

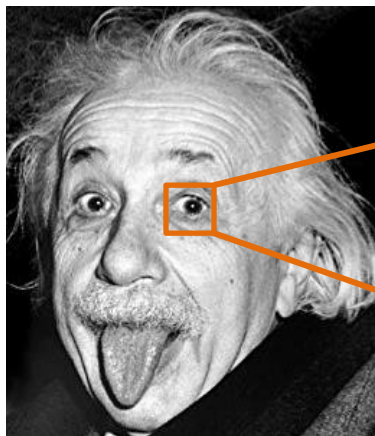


IMAGE CONVOLUTION

HYUN SOO PARK

IMAGE SPATIAL FILTERING ~ CORRELATION

$$I_f(i, j) = \sum_{k,l} I(i+k, j+l)z(k,l)$$



2	1	4	4	7
1	2	2	3	6
3	3	5	8	9
5	2	2	6	7
8	3	2	1	3

I

Image

\otimes

a	b	c
d	e	f
g	h	i

z

Filter

=

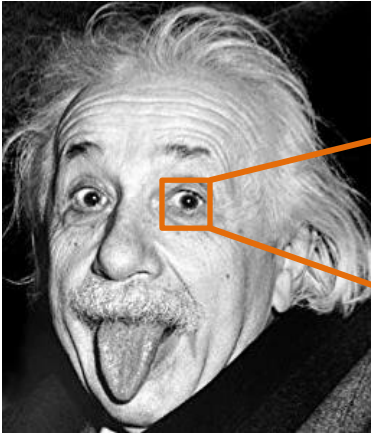
	y			

I_f

Filtered image

$$y = 2a + b + 4c + d + 2e + 2f + 3g + 3h + 5i$$

BOUNDARY



2	1	4	4	7
1	2	2	3	6
3	3	5	8	9
5	2	2	6	7
8	3	2	1	3

I
Image

$$I_f(i, j) = \sum_{k,l} I(i+k, j+l)z(k,l)$$

\otimes

a	b	c
d	e	f
g	h	i

z
Filter

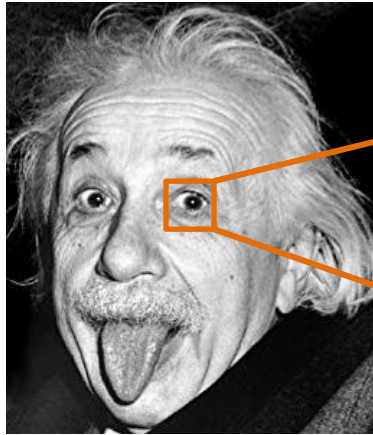
=

y				

I_f
Filtered image

$y = ?$

BOUNDARY



0	0	0	0	0	0	0
0	2	1	4	4	7	0
0	1	2	2	3	6	0
0	3	3	5	8	9	0
0	5	2	2	6	7	0
0	8	3	2	1	3	0
0	0	0	0	0	0	0

I
Image

Zero padding

$$I_f(i, j) = \sum_{k,l} I(i+k, j+l)z(k,l)$$

\otimes

a	b	c
d	e	f
g	h	i

z
Filter

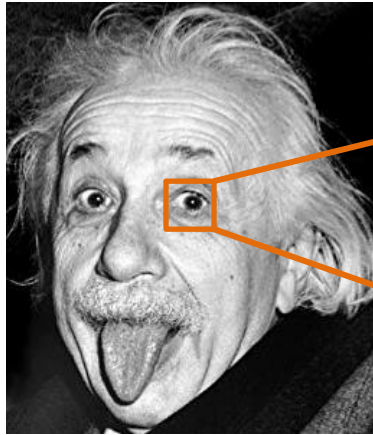
=

y			

I_f
Filtered image

$$y = 2e + f + h + 2i$$

BOUNDARY



0	0	0	0	0	0	0
0	2	1	4	4	7	0
0	1	2	2	3	6	0
0	3	3	5	8	9	0
0	5	2	2	6	7	0
0	8	3	2	1	3	0
0	0	0	0	0	0	0

I
Image

Zero padding

\otimes

a	b	c
d	e	f
g	h	i

z
Filter

=

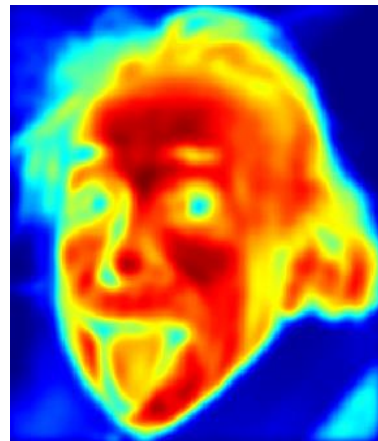
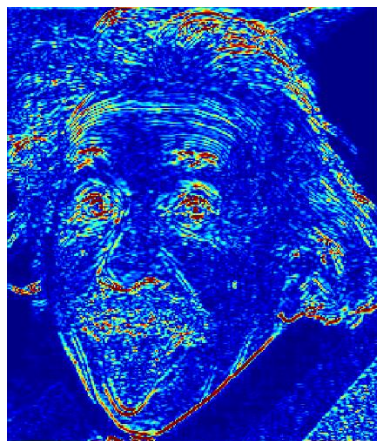
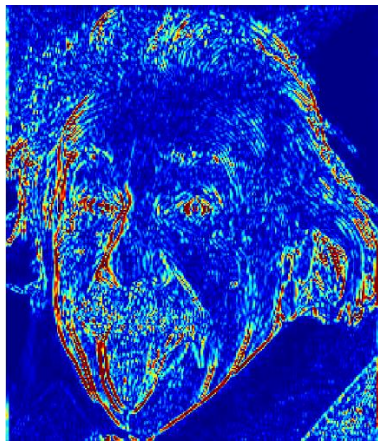
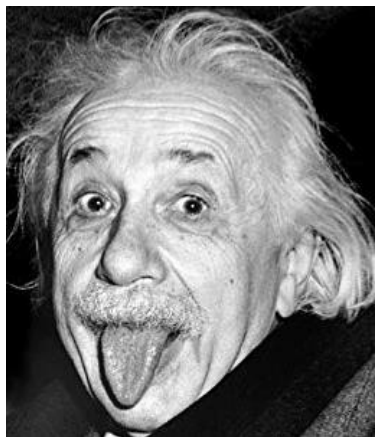
				y

