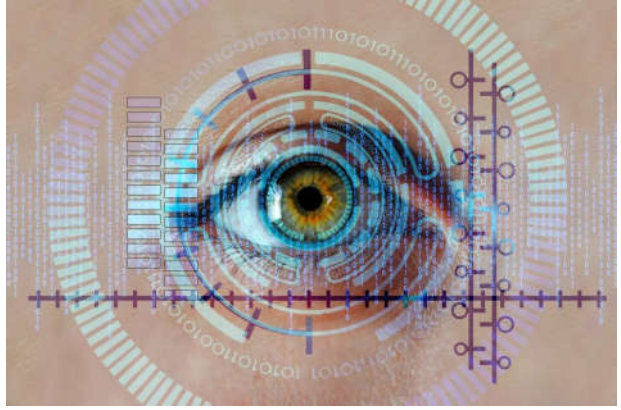
北京理工大学
BEIJING INSTITUTE OF TECHNOLOGY

Computer Vision



Lecture 2 Image Filtering

School of Computer Science and Technology

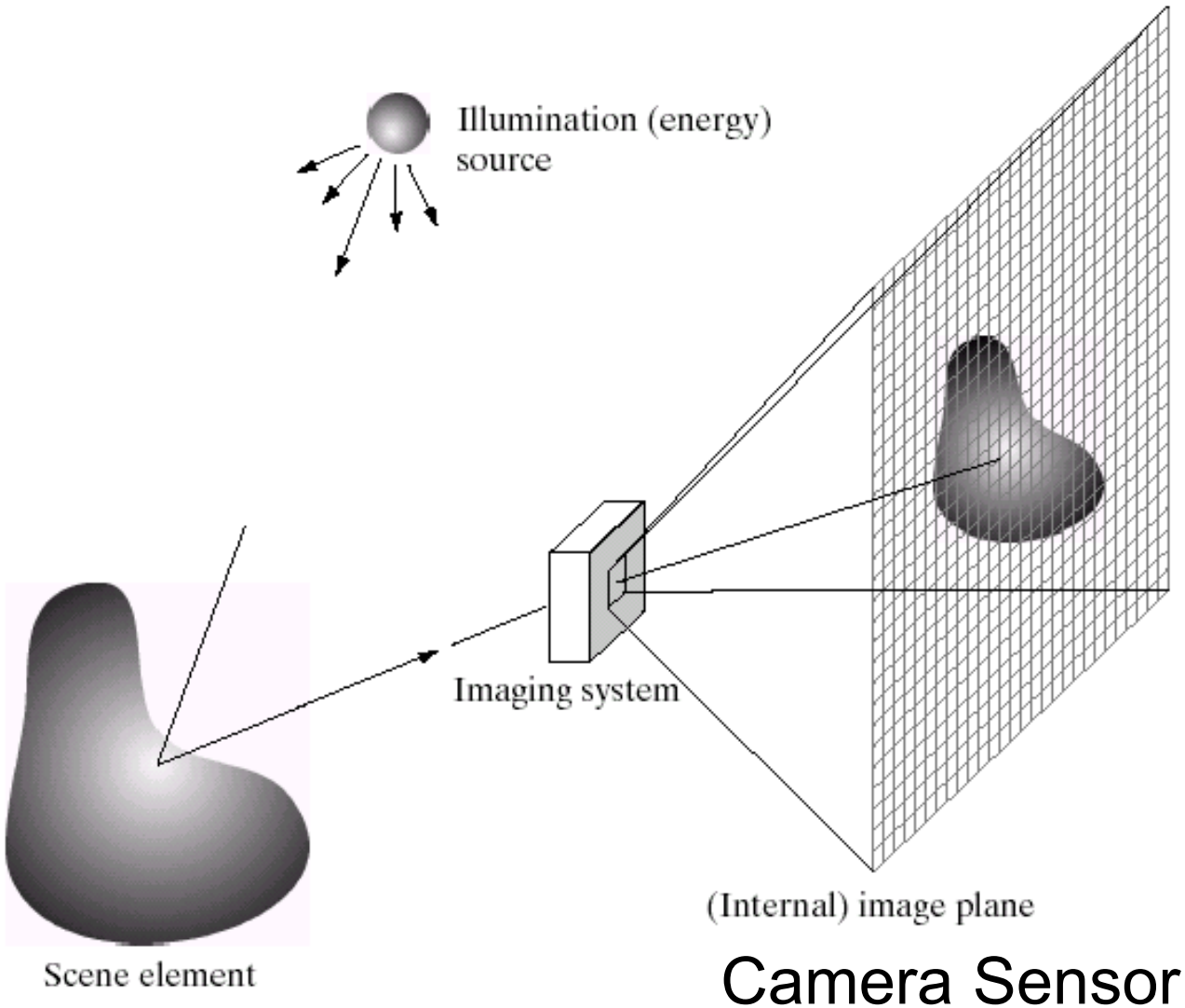
Ying Fu


fuying@bit.edu.cn

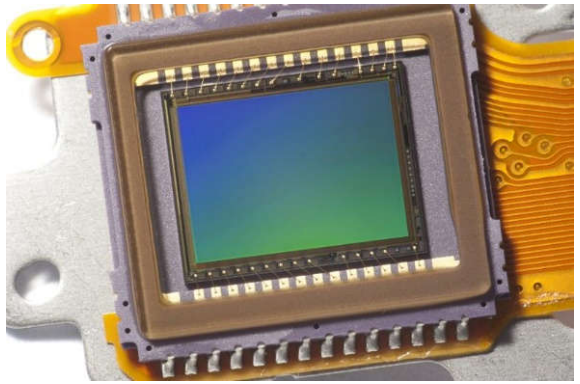
Outline

- What is an image?
- Image Filtering

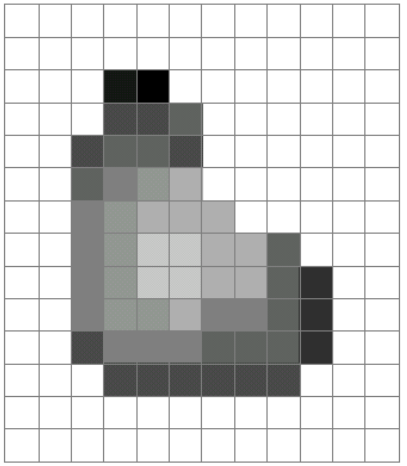
Digital image



Camera Sensor



CMOS sensor



Output Image

Digital image

- Signal: function depending on some variable with physical meaning.

- Image: sampling of that function.
  - 2 variables: xy coordinates
  - 3 variables: xy + time (video)
  - ‘Brightness’ is the value of the function for visible light

Digital image

- Types of Images

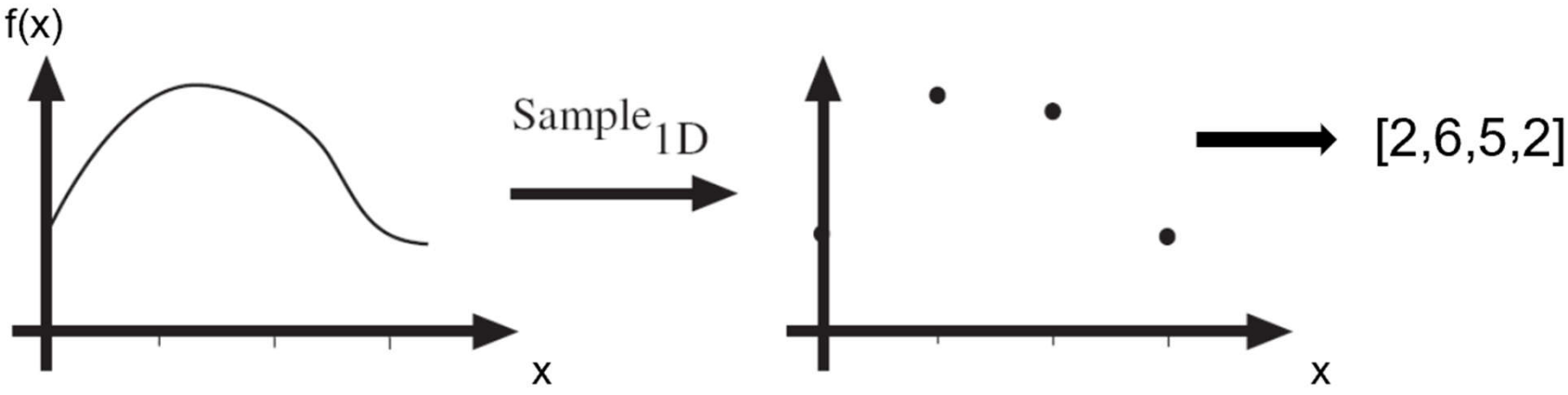
Binary

Graysc
ale

Color

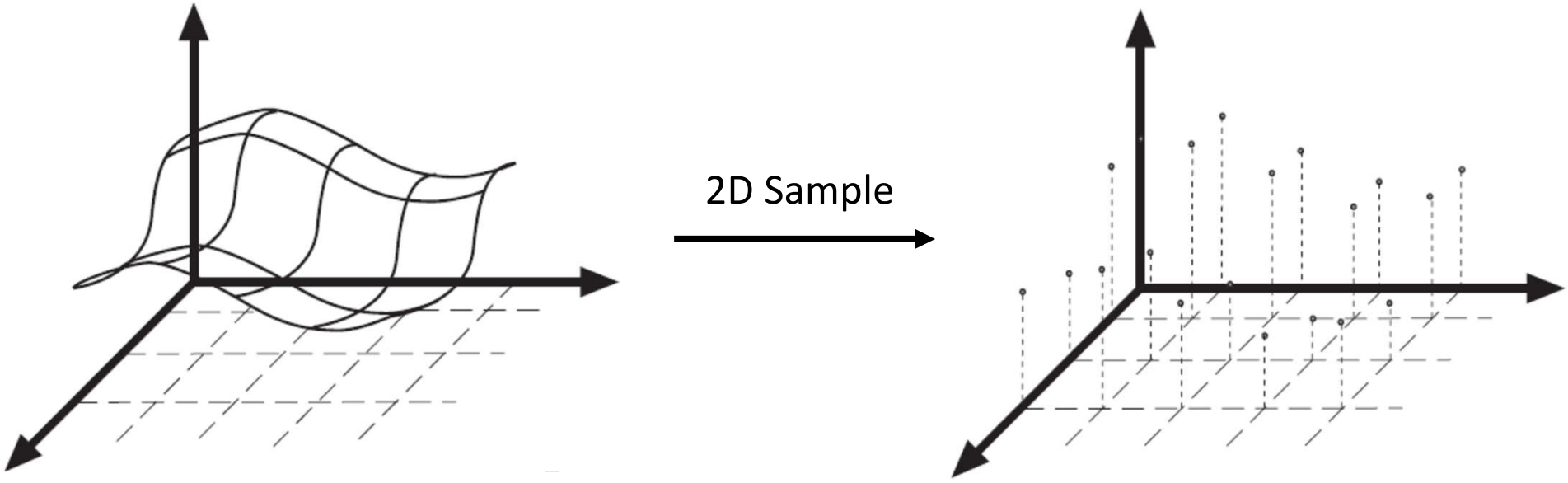
Digital image

- Sampling in 1D
  - Sampling in 1D takes a function and returns a vector whose elements are values of that function at the sample points.



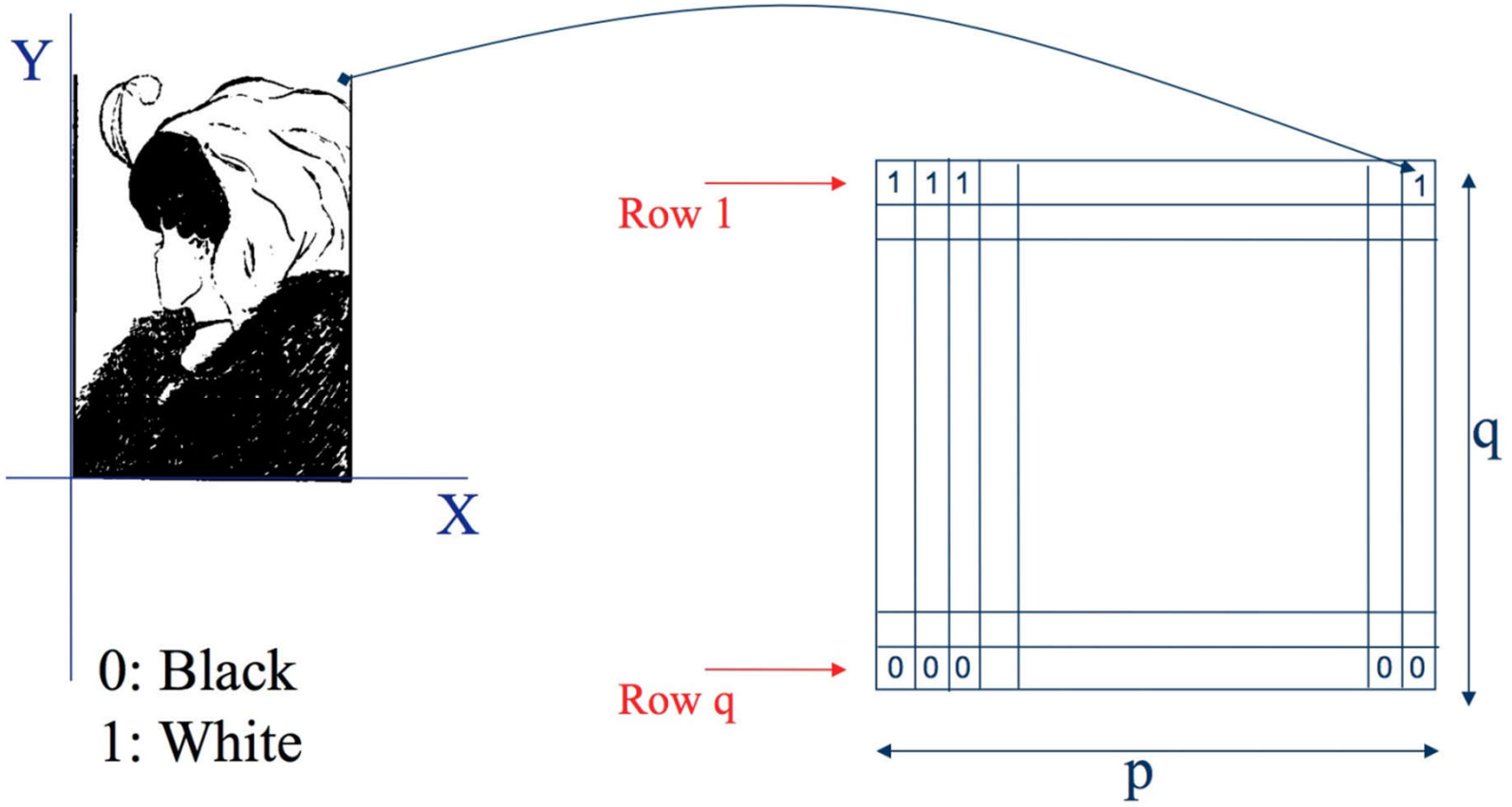
Digital image

- Sampling in 2D
  - Sampling in 2D takes a function and returns a matrix.



