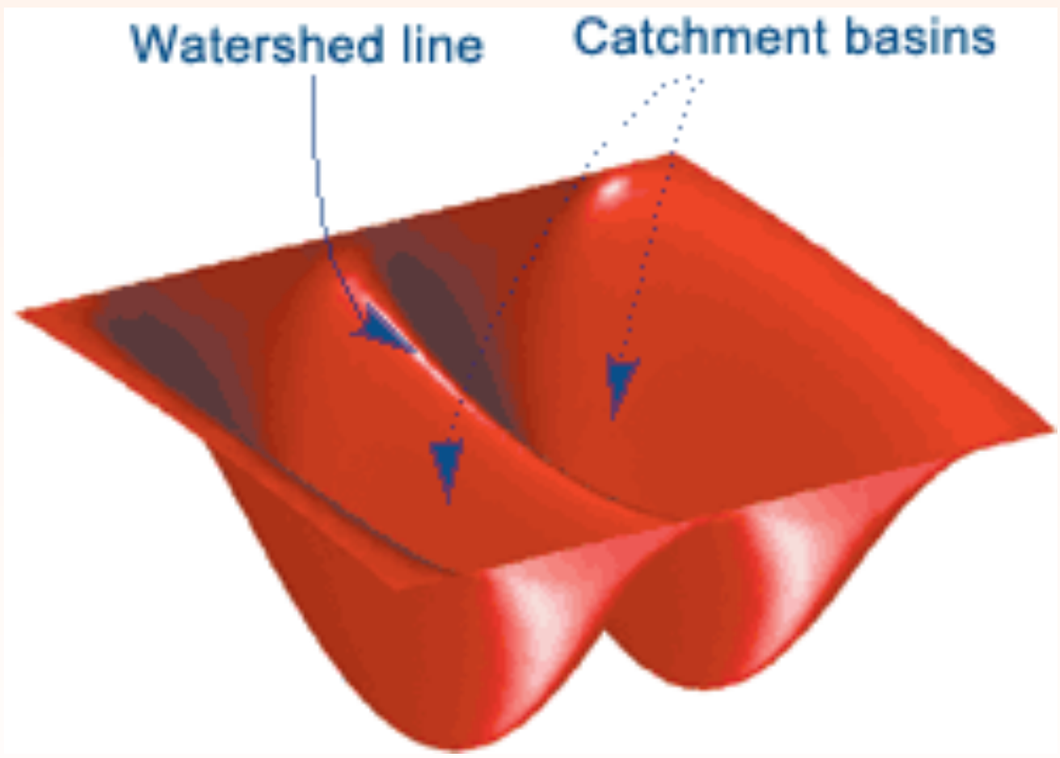
Source: Material Customization: Digital Fabrication Workshop at ISCTE/IUL Jose Pedro Sousa

Watershed algorithm

Works on the premise that we think of our image as being a topological map.



We start in our valleys and slowly raise the water level. When two pools of water meet we draw a boundary.

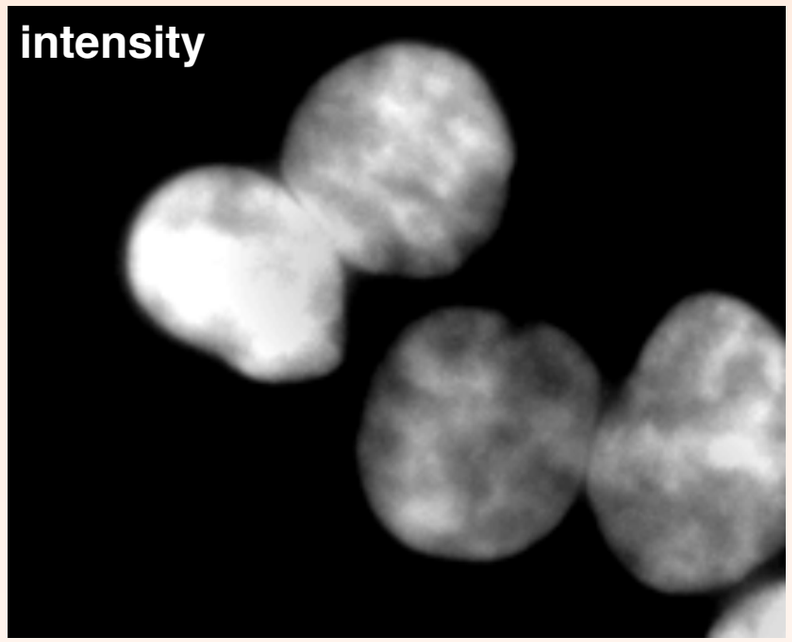


Although it doesn't explicitly require segmentation, one does need to separate foreground from background for it to work nicely.

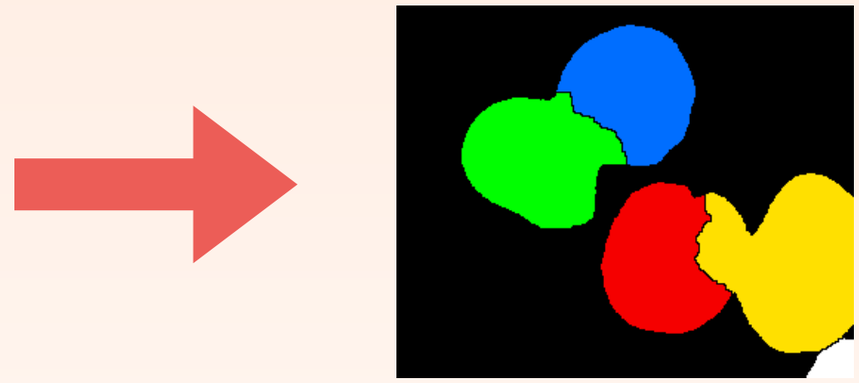
Source:

Watershed algorithm

We could apply watershed to our raw intensity image.

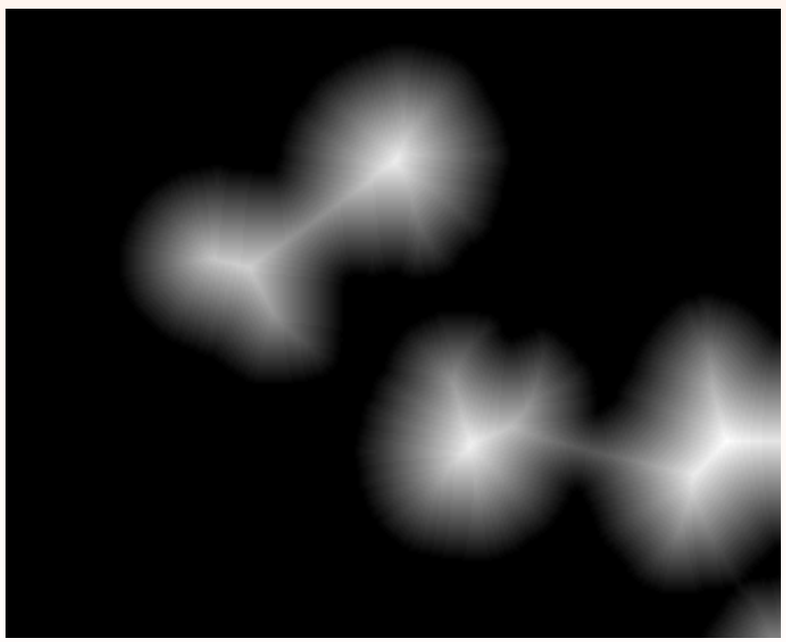


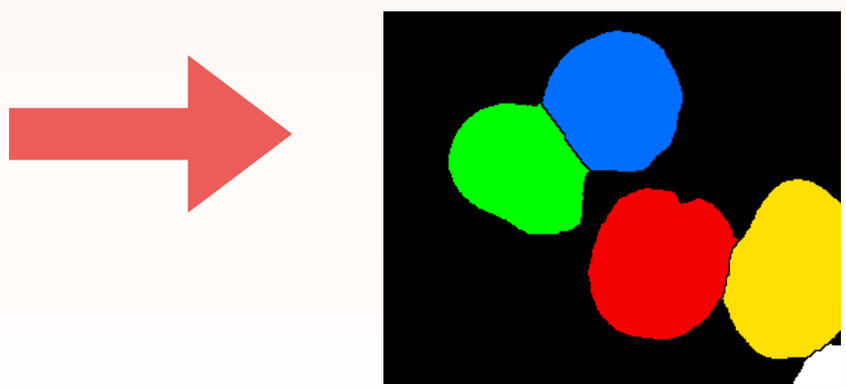
But this wouldn't work very well. The intensity distribution is complex and noisy.



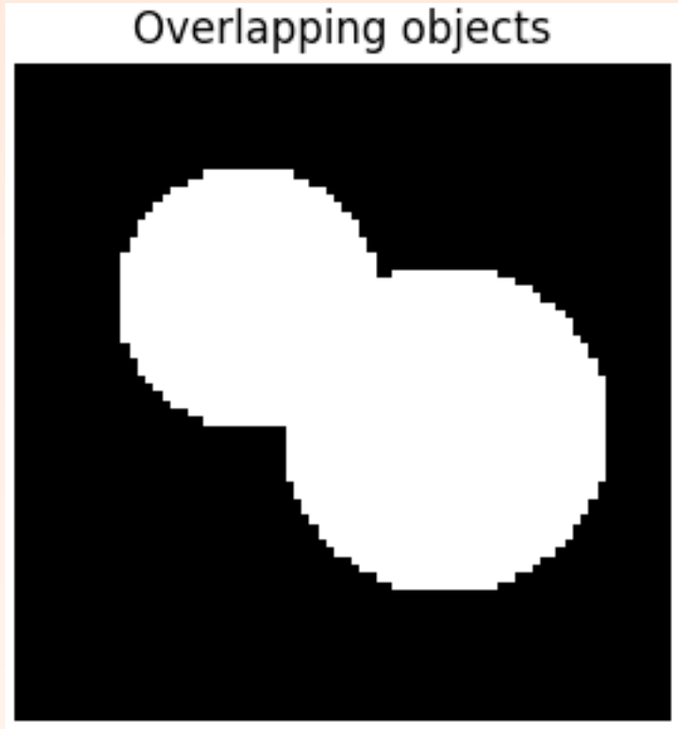
A better method for this kind of data is to substitute a better topological representation.

This can be done with the distance transform. This will separate based on their size and proximity to one another.

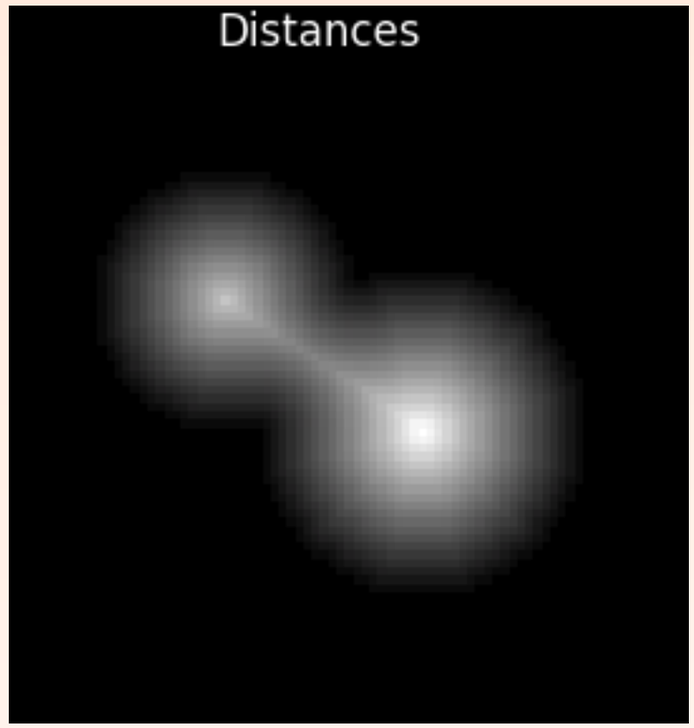




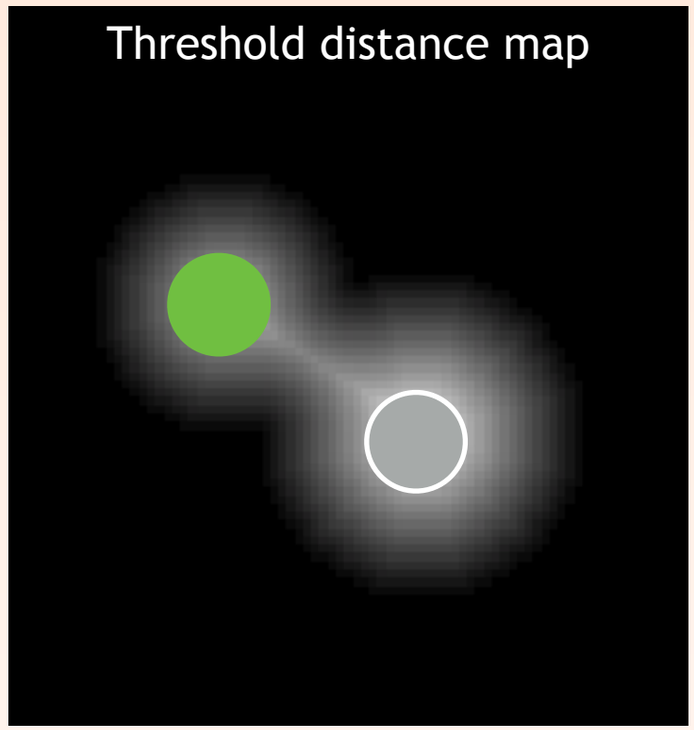
Watershed applied to distance transform.



