Lecture 6: Separating objects with the watershed algorithm

MCDB/BCHM 4312/5312

5312 (non IQ): Reminder to sign up for journal club

Dates still available:

- Nov 15 (Friday)
- Dec 2 (Monday)
- Dec 6 (Friday)

Please email with preferred date and paper

Upcoming MATLAB Class Schedule

Date	Topic
10/11	Separating objects
10/18	Image intensity corrections
10/25	Plots, curve fitting, and for loops
11/1	FRET imaging
11/11 (Mon)	Tracking bees
11/20 (Wed)	Particle detection and localization

Topics we probably won't get to cover

Programming:

- Functions
- Logical functions: `&`, `|`, `any`, `all`
- While loops
- Debugging in MATLAB

Image analysis:

- Validating segmentation accuracy
- Image registration – drift correction, rotations and scaling
- Fourier image transforms – structured illumination microscopy
- 3D object segmentation

Is anyone interested in any of these topics?

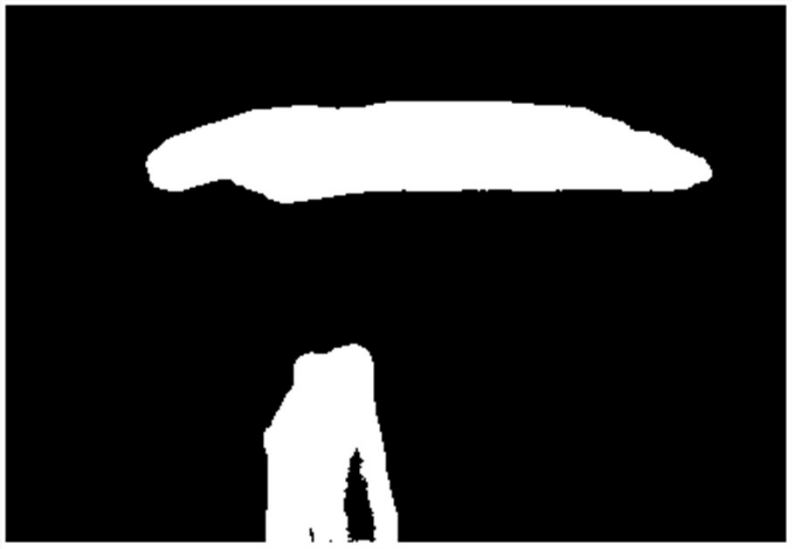
Learning goals

- MATLAB tips
  - `imclearborder`
- Separating objects in masks:
  - The distance transform
  - h-minima transform
  - Watershed transform
  - Often just referred to collectively as "watershedding"

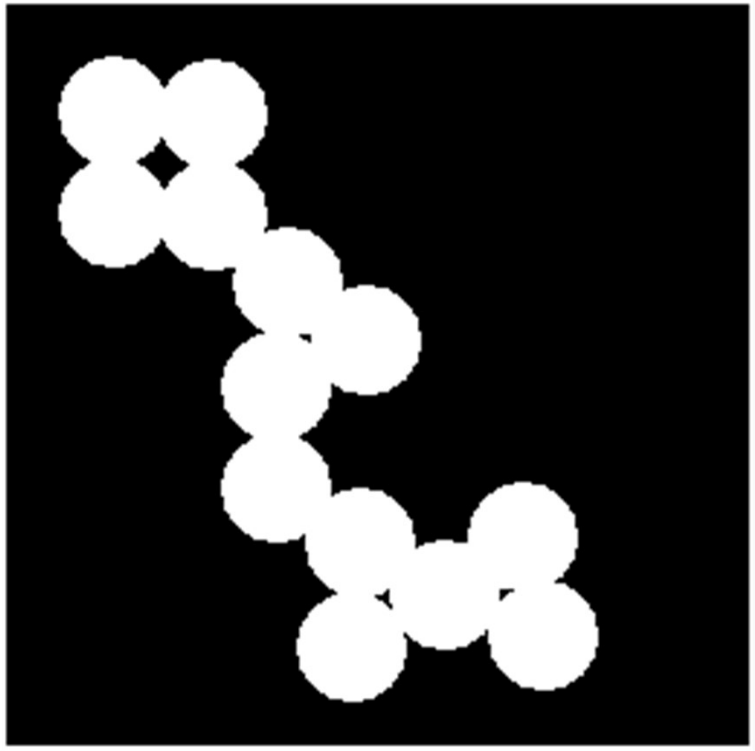
A few MATLAB tips

Removing masks that intersect with the image edge

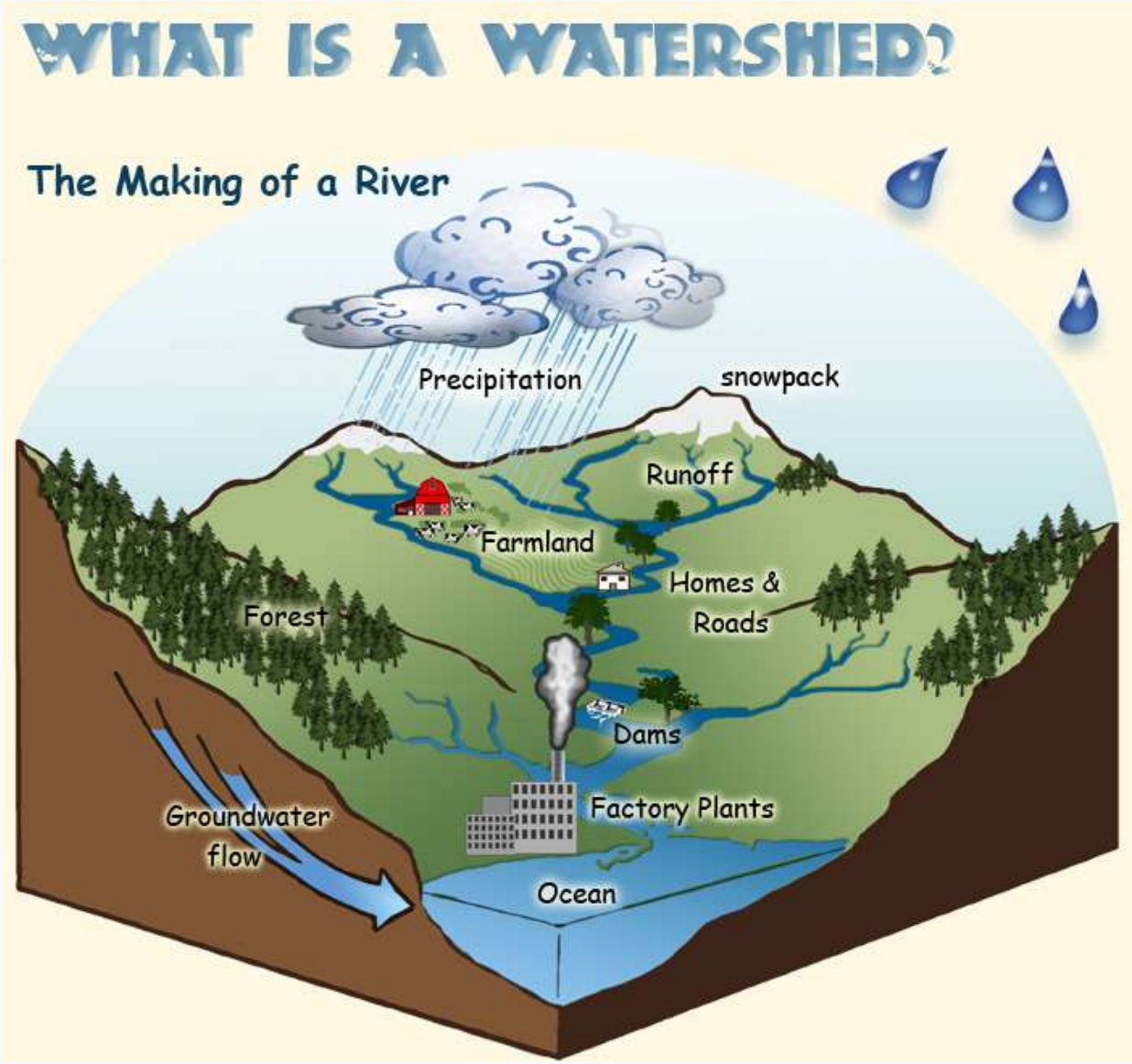
```
imclearborder(mask)
```



Separating connected objects using the watershed algorithm



`regionprops` treats connected pixels in the mask as the same object



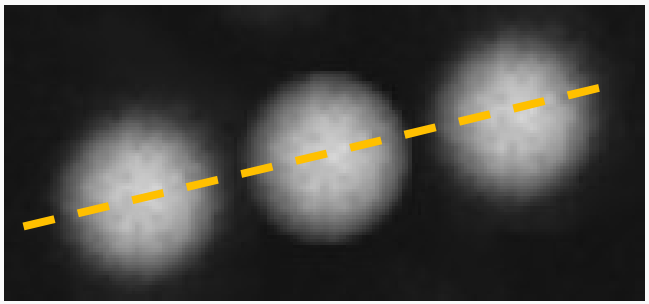
A Watershed is an area of land where all of the water that … drains off of it collects into the same place

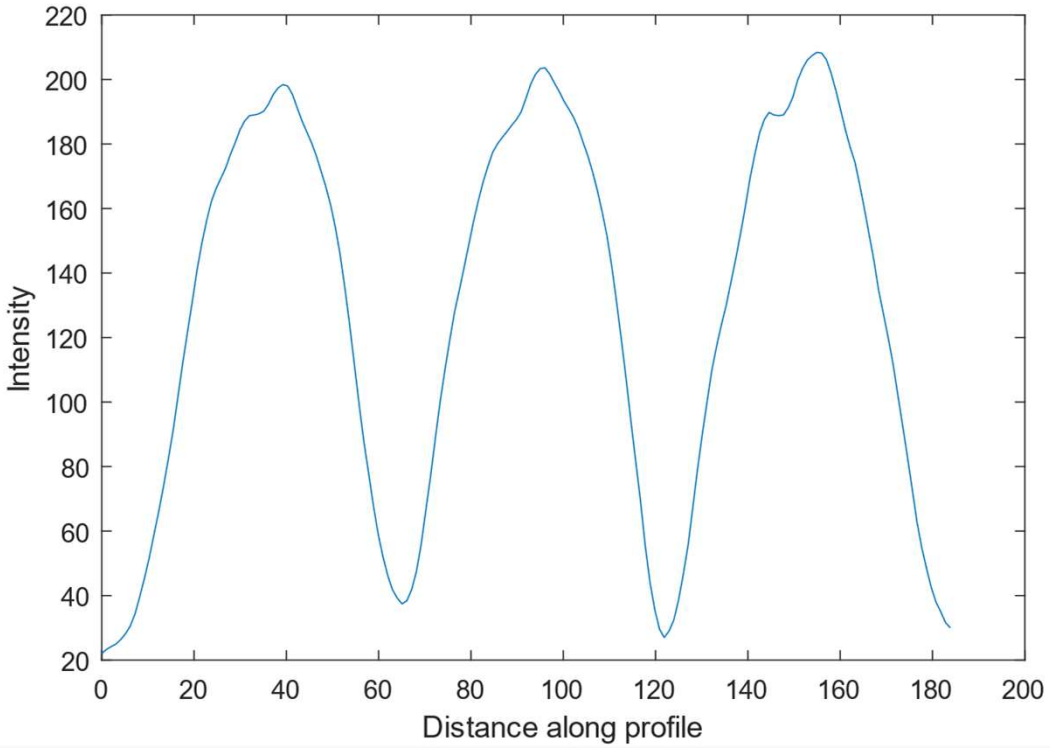
https://www.fs.fed.us/rm/boise/research/techtrans/projects/scienceforkids/watersheds.shtml

What is a watershed?

