Filtering - I

-Noise removal

-Edge Detection

Suggested Reading

- Chapter 7 & 8, David Forsyth and Jean Ponce, "*Computer Vision: A Modern Approach*“
- Chapter 4, Trucco & Verri, "*Introductory Techniques for 3-D Computer Vision*"
- Chapter 2, Mubarak Shah, “*Fundamentals of Computer Vision*”

Noise in Images

- Image contains noise due to
  - Lighting variations
  - Lens de-focus
  - Camera electronics
  - Surface reflectance
- Remove noise
  - Averaging
  - Weighted averaging

filtering

- Consider a 3 x 3 image block organized as a vector I
- Take a 3 x 3 filter operator organized as a vector F
- The operator is applied by replacing the central pixel of the original 3x3 block with the dot product I.F

Linear Filtering

- The output is the linear combination of the neighborhood pixels

$$f(p) = \sum_{q_i \in N(p)} a_i q_i$$

- The coefficients of this linear combination combine to form the "filter-kernel"

1	3	0
2	10	2
4	1	1

Image

$\otimes$

1	0	-1
1	0.1	-1
1	0	-1

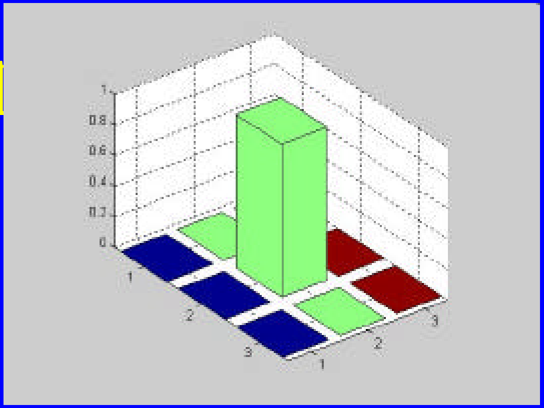
Kernel

$=$

	5	

Filter Output

Linear Filteri

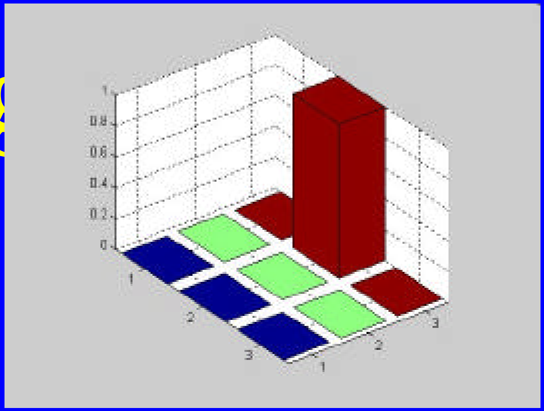


*

0	0	0
0	1	0
0	0	0

=

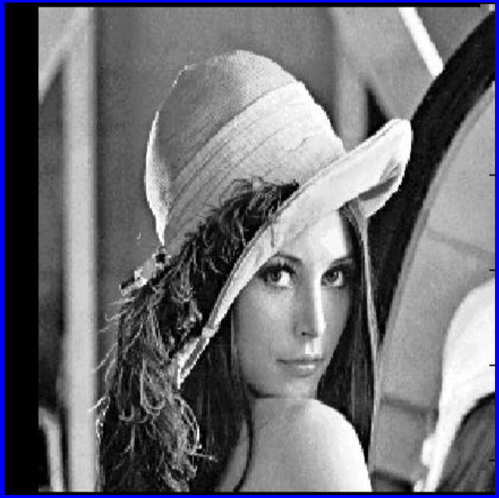
Linear Filtering



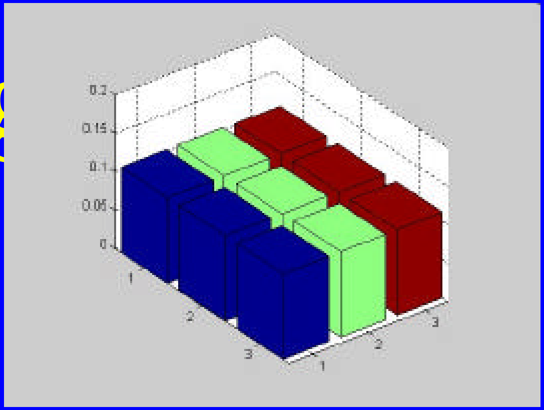
$*$

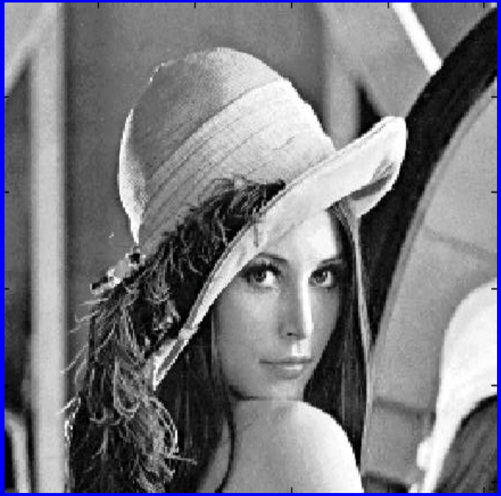
0	0	0
0	0	1
0	0	0

$=$

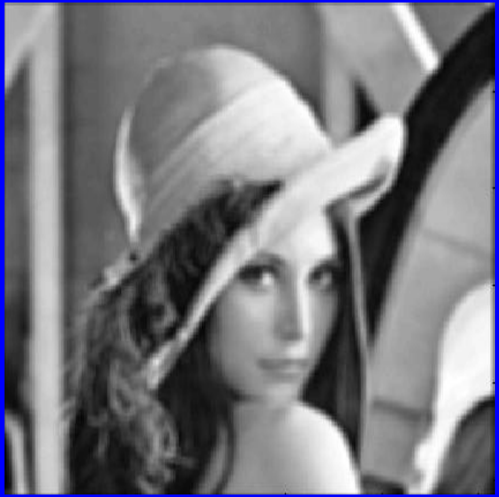


Linear Filtering





$$ * \frac{1}{9} \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array} = $$

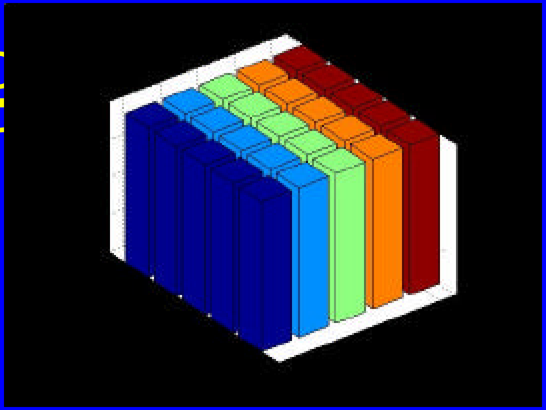


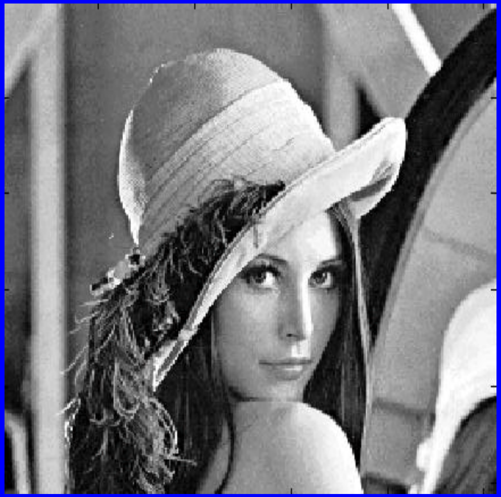
Averaging / Smoothing

- The average around a pixel results in a smoothing operation
- The process is repeated for each pixel in scan-line fashion, i.e. left to right and top to bottom.
- Larger the window size, more pronounced will be the smoothing effect

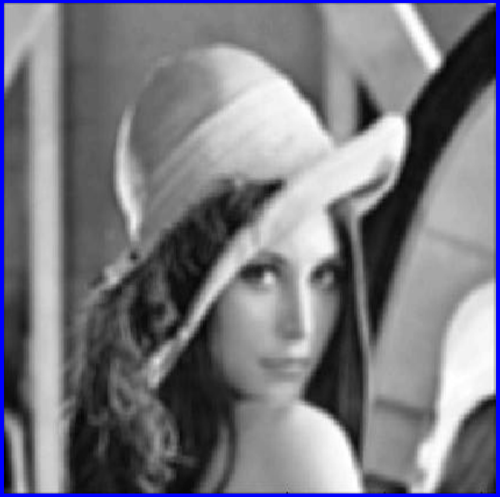
- It has the effect of blurring out the sharper details like edges and corners .
- Useful for removal of noise from the image.
- In frequency domain , this is equivalent to low pass filtering.