First we segment our image

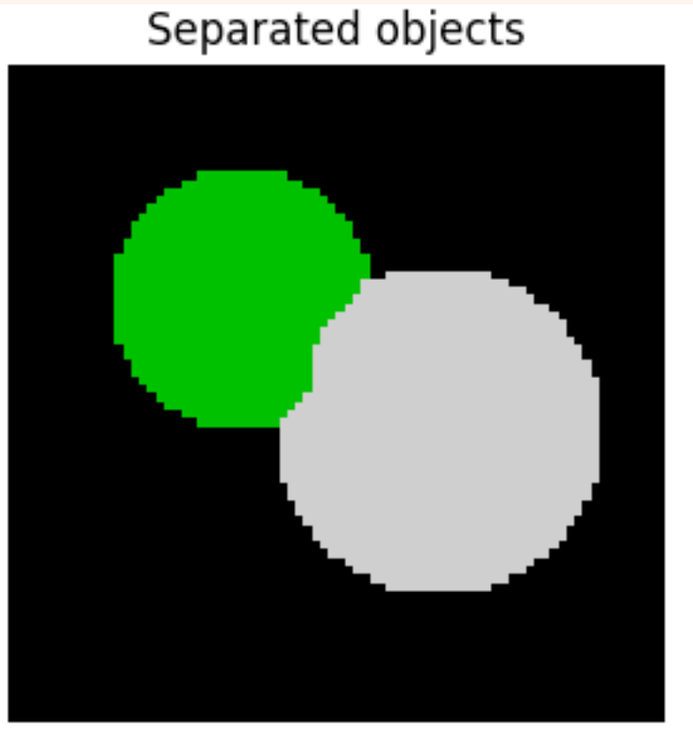


Apply distance transform



Calculate markers

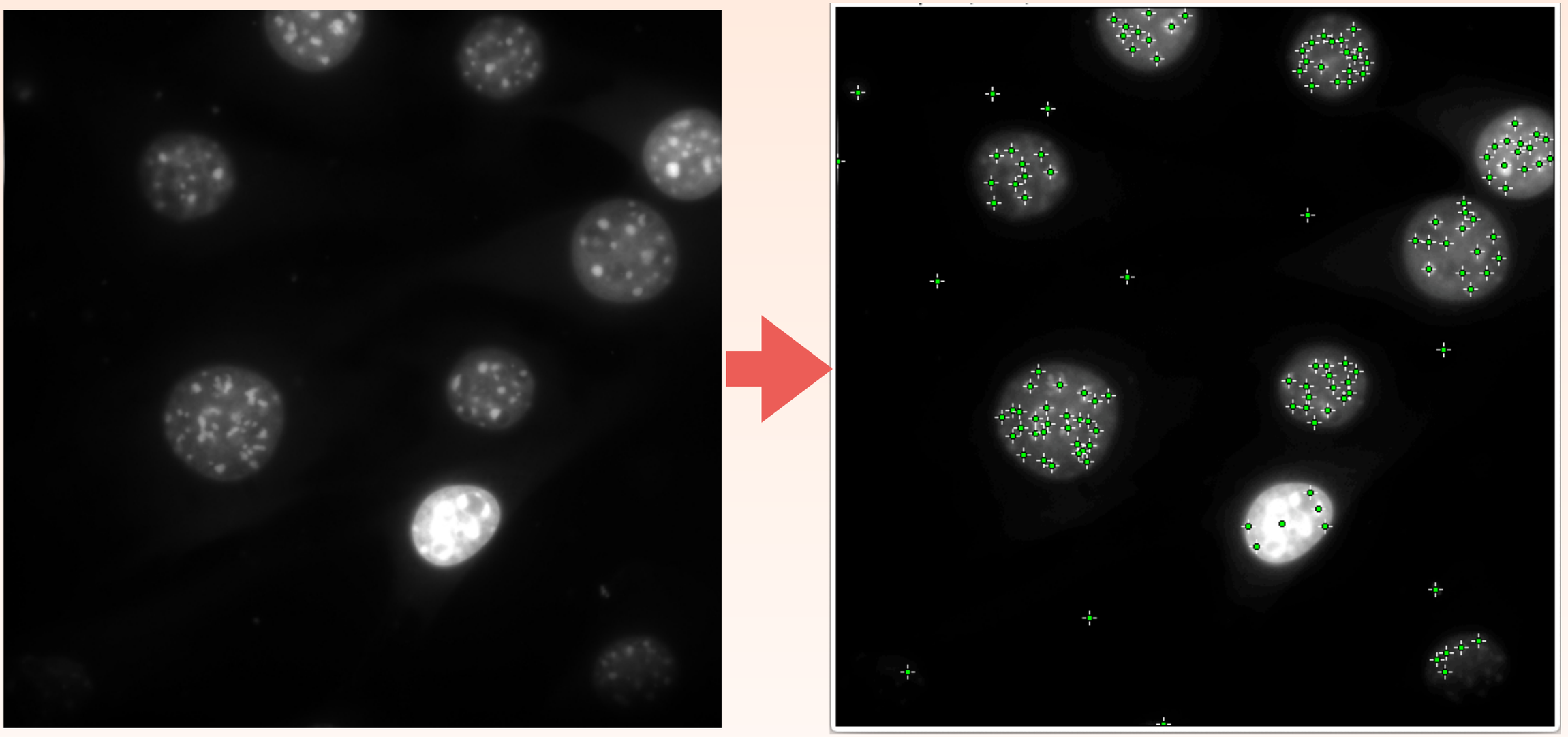
This is more often than not how you will see watershed being applied in pipelines.



Watershed algorithm

Source:

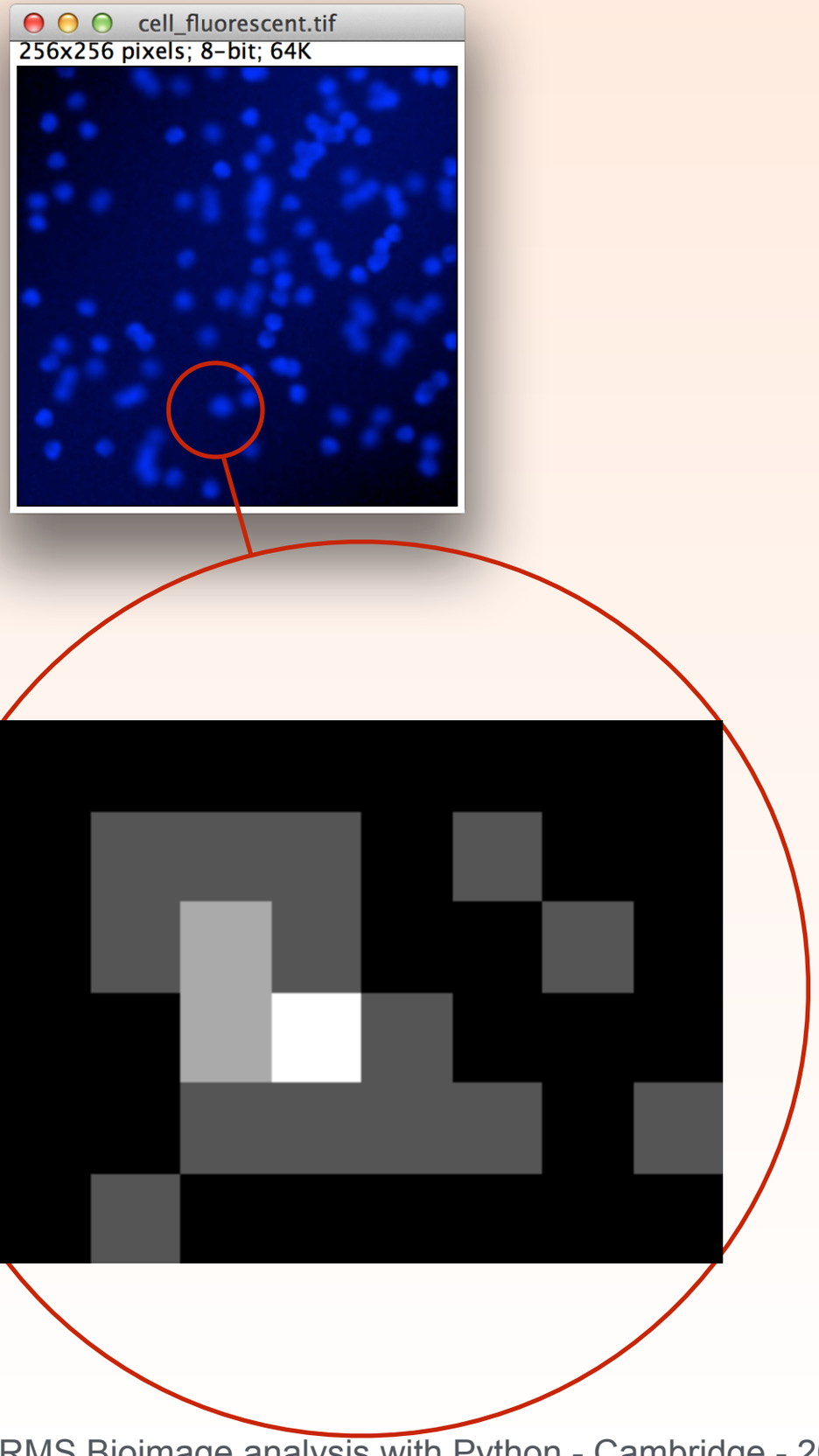
Feature finding: Peak finding algorithm.



We want to detect bright punctuate fluorescence in our images. The tops of intensity peaks.

Source:

Non-linear filter: Maximum filter (not a convolution)



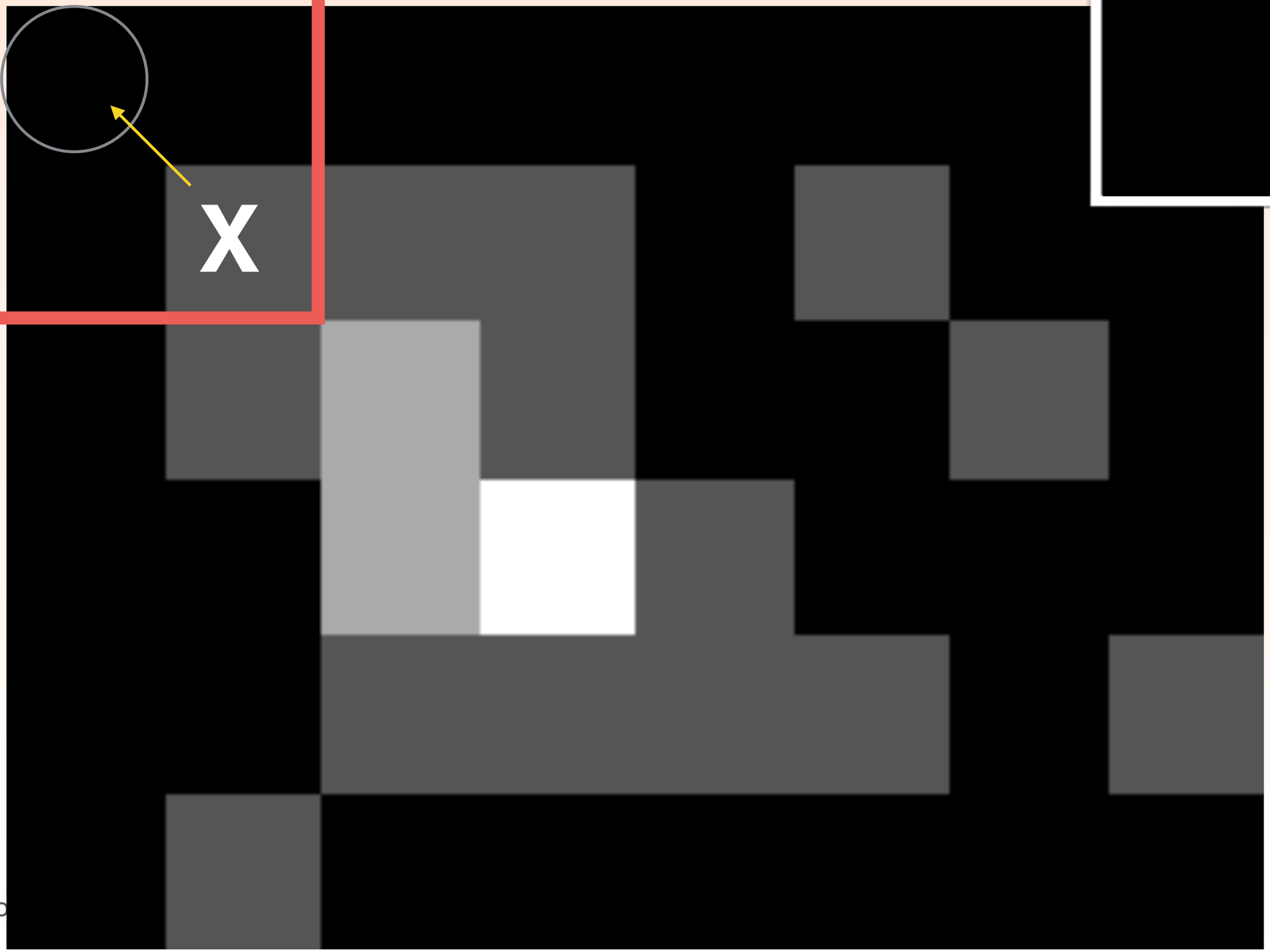
?