I_f
Filtered image

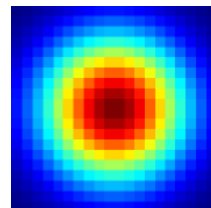
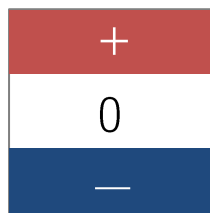
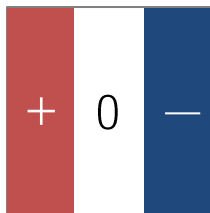
$$I_f(i, j) = \sum_{k,l} I(i+k, j+l)z(k,l)$$

$$y = 8a + 9b + 6d + 7e + g + 3h$$

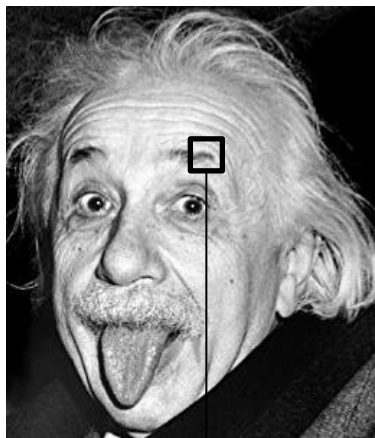
FILTERING AS A FEATURE EXTRACTION



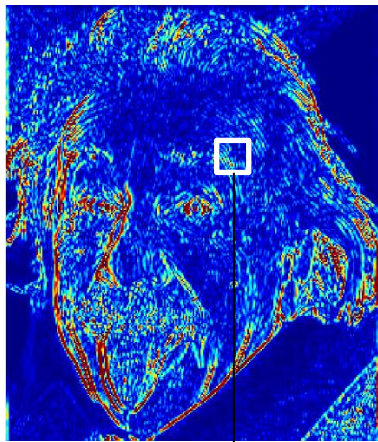
...



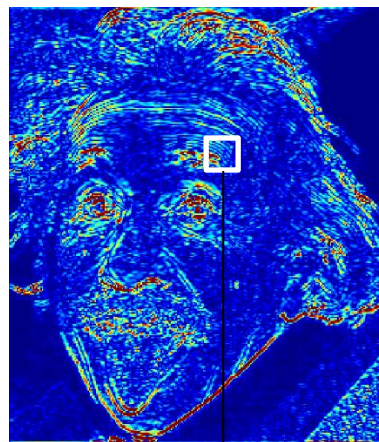
FILTERING AS A FEATURE EXTRACTION



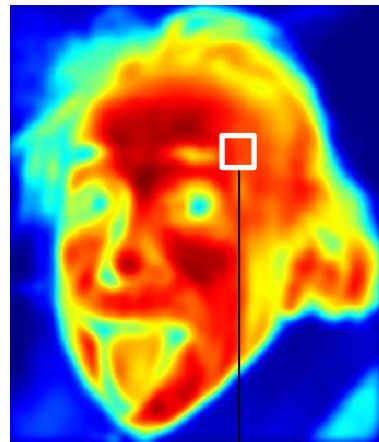
$$I(x_1) = (212, 200, 221)$$



$$f_1(x) = 0.3$$



$$f_2(x) = -0.1$$



$$f_3(x) = 0.9$$

...

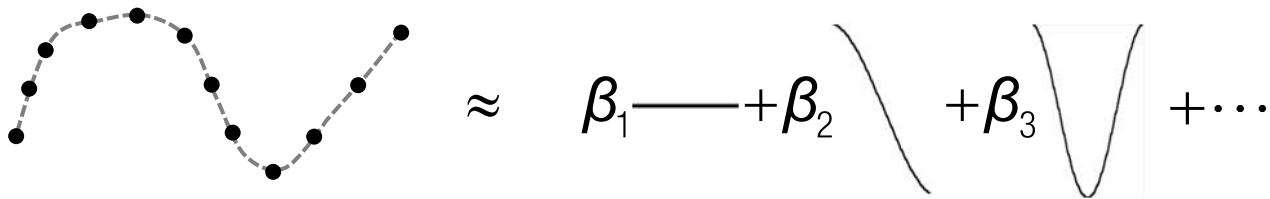
...

ANALOGY FROM FOURIER TRANSFORM



Joseph Fourier (1768-1830)

Any signal can be rewritten as a linear combination of sine and cosine of different frequencies.



Time series signal

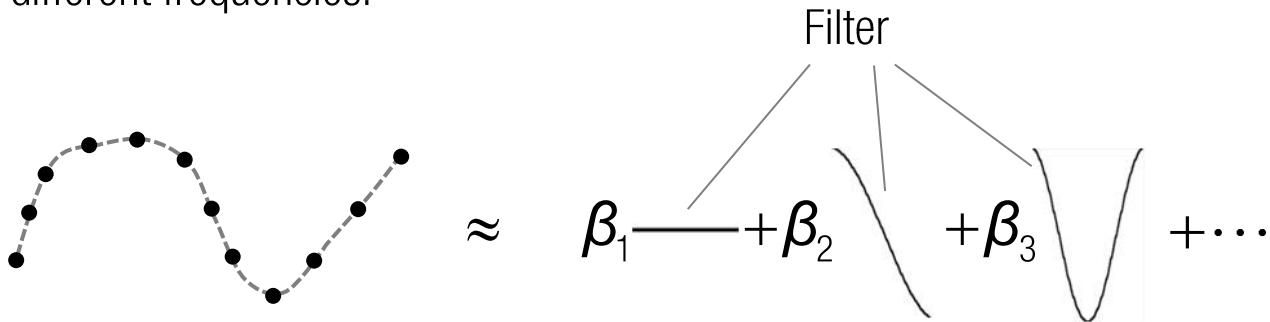
$$[x_1 \quad \cdots \quad x_t]$$

ANALOGY FROM FOURIER TRANSFORM



Joseph Fourier (1768-1830)

Any signal can be rewritten as a linear combination of sine and cosine of different frequencies.

