Filtering

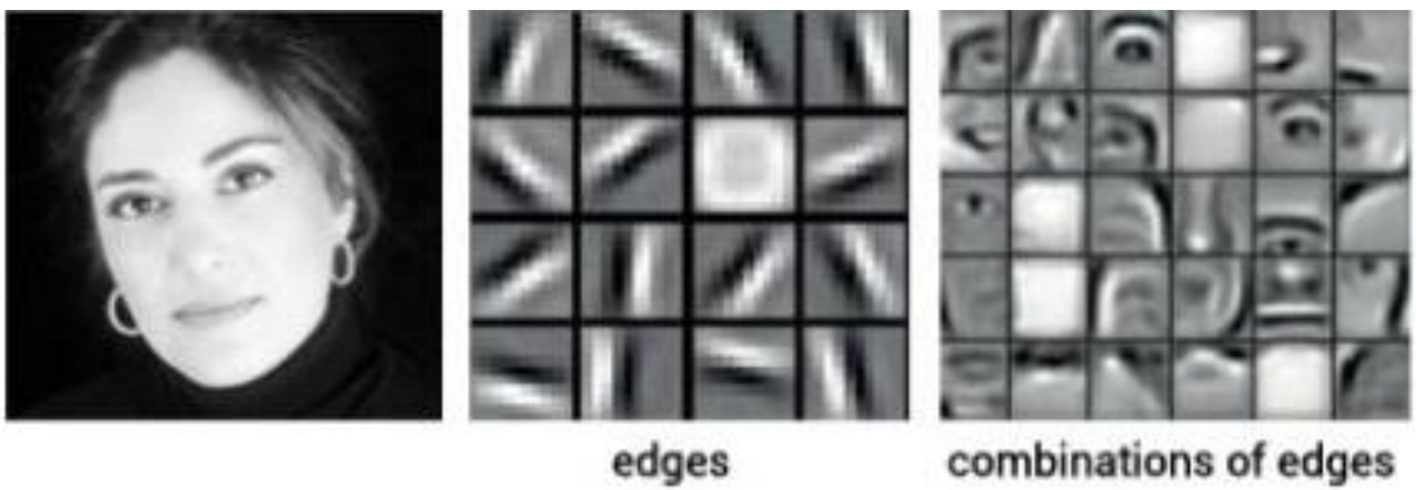
With Convolution & Correlation

If we want to get something from an image

- What would be a good choice?

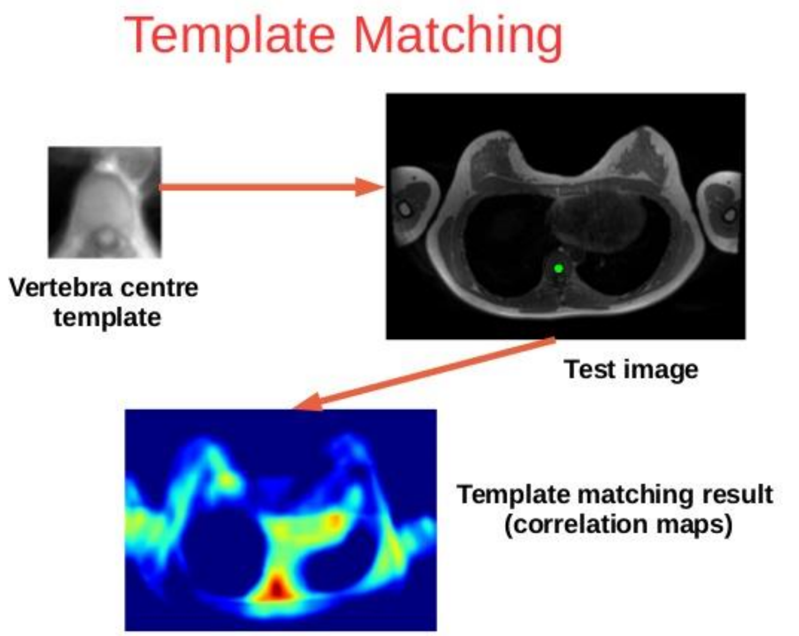
Getting what we want from an image

* Using mask or template



Having a mask, how would we get what we want?

* By finding relationship between
  * An image and our mask
    * Red -> high relationship
    * Blue -> very low relationship



How to get mask?

