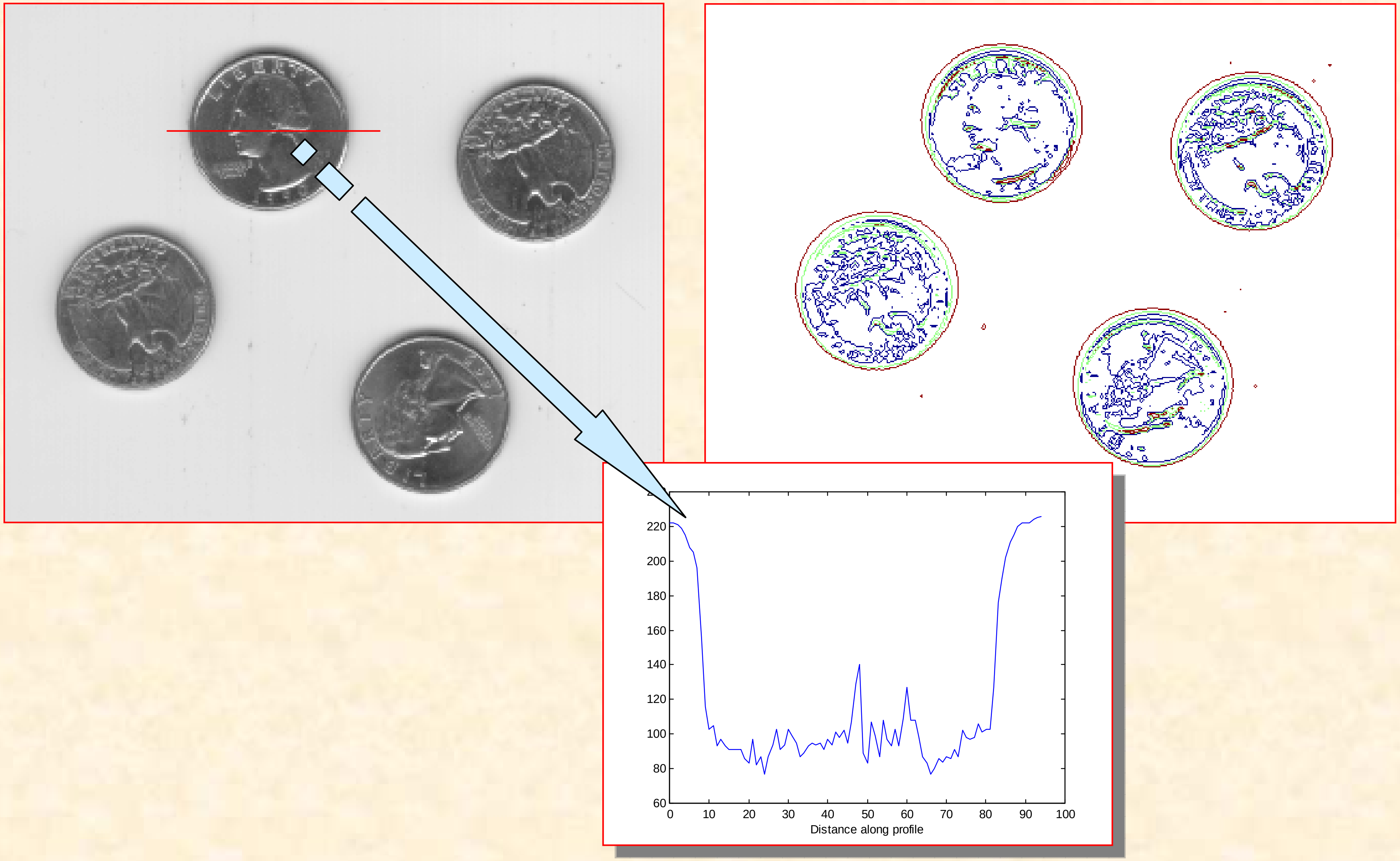
Boundary based segmentation (edge detection)

Changes (or discontinuous) in an image amplitude are important primitive characteristics of an image that carry information about object borders.

Detection methods of image discontinuities are principal approaches to image segmentation and identification of objets in a scene.

Local discontinuities in image intensity fall into three categories: ***points***, ***lines***, and ***edges***.

Segmentation by edge detection



Point and line detection

The most common way to look for an arbitrary image pattern (e.g., point, or edge) is to convolve the image with a **mask** of size $N_1 \times N_2$ (e.g., 3×3, 5×5).

The size of the mask and its content depends on the type of the detected object.

W_1	W_2	W_3
W_4	W_5	W_6
W_7	W_8	W_9

Detection masks

Vector inner product of the mask coefficients with image gray levels covered by the mask :

$$D = w_1 z_1 + w_2 z_2 + \cdots + w_N z_N = \sum_{i=1}^{N} w_i z_i = w^T z$$

W_1	W_2	W_3
W_4	W_5	W_6
W_7	W_8	W_9

A 3×3 detection mask

Point detection mask

Point detection mask:

-1	-1	-1
-1	8	-1
-1	-1	-1

The point is rendered if:

$$\boxed{|D| > T}$$

where D is a similarity measure between the image and the template, and T is a non-negative threshold.