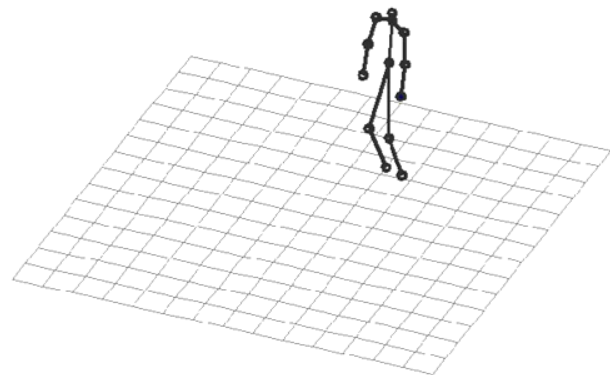


Time series signal

$$[x_1 \quad \dots \quad x_t] \longrightarrow [\beta_1 \quad \dots \quad \beta_n]$$

Frequency response

→ Unique descriptor of the signal

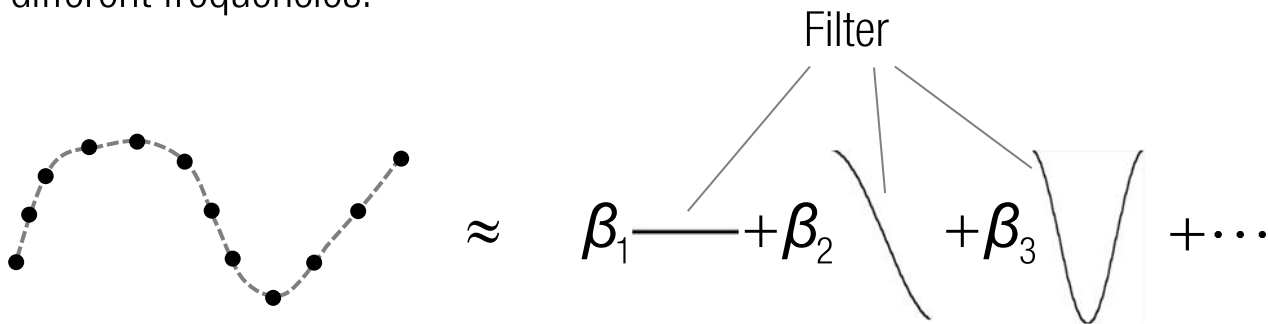


ANALOGY FROM FOURIER TRANSFORM



Joseph Fourier (1768-1830)

Any signal can be rewritten as a linear combination of sine and cosine of different frequencies.



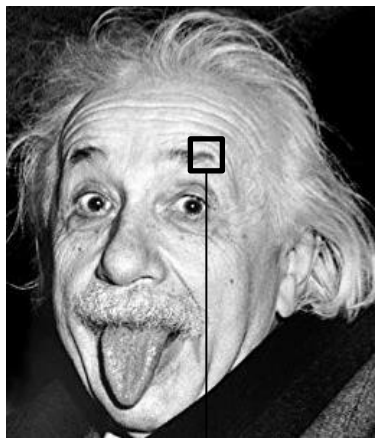
Time series signal

$$[x_1 \quad \dots \quad x_t] \longrightarrow [\beta_1 \quad \dots \quad \beta_n]$$

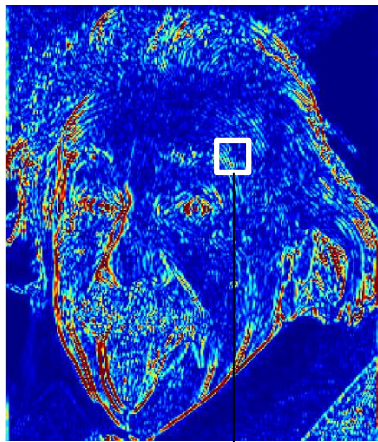
Frequency response

→ Unique descriptor of the signal

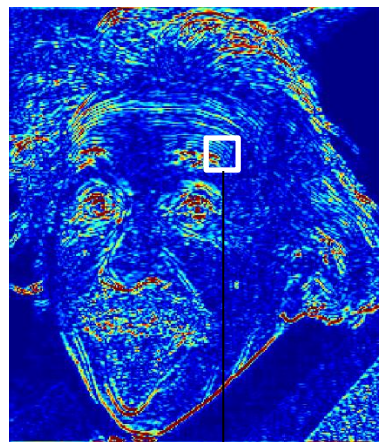
FILTERING AS A FEATURE EXTRACTION



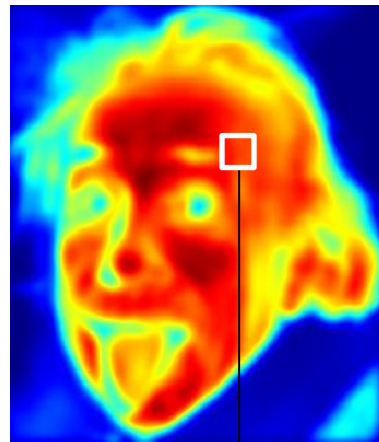
$$I(x_1) = (212, 200, 221)$$



$$f_1(x) = 0.3$$



$$f_2(x) = -0.1$$



$$f_3(x) = 0.9$$

...

...

CORRELATION VS. CONVOLUTION

Image correlation:

$$I \otimes z = I_f$$

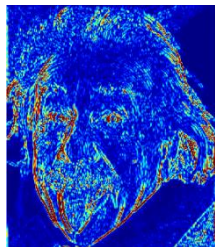
$$\sum_{k,l} I(i+k, j+l)z(k,l) = I_f(i, j)$$

2	1	4	4	7
1	2	2	3	6
3	3	5	8	9
5	2	2	6	7
8	3	2	1	3

\otimes

F

=



CORRELATION VS. CONVOLUTION

Image correlation:

$$I \otimes z = I_f$$

$$\sum_{k,l} I(i+k, j+l)z(k,l) = I_f(i, j)$$

2	1	4	4	7
1	2	2	3	6
3	3	5	8	9
5	2	2	6	7
8	3	2	1	3

\otimes



=

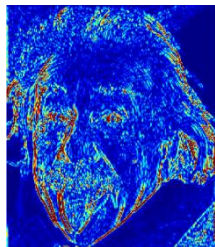


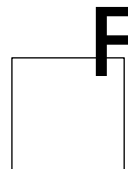
Image convolution:

$$I * z = J$$

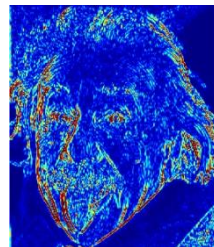
$$\sum_{k,l} I(i-k, j-l)z(k,l) = J(i, j)$$

2	1	4	4	7
1	2	2	3	6
3	3	5	8	9
5	2	2	6	7
8	3	2	1	3

*

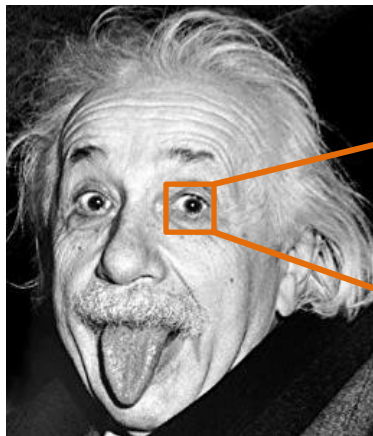


=



Flip the filter in both dimension (bottom to top, right to left)

