Image processing: Intro

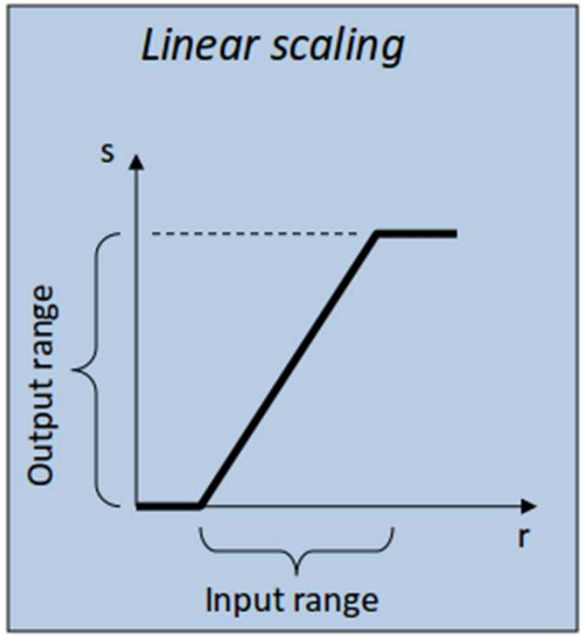
Francisco Gómez J
Computer Vision
MMS
U. Central and UJTL

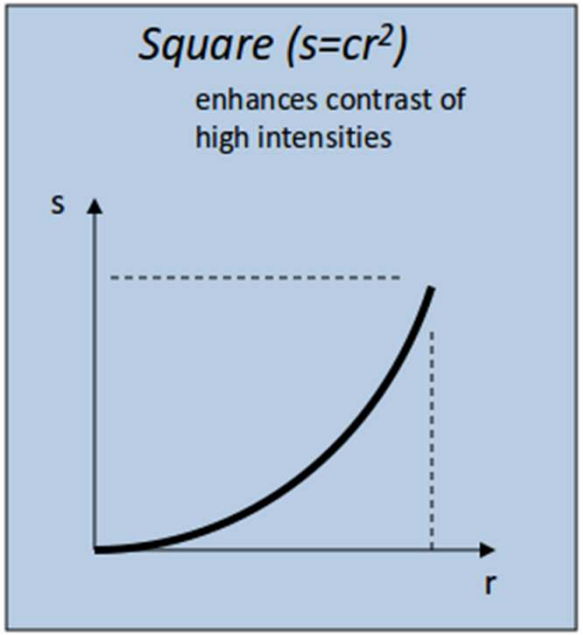
Image processing

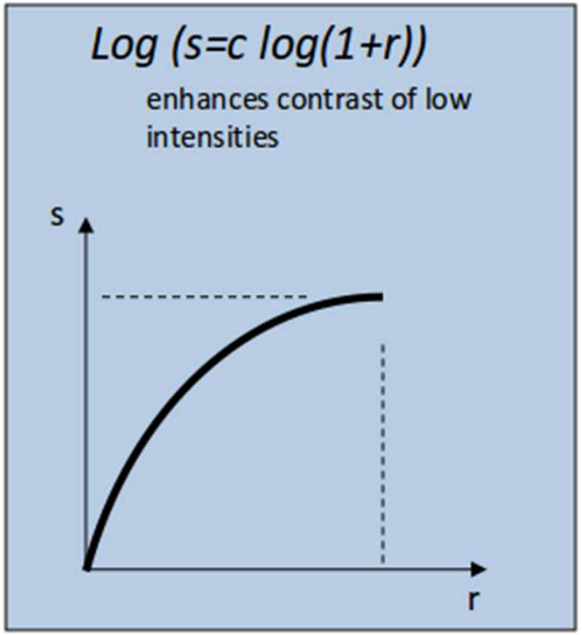
- An introduction to:
  - Reduce noise and enhace image quality
  - Detect features
- Topics:
  - Gray level (Point transformations)
  - Spatial (neighborhood) transforms
  - Binary image processing

Gray level transformations

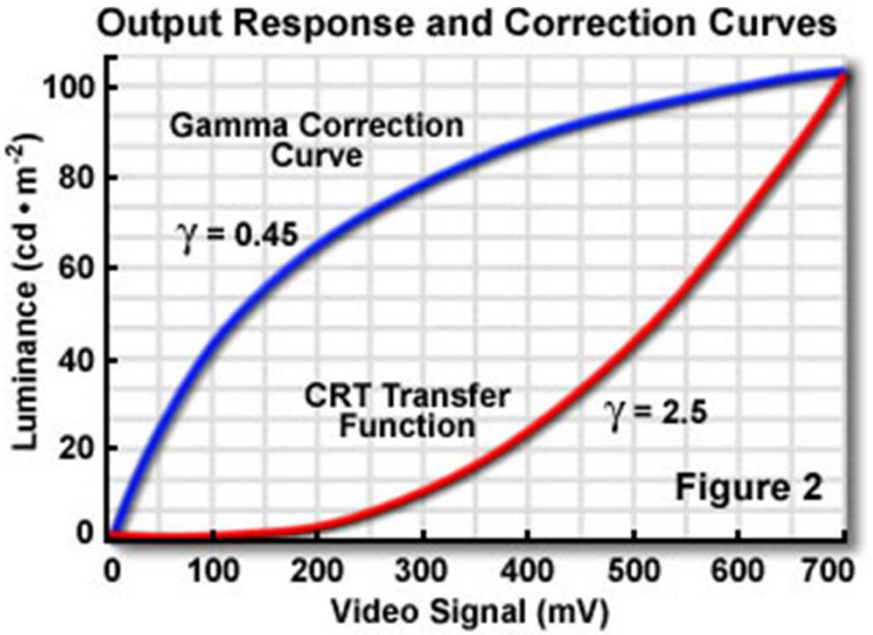
- Aim: to map color points to other color points:
  - *s=f(r)*
  - *This operation is a map from a pixel value r to a target value s*

