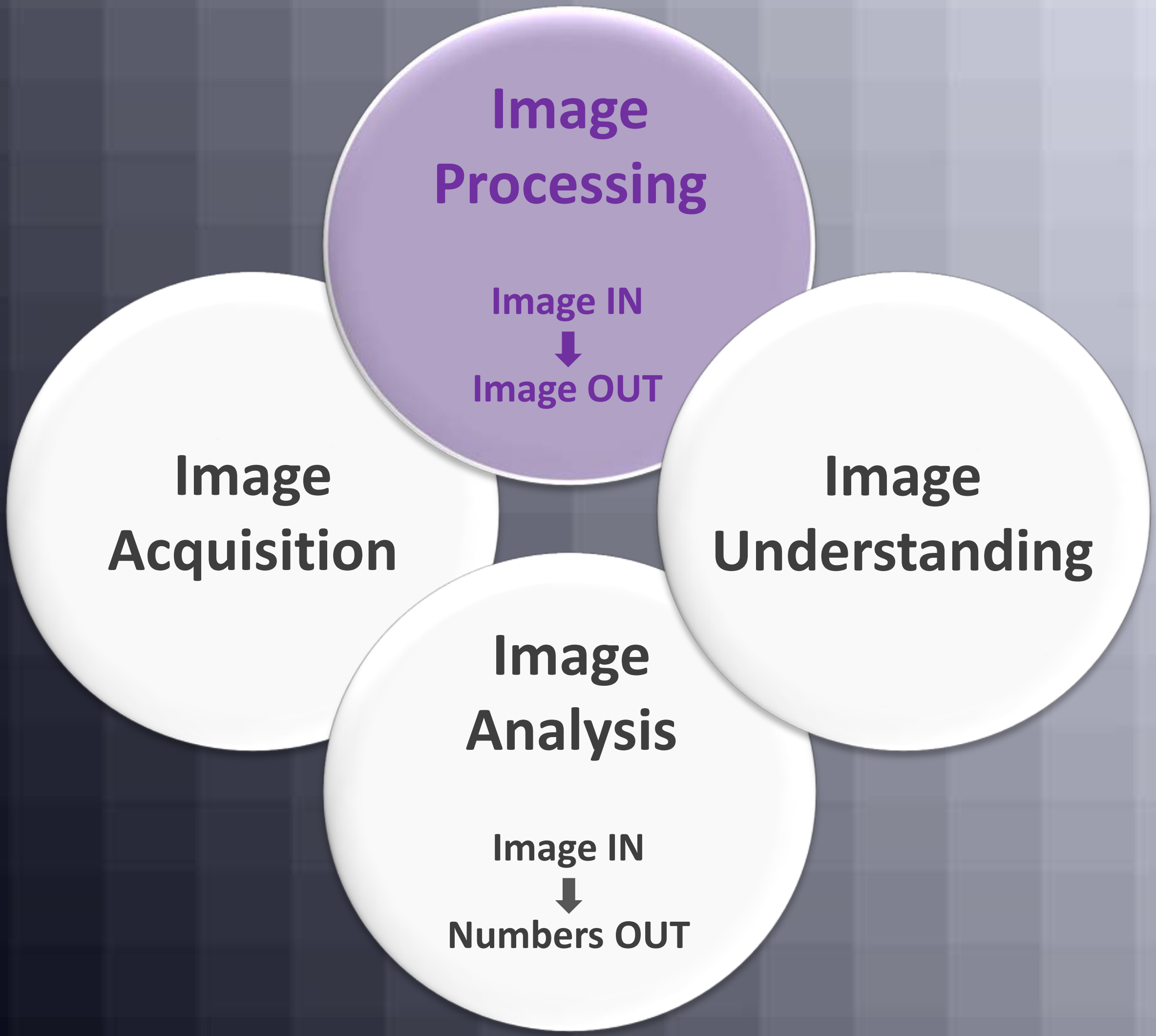


Image Processing

Image IN
⬇
Image OUT

Image Acquisition

Image Understanding

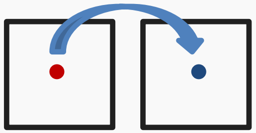
Image Analysis

Image IN
⬇
Numbers OUT

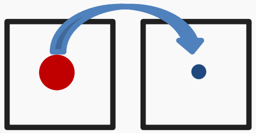
Image Processing

Basic Image Operations

Point Operations



Local Operations



Global Operations

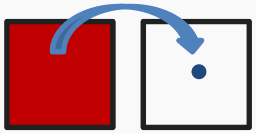


Image Processing

Local Operations

-How do local operations work?
- How do linear filters work?
- Examples of linear filters
- How do non-linear filters work?
- Examples of non-linear filters

How to…?

Local Operations

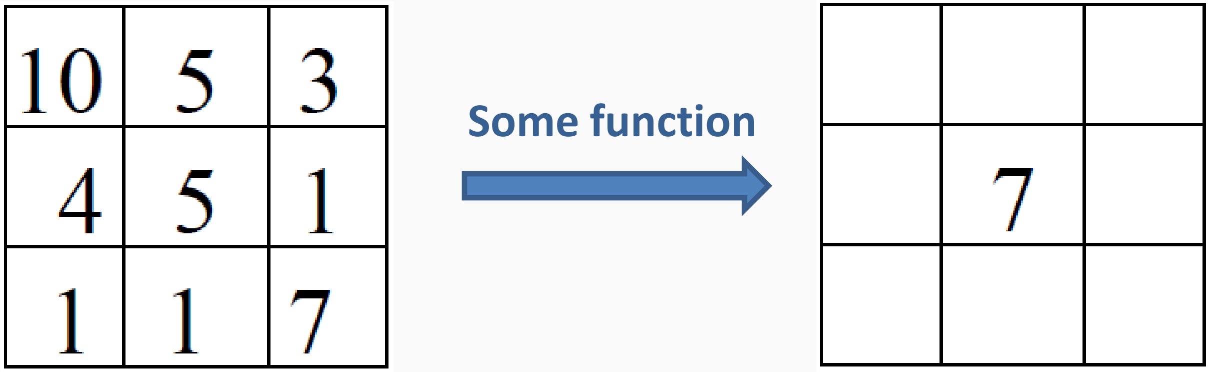
Point Operations

= all functions that are performed on each pixel of an image, independent of all other pixels in that image

$$I'(u,v) \longleftarrow f(I(u,v))$$

Local Operations – Neighborhood Operations - Filtering

= a pixel is modified based on some function of the pixel intensities in the neighborhood of this pixel



10	5	3
4	5	1
1	1	7

	7	

Image Data

Modified Image Data

Linear Filters

Replace each pixel by a linear combination of its neighbors (weighted sum).

10	5	3
4	5	1
1	1	7

Image Data

0	0	0
0	0.5	0
0	1	0.5

→

	7	

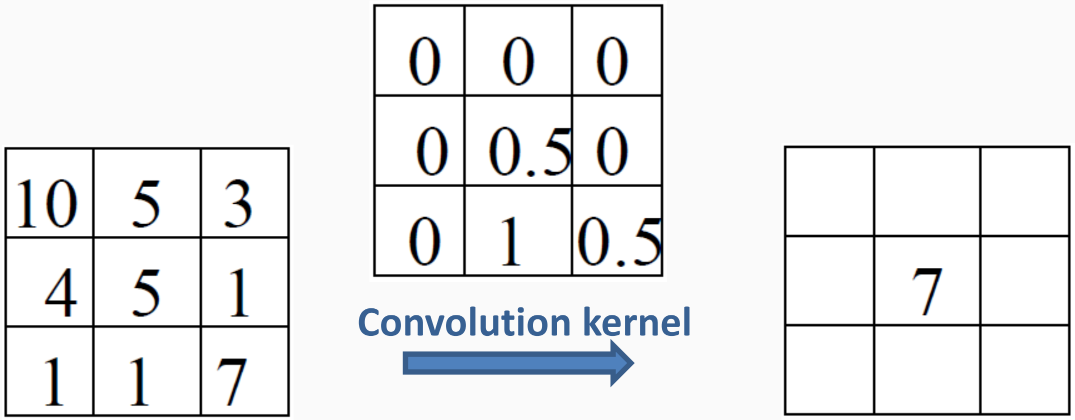
Modified Image Data

(10*0 + 5*0 + 3*0 + 4*0 + 5*0.5 + 1*0 + 1*0 + 1*1 + 7*0.5)

Local Operations

A ***convolution*** is a mathematical function that replaces each pixel by a weighted sum of its neighbors.