CORRELATION VS. CONVOLUTION



Correlation

2	1	4	4	7
1	2	2	3	6
3	3	5	8	9
5	2	2	6	7
8	3	2	1	3

\otimes

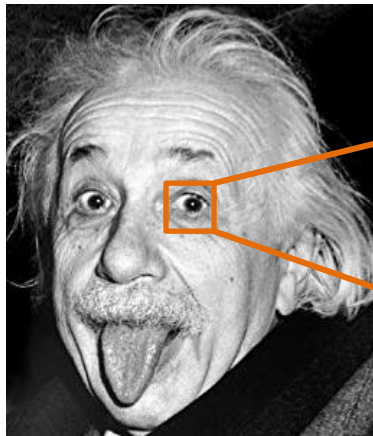
a	b	c
d	e	f
g	h	i

=

	y			

$$y = 2a + b + 4c + d + 2e + 2f + 3g + 3h + 5i$$

CORRELATION VS. CONVOLUTION



Correlation

2	1	4	4	7
1	2	2	3	6
3	3	5	8	9
5	2	2	6	7
8	3	2	1	3

\otimes

a	b	c
d	e	f
g	h	i

=

	y			

$$y = 2a + b + 4c + d + 2e + 2f + 3g + 3h + 5i$$

Convolution

2	1	4	4	7
1	2	2	3	6
3	3	5	8	9
5	2	2	6	7
8	3	2	1	3

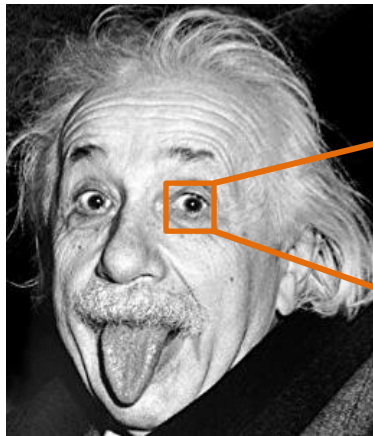
*

a	b	c
d	e	f
g	h	i

=

	z			

CORRELATION VS. CONVOLUTION



Correlation

2	1	4	4	7
1	2	2	3	6
3	3	5	8	9
5	2	2	6	7
8	3	2	1	3

\otimes

<i>a</i>	<i>b</i>	<i>c</i>
<i>d</i>	<i>e</i>	<i>f</i>
<i>g</i>	<i>h</i>	<i>i</i>

=

	<i>y</i>			

$$y = 2a + b + 4c + d + 2e + 2f + 3g + 3h + 5i$$

Convolution

2	1	4	4	7
1	2	2	3	6
3	3	5	8	9
5	2	2	6	7
8	3	2	1	3

\otimes

Flip the filter

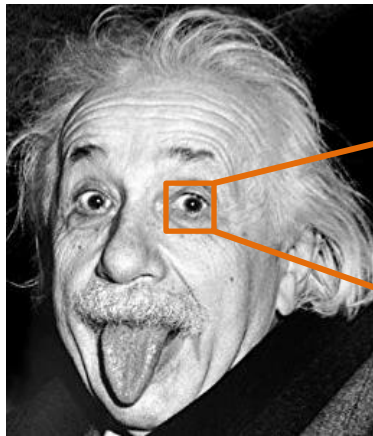
<i>i</i>	<i>h</i>	<i>g</i>
<i>f</i>	<i>e</i>	<i>d</i>
<i>c</i>	<i>b</i>	<i>a</i>

=

	<i>z</i>			

$$z = 2i + h + 4g + f + 2e + 2d + 3c + 3b + 5a$$

CORRELATION VS. CONVOLUTION



Correlation

2	1	4	4	7
1	2	2	3	6
3	3	5	8	9
5	2	2	6	7
8	3	2	1	3

\otimes

a	b	c
d	e	f
g	h	i

=

		y		

$$y = 2a + 2b + 3c + 3d + 5e + 8f + 2g + 3h + 6i$$

Convolution

2	1	4	4	7
1	2	2	3	6
3	3	5	8	9
5	2	2	6	7
8	3	2	1	3

\otimes

Flip the filter

i	h	g
f	e	d
c	b	a

=

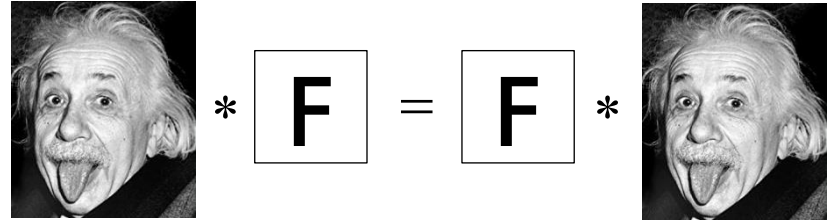
		z		

$$z = 2i + 2h + 3h + 3f + 5e + 8d + 2c + 3b + 6a$$

PROPERTIES OF CONVOLUTION

Commutative:

$$f * g = g * f$$



A filter can be filtered by an image.

PROPERTIES OF CONVOLUTION

Commutative:

$$f * g = g * f$$

Associative:

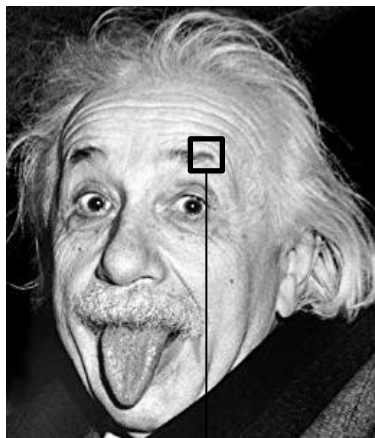
$$(f * g) * h = f * (g * h)$$

$$\left[\text{Einstein} * \boxed{\mathbf{F}} \right] * \boxed{\mathbf{G}} = \text{Einstein} * \left[\boxed{\mathbf{F}} * \boxed{\mathbf{G}} \right]$$

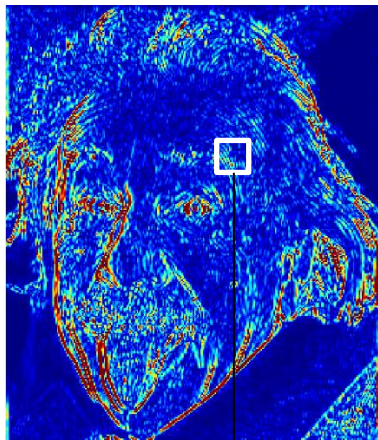
The diagram illustrates the associative property of convolution. It shows two equivalent ways to compute the convolution of an image (represented by a black and white photo of Albert Einstein sticking his tongue out) with two filters, F and G. In the first expression, the Einstein image is convolved with filter F, and the result is then convolved with filter G. In the second expression, filters F and G are first convolved together, and then the Einstein image is convolved with the result. The Einstein image is represented by a square box containing the photo, and the filters F and G are represented by square boxes with their respective letters.

