Linear filtering

Roadmap

Machine Vision Technology							
Semantic information				Metric 3D information			
Pixels	Segments	Images	Videos	Camera		Multi-view Geometry	
Convolutions Edges & Fitting Local features Texture	Segmentation Clustering	Recognition Detection	Motion Tracking	Camera Model	Camera Calibration	Epipolar Geometry	SFM

Types of Images

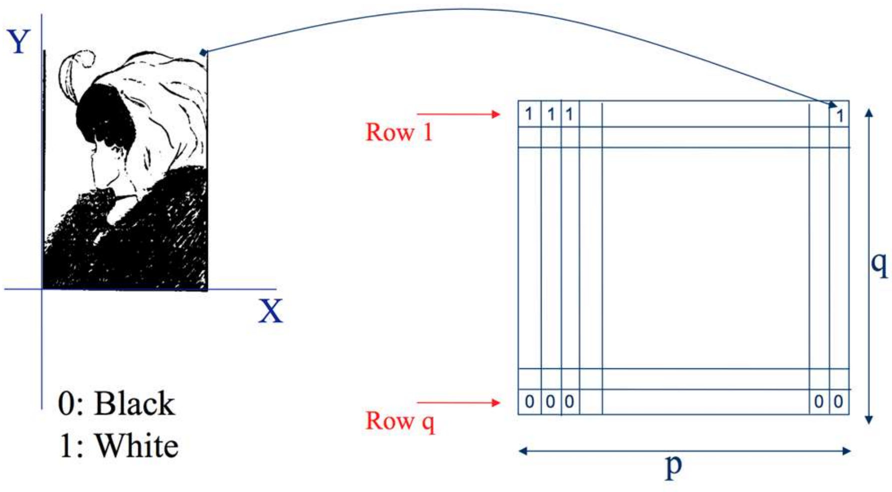
Binary

Gray Scale

Color

Source：Ulas Bagci

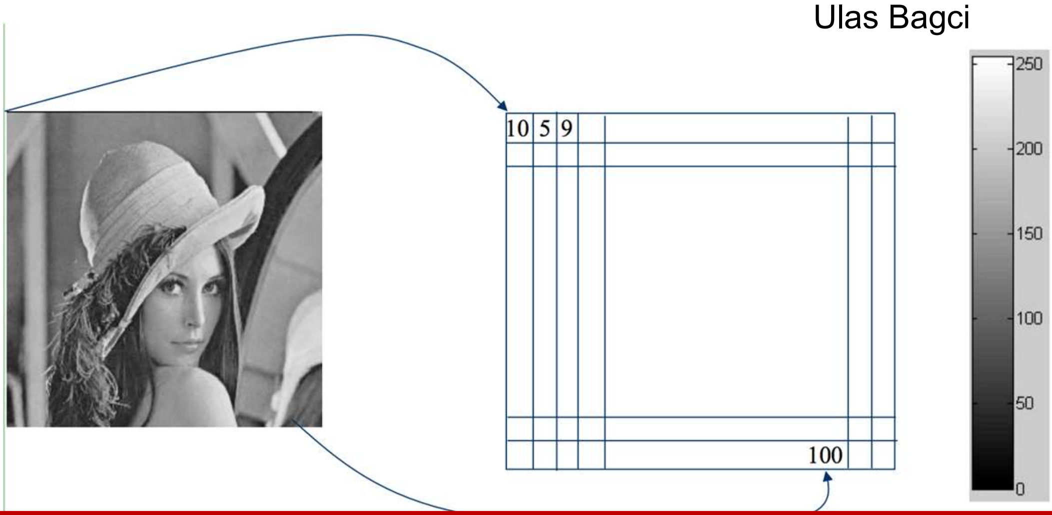
Binary image representation



Source：Ulas Bagci

Grayscale image representation

Slide credit:
Ulas Bagci



Source: Ulas Bagci

Color Image - one channel

Source: Ulas Bagci

Color image representation

Source： Ulas Bagci

Motivation: Image denoising

- How can we reduce noise in a photograph?