The prescription for the linear combination is called the “***convolution kernel***”.



10	5	3
4	5	1
1	1	7

Image Data

0	0	0
0	0.5	0
0	1	0.5

Convolution kernel

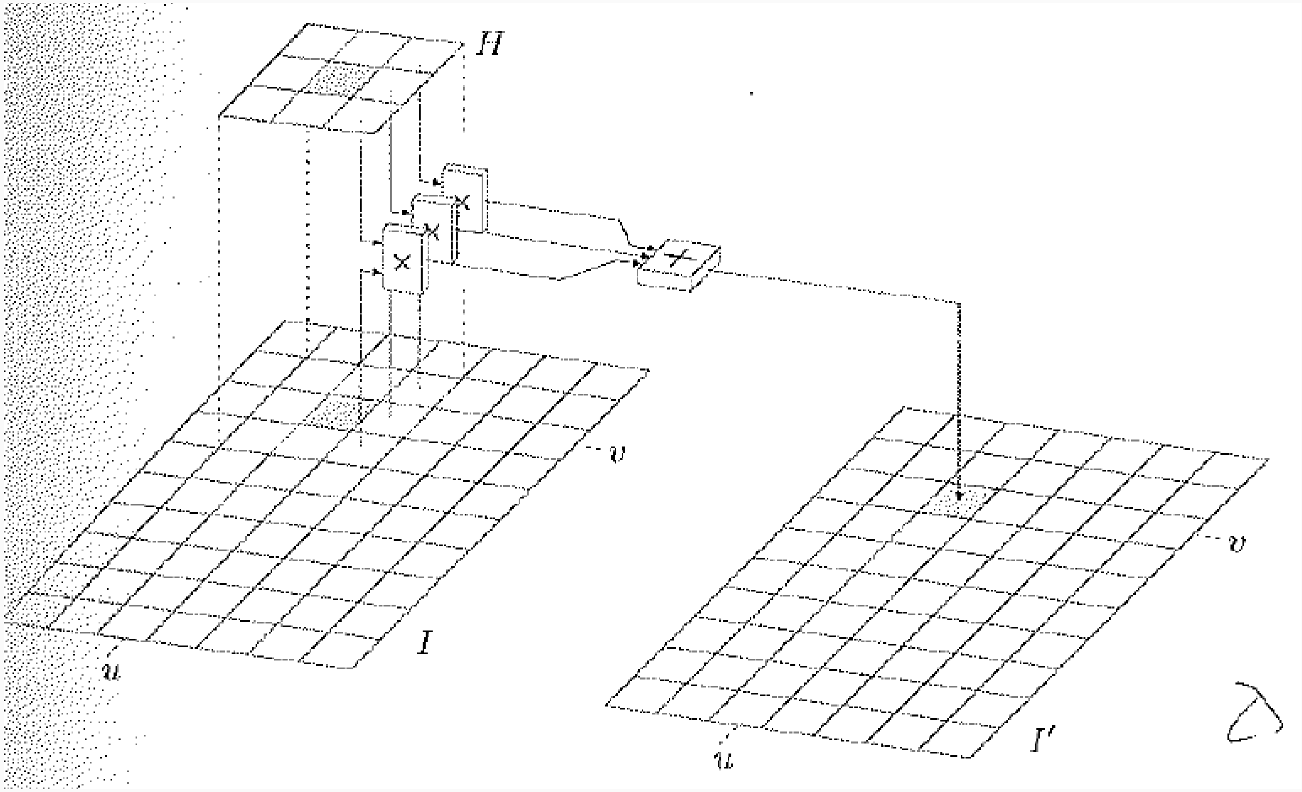
	7	

Modified Image Data

$$f[m,n] = I \otimes g = \sum_{k,l} I[m-k, n-l]\, g[k,l]$$

(10*0 + 5*0 + 3*0 + 4*0 + 5*0.5 + 1*0 + 1*0 + 1*1 + 7*0.5)

Local Operations



$$f[m,n] = I \otimes g = \sum_{k,l} I[m-k, n-l]g[k,l]$$

(10*0 + 5*0 + 3*0 + 4*0 + 5*0.5 + 1*0 + 1*0 + 1*1 + 7*0.5)

Linear Filters

Linear Filters

-Box Filter/Mean Filter

- Gauss Filter

- Sharpen Filter

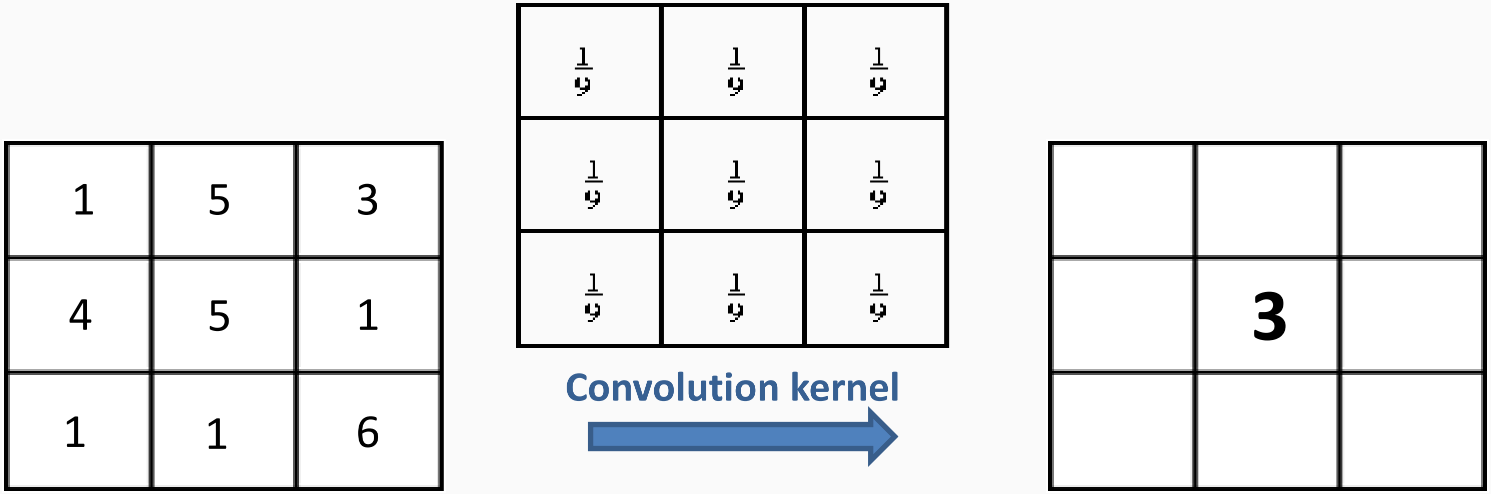
- Laplace Filter

Mean Filter

Box/Mean Filter

The value of a pixel is replaced by the mean of the pixel intensity in neighbor pixels

Average of a 3x3 matrix



1	5	3
4	5	1
1	1	6

Image Data

$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$
$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$
$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$

Convolution kernel

	3	

Modified Image Data

Mean Filter

Box/Mean Filter

The value of a pixel is replaced by the mean of the pixel intensity in neighbor pixels

$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$
$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$
$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$

