

Image Filters

modify image pixels based on some function of a local neighbourhood of each pixel

Box filter

$$\begin{bmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \end{bmatrix}$$

Gaussian filter

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

edge filter

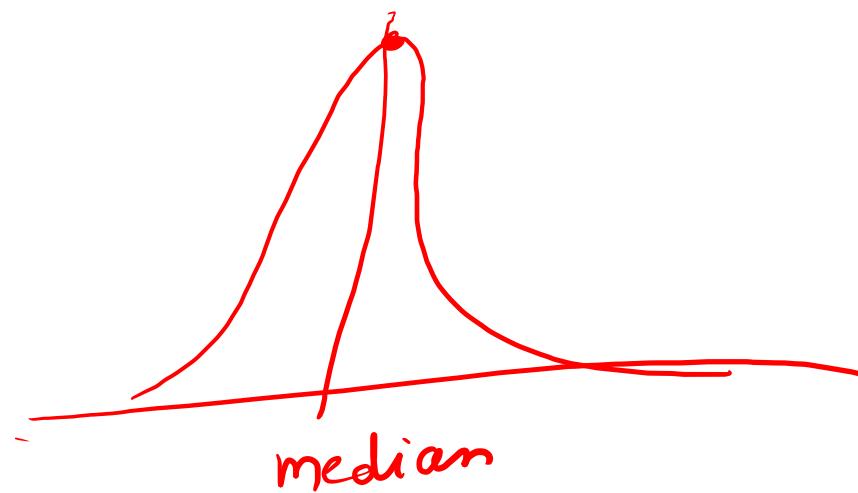


Image Filters

modify image pixels based on some function of a local neighbourhood of each pixel

$$36/g = 4$$

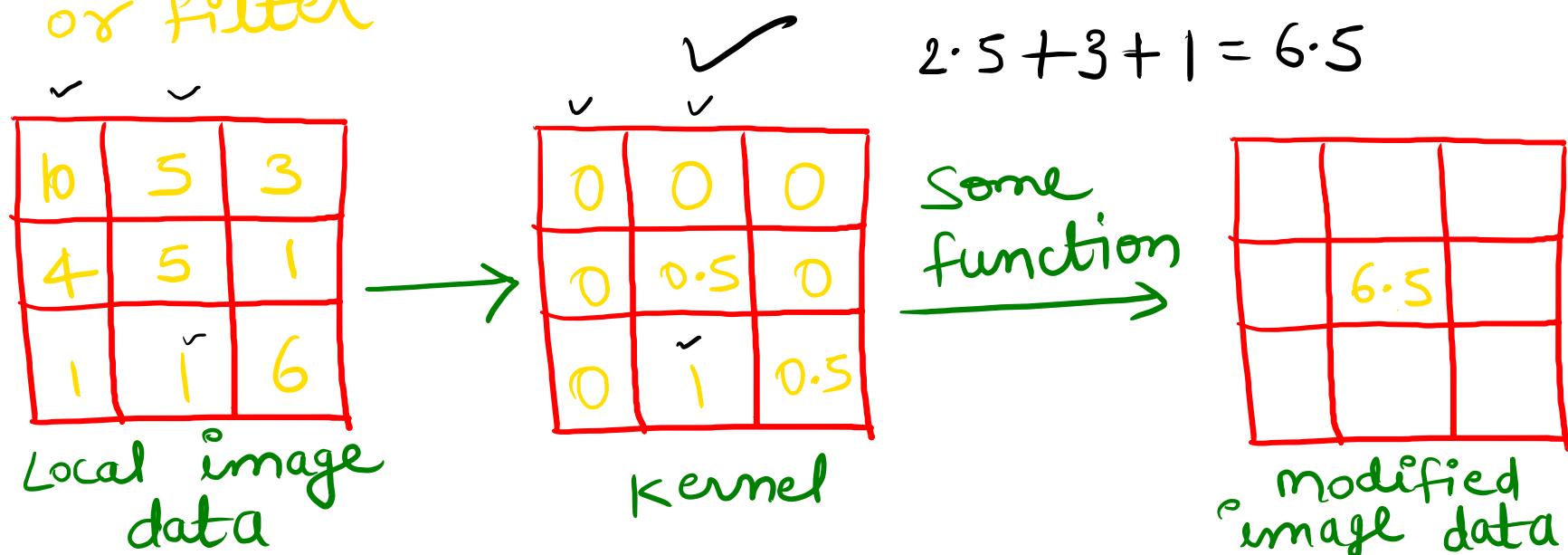
10	5	3
4	5	1
1	1	6

Some
function →

	4	

Linear Filters

- Replace each pixel by linear combinational (a weighted sum) of neighbours
- Linear combinations called kernel, mask or filter



Linear Filter: cross-correlation

Given a kernel of size $(2K+1) \times (2K+1)$

* Correlation defined as

$$C(i,j) = \frac{1}{(2K+1)^2} \sum_{u=-K}^K \sum_{v=-K}^K I(i+u, j+v)$$

uniform weight to each pixel

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

Loop over pixels in considered neighbourhood around $I(i,j)$

* Cross correlation is defined as

$$C_I(i, j) = \sum_{u=-K}^K \sum_{v=-K}^K H(u, v) I(i+u, j+v)$$

↓
Non uniform
weights

* Cross-Correlation is denoted as

$$C_I = H \otimes I$$

- * Can be viewed as dot product between local neighbourhood and kernel for each pixel
- * Entries of kernel or mask $H(u, v)$ called filter coefficients

Local operations (moving average linear filter)

$I(i, j)$

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	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	0	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

$h(i, j)$

0	10	20	30	30
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

$$\otimes H(u, v) \quad \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}_{3 \times 3}$$