Linear Filtering

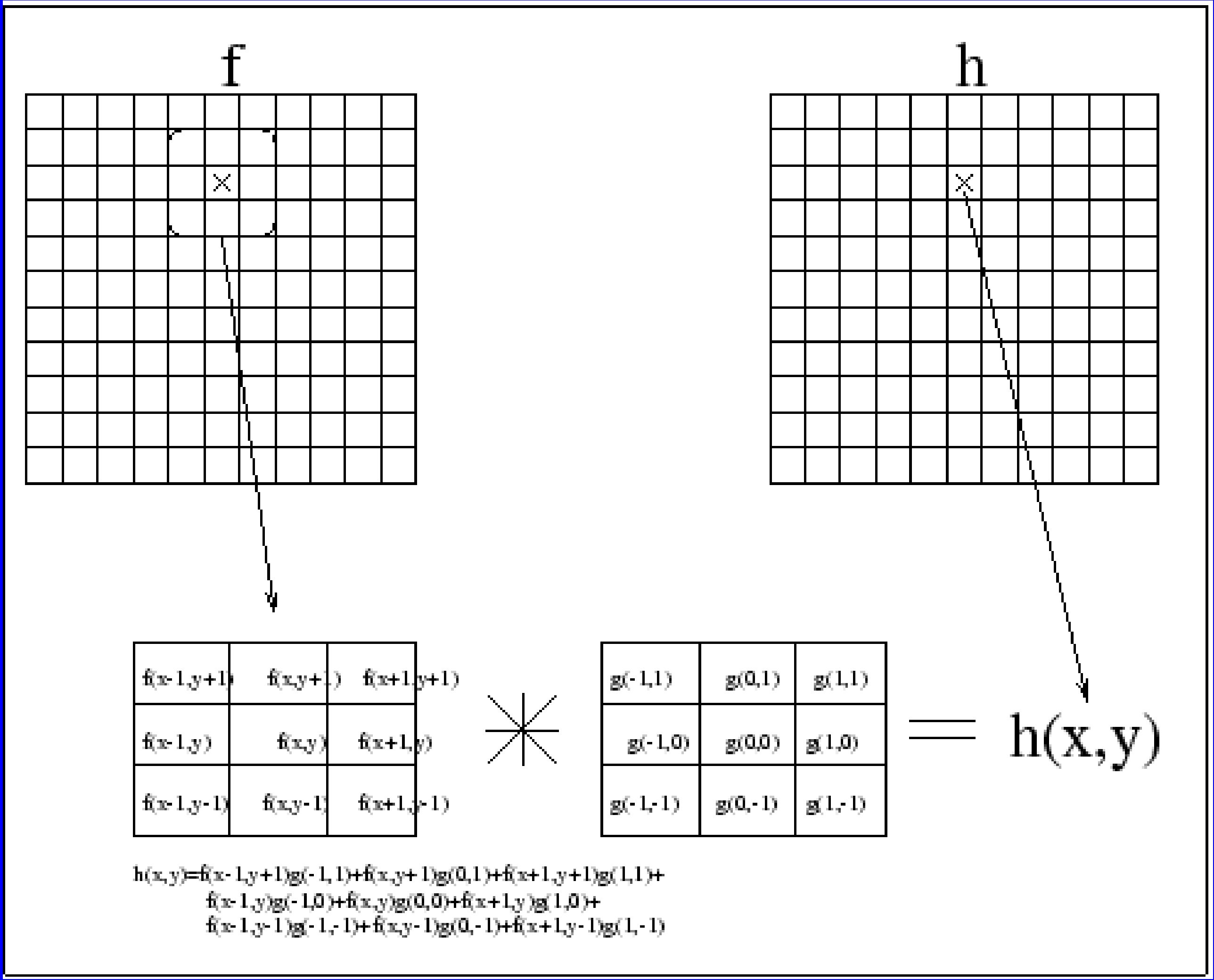




$$* \frac{1}{25} \begin{array}{|c|c|c|c|c|} \hline 1 & 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 \\ \hline \end{array} =$$



Convolution



f(x-1,y+1)	f(x,y+1)	f(x+1,y+1)
f(x-1,y)	f(x,y)	f(x+1,y)
f(x-1,y-1)	f(x,y-1)	f(x+1,y-1)

$*$

g(-1,1)	g(0,1)	g(1,1)
g(-1,0)	g(0,0)	g(1,0)
g(-1,-1)	g(0,-1)	g(1,-1)

$= h(x,y)$

$$\begin{aligned} h(x,y)=\;& f(x-1,y+1)g(-1,1)+f(x,y+1)g(0,1)+f(x+1,y+1)g(1,1)+\\ & f(x-1,y)g(-1,0)+f(x,y)g(0,0)+f(x+1,y)g(1,0)+\\ & f(x-1,y-1)g(-1,-1)+f(x,y-1)g(0,-1)+f(x+1,y-1)g(1,-1)\end{aligned}$$

Convolution (contd)

$$h(x, y) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} f(x+i, y+j) g(i, j)$$

$$h(x, y) = f(x, y) * g(x, y)$$

Convolution

$$f(i,j) = I * H = \sum_k \sum_l I(k,l)\,H(i-k,\, j-l)$$

I = Image

H = Kernel

$$\begin{tikzcd}
 & \begin{array}{|c|c|c|}\hline H_7 & H_8 & H_9 \\ \hline H_4 & H_5 & H_6 \\ \hline H_1 & H_2 & H_3 \\ \hline\end{array} \arrow[d, "Y-flip"] & \overset{H}{\begin{array}{|c|c|c|}\hline H_1 & H_2 & H_3 \\ \hline H_4 & H_5 & H_6 \\ \hline H_7 & H_8 & H_9 \\ \hline\end{array}} \arrow[l, "X-flip"'] \\
\overset{I}{\begin{array}{|c|c|c|}\hline I_1 & I_2 & I_3 \\ \hline I_4 & I_5 & I_6 \\ \hline I_7 & I_8 & I_9 \\ \hline\end{array}} \otimes & \begin{array}{|c|c|c|}\hline H_9 & H_8 & H_7 \\ \hline H_6 & H_5 & H_4 \\ \hline H_3 & H_2 & H_1 \\ \hline\end{array} \arrow[r] & \begin{aligned} I * H = {} & I_1H_9 + I_2H_8 + I_3H_7 \\ & + I_4H_6 + I_5H_5 + I_6H_4 \\ & + I_7H_3 + I_8H_2 + I_9H_1 \end{aligned}
\end{tikzcd}$$

Weighted Average

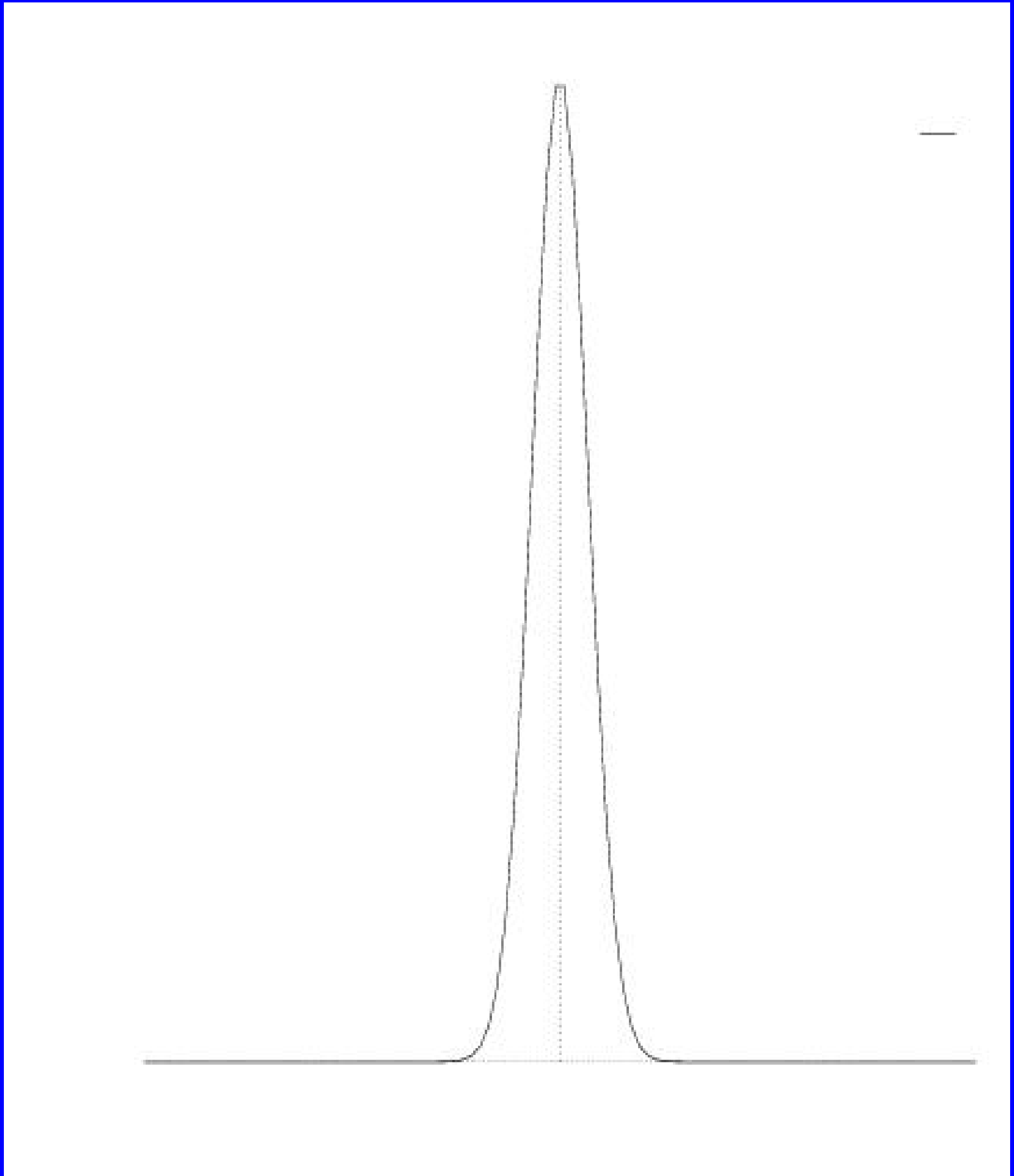
$$I = \frac{w_1 I_1 + w_2 I_2 + \ldots + w_n I_n}{n} = \frac{\sum_{i=1}^{n} w_i I_i}{n}$$