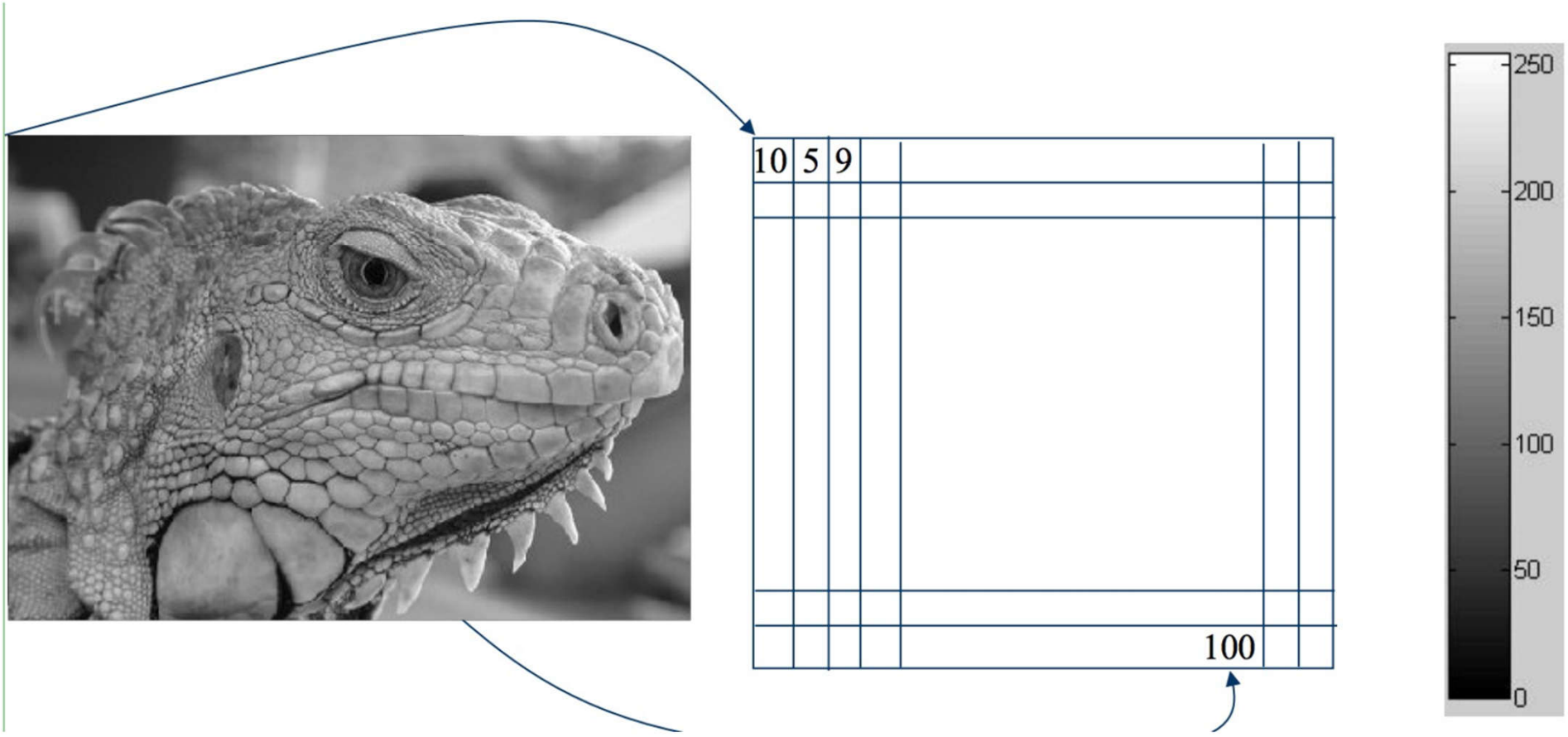
Digital image

- Binary image representation



Digital image

- Grayscale image representation



Digital image

- Color image representation

B channel

G channel

R channel

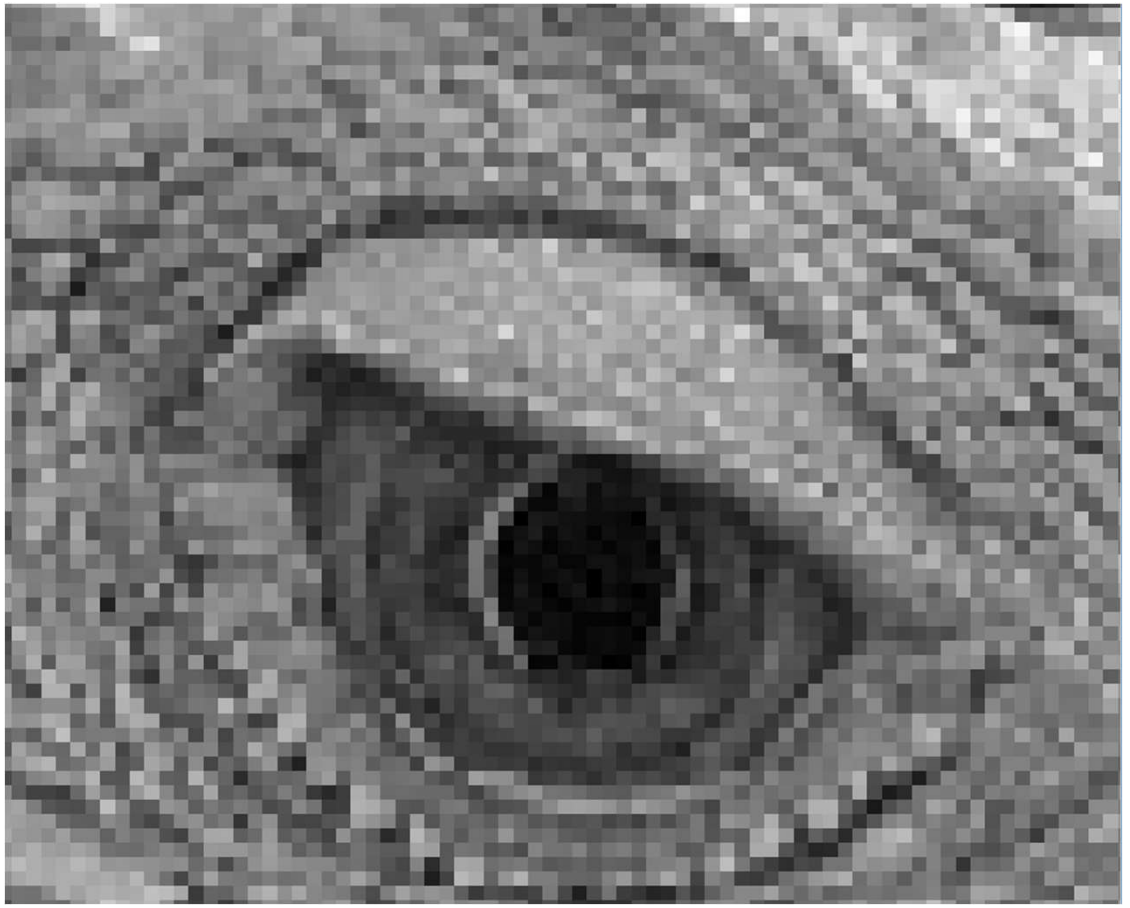
Digital image

- Color image – one channel

R channel

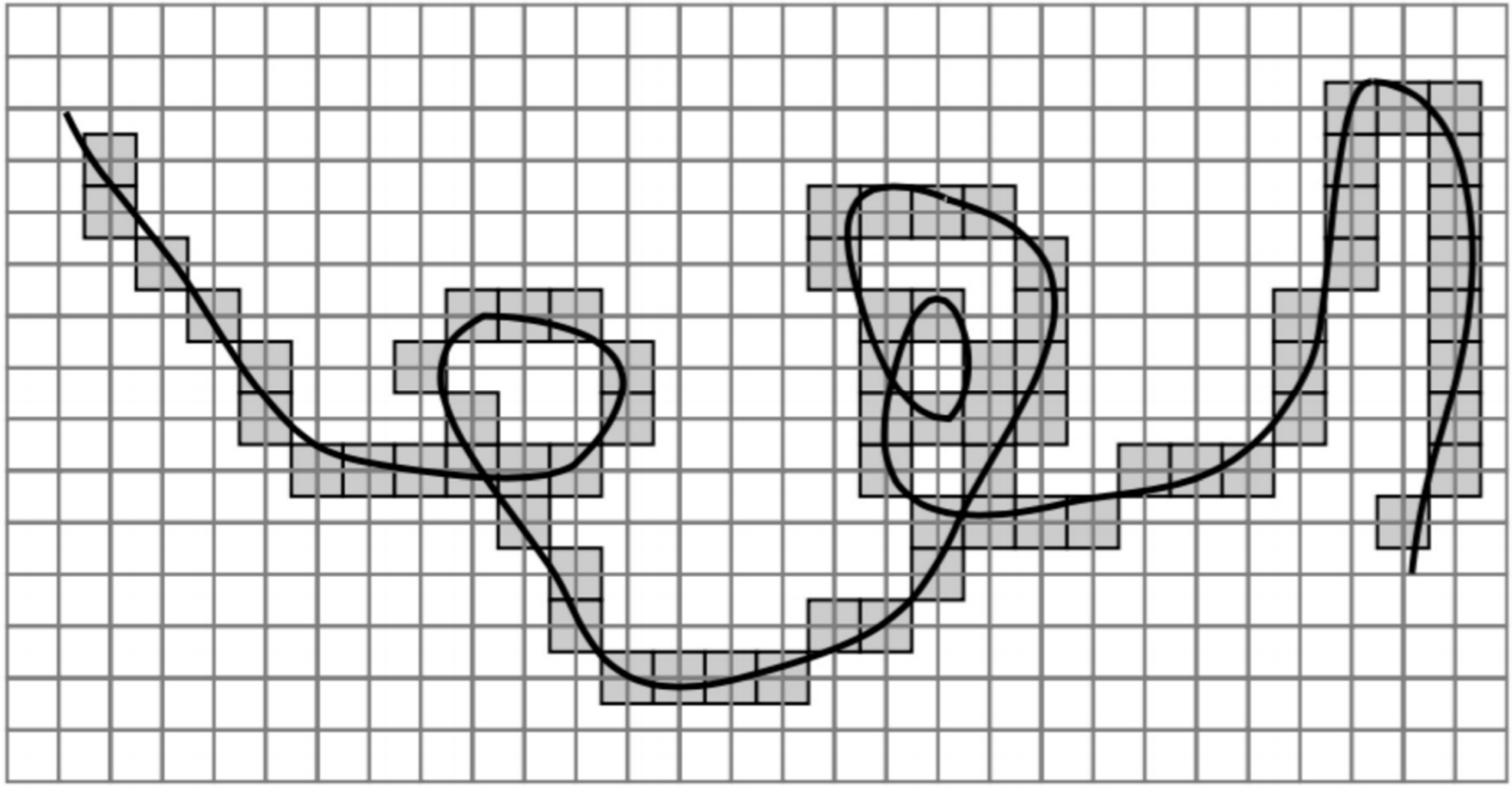
Digital image

- Digital images are sampled
  - What happens when we zoom into the images we capture?



Digital image

- Errors due to sampling

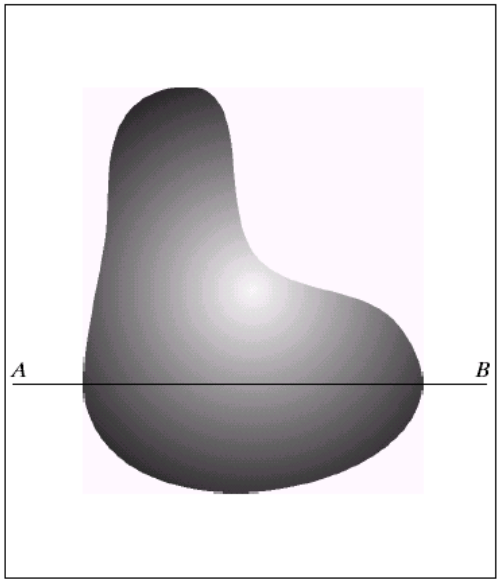


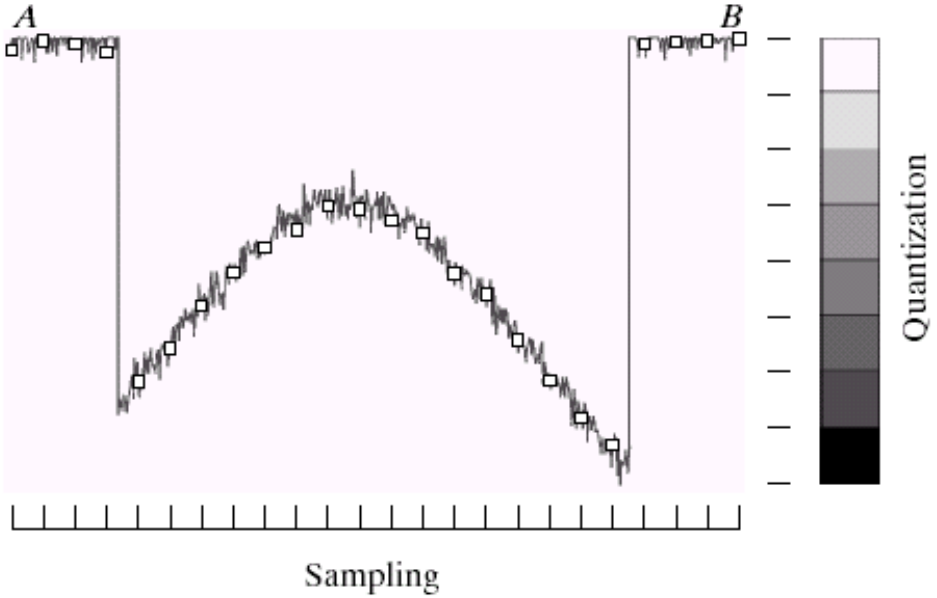
Digital image

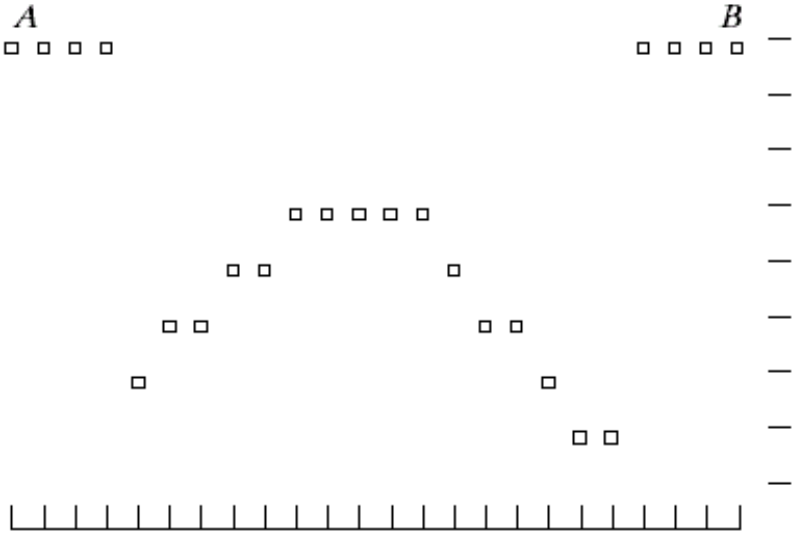
- Resolution is a **sampling** parameter, defined in dots per inch (DPI) or equivalent measures of spatial pixel density

Digital image

- Quantization



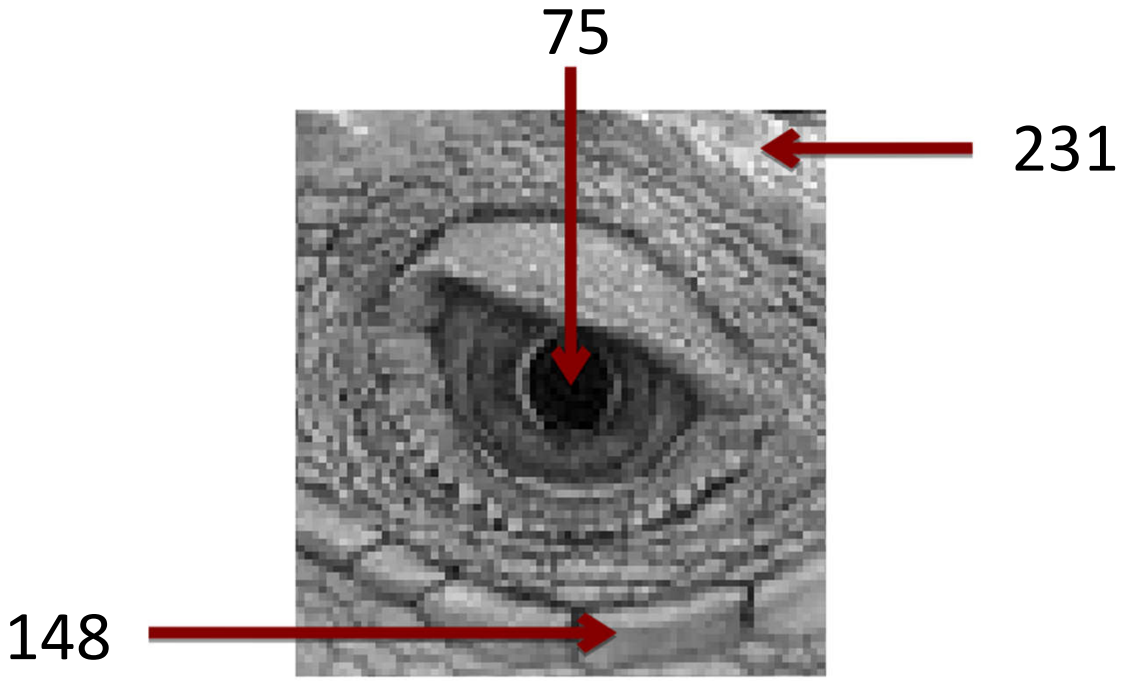




Digital image

- Images are sampled and quantized
- An image contains discrete number of pixels

  –Pixel value: •“grayscale” (or “intensity”): [0,255]

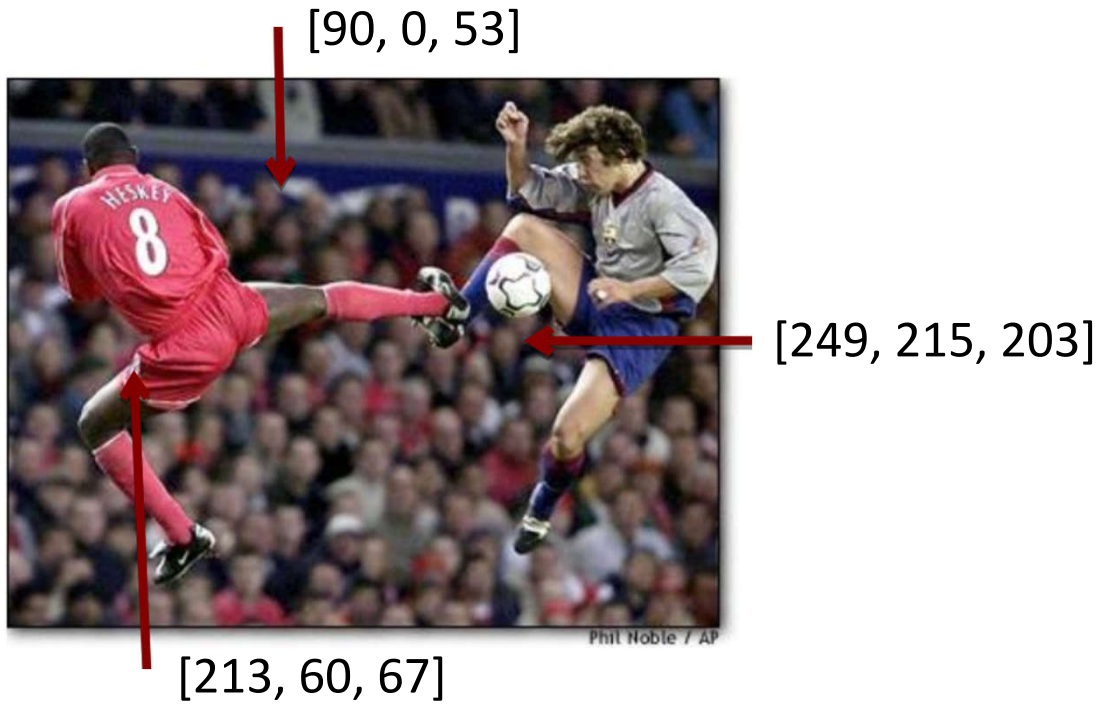


Digital image

- Images are sampled and quantized
- An image contains discrete number of pixels

  –Pixel value: •"grayscale" (or "intensity"): [0,255]

  •"color" –RGB: [R, G, B]

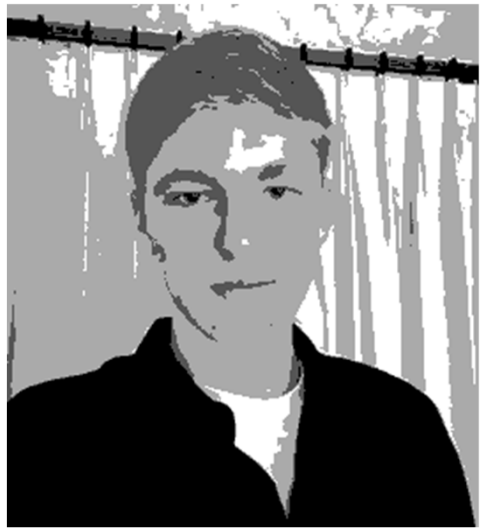


Digital image

- Quantization Effects – Radiometric Resolution

8 bit – 256 levels

4 bit – 16 levels



2 bit – 4 levels

1 bit – 2 levels

We often call this *bit depth.*
For photography, this is also related to *dynamic range*.