Source： S. Lazebnik

Moving average

- Let’s replace each pixel with a *weighted* average of its neighborhood
- The weights are called the *filter kernel*
- What are the weights for the average of a 3x3 neighborhood?

$$\frac{1}{9}\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

“box filter”

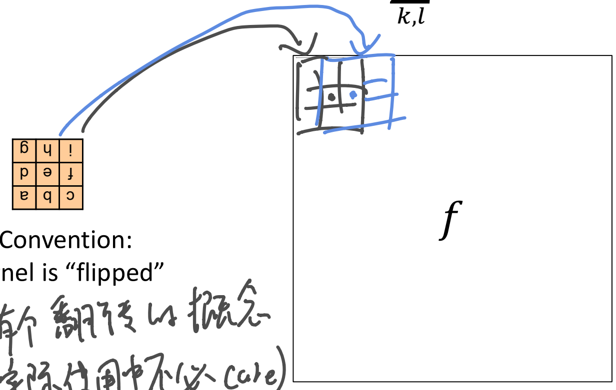
Source： S. Lazebnik

Defining convolution

- Let f be the image and g be the kernel. The output of convolving f with g is denoted $f * g$.

$$(f * g)[m,n] = \sum_{k,l} f[m-k, n-l]\, g[k,l]$$

卷积运算



Convention:
kernel is “flipped”

这里有个翻转的概念
(实际使用中不必care)

$$\int_0^t f(x)\, g(t-x)\, dx$$

m, n 是核g在f中的坐标

k, l 是核g内的坐标

Key properties

- **Linearity:** filter($f_1 + f_2$) = filter(f_1) + filter(f_2)
- **Shift invariance:** same behavior regardless of pixel location: filter(shift(f)) = shift(filter(f))
- Theoretical result: any linear shift-invariant operator can be represented as a convolution

Source: S. Lazebnik

Properties in more detail

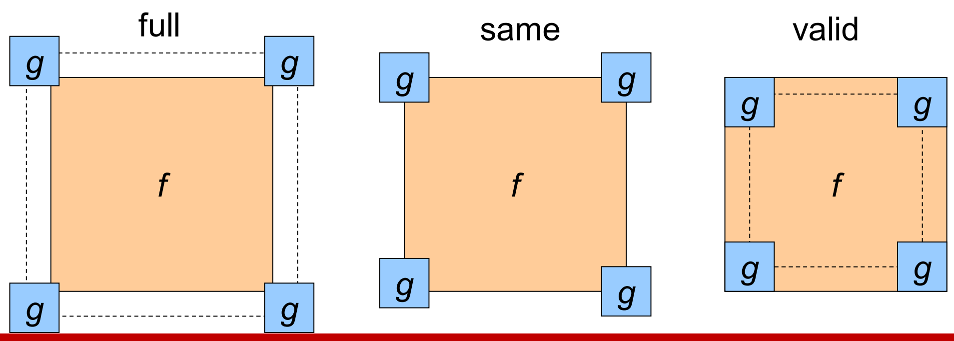
- Commutative: $a * b = b * a$
  - Conceptually no difference between filter and signal
- Associative: $a * (b * c) = (a * b) * c$
  - Often apply several filters one after another: $(((a * b_1) * b_2) * b_3)$
  - This is equivalent to applying one filter: $a * (b_1 * b_2 * b_3)$
- Distributes over addition: $a * (b + c) = (a * b) + (a * c)$
- Scalars factor out: $ka * b = a * kb = k (a * b)$
- Identity: unit impulse $e = [\ldots, 0, 0, 1, 0, 0, \ldots]$, $a * e = a$

Source: S. Lazebnik

Annoying details

What is the size of the output?

- MATLAB: filter2(g, f, *shape*)
  - *shape* = 'full': output size is sum of sizes of f and g
  - *shape* = 'same': output size is same as f
  - *shape* = 'valid': output size is difference of sizes of f and g



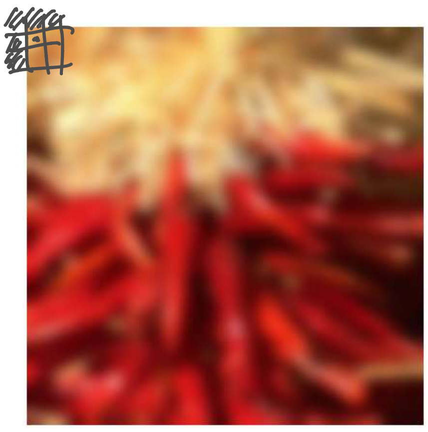
Source：S. Lazebnik

Annoying details

What about near the edge?