Gaussian

$$g(x) = e^{\frac{-x^2}{2o^2}}$$

Standard deviation

x	-3	-2	-1	0	1	2	3
g(x)	.011	.13	.6	1	.6	.13	.011



Gaussian

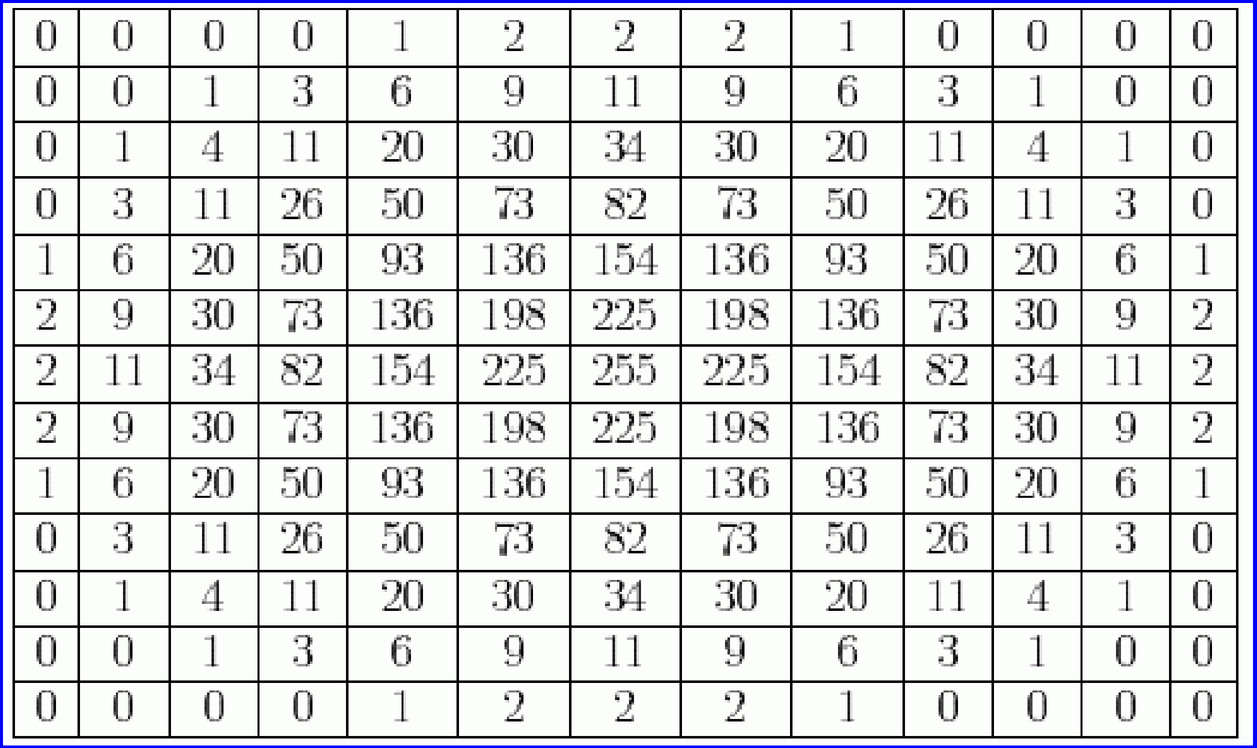
- Most natural phenomenon can be modeled by Gaussian.
- Take a bunch of random variables of any distribution, find the mean, the mean will approach to Gaussian distribution.
- Gaussian is very smooth function, it has infinite no of derivatives.

Gaussian

- Fourier Transform of Gaussian is Gaussian.
- If you convolve Gaussian with itself, it is again Gaussian.
- There are cells in human brain which perform Gaussian filtering.
  – Laplacian of Gaussian edge detector

2-D Gaussian

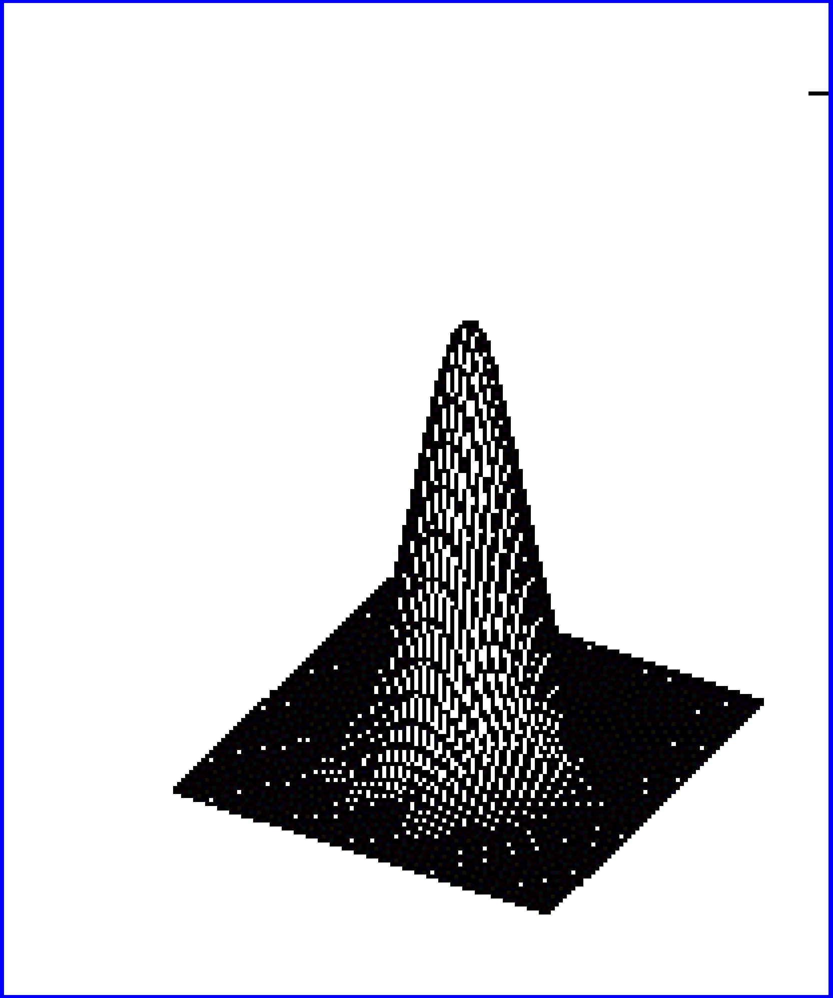
$$g(x,y)=e^{\frac{-(x^2+y^2)}{2o^2}}$$



0	0	0	0	1	2	2	2	1	0	0	0	0
0	0	1	3	6	9	11	9	6	3	1	0	0
0	1	4	11	20	30	34	30	20	11	4	1	0
0	3	11	26	50	73	82	73	50	26	11	3	0
1	6	20	50	93	136	154	136	93	50	20	6	1
2	9	30	73	136	198	225	198	136	73	30	9	2
2	11	34	82	154	225	255	225	154	82	34	11	2
2	9	30	73	136	198	225	198	136	73	30	9	2
1	6	20	50	93	136	154	136	93	50	20	6	1
0	3	11	26	50	73	82	73	50	26	11	3	0
0	1	4	11	20	30	34	30	20	11	4	1	0
0	0	1	3	6	9	11	9	6	3	1	0	0
0	0	0	0	1	2	2	2	1	0	0	0	0

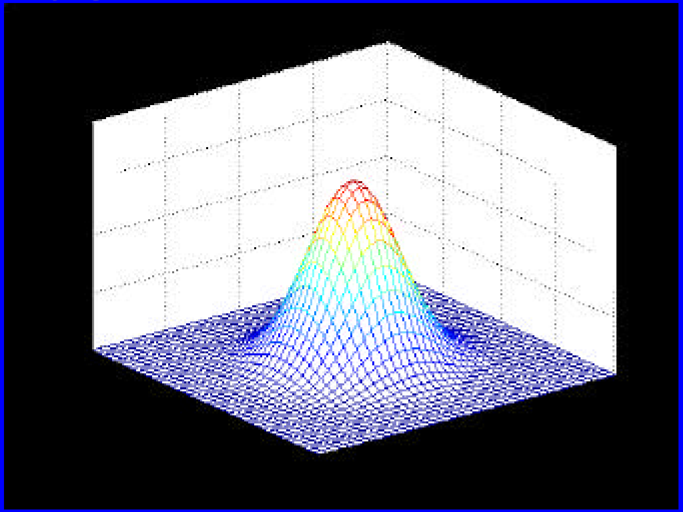
$s = 2$

2-D Gaussian



Gaussian Filter

$$G_s\left(x,y\right)=\frac{1}{2ps^{2}}\exp\left(-\frac{\left(x^{2}+y^{2}\right)}{2s^{2}}\right)$$

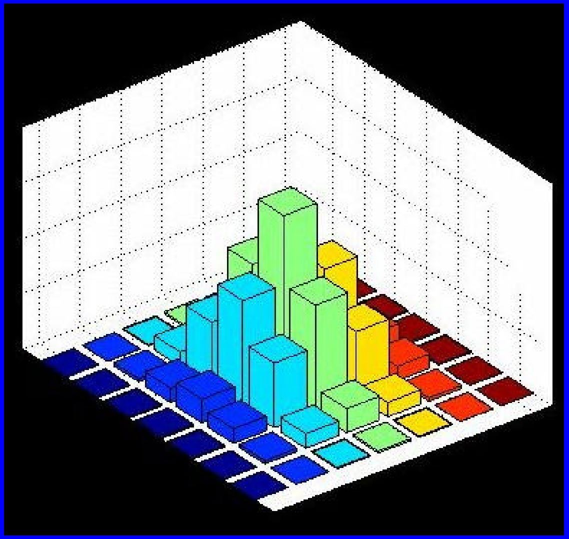


$$H\left(i,j\right)=\frac{1}{2ps^{2}}\exp\left(-\frac{\left(\left(i-k-1\right)^{2}+\left(j-k-1\right)^{2}\right)}{2s^{2}}\right)$$

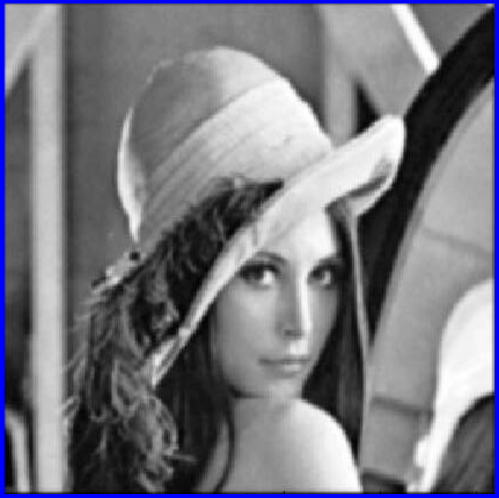
where $H\left(i,j\right)$ is $\left(2k+1\right)\times\left(2k+1\right)$ array

Linear Filtering(Gaussian Filter)