Local operations (moving average linear filter)

$I(i, j)$

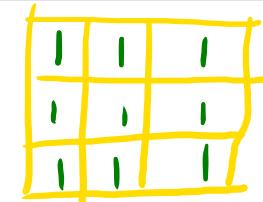
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	0	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

$H(i, j)$

0	10	20	30	30
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

$$\otimes H(u, v)$$

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

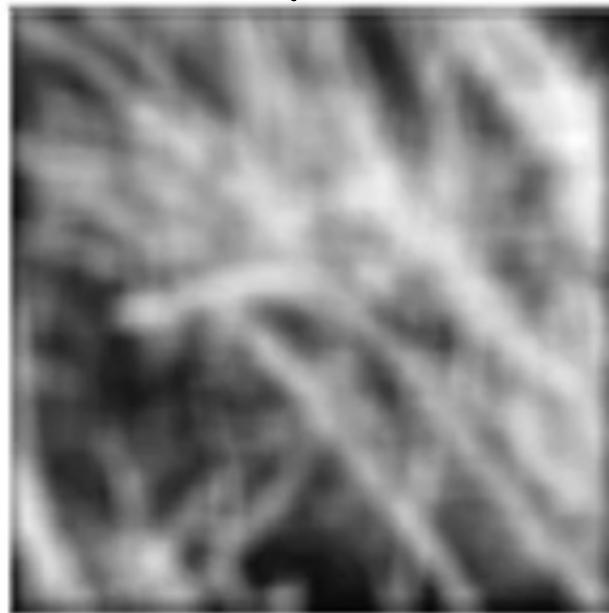


-Box
filter

Effect of Moving average filter

Input

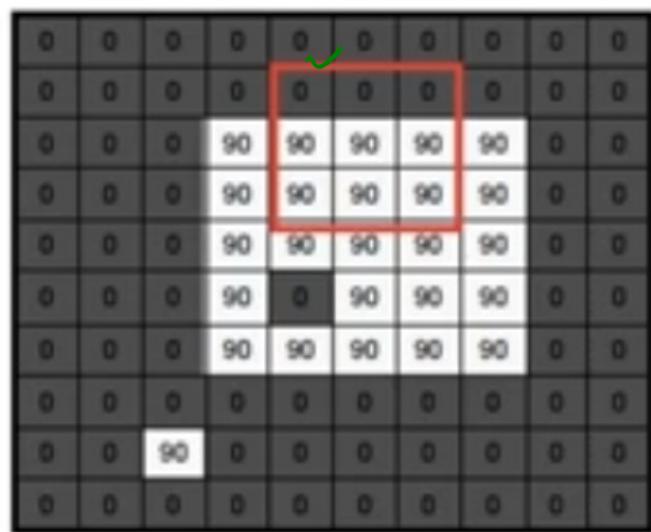
Output



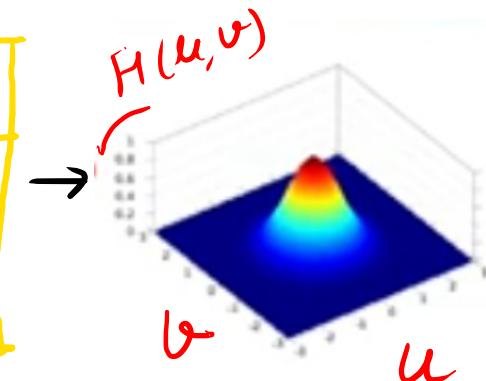
Gaussian Average filter

what if want nearest neighbouring pixels to have the most influence on the output

$I(i, j)$



$$\otimes H(u, v) \rightarrow \frac{1}{16} \begin{matrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{matrix}$$



This kernel is an approximation of a
2D Gaussian function

$$h(u, v) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{u^2+v^2}{2\sigma^2}}$$

$$\mu = 0$$

$$e^{-\frac{u^2+v^2}{2\sigma^2}}$$

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$$

$$= \frac{1}{e^{\frac{u^2+v^2}{2\sigma^2}}}$$

$$e \text{ power } x = 1 + x + \text{sqr } x / \text{fact (2)} + \text{cube (x)} / \text{fact (3)} + \dots$$

$$e \text{ power } 1 = 1 + 1/1! + 1/2! + 1/3! + 1/4! + 1/5! + \dots e \text{ power } 1 = 2.718$$