kernel can not be expressed as is non-linear

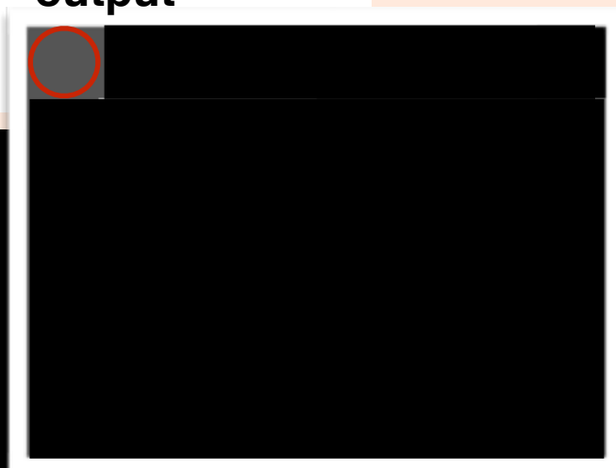
```
[[1, 1, 1, 1, 1, 1, 1, 1],
 [1, 2, 2, 2, 1, 2, 1, 1],
 [1, 2, 3, 2, 1, 1, 2, 1],
 [1, 1, 3, 4, 2, 1, 1, 1],
 [1, 1, 2, 2, 2, 2, 1, 2],
 [1, 2, 1, 1, 1, 1, 1, 1]]
```