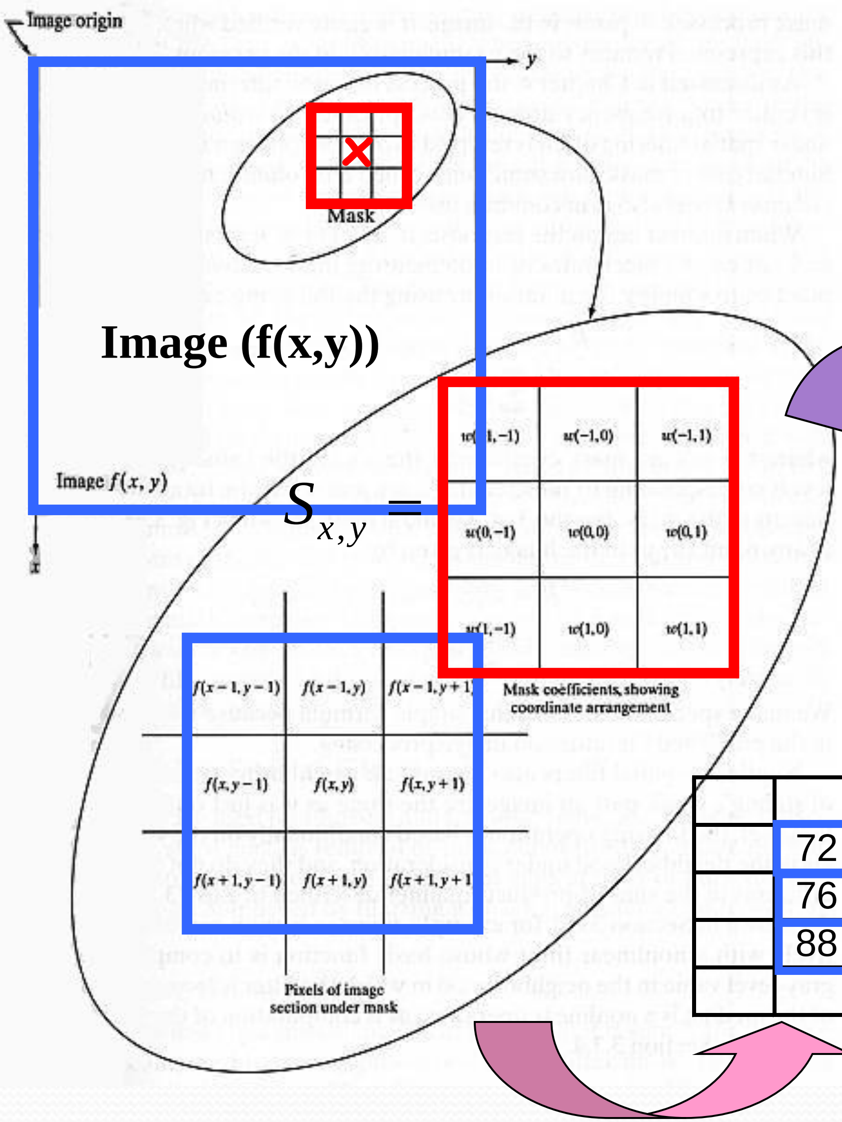
- Fix masks
  - Human Learning by experiments
  - Mathematical model
- Adaptive masks
  - Eigen based
  - Machine Learning from training
    - Convolutional Neural Network (CNN)
      - Try to learn optimum mask for training data

Relationship finding with

Template / mask

Sliding Window

Sliding Window Technique



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

Template / mask coefficients

	72	53	60	
	76	56	65	
	88	78	82	

Image pixels covered by window

Sliding Window technique for
Spatial Filtering

Convolution

$$g(x,y)=\sum_{s=-a}^{a}\sum_{t=-b}^{b}w(s,t)f(x-s,y-t)$$

Correlation

$$g(x,y)=\sum_{s=-a}^{a}\sum_{t=-b}^{b}w(s,t)f(x+s,y+t)$$

Linear Filter

Dot Product: Weighted sum

Arithmetic or Statistics

Non-Linear Filter

Nonlinear statistics

Convolution vs Correlation
(Linear filtering)

A	B	C
D	E	F
G	H	I

From I/P image

Corr

a	b	c
d	e	f
g	h	i

Template (window)

sum (

A*a	B*b	C*c
D*d	E*e	F*f
G*g	H*h	I*i

)

Correlation
(similarity measure)

a	b	c
d	e	f
g	h	i

Template (window)

Conv

i	h	g
f	e	d
c	b	a

180 degree rotated
Template (window)

sum (

A*i	B*h	C*g
D*f	E*e	F*d
G*c	H*b	I*a

)

Convolution

What would be the case

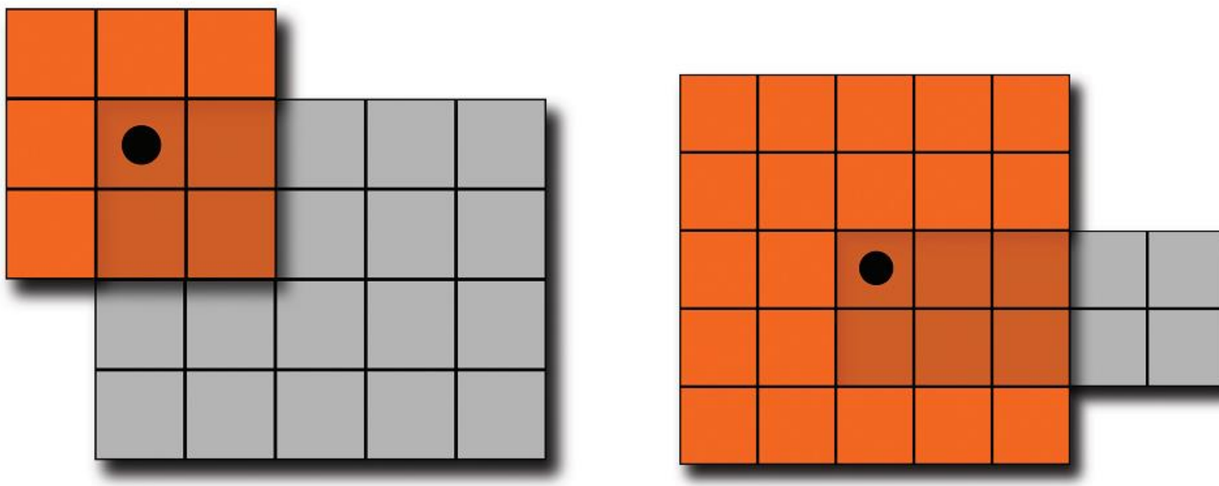
which makes Correlation and Convolution given the same result?