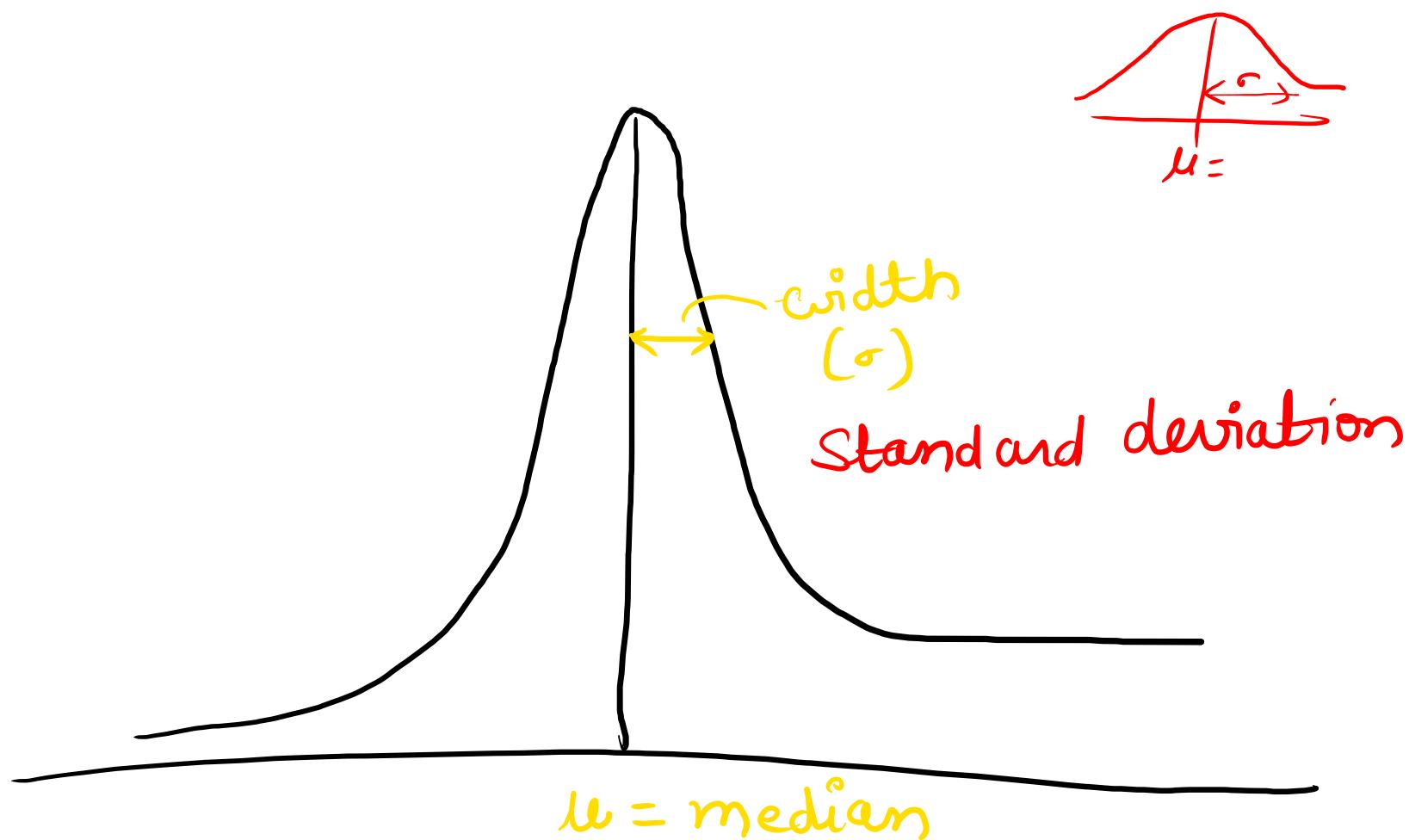
$$e^{-1} = \frac{e^1 - 1}{e^1} = \frac{2.718281828459045 - 1}{2.718281828459045} = 0.36787944117$$

γ_e

✓

1D Gaussian function

$$Cr(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$$
$$= \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2} \left(\frac{x-\mu}{\sigma}\right)^2}$$