A composition of convolutions can be pre-computed.

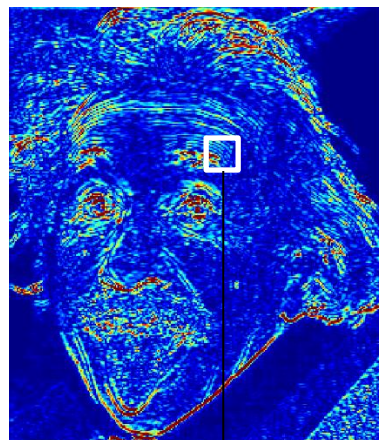
FILTERING AS A FEATURE EXTRACTION



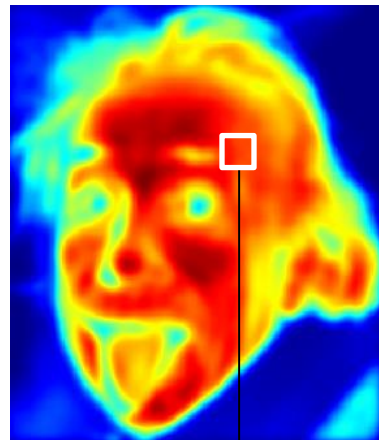
$$I(x_1) = (212, 200, 221)$$



$$f_1(x) = 0.3$$



$$f_2(x) = -0.1$$

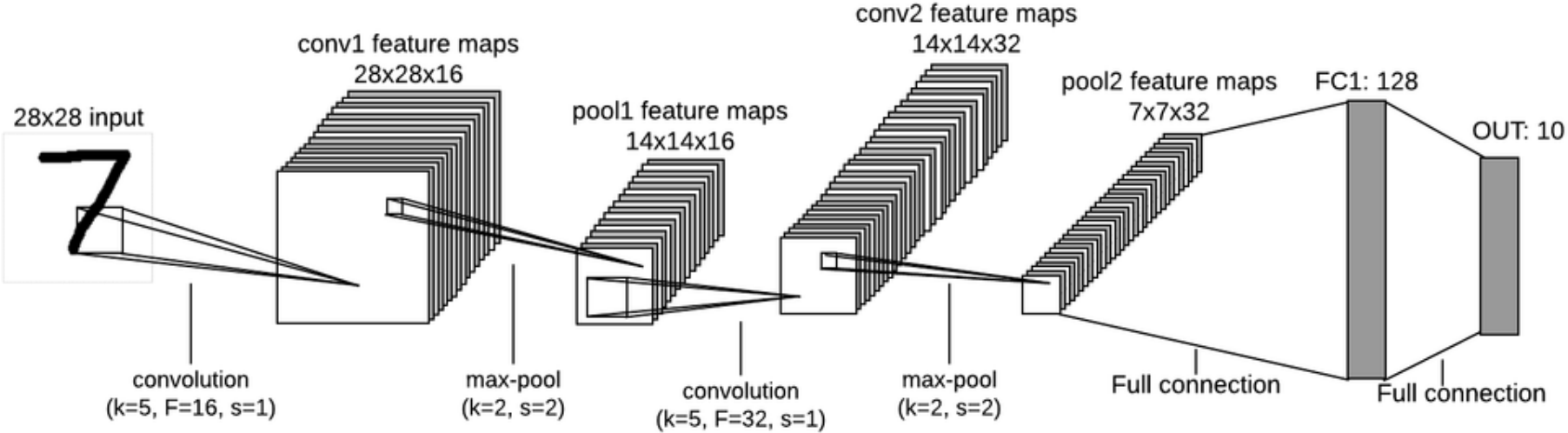
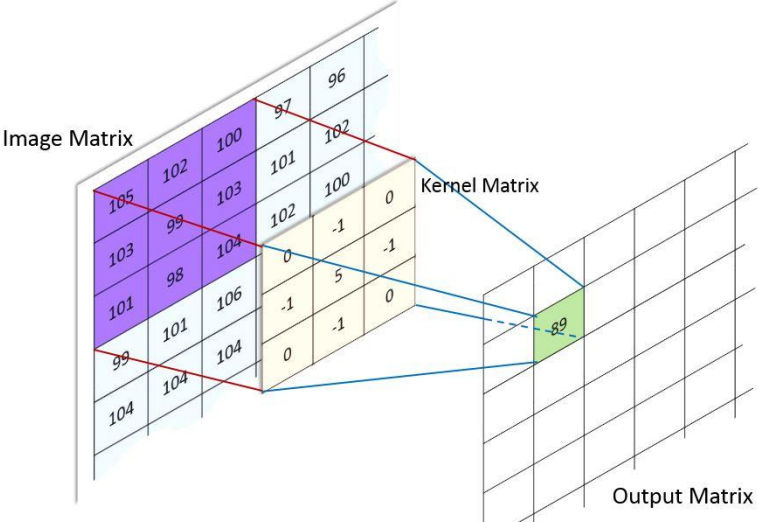


$$f_3(x) = 0.9$$

...

...