Feature finding methods

Non-linear filter: Maximum Filter

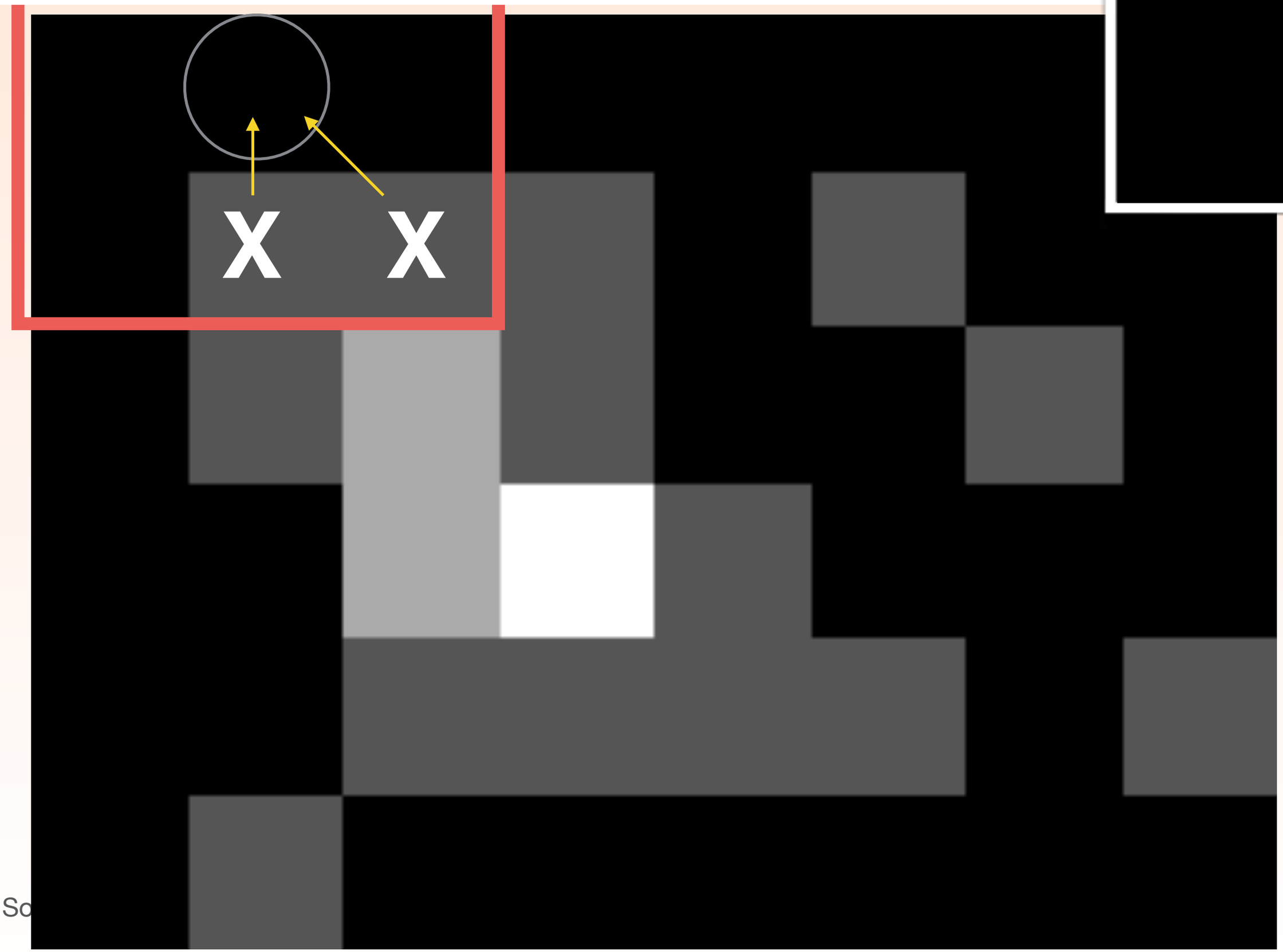


output



So

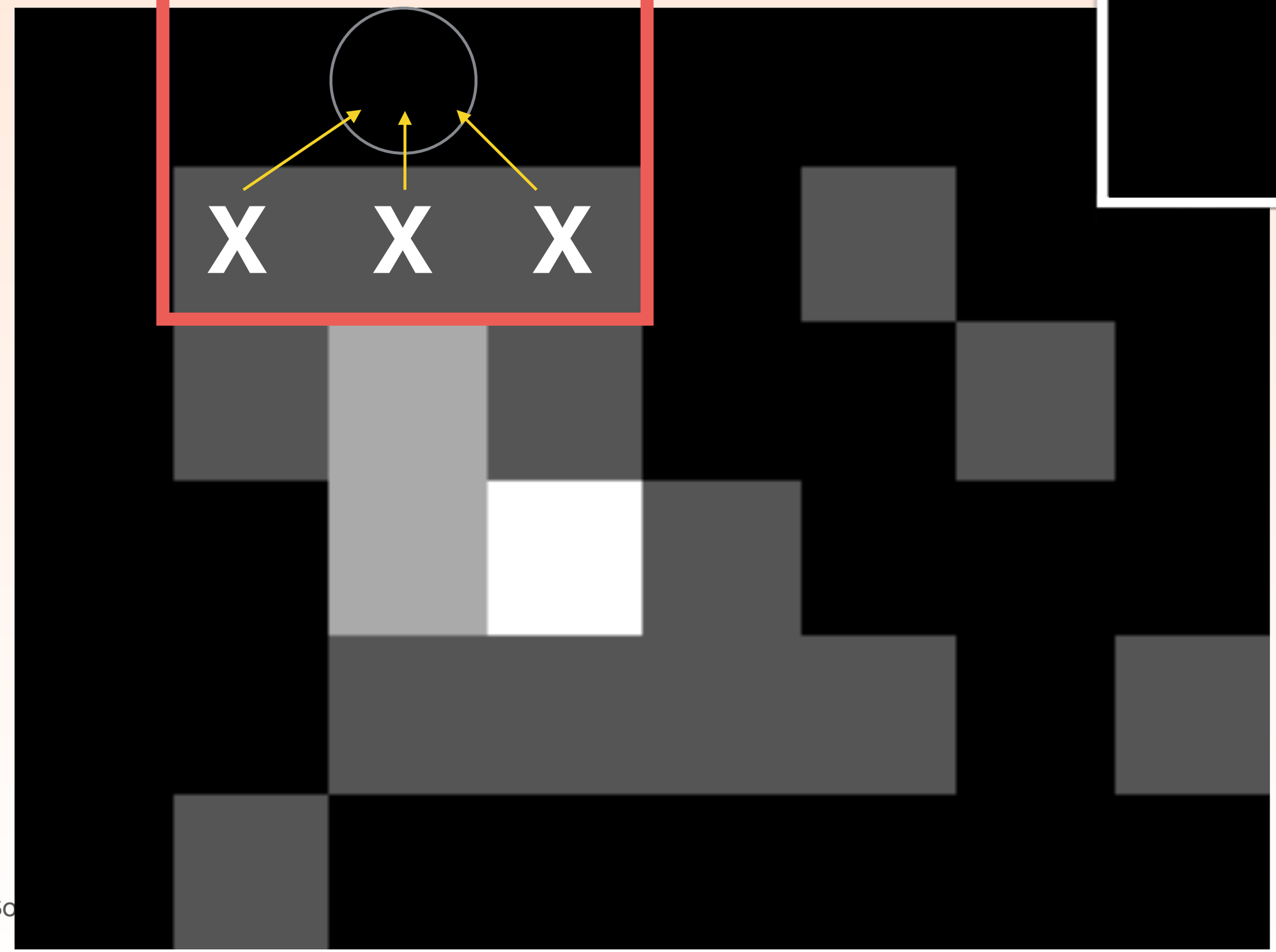
Non-linear filter: Maximum Filter



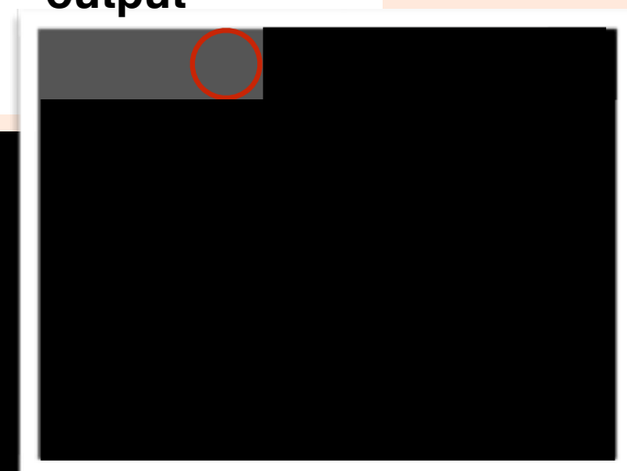
output

So

Non-linear filter: Maximum Filter

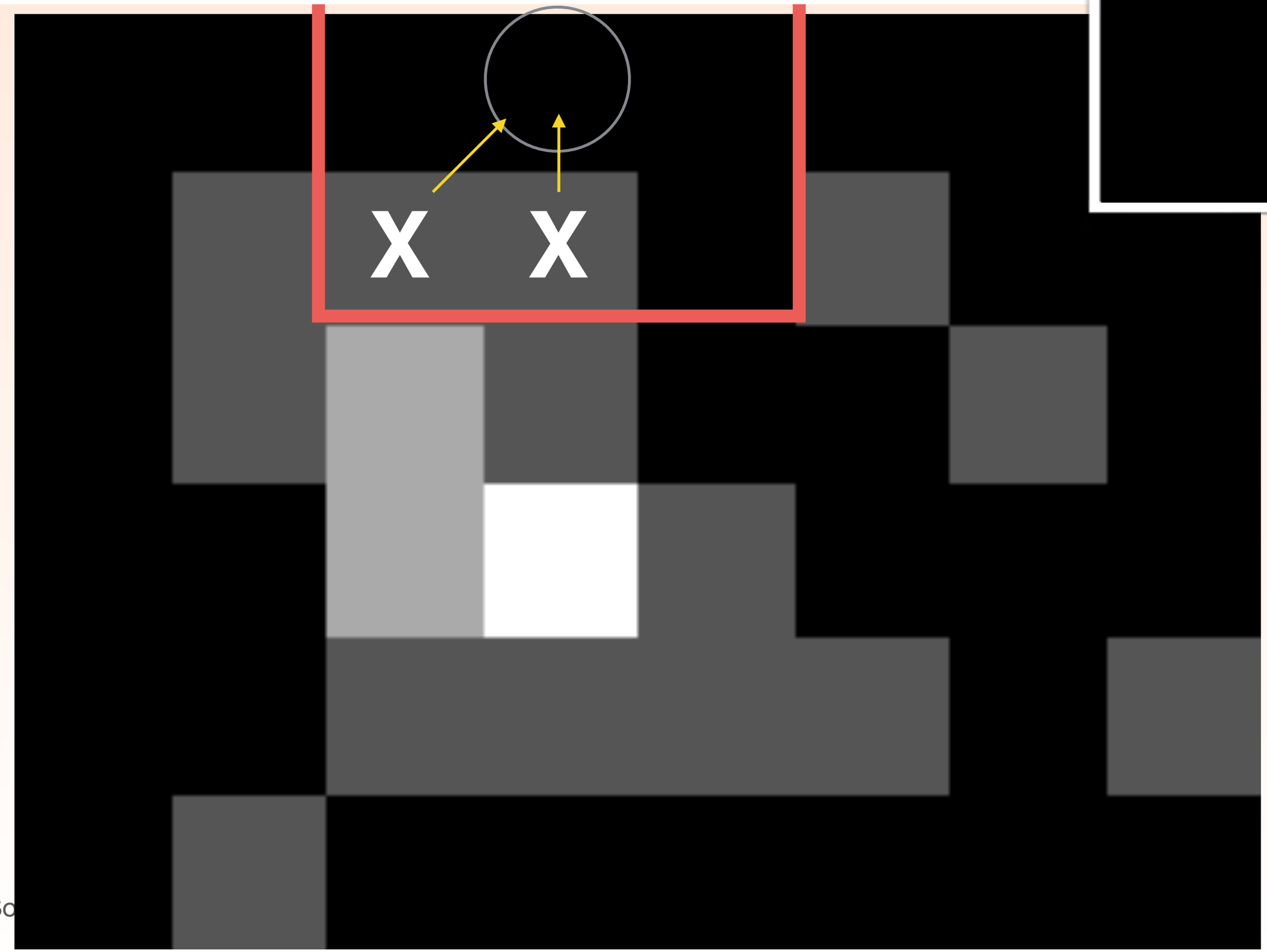


output

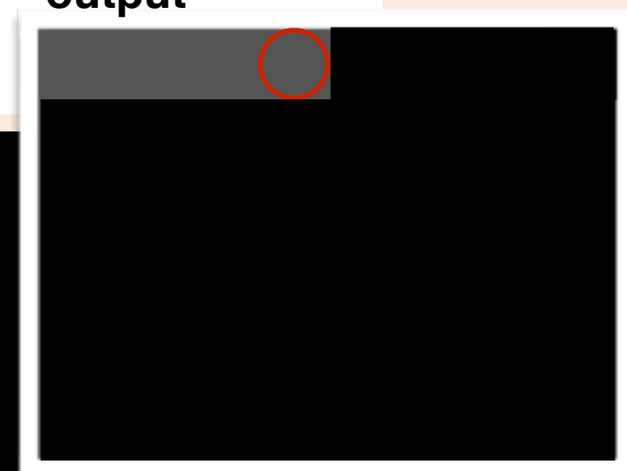


So

Non-linear filter: Maximum Filter

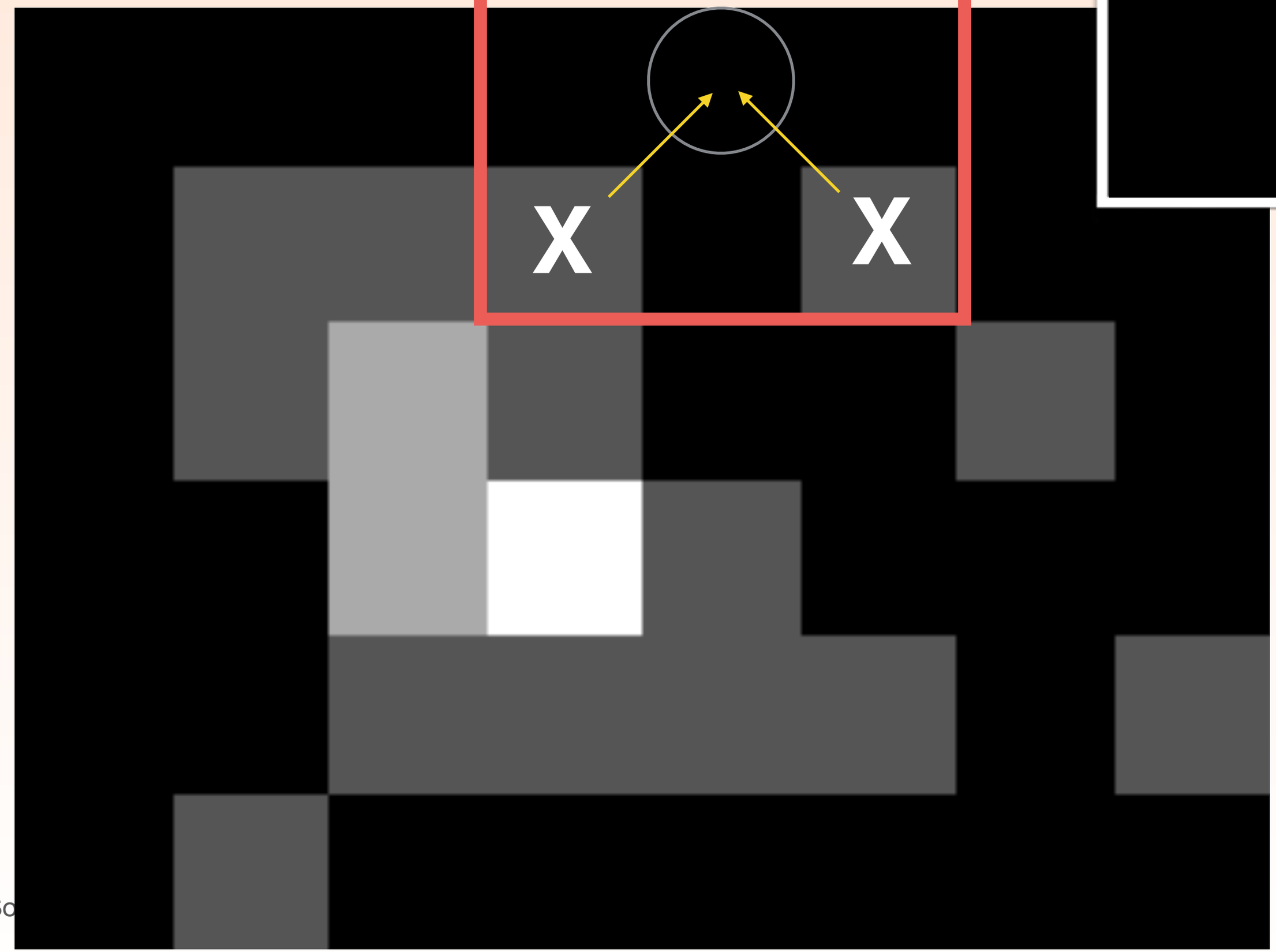


output

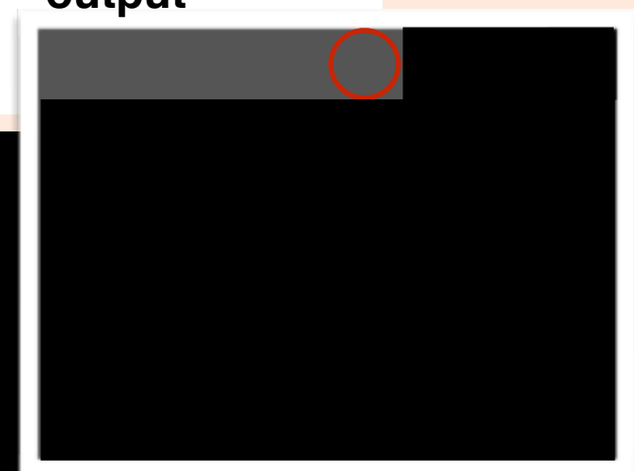


So

Non-linear filter: Maximum Filter

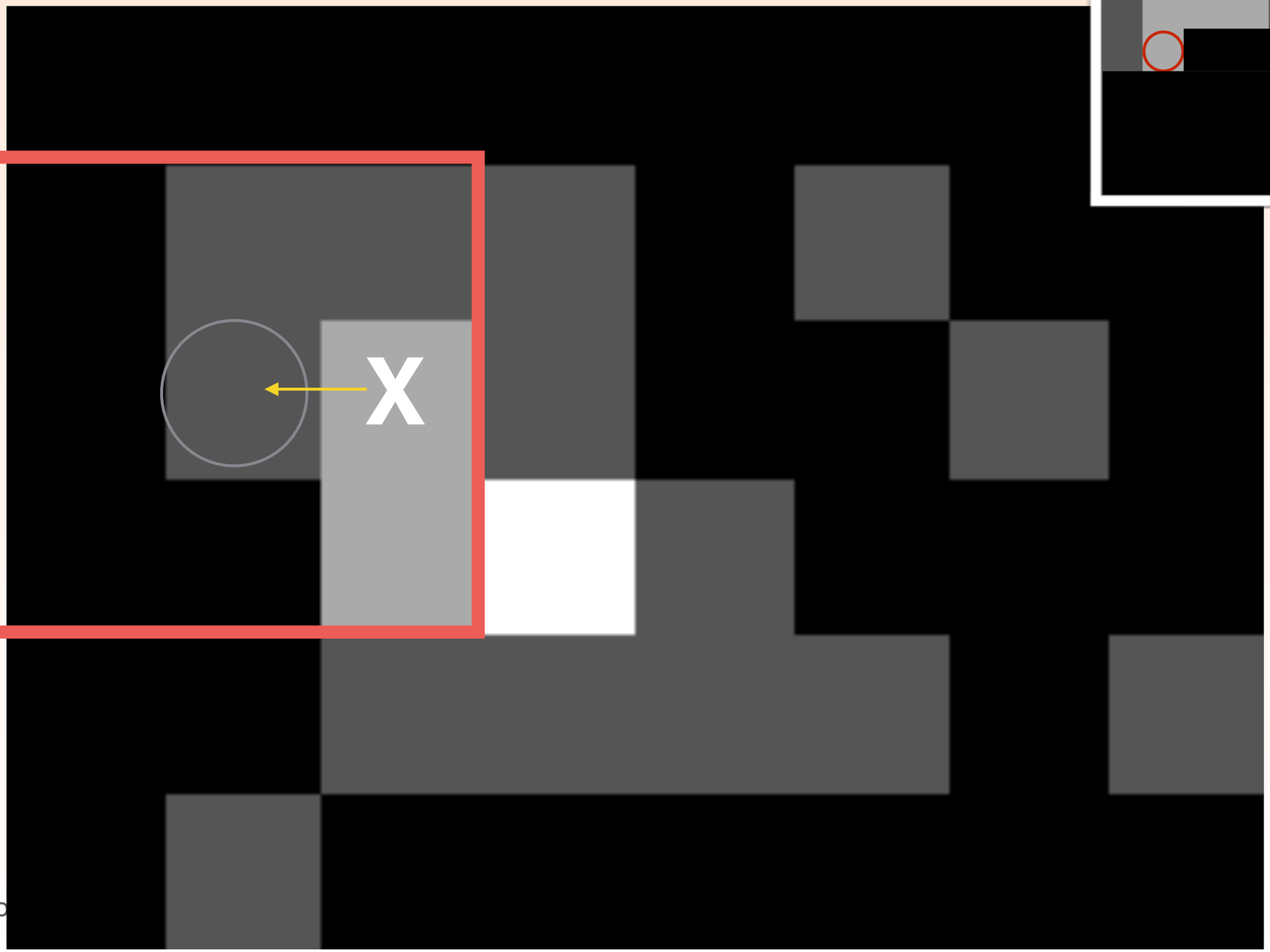