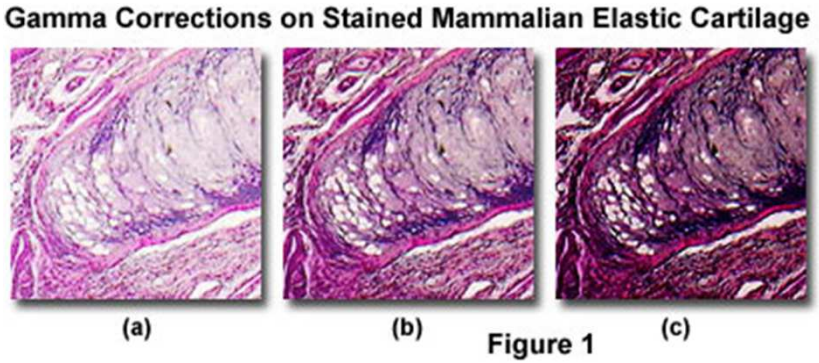






Example: gamma correction





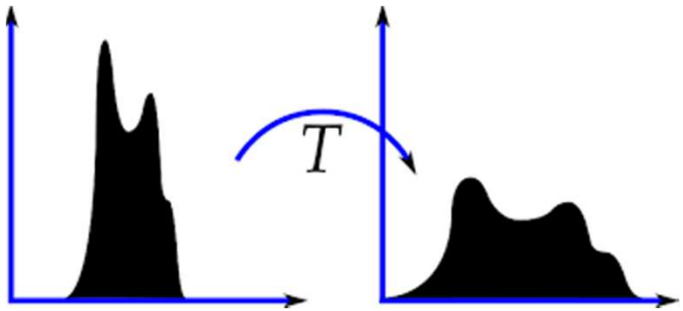
http://www.olympusmicro.com/primer/java/digitalimaging/processing/gamma/

Demo

- *I = imread('pout.tif'); % read the image*
- *imhist(I); % show the histogram*
- *imtool(I); % tool to perform constrast adjusts*

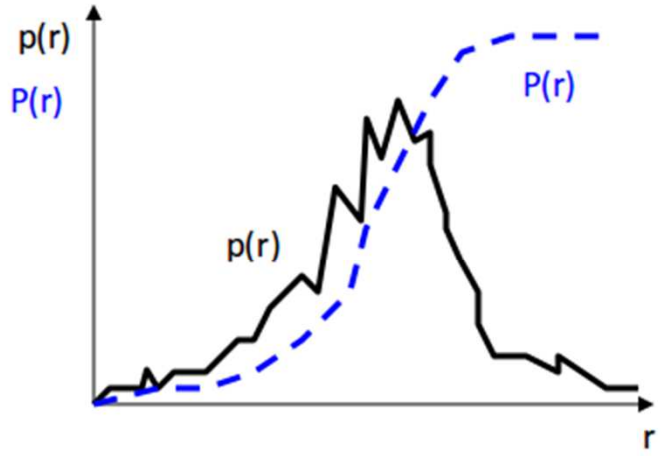
Histogram equalization

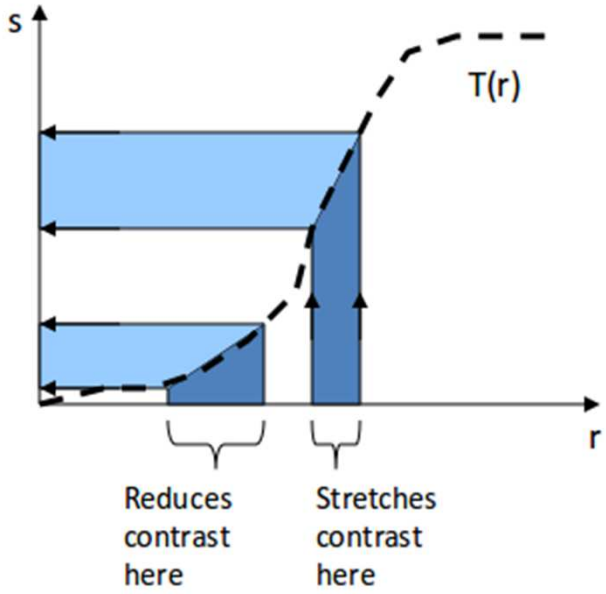
- The histogram is the probability distribution function of the colors in the image
- Equalize means flat the colors



- Large number of pixels → should be streched
- Small number of pixels → should be enlarged

Histogram equalization





The cummulative distribution function can be used as a mapping

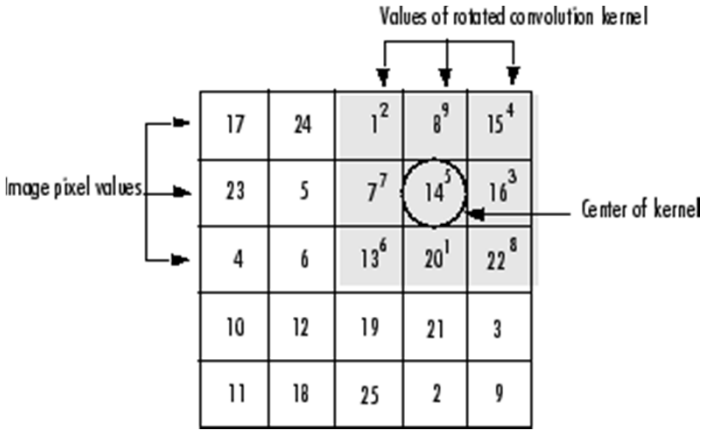
Demo

```
I = imread('liftingbody.png'); % read image
imshow(I); % show the image
[I2 T] = histeq(I); % perform the equalization
[I3] = adapthisteq(I); % apply to local neigborhoods
imshow(I); % show the image
```

Spatial filtering

- An image f, size M x N
- A mask w, size m x n
- Sum of the products of mask coefficients with corresponding pixels under mask.

- Slide the mask...