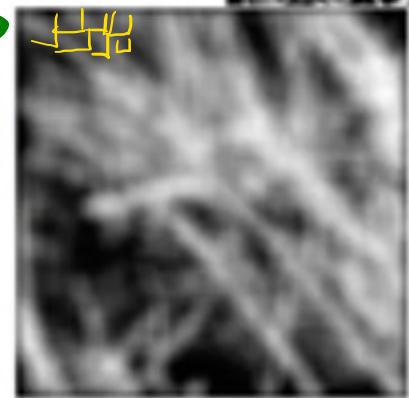
A Comparison in Averaging filter

Input



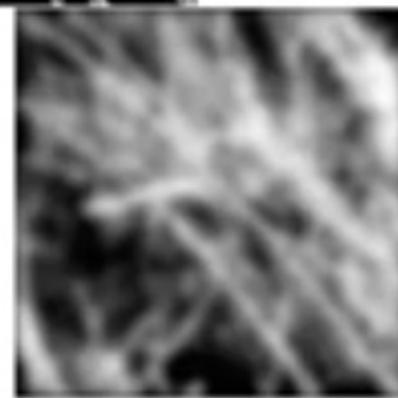
output 1

Box
filter

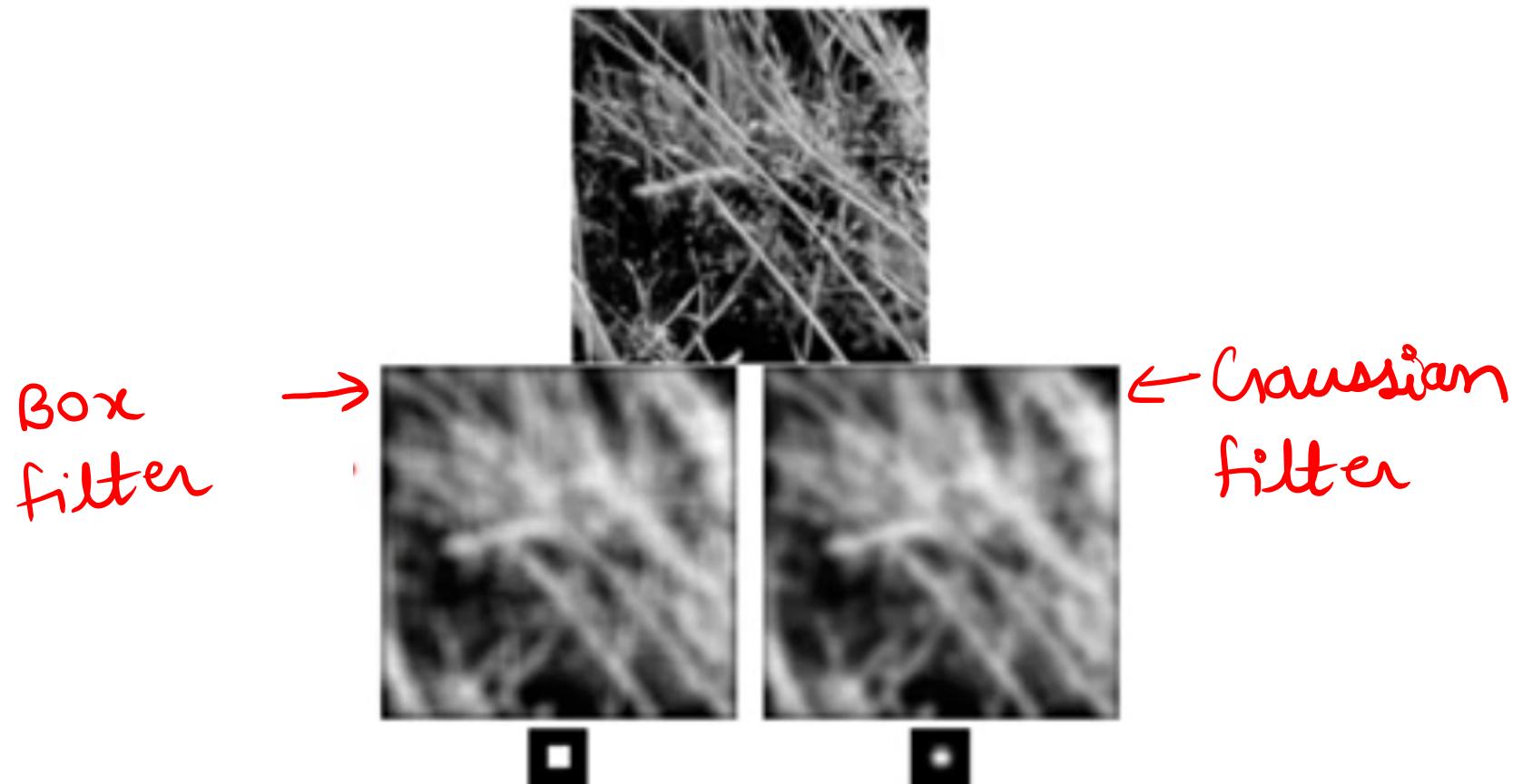


output 2

Gaussian
filter



A Comparison in Averaging filter

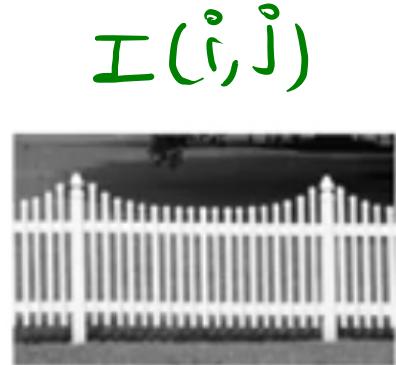


Box filter vs Gaussian Filter



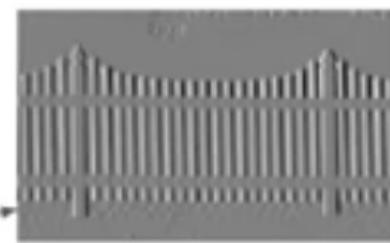
Edge filter

What should H look like to find edges in
a given image?



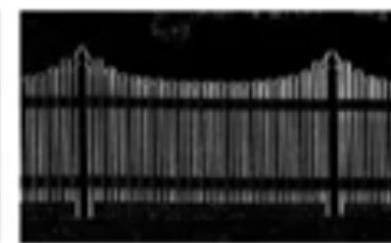
$$H(u, v)$$

$H(u, v)$ for
Vertical Edges?

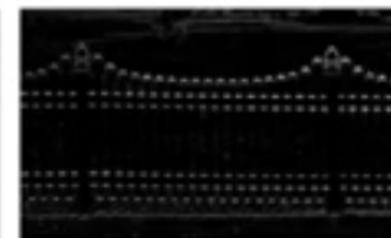
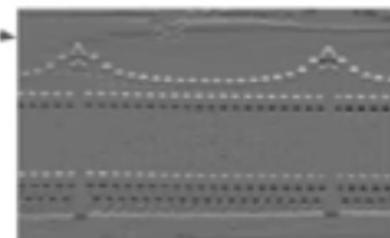


$$G(i, j)$$

$H(u, v)$ for
Horizontal Edges?



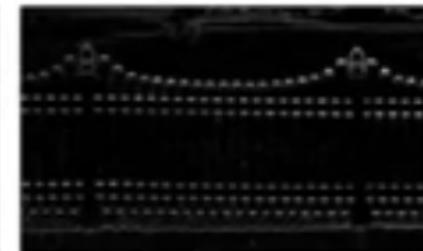
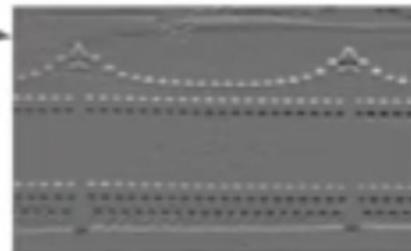
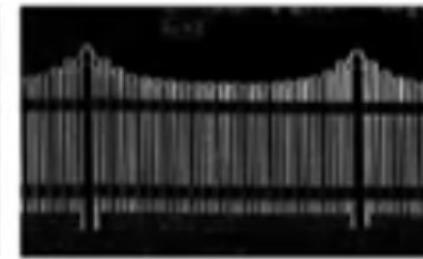
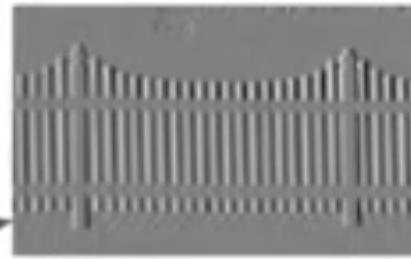
$$|G(i, j)|$$