How to deal with Boundary Problems

สำหรับการคำนวณ

Correlation and Convolution

Convolution: The Edge Problem



(a) Kernel at $I(0,0)$. (b) Kernel larger than the source.

Figure 6.4. Illustration of the edge handling problem.

การจัดการปัญหาการคำนวณบริเวณขอบภาพ

- ปัญหาการคำนวณ
  - ต้องใช้ค่าจากตำแหน่งจุดภาพนอกกรอบภาพ ซึ่งไม่มีค่าจริง
- เทคนิคการกำหนดค่าของตำแหน่งจุดภาพนอกกรอบภาพ
  - เลือกค่า default เช่น กำหนดให้มีค่าเป็น 0
  - Repeated boundary (ขยายค่าตำแหน่งจุดภาพที่ขอบนอกของภาพออกไป)
  - Reflected Index (ขยายแบบสะท้อนตำแหน่งจุดภาพ)
    - e.g. column[-1] = column[1], column[-2] = column[2]
  - Wrap the image values (Circular Index)
    - e.g. column[-1] = column[width-1], column[-2] = column[width-2]

Image Padding Examples

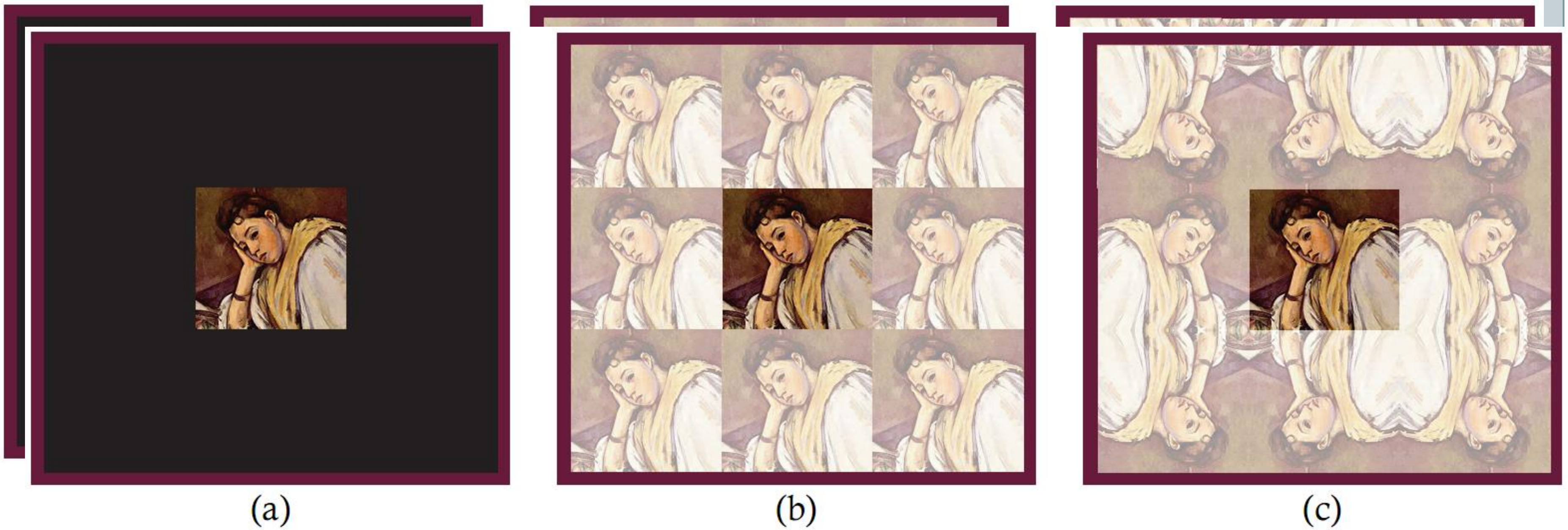


Figure 6.5. (a) Zero padding, (b) circular indexing, and (c) reflected indexing.

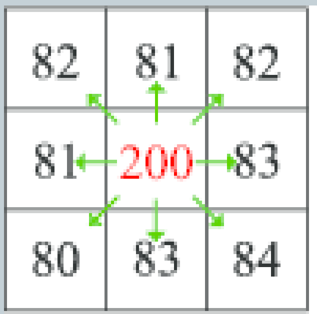
Let’s Play

WWW.KAHOOT.IT

ความแตกต่างของ Template

มีผลอย่างไรต่อภาพผลลัพธ์หลัง Filtering

Excercise



$$\frac{1}{9}\times\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

$$\frac{1}{16}\times\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

$$w=\begin{bmatrix}0 & 1 & 0\\ 1 & -4 & 1\\ 0 & 1 & 0\end{bmatrix}$$

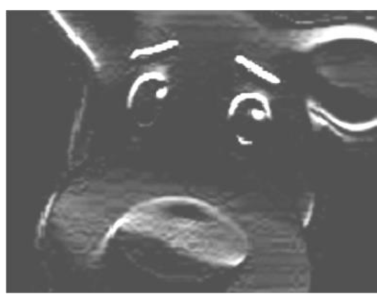
$$w_x=\begin{bmatrix}-1 & -2 & -1\\ 0 & 0 & 0\\ 1 & 2 & 1\end{bmatrix}$$