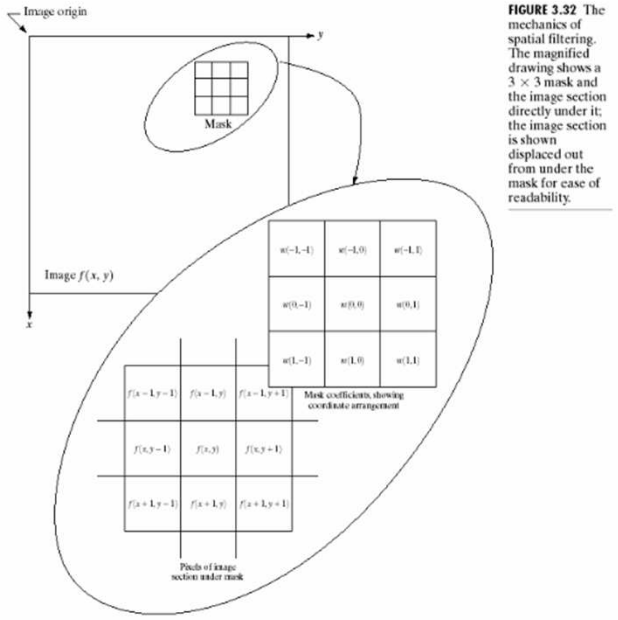
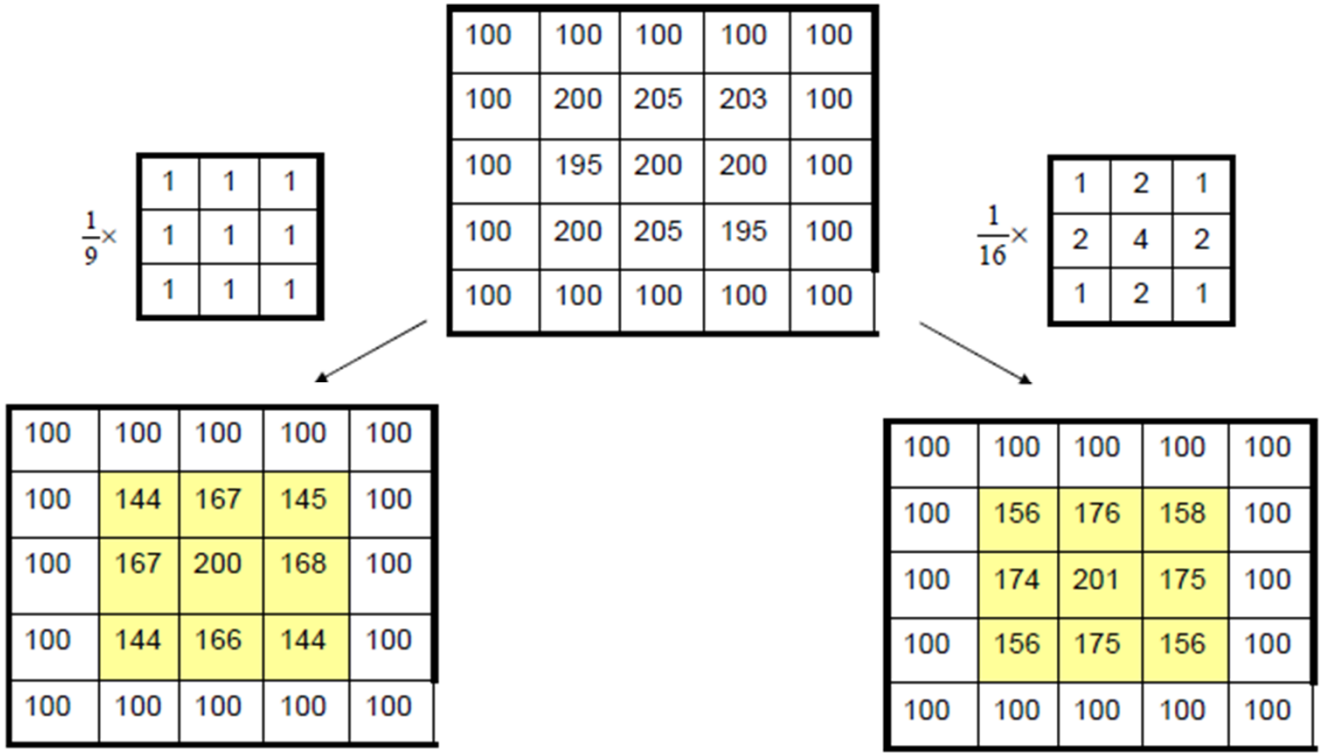


FIGURE 3.32 The mechanics of spatial filtering. The magnified drawing shows a 3×3 mask and the image section directly under it; the image section is shown displaced out from under the mask for ease of readability.

$$g(x,y) = \sum_{s=-m/2}^{m/2} \sum_{t=-n/2}^{n/2} w(s,t)\ f(x+s, y+t)$$

$$= w(x,y) \otimes f(x,y)$$

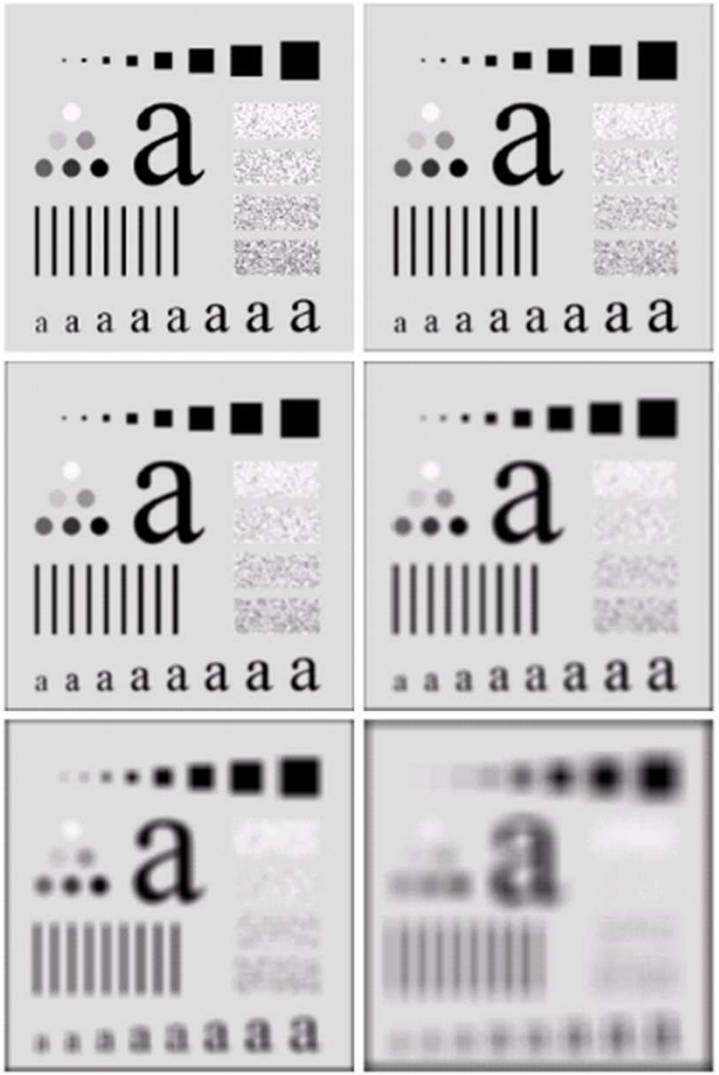
Example



http://eeweb.poly.edu/~yao/EE3414/image_filtering.pdf

This filter can be used to remove noise or generate blur

Example



a b
c d
e f

FIGURE 3.35 (a) Original image, of size 500 × 500 pixels. (b)–(f) Results of smoothing with square averaging filter masks of sizes $n = 3, 5, 9, 15$, and 35, respectively. The black squares at the top are of sizes 3, 5, 9, 15, 25, 35, 45, and 55 pixels, respectively; their borders are 25 pixels apart. The letters at the bottom range in size from 10 to 24 points, in increments of 2 points; the large letter at the top is 60 points. The vertical bars are 5 pixels wide and 100 pixels high; their separation is 20 pixels. The diameter of the circles is 25 pixels, and their borders are 15 pixels apart; their gray levels range from 0% to 100% black in increments of 20%. The background of the image is 10% black. The noisy rectangles are of size 50 × 120 pixels.