CONVOLUTIONAL NEURAL NETWORK



RECALL: EDGE RESPONSE

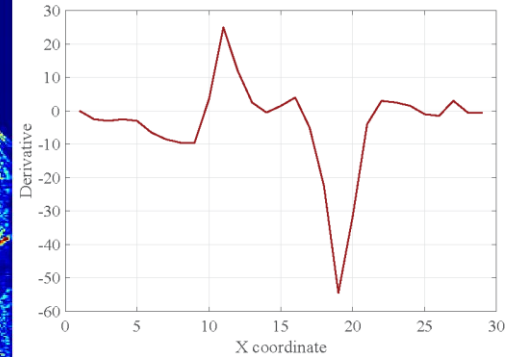
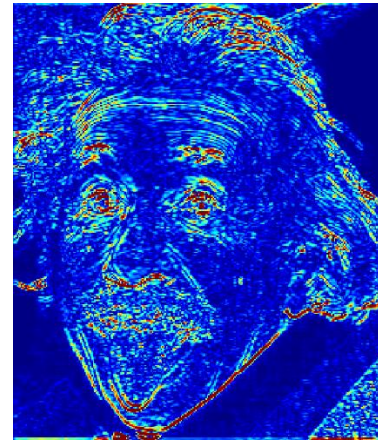
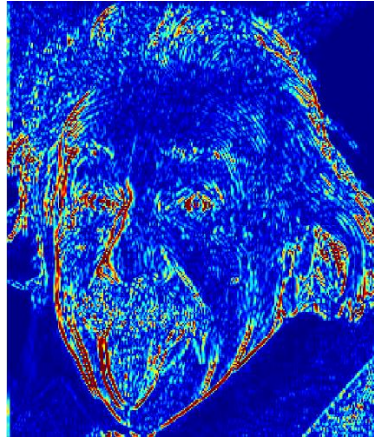
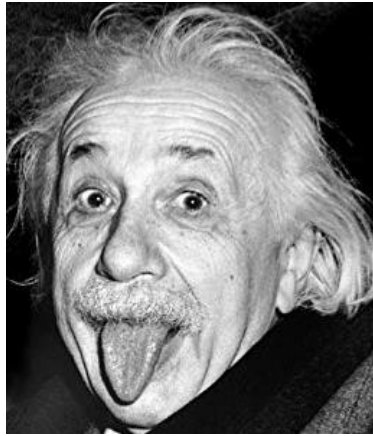
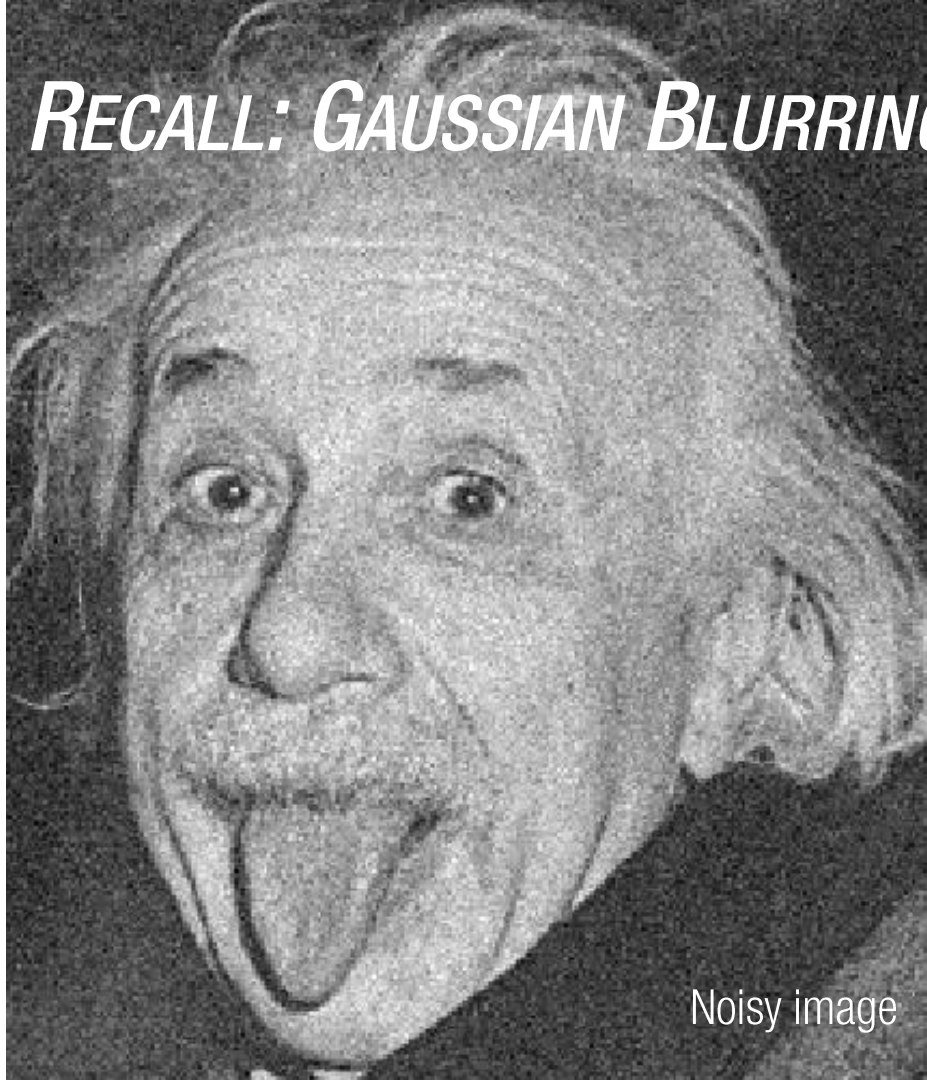
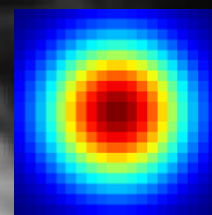