vertical
 $H(u, v)$

$$1/9 \begin{matrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{matrix}$$

$$1/9 \begin{matrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{matrix}$$



$H(u, v)$
horizontal

Edge filter

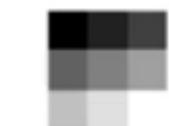
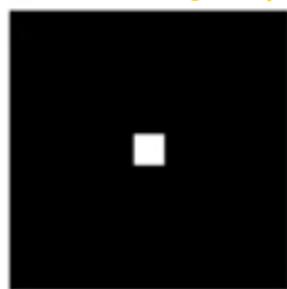
- A vertical filter would have $-1, -1, -1$ on one side and $1, 1, 1$ on the other side.
- So effectively filter looks for the places in the image where the significant difference between the left of the pixel and right of the pixel vertically.
- You try to exaggerate those pixels in the output image.

- A horizontal edge filter will do the same thing, but along the horizontal directions.
- The normalizing factor is $\frac{1}{9}$, but you typically do not need that in an edge filter because we are not interested in absolute edge value, but we are interested in high intensity where there is an edge.

Beyond Correlation (limitation of cross-correlation)

What is the result of the filtering impulse signal (image) I with the arbitrary kernel H ?

$$I(i,j) \quad \otimes H(u,v) \quad C(i,j)$$



a	b	c
d	e	f
g	h	i

?

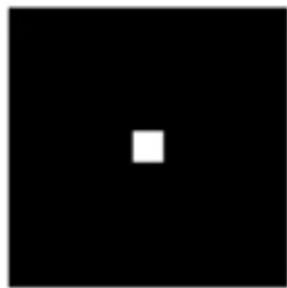
✓



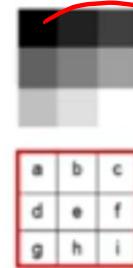
neither box filter nor
Gaussian filter

Detailed explanation

$I(i, j)$

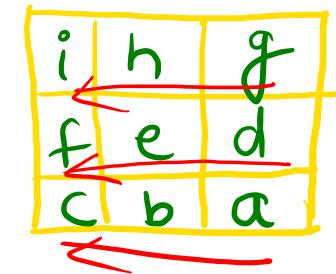
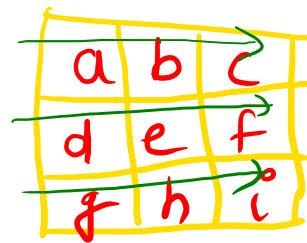
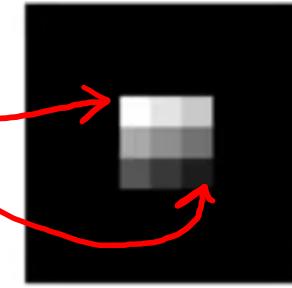


$\otimes H(u, v)$



a	b	c
d	e	f
g	h	i

$C(i, j)$



flip horizontally

a	b	c
d	e	f
g	h	i

-

g	h	i
d	e	f
a	b	c

1

1

i	h	g
f	e	d
c	b	a

vertical flip

double flipping

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

a b c
d e f
g h i



1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25



7 8 9
12 13 14
15 16 17

①

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

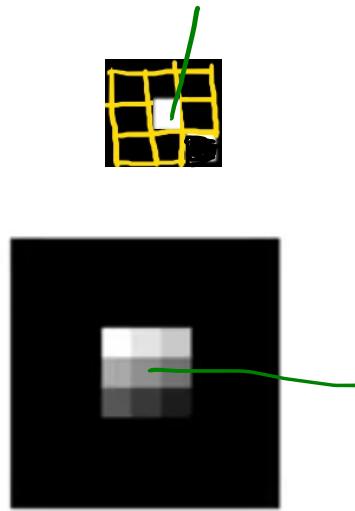
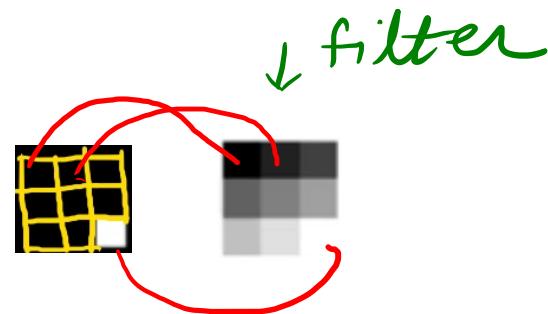
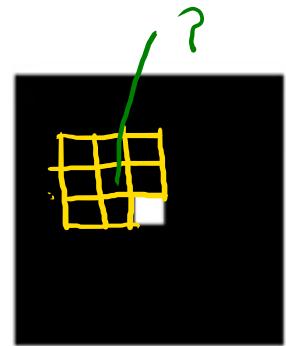
1 h g
f c d
b a

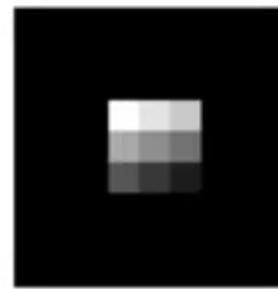
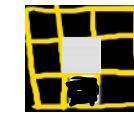
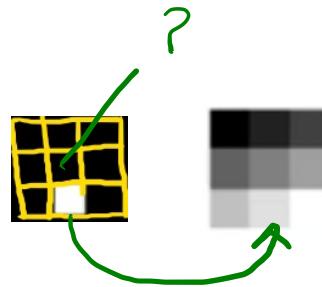
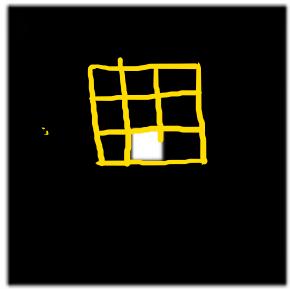
1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