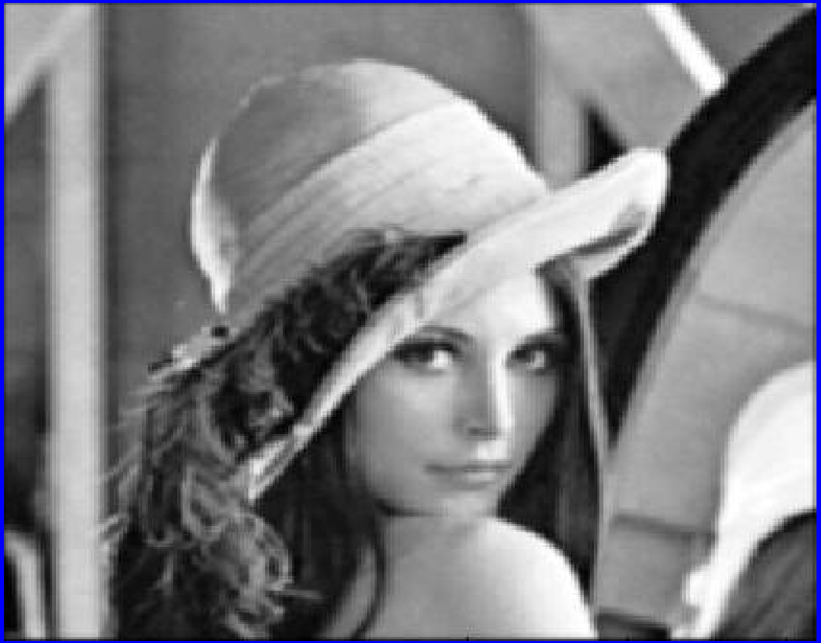
*

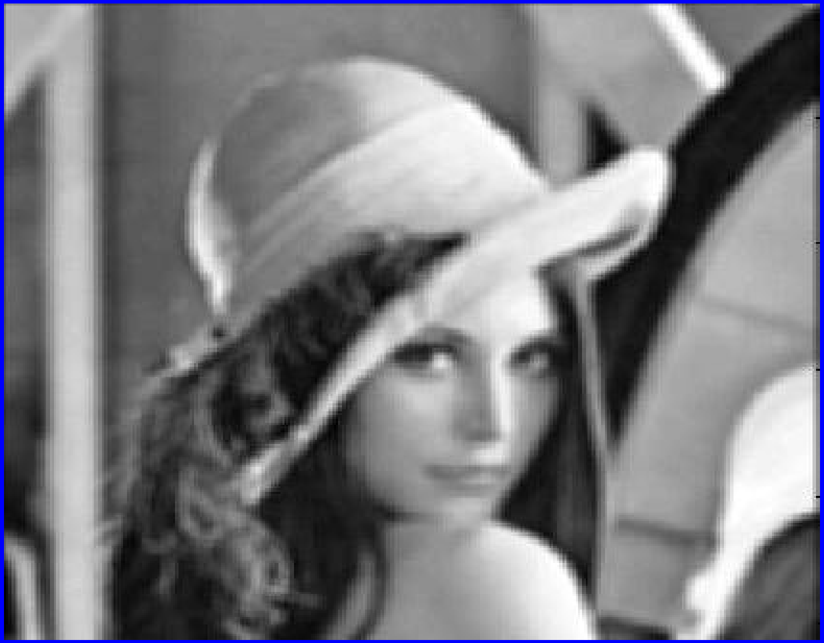


=

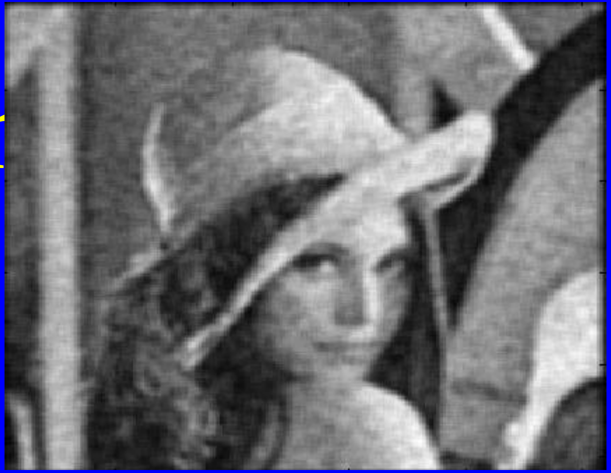


Gaussian Vs Average

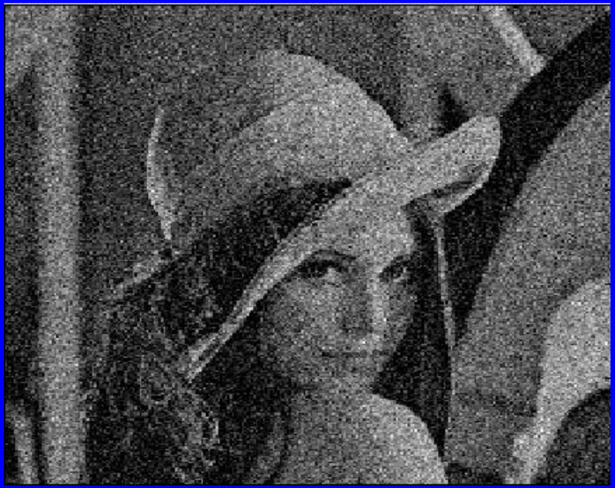

Gaussian Smoothing


Smoothing by Averaging

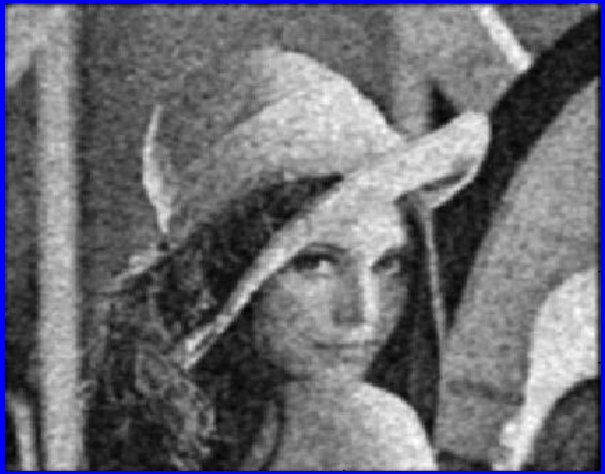
Noise Filter



After Averaging

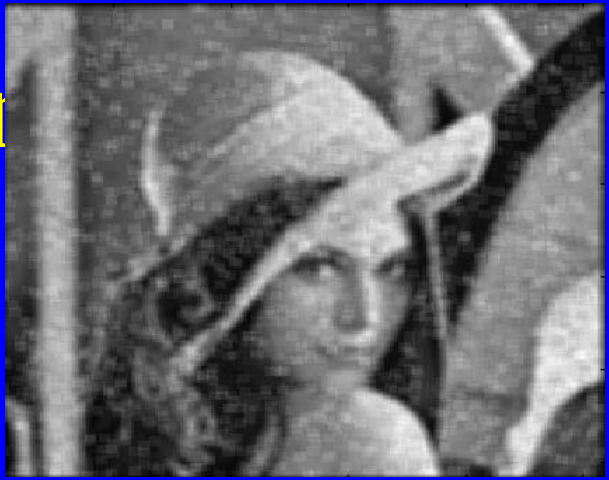


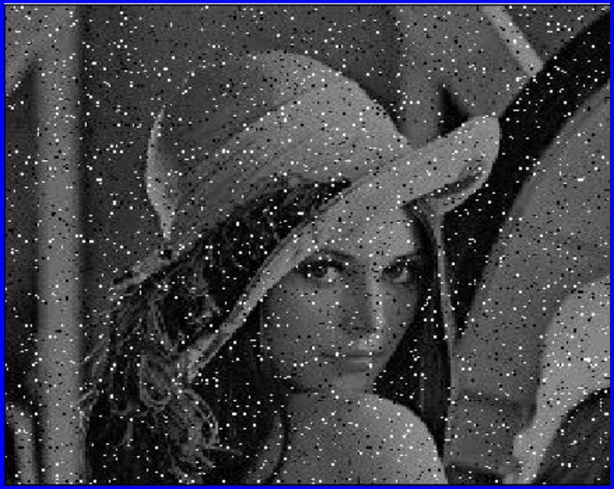
Gaussian Noise

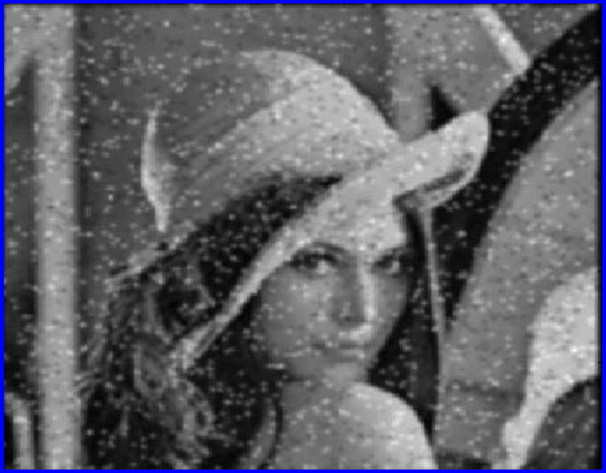


After Gaussian Smoothing

Noise Filter


After Averaging

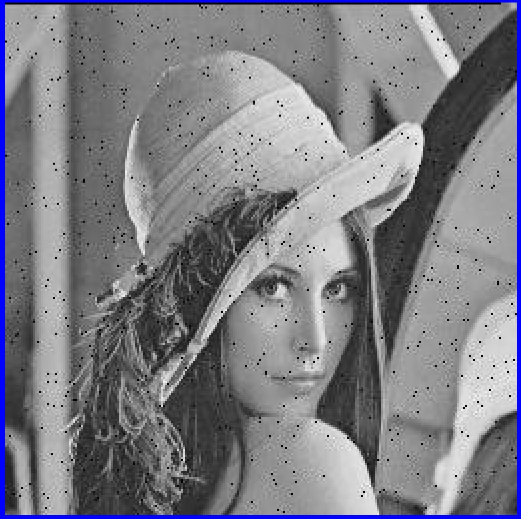

Salt & Pepper Noise

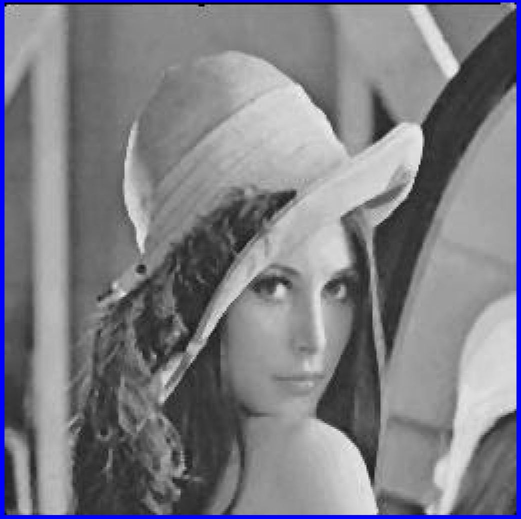

After Gaussian Smoothing

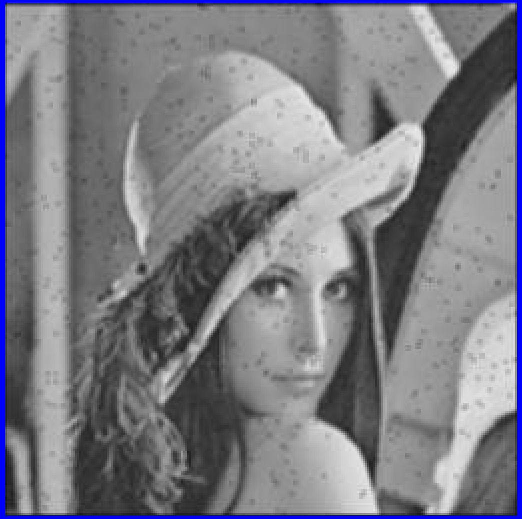
Median Filtering

- Averaging reduces the spike but spoils the neighbouring images
- It blurs the edges and other sharp details.
- Median filtering replaces the central pixel with the median of 3 x 3 pixel window
- This picks the “true” average value