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

$$\frac{1}{16} \times \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

$$w_x = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

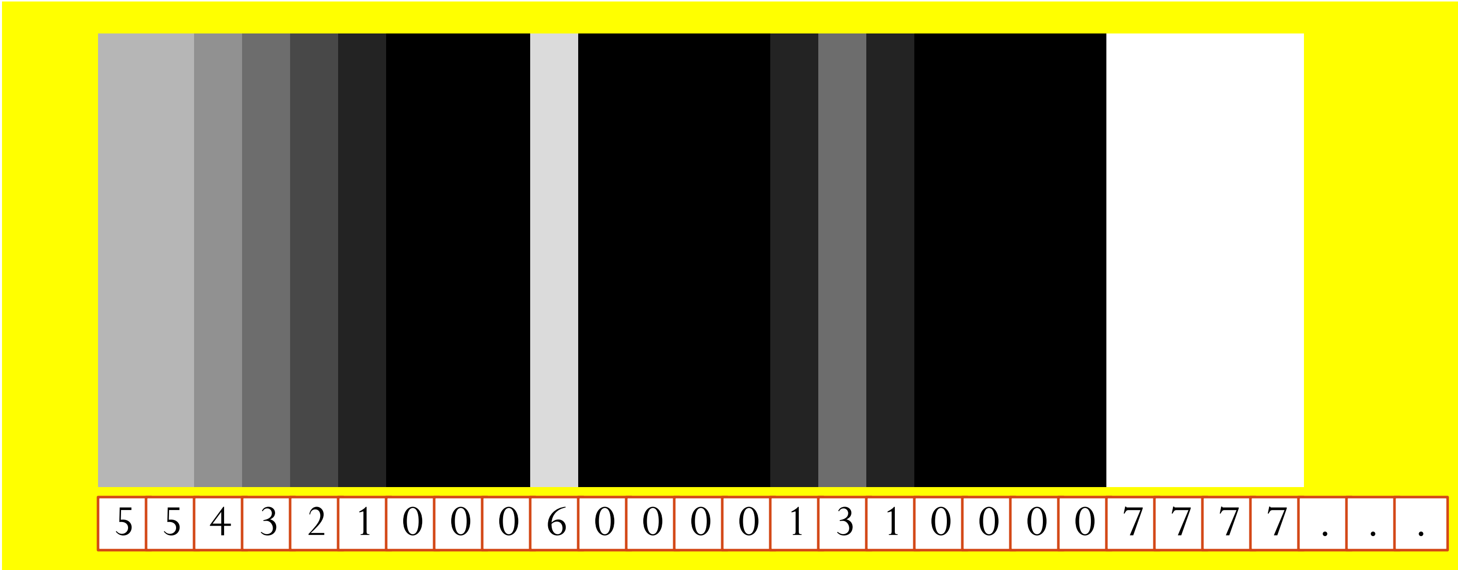
$$w = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

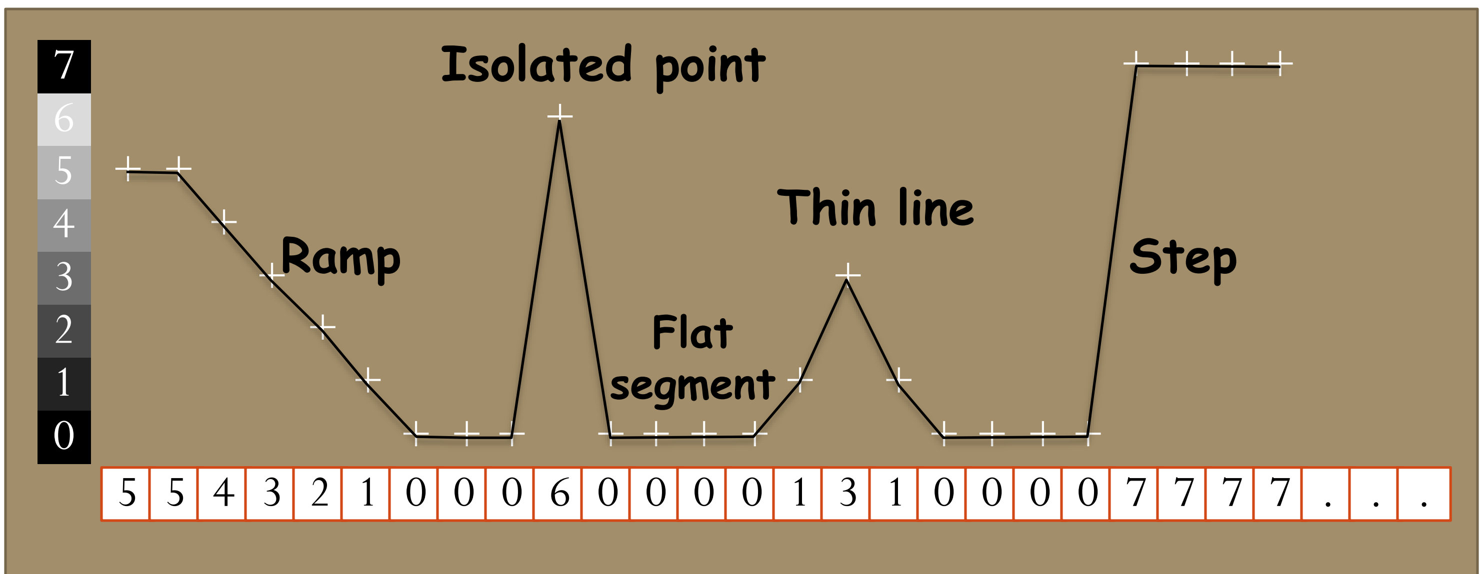


Without Noise

With Gaussian Noise

Image sharpening





Sharpening Technique
First-order derivative

- used to emphasize the boundary of the object

$$\nabla F = \begin{bmatrix} F_x \\ F_y \end{bmatrix} = \begin{bmatrix} \dfrac{\partial f}{\partial x} \\[2ex] \dfrac{\partial f}{\partial y} \end{bmatrix}$$