http://eeweb.poly.edu/~yao/EE3414/image_filtering.pdf

Demo

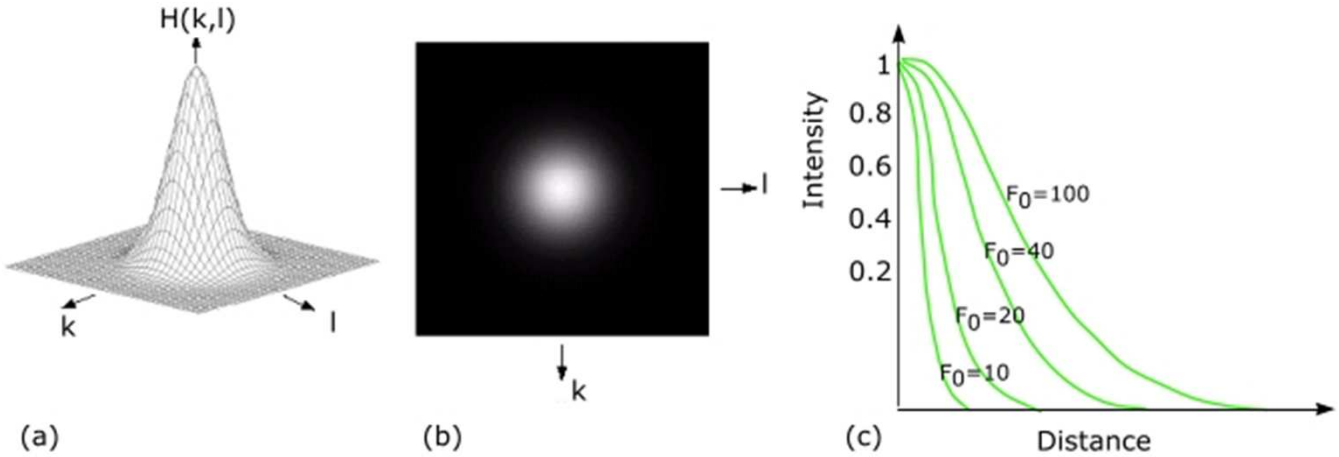
```
I = zeros(100,100); % create an image with an square
I(30:70,30:70) = 1;
figure
imshow(I,[]);
w = ones(3,3)/3; % create the mask
I2 = imfilter(I,w); % filter the image
figure
imshow(I2,[]);
w =fspecial('average', [15 15]); % create a mask
I2 = imfilter(I,w); % filter the image
figure
imshow(I2,[]);

I = I + 0.5 * rand(size(I));
figure
imshow(I,[]);
I2 = imfilter(I,w); % filter the image
figure
imshow(I2,[]);
```

Gaussian filtering

- Gaussian filter works better in the frequency domain

$$h(x,y) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/(2\sigma^2)}$$



(a) a plot of Gaussian function, (b) the inverse Fourier transform of Gaussian, (c) Frequ