Line detection masks

-1	-1	-1
2	2	2
-1	-1	-1

Horizontal

-1	2	-1
-1	2	-1
-1	2	-1

Vertical

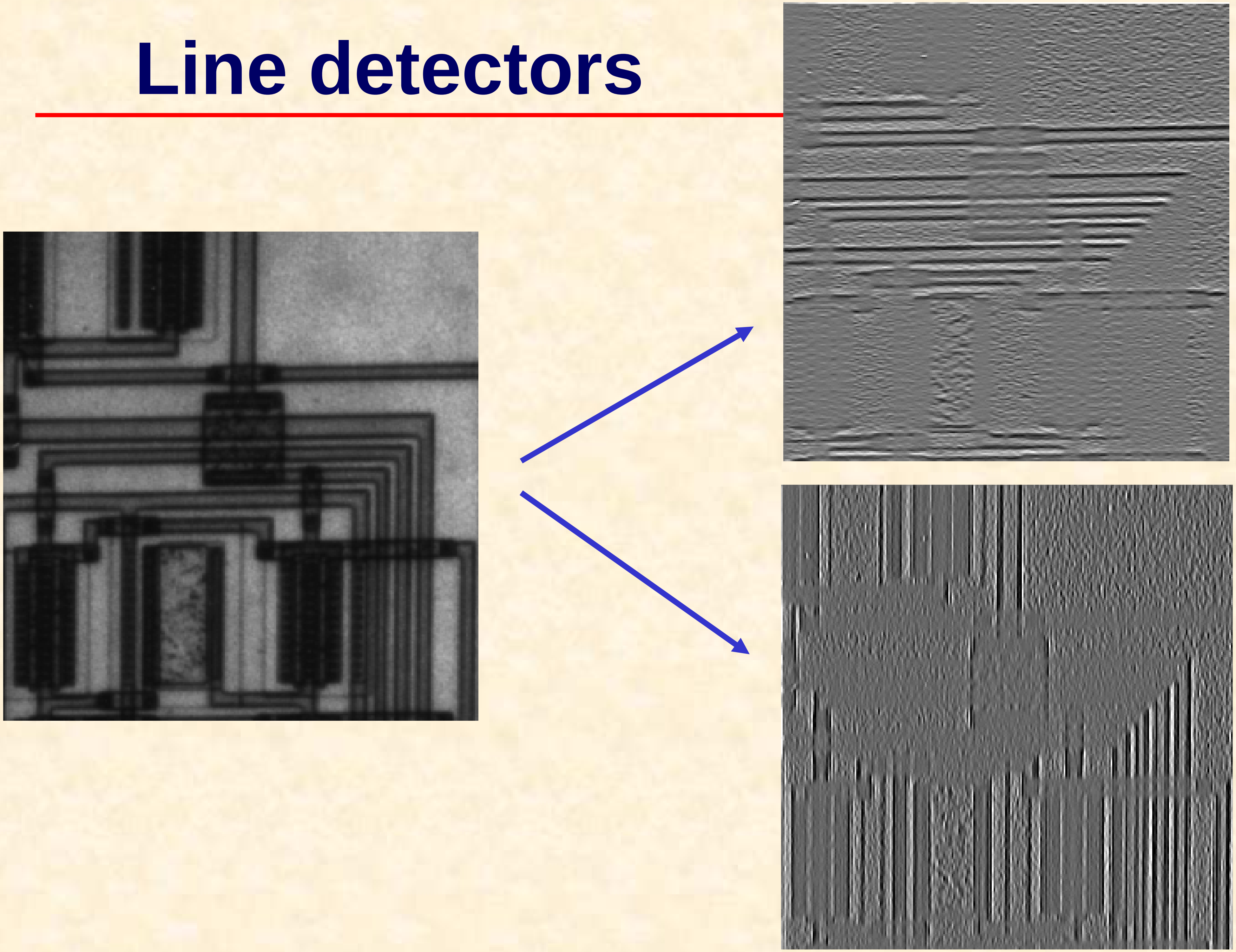
-1	-1	2
-1	2	-1
2	-1	-1

+45°

2	-1	-1
-1	2	-1
-1	-1	2

-45°

Line detectors

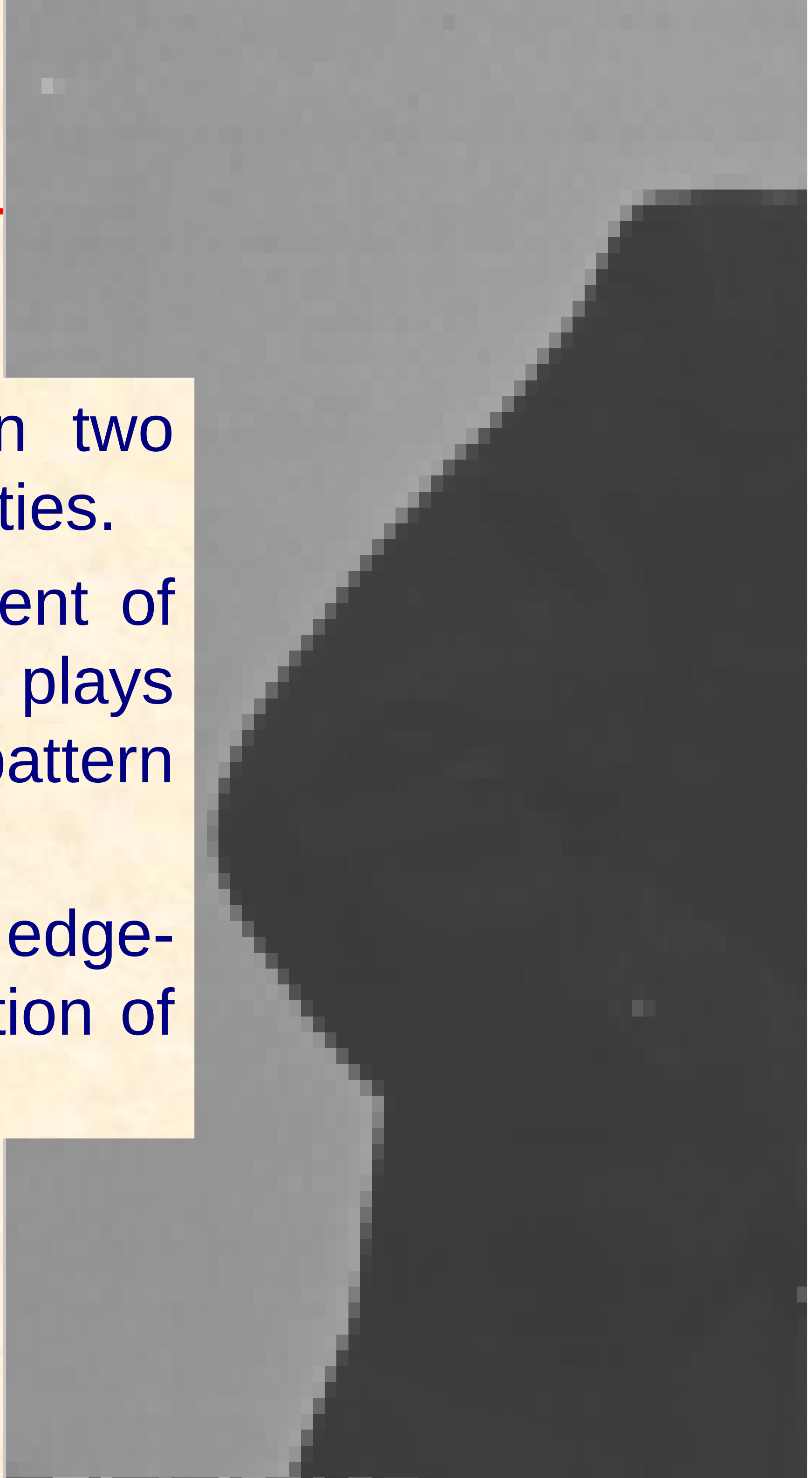


Edge detection

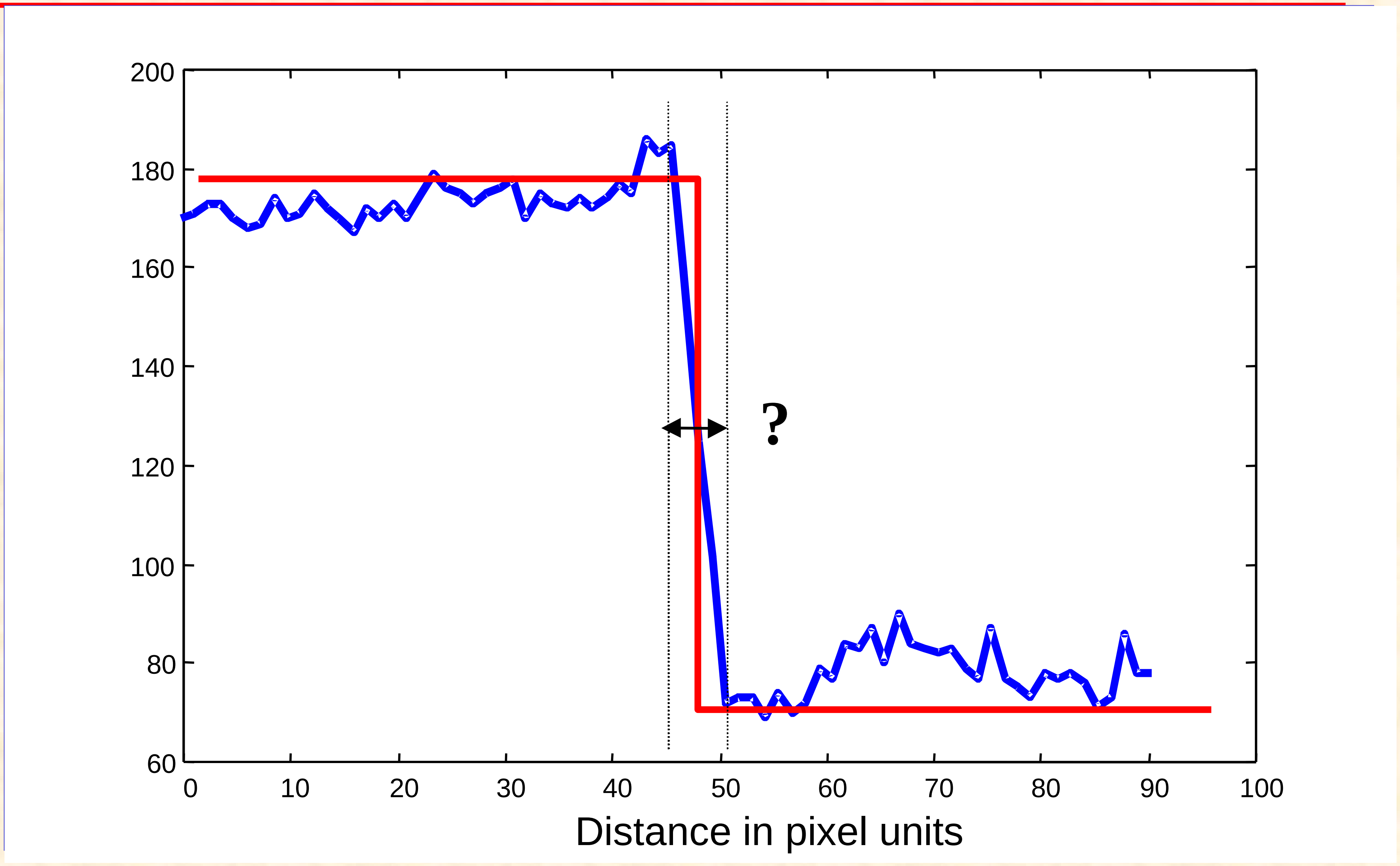
An **edge** is the boundary between two regions with distinct gray-level properties.

Edges characterise the physical extent of objects thus their accurate detection plays a key role in image analysis and pattern recognition problems.

The main idea underlying most edge-detection techniques is the computation of a local derivative of an image.