output

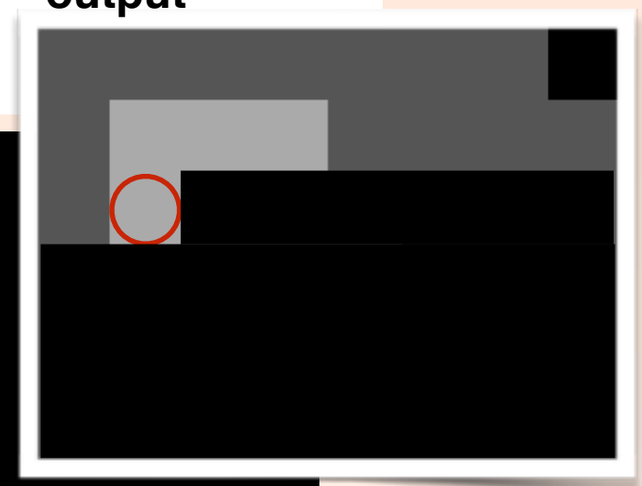


So

Non-linear filter: Maximum Filter

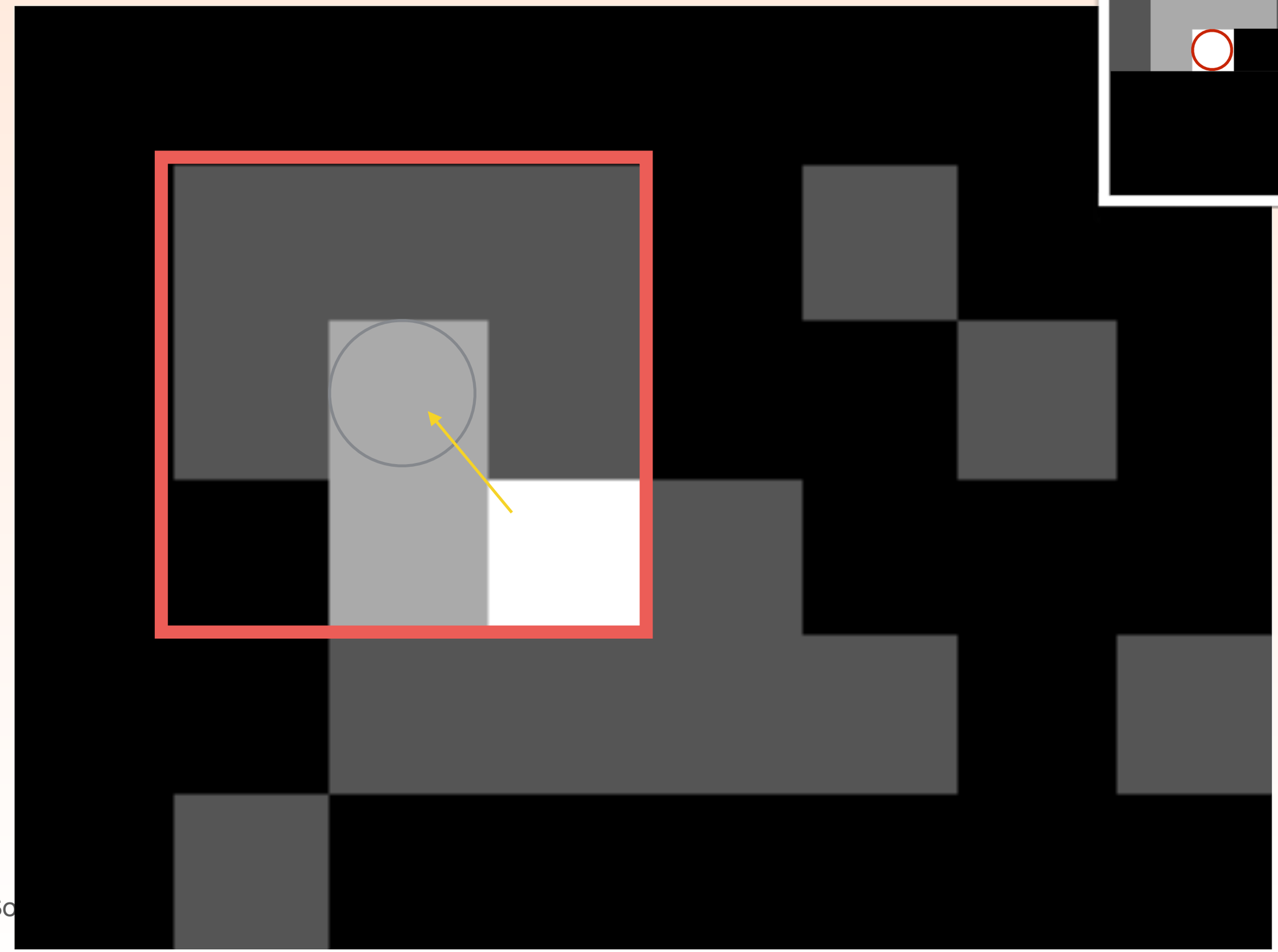


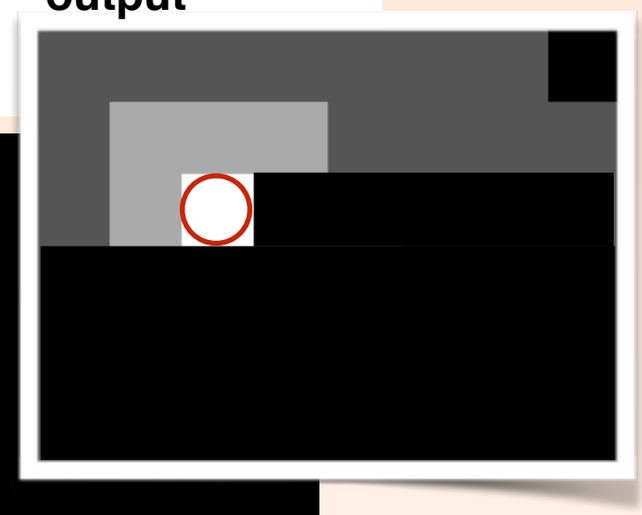
output



So

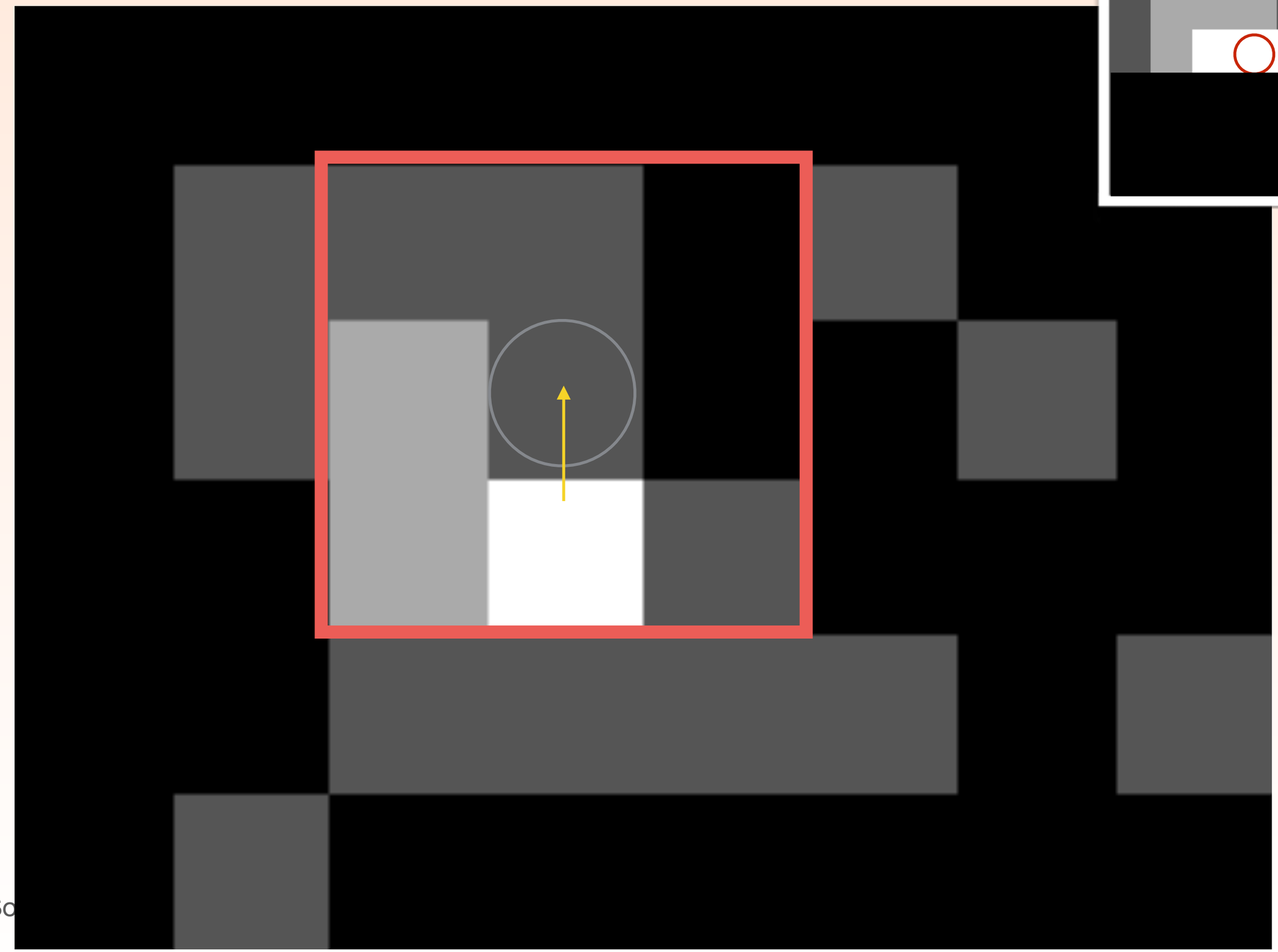
Non-linear filter: Maximum Filter

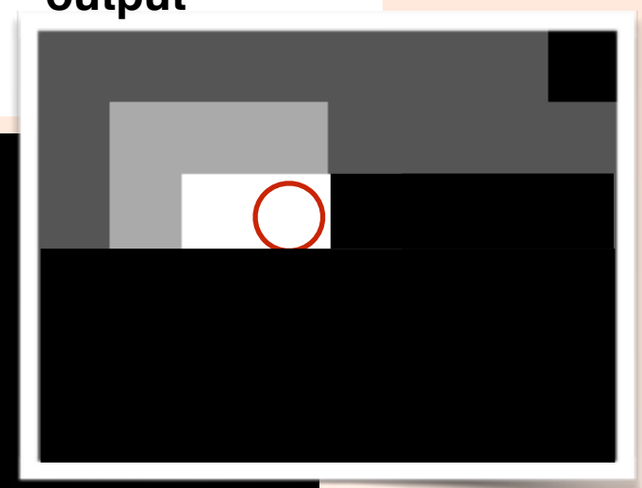




So

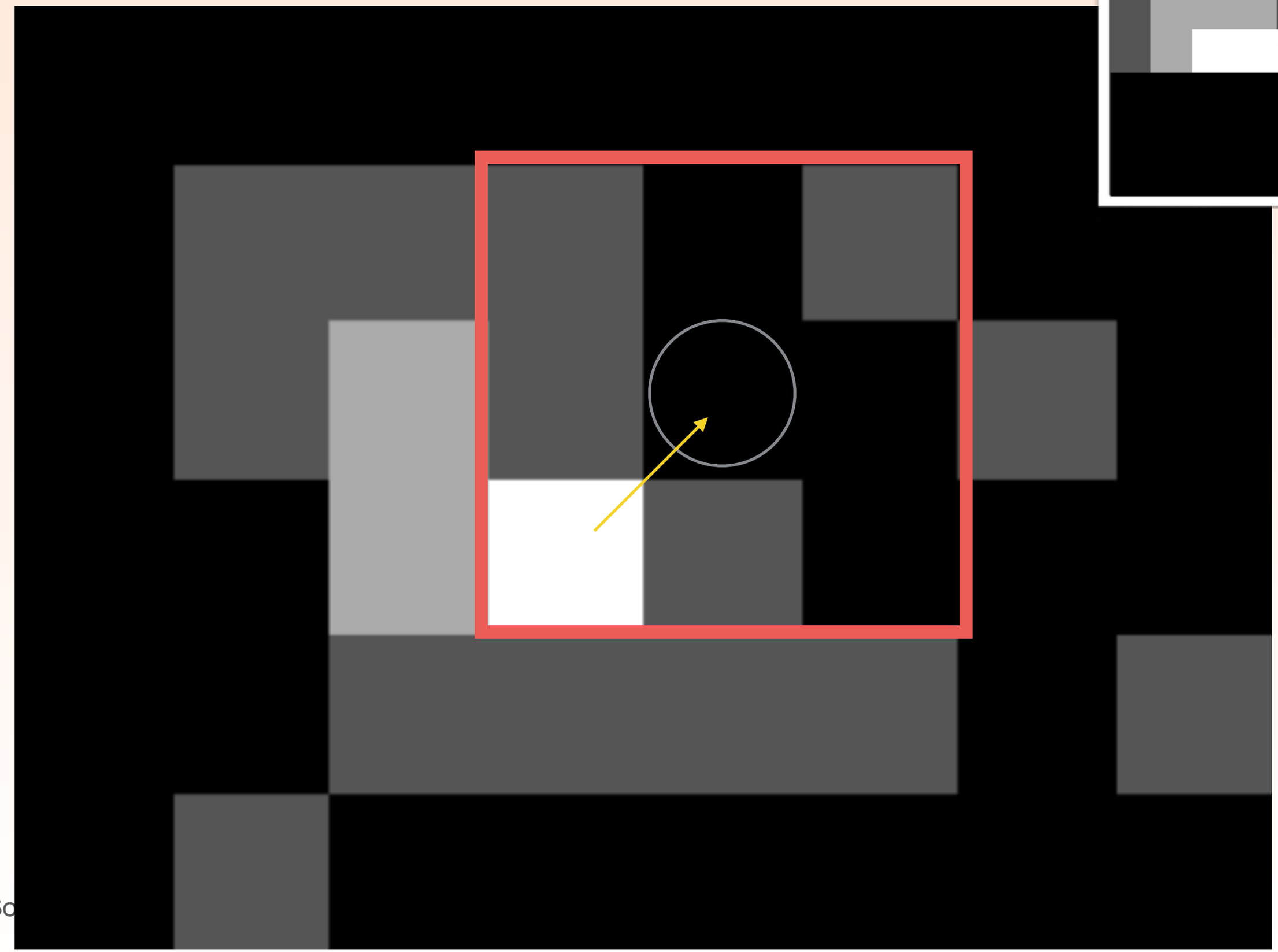
Non-linear filter: Maximum Filter

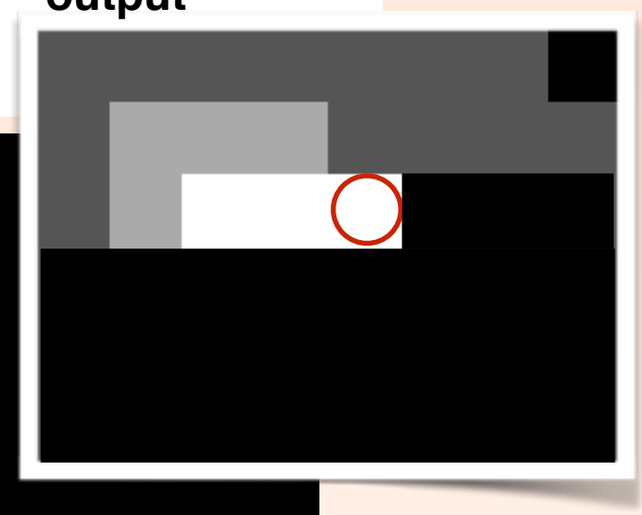




So

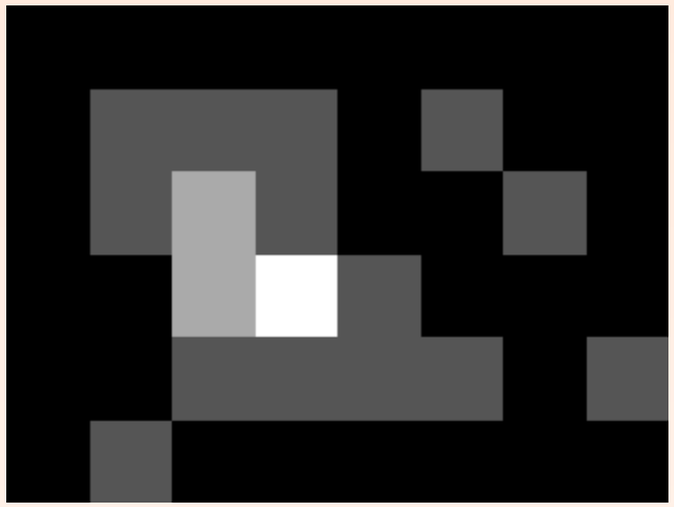
Non-linear filter: Maximum Filter



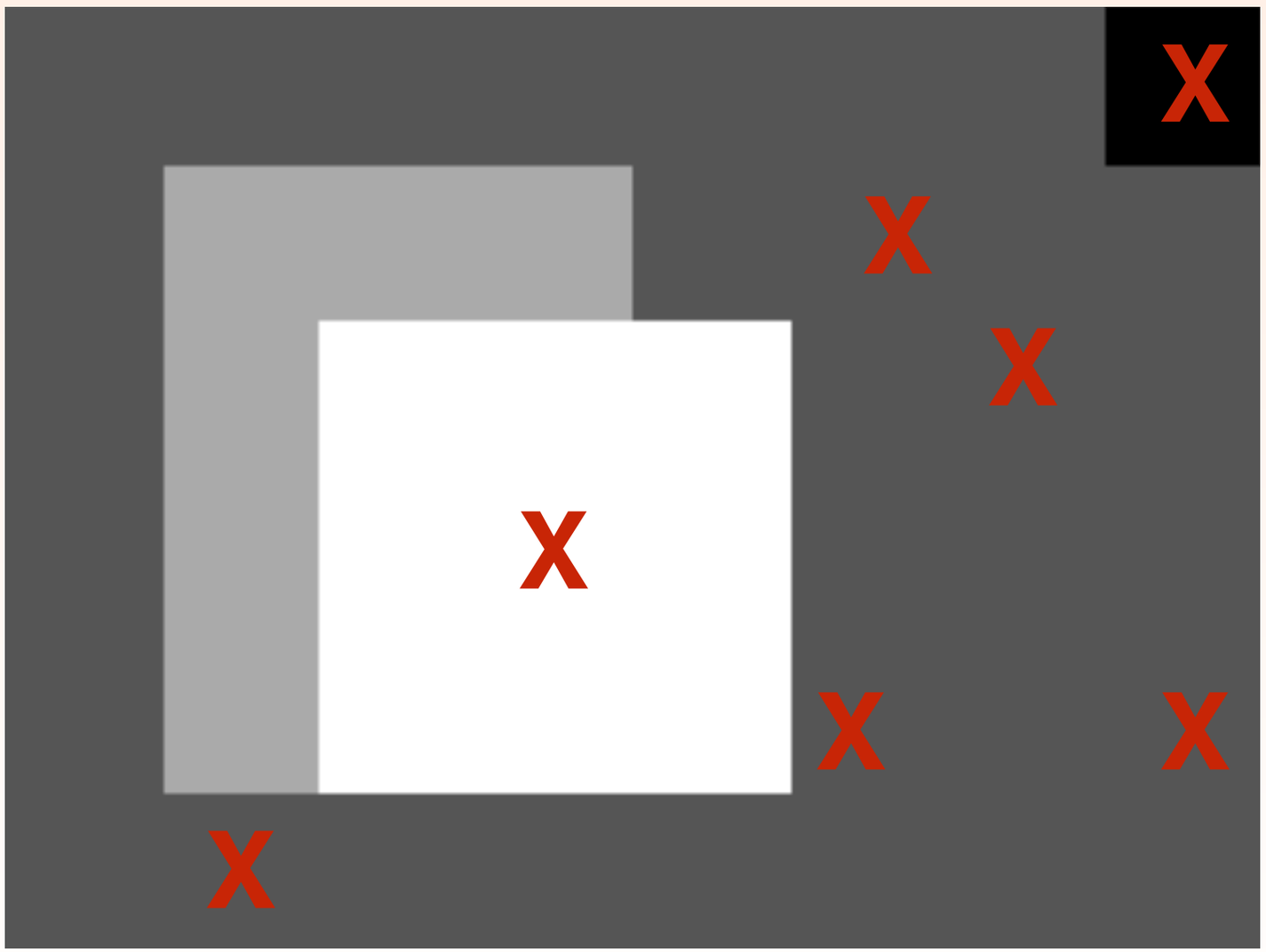


So

Result: local maxima finding