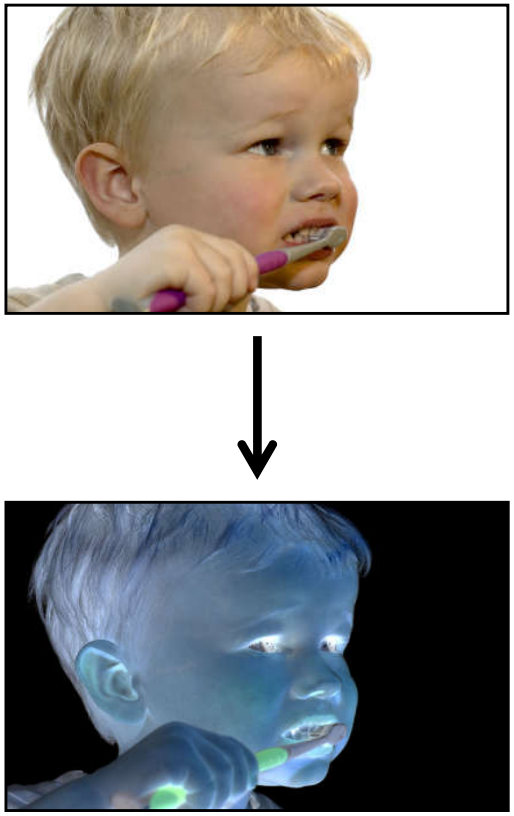
Outline

- What is an image?
- Image Filtering
  - Types of image transformations
  - Point image processing
  - Linear shift-invariant image filtering
  - Convolution

Digital image

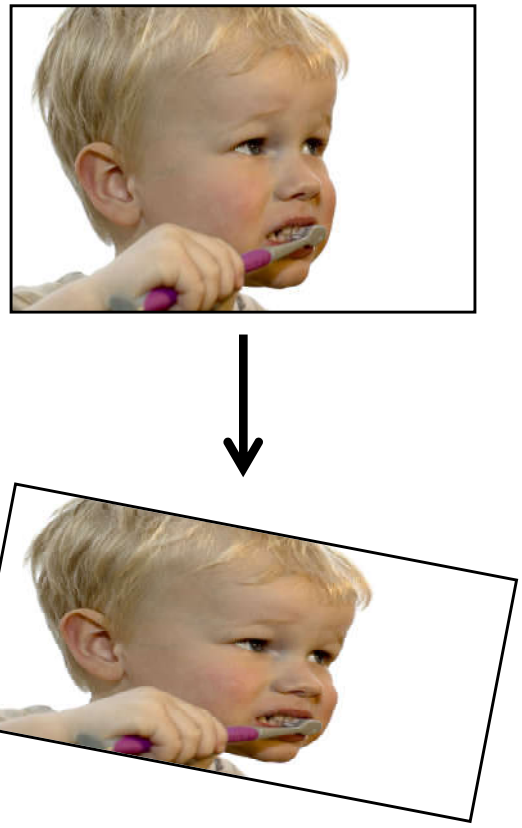
- What types of image transformations can we do?

Filtering



changes pixel *values*

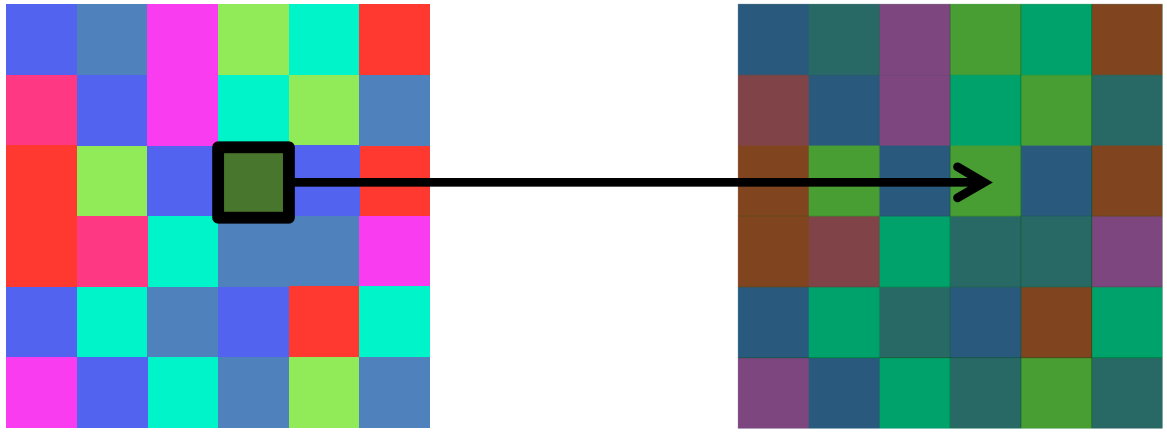
Warping



changes pixel *locations*

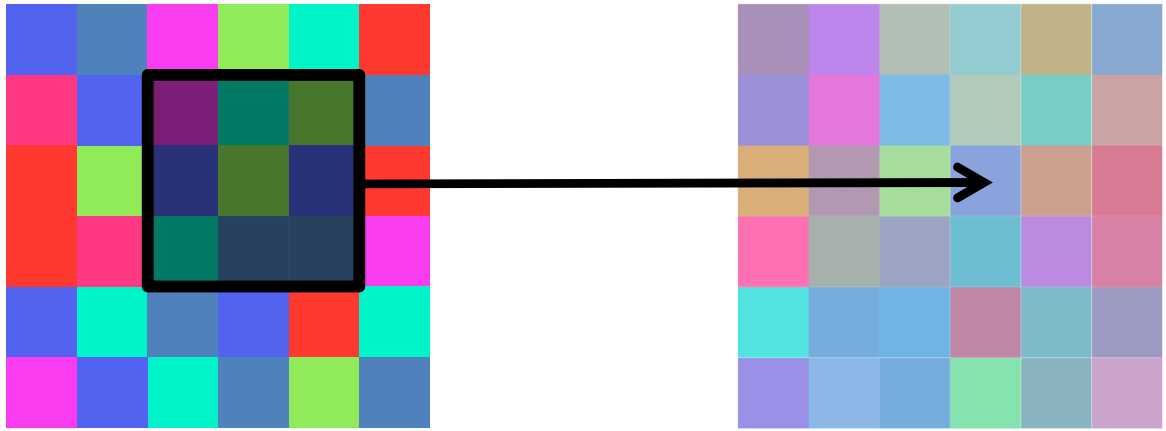
Image operation

- Point operation



point processing

- Neighborhood operation



“filtering”

Outline

- What is an image?
- Image Filtering
  - Types of image transformations
  - Point image processing
  - Linear shift-invariant image filtering
  - Convolution

Point processing

- Examples of point processing

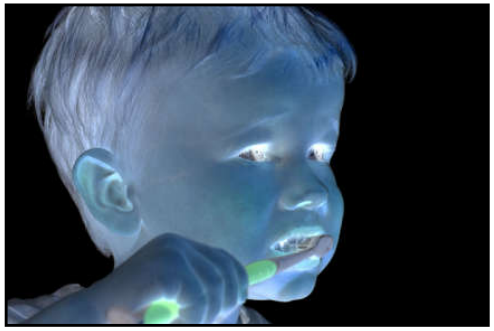
original

darken

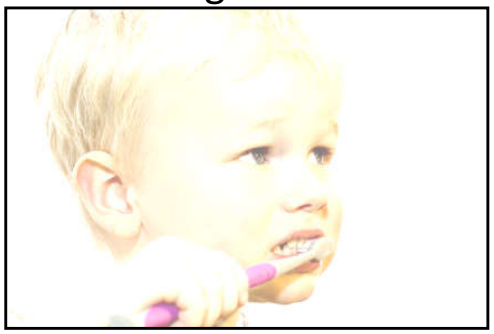
lower contrast

non-linear lower contrast

invert



lighten



raise contrast

non-linear raise contrast

Point processing

- How to implement these?

original

x

darken

$x - 128$

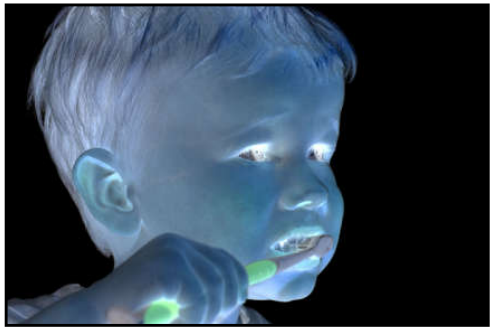
lower contrast

$\frac{x}{2}$

non-linear lower contrast

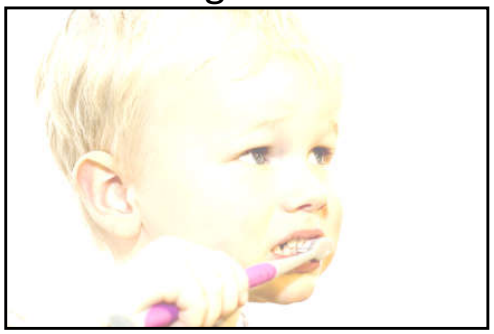
$\left(\frac{x}{255}\right)^{1/3} \times 255$

invert



$255 - x$

lighten



$x + 128$

raise contrast

$x \times 2$

non-linear raise contrast

$\left(\frac{x}{255}\right)^{2} \times 255$

Point processing

- Many other types of point processing

camera output

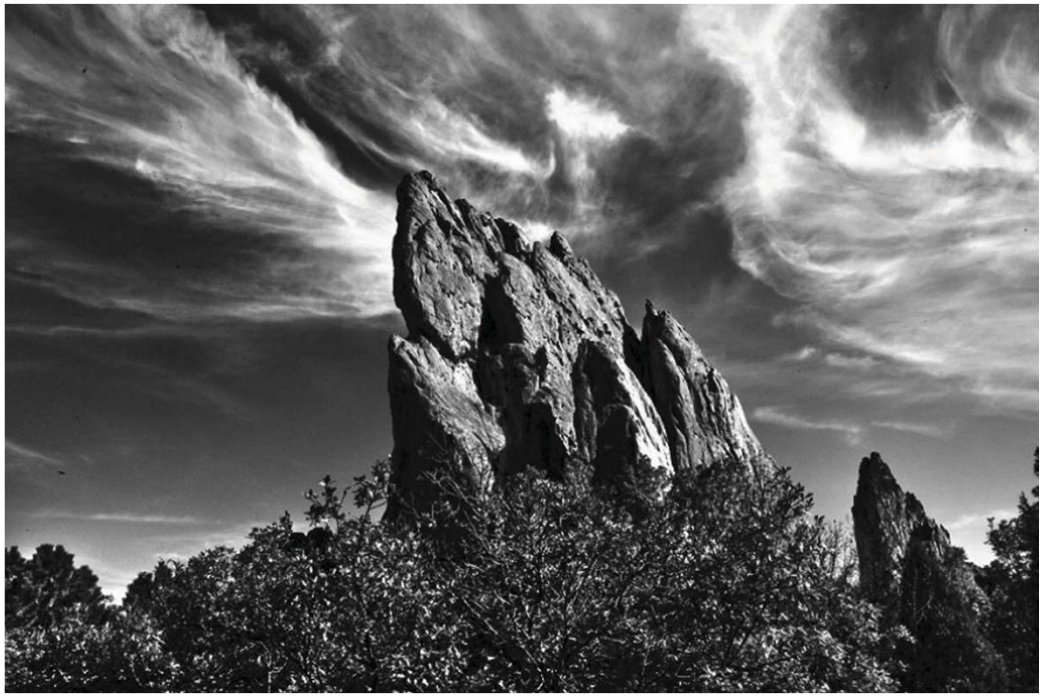
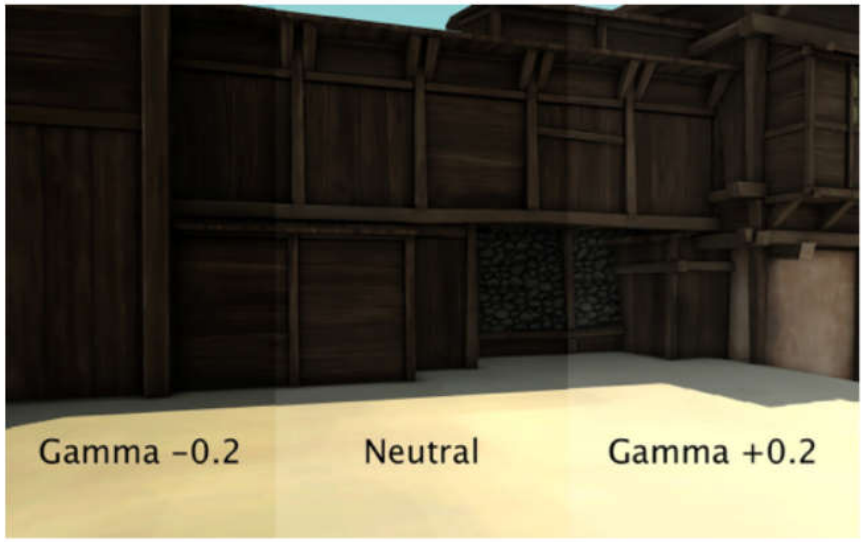


image after stylistic tonemapping

[Bae et al., SIGGRAPH 2006]

Point processing

- Many other types of point processing



Gamma curve

图1 CRT显示器的亮度响应曲线图

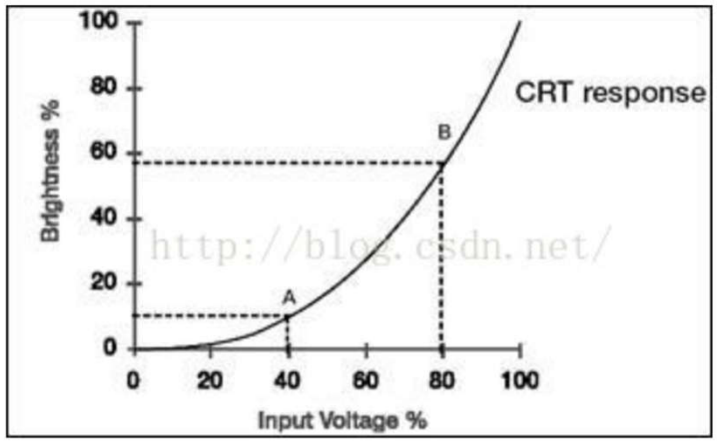


图2按图进行曲线补偿

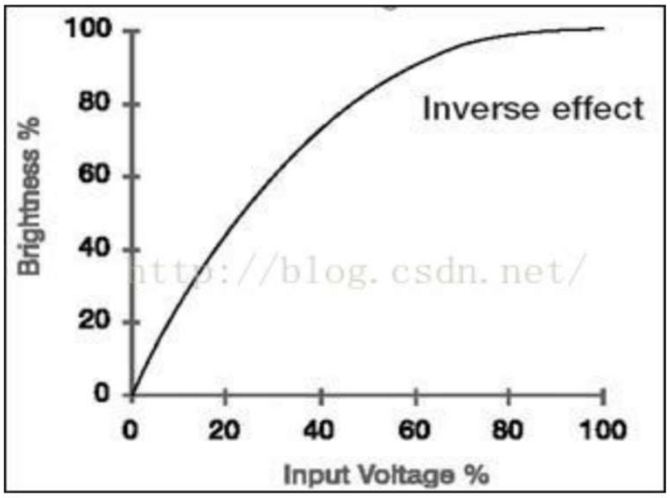
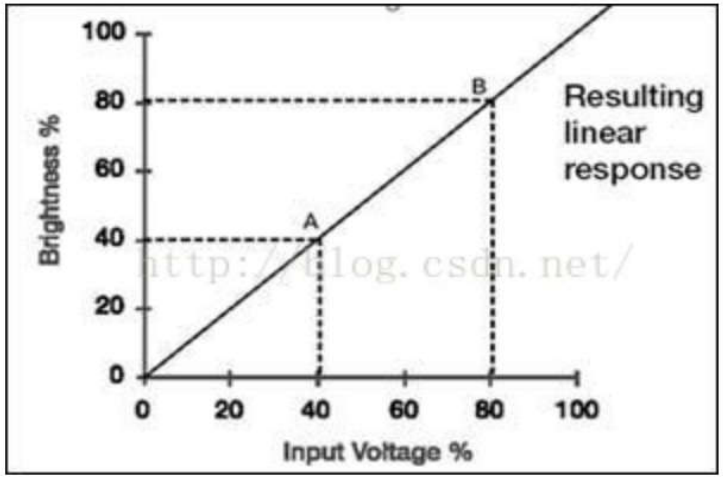


图3理想状态下的曲线



$$corrected = 255 * \left(\frac{uncorrected}{255} \right)^{gamma}$$

fig. 3: Gamma-corrected colors

Outline

- What is an image?
- Image Filtering
  - Types of image transformations
  - Point image processing
  - Linear shift-invariant image filtering
  - Convolution

Linear shift-invariant image filtering

- Replace each pixel by a *linear* combination of its neighbors (and possibly itself).
- The combination is determined by the filter's *kernel*.
- The same kernel is *shifted* to all pixel locations so that all pixels use the same linear combination of their neighbors.