$\frac{1}{9} *$

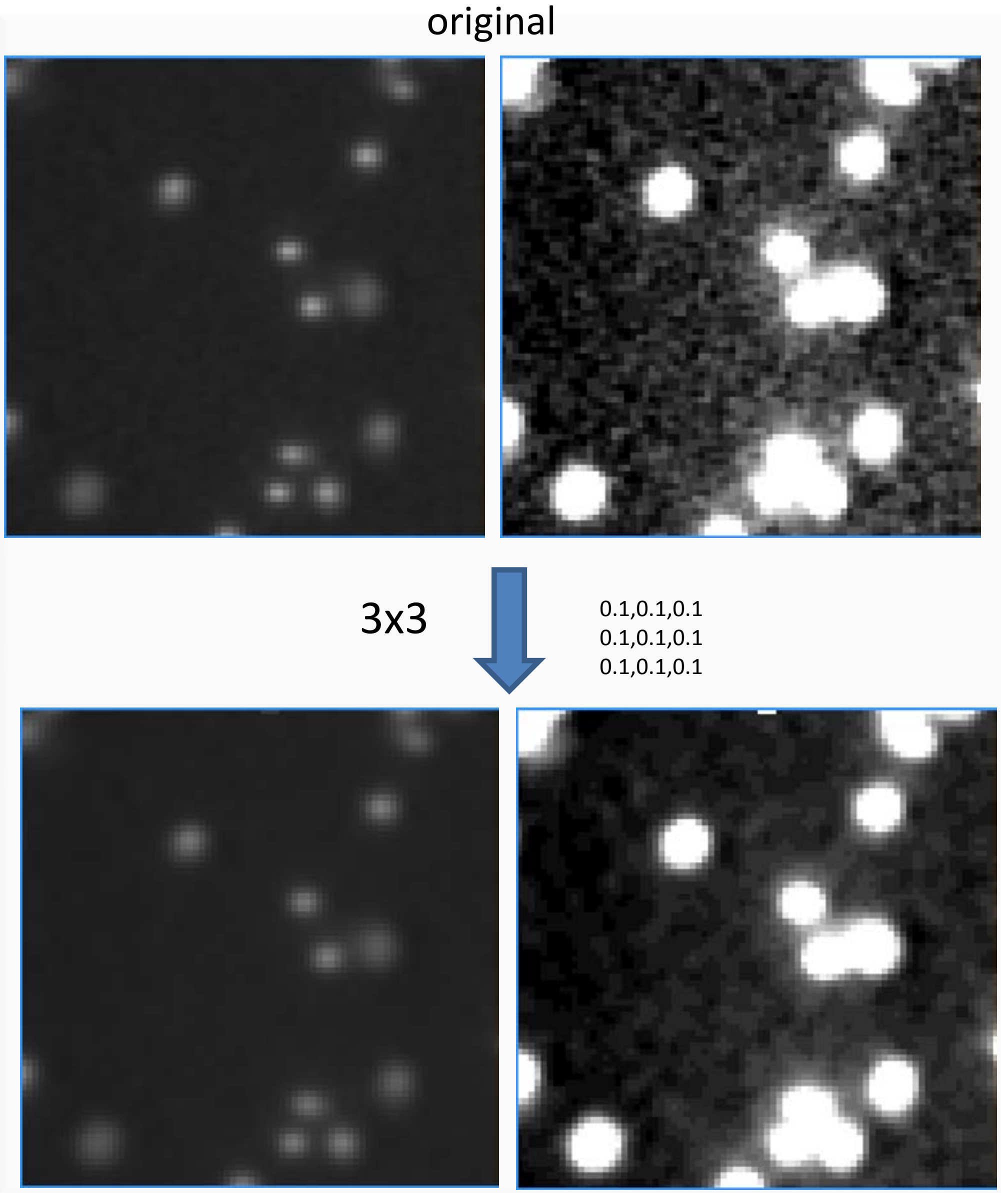
1	1	1
1	1	1
1	1	1

1	1	1
1	1	1
1	1	1

- low-pass filter
  = "smoothing"
  = a filter that passes low-frequency signals but attenuates (reduces the amplitude of) signals with higher frequencies
- kernel size influence

+ number of successive applications
+ simplest filter – fast

Mean Filter

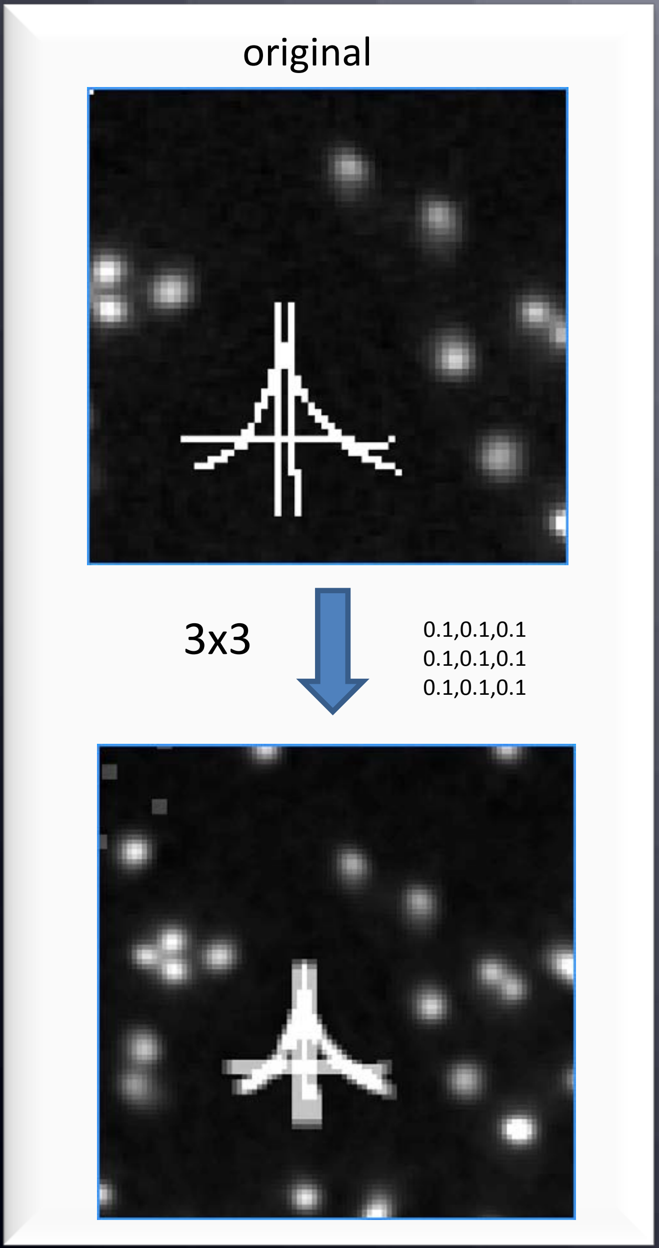


0.1,0.1,0.1
0.1,0.1,0.1
0.1,0.1,0.1

- low-pass filter (smoothes small objects)
- kernel size influence

+ number of successive applications
+ simplest filter – fast
+ **averages noise, does not eliminate it**
+ **works against Gaussian and Poisson noise**