http://mipav.cit.nih.gov/pubwiki/index.php/Filters_%28Frequency%29

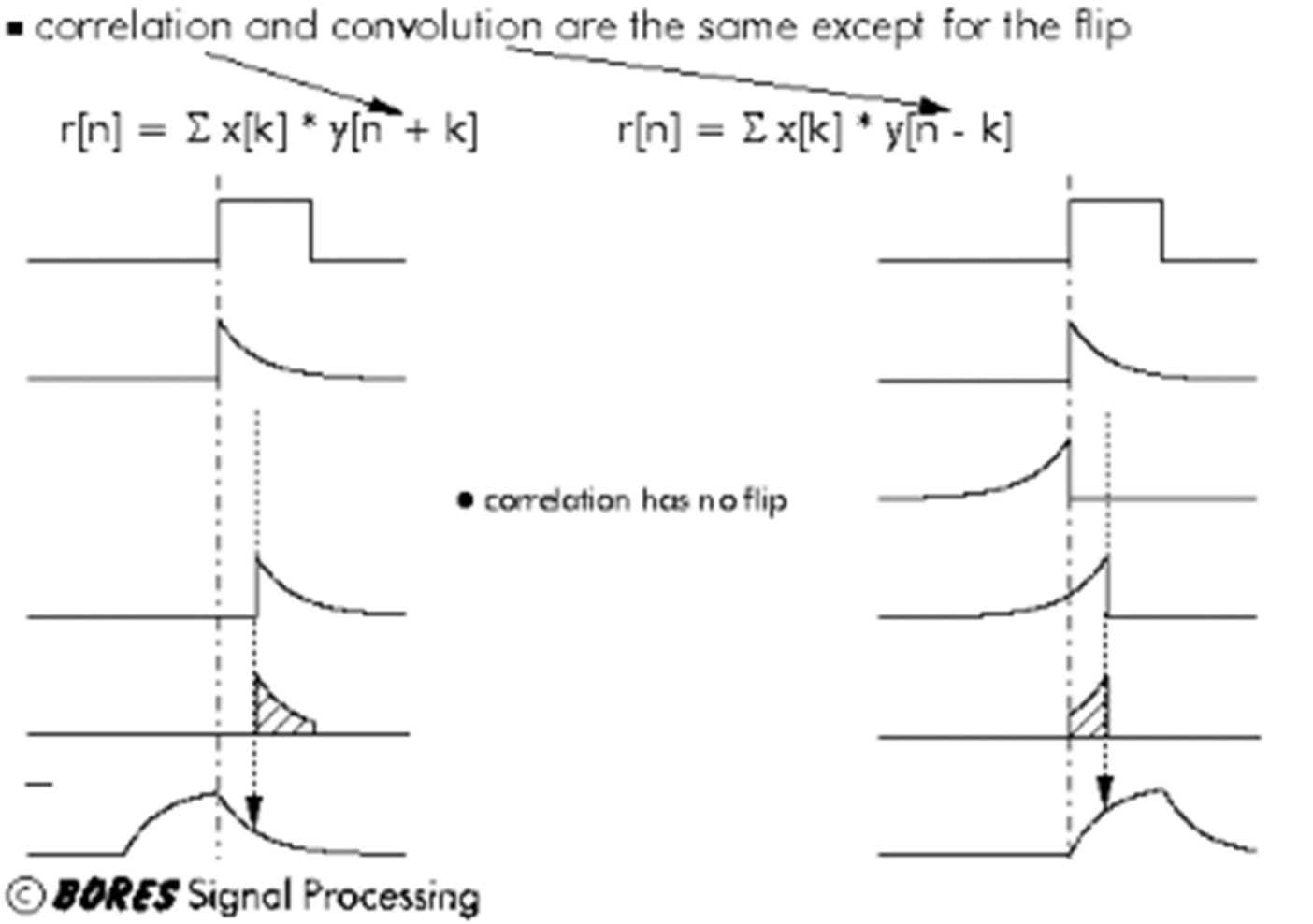
Demo

- w = fspecial('gaussian',10);

noisy lena

Gaussian filter

Convolution vs Correlation



http://www.bores.com/courses/intro/time/2_concor.htm

Sharpening spatial filtering

- First derivative

$$\frac{\partial f}{\partial x} \approx f(x+1) - f(x)$$

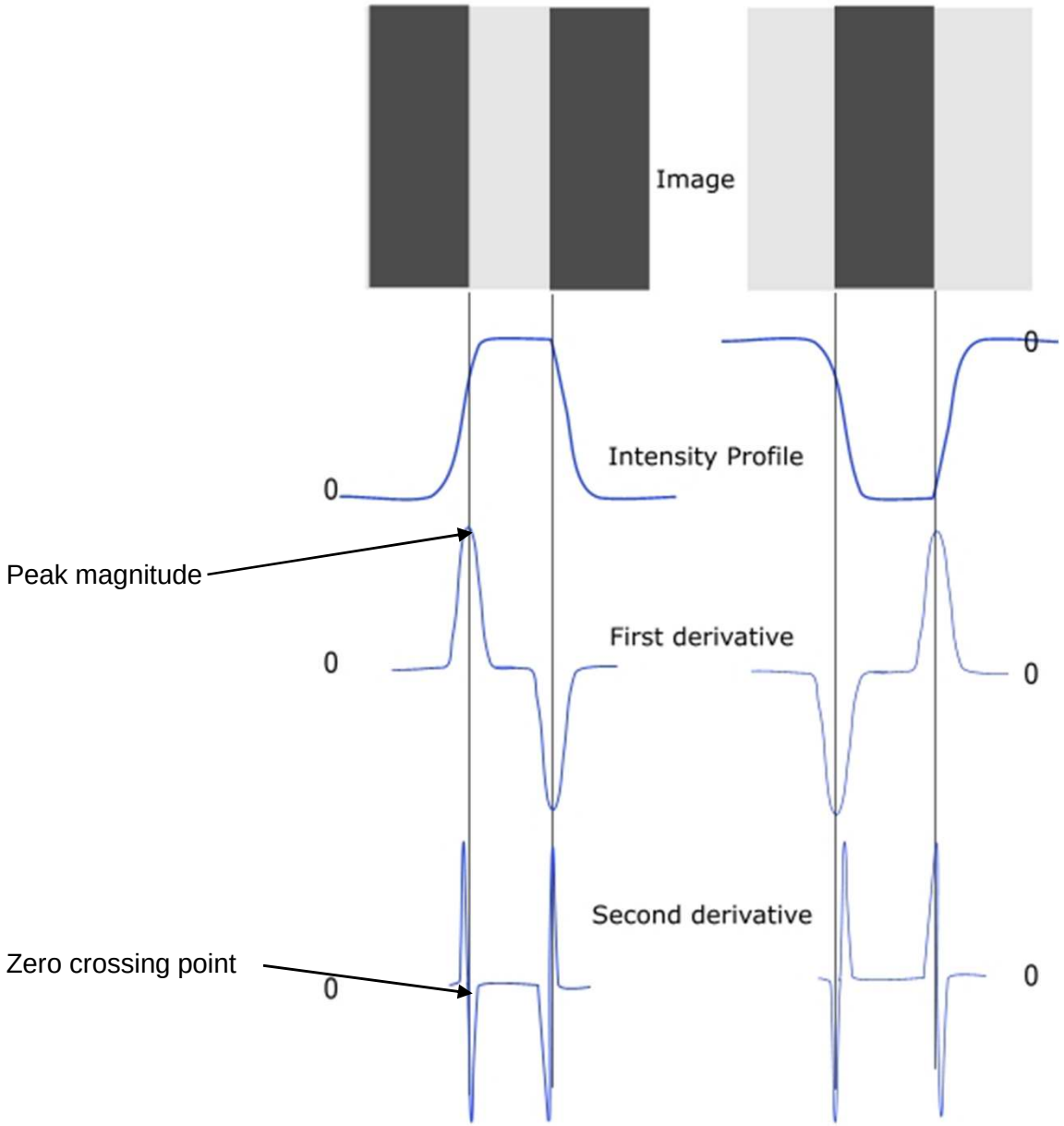
-1	+1

- Second derivative

$$\frac{\partial^2 f}{\partial x^2} \approx f(x+1) - 2f(x) + f(x-1)$$

+1	-2	+1

Peak detection



Edge operators

$$G_x = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \qquad G_y = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Sobel Horizontal

Sobel Vertical

Sobel Both Directions

Demo

- *Moon.tif*

- *hx= [-1 0 1;-2 0 2;-1 0 1];*
- *hy=hx';*

Image filtering

- Gradient

$$\nabla \mathbf{f} = \begin{bmatrix} \partial f / \partial x \\ \partial f / \partial y \end{bmatrix}$$

- Magnitude

$$|\nabla \mathbf{f}| = \left[\left(\frac{\partial f}{\partial x} \right)^2 + \left(\frac{\partial f}{\partial y} \right)^2 \right]^{1/2}$$

- Angles

$$\theta = \tan^{-1} \left(\frac{\partial f}{\partial y} \Big/ \frac{\partial f}{\partial x} \right)$$