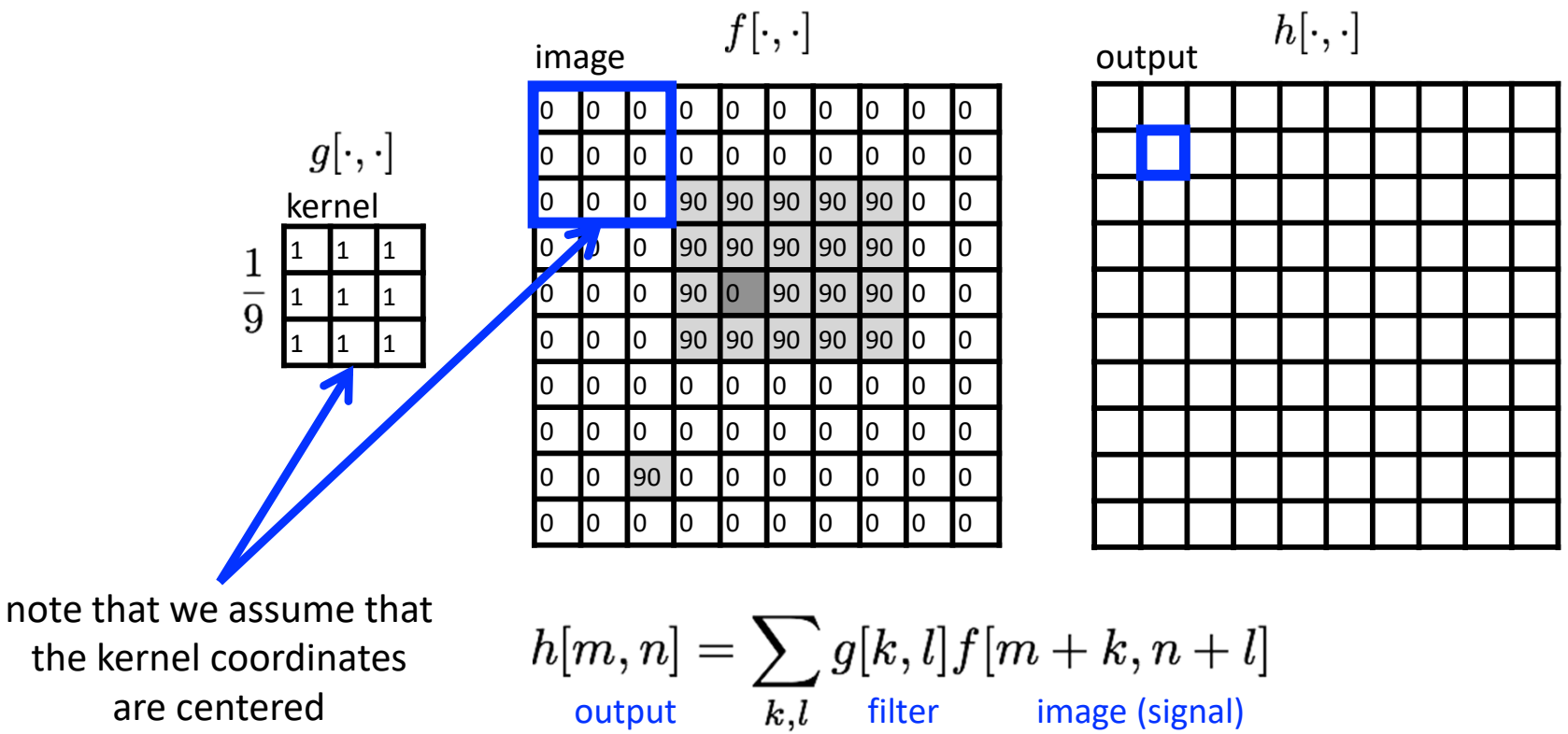
Example: the box filter

- also known as the 2D Rect filter
- also known as the square mean filter
- replaces pixel with local average
- has smoothing (blurring) effect

kernel $g[\cdot,\cdot] = \frac{1}{9}$

1	1	1
1	1	1
1	1	1

Let's run the box filter

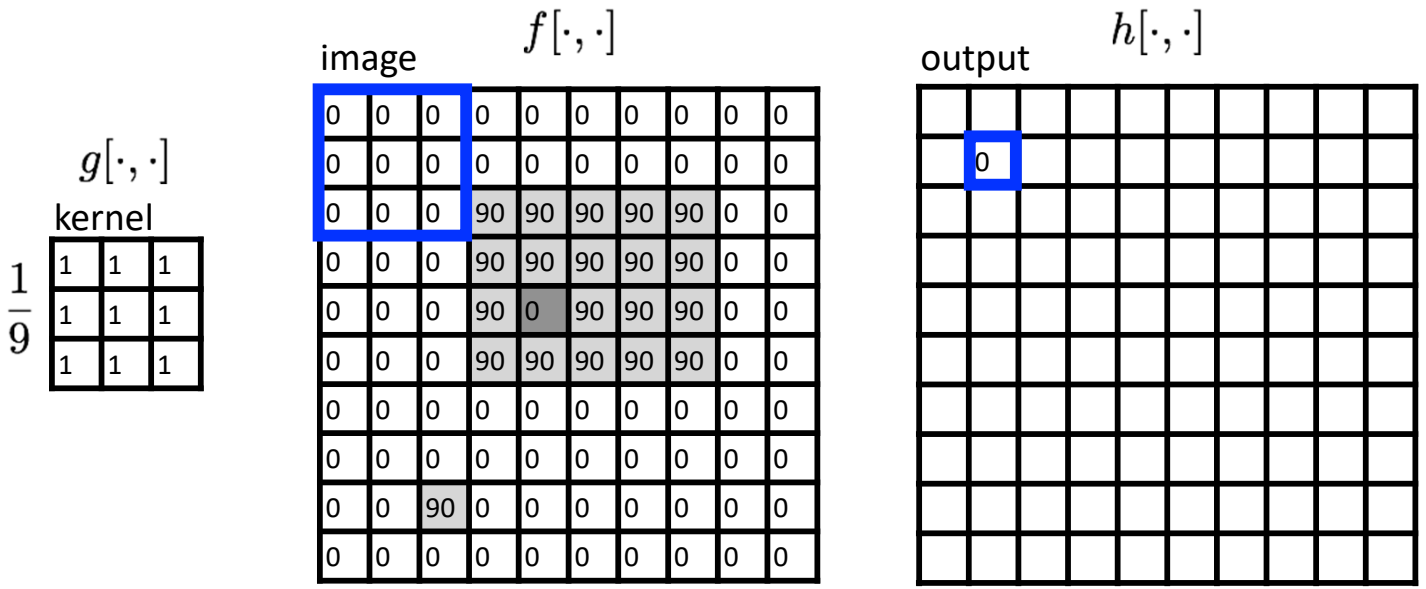


note that we assume that the kernel coordinates are centered

$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

output filter image (signal)

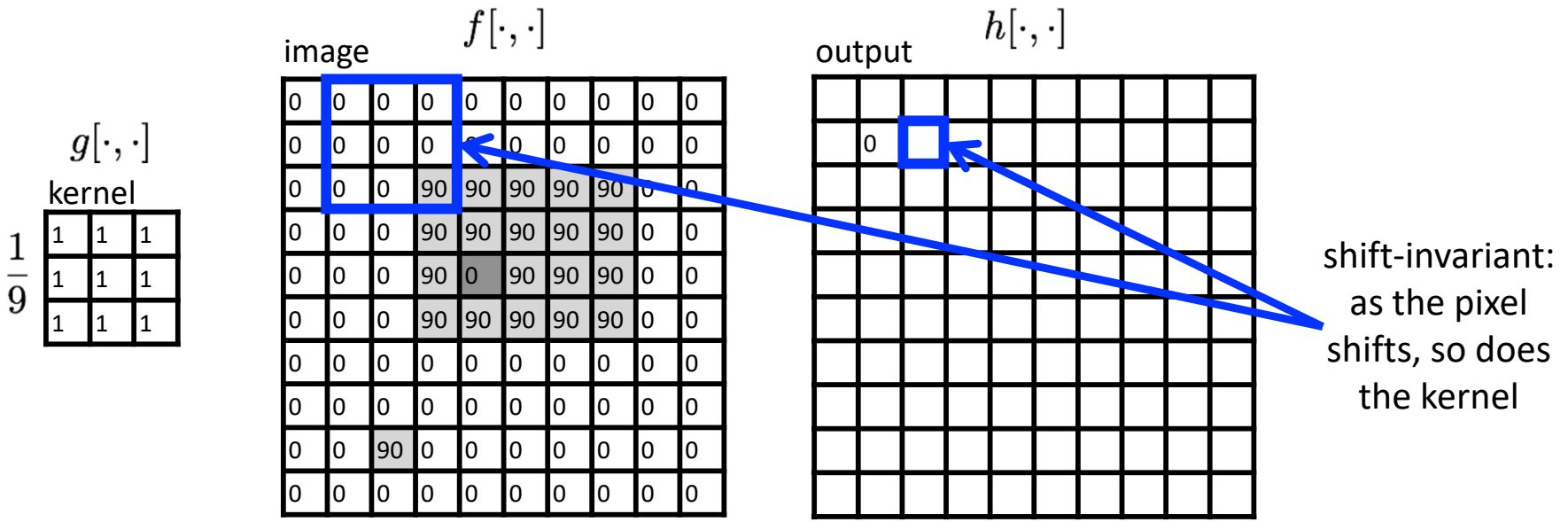
Let's run the box filter



$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

output filter image (signal)

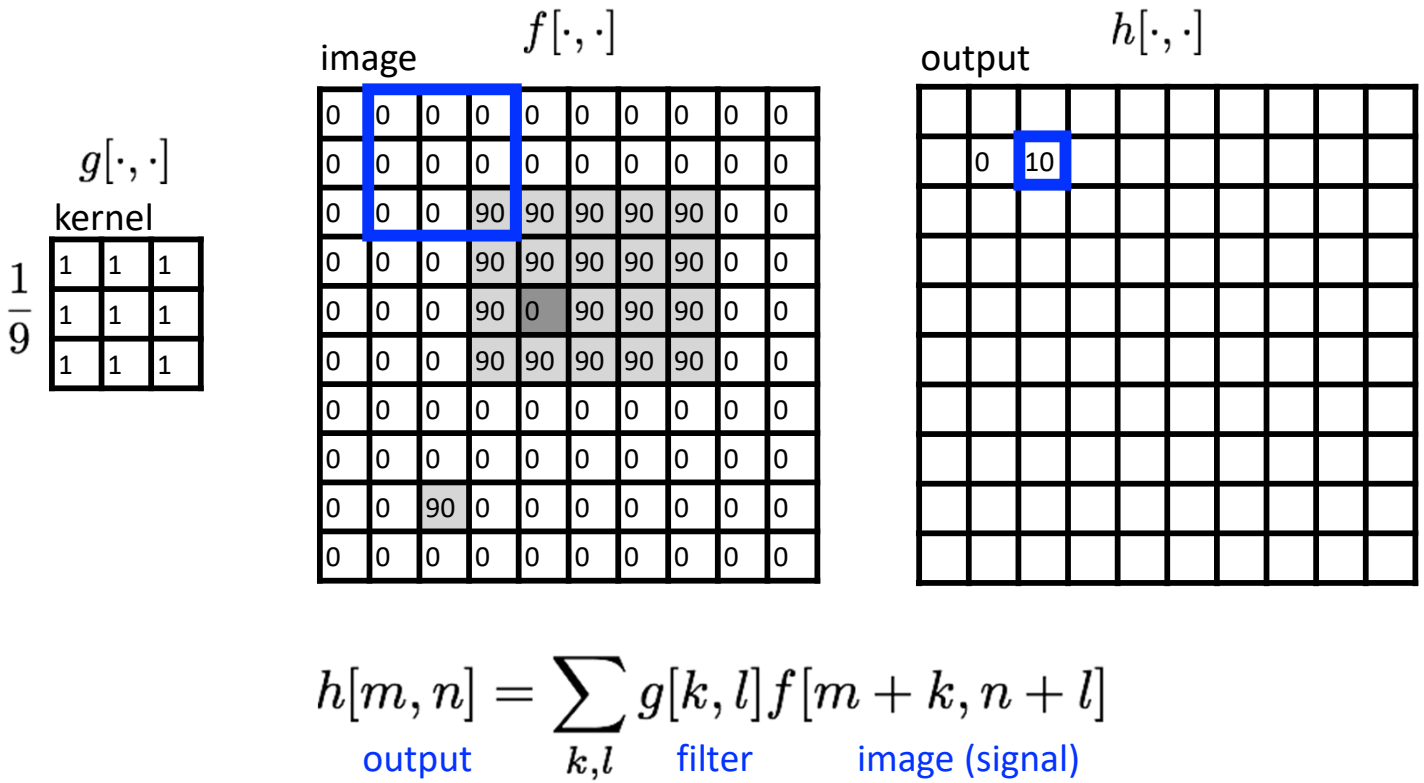
Let's run the box filter



$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

output filter image (signal)

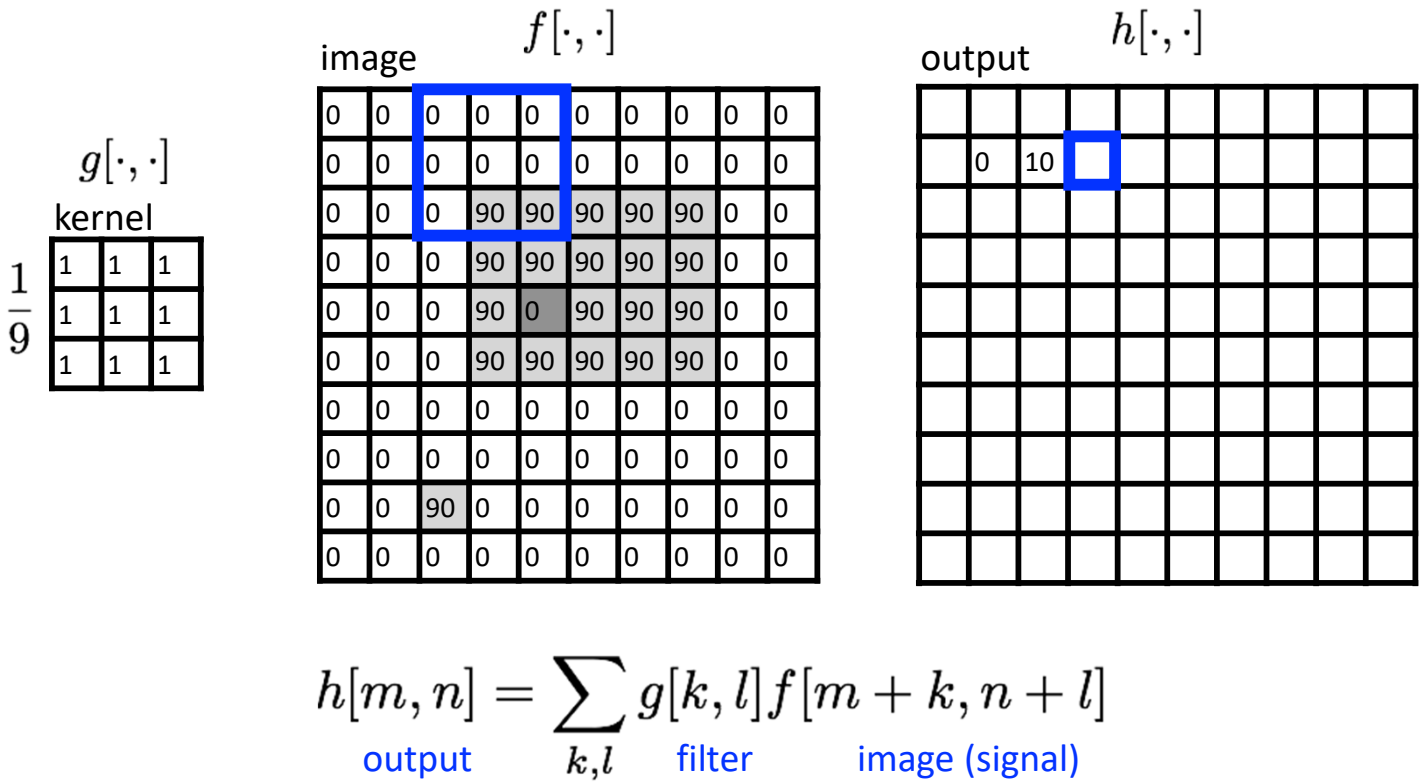
Let's run the box filter



$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

output filter image (signal)