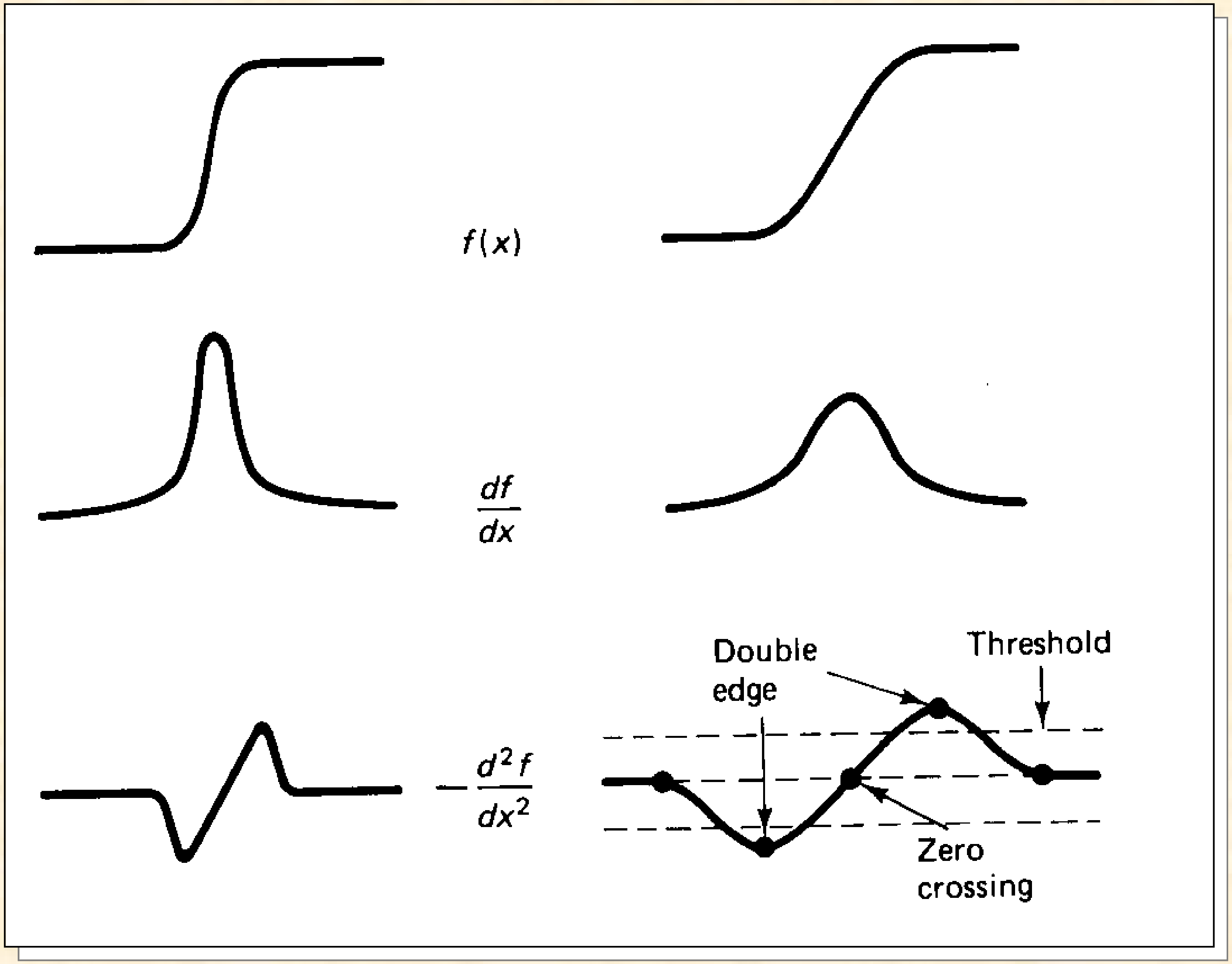


Edge detection

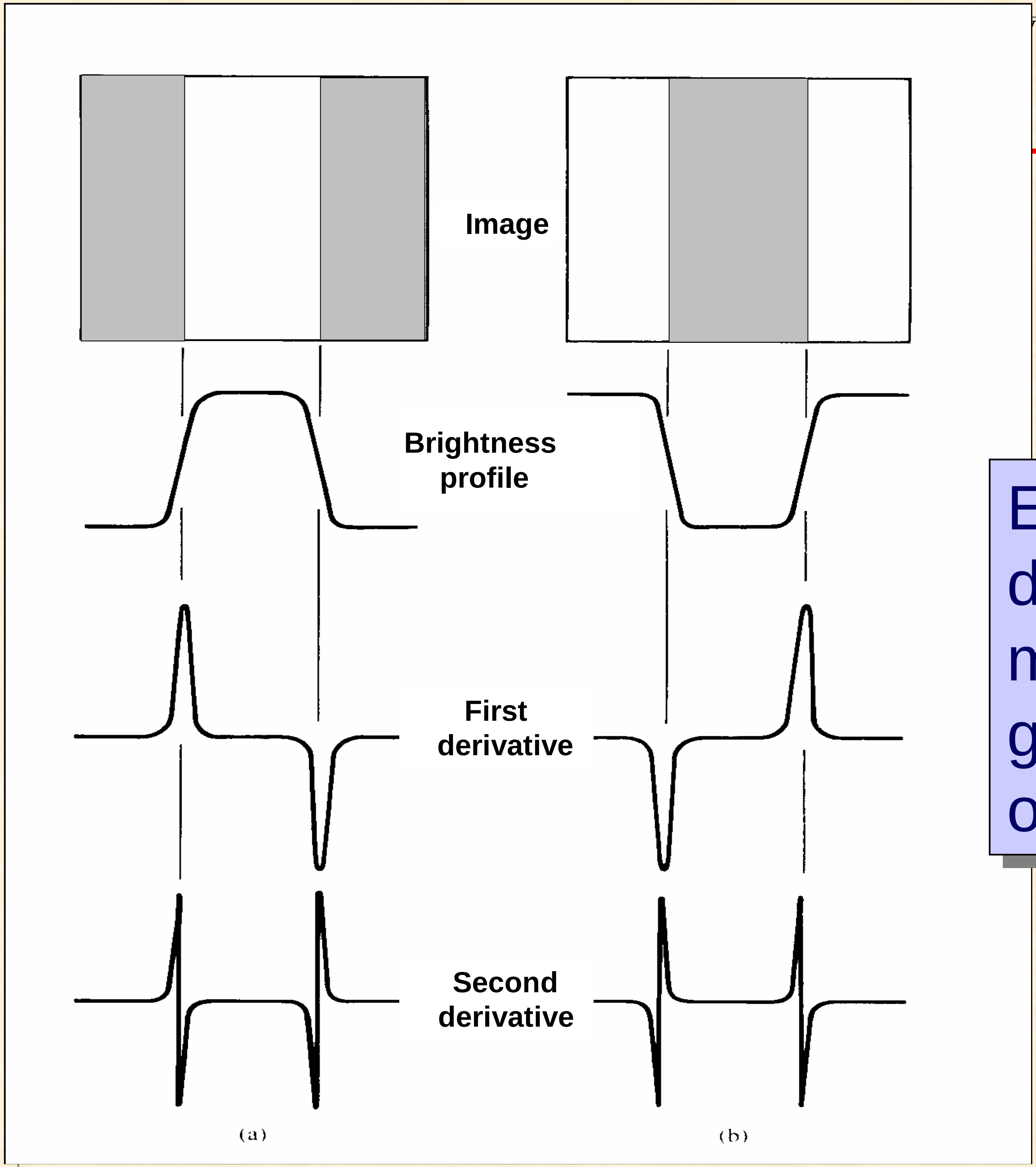


“Ideal” and real edge profiles

How gradient operators work?



Edge detection by means of gradiant operators



Edge detection

Note the following points about image derivative operators:

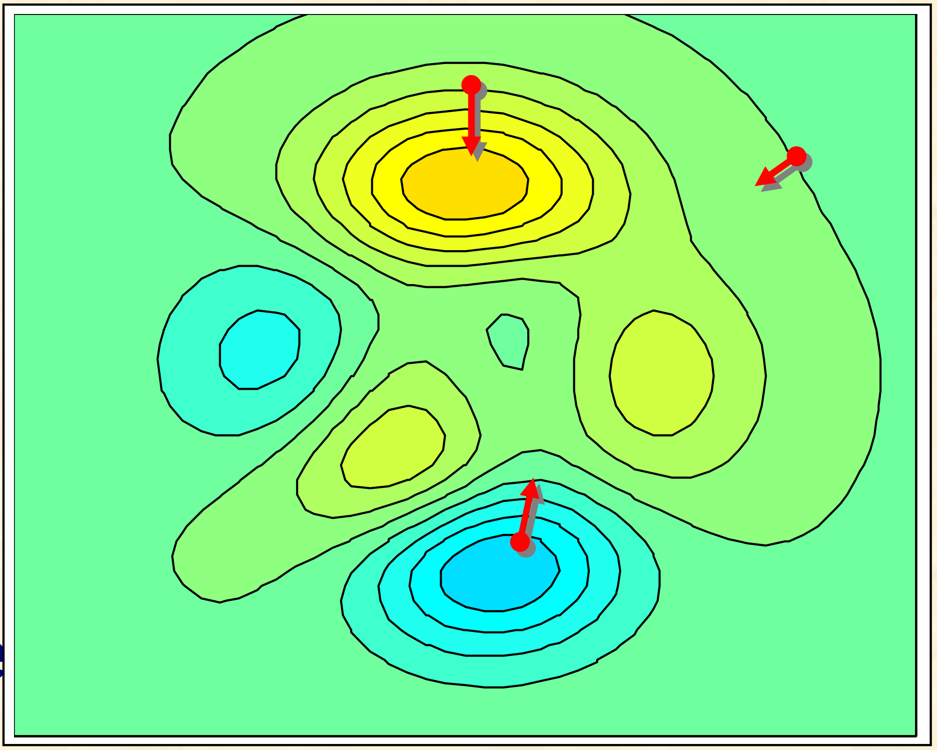
- the magnitude of the first derivative can be used to detect the presence of an edge in an image,
- the sign of the second derivative can be used to determine whether an edge pixel is on the dark or light side of an edge,

the second derivative has z zero crossing at the midpoint of a gray-level transition.