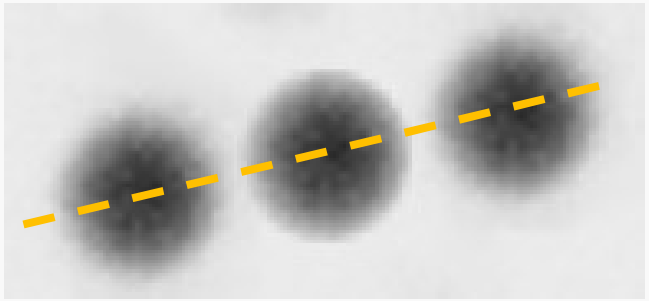
https://commons.wikimedia.org/wiki/File:NorthAmericaDivides.gif

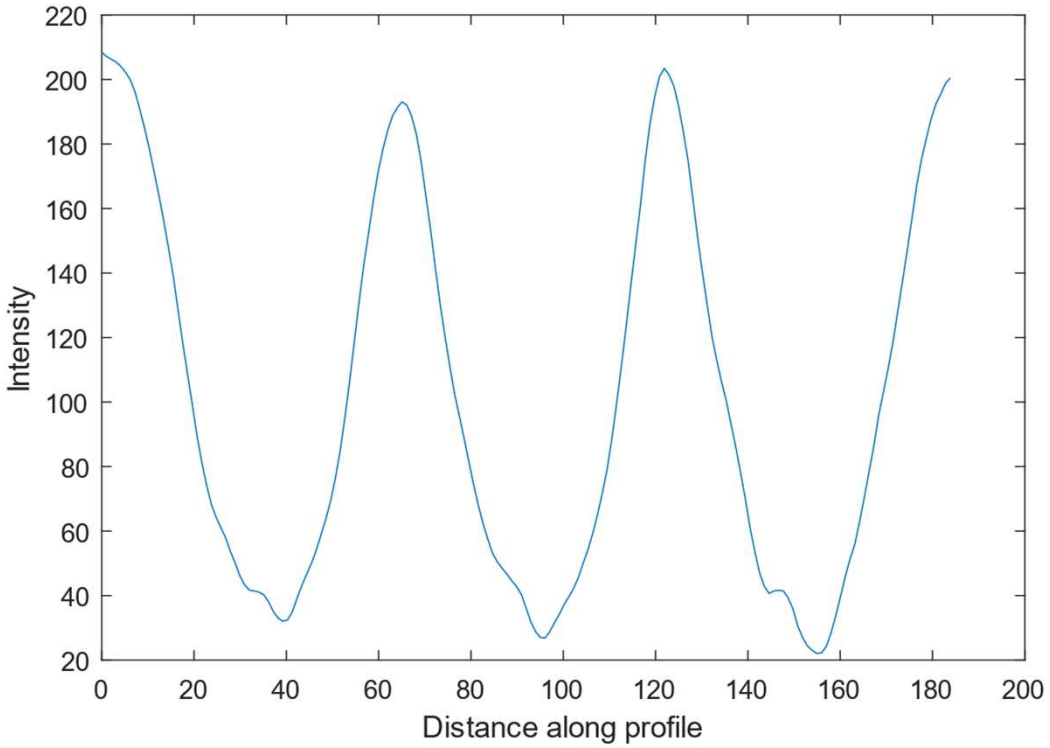
Separating connected objects using the watershed algorithm





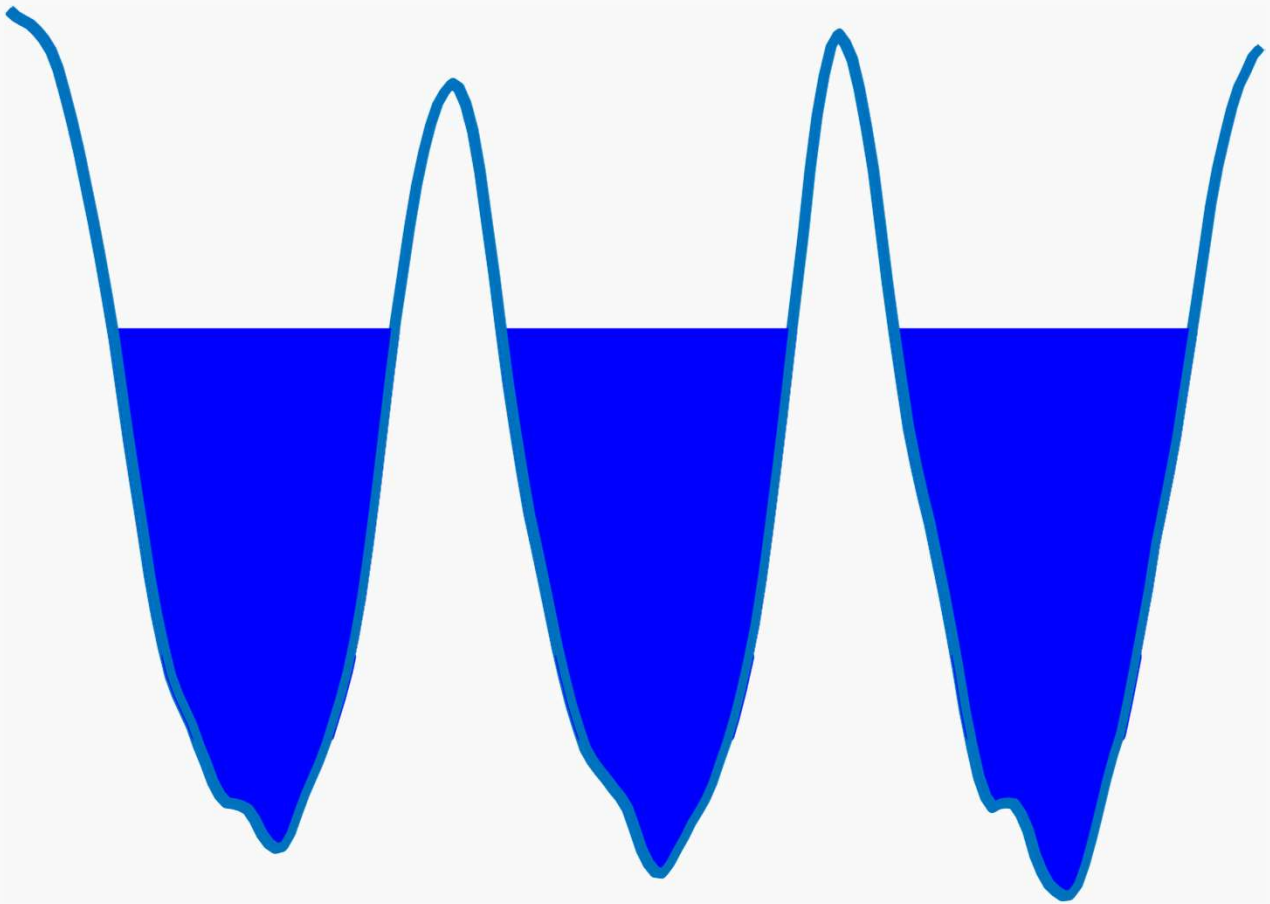
Separating connected objects using the watershed algorithm





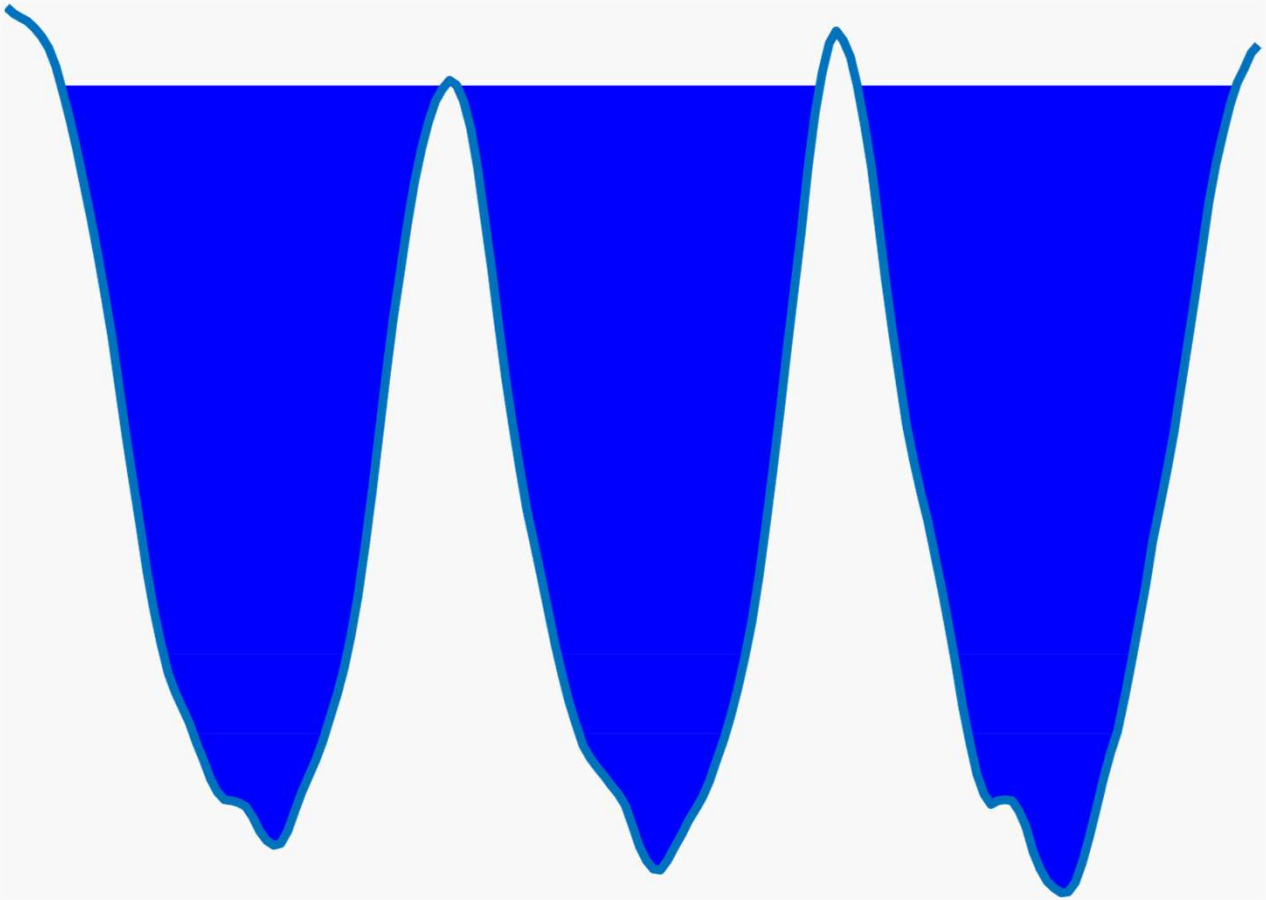
Separating connected objects using the watershed algorithm

Imagine it is raining and the basins are filling up



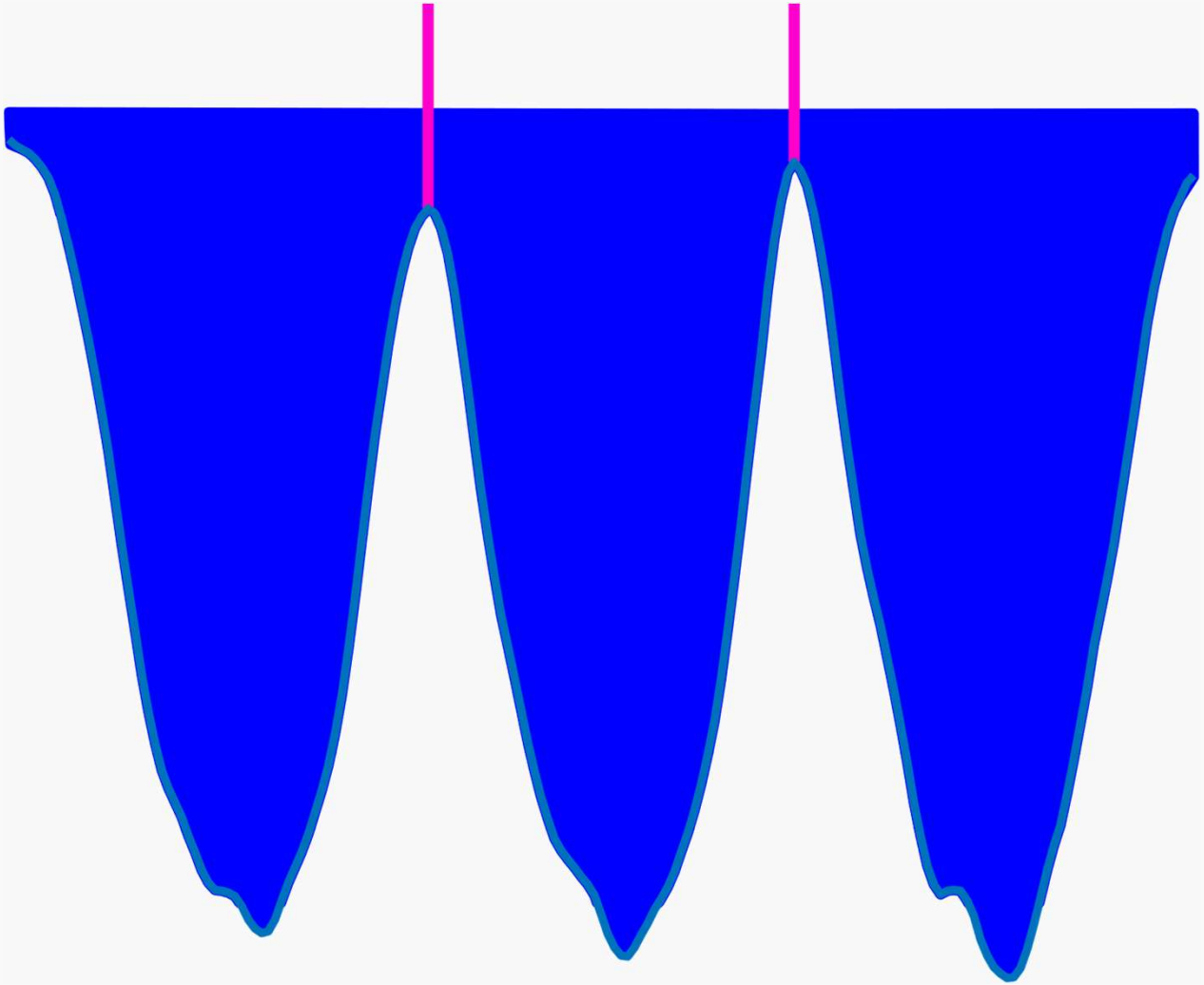
Separating connected objects using the watershed algorithm