Median vs. Averaging Filter


Salt & pepper noise


Median filter


Averaging filter

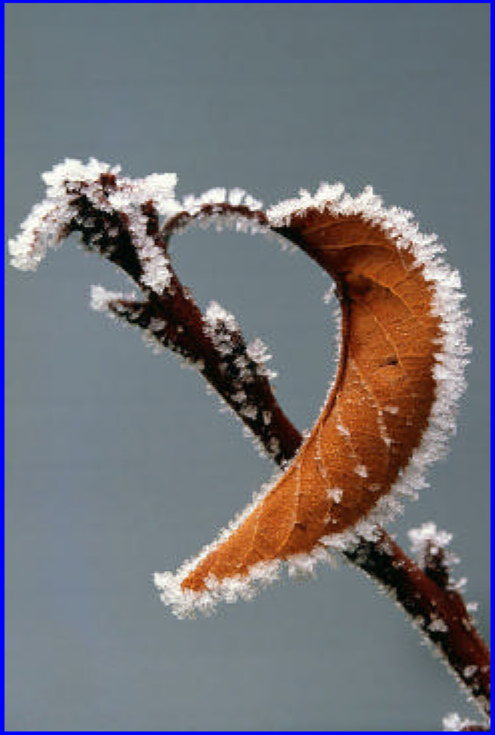
Edge Detection

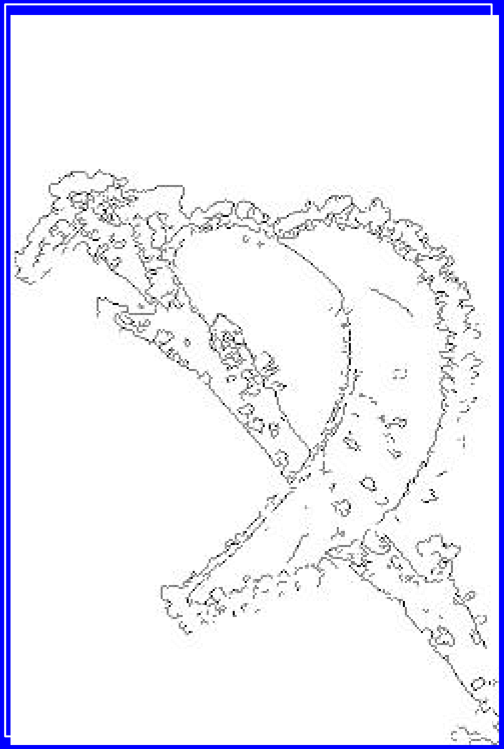
Edge Detection

- Find edges in the image
- Edges are locations where intensity changes the most
- Edges can be used to represent a shape of an object

Edge Detection in Images

- Finding the contour of objects in a scene

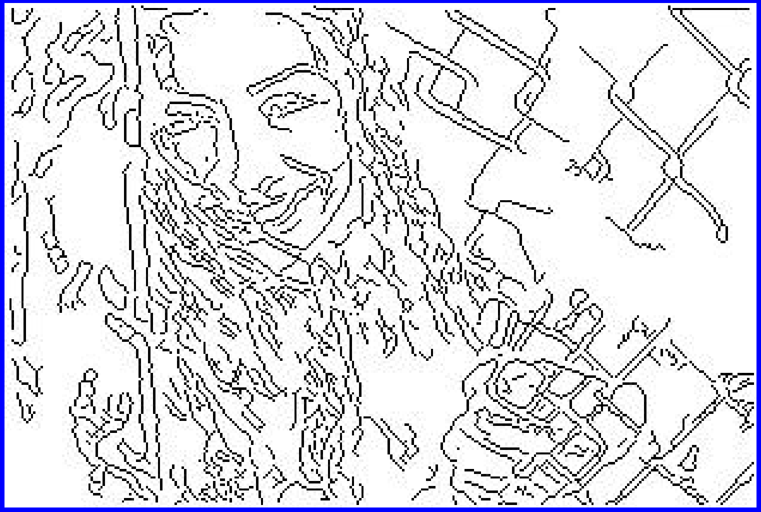




Edge Detection in Images

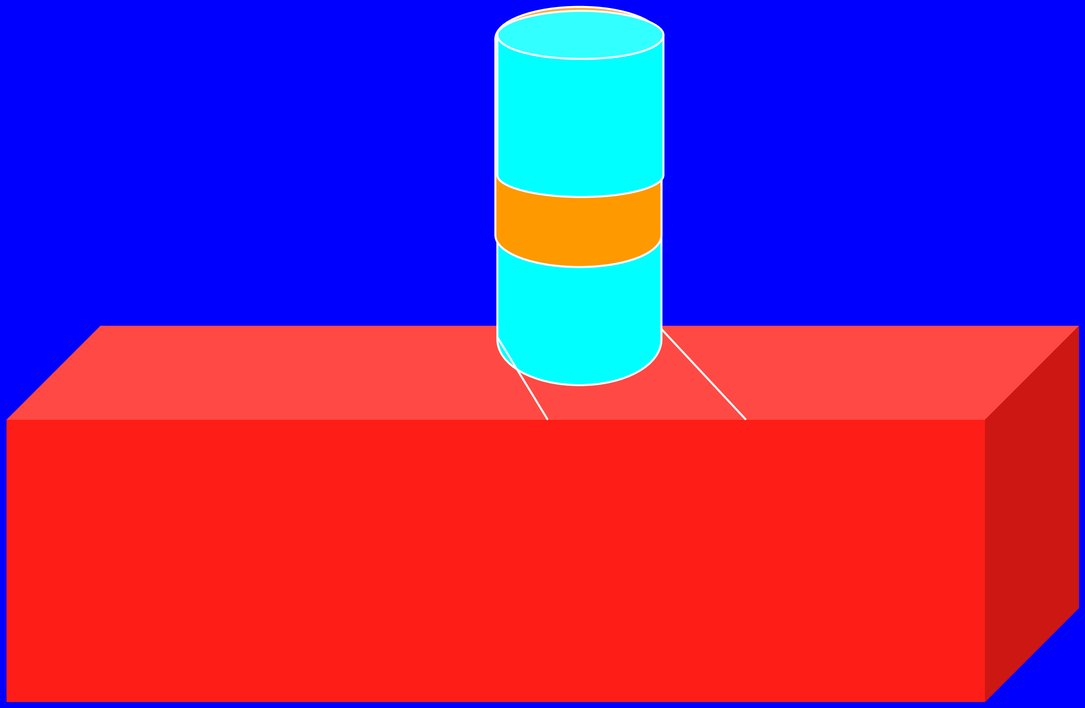
- What is an object?

  It is one of the goals of computer vision to identify objects in scenes.



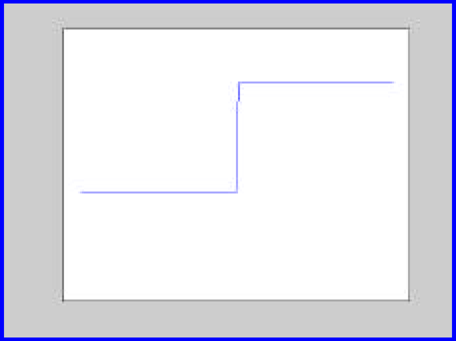
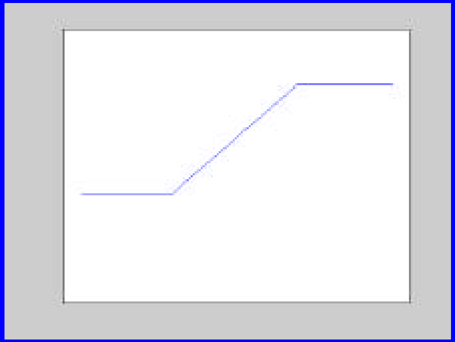
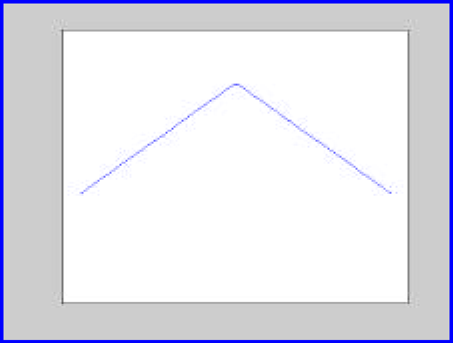
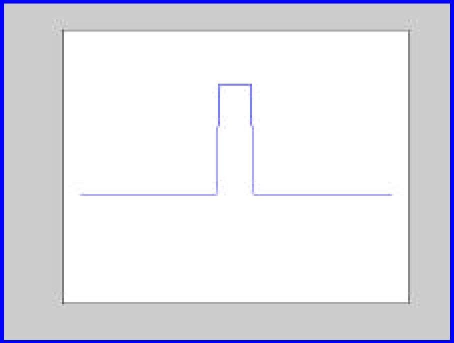
Edge Detection in Images

- Edges have different sources.



What is an Edge

- Lets define an edge to be a discontinuity in image intensity function.
- Edge Models
  - Step Edge
  - Ramp Edge
  - Roof Edge
  - Spike Edge

Detecting Discontinuities

- Discontinuities in signal can be detected by computing the derivative of the signal.
- If the signal is constant (over space), its derivative will be zero
- If there is a sharp difference in signal , then it will produce a high derivative value.

Differentiation and convolution

- Recall

$$\frac{\partial f}{\partial x} = \lim_{e \to 0}\left(\frac{f(x+e) - f(x)}{e}\right)$$

- Now this is linear and shift invariant, so must be the result of a convolution.