避免外侧卷积不到，在外侧填充一圈像素

- the filter window falls off the edge of the image
- need to extrapolate
- methods:
  - clip filter (black)
  - wrap around
  - copy edge
  - reflect across edge



Source: S. Marschner

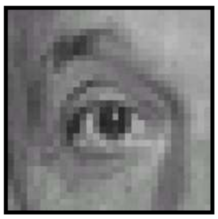
Annoying details

What about near the edge?

- the filter window falls off the edge of the image
- need to extrapolate
- methods (MATLAB):
  - clip filter (black): imfilter(f, g, 0)
  - wrap around: imfilter(f, g, 'circular')
  - copy edge: imfilter(f, g, 'replicate')
  - reflect across edge: imfilter(f, g, 'symmetric')

Source: S. Marschner

Practice with linear filters



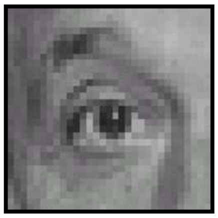
Original

0	0	0
0	1	0
0	0	0

?

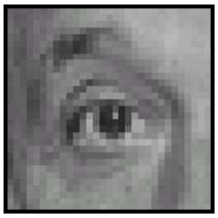
Source: D. Lowe

Practice with linear filters



Original

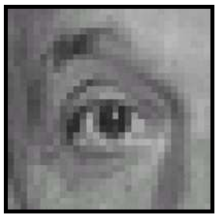
0	0	0
0	1	0
0	0	0



Filtered
(no change)

Source: D. Lowe

Practice with linear filters



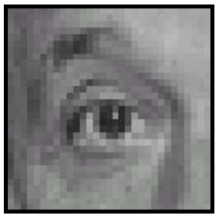
Original

0	0	0
0	0	1
0	0	0

?

Source: D. Lowe

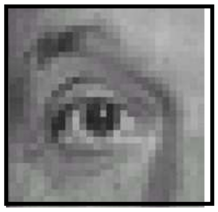
Practice with linear filters



Original

0	0	0
0	0	1
0	0	0

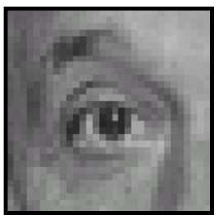
左移



Shifted *left*
By 1 pixel

Source: D. Lowe

Practice with linear filters



Original

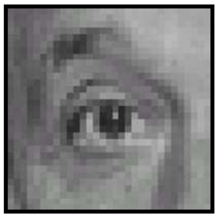
$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

?

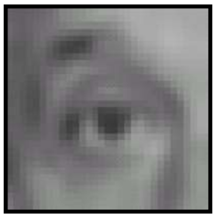
Source: D. Lowe

Practice with linear filters



Original

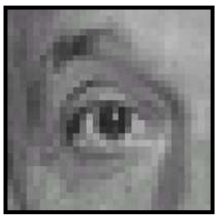
$$\frac{1}{9}\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline\end{array}$$



Blur (with a box filter)

Source: D. Lowe

Practice with linear filters



Original

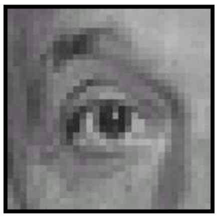
$$\begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & 2 & 0 \\ \hline 0 & 0 & 0 \\ \hline\end{array} - \frac{1}{9}\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline\end{array} \qquad ?$$

(Note that filter sums to 1)