Gradient vector

Value depends on the change in intensity

Magnitude of change

$$\left|\nabla F\right| = mag(\nabla F) = \sqrt{F_x^2 + F_y^2} = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

Angle of change

$$\theta = \tan^{-1}\left(\frac{F_y}{F_x}\right)$$

First-order derivative (Convolution)

$$g(x,y)=\nabla F(x,y)=\sum_{s=-a}^{a}\sum_{t=-b}^{b} w(s,t)\,f(x-s,y-t)$$

$$g(x,y)=w*f(x,y) \;\rightarrow\; \begin{cases} F_x(x,y)=w_x*f(x,y) \\ F_y(x,y)=w_y*f(x,y) \end{cases}$$

Sobel filter (Prewitt kernel)

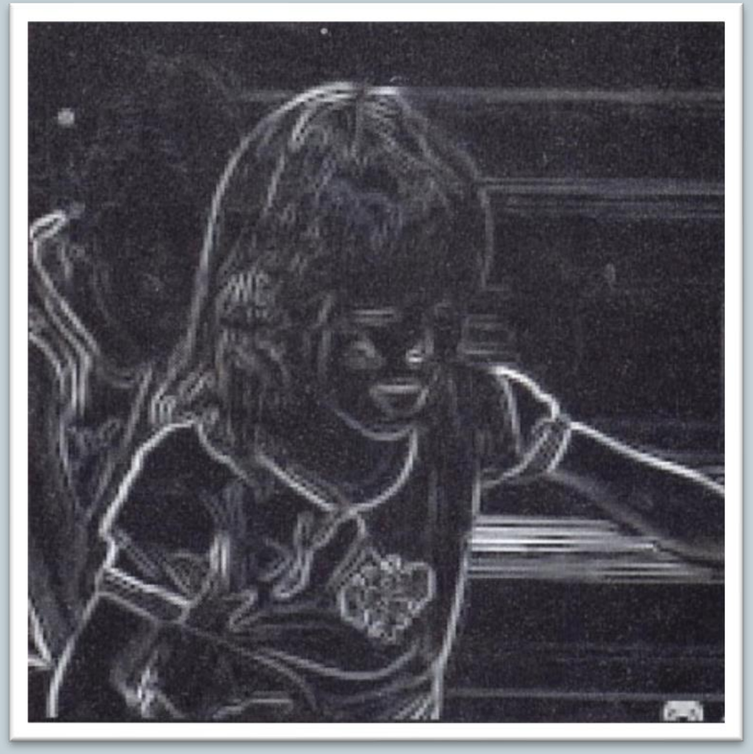
$$w_x=\begin{bmatrix}-1 & -2 & -1\\ 0 & 0 & 0\\ 1 & 2 & 1\end{bmatrix} \qquad w_y=\begin{bmatrix}-1 & 0 & 1\\ -2 & 0 & 2\\ -1 & 0 & 1\end{bmatrix}$$

Robert cross gradient operator

$$w_x=\begin{bmatrix}-1 & 0\\ 0 & 1\end{bmatrix} \qquad w_y=\begin{bmatrix}0 & -1\\ 1 & 0\end{bmatrix}$$

Sharpening (1st order derivative, Sobel filter)