- We could approximate this as

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

(which is obviously a convolution with Kernel $\begin{bmatrix} 1 & -1 \end{bmatrix}$; it's not a very good way to do things, as we shall see)

Finite Difference in 2D

$$\frac{\partial f(x,y)}{\partial x} = \lim_{e \to 0}\left(\frac{f(x+e,y) - f(x,y)}{e}\right)$$

$$\frac{\partial f(x,y)}{\partial y} = \lim_{e \to 0}\left(\frac{f(x,y+e) - f(x,y)}{e}\right)$$

Definition

$$\frac{\partial f(x,y)}{\partial x} \approx \frac{f(x_{n+1},y_m) - f(x_n,y_m)}{\Delta x}$$

$$\begin{bmatrix} 1 & -1 \end{bmatrix}$$

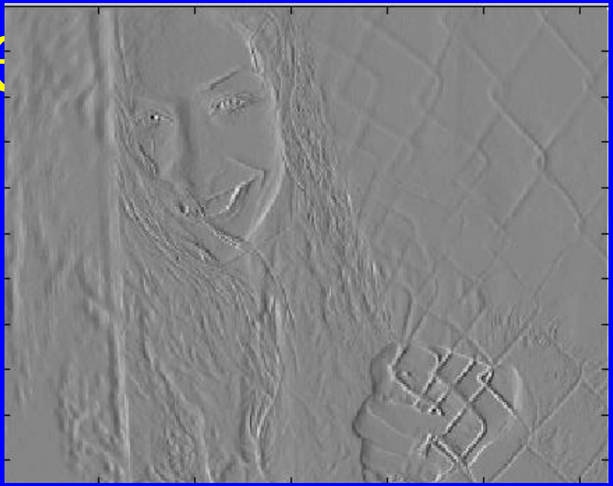
$$\frac{\partial f(x,y)}{\partial y} \approx \frac{f(x_n,y_{m+1}) - f(x_n,y_m)}{\Delta x}$$

$$\begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

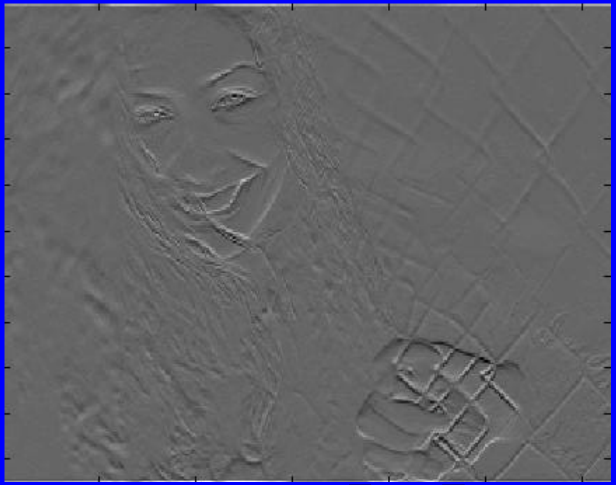
Discrete Approximation

Convolution Kernels

Finite differe



$$I_x = I * \begin{bmatrix} 1 & -1 \end{bmatrix}$$



$$I_y = I * \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

I

Edge Detectors

- Prewit
- Sobel
- Roberts
- Marr-Hildreth (Laplacian of Gaussian)
- Canny (Gradient of Gaussian)
- Haralick (Facet Model)

Discrete Derivative

$$\frac{df}{dx} = \lim_{\Delta x \to 0} \frac{f(x) - f(x - \Delta x)}{\Delta x} = f'(x)$$

$$\frac{df}{dx} = \frac{f(x) - f(x-1)}{1} = f'(x)$$

$$\frac{df}{dx} = f(x) - f(x-1) = f'(x)$$

(Finite Difference)

Discrete Derivative

$$\frac{df}{dx} = f(x) - f(x-1) = f'(x)$$ Left difference

$$\frac{df}{dx} = f(x) - f(x+1) = f'(x)$$ Right difference

$$\frac{df}{dx} = f(x+1) - f(x-1) = f'(x)$$ Center difference

Derivatives in Two Dimensions

$$f(x,y)$$

(partial Derivatives)

$$\frac{\partial f}{\partial x} = f_x = \lim_{\Delta x \to 0} \frac{f(x,y) - f(x-\Delta x, y)}{\Delta x}$$

$$\frac{\partial f}{\partial y} = f_y = \lim_{\Delta y \to 0} \frac{f(x,y) - f(x, y-\Delta y)}{\Delta y}$$

(f_x, f_y) Gradient Vector

$$\text{magnitude} = \sqrt{(f_x^2 + f_y^2)}$$

$$\text{direction} = q = \tan^{-1} \frac{f_y}{f_x}$$

$$\Delta^2 f = f_{xx} + f_{yy} = \text{Laplacian}$$

Derivatives of an Image(along x)

Derivative & average

$$\begin{matrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{matrix} \qquad \begin{matrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{matrix}$$

$$f_x \qquad\qquad f_y$$

Prewit

$$I(x,y) = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix} \qquad I_x = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 30 & 30 & 0 & 0 \\ 0 & 30 & 30 & 0 & 0 \\ 0 & 30 & 30 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Derivatives of an Image (along y)

$$I(x,y) = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix} \qquad I_y = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Derivatives of an Image

P
Sobel

$$\begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix} \qquad \begin{matrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{matrix}$$

$$f_x \qquad\qquad f_y$$

$$\begin{matrix} 0 & 1 \\ -1 & 0 \end{matrix} \qquad \begin{matrix} 1 & 0 \\ 0 & -1 \end{matrix}$$

Roberts

$$f_x \qquad\qquad f_y$$

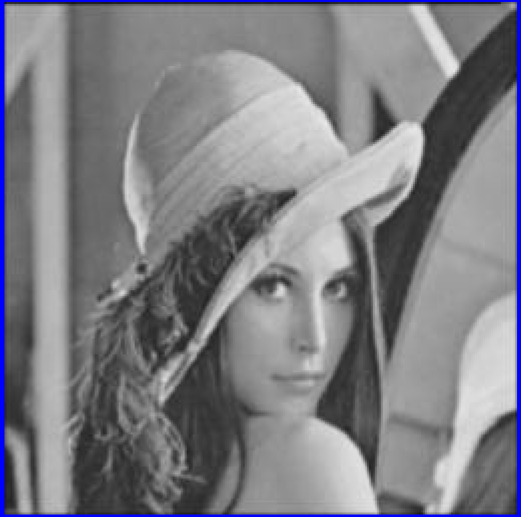
Detecting Edges in Image

- Sobel Edge Detector

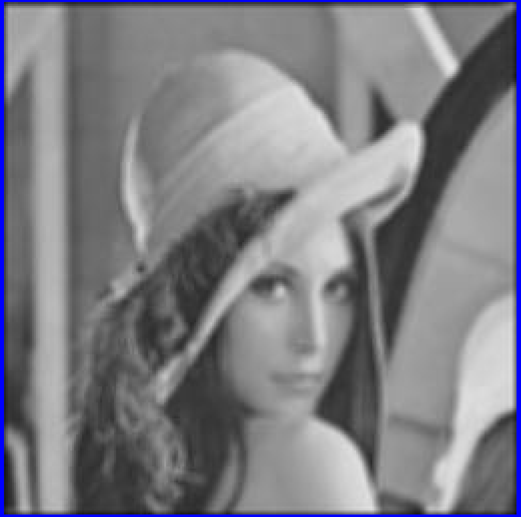
$$\begin{tikzcd}[ampersand replacement=\&]
\& \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \arrow[dr, "\frac{d}{dx}I"] \& \& \& \\
\textit{Image } I \arrow[ur, "*"] \arrow[dr, "*"'] \& \& \bigcirc \arrow[r, "\sqrt{\left(\frac{d}{dx}I\right)^2+\left(\frac{d}{dy}I\right)^2}"] \& \boxed{\text{Threshold}} \arrow[r, "\text{Edges}"] \& {} \\
\& \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \arrow[ur, "\frac{d}{dy}I"'] \& \& \&
\end{tikzcd}$$

Image Filter

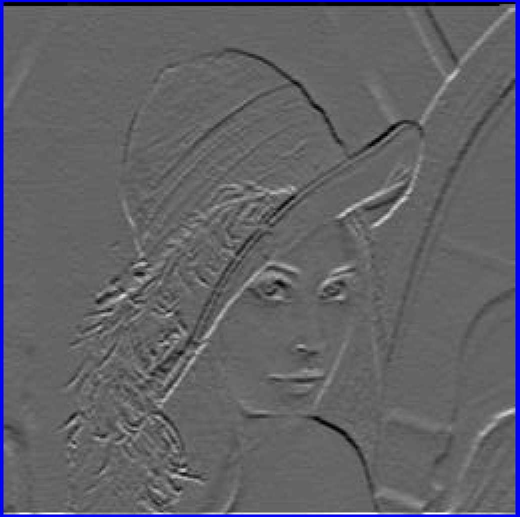
Original Image


Gaussian filter

$$\begin{bmatrix} 0.0113 & 0.0838 & 0.0113 \\ 0.0838 & 0.6193 & 0.0838 \\ 0.0113 & 0.0838 & 0.0113 \end{bmatrix}$$


Average filter

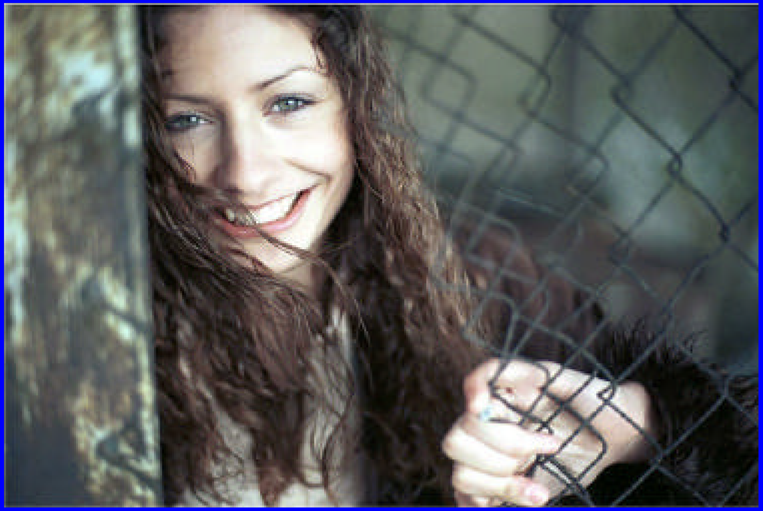
$$\tfrac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$