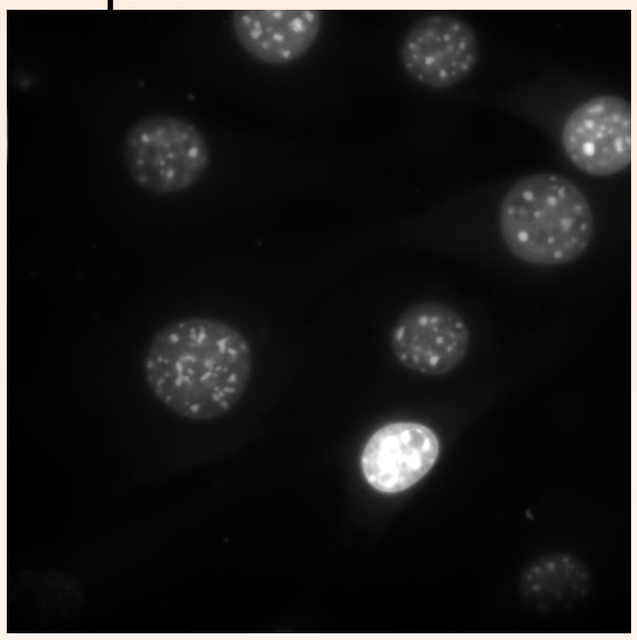
Where the pixel is the same in the input as the maximum filtered image you find your local maxima



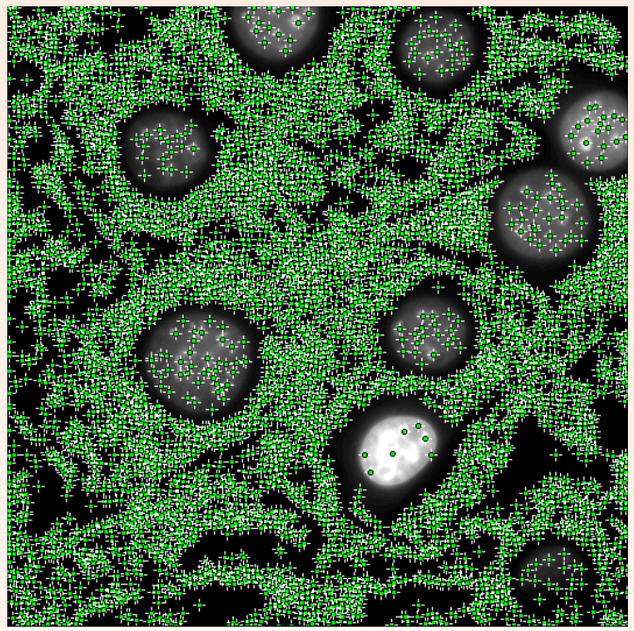
Source:

Local maxima to

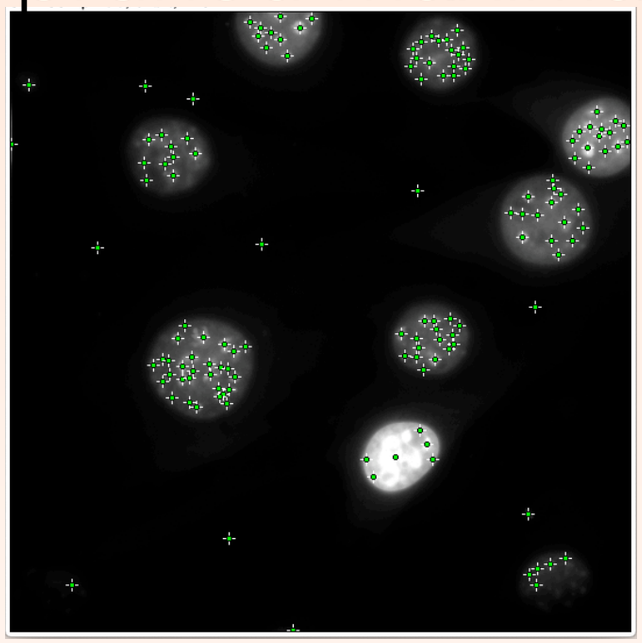
input



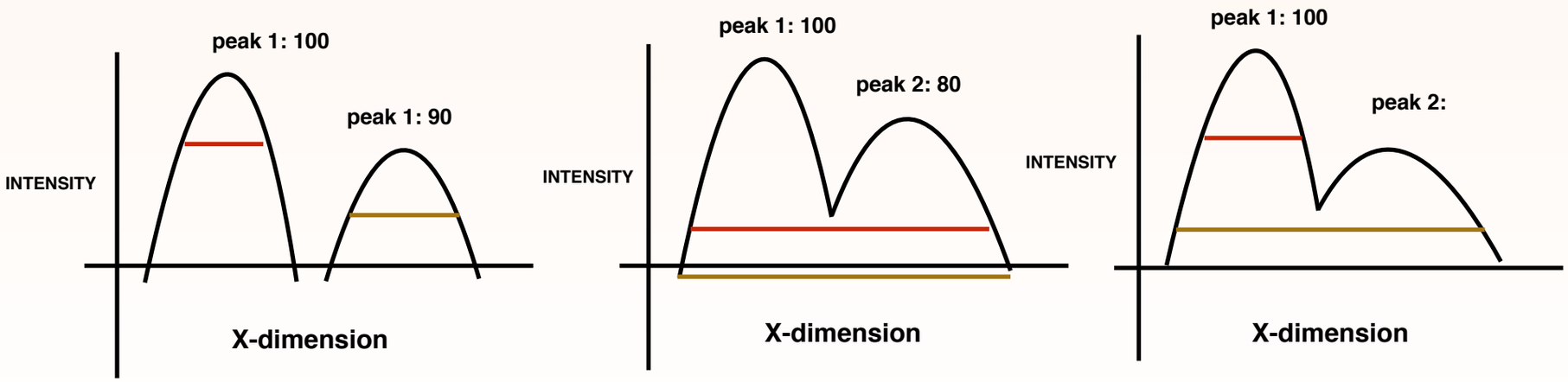
local maxima



pruned maxima



- The result is a little bit overwhelming and requires ‘pruning’.
- Many different strategies for pruning the foci away.
- In the practical we retain those foci who above the mean intensity + 3x standard deviations and also optionally try more advanced approaches.