Original image

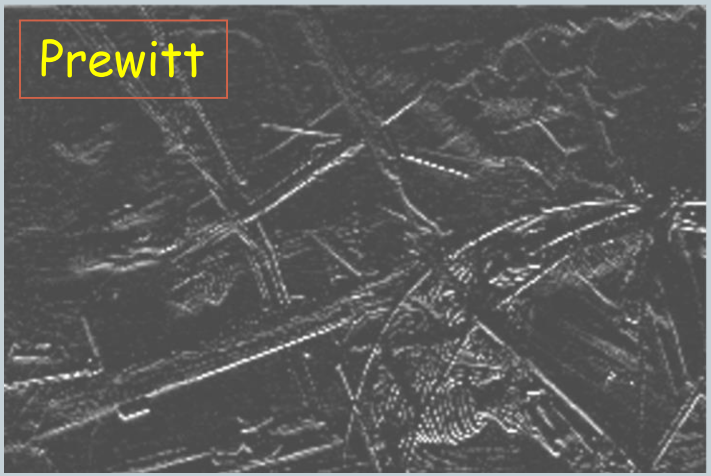


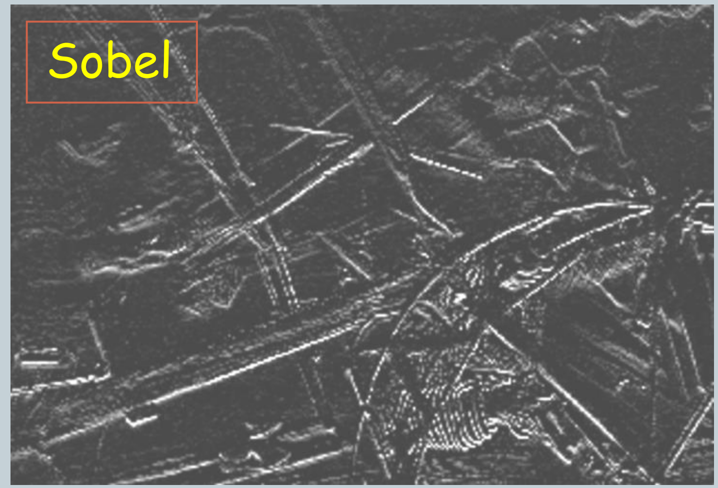
Magnitude gradient image

Edge Detection Results

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

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



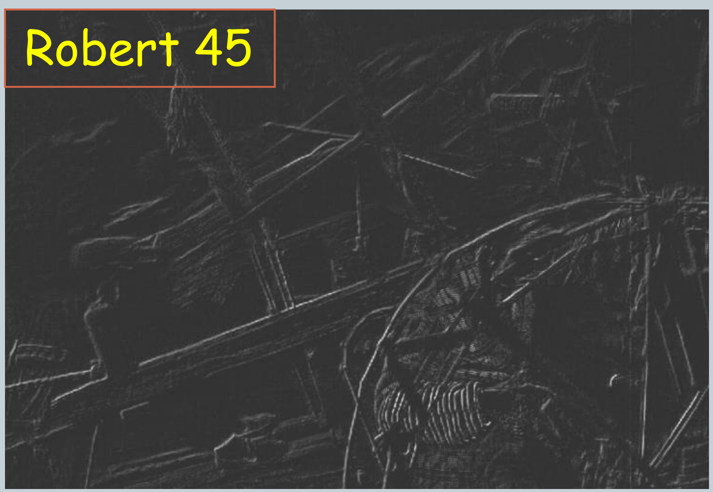


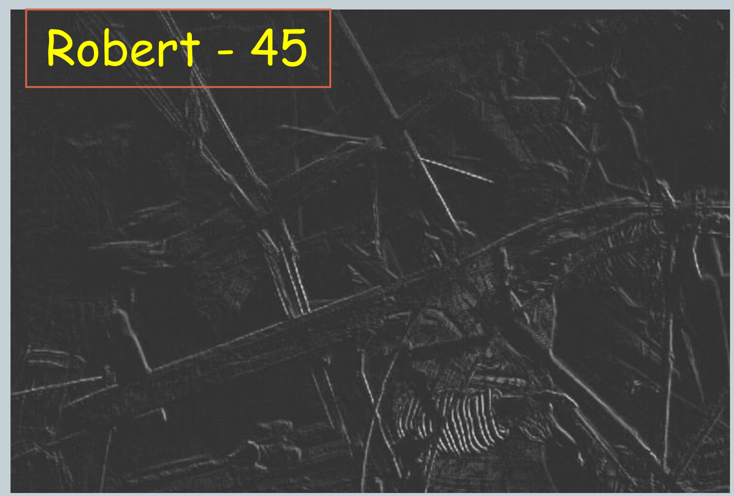
$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

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

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

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





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

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

Second-order derivative

$$\nabla^2 F = \begin{bmatrix} F_x^2 \\ F_y^2 \end{bmatrix} = \begin{bmatrix} \dfrac{\partial^2 f}{\partial x^2} \\ \\ \dfrac{\partial^2 f}{\partial y^2} \end{bmatrix}$$

$$g(x, y) = \nabla^2 F(x, y) = w * f(x, y)$$

Laplacian Operator

$$w = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad \textbf{or} \quad w = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

$$w = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad \textbf{or} \quad w = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Edge detection

from Difference images between original and blur version

- Type equation here.Edge from intensity difference
  - Edge_diff(x,y) = f_original_image(x,y) – f_blur_image(x,y)

$$f_{blur_{image}}(x,y) = f_{original\_image}(x,y) * h_{gaussian}(x,y)$$

$$h_{gaussian}(x,y) = e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

0.0030	0.0133	0.0219	0.0133	0.0030
0.0133	0.0596	0.0983	0.0596	0.0133
0.0219	0.0983	0.1621	0.0983	0.0219
0.0133	0.0596	0.0983	0.0596	0.0133
0.0030	0.0133	0.0219	0.0133	0.0030

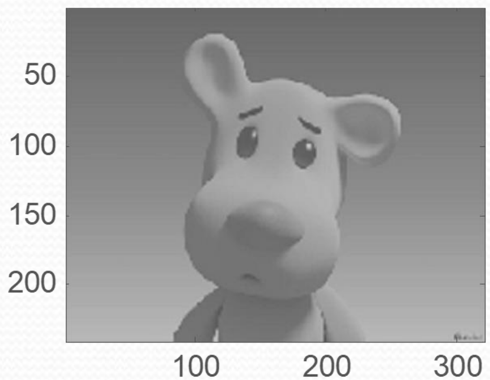
$$h_{gaussian}(\sigma = 1)$$

0.0318	0.0375	0.0397	0.0375	0.0318
0.0375	0.0443	0.0469	0.0443	0.0375
0.0397	0.0469	0.0495	0.0469	0.0397
0.0375	0.0443	0.0469	0.0443	0.0375
0.0318	0.0375	0.0397	0.0375	0.0318