Gradient operators

The **gradient** of an image *f(x,y)* at location *(x,y)* is defined as the **vector**:

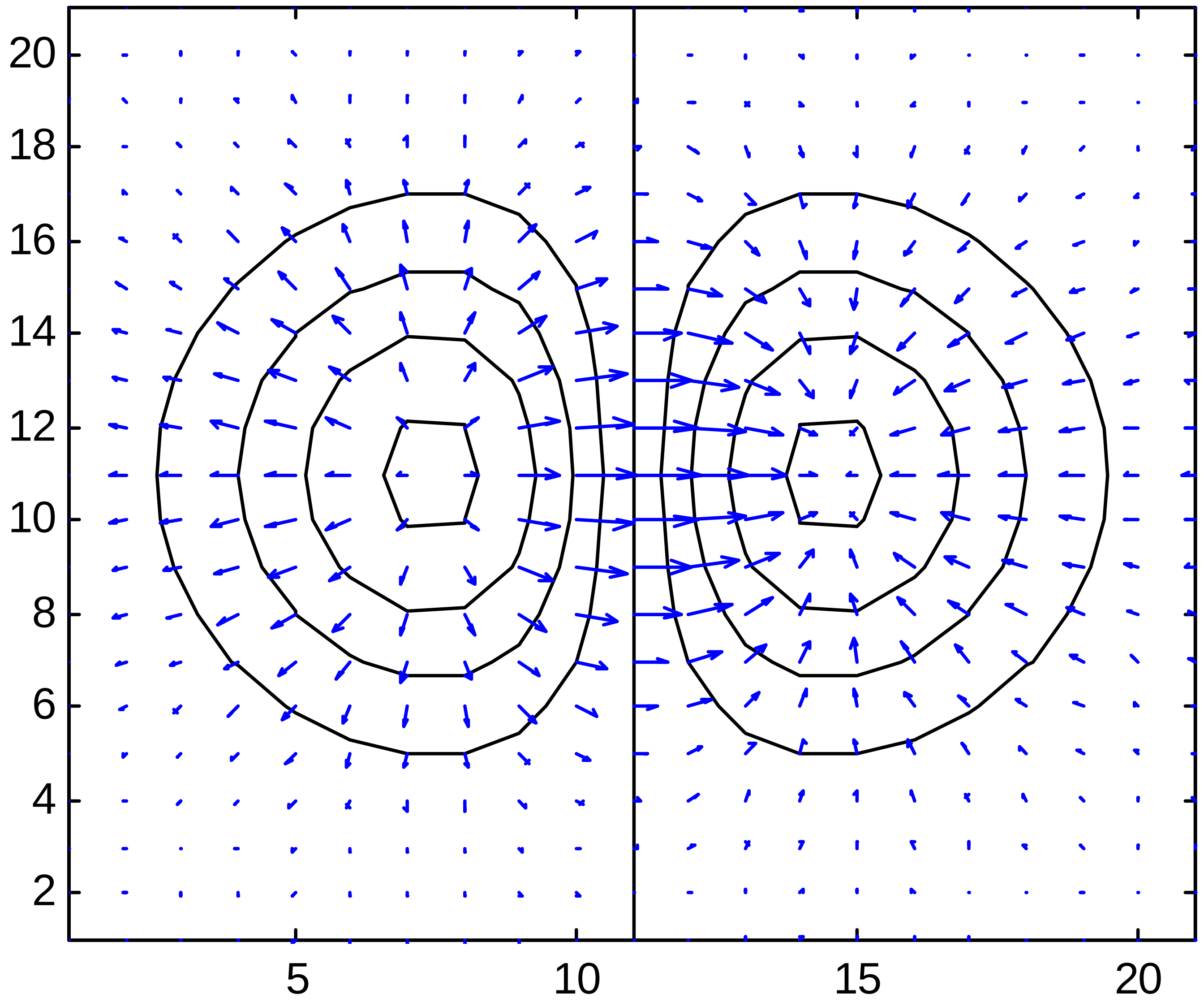
$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$



The gradient vector points in the direction of maximum rate of change of at *(x,y)*. Magnitude of this vector, called simply the *gradient*, is computed from:

$$\nabla f = mag(\nabla f) = \left[G_x^2 + G_y^2\right]^{1/2} \approx |G_x| + |G_y|$$

Gradient of an image



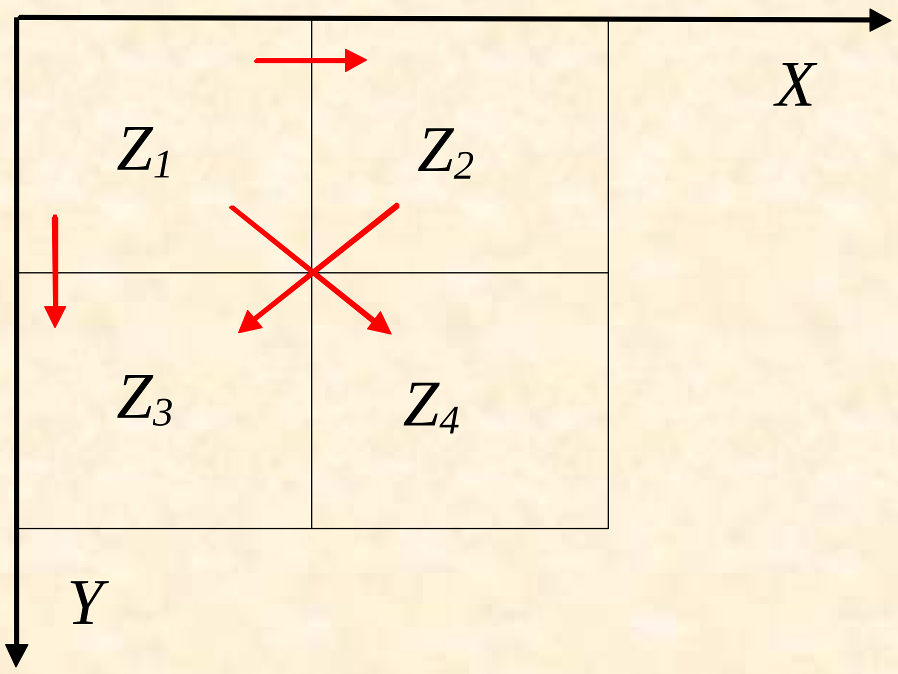
Gradient operators

Gradient for a discrete image:

$$\nabla f \approx |z_1 - z_2| + |z_1 - z_3|$$

or for diagonal directions:

$$\nabla f \approx |z_1 - z_4| + |z_2 - z_3|$$



Examples of gradiant masks

Roberts

1	0
0	-1

h_1

0	1
-1	0

h_2

$$g(x,y) = |h_1**f(x,y)| + |h_2**f(x,y)|$$

Examples of gradiant masks

```
%Matlab
h=fspecial('prewitt')
```

Prewitt

-1	-1	-1
0	0	0
1	1	1

h_1

-1	0	1
-1	0	1
-1	0	1

h_2

$$g(x,y) = |h_1**f(x,y)| + |h_2**f(x,y)|$$

$$g(x,y) = ([h_1**f(x,y)]^2 + [h_2**f(x,y)]^2)^{1/2}$$

Examples of gradiant masks

```
%Matlab
h=fspecial('sobel')
```

Sobel

1	2	1
0	0	0
-1	-2	-1

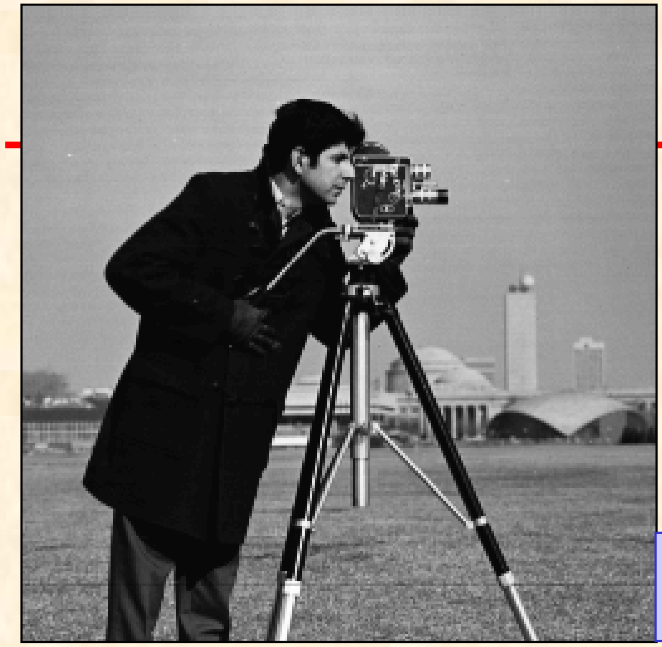
h_1

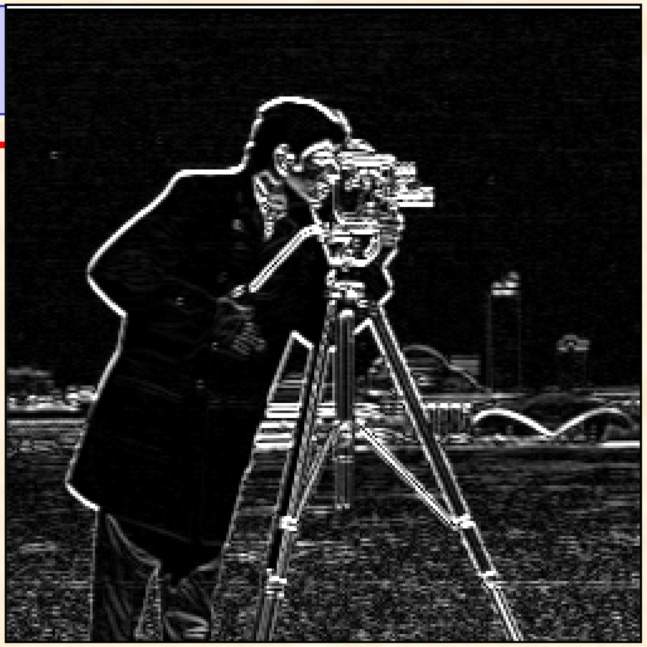
-1	0	1
-2	0	2
-1	0	1

h_2

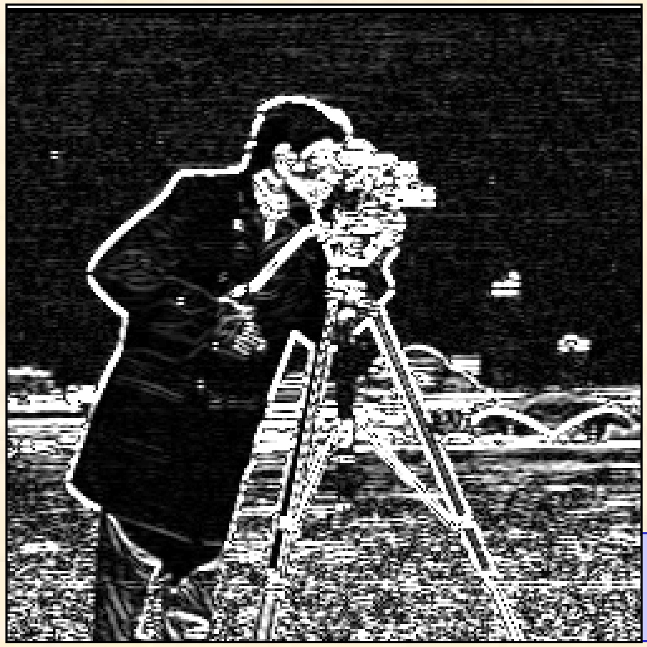
$$g(x,y) = |h_1{**}f(x,y)| + |h_2{**}f(x,y)|$$