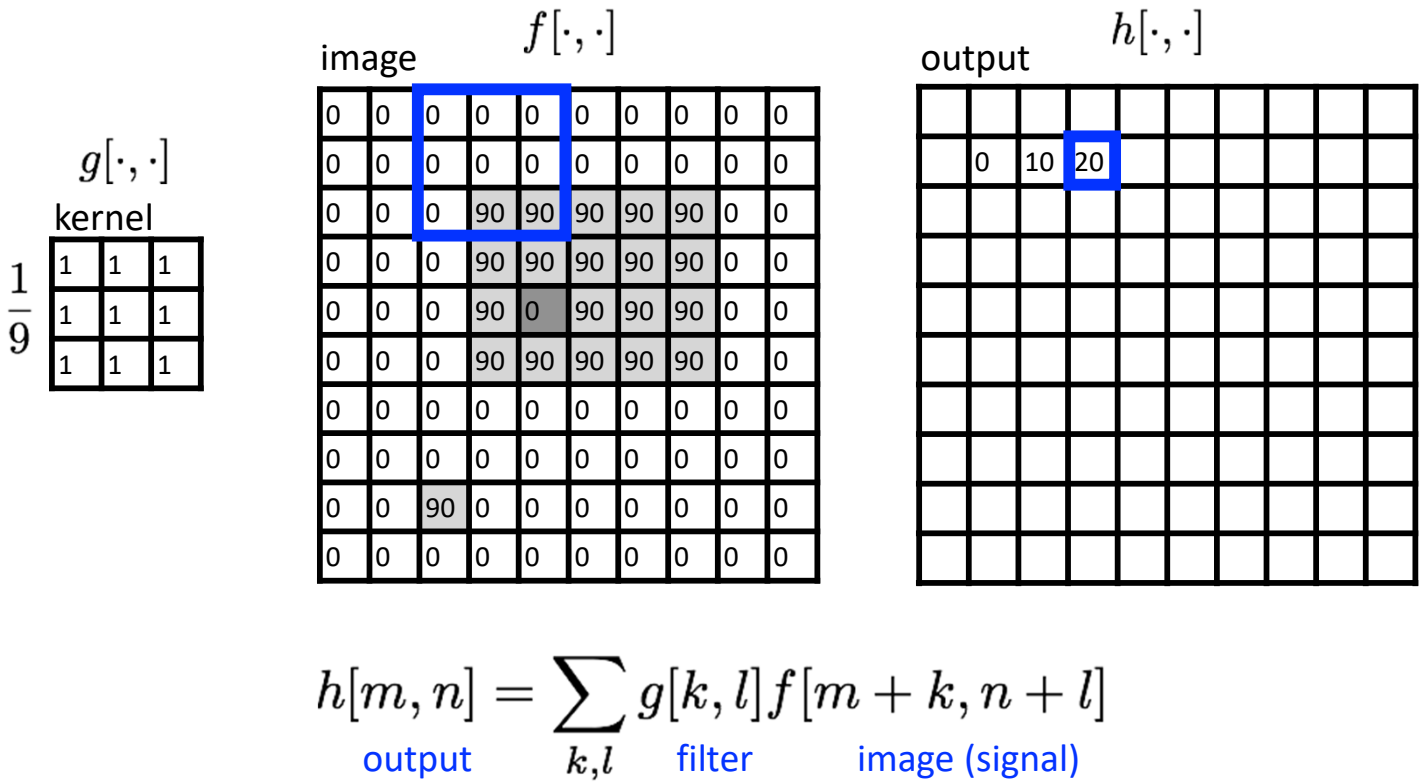
Let's run the box filter



$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

output filter image (signal)

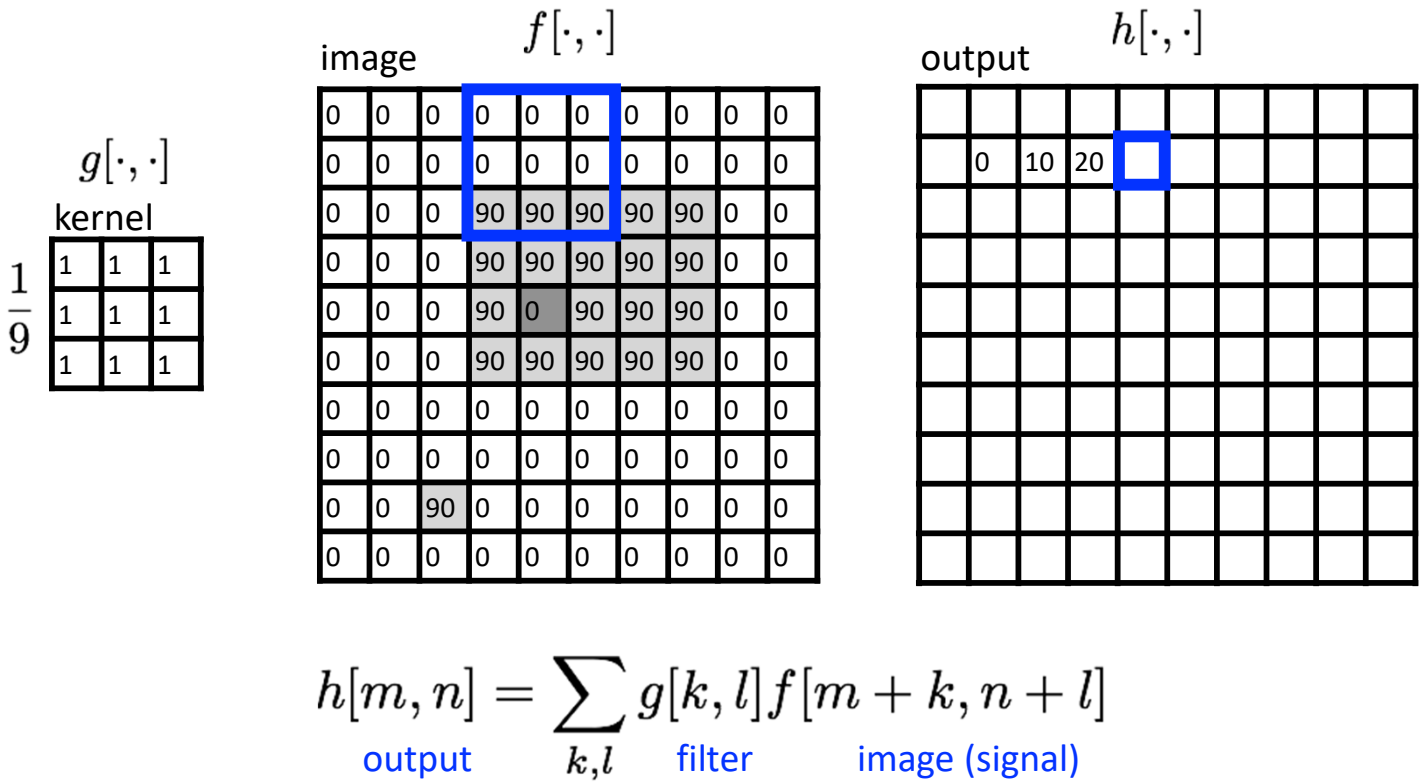
Let's run the box filter



$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

output filter image (signal)

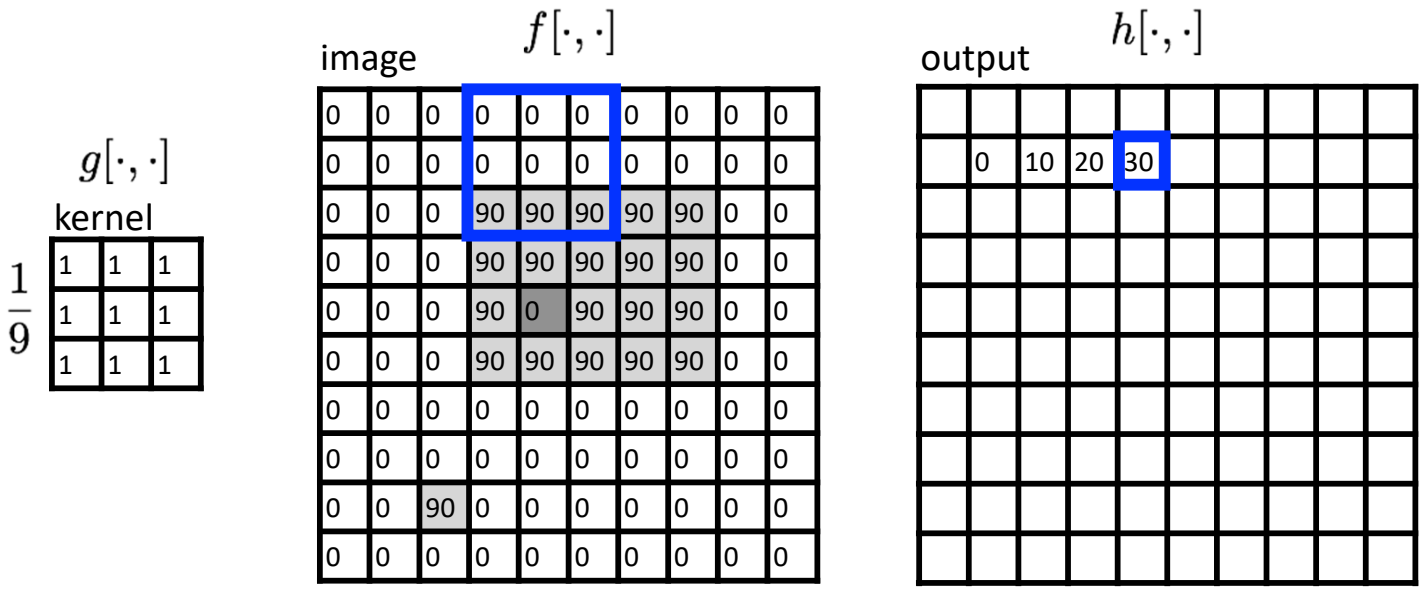
Let's run the box filter



$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

output filter image (signal)

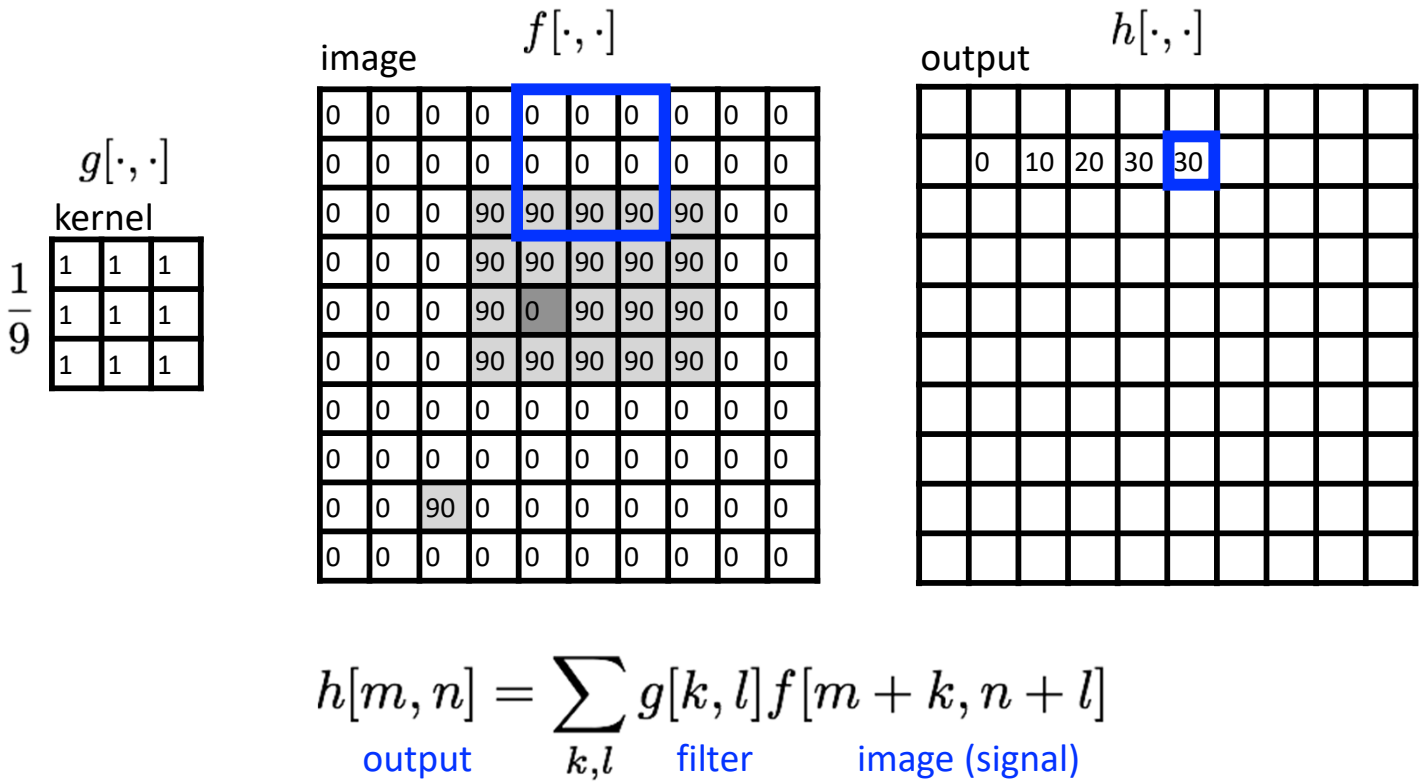
Let's run the box filter



$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

output filter image (signal)

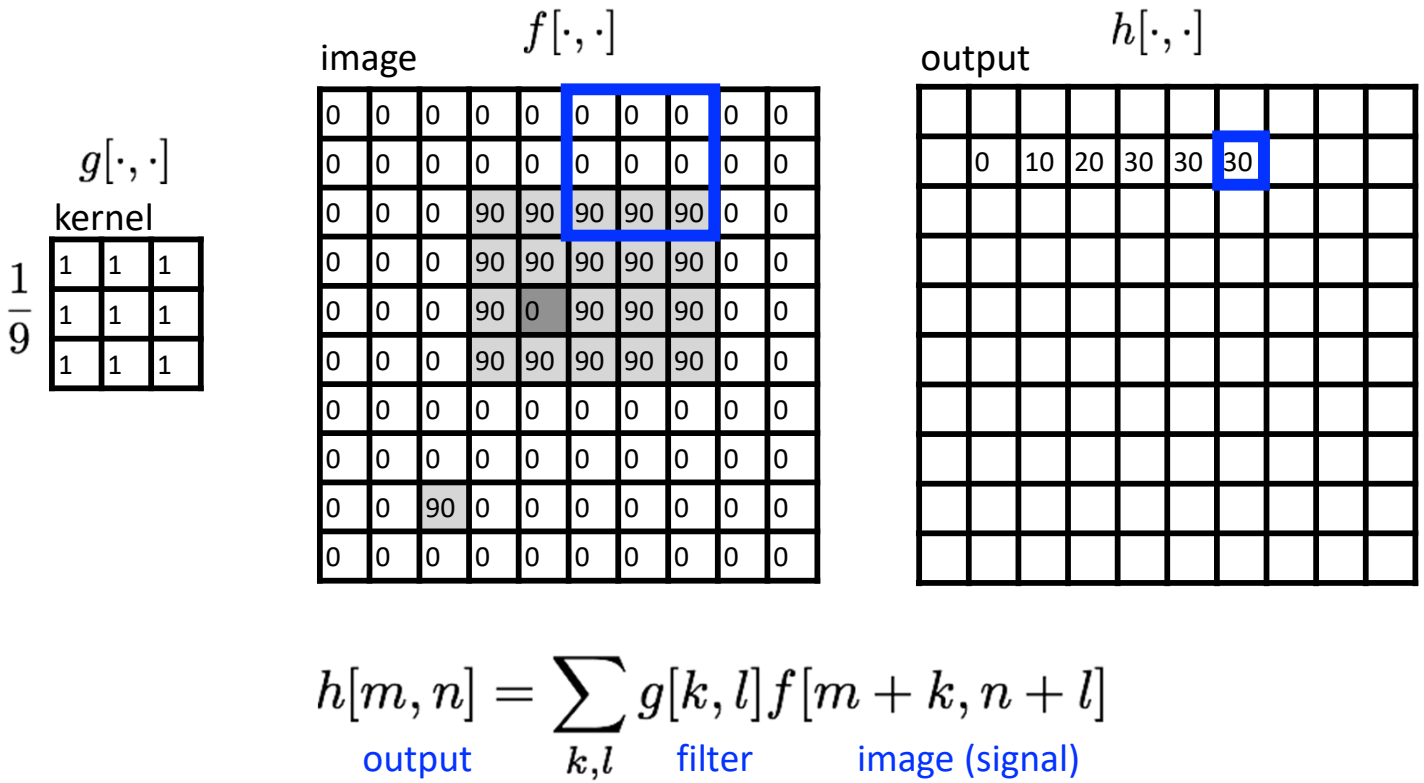
Let's run the box filter



$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

output filter image (signal)