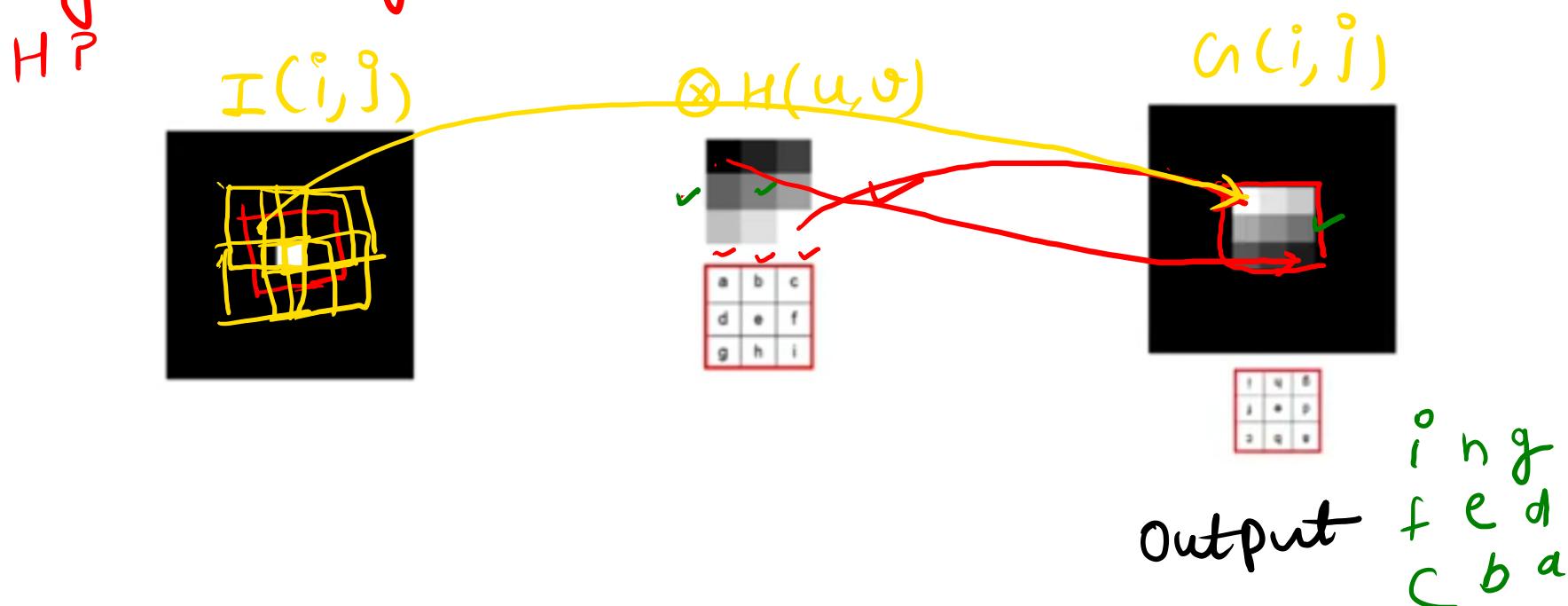
1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25





Beyond Correlation

What is the result of the filtering impulse signal (image) I with the arbitrary kernel H ?



- The output image is completely flipped
- The bottom right swapped with top left.
- your output is the double flipped version of your input
- white value is the only value that will get multiplied by 1 and other kernel value get multiplied with by corresponding value 0

- which means the output at this particular location is going to be white and we will keep doing this over and over again at each pixel

Convolution

Given a Kernel of size $(2k+1) \times (2k+1)$

* Convolution defined as

$$C(i, j) = \frac{1}{(2k+1)^2} \sum_{u=-k}^k \sum_{v=-k}^k H(u, v) I(i-u, j-v)$$

* Equivalent to flip the filter in both directions (bottom to top, right to left) and apply correlation

* denoted by $G = H * I$



Convolution

Cross-correlation vs convolution

$$G = H \otimes I$$

(cross-correlation)

$$G(i, j) = \sum_{u=-K}^K \sum_{v=-K}^K H(u, v) I(i+u, j+v)$$



$$G = H * I$$

(convolution)