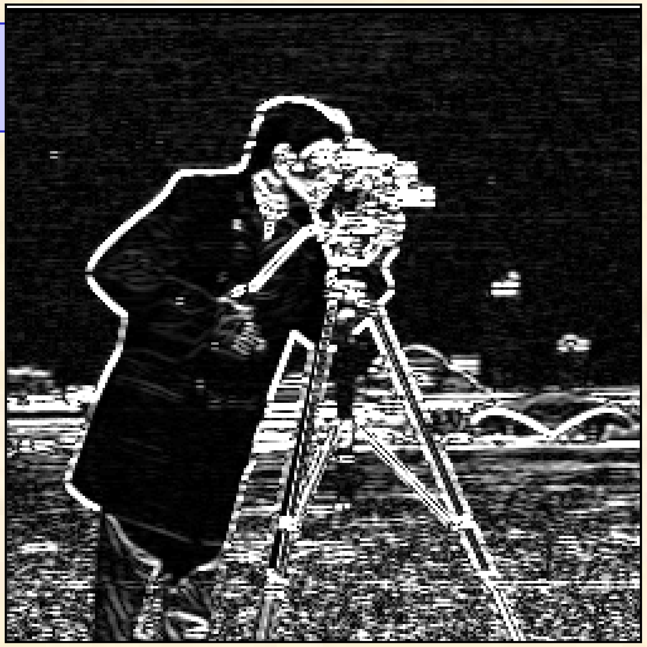
$$g(x,y) = ([h_1{**}f(x,y)]^2 + [h_2{**}f(x,y)]^2)^{1/2}$$


Original

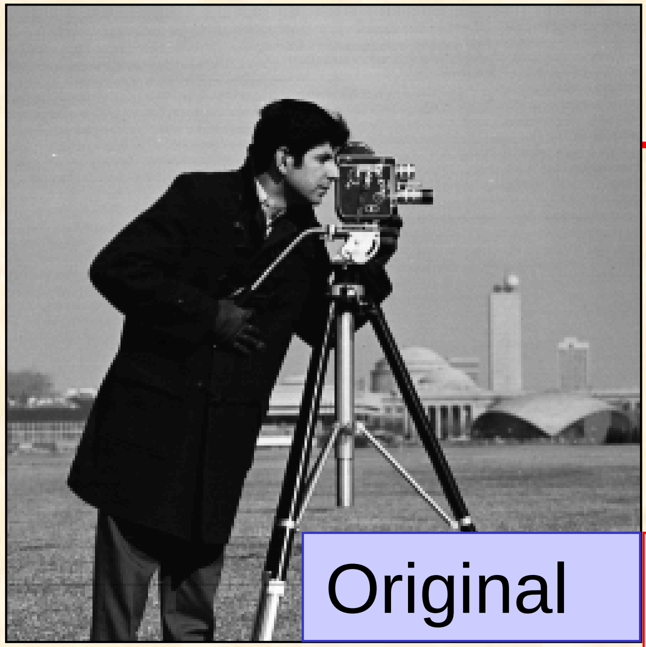

Roberts

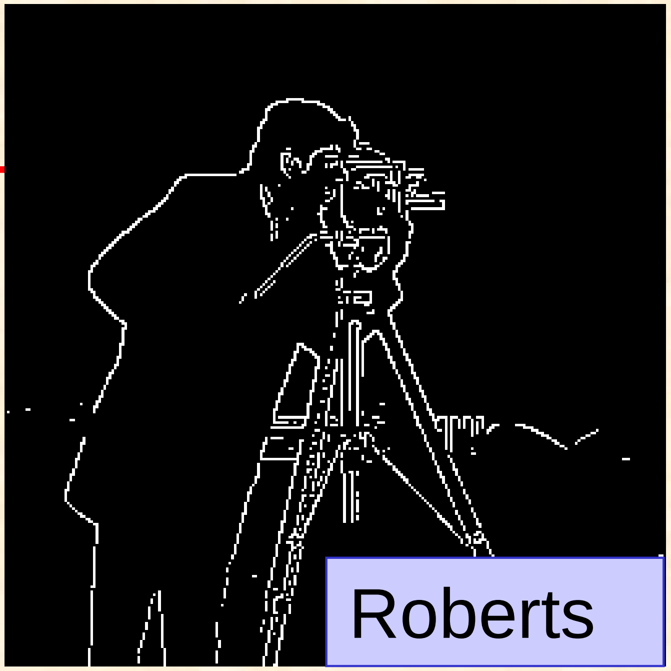
Gradient images


Sobel

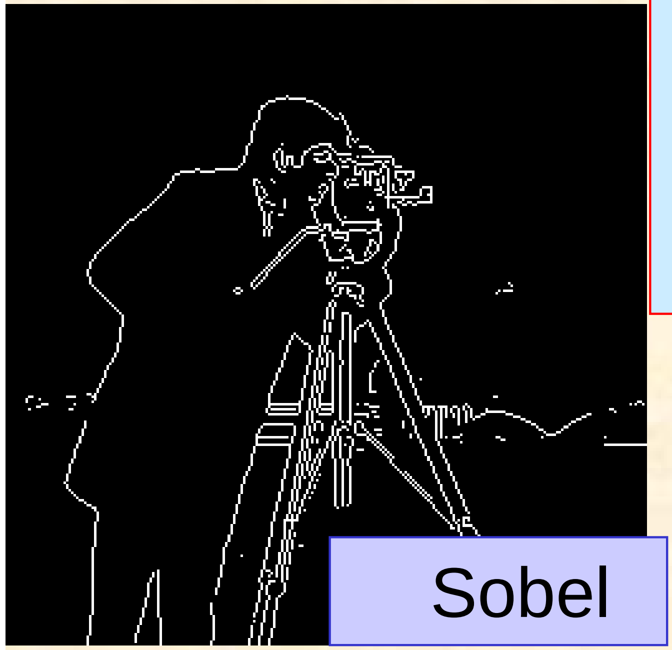

Prewitt

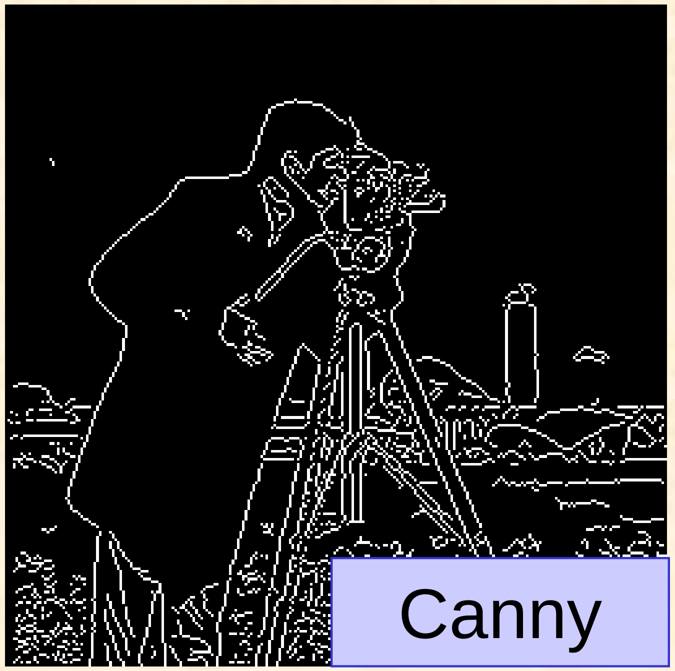
Gradient images





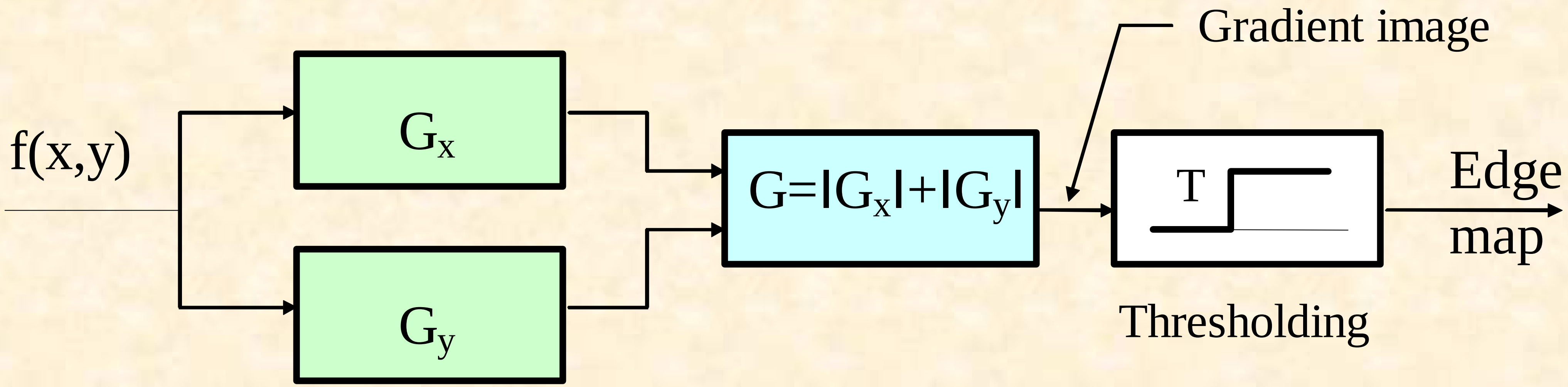
```
%MATLAB
I = imread(‘cameraman.tif');
BW1 = edge(I,‘roberts');
BW2 = edge(I,'sobel');
BW3 = edge(I,'canny');
figure, imshow(BW1);
figure, imshow(BW2);
figure, imshow(BW3);
```