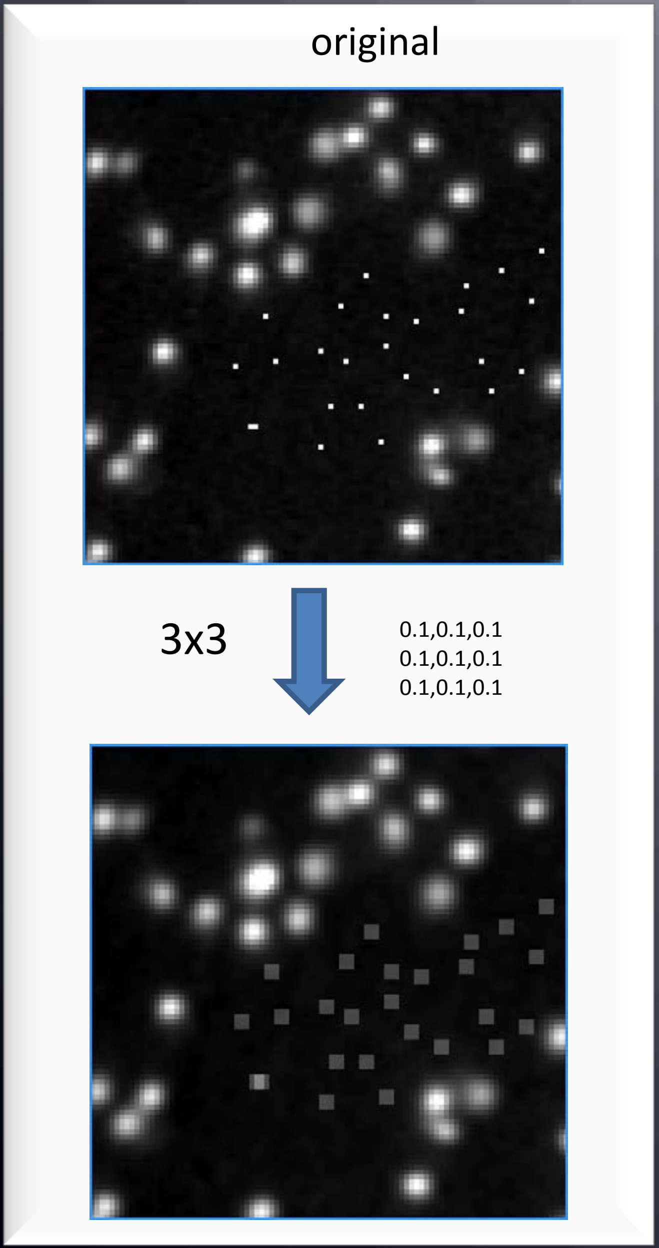
Mean Filter



- low-pass filter (smooths small objects)
- kernel size influence

+ number of successive applications
+ simplest filter – fast
+ averages noise, does not eliminate it
+ works against Gaussian and Poisson noise
– **blurs images – small details are lost (low pass filter)**
– **smoothes edges dramatically**

Mean Filter



- low-pass filter (smooths small objects)
- kernel size influence

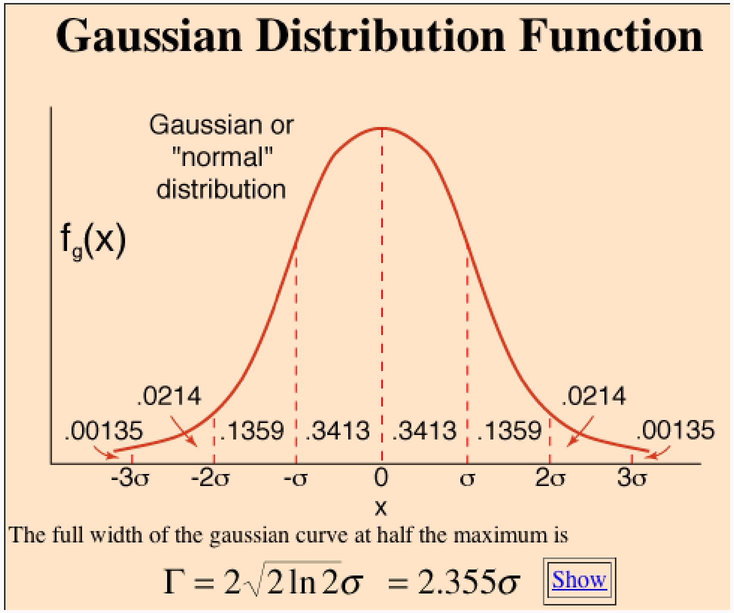
+ number of successive applications
+ simplest filter – fast
+ averages noise, does not eliminate it
+ works against Gaussian and Poisson noise
– blurs images – small details are lost (low pass filter)
– smoothes edges dramatically
– **fails for salt & pepper noise**

Gauss Filter

In probability theory, the **normal** (or **Gaussian**) **distribution**, is a continuous probability distribution that is often used as a first approximation to describe real-valued random variables that tend to cluster around a single mean value. The graph of the associated probability density function is “bell”-shaped, and is known as the Gaussian function or bell curve

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

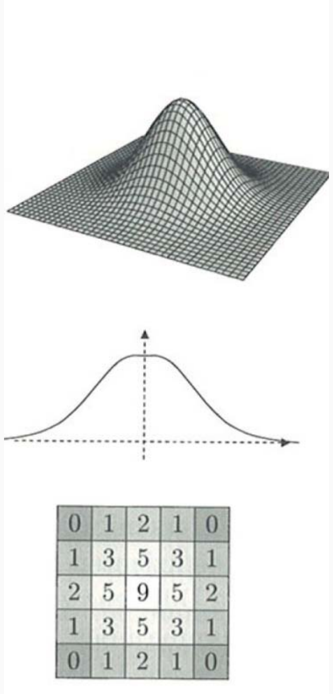
μ … *mean* (location of the peak)
σ^2 … the *variance* (the measure of the width of the distribution).



The full width of the gaussian curve at half the maximum is

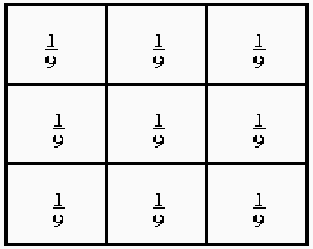
$$\Gamma = 2\sqrt{2\ln 2}\,\sigma \;= 2.355\sigma$$

Show



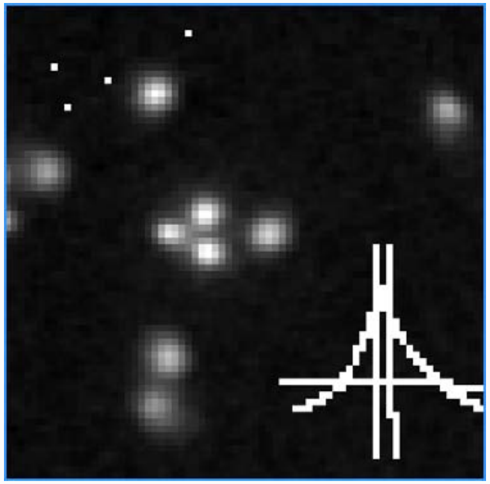
0	1	2	1	0
1	3	5	3	1
2	5	9	5	2
1	3	5	3	1
0	1	2	1	0

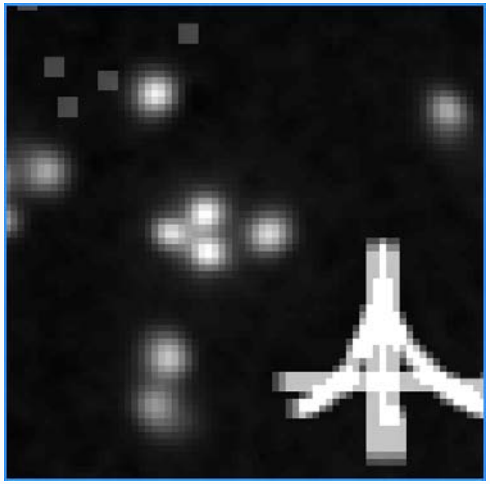
Gauss Filter

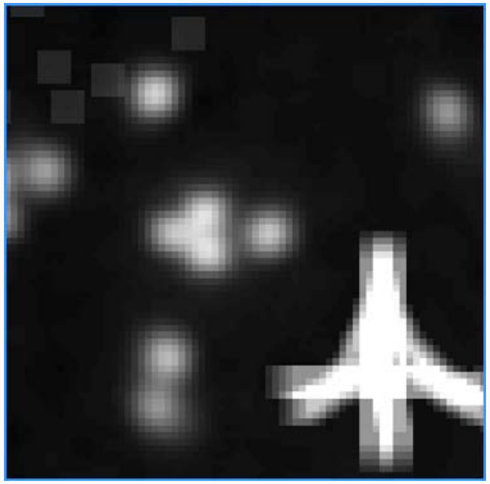


$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$
$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$
$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$