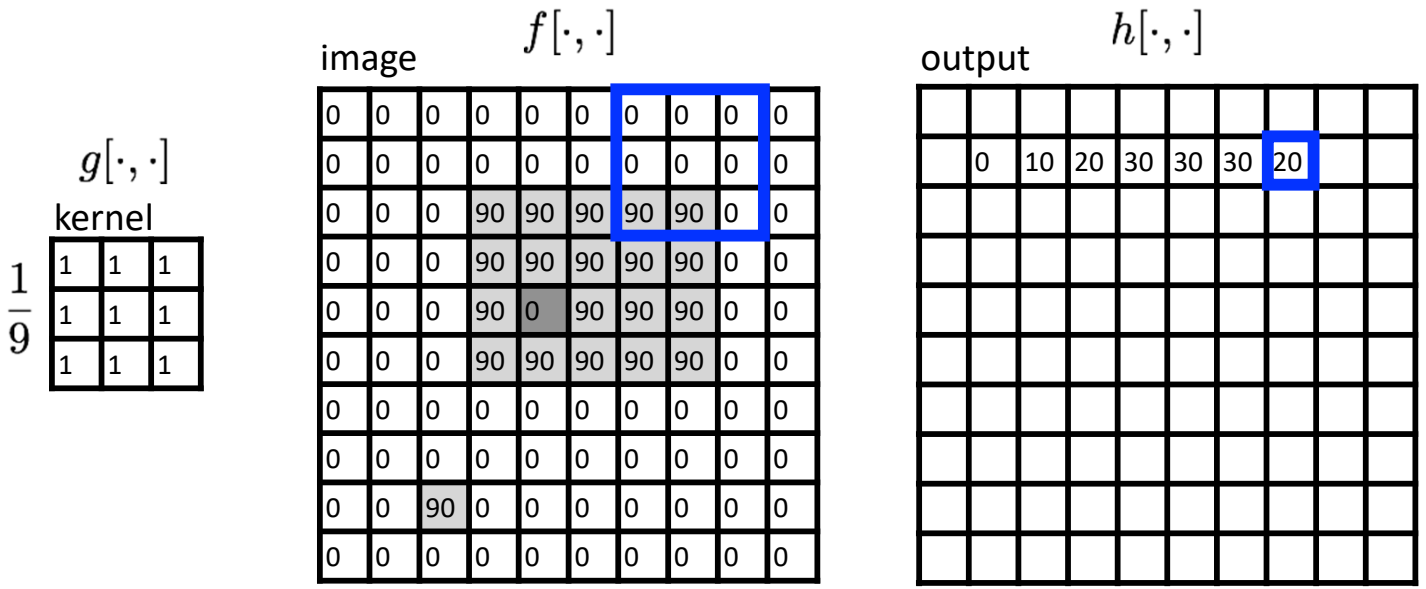
Let's run the box filter



$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

output filter image (signal)

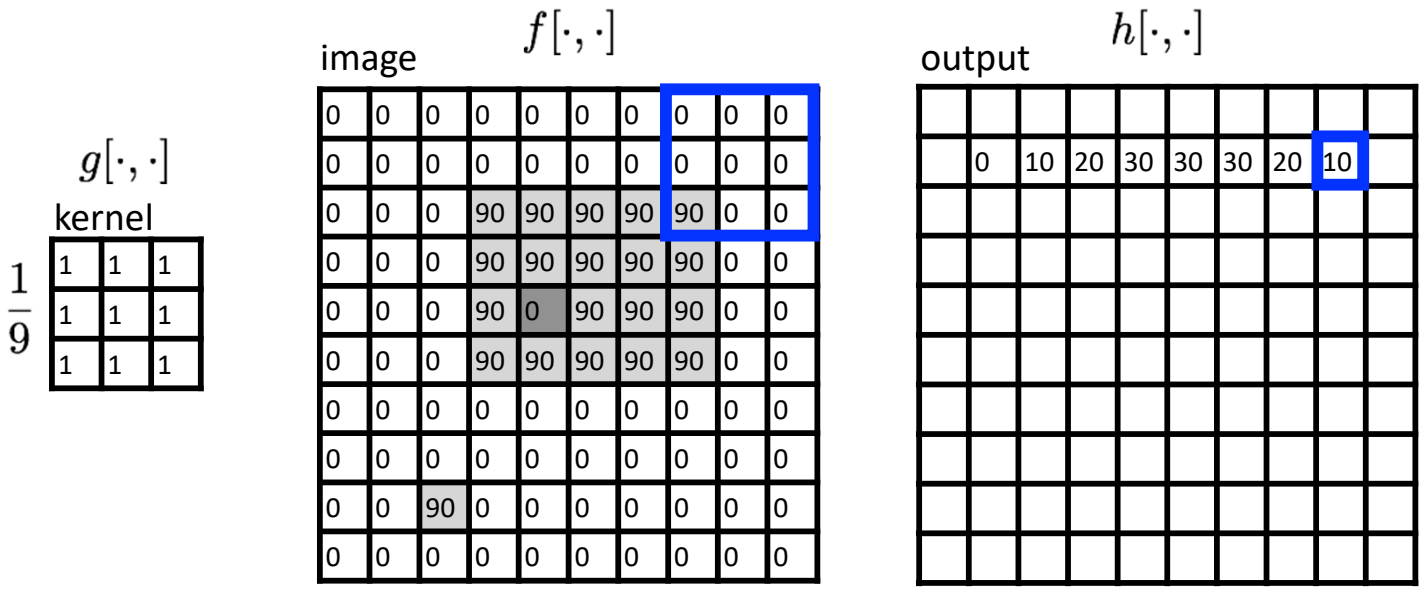
Let's run the box filter



$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

output filter image (signal)

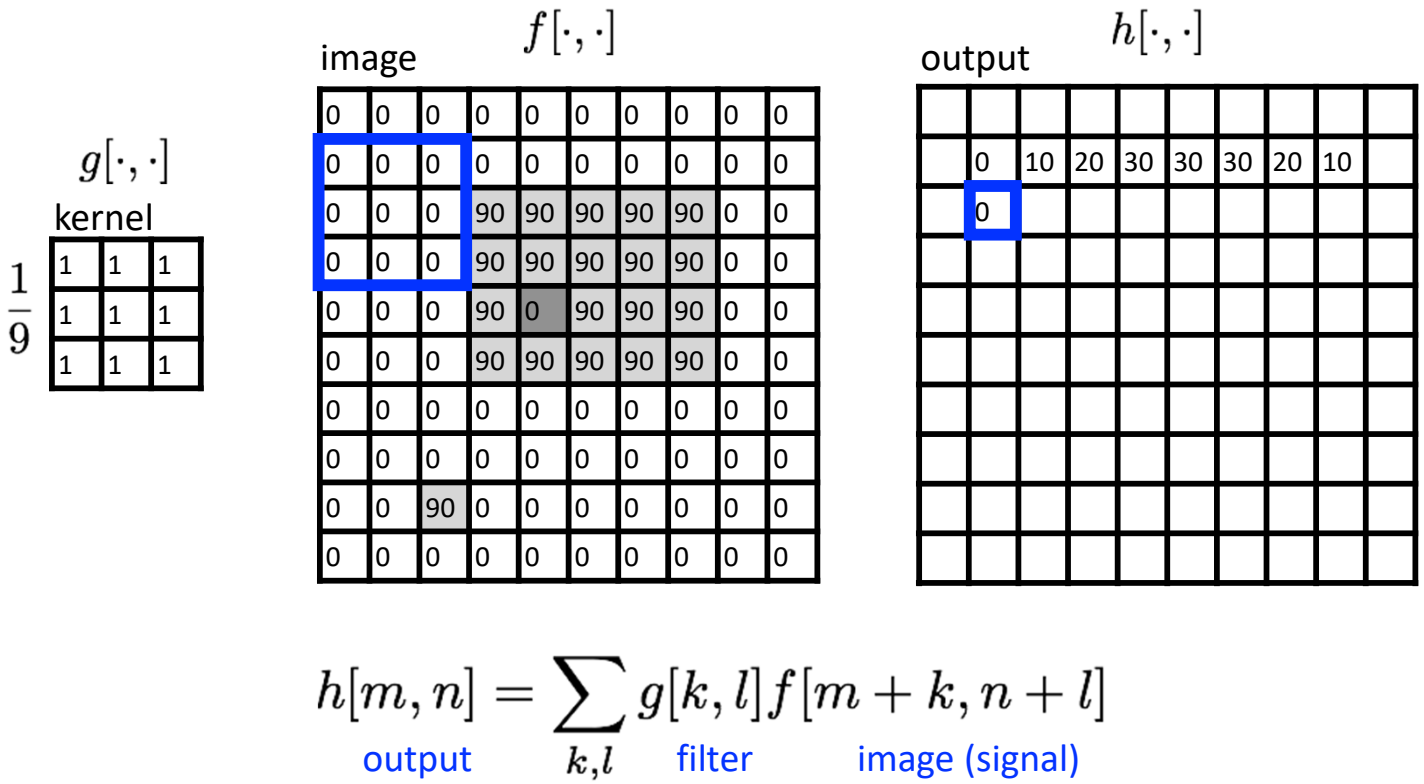
Let's run the box filter



$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

output filter image (signal)

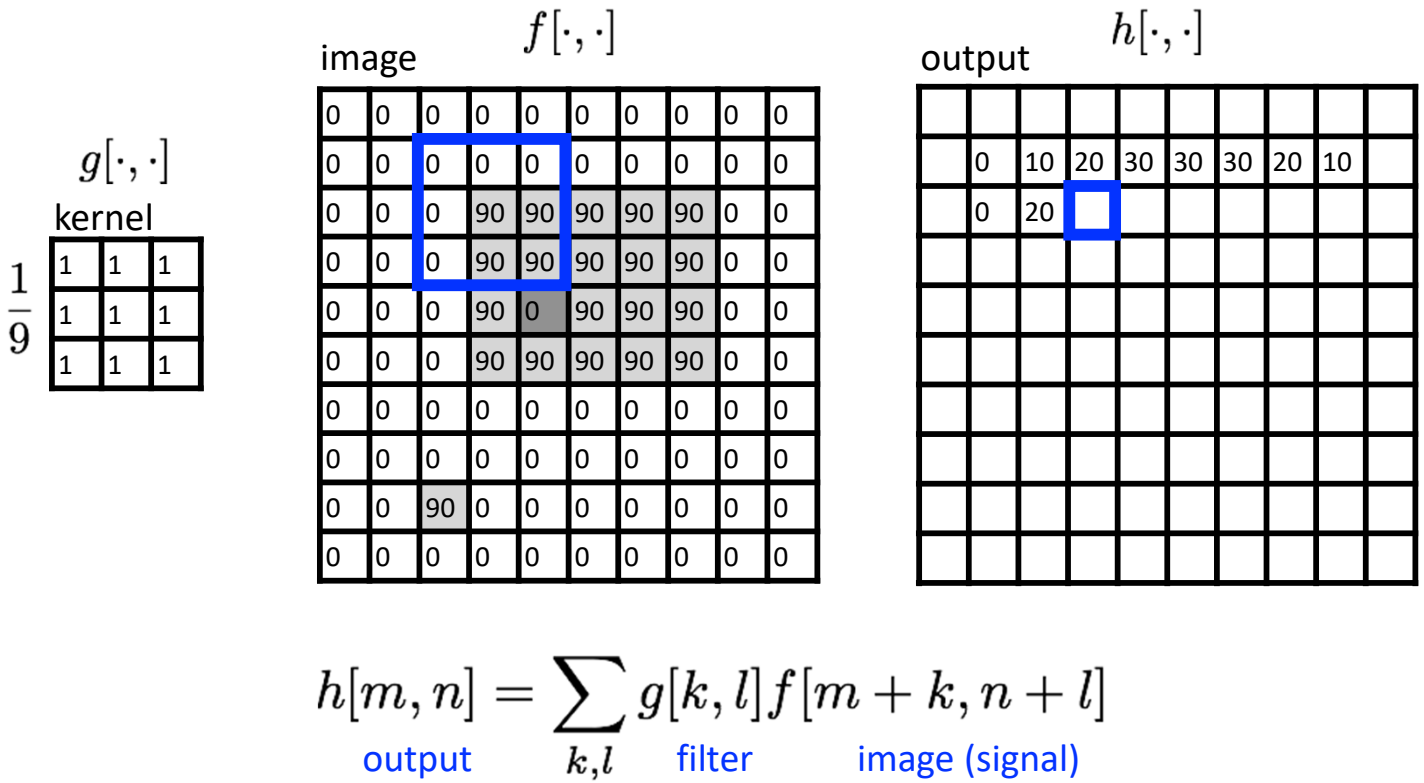
Let's run the box filter



$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

output filter image (signal)

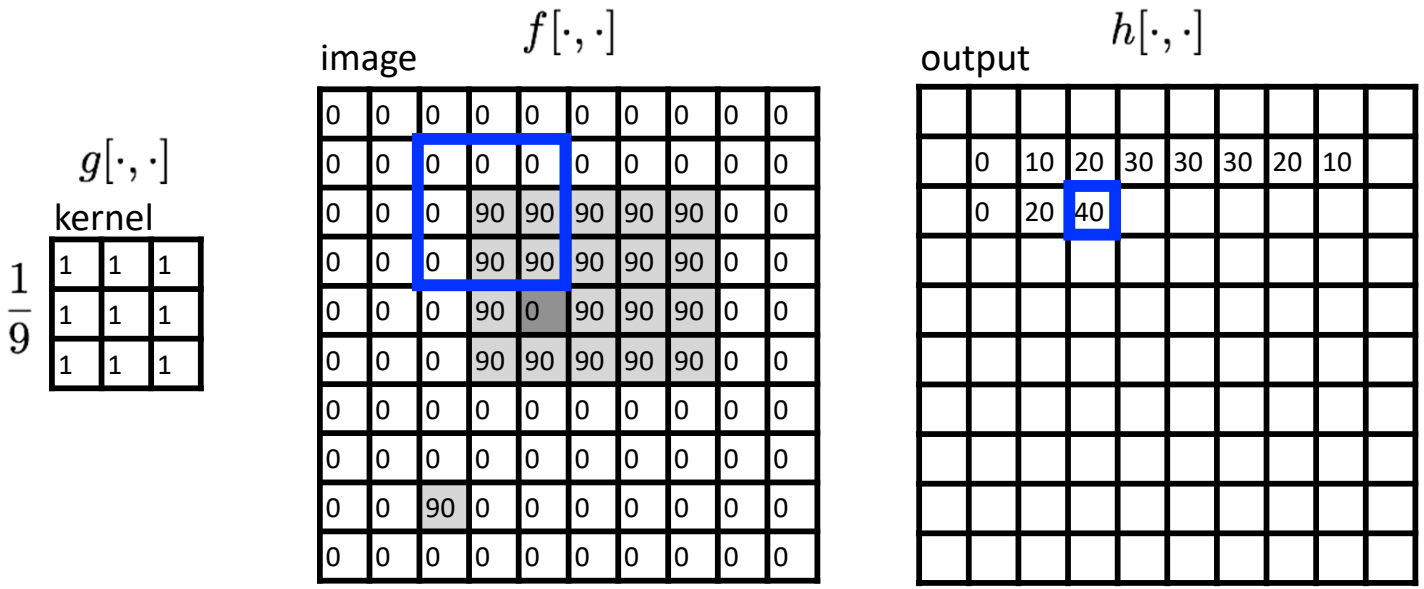
Let's run the box filter



$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

output filter image (signal)