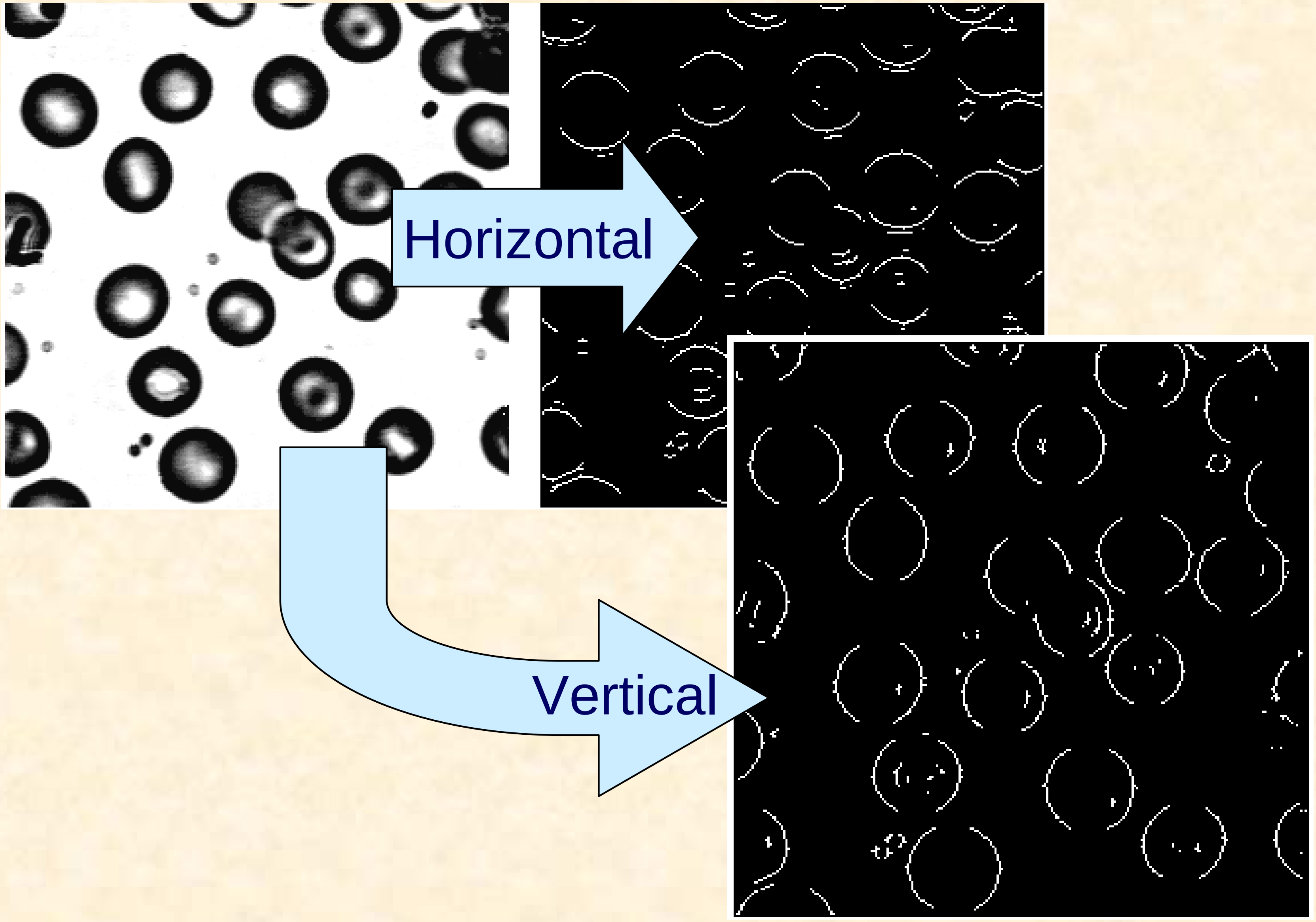
Edge detection procedure

The pixel location is considered as an edge location if $\nabla f(x,y)$ exceeds some threshold T. Typically, T may be selected using the **cumulative histogram** of the gradient image.



The **edge map** (a binary image) gives the necessary data for tracing the object boundaries in an image.

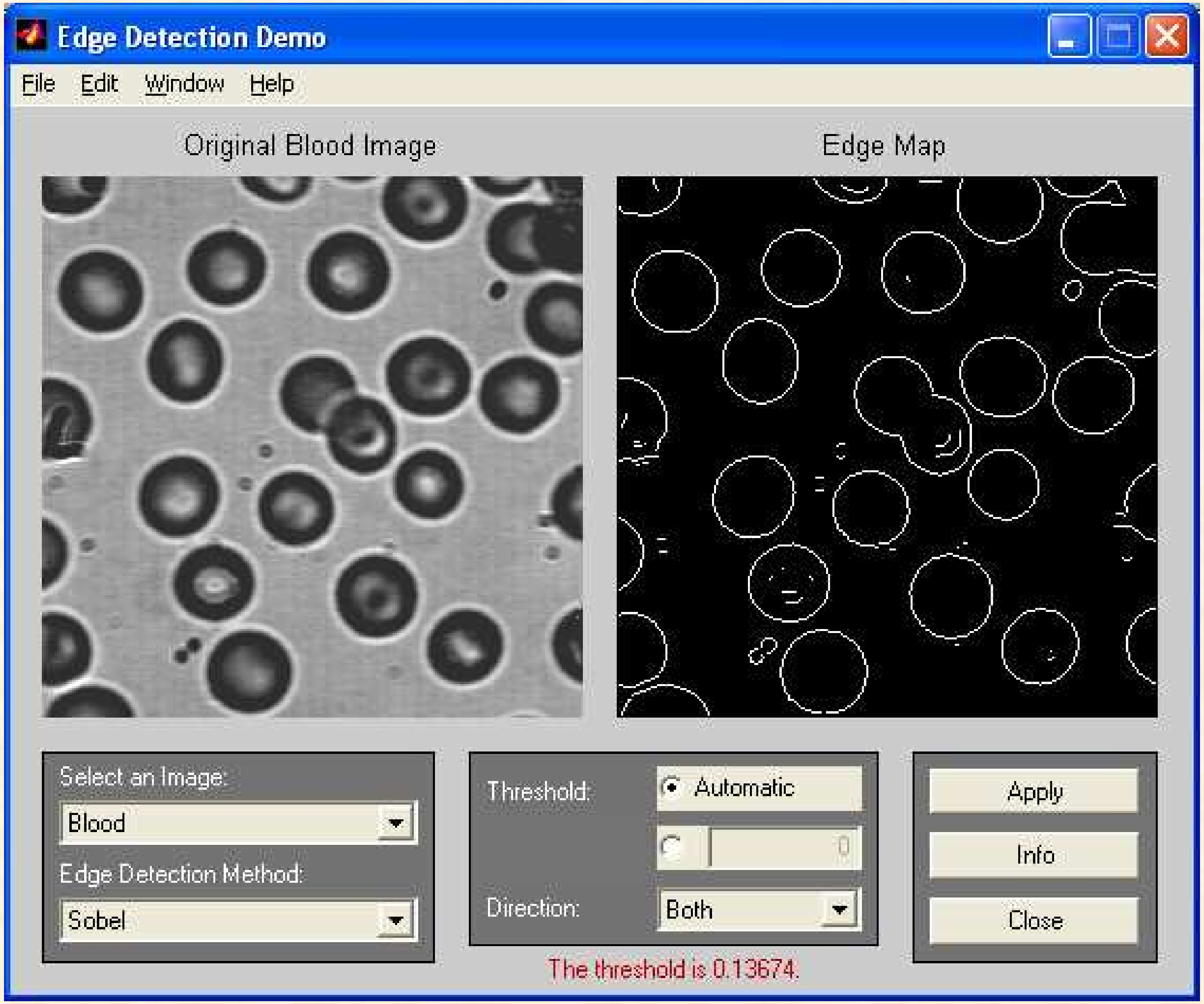
Edge detection



Edge detection

MATLAB

Demo


Edge Detection Demo

File Edit Window Help

Original Blood Image

Edge Map

Select an Image: Blood

Edge Detection Method: Sobel

Threshold: [✓ Automatic]

Direction: Both

Apply

Info

Close

The threshold is 0.13674.