$$h_{gaussian}(\sigma = 3)$$

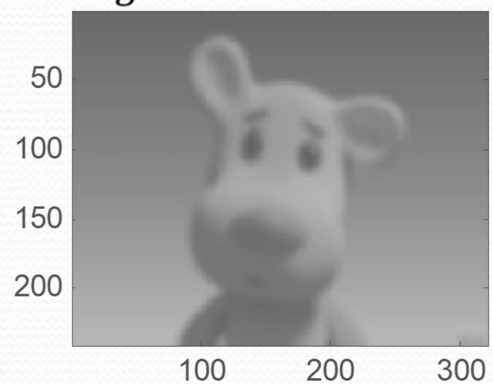
Edge detection

from Difference images between original and blur version

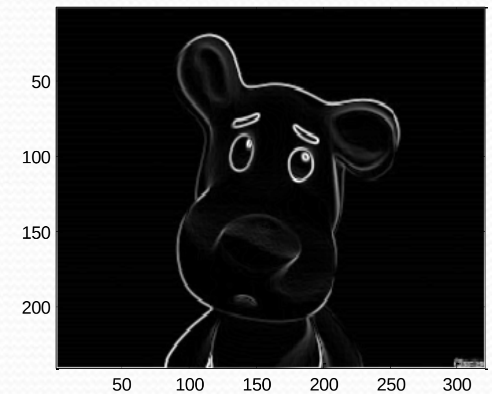


f_orignial(x,y)

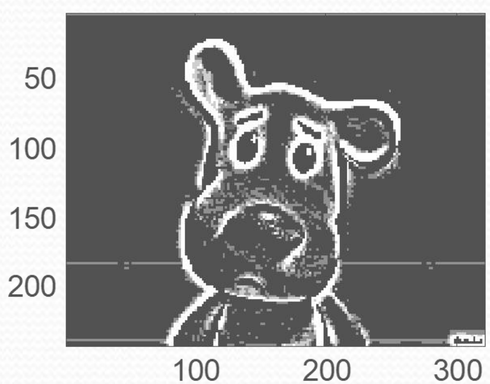
$h_{gaussian}(\sigma = 3)$



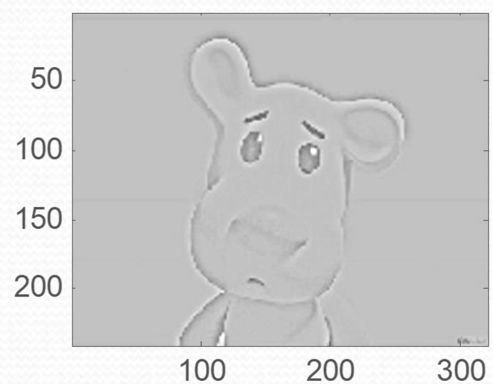
f_blur(x,y)



f_magnitude_sobel(x,y)



f_orignial(x,y) – f_blur(x,y)



Log(f_orignial(x,y)) – Log(f_blur(x,y))

What template could be used

for Noise filtering?

Types of Spatial Filtering

- Mean Filtering
  - Arithmetic Mean
  - Geometric Mean
  - Harmonic Mean
  - Contraharmonic Mean
- Order-Statistics Filtering
- Adaptive filtering

Mean Filtering

Arithmetic Mean

$$g(x,y)=\frac{1}{mn}\sum_{s=-a}^{a}\sum_{t=-b}^{b}w(s,t)\,f(x\pm s,y\pm t)$$

Geometric Mean

$$g(x,y)=\prod_{s,t}w(s,t)\,f(x\pm s,y\pm t)$$

Harmonic Mean

$$g(x,y)=\frac{mn}{\displaystyle\sum_{s=-a}^{a}\sum_{t=-b}^{b}\frac{1}{w(s,t)\,f(x\pm s,y\pm t)}}$$

Contra-Harmonic Mean

$$g(x,y)=\frac{\displaystyle\sum_{s=-a}^{a}\sum_{t=-b}^{b}f(x\pm s,y\pm t)^{q+1}}{\displaystyle\sum_{s=-a}^{a}\sum_{t=-b}^{b}f(x\pm s,y\pm t)^{q}}$$

Examples of Average Filter Template (w(*s*,*t*))