Demo

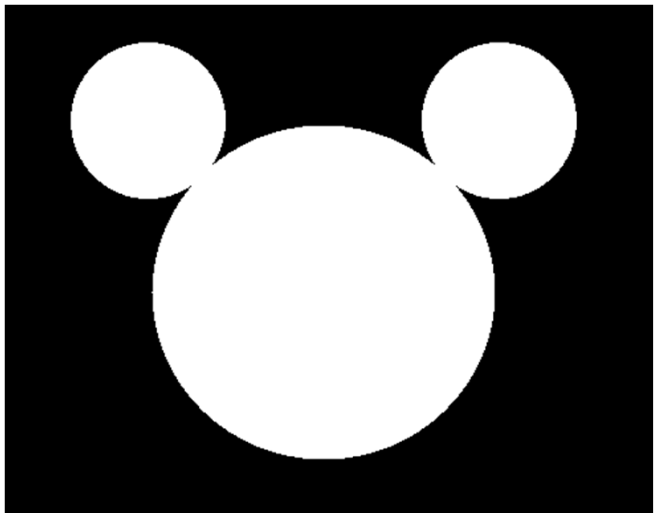
- *Dx = imfilter(I,hx)*
- *Dy = imfilter(I,hy)*
- *Compute the gradient magnitude*
- *atan2(Dy,Dx)*
- *colormap jet*
- *colorbar*

Excersices

- *Improve the quality of 'forest.tif'*
- *Read the “coins.png” image. Show the denoised gradient angle*

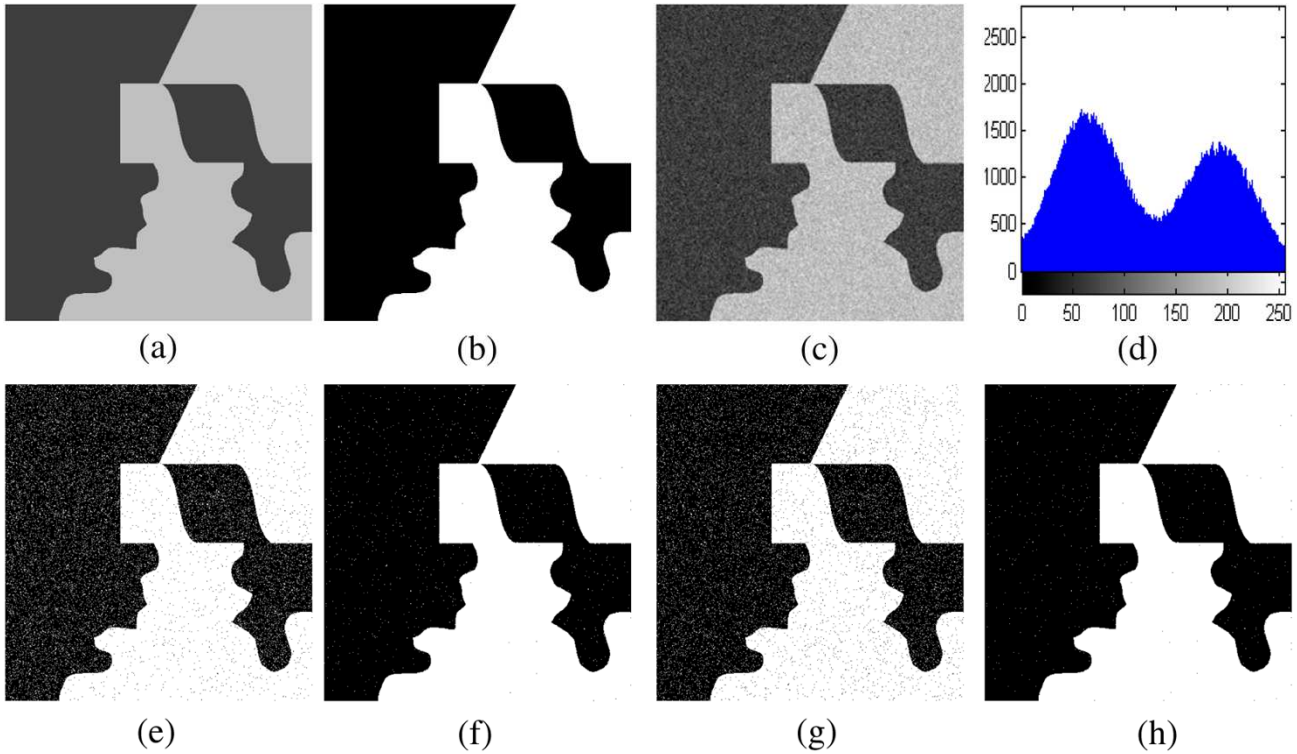
Binary image processing

- Binary images are composes by 0 (false) and 1 (true)
- Tipically obtained by using thresholding operations or some feature detection process
- Tipically we are interested in measure some properties



Thresholding

- Converting gray images to 0s and 1s



Otsu thresholding

- Find the threshold that minimize the weighted whitin-class variance

$$\sigma^2_{\text{Within}}(T) = n_B(T)\sigma^2_B(T) + n_O(T)\sigma^2_O(T)$$

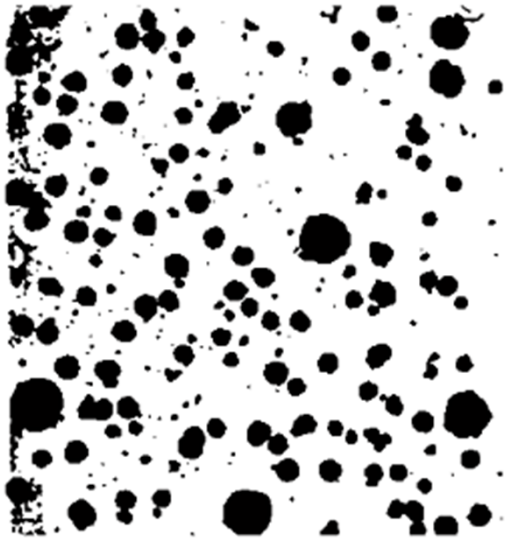
where

$$\begin{aligned}
n_B(T) &= \sum_{i=0}^{T-1} p(i) \\
n_O(T) &= \sum_{i=T}^{N-1} p(i) \\
\sigma^2_B(T) &= \text{the variance of the pixels in the background (below threshold)} \\
\sigma^2_O(T) &= \text{the variance of the pixels in the foreground (above threshold)}
\end{aligned}$$