Sobel filter

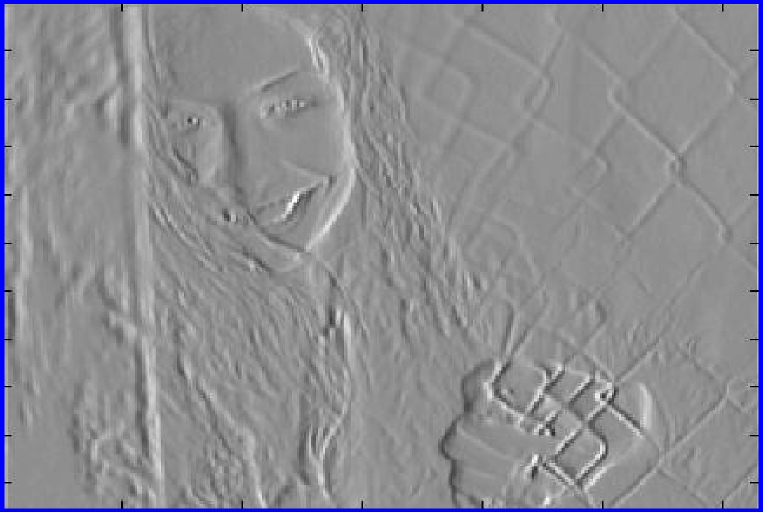
$$\tfrac{1}{8}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Sobel Edge Detector

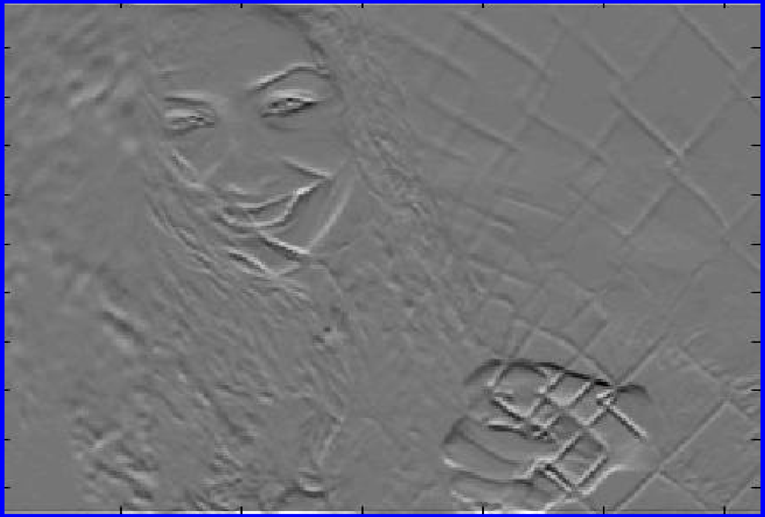
I



$\frac{d}{dx} I$

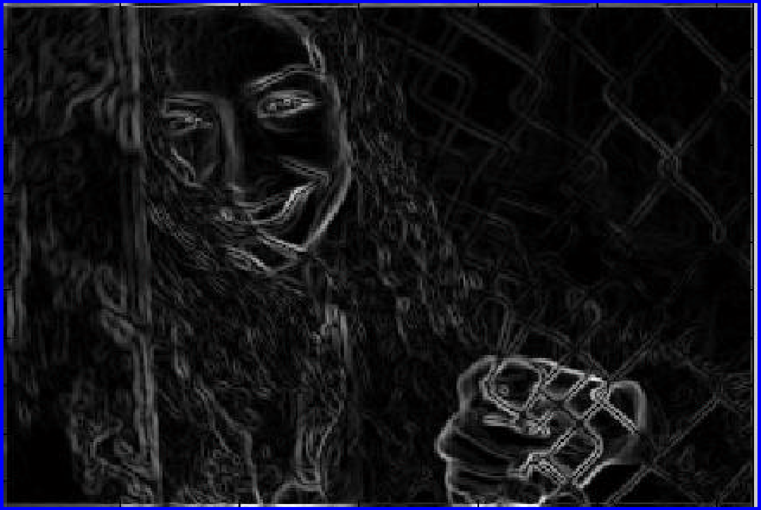


$\frac{d}{dy} I$

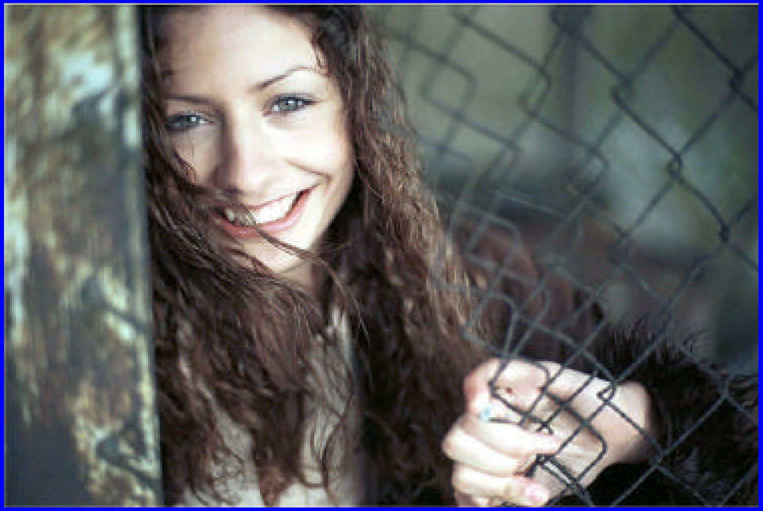


Sobel Edge Detector

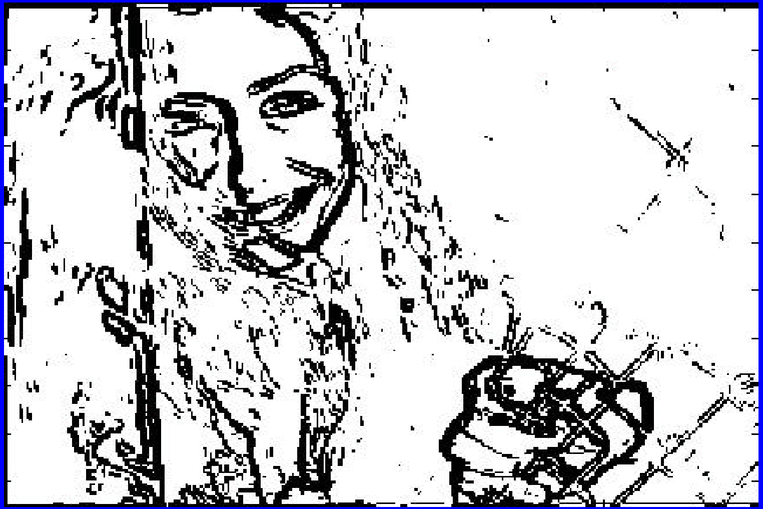
$$\Delta = \sqrt{\left(\frac{d}{dx} I\right)^2 + \left(\frac{d}{dy} I\right)^2}$$



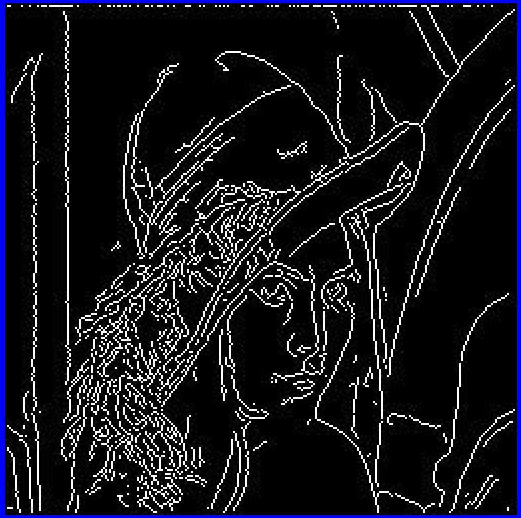
I



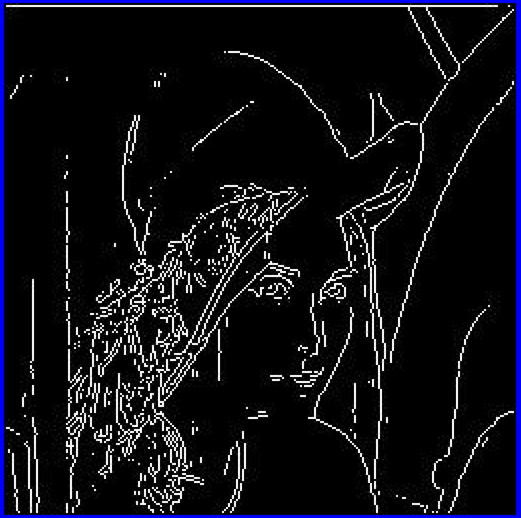
$\Delta \geq Threshold = 100$



Edge Detector

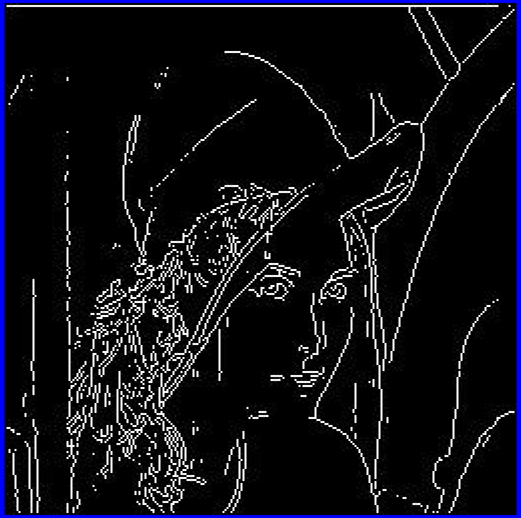


Canny edge detector using gaussian filter



Prewitt edge detector using Prewitt filter

$$\frac{1}{6}\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$



Sobel edge detector using Sobel filter

$$\frac{1}{8}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

High-boost Filtering

An image with sharp features implies that there will be high frequency component, which are ignored by averaging filter(A lowpass filter – which allows only low frequencies to go through).

Highpass = Original – lowpass

High-boost = A (original) – lowpass

= (A – 1) original + original – lowpass

= (A – 1) original + highpass

A can be chosen as 1.1, 1.15, ….. 1.2 (beyond that results no good)

Edge Detection in Noisy Images

- Images contain noise, need to remove noise by averaging, or weighted averaging
- To detect edges compute derivative of an image (gradient)
- If gradient magnitude is high at pixel, intensity change is maximum, that is an edge pixel
- If at a pixel the first derivative is maximum, the Laplacian (second derivative) would be zero and that point can be declared an   edge pixel.