Source: D. Lowe

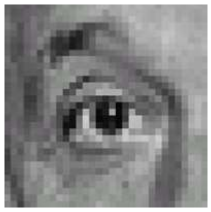
Practice with linear filters

锐化



Original

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



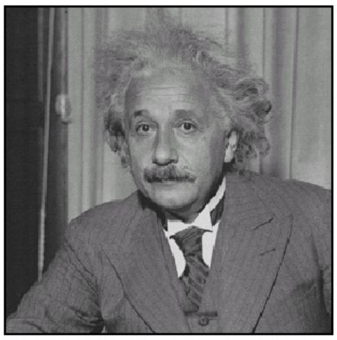
Sharpening filter

- Accentuates differences with local average

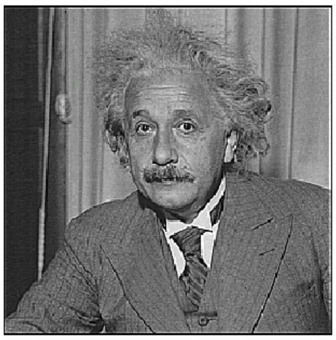
Source: D. Lowe

23

Sharpening



before



after

Source: D. Lowe

24

Sharpening

What does blurring take away?

原图　　平滑

减弱了变化较大的地方（轮廓边界）

相减后，就剩下轮廓

Source: D. Lowe

Sharpening

What does blurring take away?

E ⇑

$src * \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} =$

$= src * (E - G)$

$\Rightarrow G$

$src * \frac{1}{9}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$

Let's add it back:

$src * E$

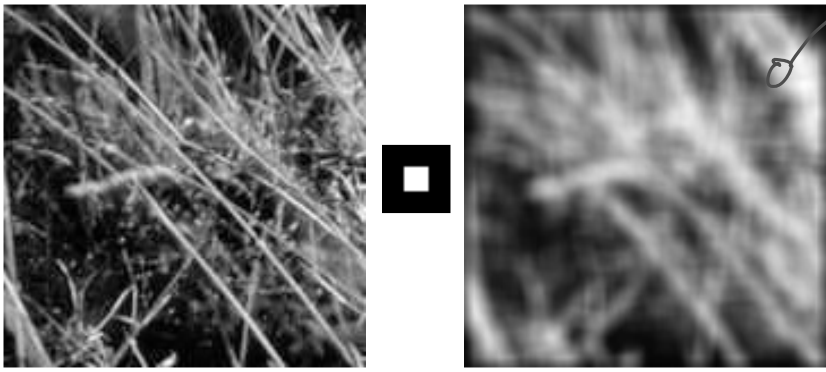
$= src * (2E - G)$

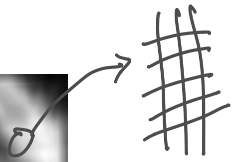
Source: D. Lowe

原图 + 轮廓 = 锐化

Smoothing with box filter revisited

- What’s wrong with this picture?
- What’s the solution?





格子效果明显

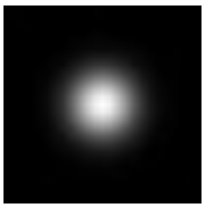
⇓

振临现象

Source: D. Forsyth

Smoothing with box filter revisited

- What’s wrong with this picture?
- What’s the solution?
  - To eliminate edge effects, weight contribution of neighborhood pixels according to their closeness to the center

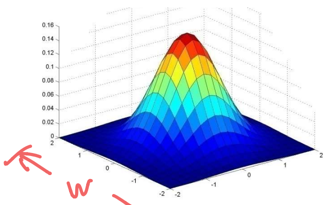


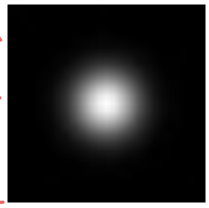
“fuzzy blob”

Source： S. Lazebnik

Gaussian Kernel

$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$





卷积核filter宽度 w

0.003	0.013	0.022	0.013	0.003
0.013	0.059	0.097	0.059	0.013
0.022	0.097	0.159	0.097	0.022
0.013	0.059	0.097	0.059	0.013
0.003	0.013	0.022	0.013	0.003

5 x 5, σ = 1

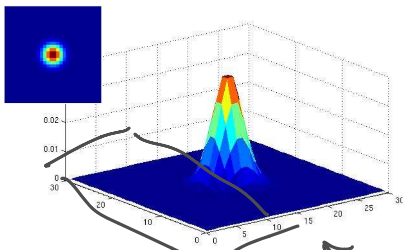
- Constant factor at front makes volume sum to 1 (can be ignored when computing the filter values, as we should renormalize weights to sum to 1 in any case)