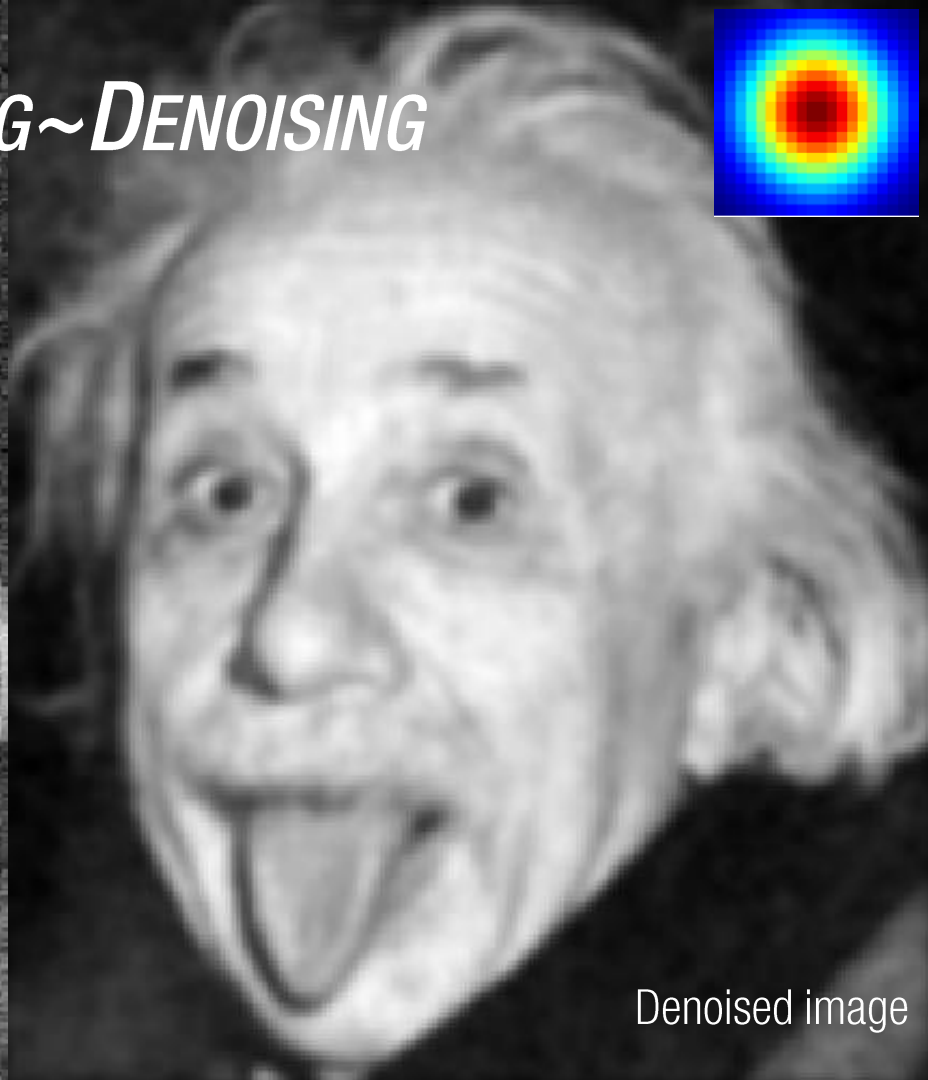
Image differentiation can be used for edge detection but it is VERY noisy.

How to detect edge reliably?