Laplacian of Gaussian

- Filter the image by weighted averaging (Gaussian)
- Find Laplacian of image
- Detect zero-crossings

$$\Delta^2 f = f_{xx} + f_{yy} = \text{Laplacian}$$

Marr and Hildreth Edge Operator

- Smooth by Gaussian

$$S = G_s * I \qquad\qquad G_s = \frac{1}{\sqrt{2p\,s}}\,e^{-\frac{x^2+y^2}{2s^2}}$$

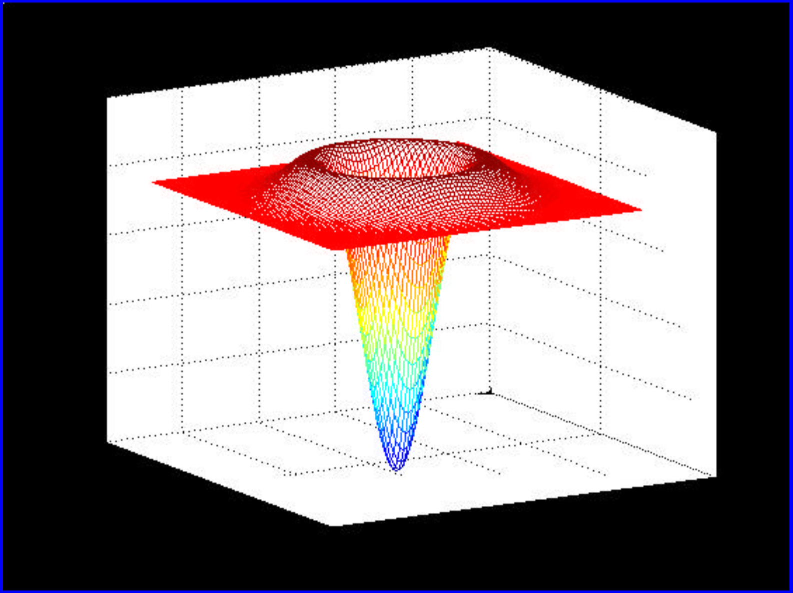
- Use Laplacian to find derivatives

$$\Delta^2 S = \frac{\partial^2}{\partial x^2} S + \frac{\partial^2}{\partial y^2} S$$

Marr and Hildreth Edge Operator

$$\Delta^2 S = \Delta^2 \left(G_s * I \right) = \Delta^2 G_s * I$$

$$\Delta^2 G_s = -\frac{1}{\sqrt{2p}\, s^3} \left(2 - \frac{x^2 + y^2}{s^2} \right) e^{\frac{x^2+y^2}{2s^2}}$$



Marr and Hildreth Edge Operator

$$\Delta^2 G_s = -\frac{1}{\sqrt{2p}\,s^3}\left(2-\frac{x^2+y^2}{s^2}\right)e^{-\frac{x^2+y^2}{2s^2}}$$

Y

X

0.0008	0.0066	0.0215	0.031	0.0215	0.0066	0.0008
0.0066	0.0438	0.0982	0.108	0.0982	0.0438	0.0066
0.0215	0.0982	0	-0.242	0	0.0982	0.0215
0.031	0.108	-0.242	-0.7979	-0.242	0.108	0.031
0.0215	0.0982	0	-0.242	0	0.0982	0.0215
0.0066	0.0438	0.0982	0.108	0.0982	0.0438	0.0066
0.0008	0.0066	0.0215	0.031	0.0215	0.0066	0.0008

The image can be convolved with Laplacian of Gaussian .

Mark the points with zero crossings.

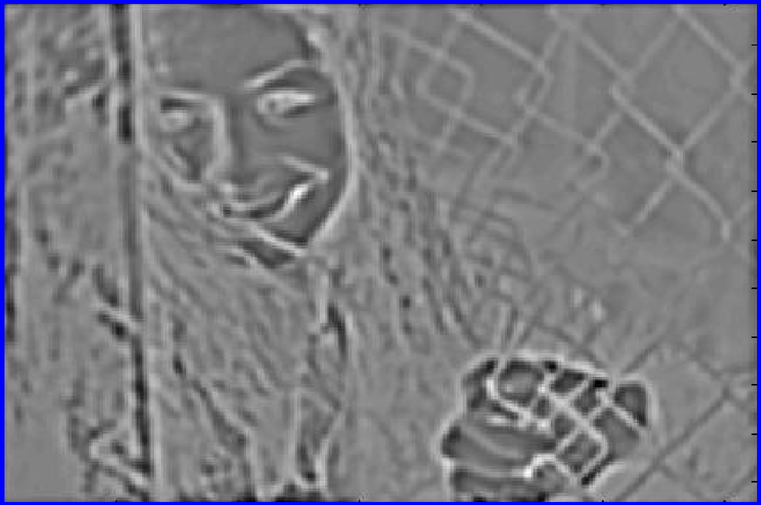
Verify that gradient Magnitudes are large here.

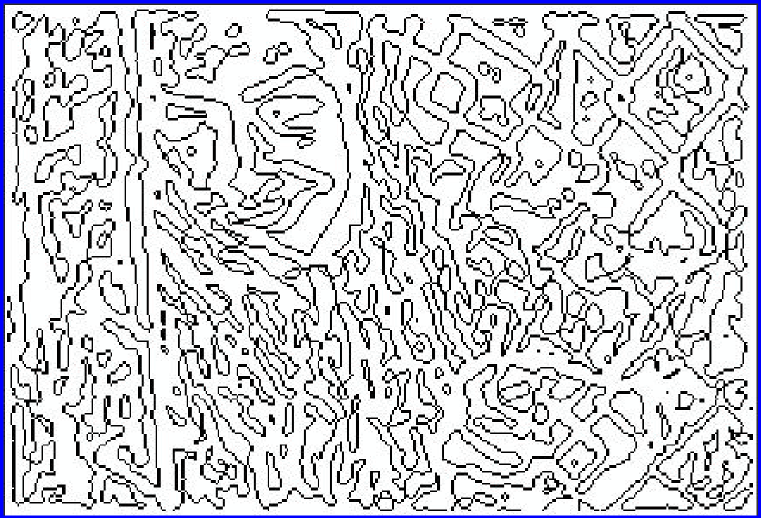
Response of L-o-G is positve on one side of an edge and negative on another.

Adding some percentage of this response to the original image yields a picture with sharpened edges.

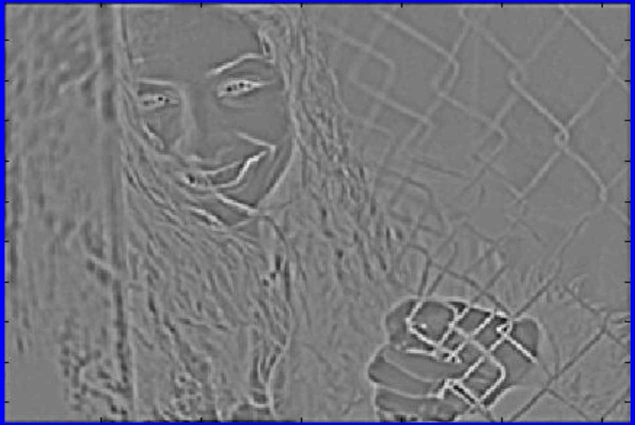
Marr and Hildreth Edge Operator

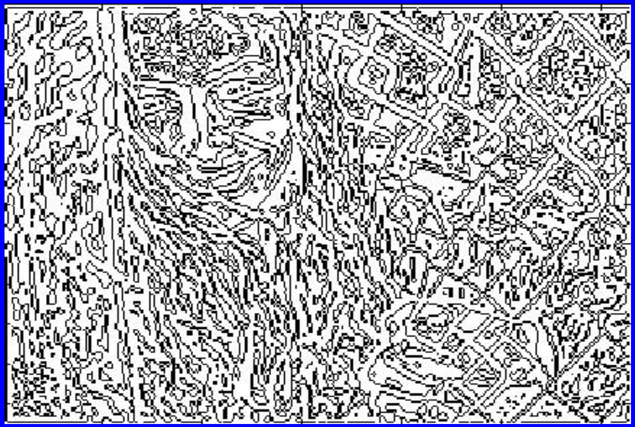
$$\begin{tikzcd}[column sep=huge] \text{Image } I \arrow[r] & \boxed{*\Delta^2 G_s} \arrow[r, "\Delta^2 G_s * I"] & \boxed{\begin{array}{l}\text{Zero Crossings}\\ \text{Detection}\end{array}} \arrow[r, "\text{Edge Image}"] & {} \end{tikzcd}$$


$\Delta^2 G_s * I$

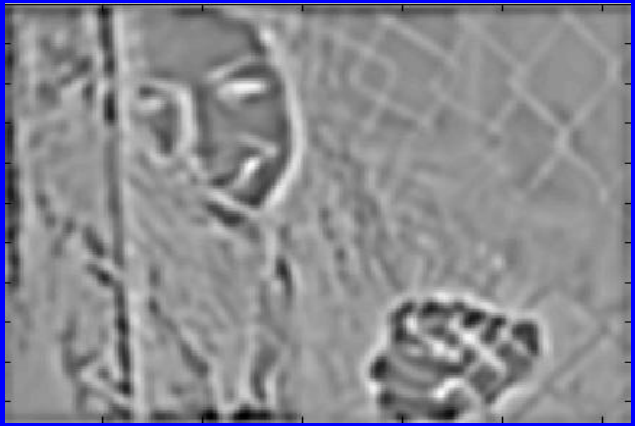

Zero Crossings

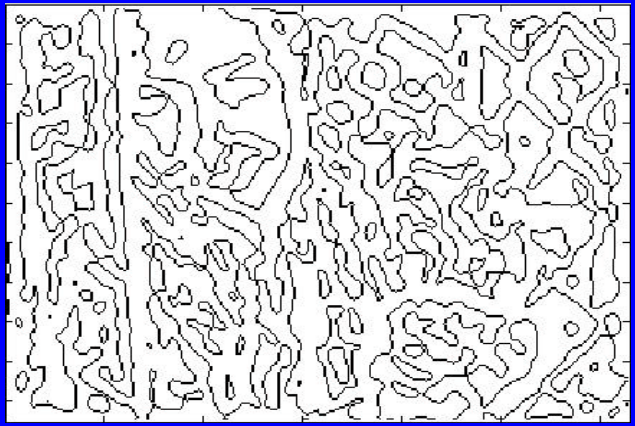
$s = 1$





$s = 3$





$s = 6$