Arithmetic Mean

$$g(x,y)=\frac{1}{mn}\sum_{s=-a}^{a}\sum_{t=-b}^{b}w(s,t)f(x\pm s,y\pm t)$$

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

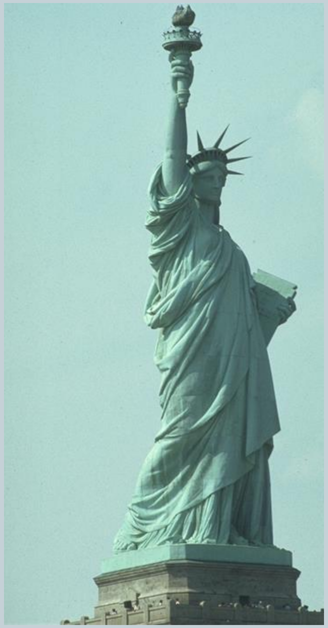
1	1	1
1	1	1
1	1	1

The most well-known average filter template

$\frac{1}{16}\times$

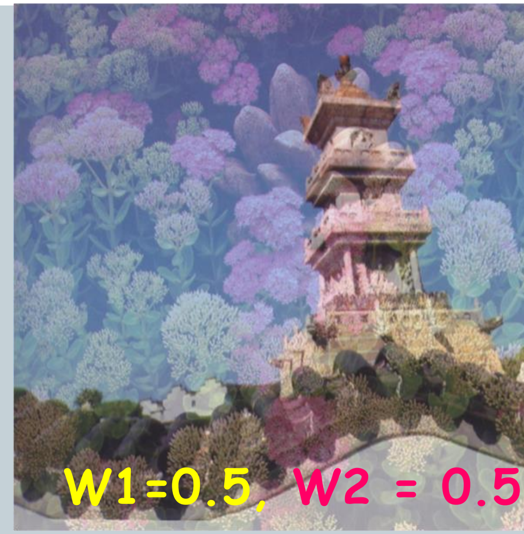
1	2	1
2	4	2
1	2	1

Image Addition



$$g(x, y) = w_1 f_1(x, y) + w_2 f_2(x, y)$$

Image Addition with
different weights



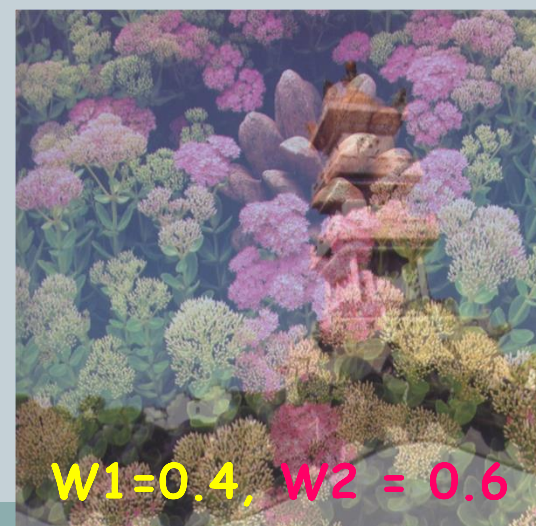


Image Subtraction

$$g(x, y) = f_1(x, y) - f_2(x, y)$$

Image Averaging

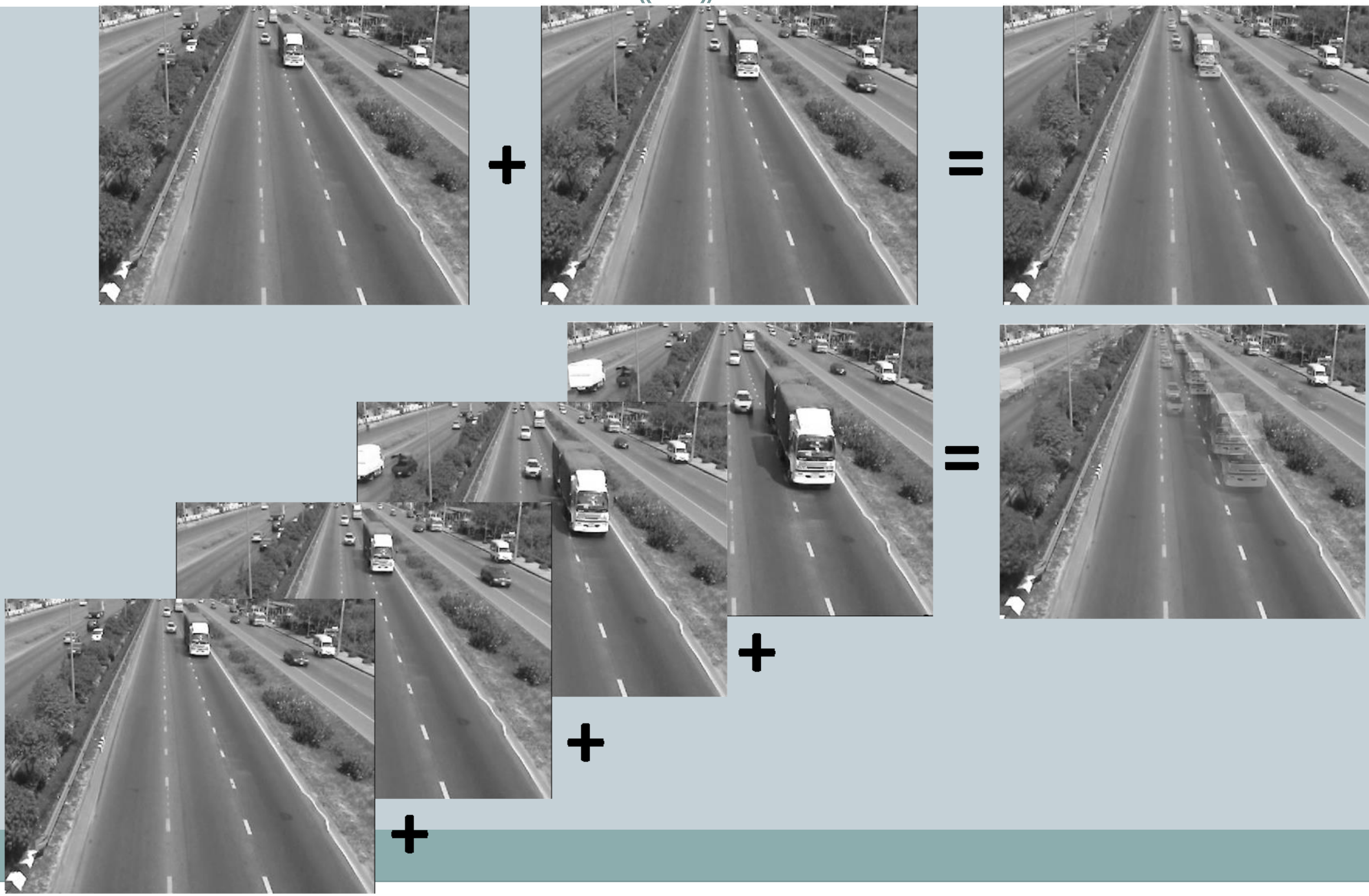


Image Averaging