Laplacian

Laplacian of a 2-D function f(x,y) is a second derivative defined as:

$$\boxed{\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}}$$

The Laplacian of a discrete image can be approximated by a difference equation:

$$\boxed{\nabla^2 f \approx 4z_5 - \left(z_2 + z_4 + z_6 + z_8\right)}$$

Laplacian

```
%Matlab
h=fspecial('laplacian')
```

0	-1	0
-1	4	-1
0	-1	0

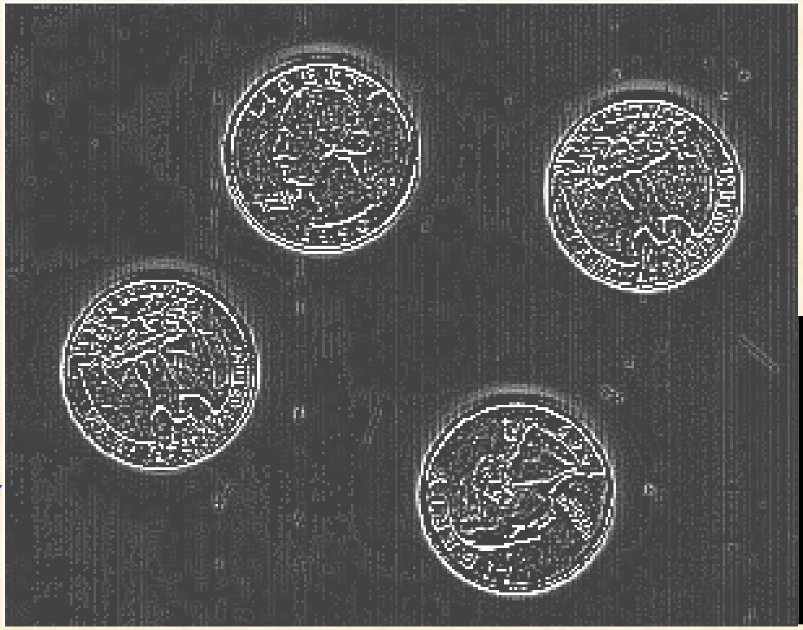
(a)

-1	-1	-1
-1	8	-1
-1	-1	-1

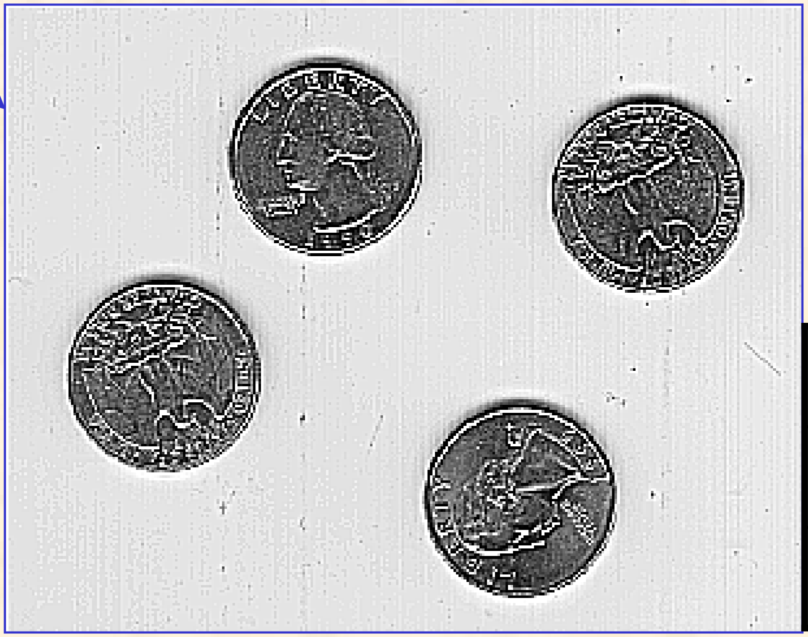
(b)

Two possible versions of the Laplacian masks.

Gradient operators at work



-1	-1	-1
-1	8	-1
-1	-1	-1



-1	-1	-1
-1	9	-1
-1	-1	-1

```
%MATLAB
output=filter2(h,input);

help fspecial
```

Laplacian in frequency domain

Fourier transform of a derivative of a function

$$\boxed{\Im\left\{f^{n}(x)\right\}=(j\omega)^{n}F(\omega)}$$

For Laplacian one gets:

$$\boxed{\Im\left\{\nabla^{2}f\right\}=-\left(\omega_x^2+\omega_y^2\right)F\left(\omega_x,\omega_y\right)}$$

higher frequency components are „amplified”.

Laplacian

The Laplacian plays a secondary role in edge detection due to the following shortcomings:

- is unacceptably sensitive to noise (second derivative),
- produces double edges,
- unable to detect edge direction.

$$\boxed{\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}}$$

Laplacian

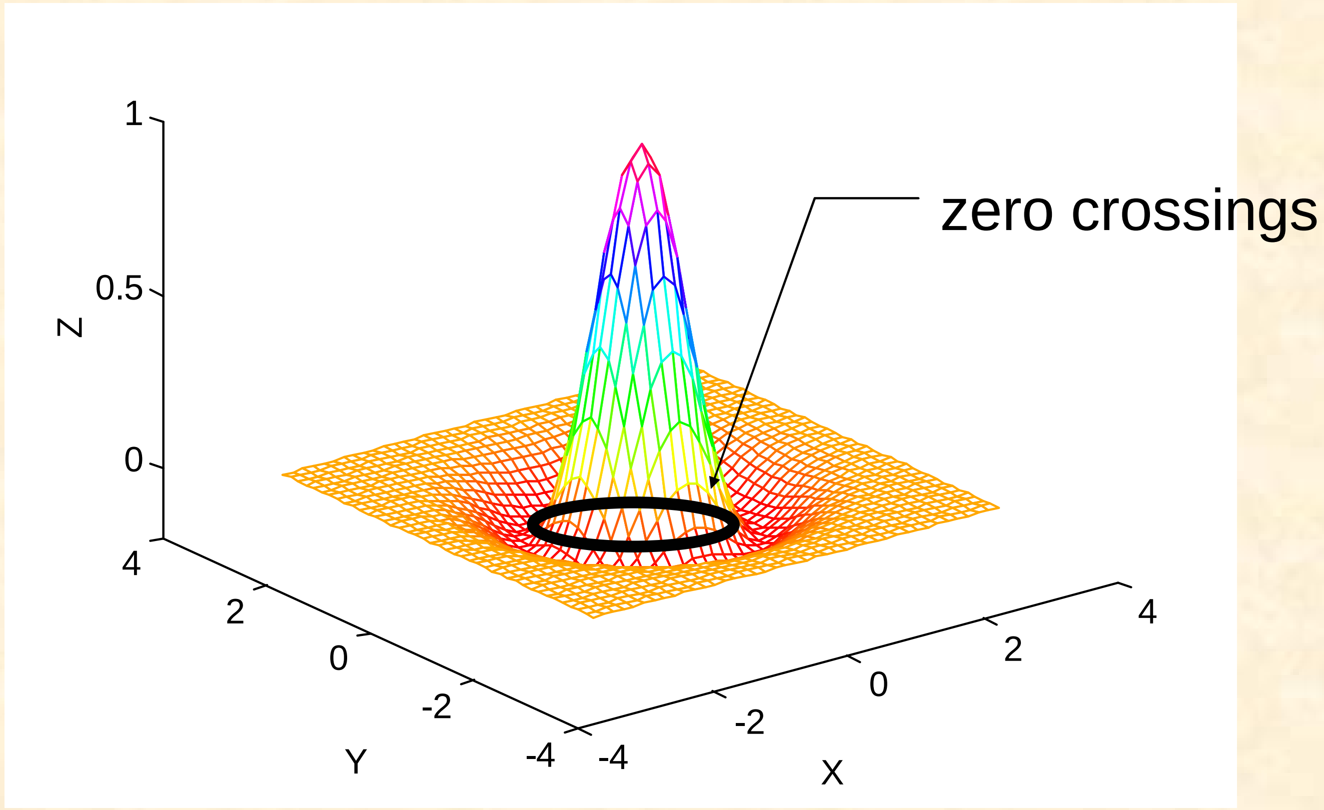
A more suitable use of the Laplacian is in finding the location of edges using its *zero-crossing* property. This concept is based on convolving an image with the Laplacian of a 2-D Gaussian function of the form:

$$\boxed{h(x,y) = exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)}$$

where σ is the standard deviation. Assume $r^2=x^2+y^2$. Then, the Laplacian of h with respect to r is:

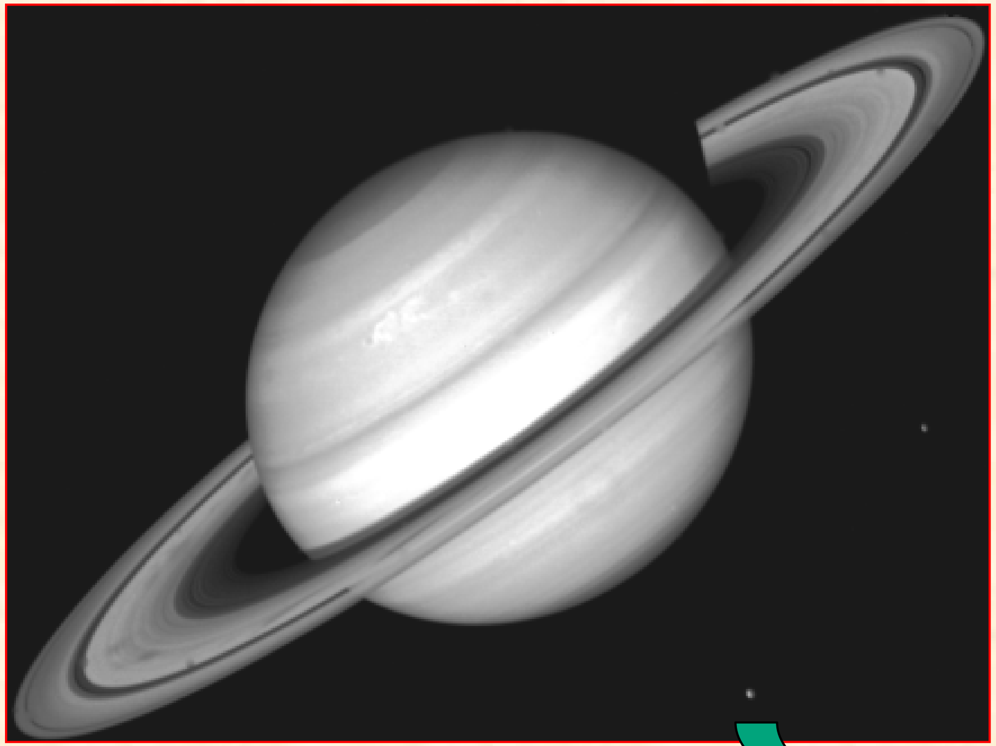
$$\boxed{\nabla^2 h = \left(\frac{r^2-\sigma^2}{\sigma^4}\right) exp\left(-\frac{r^2}{2\sigma^2}\right)}$$

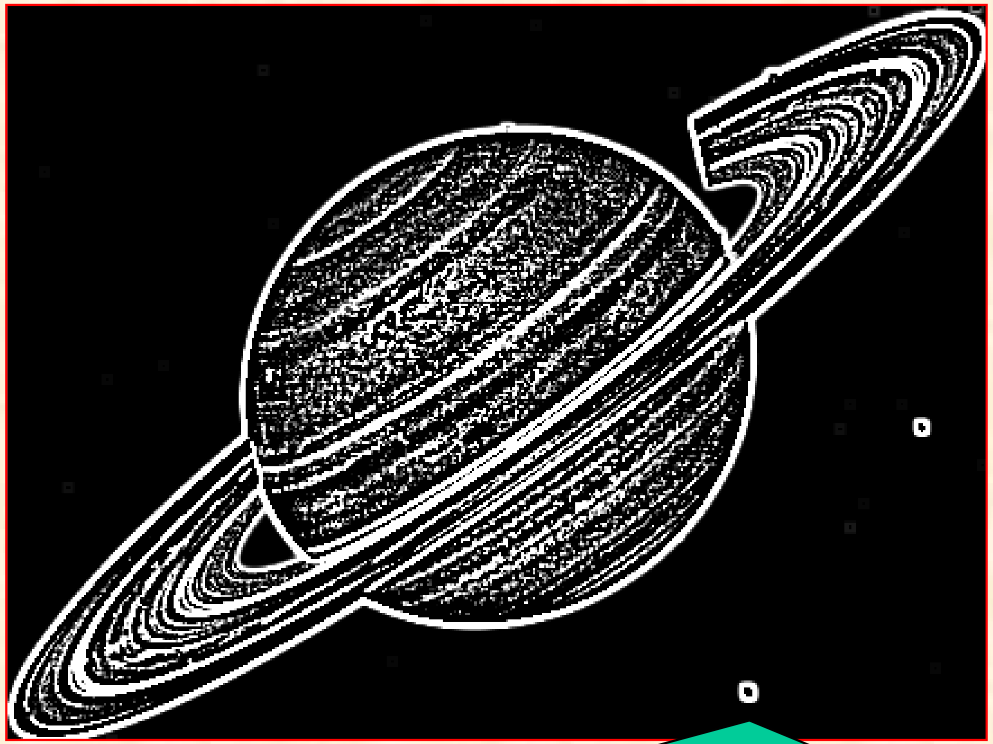
Laplacian



Laplacian of a 2-D Gaussian function for $\sigma=1$ (also called the Mexican hat function).

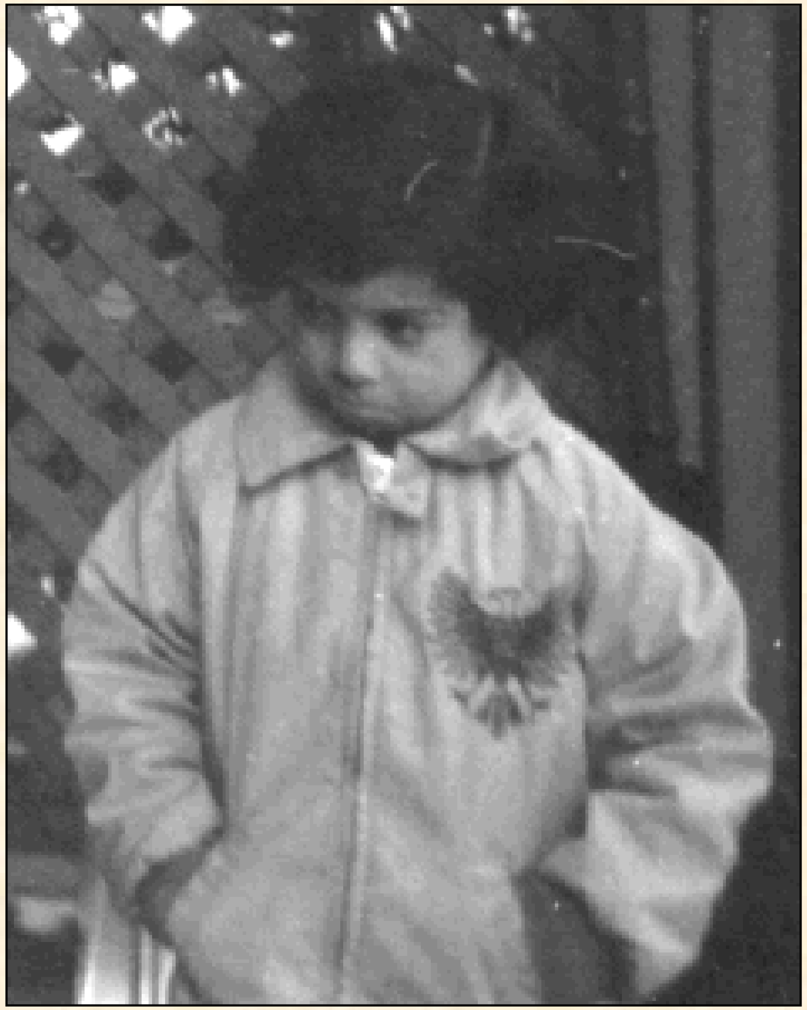
Operator **LoG** (**L**aplacian **of** **G**aussian)

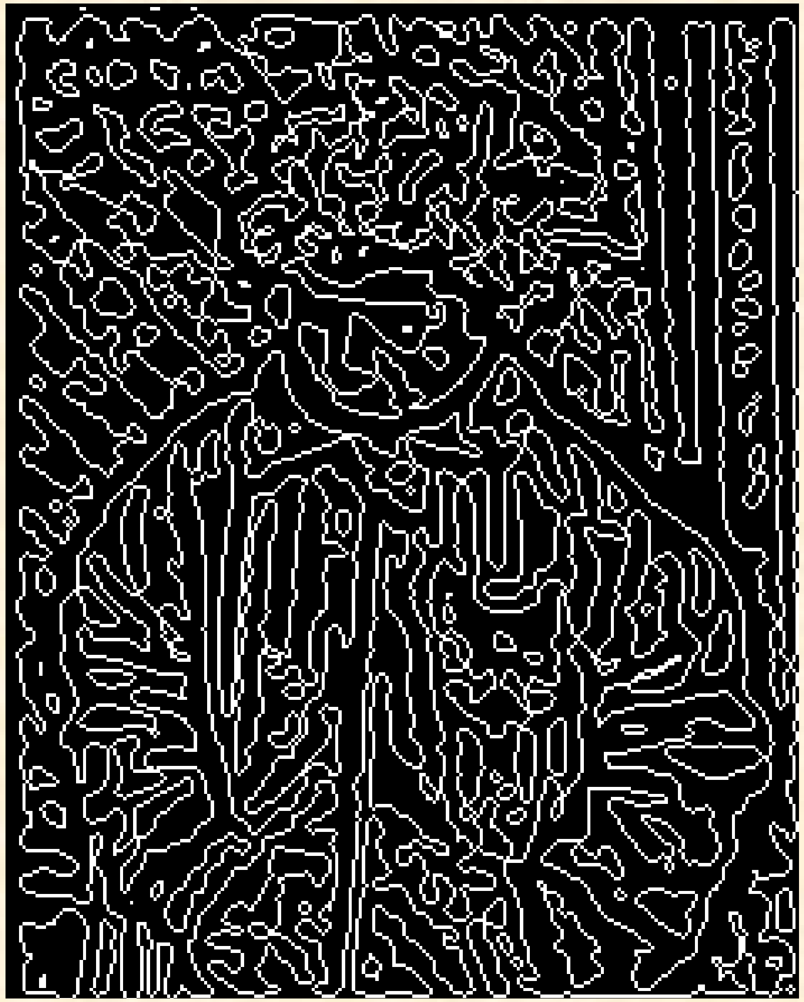




```
%MATLAB
I = imread('saturn.tif');
h = fspecial('log',[5 5], 0.2);
I2 = filter2(h,I)/255;
imshow(I), figure, imshow(I2)
```

Lapalcian of an image





An image and its zero-crossings.