Imagine it is raining and the basins are filling up

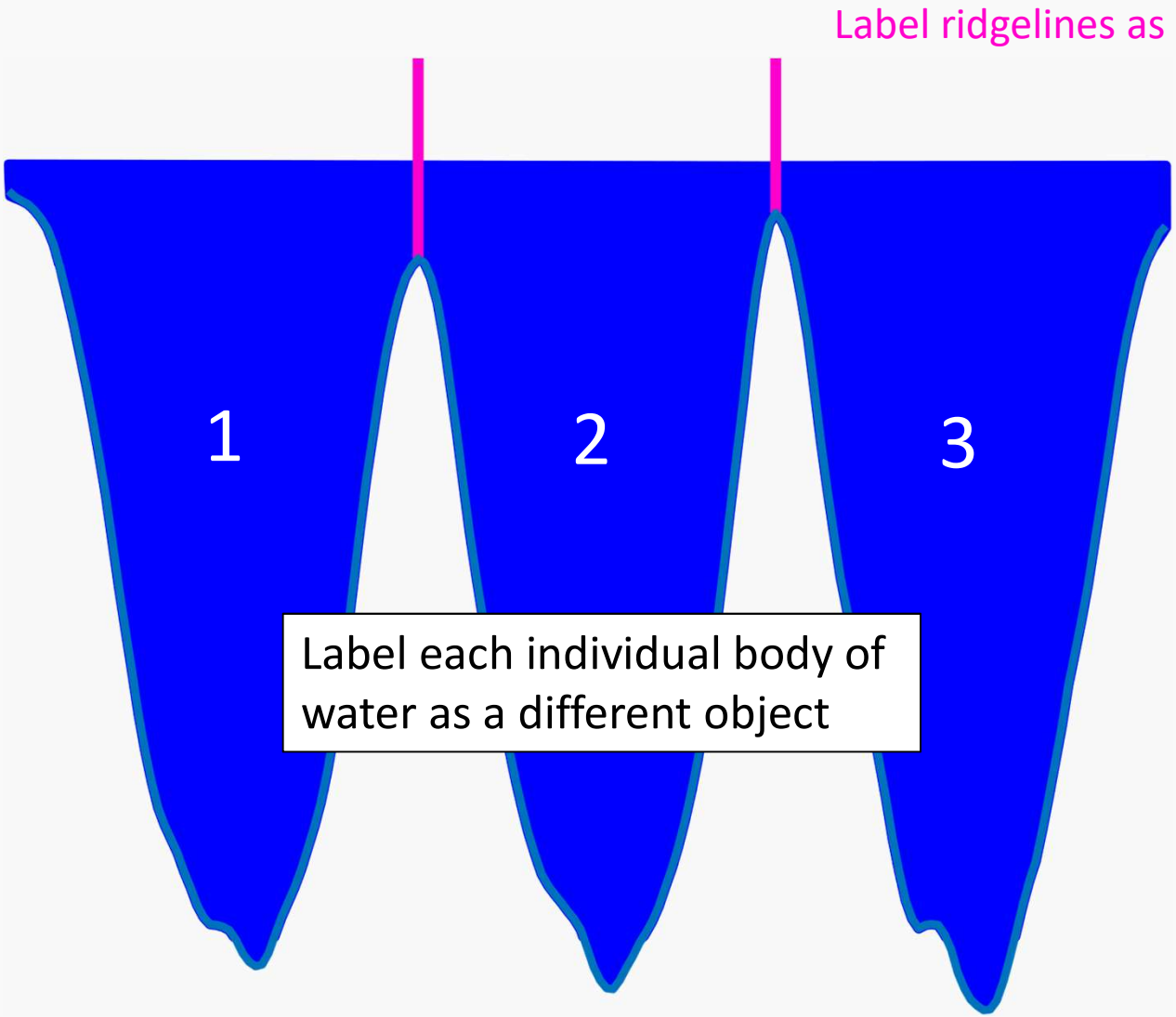


Separating connected objects using the watershed algorithm

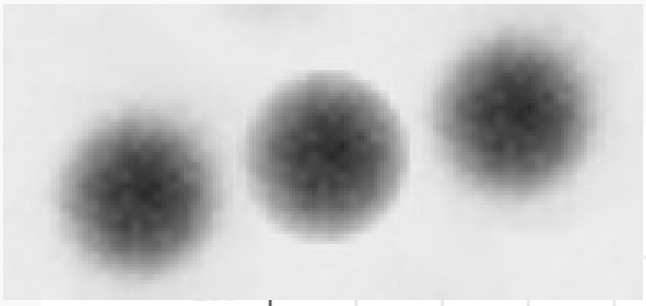
Mark locations where the bodies of water touch
(ridge lines)