Source: C. Rasmussen

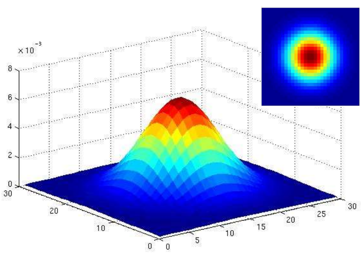
标准差 σ / 方差 σ^2 与 w，固定其一，加大另一个，会导致更平滑更模糊

Gaussian Kernel

$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$



σ = 2 with 30 x 30 kernel



σ = 5 with 30 x 30 kernel

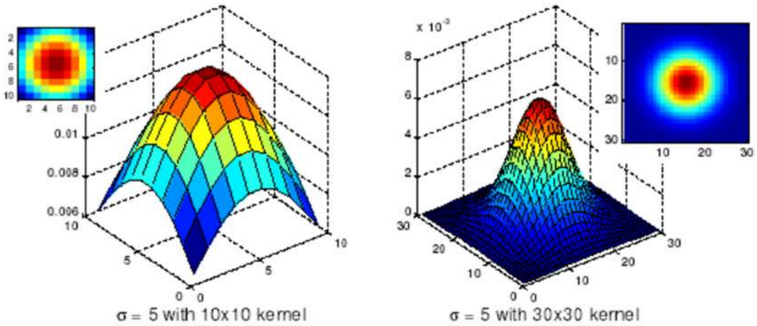
- Standard deviation σ: determines extent of smoothing

Source: K. Grauman

一般 w 取 3 倍 σ，不然 filter 外围都是 0，意义不大

Choosing kernel width

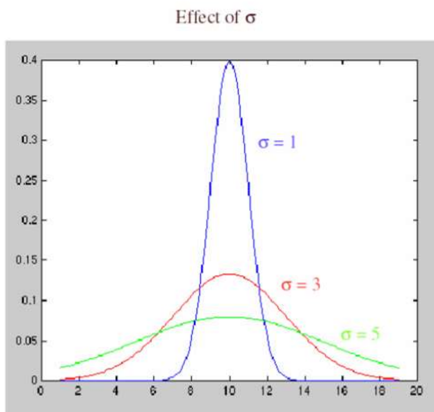
- The Gaussian function has infinite support, but discrete filters use finite kernels



Source: K. Grauman

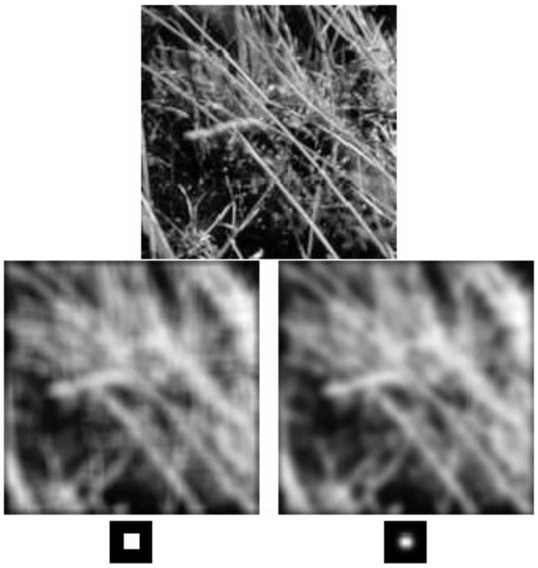
Choosing kernel width

- Rule of thumb: set filter half-width to about 3σ



Source： S. Lazebnik

Gaussian vs. box filtering



高斯模糊

振铃现象消失

Source：S. Lazebnik

Gaussian filters

- Remove “high-frequency” components from the image (low-pass filter)
- Convolution with self is another Gaussian
  - So can smooth with small-σ kernel, repeat, and get same result as larger-σ kernel would have
  - Convolving two times with Gaussian kernel with std. dev. σ is same as convolving once with kernel with std. dev. $\sigma\sqrt{2}$
- *Separable* kernel
  - Factors into product of two 1D Gaussians

Source: K. Grauman

Separability of the Gaussian filter

$$G_\sigma(x,y) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$= \left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{x^2}{2\sigma^2}}\right)\left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{y^2}{2\sigma^2}}\right)$$