original





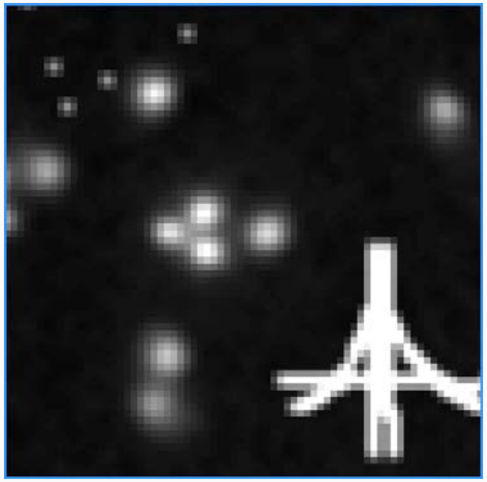


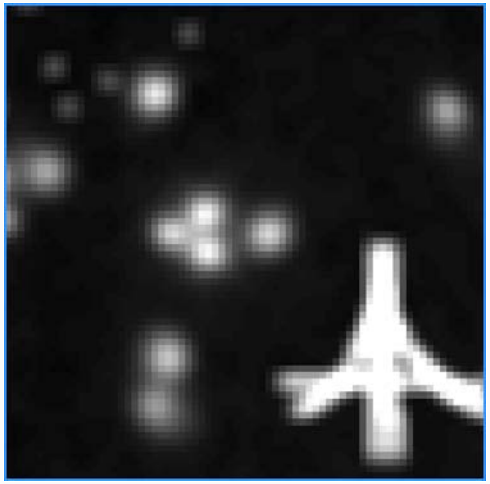
Mean Filter

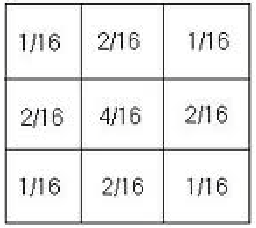
3x3

5x5

Gauss Filter







1/16	2/16	1/16
2/16	4/16	2/16
1/16	2/16	1/16

Sharpen Filter

Sharpen Filter

- Used to enhance the edges of objects by enhancing the changes of grayscale values that occur over a short distance.
- High Pass Filter (tends to retain the high frequency information within an image while reducing the low frequency information)

$$\begin{bmatrix} -1/9 & -1/9 & -1/9 \\ -1/9 & 8/9 & -1/9 \\ -1/9 & -1/9 & -1/9 \end{bmatrix}$$

Metamorph Sharpen Filter

Low

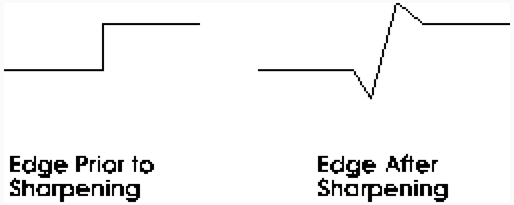
$$\begin{matrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{matrix}$$

Medium

$$\begin{matrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{matrix}$$

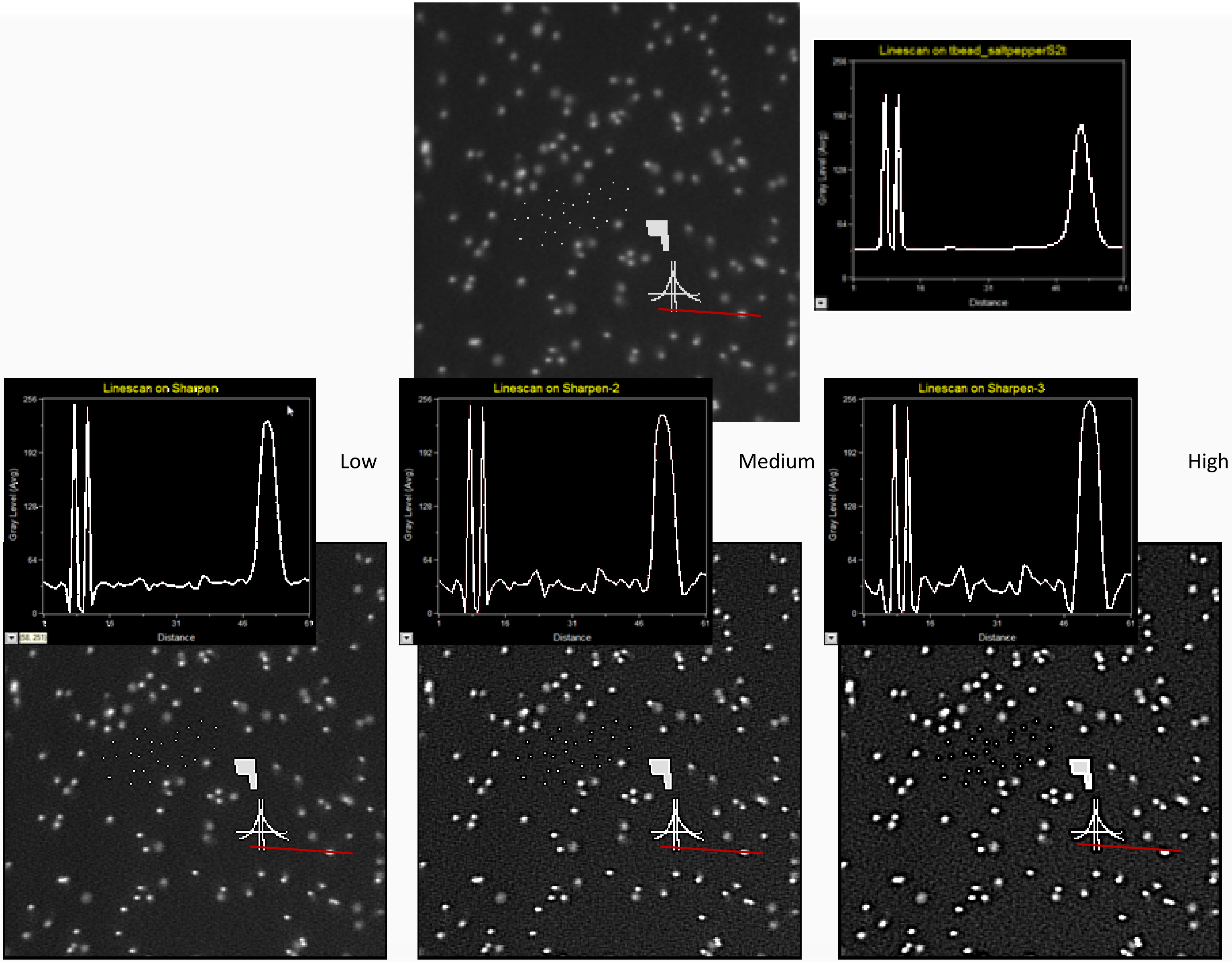
High

$$\begin{matrix} 0 & -1 & -1 & -1 & 0 \\ -1 & -2 & -2 & -2 & -1 \\ -1 & -2 & 31 & -2 & -1 \\ -1 & -2 & -2 & -2 & -1 \\ 0 & -1 & -1 & -1 & 0 \end{matrix}$$



Used for segmentation

Sharpen Filter



Low

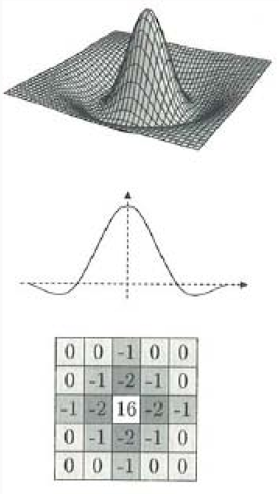
Medium

High

Laplace Filter

Laplace Filter

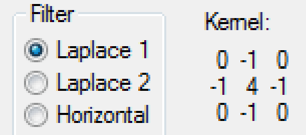
- “edge detection filter”



0	0	-1	0	0
0	-1	-2	-1	0
-1	-2	16	-2	-1
0	-1	-2	-1	0
0	0	-1	0	0

Metamorph Laplace Filter

Laplace 1



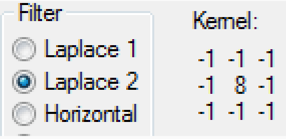
Filter

● Laplace 1
○ Laplace 2
○ Horizontal

Kernel:

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

Laplace 2



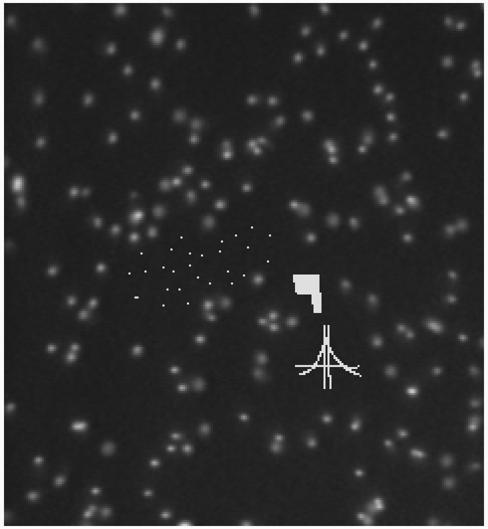
Filter

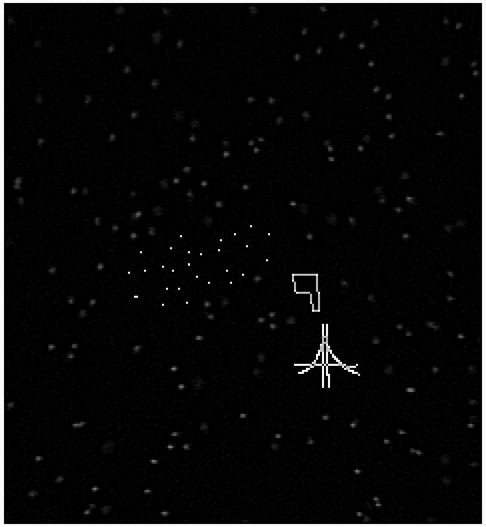
○ Laplace 1
● Laplace 2
○ Horizontal

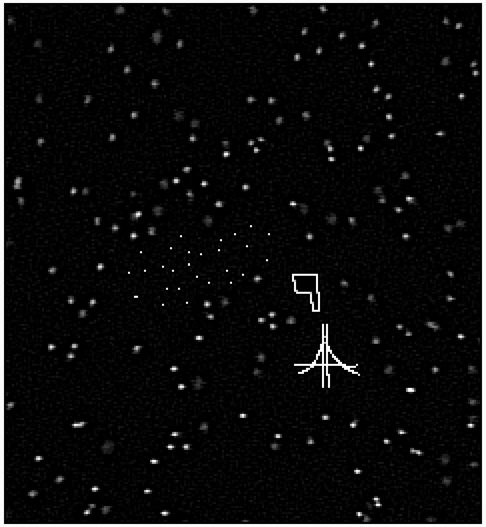
Kernel:

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

original







Combined Filters

Combined Filter

especially filters designed for edge detection are often a series of different kernels

Examples:

Prewitt

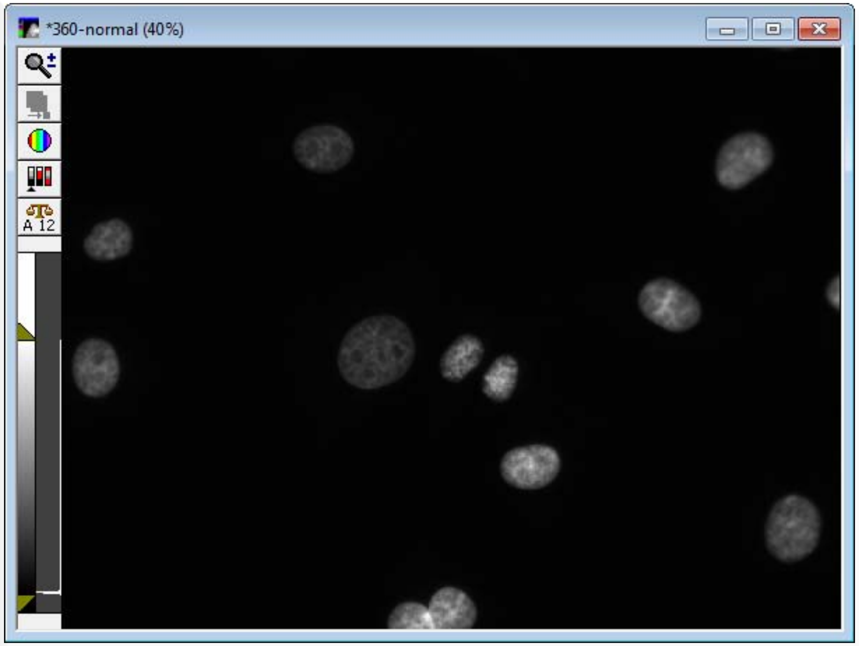
1	1	1		-1	0	1
0	0	0		-1	0	1
-1	-1	-1		-1	0	1
First Pass				Second Pass		