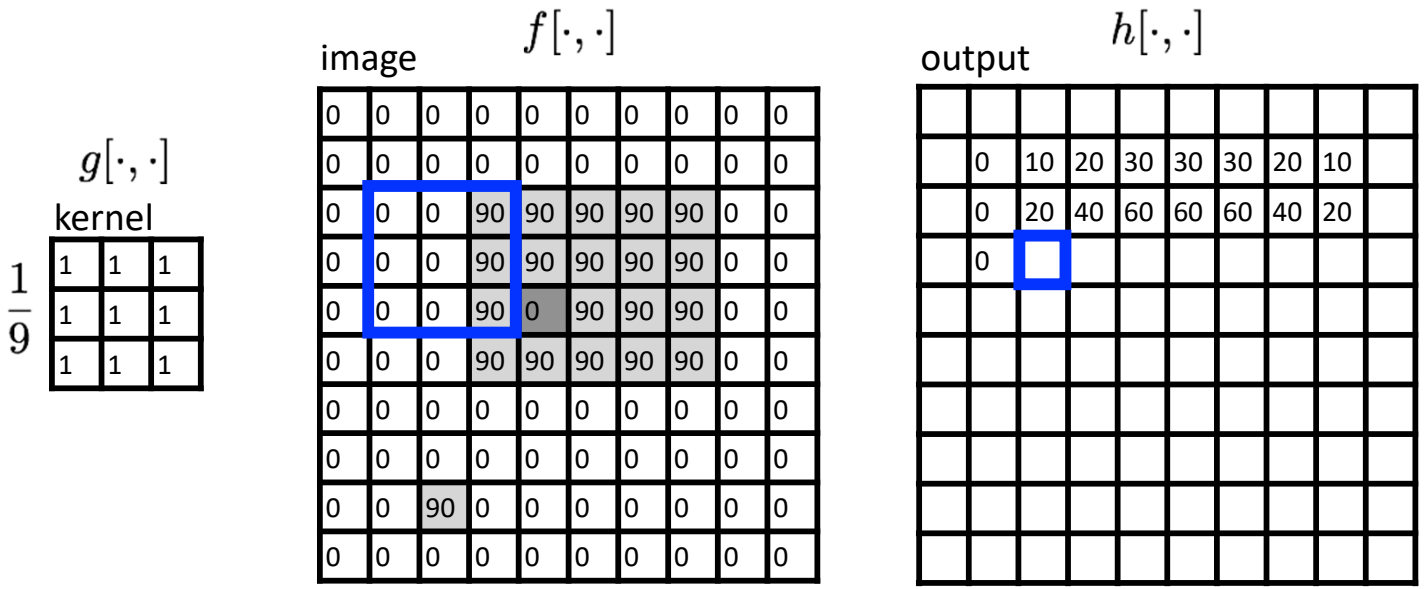
Let's run the box filter



$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

output filter image (signal)

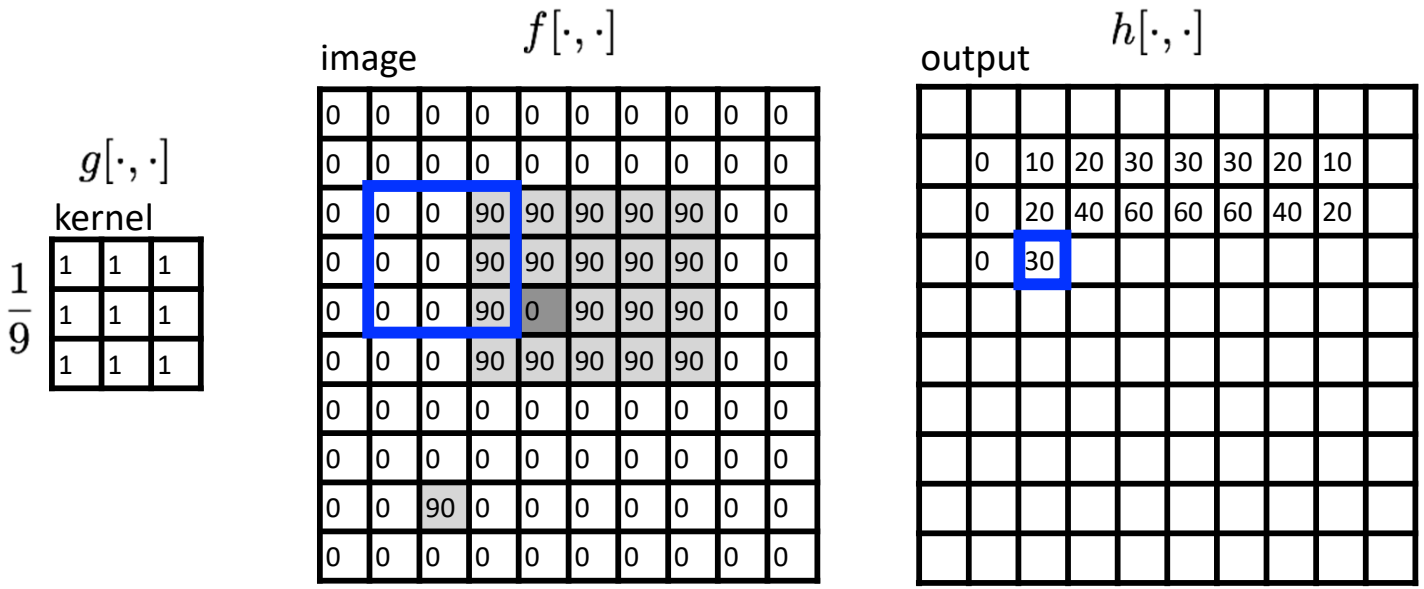
Let's run the box filter



$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

output filter image (signal)

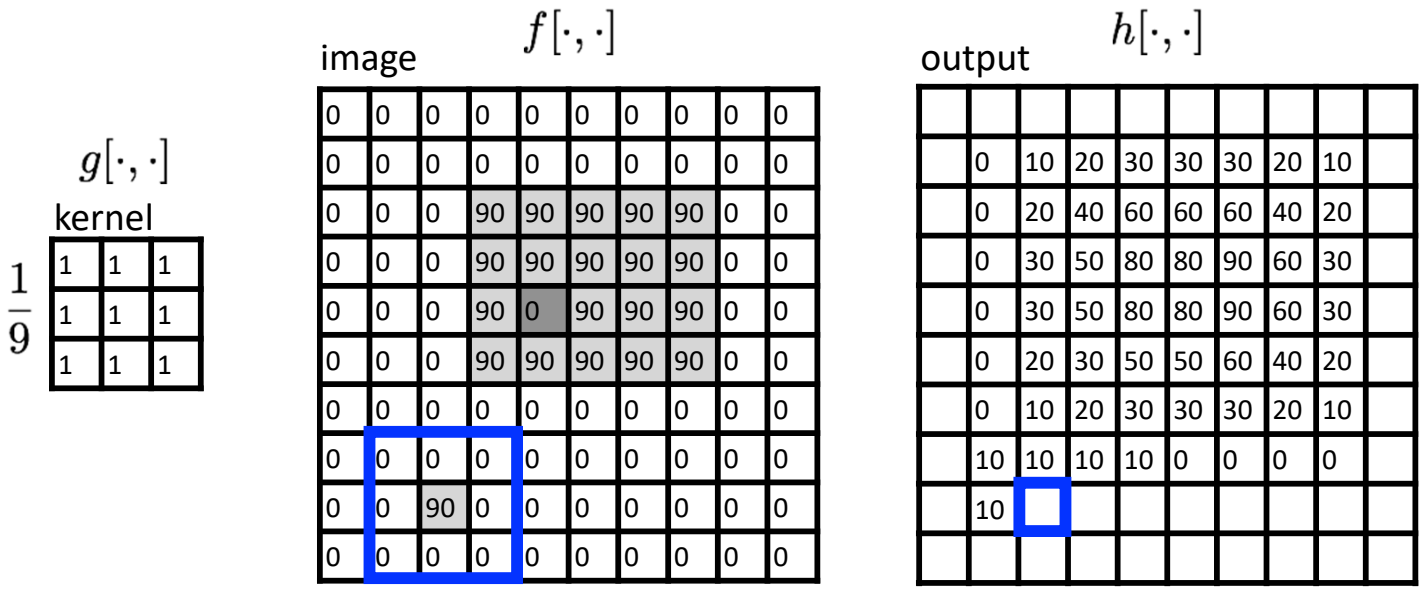
Let's run the box filter



$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

output filter image (signal)

Let's run the box filter



$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

output filter image (signal)

Let's run the box filter

$g[\cdot,\cdot]$

kernel

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

image $f[\cdot,\cdot]$

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

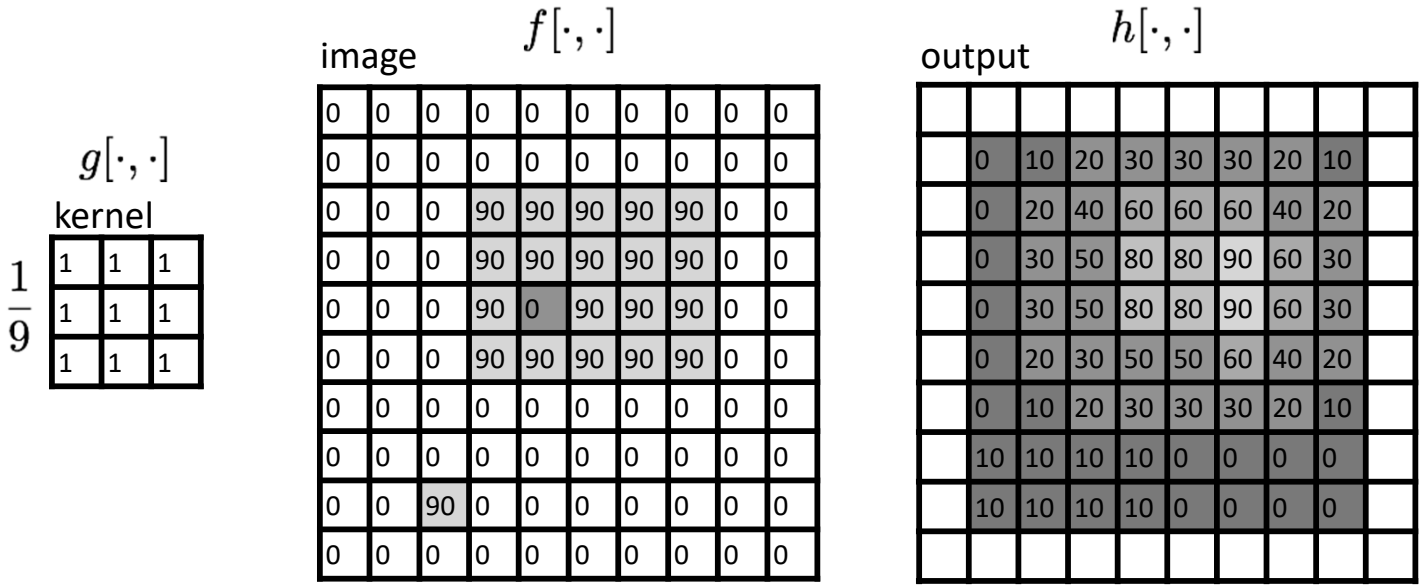
output $h[\cdot,\cdot]$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	0	10	20	30	30	30	20	10	
	10	10	10	10	0	0	0	0	
	10	10	10	10	0	0	0	0	

$$h[m,n] = \sum_{k,l} g[k,l]\, f[m+k, n+l]$$

output filter image (signal)

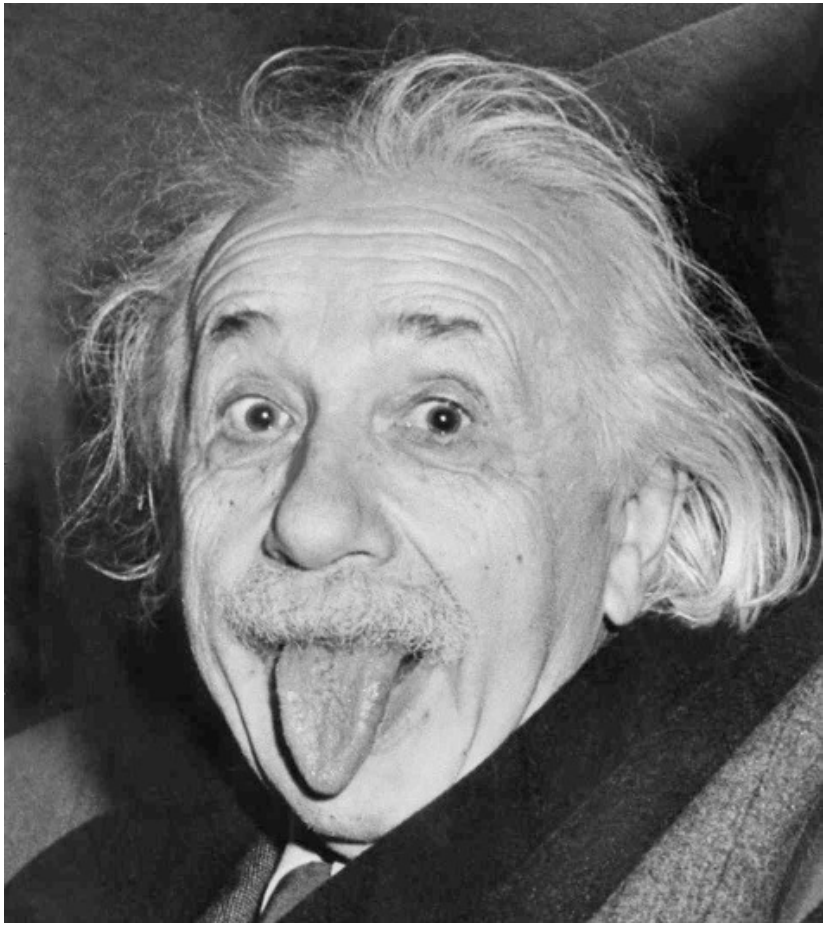
… and the result is

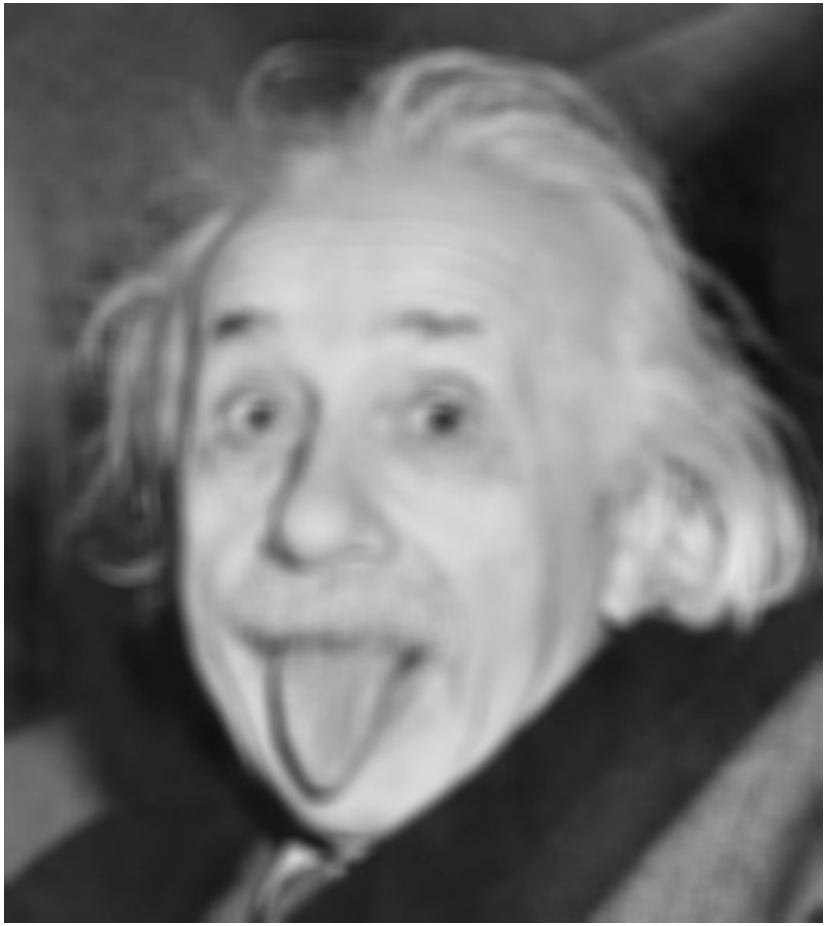


$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

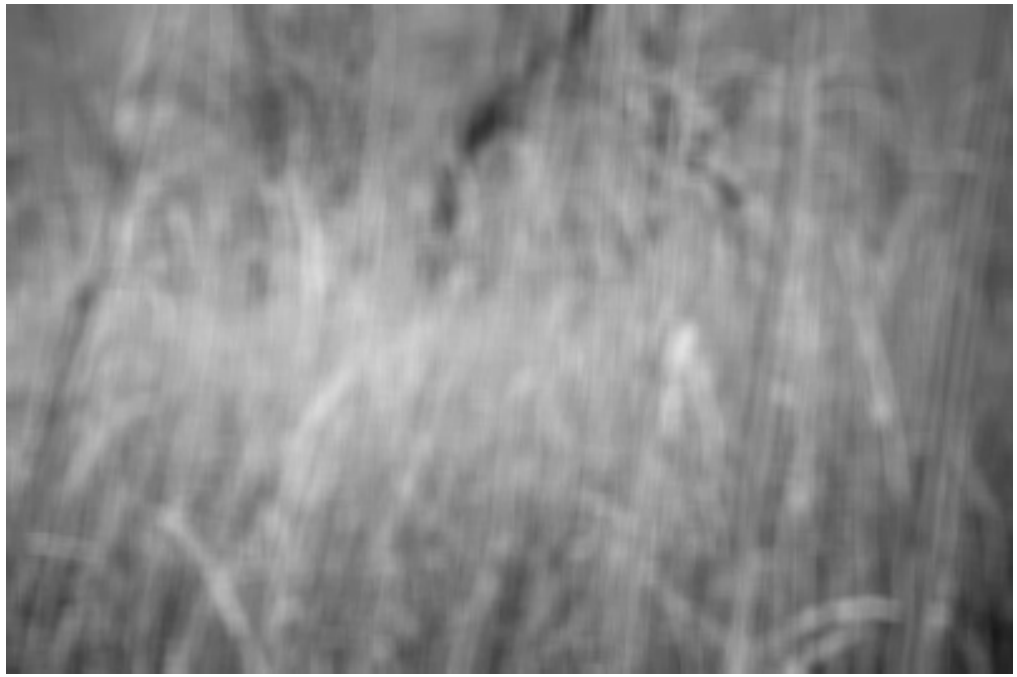
output filter image (signal)

Some more realistic examples





Some more realistic examples



Some more realistic examples

Outline

- What is an image?
- Image Filtering
  - Types of image transformations
  - Point image processing
  - Linear shift-invariant image filtering
  - Convolution

Convolution

- Convolution for 1D continuous signals

Definition of filtering as convolution:

notice the flip

$$(f * g)(x) = \int_{-\infty}^{\infty} f(y)g(x-y)dy$$

filtered signal

filter

input signal

Convolution

- Convolution for 1D continuous signals

Definition of filtering as convolution:

notice the flip

$$(f * g)(x) = \int_{-\infty}^{\infty} f(y)g(x-y)dy$$

filtered signal

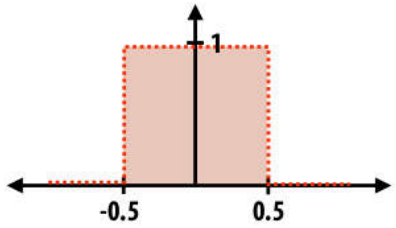
filter

input signal

Consider the box filter example:

1D continuous box filter

$$f(x) = \begin{cases} 1 & |x| \leq 0.5 \\ 0 & otherwise \end{cases}$$



filtering output is a blurred version of g

$$(f * g)(x) = \int_{-0.5}^{0.5} g(x-y)dy$$

Convolution

- Convolution for 2D discrete signals

Definition of filtering as convolution:

notice the flip

$$(f * g)(x, y) = \sum_{i,j=-\infty}^{\infty} f(i, j) I(x - i, y - j)$$

filtered image

filter

input image

Convolution

- Convolution for 2D discrete signals

Definition of filtering as convolution:

notice the flip

$$(f * g)(x, y) = \sum_{i,j=-\infty}^{\infty} f(i, j) I(x - i, y - j)$$

filtered image — filter — input image

If the filter $f(i, j)$ is non-zero only within $-1 \leq i, j \leq 1$, then

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

The kernel we saw earlier is the 3x3 matrix representation of $f(i, j)$.