- This quantity can be computed for all the possible thresholds

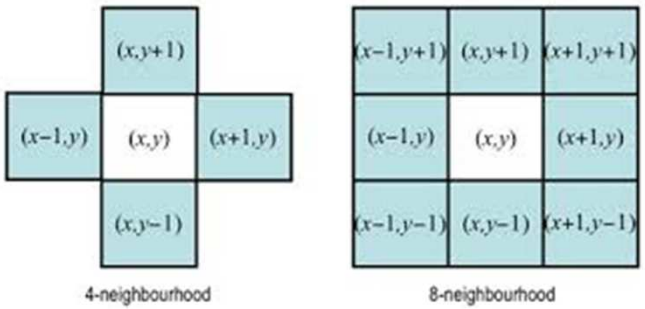
Demo

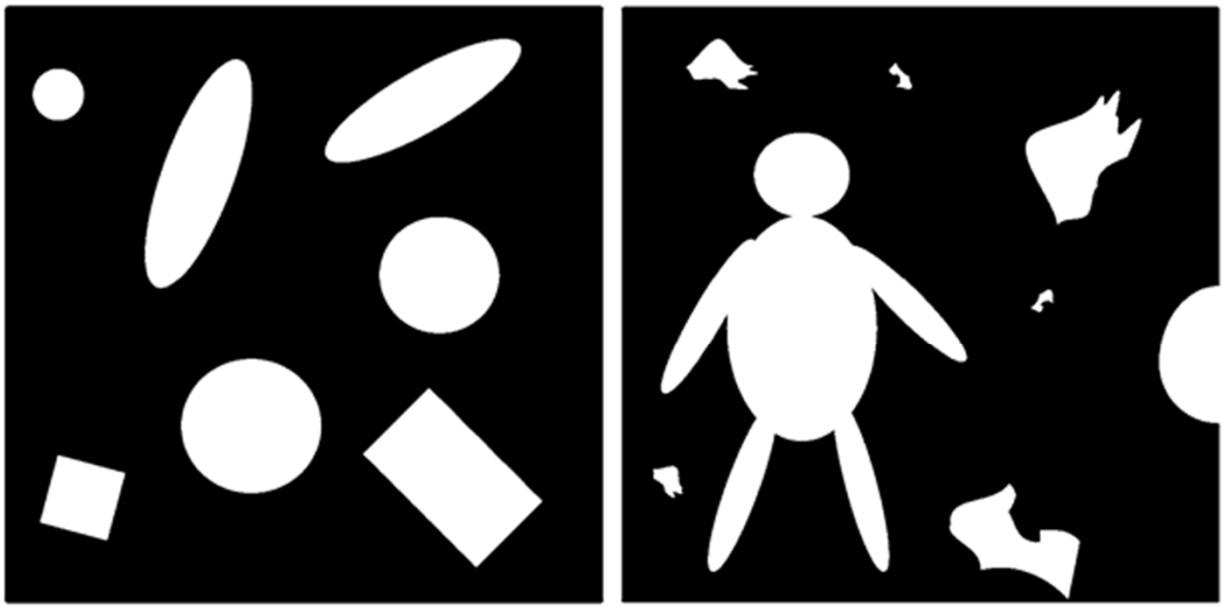
- 'nodules.tif'
- graythresh % peforms the Otsu thresholding



Connected components

- A blob exists if there is a path between all the pair of points
- Labeling is assigning numbers to any connected component





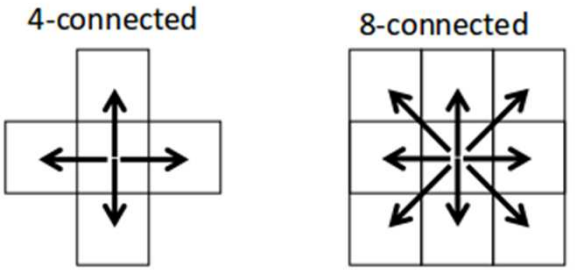
(a) (b)

Example

- How many components there are?

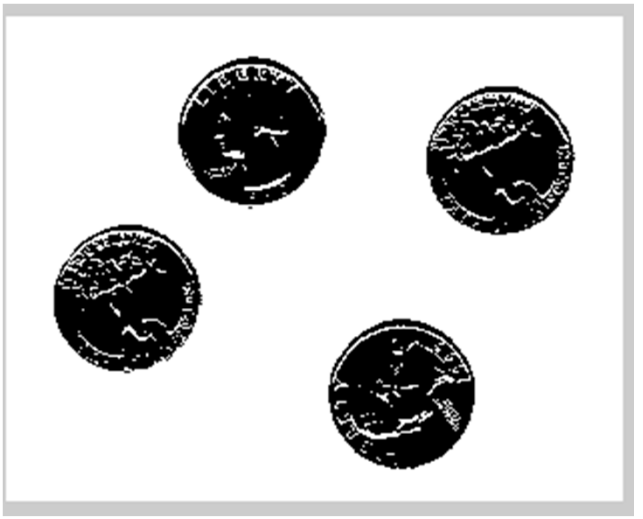
	1	1				
	1	1				
			1	1		
	1		1	1		
	1		1			1

Binary image



Demo

- *bwlabel % component labeling*
- *im2bw % perform*
- *label2rgb*
- *“eight.tif”*
- *Plot the histogram*
- *Extract the coins*



Digital Morphology: Dilation
Expand Regions

$$B \oplus S = \bigcup_{b \in B} S_b$$

Sweep S over B

Everywhere the origin of S touches a 1, OR S with B

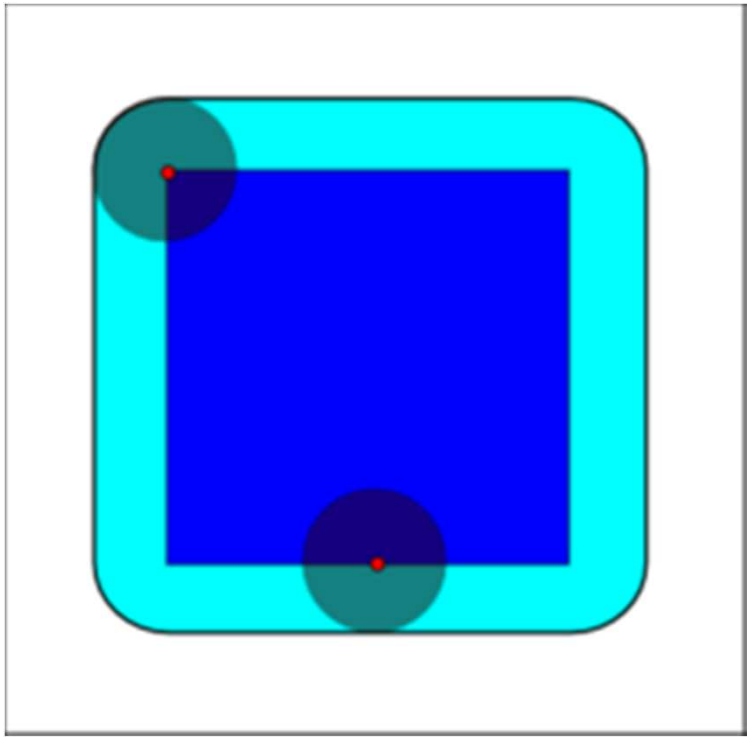
S_b is the structuring element S, shifted to b