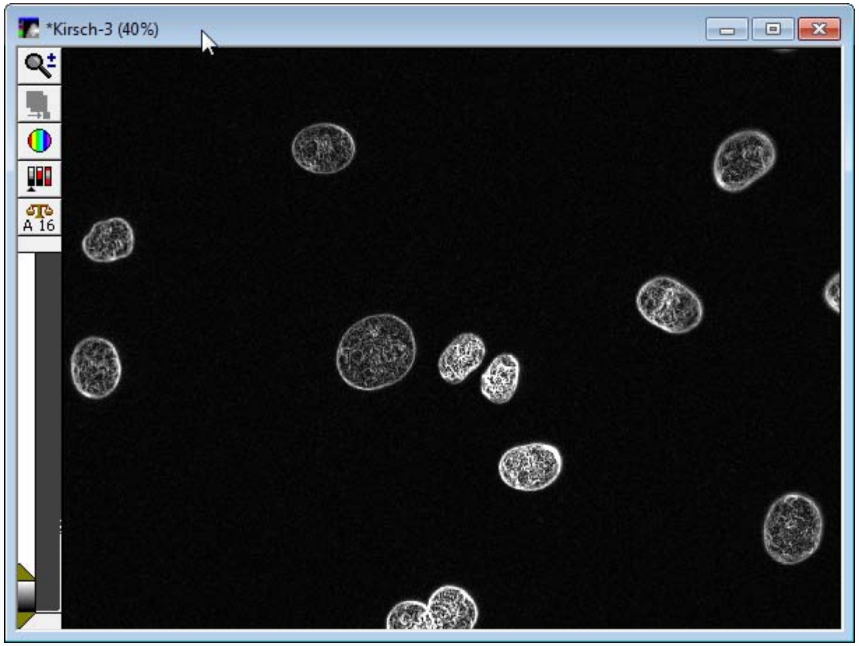
Sobel

1	2	1		-1	0	1
0	0	0		-2	0	2
-1	-2	-1		-1	0	1
First Pass				Second Pass		

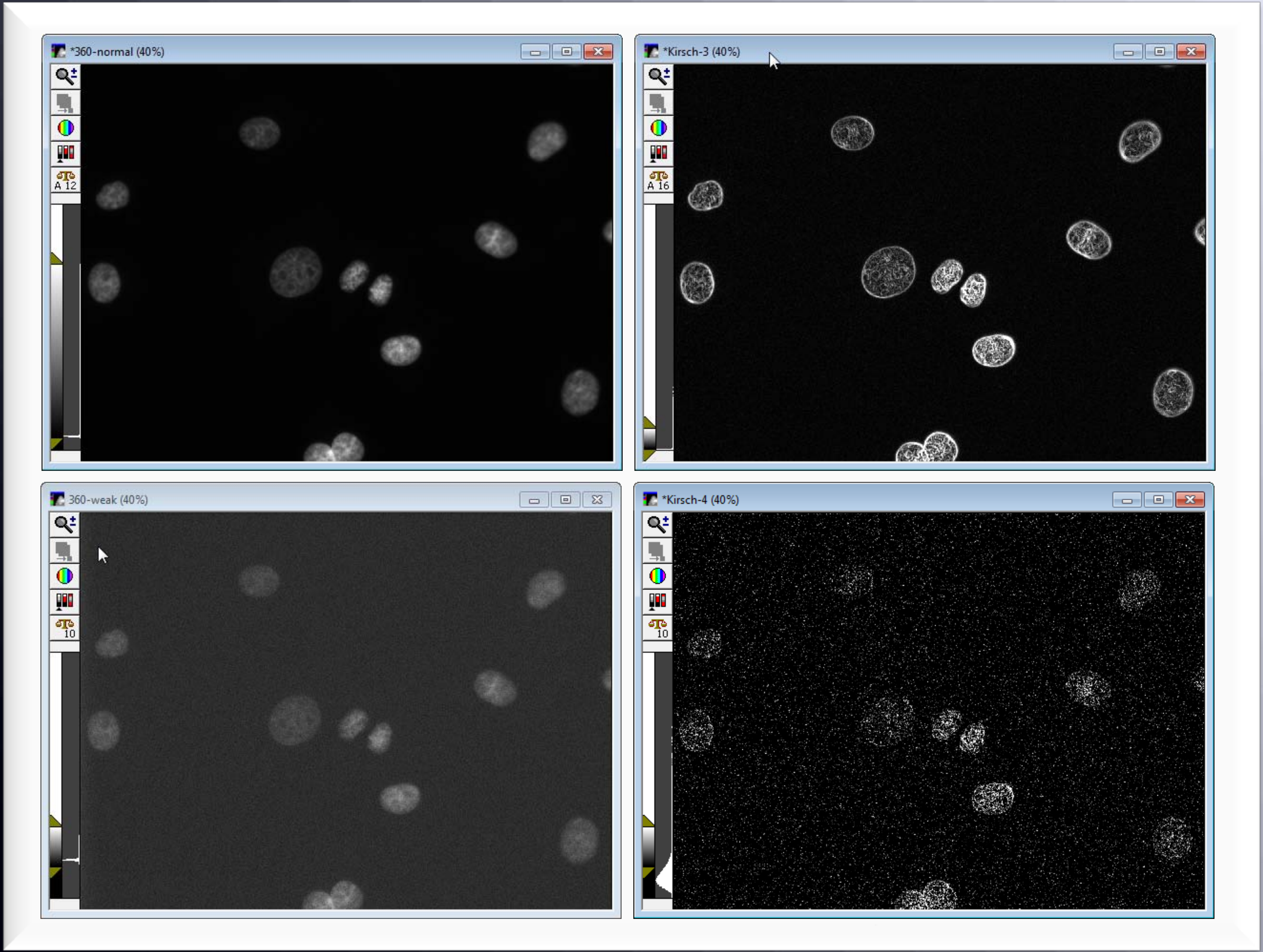
Kirsch

5	5	5		5	5	-3
-3	0	-3		5	0	-3
-3	-3	-3		-3	-3	-3
First Pass				Second Pass		
5	-3	-3		-3	-3	-3
5	0	-3		5	0	-3
5	-3	-3		5	5	-3
Third Pass				Fourth Pass		





Combined Filters



Non-Linear Filters

Non-Linear Filters

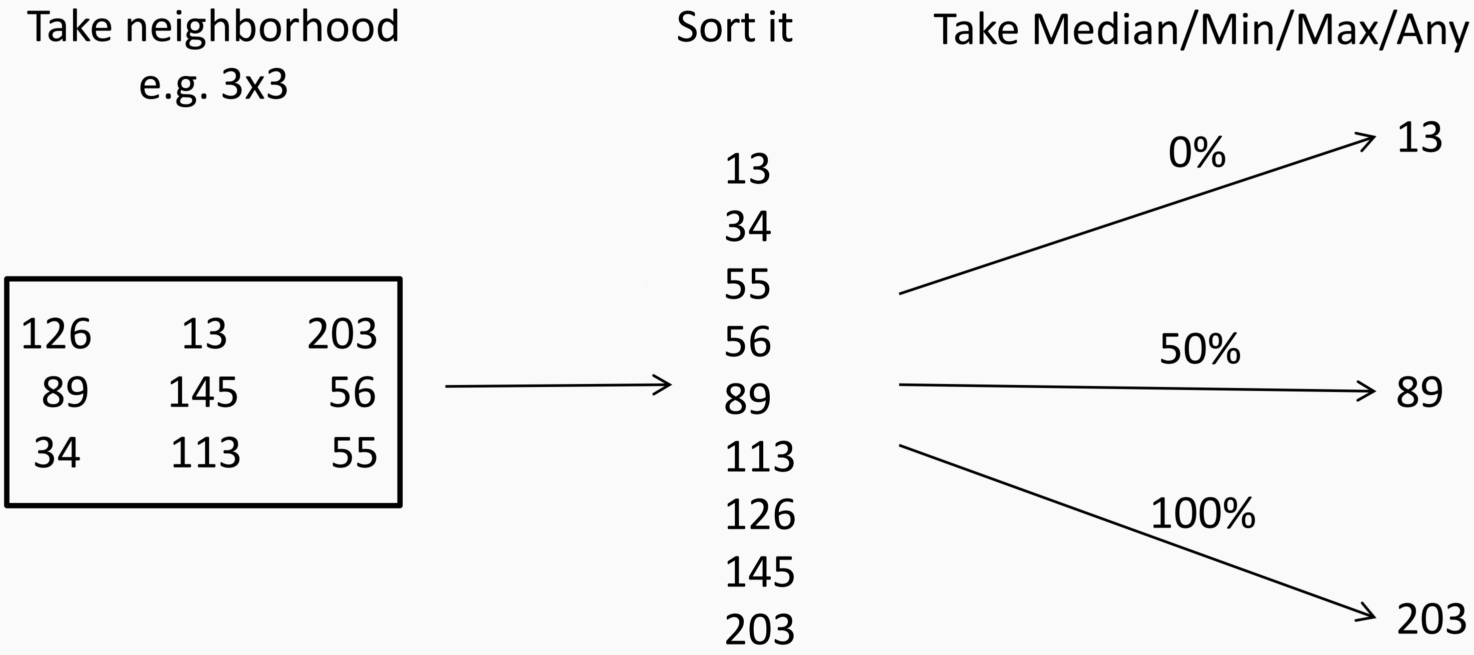
The output is not a linear function of its input.

Can produce different effects on signal and noise under certain circumstances.

- Rank Filter
    - Median Filter
    - Min/Max Filter

Rank Filter

The value of a pixel is replaced a specified value of the pixel intensity in neighbor pixels



Take neighborhood
e.g. 3x3

126	13	203
89	145	56
34	113	55

Sort it

13
34
55
56
89
113
126
145
203

Take Median/Min/Max/Any

0% → 13

50% → 89

100% → 203

Median Filter

Original:

5	9	6	6	9	5	9
9	5	9	7	8	7	9
8	9	8	6	7	9	9
9	9	7	200	9	6	9
6	5	8	6	9	6	7
9	7	9	9	8	6	7
7	9	5	6	7	6	6