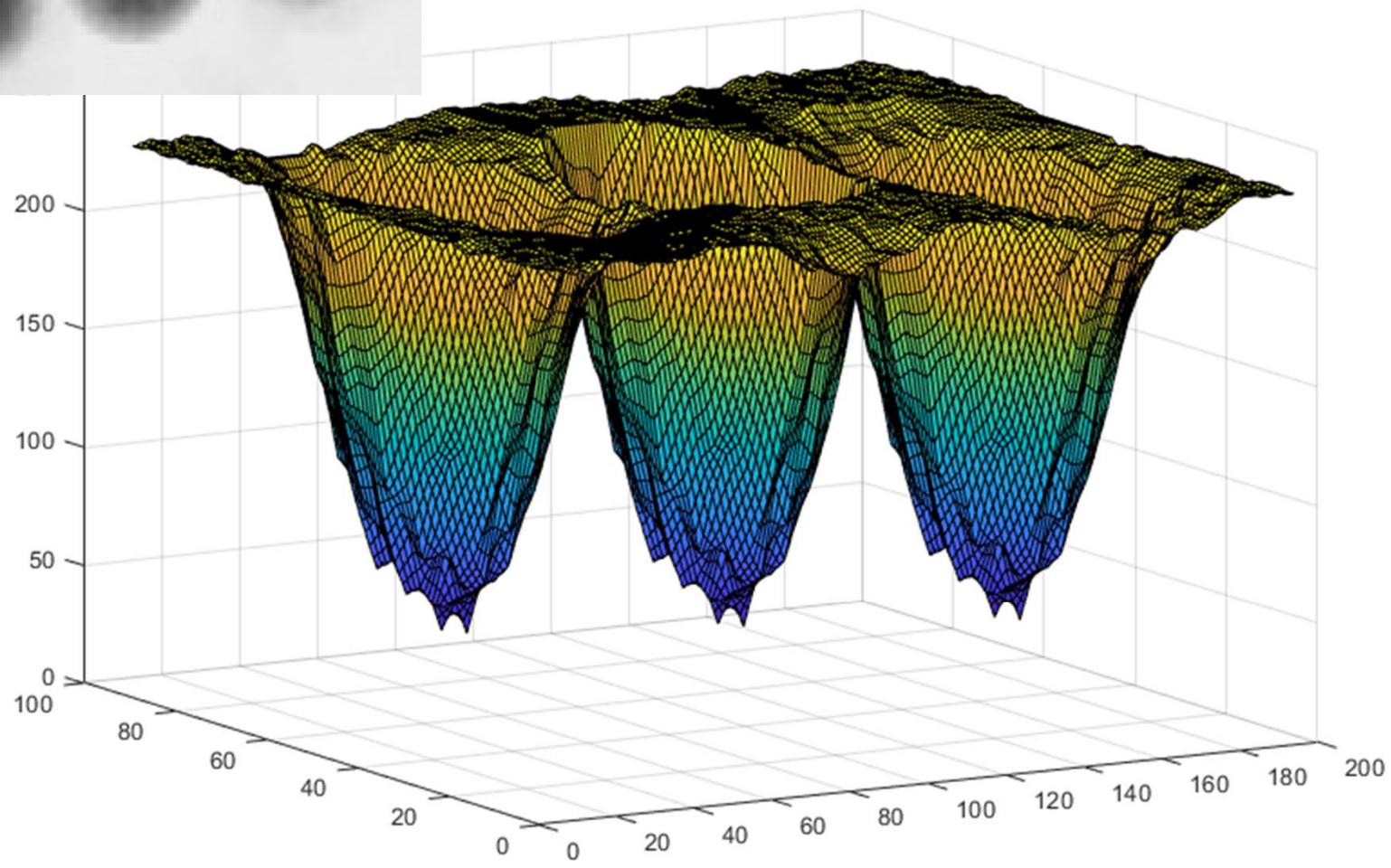


Separating connected objects using the watershed algorithm

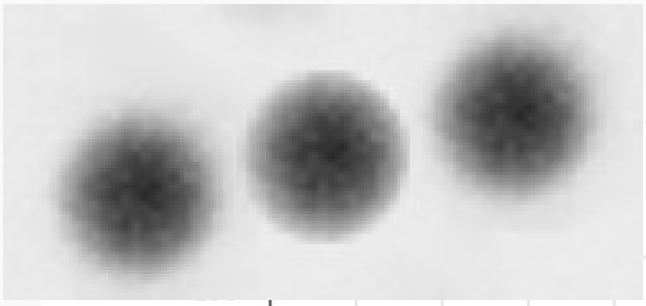


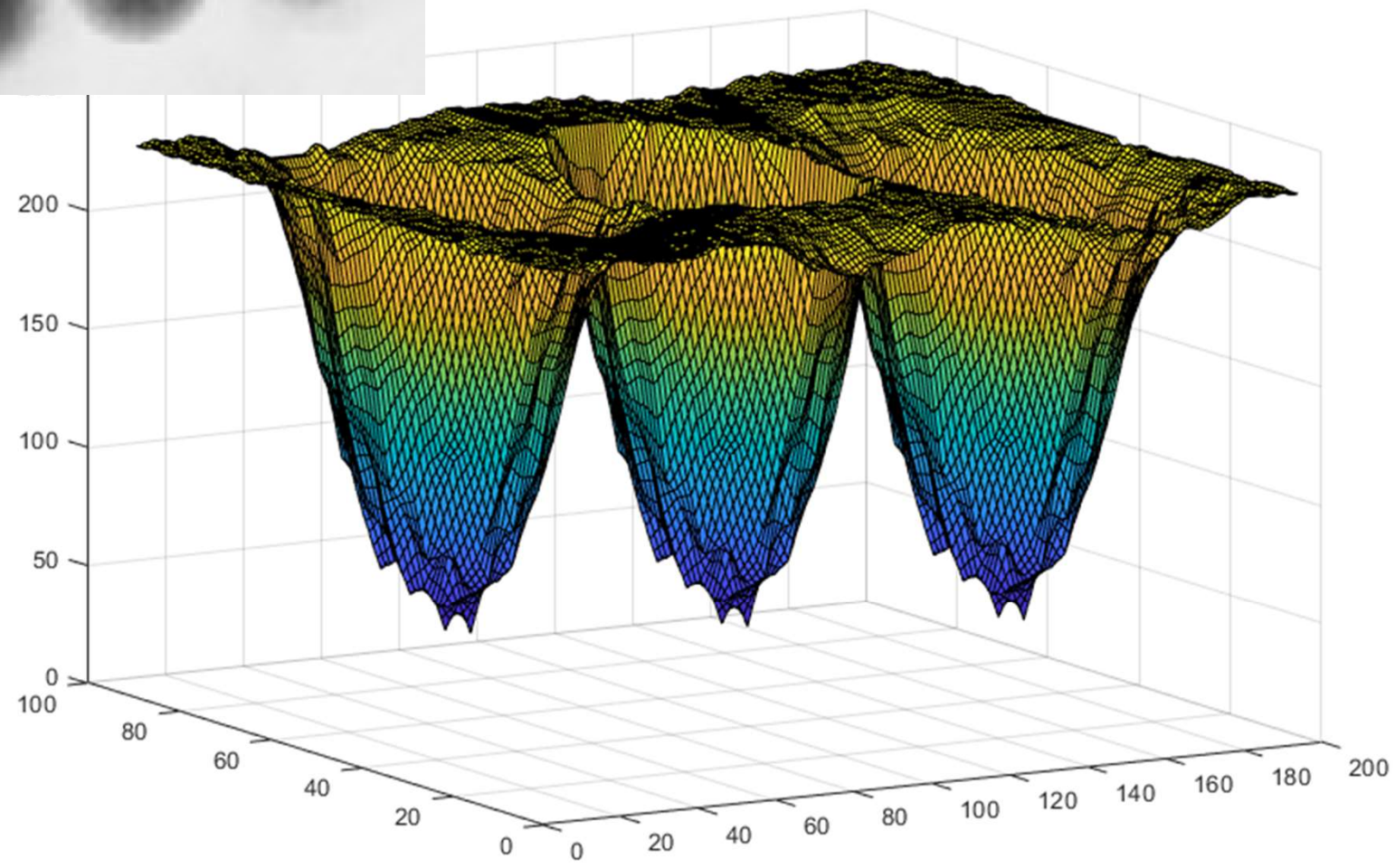
Actually works in 3D



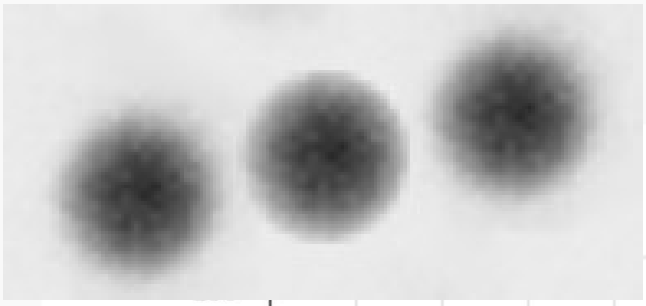


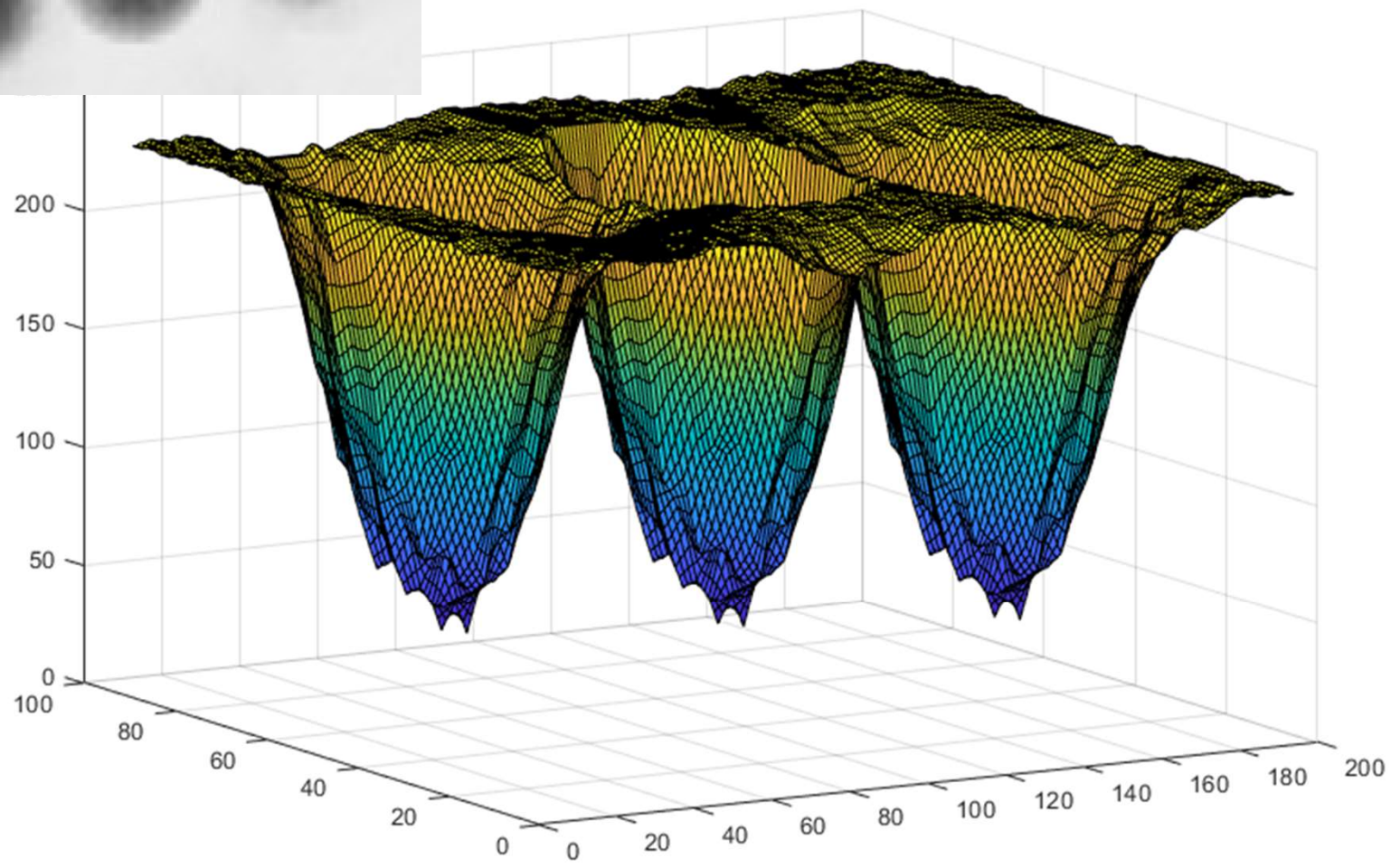
Must have a gradient for each object



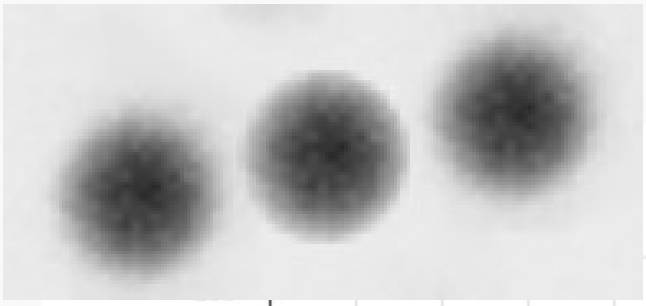


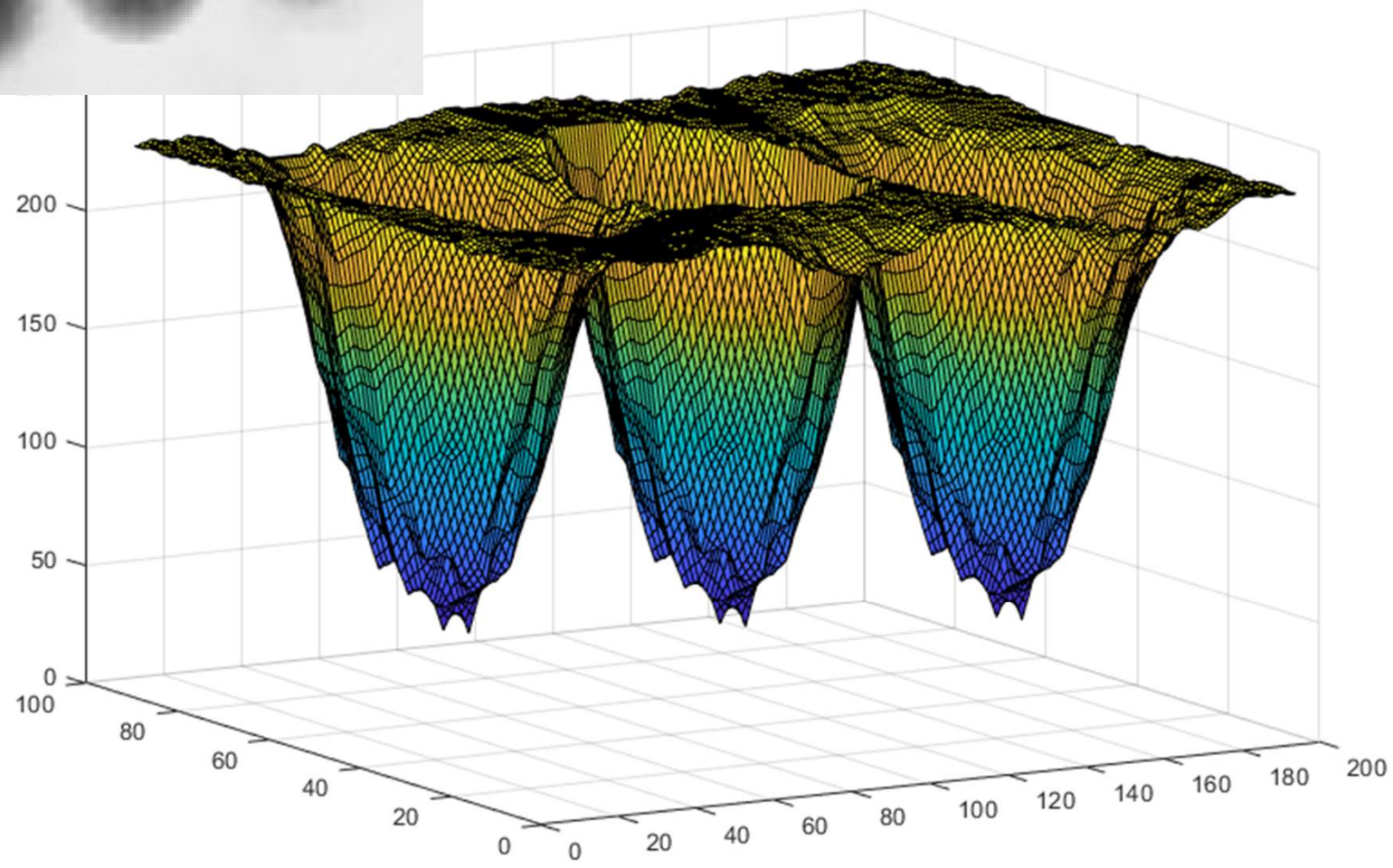
Objects should be basins





There should only be one local minima per object





Applying the watershed algorithm in practice

1. Generate a mask using an intensity threshold

2. Transform the mask into a topological graph using the inverse distance transform

3. Run the watershed algorithm

4. Update the mask

Generate a mask using an intensity threshold

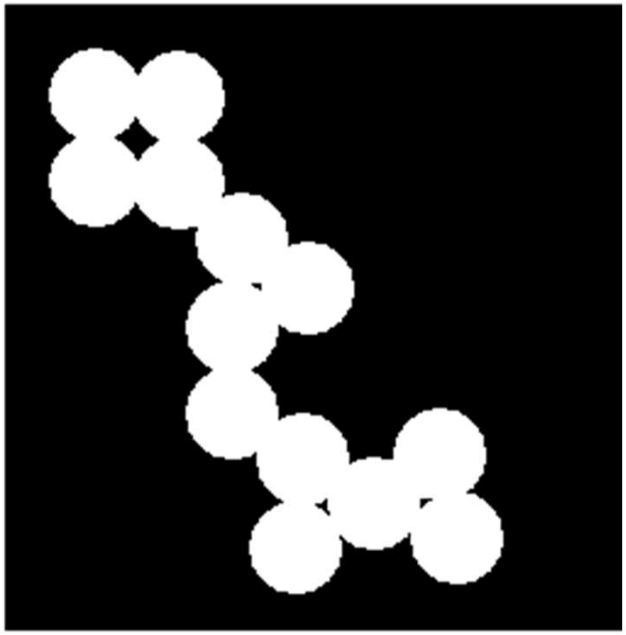
- Do this as a script

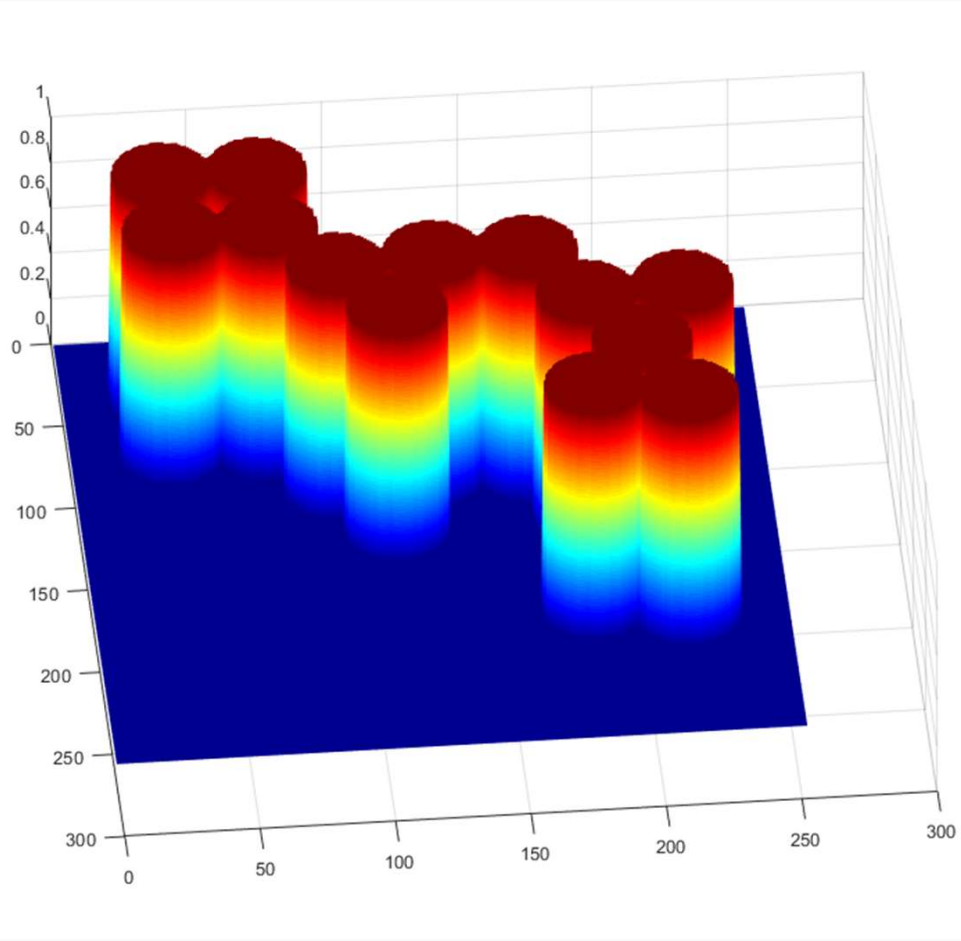
- Read in the file 'circles.png'. Pretend that this is your segmented mask.

```
mask = imread('circles.png')
```