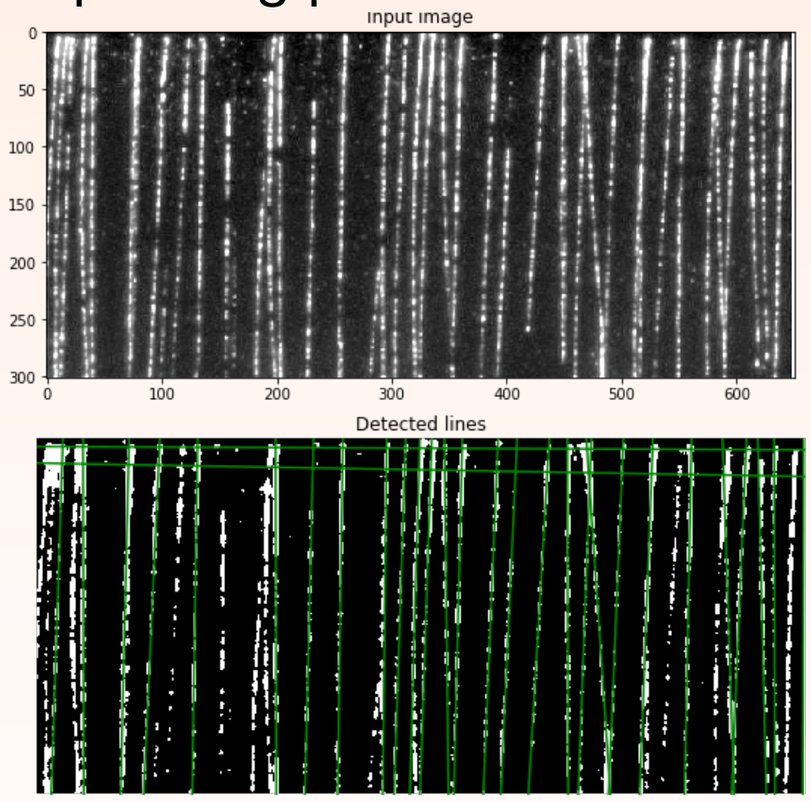
Feature finding: Hough line transform

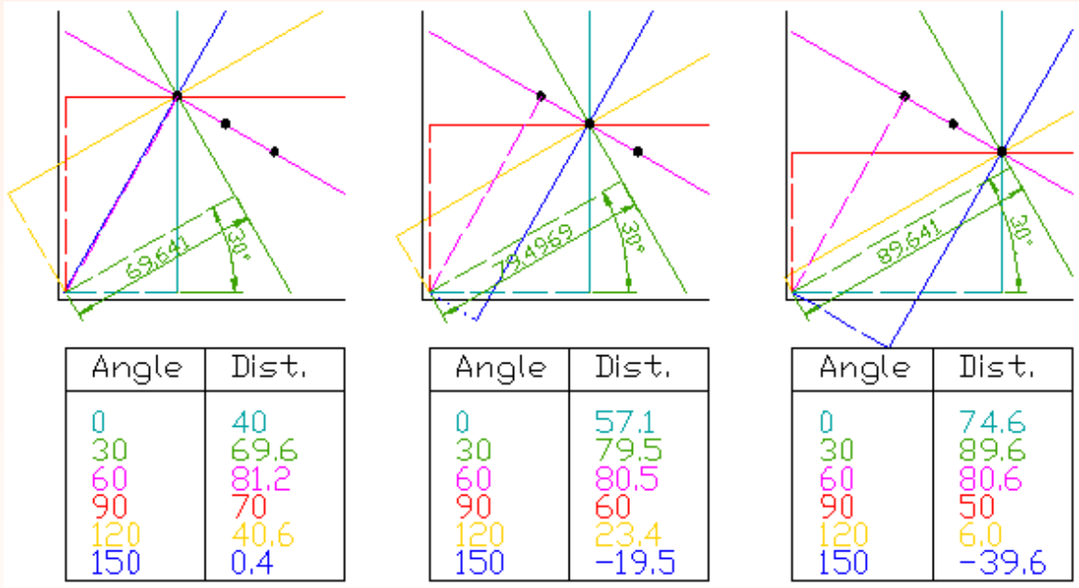
Here we look for structures in an image which are line like.

This involves positioning a line at each point in the image, and then effectively* spinning it round.

Where the line intersects many points its position and rotation will obtain many votes, and vice-versa.

Top voting positions and rotations are retained.





Angle	Dist.
0	40
30	69.6
60	81.2
90	70
120	40.6
150	0.4

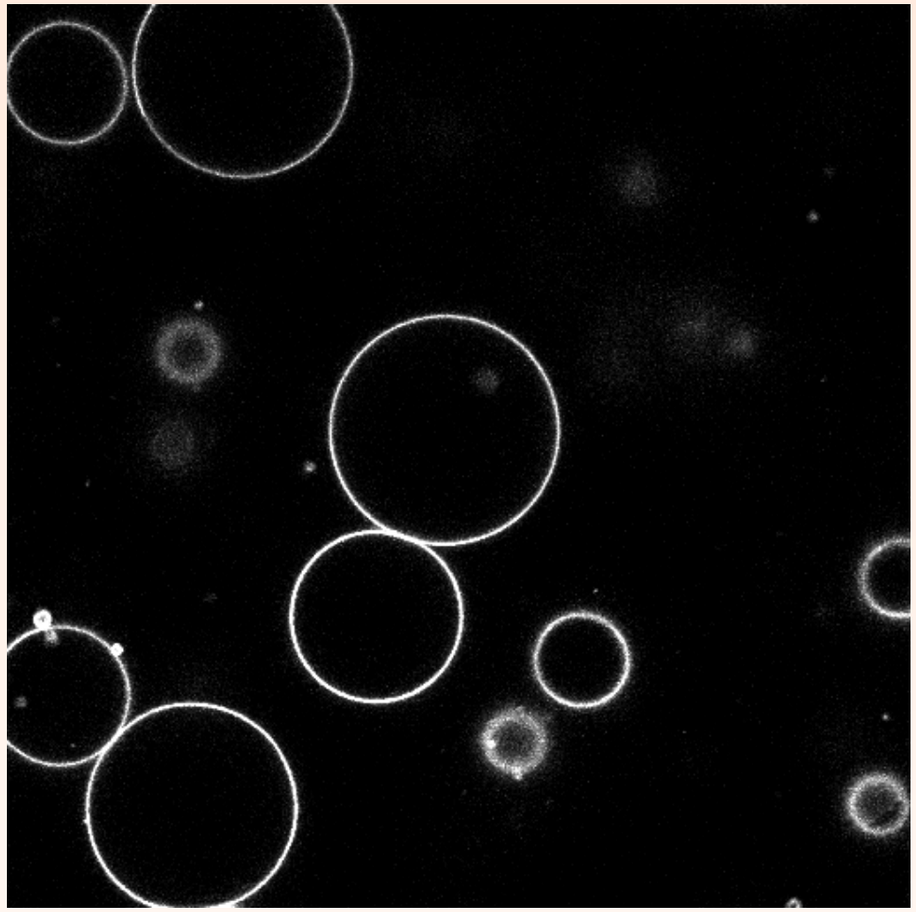
Angle	Dist.
0	57.1
30	79.5
60	80.5
90	60
120	23.4
150	-19.5

Angle	Dist.
0	74.6
30	89.6
60	80.6
90	50
120	6.0
150	-39.6

* Technically lines are perpendicular to a line connected to the origin. Which avoids problems with numerical stability.

Source: Image Citation: Payne AC, Andregg M, Kemmish K, Hamalainen M, Bowell C, et al. (2013) Molecular Threading: Mechanical Extraction, Stretching and Placement of DNA Molecules#from a Liquid-Air Interface. PLoS ONE 8(7): e69058. doi:10.1371/journal.pone.0069058

Feature finding: Hough Circular transform.



For a Hough circular transform we scan the image with a serious of rings.

At each pixel we scan the perimeter of the circle and accumulate votes for this location, based on the intensity levels around the perimeter.

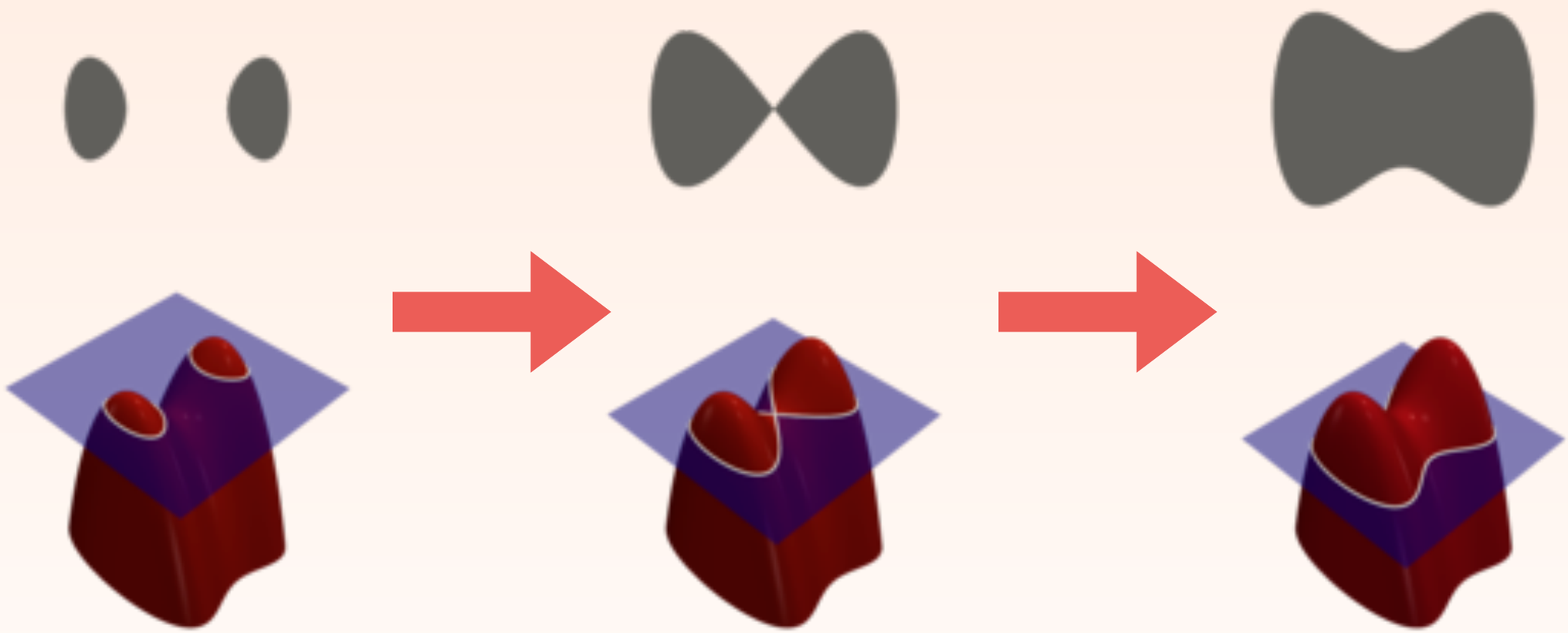
In a separate image for each size we stores the votes in each location. We then take the most significant accumulated peaks.

Source: Image Credit (Erdinc Sezgin)

Advanced numerical segmentation methods

Level sets methods.

Previous parametric methods work well in specific cases, but in many cases the relationships between object shapes and other objects is too complex, we need another approach.



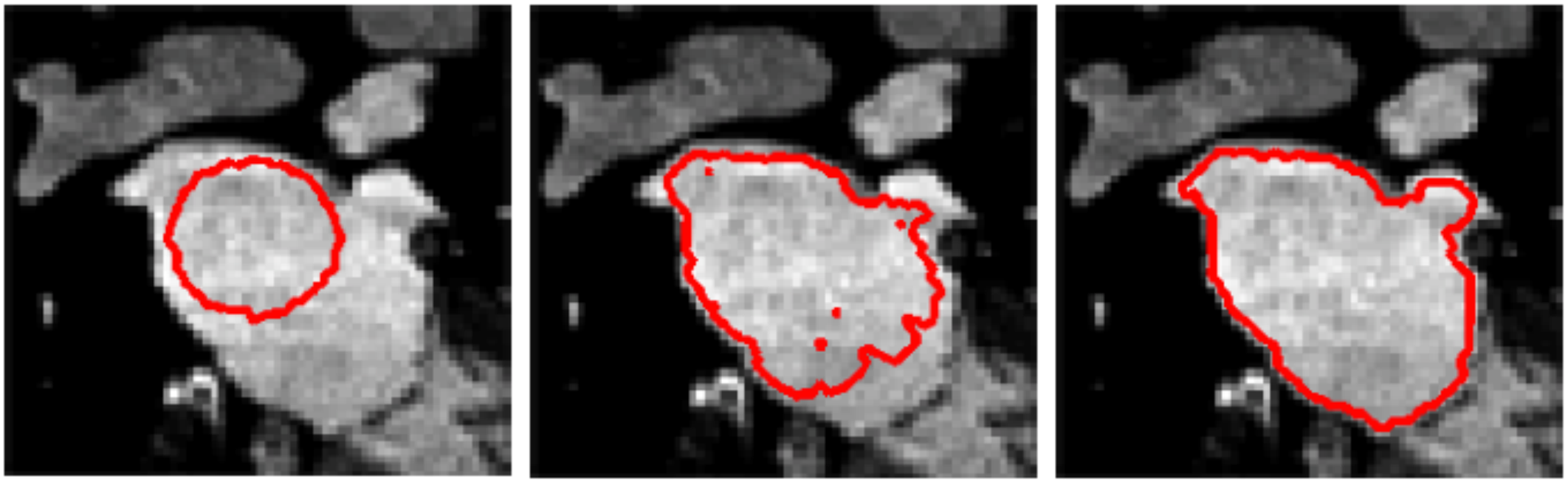
A level set i.e., the set of points in the domain of a function where the function is constant. We can use these to analyse shapes without having to parameterise anything.

Level-set method (LSM)

1) Typically we evolve our level set from one or more seed points.

2) Level sets advance like a rubber band looking for an energetic optimum.

3) Level sets are typically designed to stop where the boundary is sharp (i.e where the intensity gradient is strong at edges).



(a) initial contour (b) intermediate step (c) stopped by the edges

Many different forms of level set formulation exist with different properties and constraints.

Source: images adapted from https://profs.etsmtl.ca/hlombaert/levelset/

skimage chan_vese level set algorithm.

```
from skimage.segmentation import chan_vese

image = img_as_float(input)

cv = chan_vese(image, mu=0.25, lambda1=1, lambda2=1,
tol=1e-3, max_iter=200,dt=0.5,
init_level_set="checkerboard", extended_output=True)
```

Typical values for **mu** are between 0 and 1, though higher values can be used when dealing with shapes with very ill-defined contours.