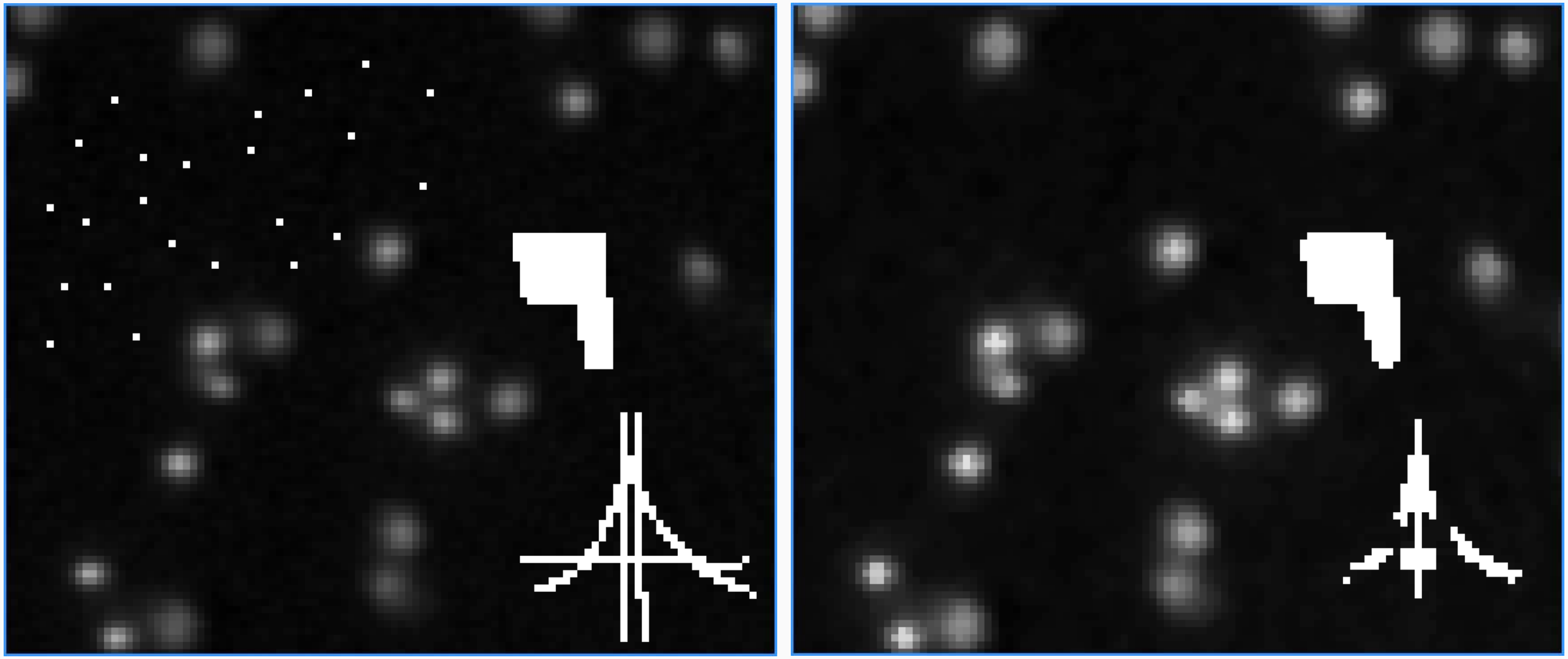
outlier

Median filtered:

0	5	6	6	6	7	0
5	8	7	7	7	9	7
8	9	8	8	7	9	7
6	8	8	8	7	9	6
6	8	8	9	8	7	6
6	7	7	8	6	7	6
0	7	6	6	6	6	0

The outlier value has been completely removed from the dataset

Median Filter



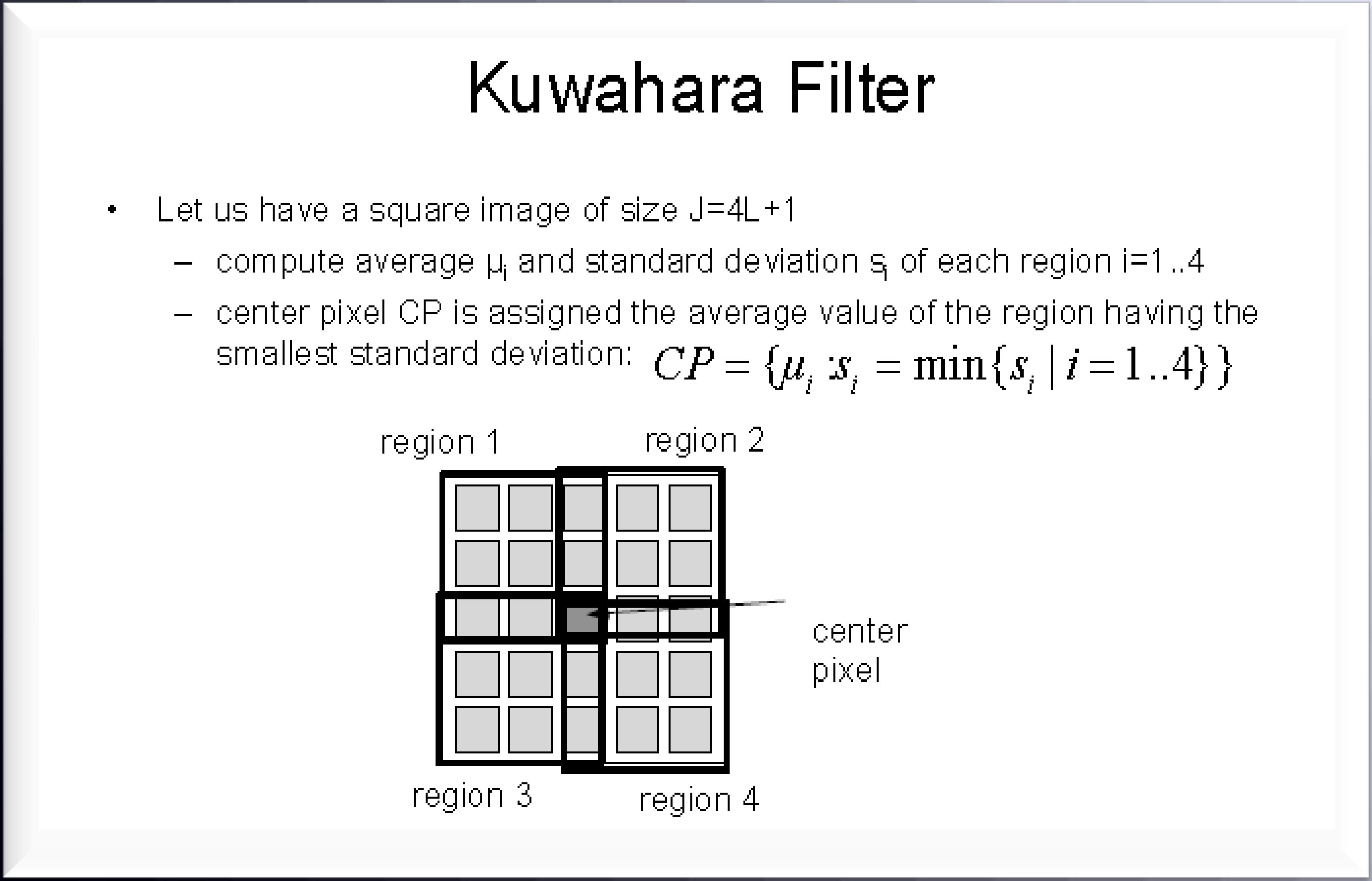
original

3x3 median filtered

+ Typically good for “Salt & pepper” noise removal
+ Eliminates noise
+ Edge-preserving

- Slower than mean (not such a problem anymore… computers are fast)
- NOT linear

Kuwahara Filter



Kuwahara Filter

- Let us have a square image of size J=4L+1
  - compute average μ_i and standard deviation s_i of each region i=1..4
  - center pixel CP is assigned the average value of the region having the smallest standard deviation: $CP = \{\mu_i : s_i = \min\{s_i \mid i = 1..4\}\}$

region 1
region 2
center pixel
region 3
region 4

Exercise

Exercise

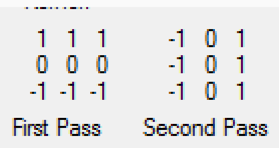
- use images 360-normal.tif, 360-weak.tif, 480-normal.tif, 480weak.tif and 360-weak-saltPepper

- explore different filters on low and high noise images, with fine and coarse structures – report results

- convolve images with different Kernels – see what happens

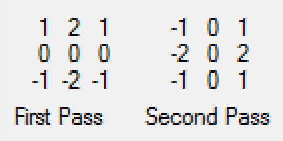
- Implement edge detection filters

Prewitt



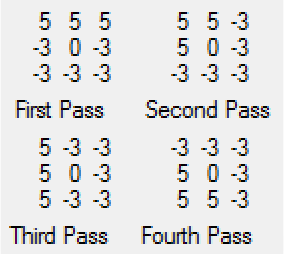
```
 1  1  1      -1  0  1
 0  0  0      -1  0  1
-1 -1 -1      -1  0  1
First Pass   Second Pass
```

Sobel



```
 1  2  1      -1  0  1
 0  0  0      -2  0  2
-1 -2 -1      -1  0  1
First Pass   Second Pass
```

Kirsch



```
 5  5  5       5  5 -3
-3  0 -3       5  0 -3
-3 -3 -3      -3 -3 -3
First Pass   Second Pass
 5 -3 -3      -3 -3 -3
 5  0 -3       5  0 -3
 5 -3 -3       5  5 -3
Third Pass   Fourth Pass
```