Must have intensity gradients

- We need to transform the mask into a topological graph

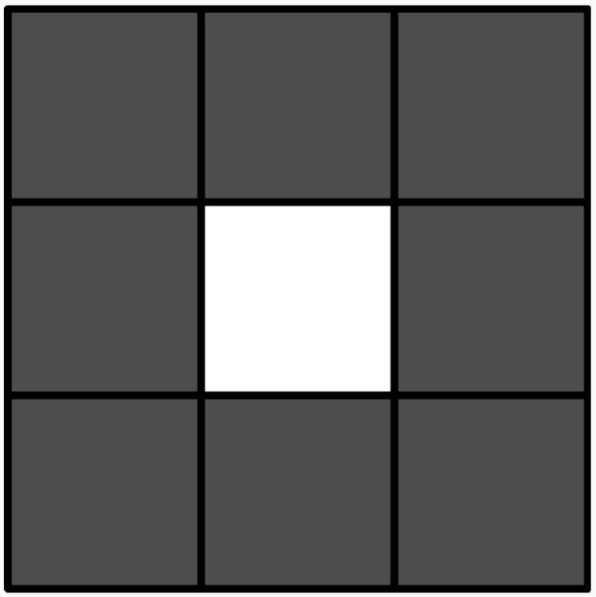




The distance transform `bwdist`

- `bwdist` computes the distance of a pixel to the nearest TRUE pixel

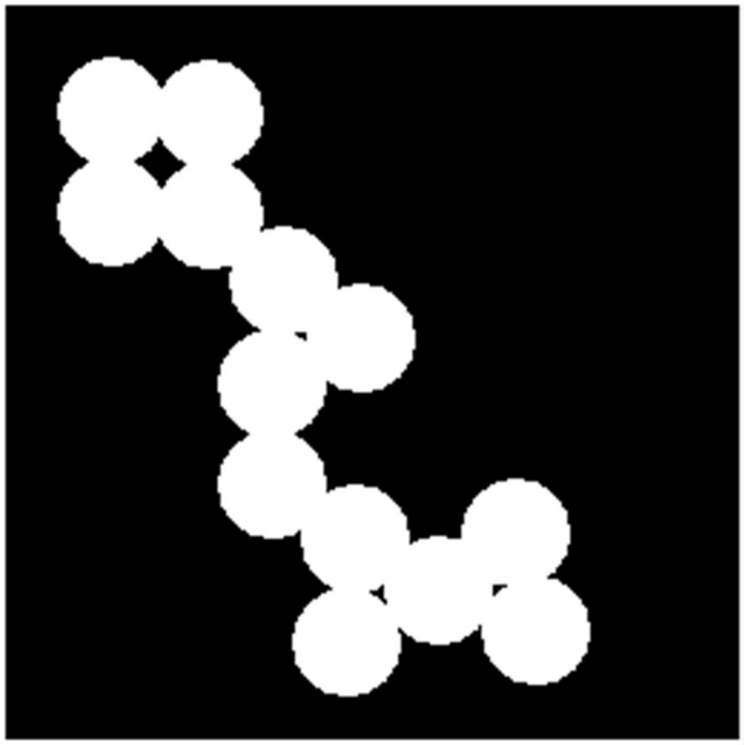
Mask

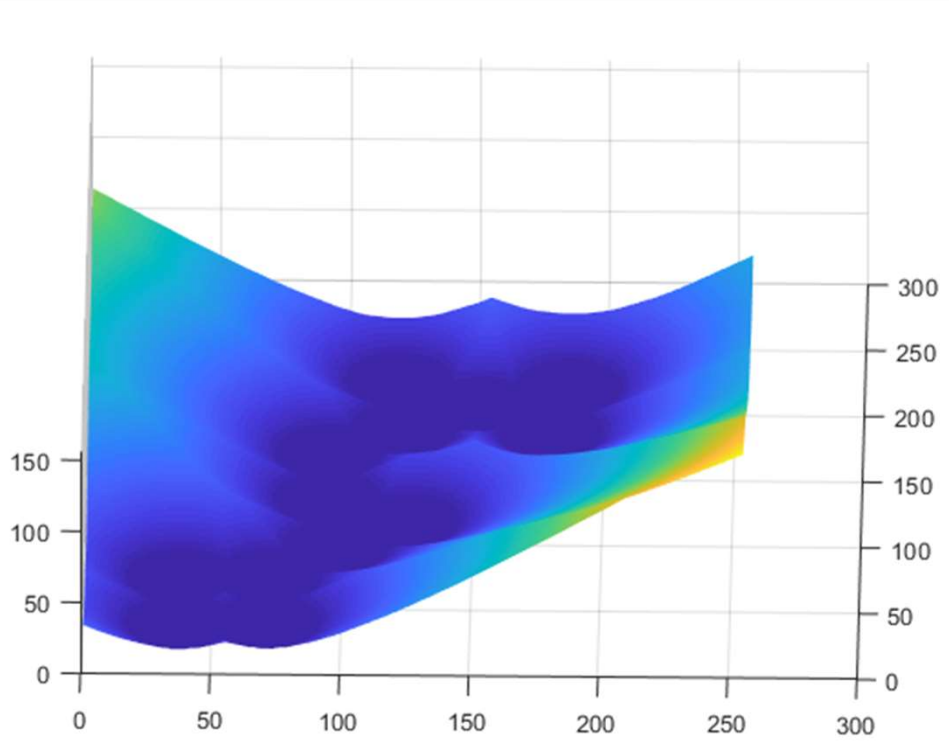


1.41	1	1.41
1	0	1
1.41	1	1.41

Problem: all the true pixels have a distance of zero

Every object is just one big basin



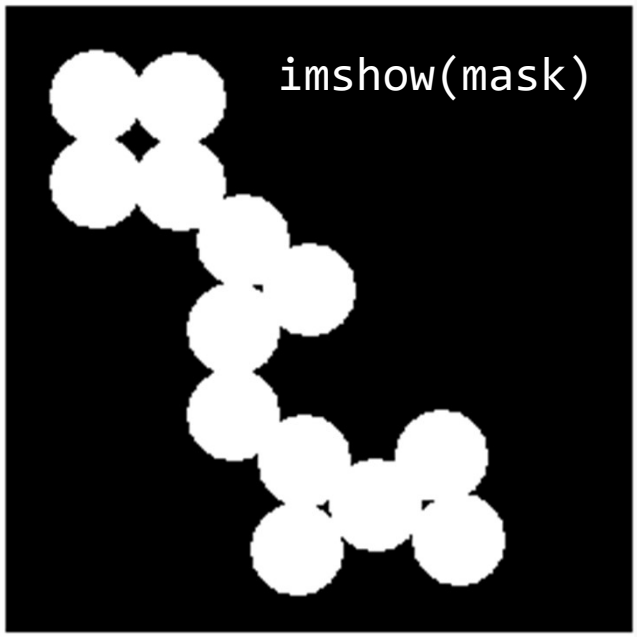


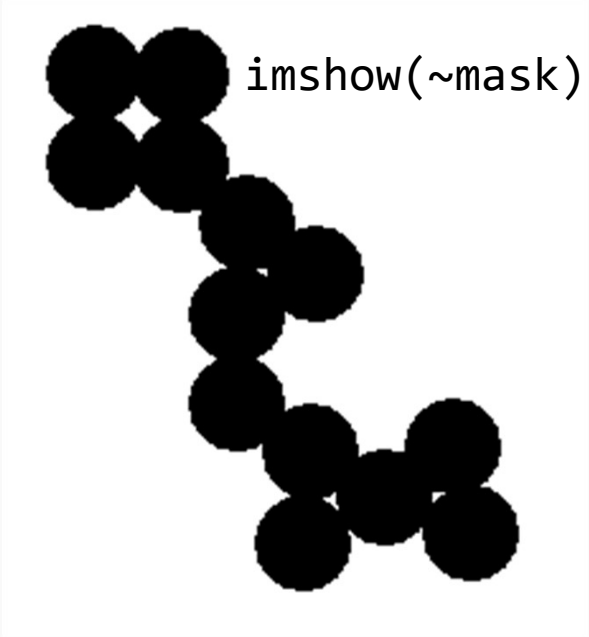
There must be a gradient WITHIN each object

Compute the distance transform of the inverted mask

- We have to compute the (negative) distance transform of the inverse of the cell mask