The 2D Gaussian can be expressed as the product of two functions, one a function of x and the other a function of y

In this case, the two functions are the (identical) 1D Gaussian

Source: D. Lowe

Separability example

2D convolution (center location only)

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} * \begin{bmatrix} 2 & 3 & 3 \\ 3 & 5 & 5 \\ 4 & 4 & 6 \end{bmatrix}$$

卷积运算 $= 65$

The filter factors into a product of 1D filters:

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

矩阵运算

卷积运算

Perform convolution along rows:

$$\begin{bmatrix} 1 & 2 & 1 \end{bmatrix} * \begin{bmatrix} 2 & 3 & 3 \\ 3 & 5 & 5 \\ 4 & 4 & 6 \end{bmatrix} = \begin{pmatrix} 11 \\ 18 \\ 18 \end{pmatrix}$$

Followed by convolution along the remaining column:

$$\begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix} * \begin{pmatrix} 11 \\ 18 \\ 18 \end{pmatrix} = 65$$

Source: K. Grauman

若 $\vec{B} \cdot \vec{C} = A$ I是原图

(矩阵乘法)

$A * I = \vec{B} * \vec{C} * I$

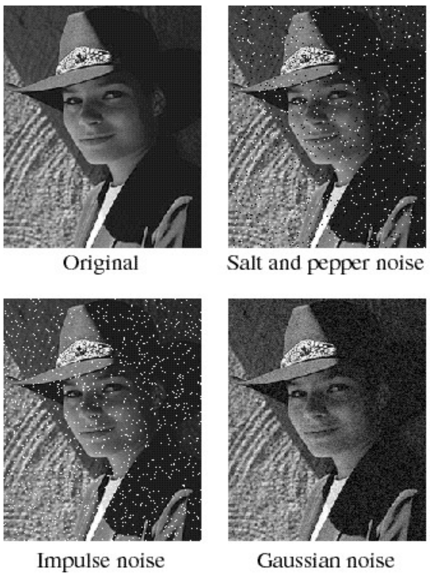
以上的做法可以降低运算复杂度

Why is separability useful?

- What is the complexity of filtering an $n \times n$ image with an $m \times m$ kernel?
  - $O(n^2 m^2)$
- What if the kernel is separable?
  - $O(n^2 m)$

Source： S. Lazebnik

Noise



- **Salt and pepper noise**: contains random occurrences of black and white pixels
- **Impulse noise:** contains random occurrences of white pixels
- **Gaussian noise**: variations in intensity drawn from a Gaussian normal distribution

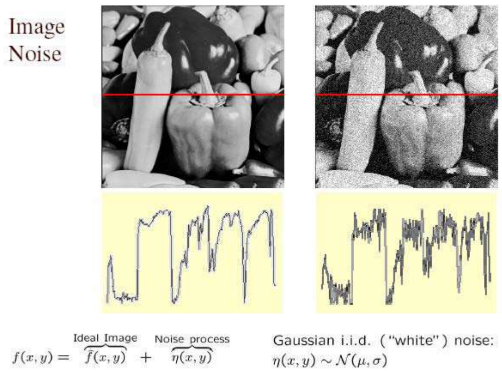
Source: S. Seitz

Gaussian noise

- Mathematical model: sum of many independent factors
- Good for small standard deviations
- Assumption: independent, zero-mean noise