Typical values for **lambda1** and **lambda2** are 1. If the ‘background’ is very different from the segmented object in terms of distribution (for example, a uniform black image with figures of varying intensity), then these values should be different from each other.

Tol is the level set variation tolerance between iterations. Will stop sooner if tolerance is high.

max_iter

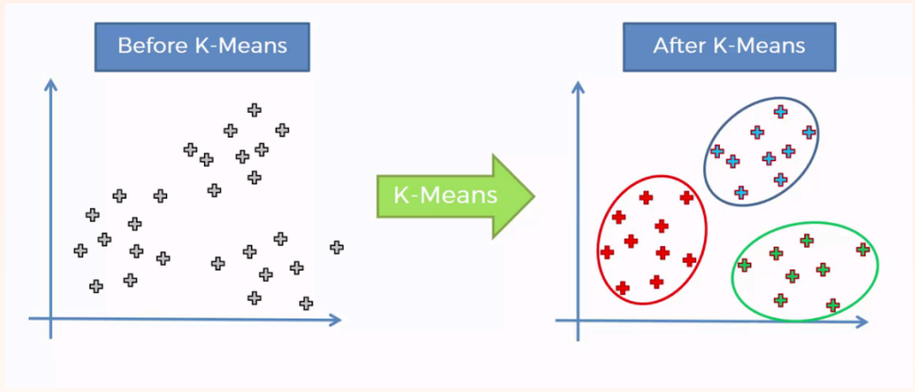
Source: Chan-Vese Segmentation, Pascal Getreuer, Image Processing On Line, 2 (2012), pp. 214-224, DOI:10.5201/ipol.2012.g-cv https://scikit-image.org/docs/dev/auto_examples/segmentation/plot_chan_vese.html

Adv. segmentation: Superpixels

Superpixels provide a method to segment an image into convenient primitives.
Each super pixel represents a group of pixels which share common characteristics.

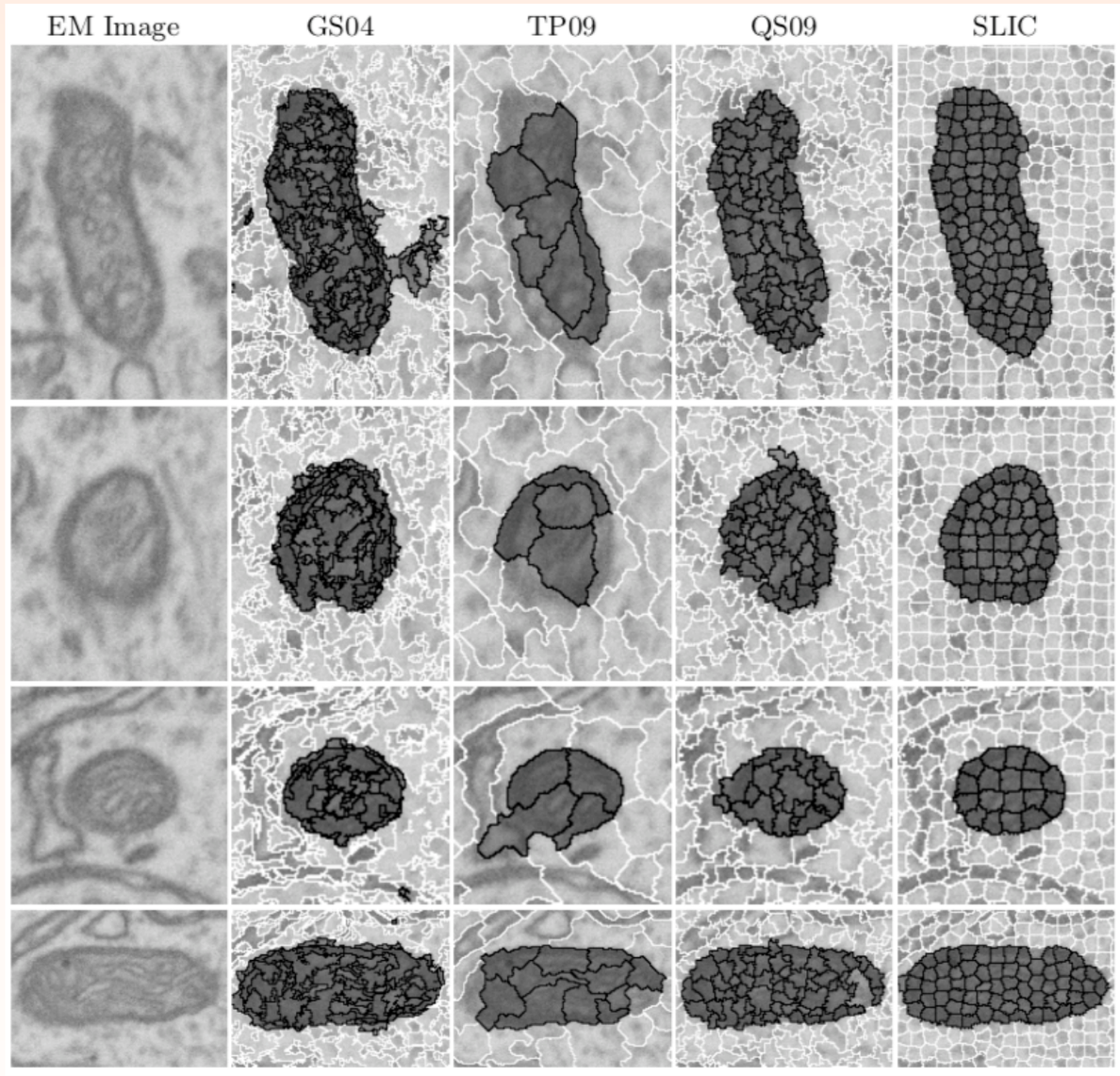
Simple Linear Iterative Clustering (SLIC)

Works by unsupervised clustering of pixels, using k-means strategy.



Works for grayscale and RGB images.

Fast and powerful approach.



Source: http://www.kev-smith.com/papers/SLIC_Superpixels.pdf

Adv. segmentation: Superpixels

```
from skimage.segmentation import slic
```

Important parameters:

n_segments, the number of superpixels in which to divide you image up into.

s**igma**, defines the smoothing Gaussian kernel which applied prior to the segmentation.

max_iter, the number of iterations for the k-means clustering.

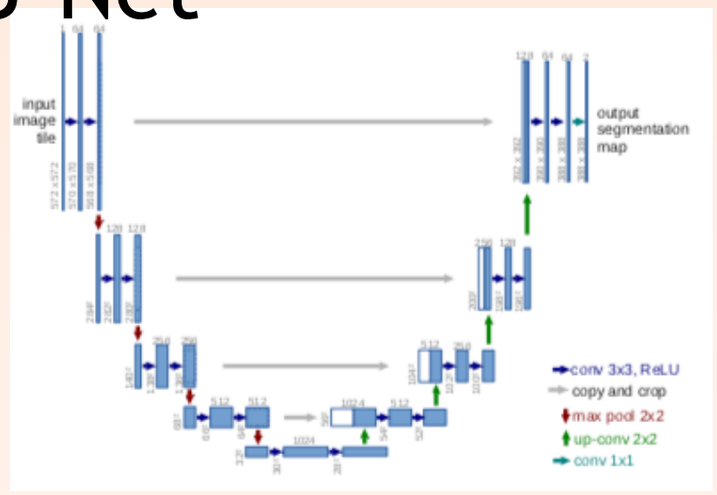
compactness, which balances the color-space proximity versus the image space-proximity. Higher values give more weight to space proximity, making superpixel shapes more square/cubic.

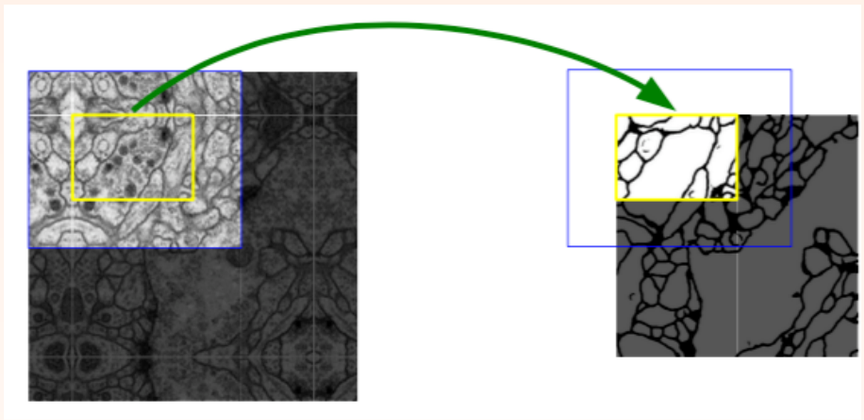
Spacing By default, slic assumes uniform spacing (same voxel resolution along z, y and x). This parameter controls the weights of the distances along z, y, and x during k-means clustering. Useful for 3-D image where voxel is anisotropic.

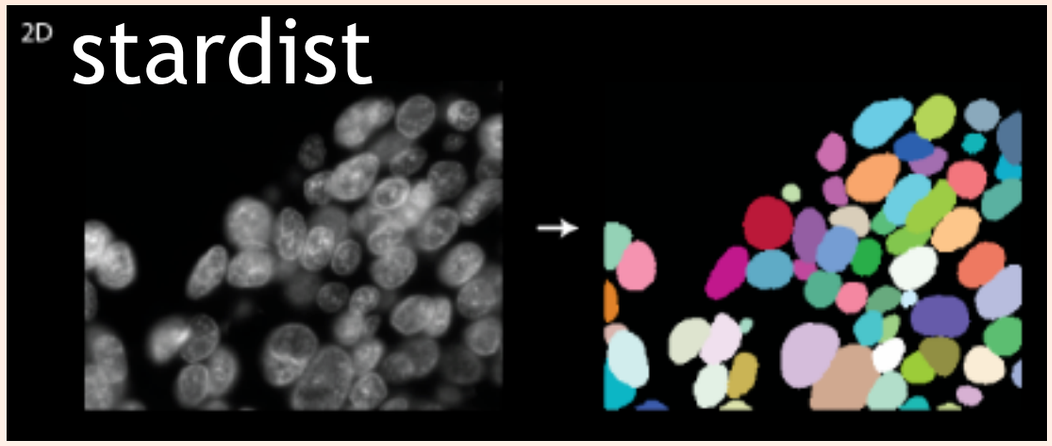
Source: https://scikit-image.org/docs/dev/api/skimage.segmentation.html#skimage.segmentation.slic

Computer Vision/Machine learning approaches

U-Net







The current state-of-the-art involves using machine learning, and especially deep learning approaches to solve segmentation. Usually require lots of training data.

We find out more about this on Friday.

Source: https://biii.eu/node/1486 https://arxiv.org/abs/1703.06870 https://arxiv.org/abs/1505.04597 https://github.com/mpicbg-csbd/stardist

Overview of methods

- Many powerful methods for segmentation
- No one answer to solving a question.
- Art is applying the right algorithm for the right problem.
- The more parameters an approach has, generally the more powerful the algorithm can be. (no guarantees).
- Worth thinking carefully about which approach is best and when to use it.

Two challenges:

- If you have enough data, most segmentation problems solvable through the application of deep learning, but still requires large volumes of specific annotated data.
- The challenge as a field is to be knowledgeable enough of the approaches that you can use them at the right-time.

Source: https://scikit-image.org/docs/dev/api/skimage.segmentation.html#skimage.segmentation.slic

Last slide, a chance to promote your things

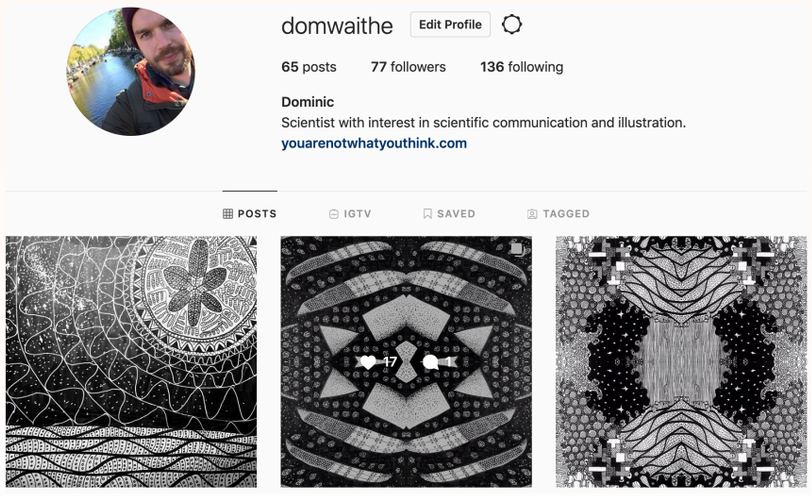
For this content and more:

https://github.com/IAFIG-RMS/Python-for-Bioimage-Analysis

https://twitter.com/dwaithe

https://github.com/dwaithe

https://instagram.com/dwaithe



dominic.waithe@imm.ox.ac.uk