$$G(i, j) = \sum_{u=-K}^K \sum_{v=-K}^K H(u, v) I(i-u, j-v)$$



a	b	c
d	e	f
g	h	i

Cross-Correlation Convolution

-1,-1	0,-1	1,-1
-1,0	0,0	1,0
-1,1	0,1	1,1



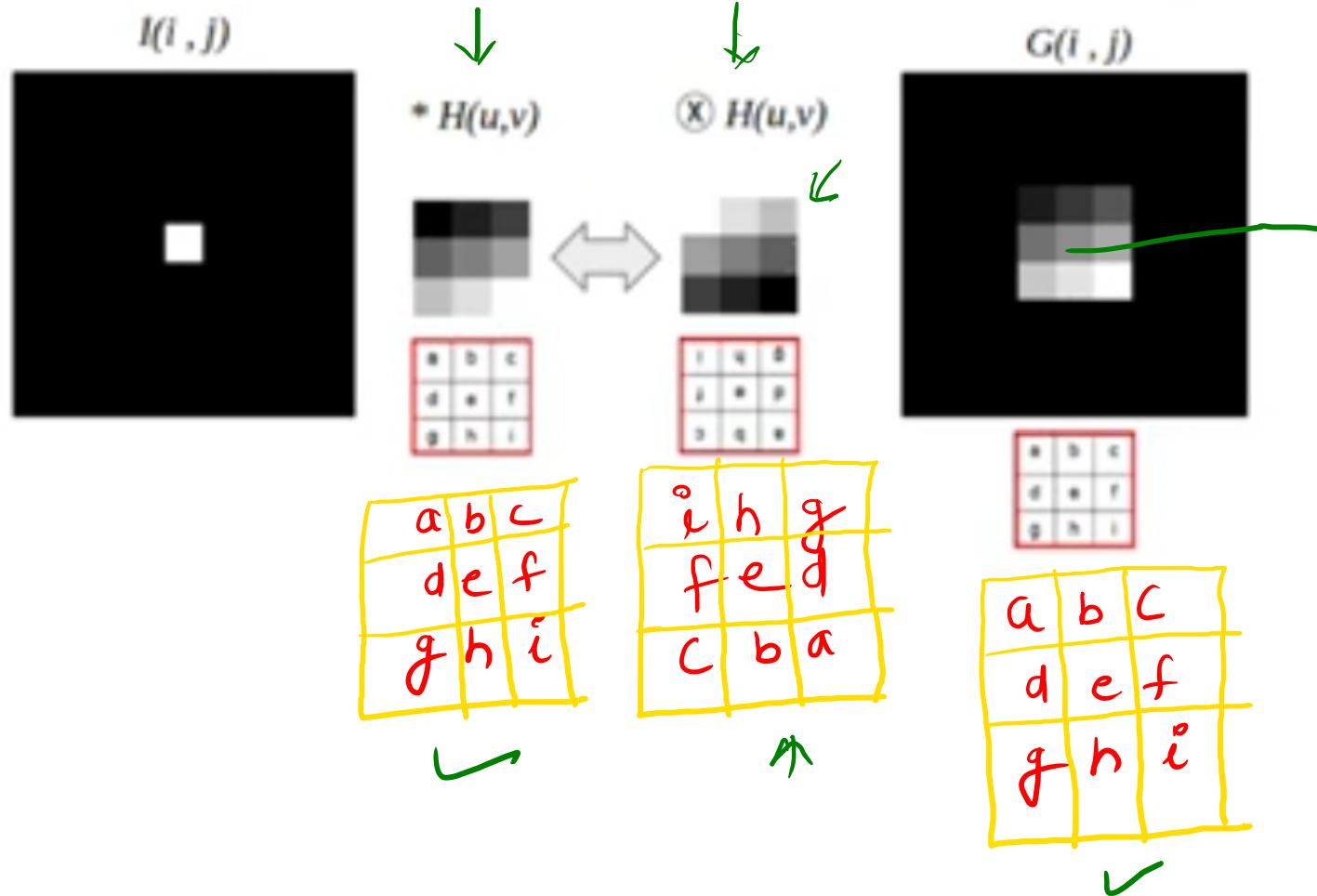
1,1	0,1	-1,1
1,0	0,0	-1,0
1,-1	0,-1	-1,-1



$$\begin{array}{r}
 1 \quad 2 \quad 3 \\
 \xrightarrow{\hspace{2cm}}
 \\[0.5ex]
 4 \quad 5 \quad 6 \\
 \xrightarrow{\hspace{2cm}}
 \\[0.5ex]
 7 \quad 8 \quad 9
 \end{array}$$