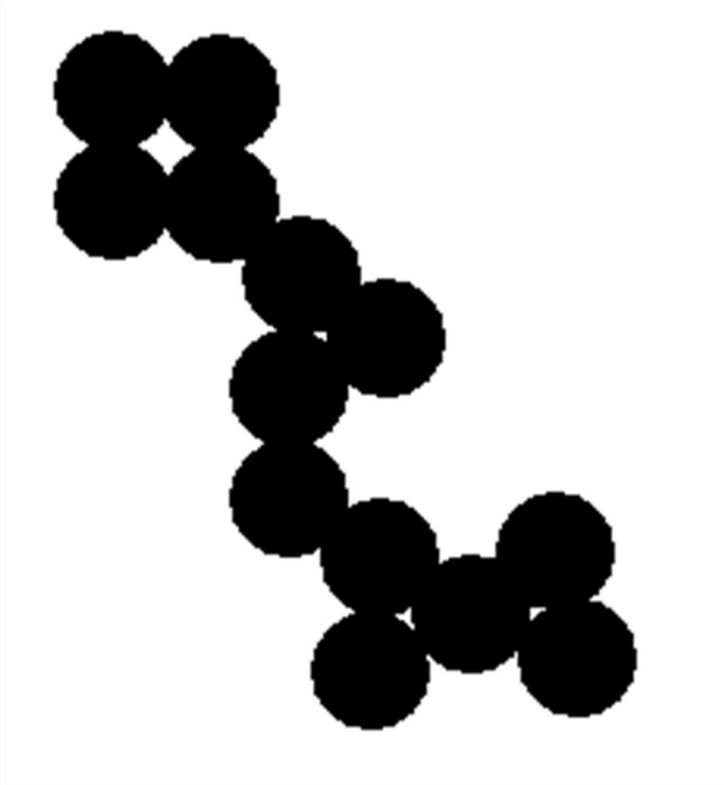
```
dd = -bwdist(~mask);
```

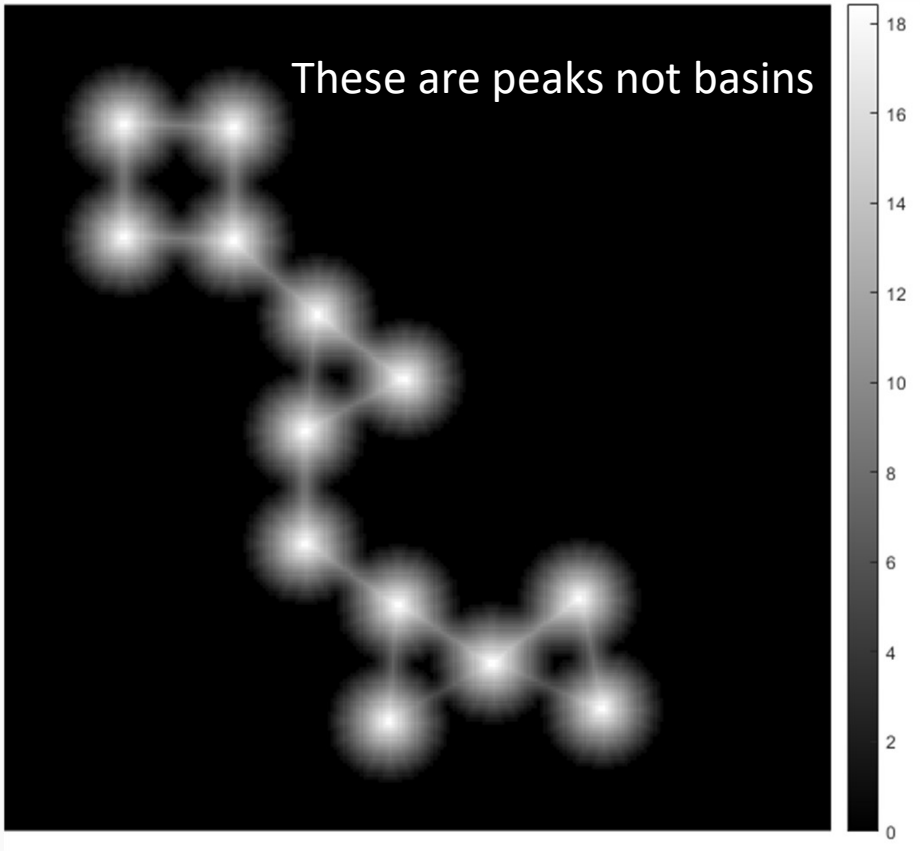




The distance transform `bwdist`

- `bwdist` computes the distance of a pixel to the nearest TRUE pixel

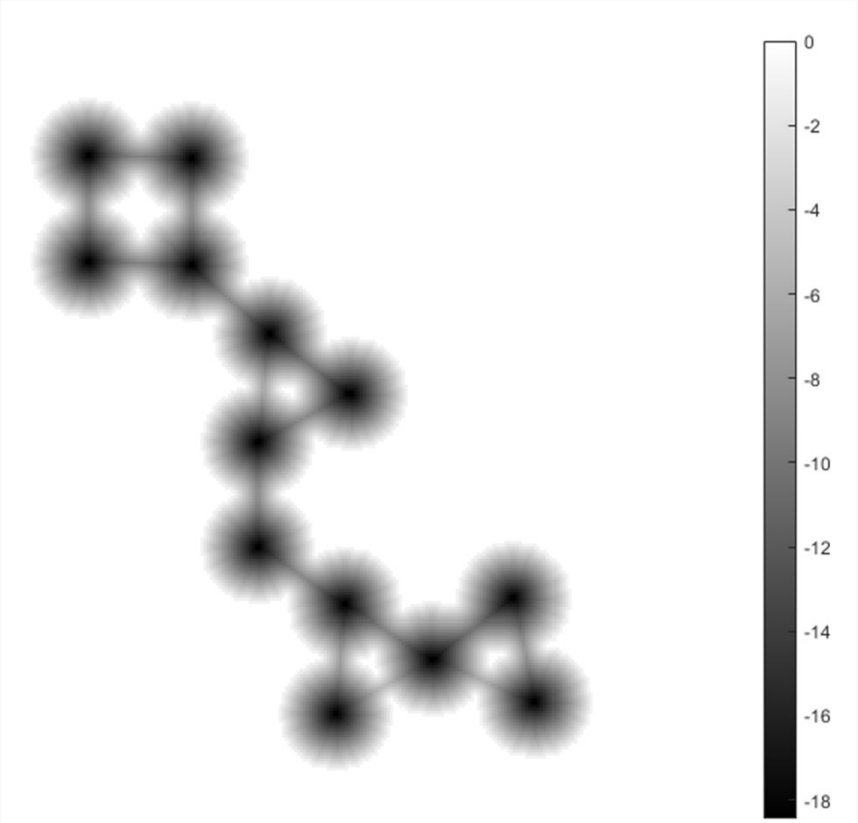


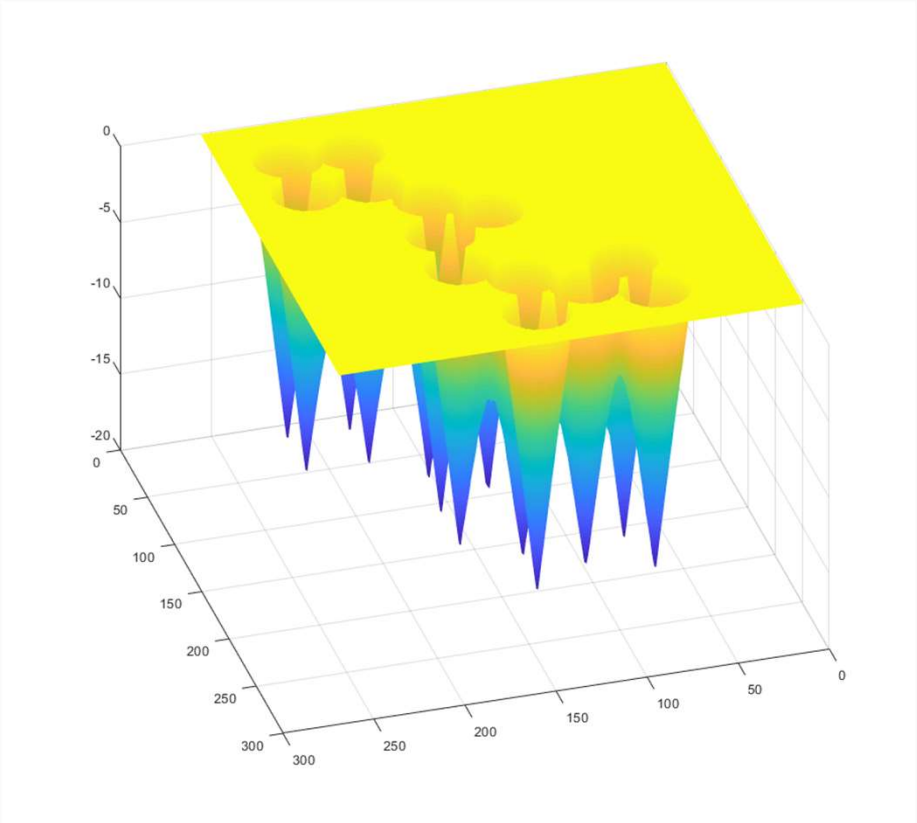


Compute the distance transform of the inverted mask

- We have to compute the (negative) distance transform of the inverse of the cell mask

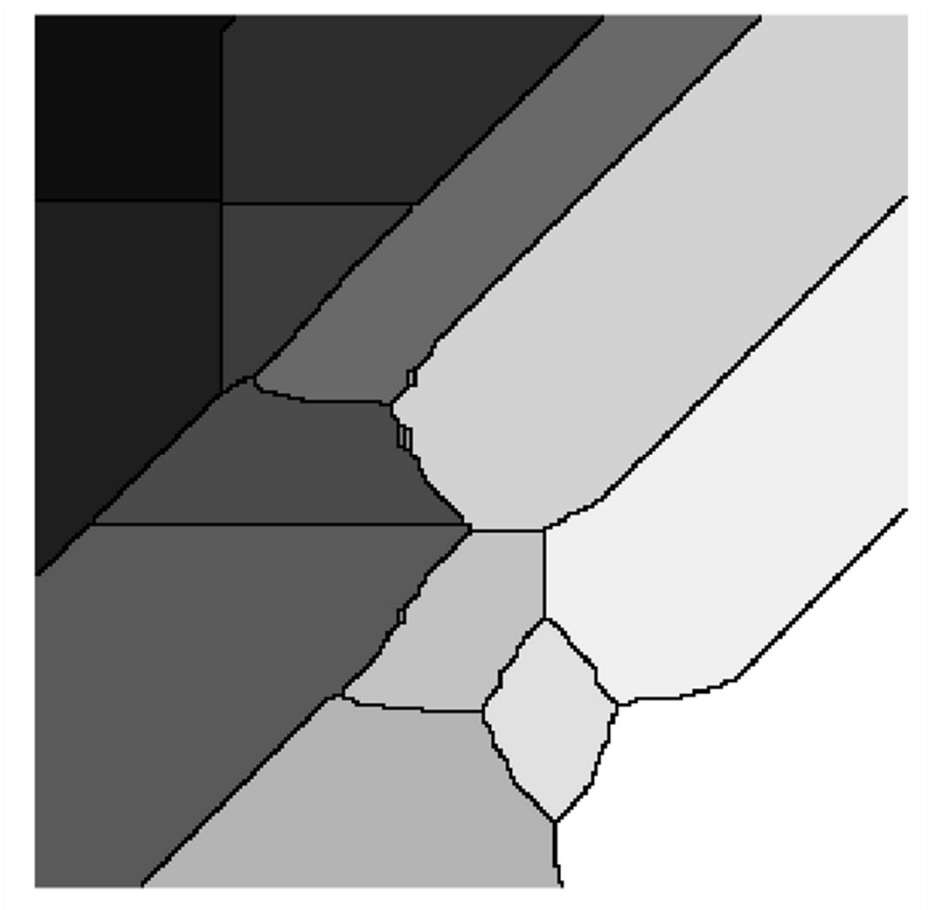
```
dd = -bwdist(~mask);
```





Run the watershed

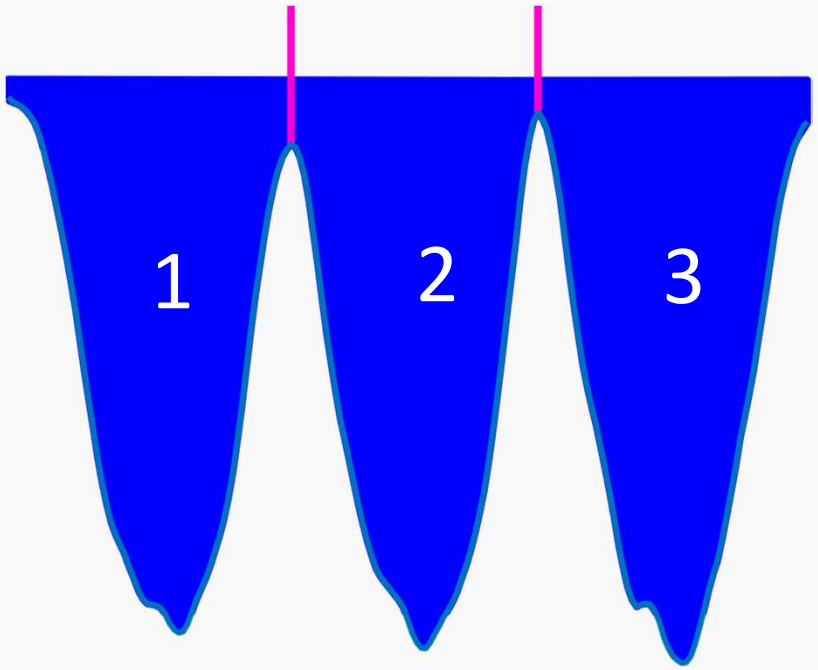
```
L = watershed(dd);
imshow(L, [])
```