Separable filters

A 2D filter is separable if it can be written as the product of a “column” and a “row”.

example:
box filter

1	1	1
1	1	1
1	1	1

=

1
1
1

column

*

1	1	1

row

What is the rank of this filter matrix?

Separable filters

A 2D filter is separable if it can be written as the product of a “column” and a “row”.

example:
box filter

1	1	1
1	1	1
1	1	1

=

1
1
1

column

*

1	1	1

row

Why is this important?

Separable filters

A 2D filter is separable if it can be written as the product of a “column” and a “row”.

example:
box filter

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

column

row

2D convolution with a separable filter is equivalent to two 1D convolutions (with the “column” and “row” filters).

If the image has M x M pixels and the filter kernel has size N x N:

- What is the cost of convolution with a non-separable filter?

Separable filters

A 2D filter is separable if it can be written as the product of a “column” and a “row”.

example:
box filter

1	1	1
1	1	1
1	1	1

=

1
1
1

column

*

1	1	1

row

2D convolution with a separable filter is equivalent to two 1D convolutions (with the “column” and “row” filters).

If the image has M x M pixels and the filter kernel has size N x N:

- What is the cost of convolution with a non-separable filter? → $M^2 \times N^2$
- What is the cost of convolution with a separable filter? → $2 \times N \times M^2$

A few more filters

original

3x3 box filter

do you see
any problems
in this image?

The Gaussian filter

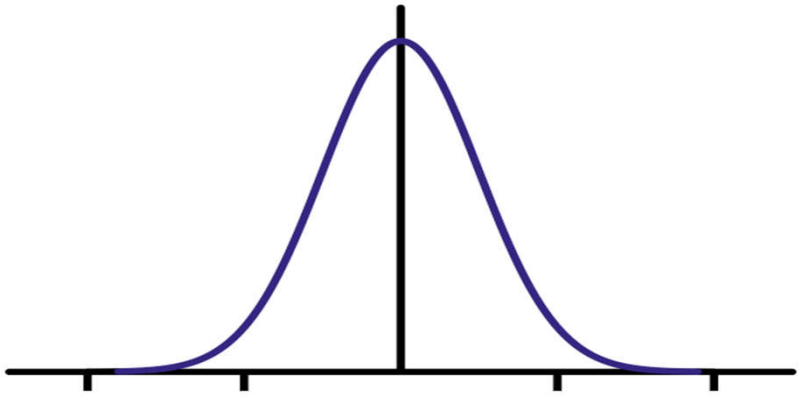
- named (like many other things) after Carl Friedrich Gauss
- kernel values sampled from the 2D Gaussian function:

$$f(i,j) = \frac{1}{2\pi\sigma^2} e^{-\frac{i^2+j^2}{2\sigma^2}}$$

- weight falls off with distance from the center pixel
- theoretically infinite, in practice truncated to some maximum distance

Any heuristics for selecting where to truncate?

- usually at 2~3σ



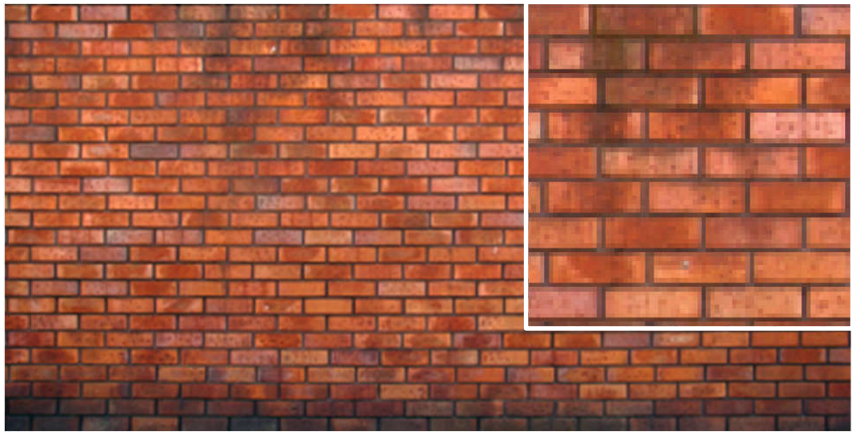
Is this a separable filter? Yes!

kernel $\frac{1}{16}$

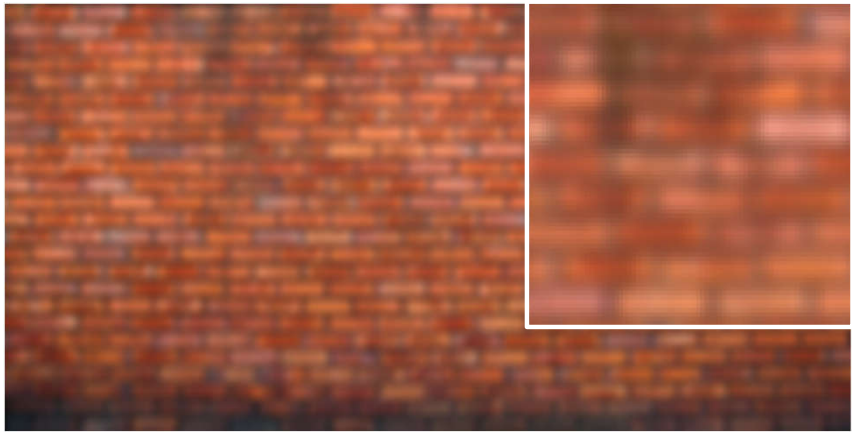
1	2	1
2	4	2
1	2	1

Gaussian filtering example

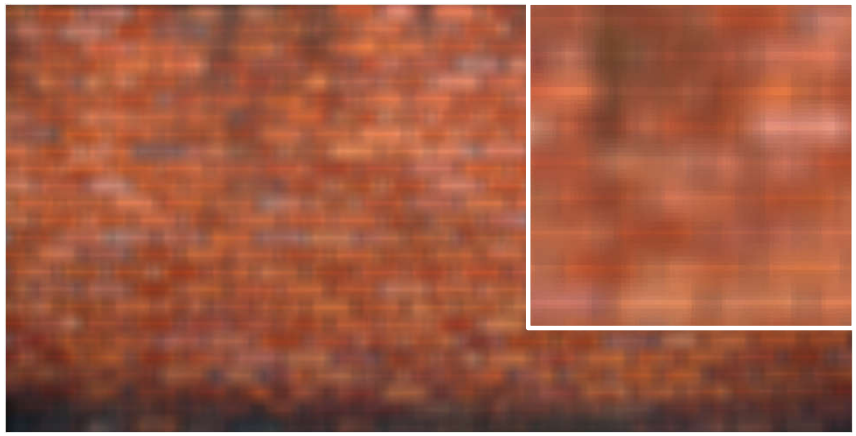
Gaussian vs box filtering



original



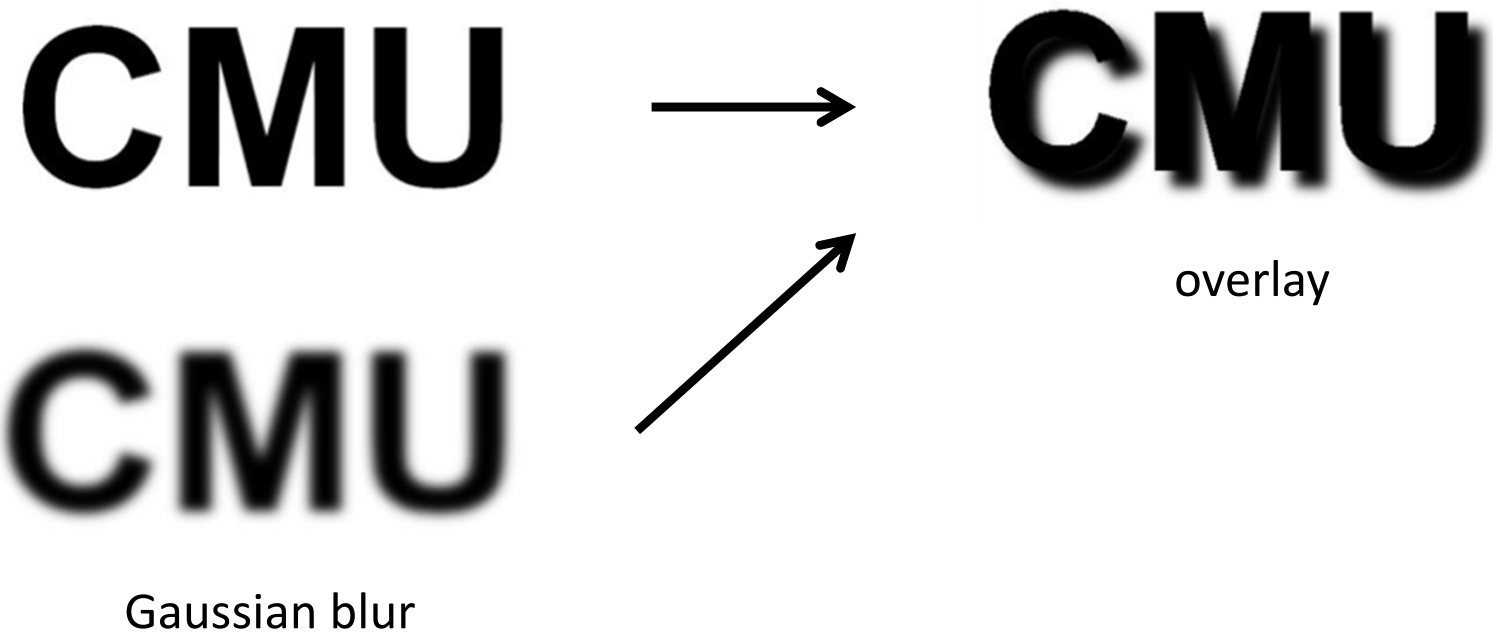
7x7 Gaussian



7x7 box

Which blur do you like better?

How would you create a soft shadow effect?



Other filters

input

filter

0	0	0
0	1	0
0	0	0

output

unchanged

input

filter

0	0	0
0	0	1
0	0	0

output

shift to left by one

Other filters

input

filter

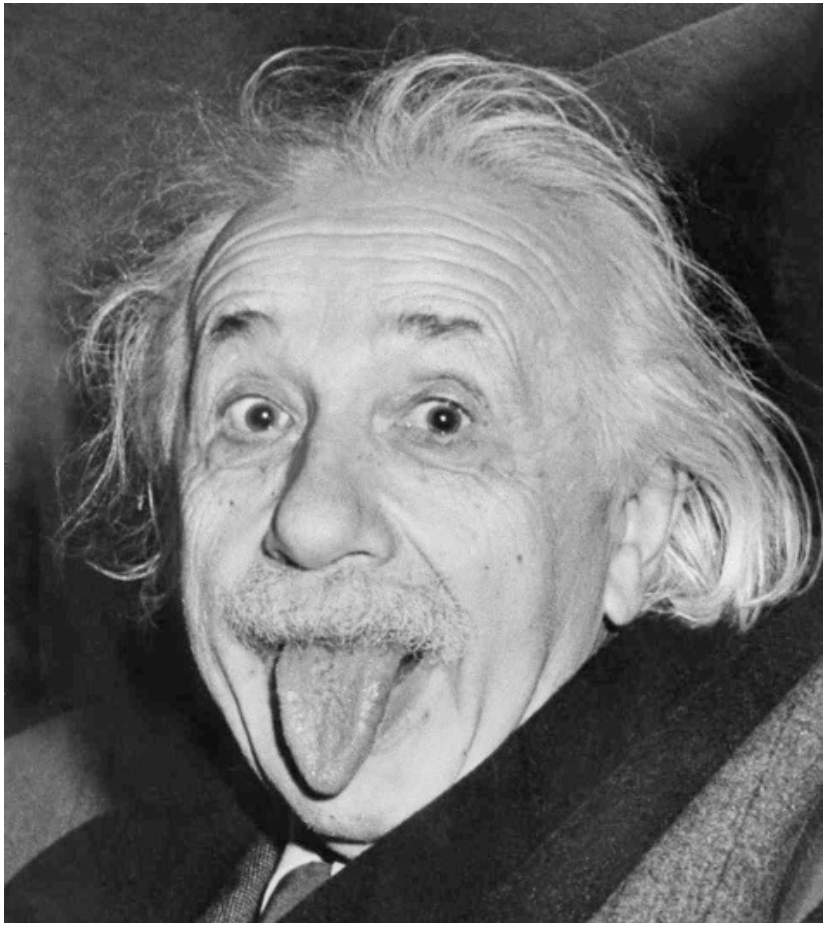
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

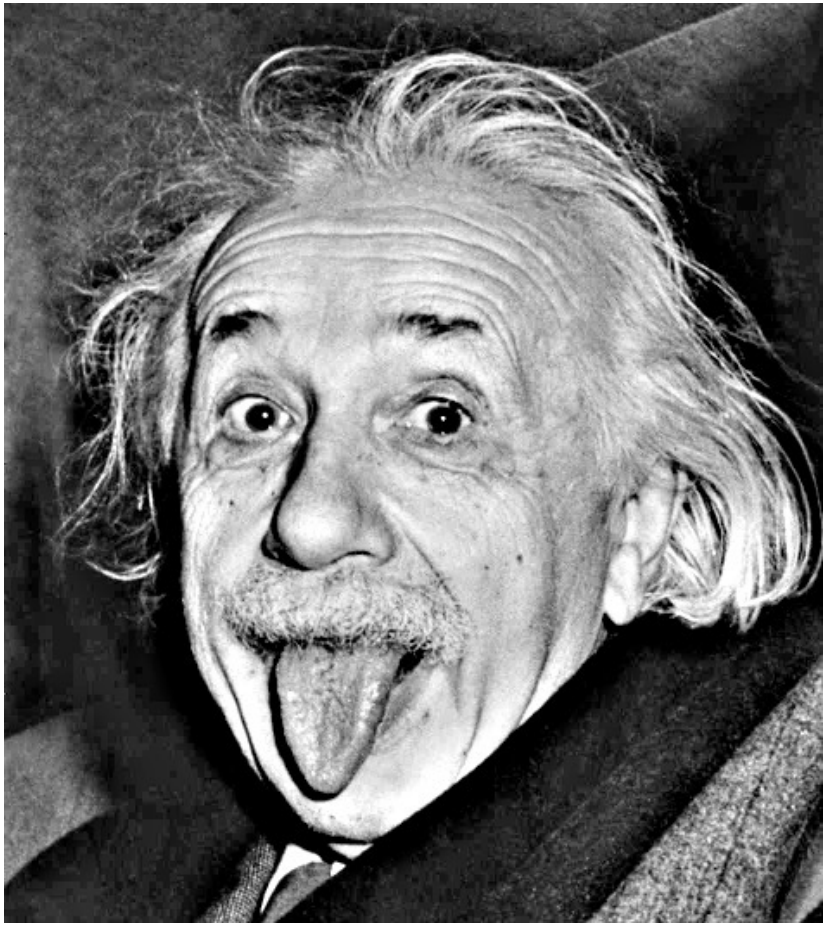
output

sharpening

- do nothing for flat areas
- stress intensity peaks

Sharpening examples





Sharpening examples

Sharpening examples

Sharpening examples

do you see any problems in this image?

Do not overdo it with sharpening

original

sharpened

oversharpened

What is wrong in this image?

References

- Basic reading:
  - Szeliski textbook, Section 3.2