9	8	7
6	5	4
3	2	1

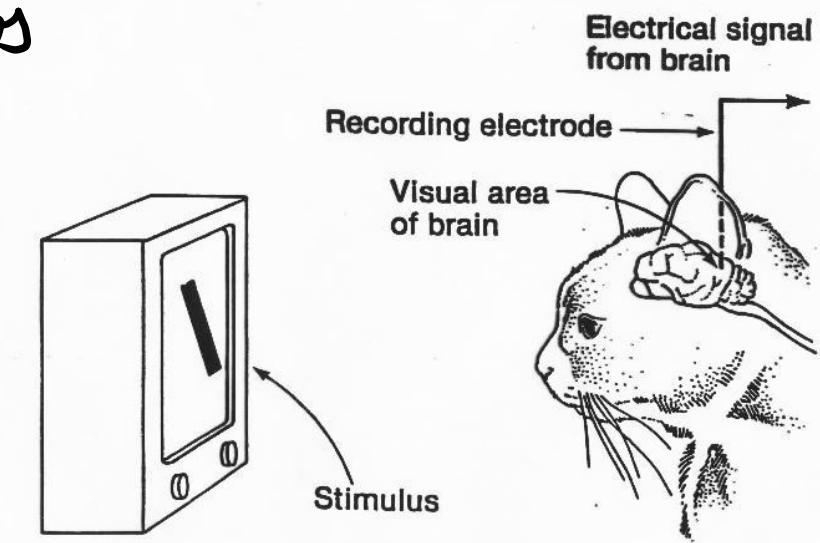
Introducing Convolution



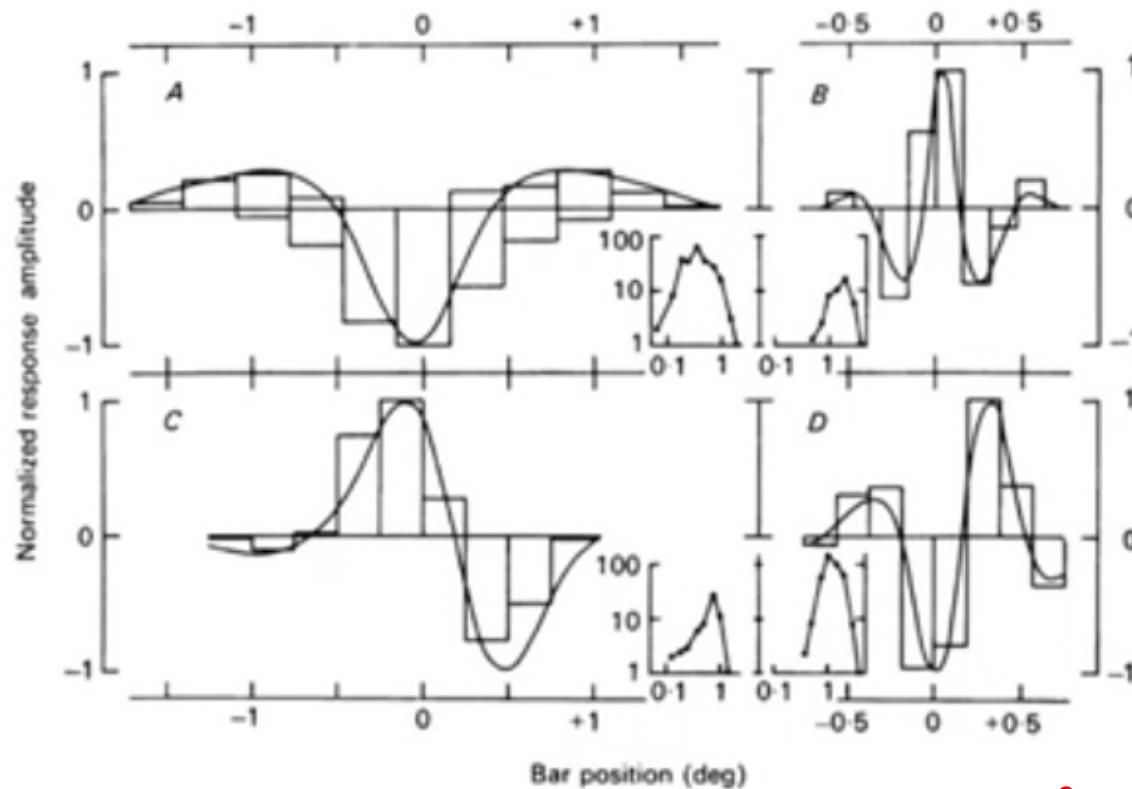
Recall early history

This experiment shows that simple and complex neurons exist.

The visual system starts with simple structures like oriented edges.



Linear Summations in the Visual Cortex



[
+ + +
+ + +]

Simple cells performs linear spatial summation over their receptive fields

Linear Shift-Invariant Operators

* Both Correlation and Convolution are Linear Shift-Invariant operators, which obey

(i) Linearity (or Superposition principle)

$$I \circ (h_0 + h_1) = I \circ h_0 + I \circ h_1$$

✓ ✓

(2) Shift-invariance: shifting (or translating) a signal commutes with applying the operator \hat{I}

$$\begin{aligned} g(i, j) &= h(i+k, j+l) \\ (f \circ g)(i, j) &= (f \circ h)(i+k, j+l) \end{aligned}$$

* Equivalent to saying that effect of operator is the same everywhere. Why do we need this in computer vision?

Properties of Convolution

(1) Commutative: $a \times b = b \times a$

- Conceptually no difference between filter
and signal

(2) Associative: $(a \times b) \times c = a \times (b \times c)$

- We often apply filters one after the
other $((a \times b_1) \times b_2) \times b_3$

- This is equivalent to applying one
filter: $a \times (b_1 \times b_2 \times b_3)$

(3) Distributive over addition:

$$a \times (b+c) = (a \times b) + (a \times c)$$

-we can combine the responses of a signal over two or more filters by combining the filters

constant

$$(4) Scalars factor out: kaxb = a \times kb = k(axb)$$

$$(5) Identity: Unit impulse, e = \{ \dots, 0, 0, 1, 0, 0, \dots \}$$

$$a * e = a$$

(6) One more important property of Convolution
is separability

Separability

- Convolution operator requires K^2 operations per pixel, where K is the width (and height) of a convolution kernel.
- This is costly operation. How can we reduce cost?

- For certain filters, can be speed up by performing a 1D horizontal convolution followed by a 1D vertical convolution, requiring $2K$ operations



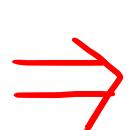
Convolution kernel is separable

$$k = v h^T$$

where v, h are 1D kernel, and k is 2D kernel

For Example 1:

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



$$\vartheta = h = \frac{1}{4}$$

$$\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$$

2D kernel

$$h^T = \frac{1}{4} [1 \ 2 \ 1]$$

$$K = \vartheta h^T$$

$$\frac{1}{4} \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}_{3 \times 1} \times \frac{1}{4} [1, 2, 1]_{1 \times 3}$$

$$= \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}_{3 \times 3}$$

Example 2

$\frac{1}{8}$

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



$$g = ?$$

$$h = ?$$

Example 2

$$\frac{1}{8}$$

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

$$\Rightarrow g = \frac{1}{4}$$

-1
-2
-1

$h = \frac{1}{2}$

1
0
1

Separable Convolution

How can we tell if a given kernel is separable?

Separable Convolution

How can we tell if a given kernel is separable?

- Visual inspection
- Analytically, look at the singular value decomposition (SVD), and if only one singular value is non-zero, then it is separable

$$K = U \Sigma V^T = \sum_i \sigma_i u_i v_i^T$$

where $\Sigma = \text{diag}(\sigma_k)$

$$\Sigma_i = \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_k \end{bmatrix}$$

$\sqrt{\sigma_i} u_i$ and $\sqrt{\sigma_i} v_i$ are the vertical
and horizontal kernels