L is a label matrix

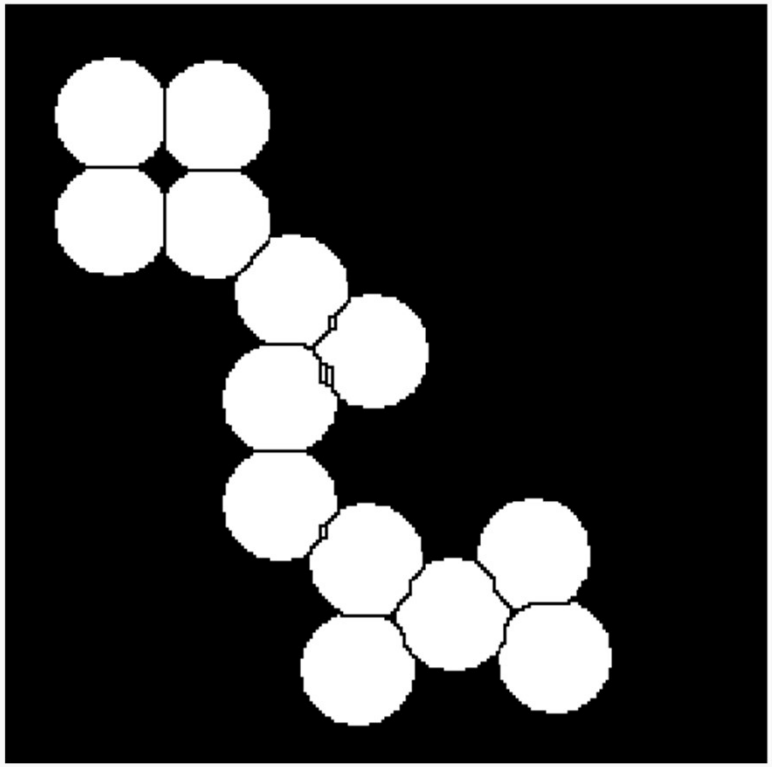
Individual objects are labeled 1, 2, 3…

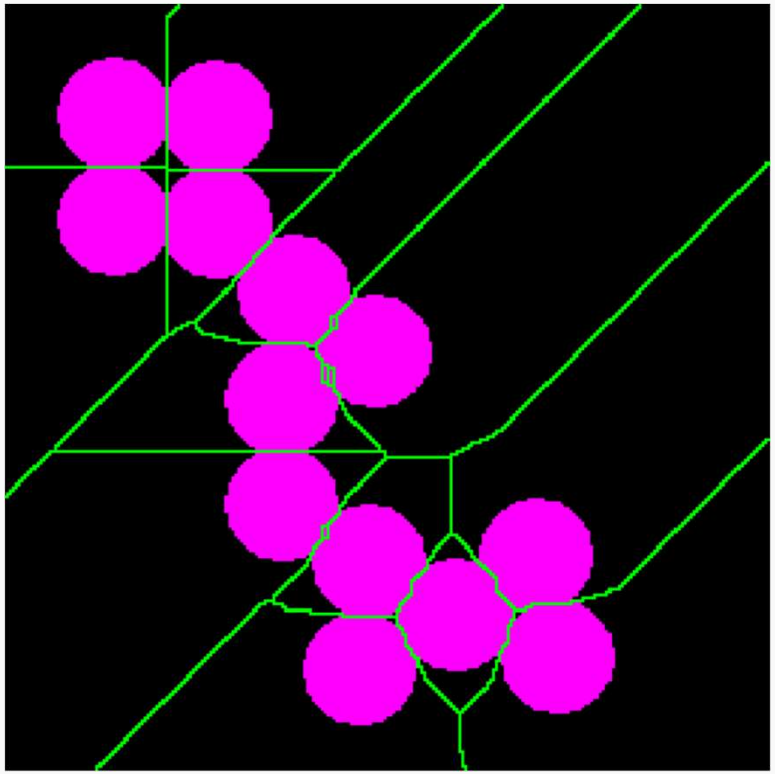
Ridgelines are labeled 0



Update the mask with the ridge lines

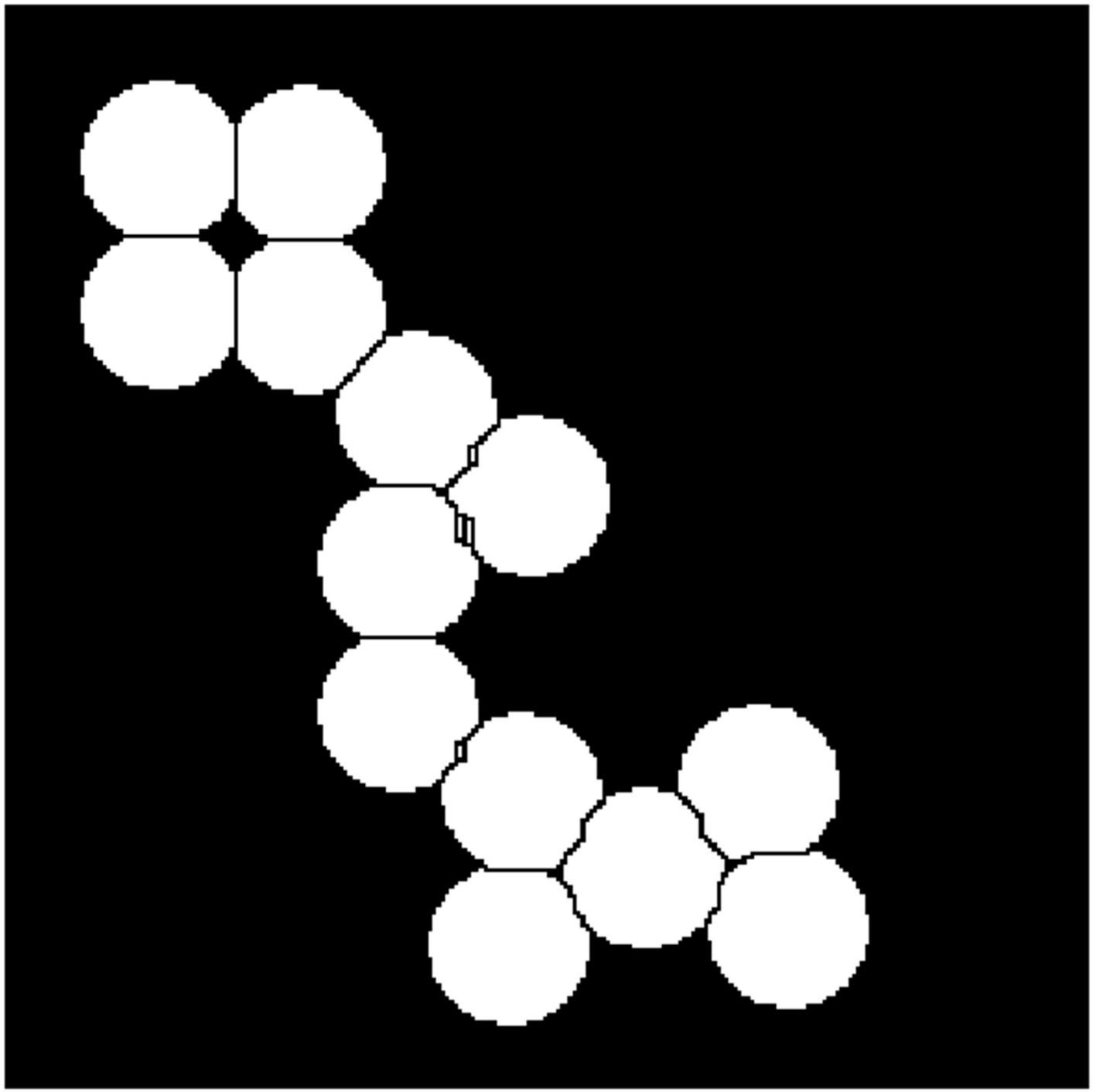
```
%Update the mask with the ridge lines
mask(L == 0) = false;
imshow(mask)
```



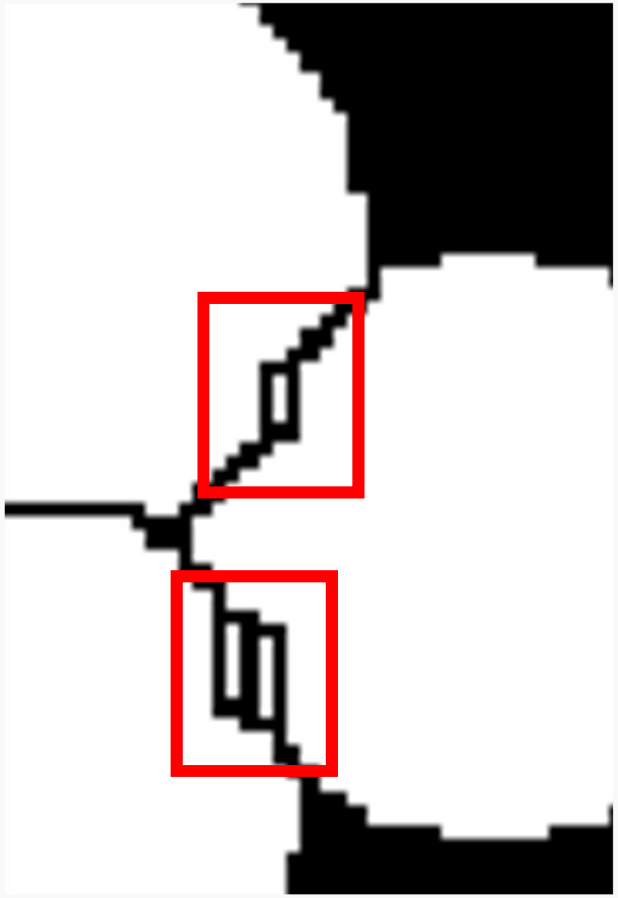


Full example watershed code

```
%Example image to represent mask of thresholded objects
mask = imread('circles.png');

%Compute the distance transform
dd = -bwdist(~ mask);

%Run the watershed
L = watershed(dd);

%Update the original mask
mask(L == 0) = false;

imshow(mask)
```