RECALL: GAUSSIAN BLURRING ~ DENOISING

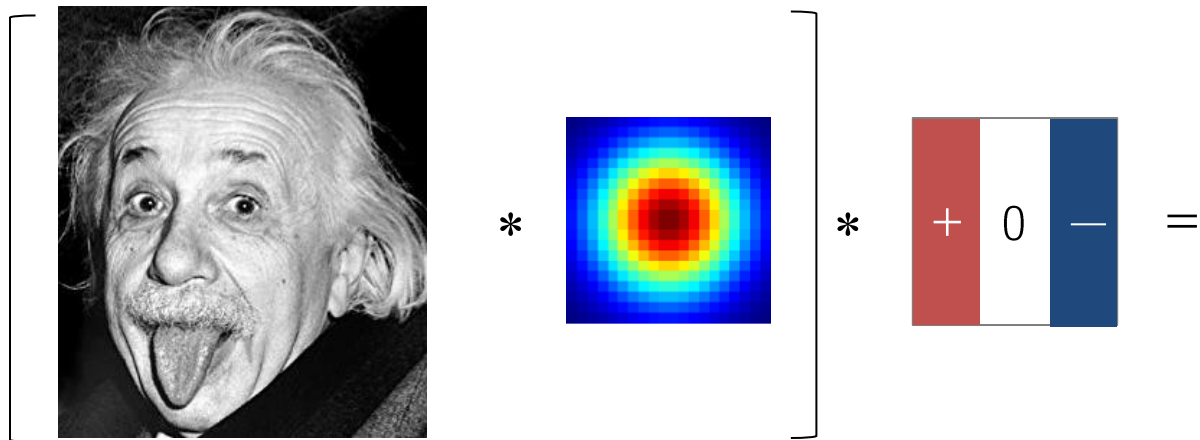


Noisy image

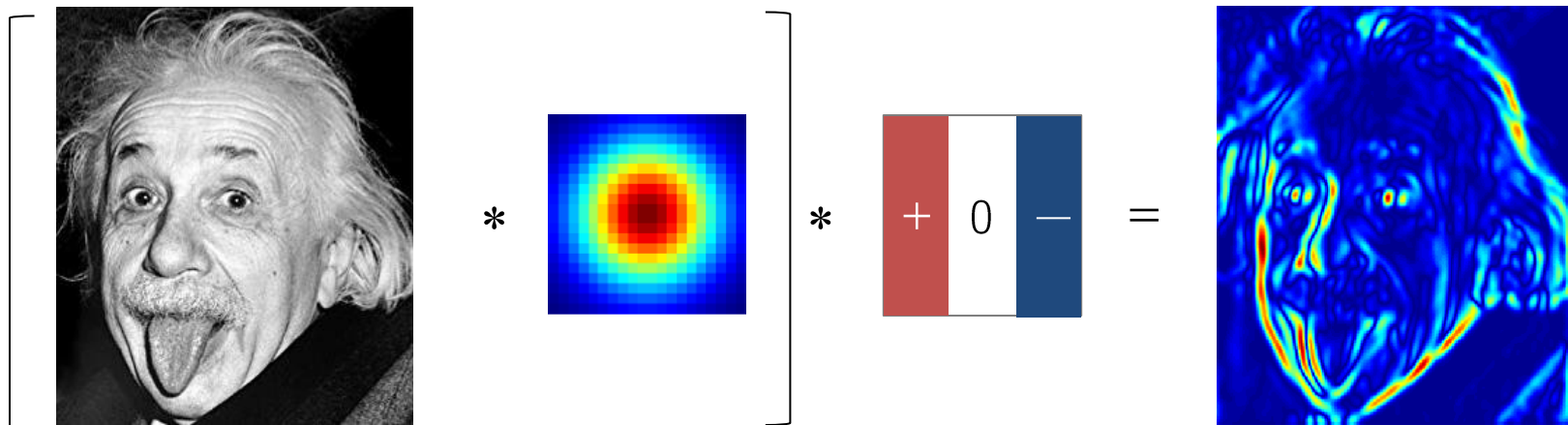


Denoised image

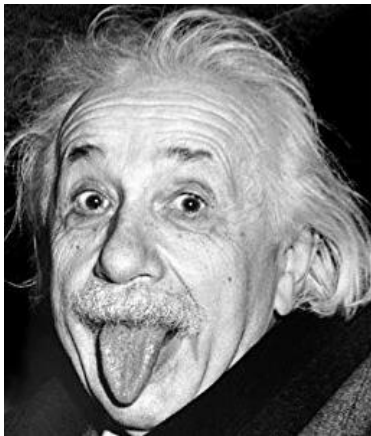
STRATEGY: DENOISE AND DIFFERENTIATE



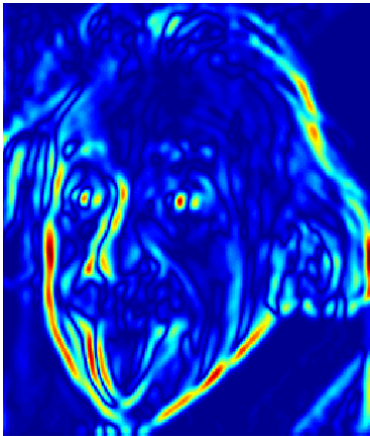
STRATEGY: DENOISE AND DIFFERENTIATE



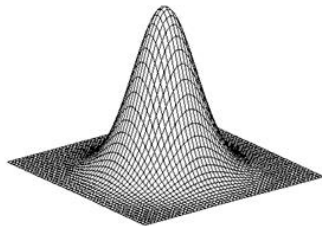
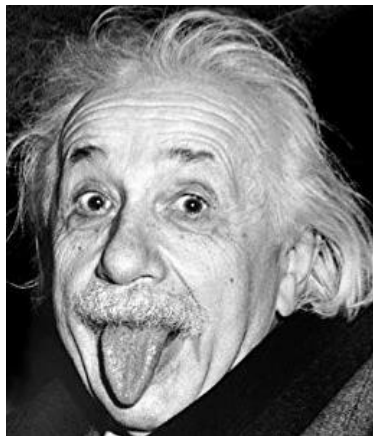
ASSOCIATIVITY



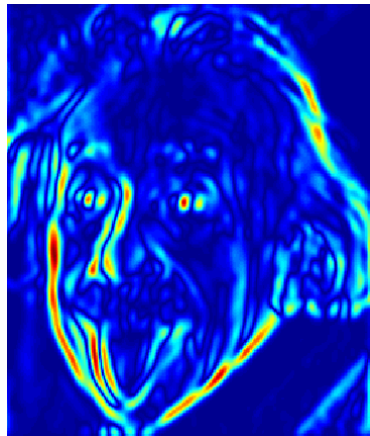
$$* \left[\begin{array}{c} \text{Heatmap} \\ \frac{1}{2\pi\sigma^2} e^{-\frac{k^2+l^2}{2\sigma^2}} \end{array} \right] * \left[\begin{array}{c} \text{Sobel Kernel} \\ \frac{\partial}{\partial u} \end{array} \right] =$$



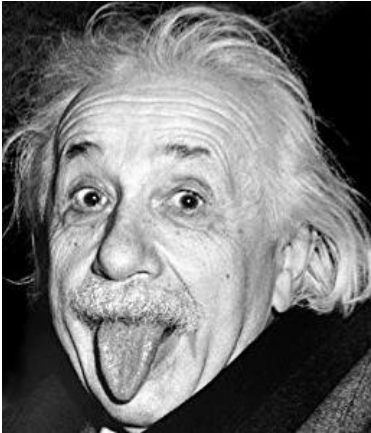
COMMUTATIVITY



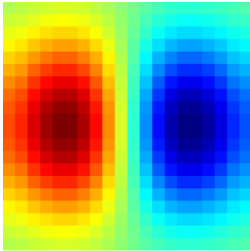
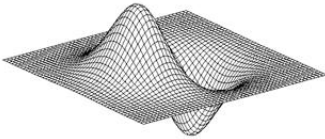
$$* \left[\begin{array}{|c|c|c|} \hline + & 0 & - \\ \hline \end{array} * \begin{array}{|c|} \hline \text{Heatmap} \\ \hline \end{array} \right] =$$
$$\frac{\partial}{\partial u} \frac{1}{2\pi\sigma^2} e^{-\frac{k^2+l^2}{2\sigma^2}}$$



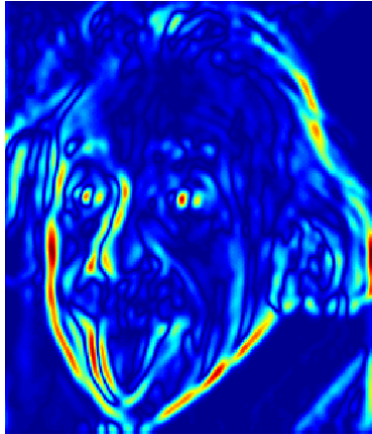
SOBEL FILTER



*



=

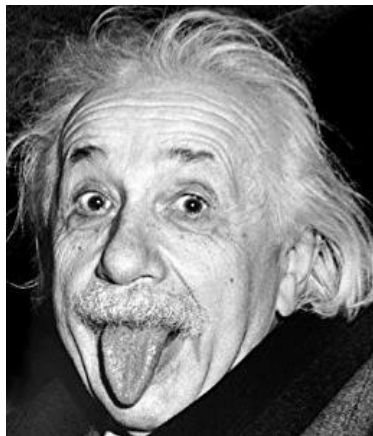


$$\frac{\partial}{\partial u} \frac{1}{2\pi\sigma^2} e^{-\frac{k^2+l^2}{2\sigma^2}}$$

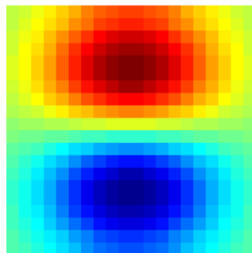
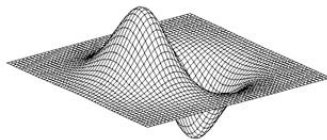
Sobel filter: derivative of Gaussian filter, e.g.,

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

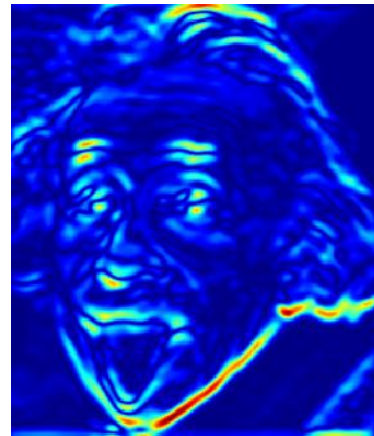
SOBEL FILTER



*



=



$$\frac{\partial}{\partial v} \frac{1}{2\pi\sigma^2} e^{-\frac{k^2+l^2}{2\sigma^2}}$$

Sobel filter: derivative of Gaussian filter, e.g.,

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