1	1	1
1	1	1
1	1	1

S

		1		1	
		1	1		

B



Digital Morphology: Dilation

a c
b

FIGURE 9.5
(a) Sample text of poor resolution with broken characters (magnified view). (b) Structuring element. (c) Dilation of (a) by (b). Broken segments were joined.

0	1	0
1	1	1
0	1	0

Digital Morphology: Erosion
Shirnks regions

$$B \Theta S = \{ b \mid b + s \in B, \forall s \in S \}$$

Sweep S over B

Everywhere S is completely contained in B, output a 1 at the origin of S

1	1	1
1	1	1
1	1	1

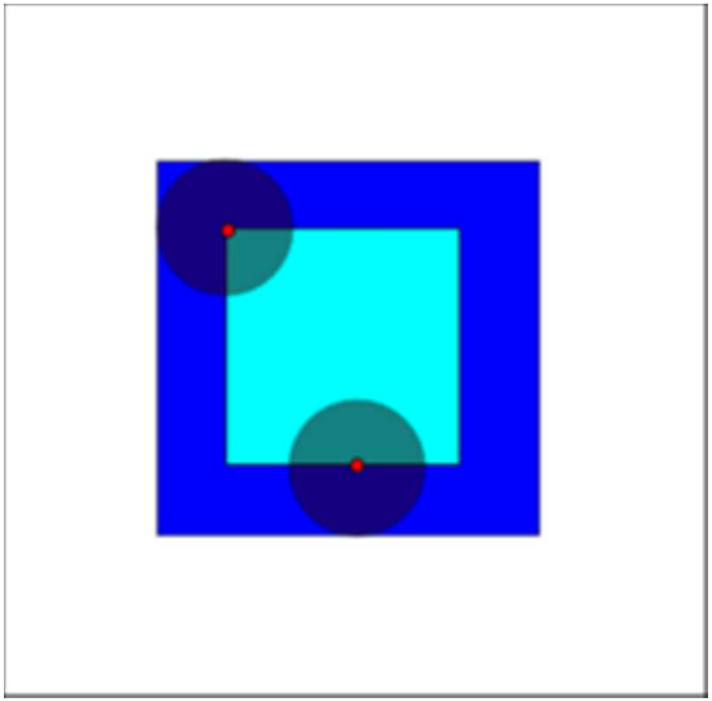
S

	1	1	1	1	1
	1	1	1	1	1
	1	1	1	1	1
	1	1	1	1	

B

Sweep S over B

Everywhere the origin of S touches a 1, OR S with B



Demo

- 'text.png'
- *S=strel(‘disk’,5) % create an structural element*
- I2 = imdilate(I,S);
- I2 = imerode(I,S);

Opening and closing

- Opening
  - Erosion followed by dilation
  - Eliminate small regions

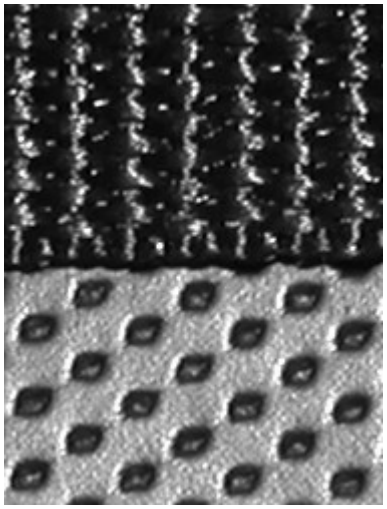
$$B \circ S = (B \ominus S) \oplus S$$

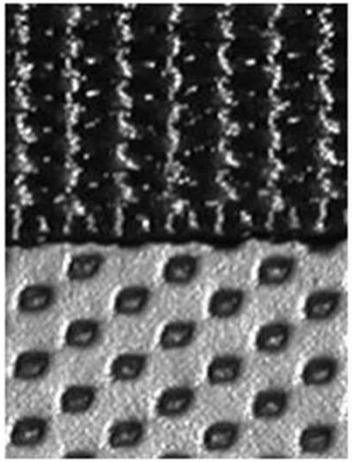
- Closing
  - Dilation followed by erosion
  - Fill small holes and gaps

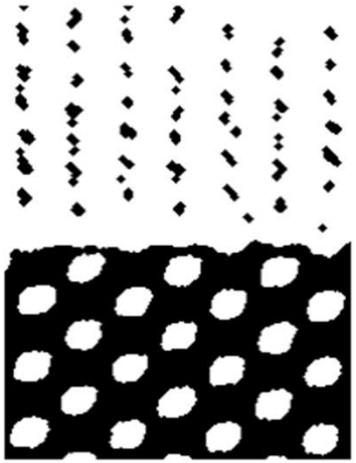
$$B \bullet S = (B \oplus S) \ominus S$$

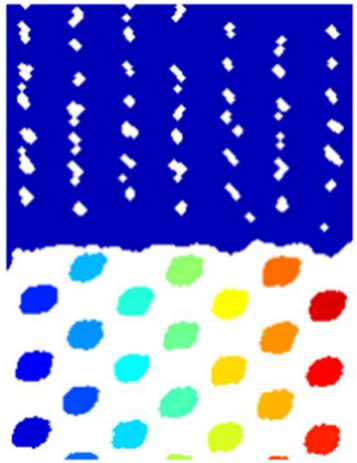
Demo

- Find the number of rounded objects and their number of pixels (regionprops)









Region properties

Area $$A = \sum_{(r,c)\in R} 1$$

Centroid $$\bar{r} = \frac{1}{A}\sum_{(r,c)\in R} r, \quad \bar{c} = \frac{1}{A}\sum_{(r,c)\in R} c$$

Bounding box

- The smallest rectangle containing the region
- Can be specified by
  - The location of the upper left corner
  - The width and height

In matlab *regionprops(L)*, L a labeled image produced by *bwlabel*