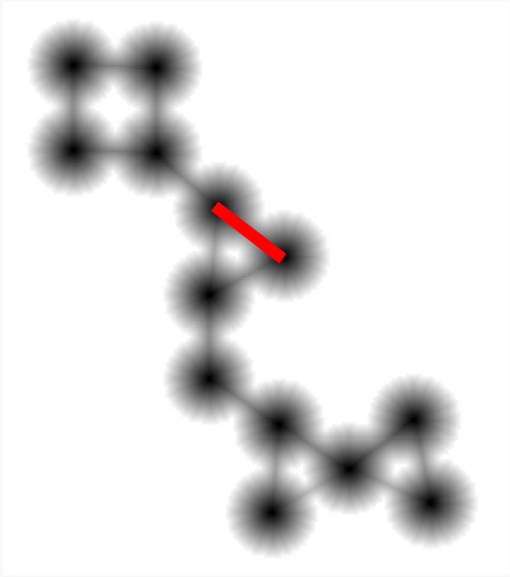
Tuning the watershed function

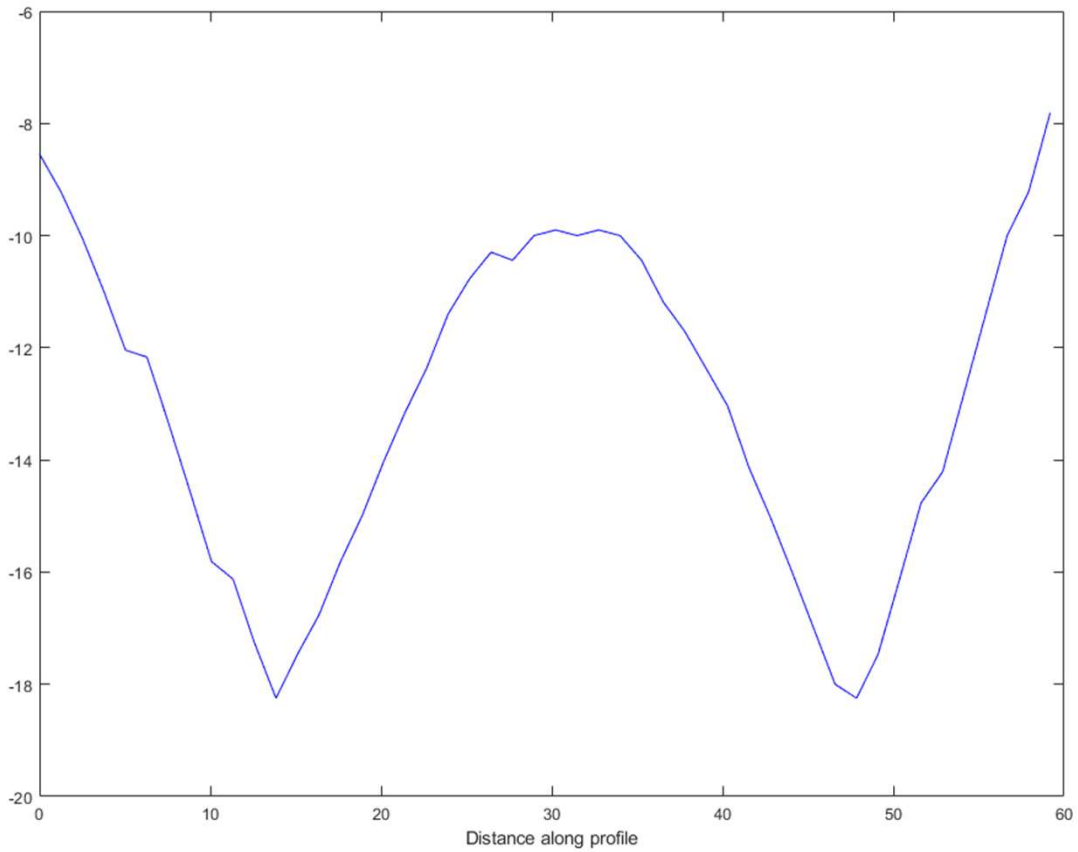


- A common problem with the watershed algorithm is **oversegmentation**

- Oversegmentation = a single object is broken up into multiple smaller objects

`improfile` to plot a cross-section of the distance transform



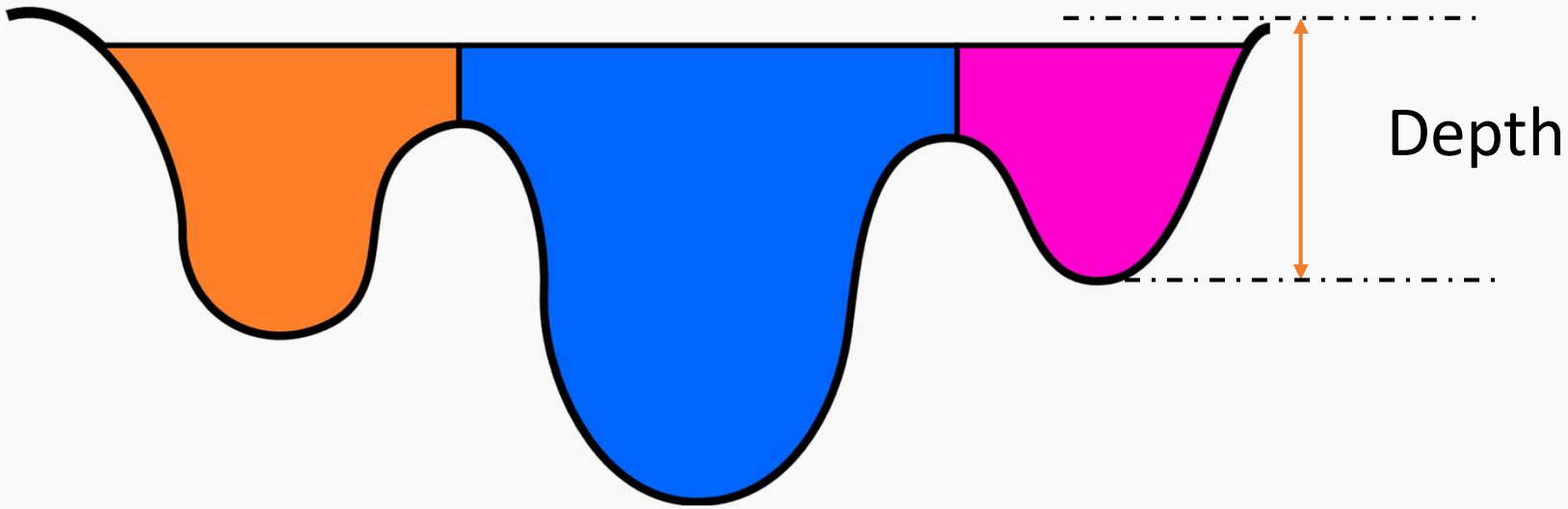


There should be ONE local minima per object

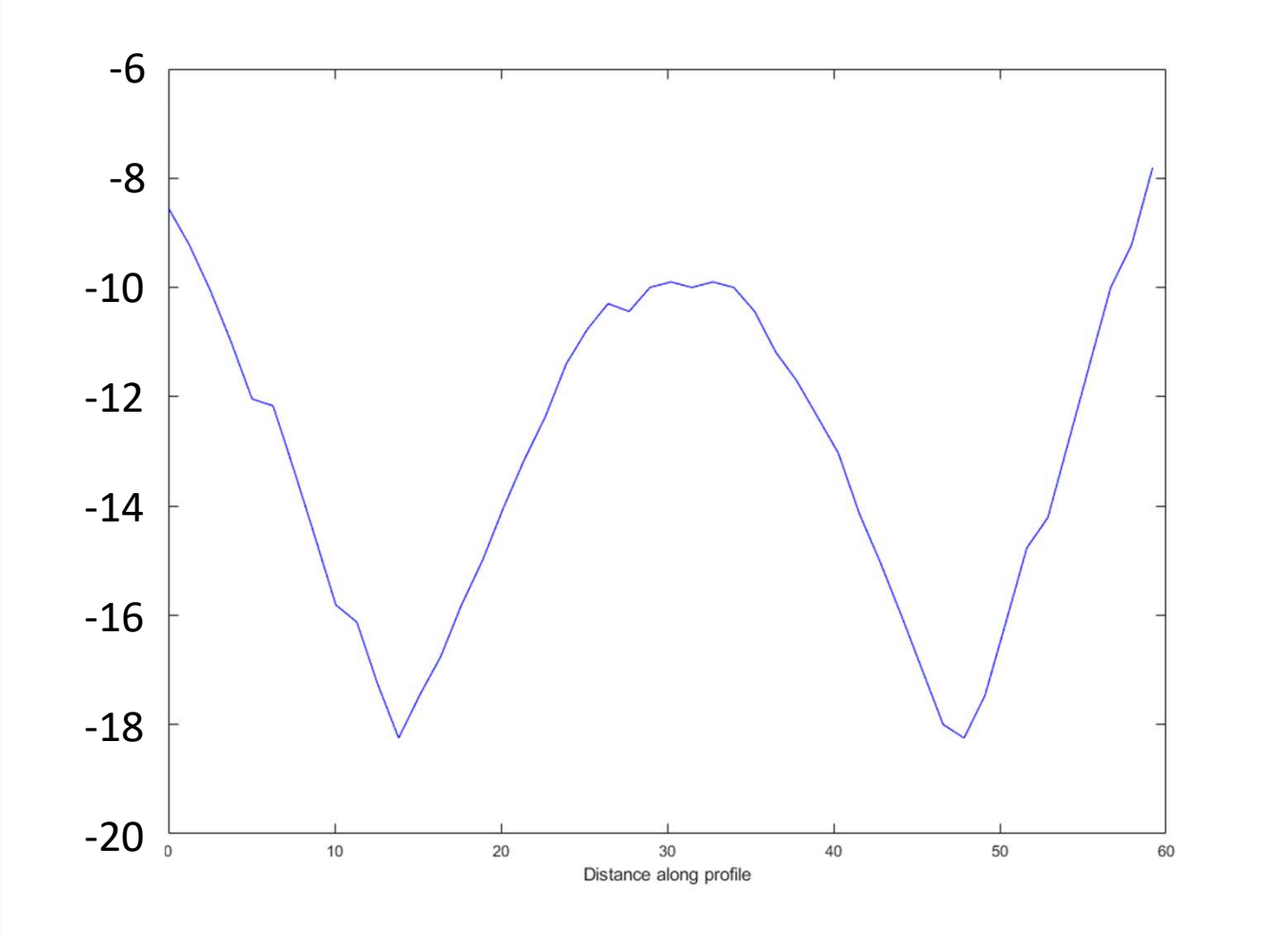
Use imhmin to suppress shallow minimas after the distance transform

- `dd = imhmin(dd, H)`

  suppresses minima with depth is less than `H`

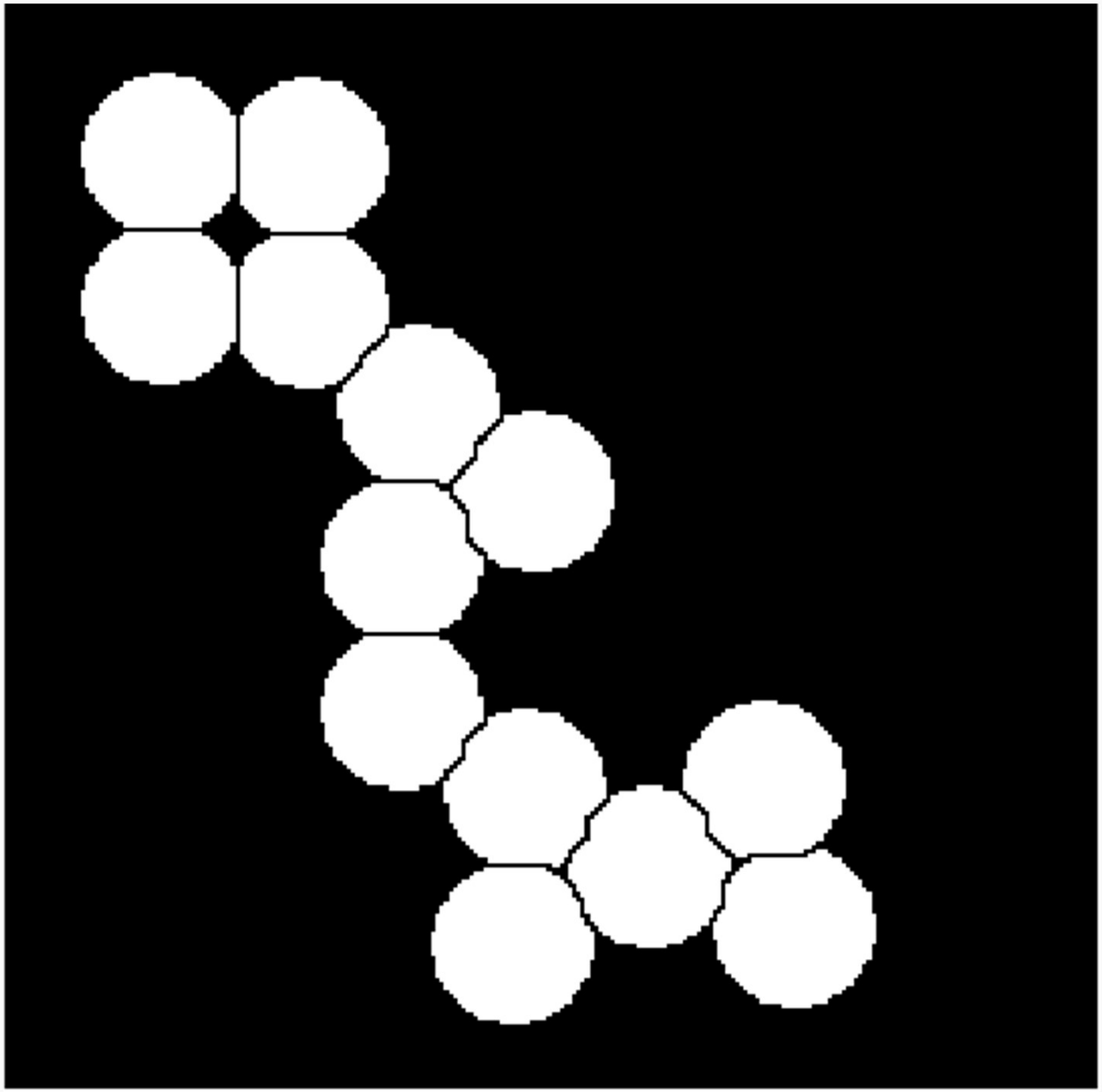


What minimum depth should we use?



Full example watershed code with imhmin

```
%Example image to represent mask of thresholded objects
mask = imread('circles.png');

%Compute the distance transform
dd = -bwdist(~mask);
dd = imhmin(dd, 5);

%Run the watershed
L = watershed(dd);

%Update the original mask
mask(L == 0) = false;

imshow(mask)
```



Implementation notes

- For the watershed algorithm to work properly, the input must have:
  - Intensity gradients within each object
  - Each object should be a "basin"
  - There should be only one local minima per object

Implementation notes

- For the distance transform to work properly, there must be SOME morphology that indicates where the objects should split

Challenges with the watershed algorithm

- Does not work very well when objects too close together
  - You could try increasing the threshold intensity

  - For those who want to refine look up "marker-controlled watershed" – happy to discuss this in office hours if interested (we won't cover this in class)

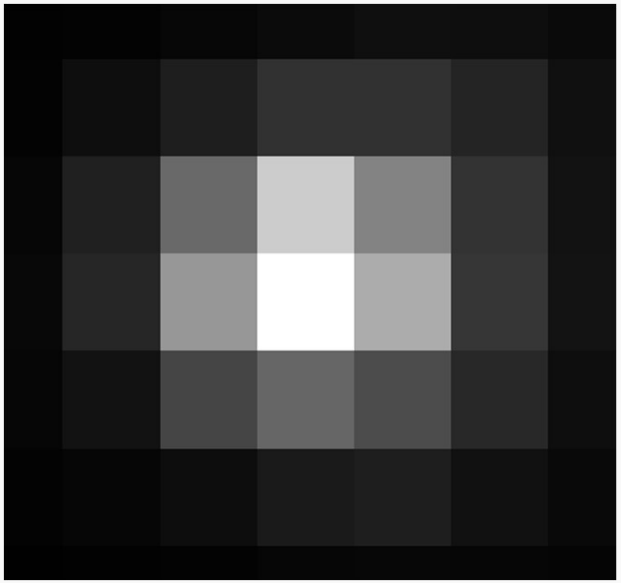
Challenges with the watershed algorithm

- Does not work very well when objects too close together
  - You could try increasing the threshold intensity

  - For those who want to refine look up "marker-controlled watershed" – we won't cover this in class but I'm happy to discuss this during office hours

Implementation notes

- The distance transform does not work well for objects that are too small (< 3-5 pixels) i.e. single particle/organelles
  - The distance transform will be very sensitive to noise/segmentation
  - One potential way to overcome this is to scale up the image (imresize)
  - Another way is to fit the data (we will do this in a couple of lectures)

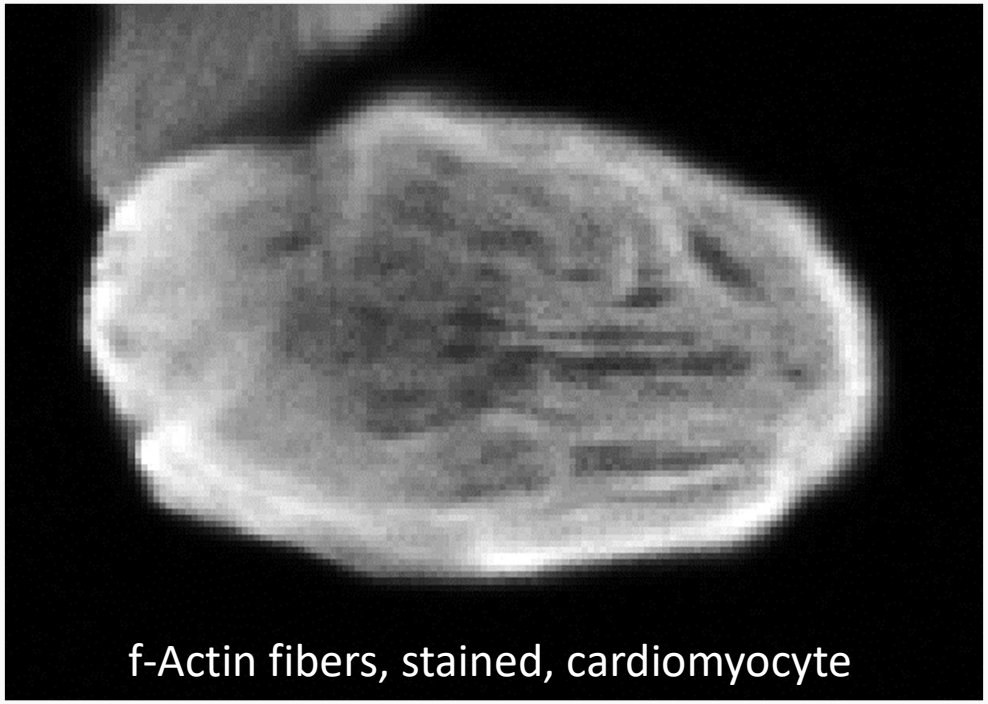


500 nm diameter fluorescent beads

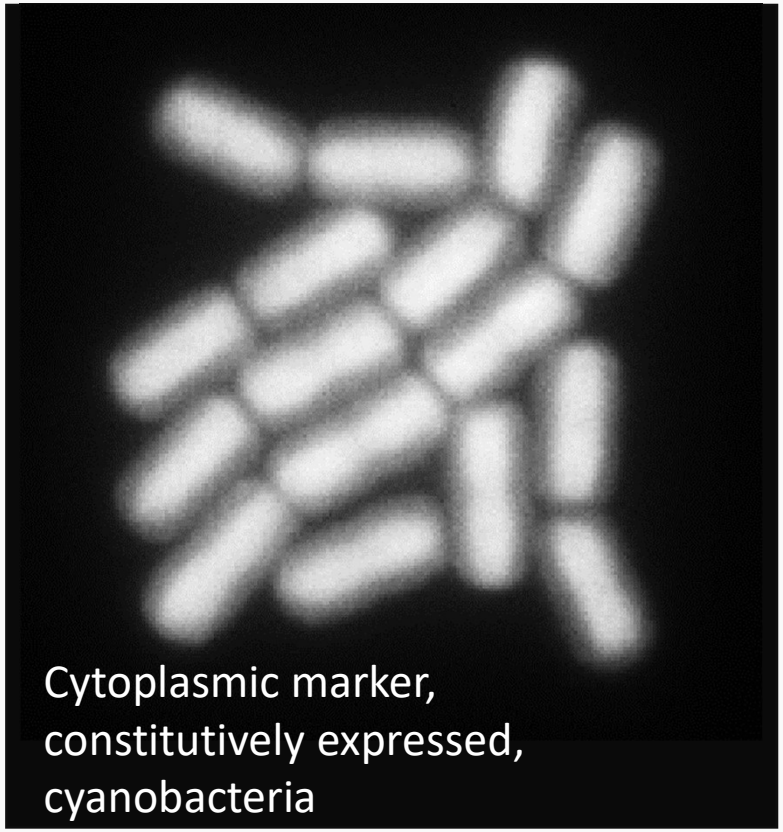
Implementation notes

- It is not always necessary (but usually easiest) to use a mask

- If the object intensity is uniform enough, you could use the `imcomplement` of the image to compute the distance transform



f-Actin fibers, stained, cardiomyocyte



Cytoplasmic marker, constitutively expressed, cyanobacteria