高斯噪声

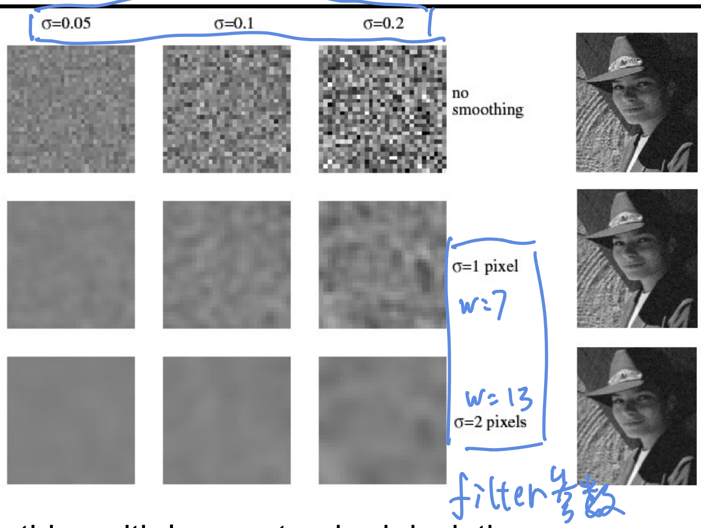
$\eta(x,y) \sim N(\mu,\sigma)$



Source: M. Hebert

Reducing Gaussian noise

高斯噪声 $N(\mu,\sigma)$ 参数

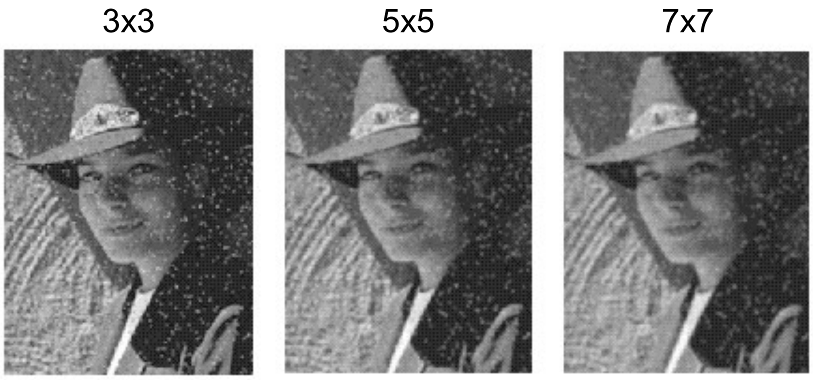


Smoothing with larger standard deviations suppresses noise, but also blurs the image

Source: S. Lazebnik

由大的σ产生的noise，可以用大的σ为参数的filter来降噪

Reducing salt-and-pepper noise



What's wrong with the results?

Gaussian 无法解决 椒盐噪声

Source: S. Lazebnik

Alternative idea: Median filtering

- A **median filter** operates over a window by selecting the median intensity in the window

中值滤波

10	15	20
23	90	27
33	31	30

Sort

Median value

10 15 20 23 $\boxed{27}$ 30 31 33 90

Replace

10	15	20
23	27	27
33	31	30

直接取周围

矩阵内, 大小适中的值

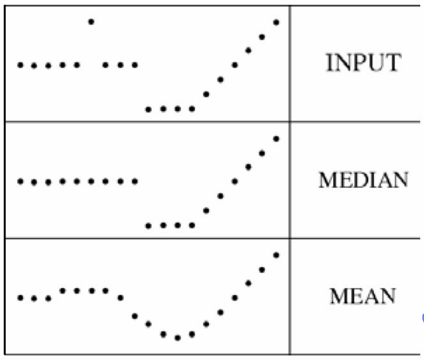
- Is median filtering linear?

Source: K. Grauman

Median filter

- What advantage does median filtering have over Gaussian filtering?
  - Robustness to outliers

filters have width 5 :



中值

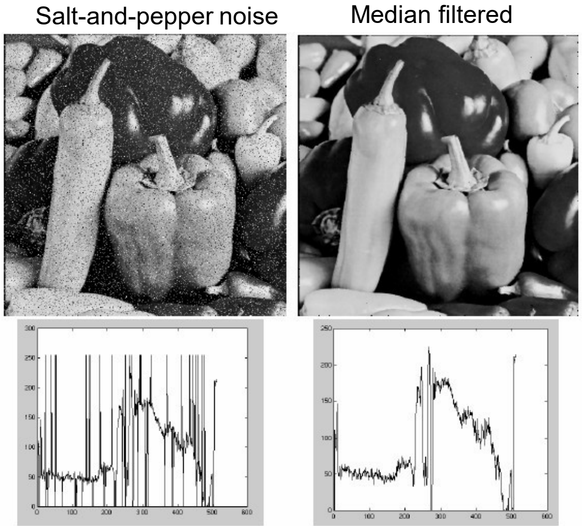
gaussian

Source: K. Grauman

Median filter

Salt-and-pepper noise

Median filtered



Source: M. Hebert

Gaussian vs. median filtering

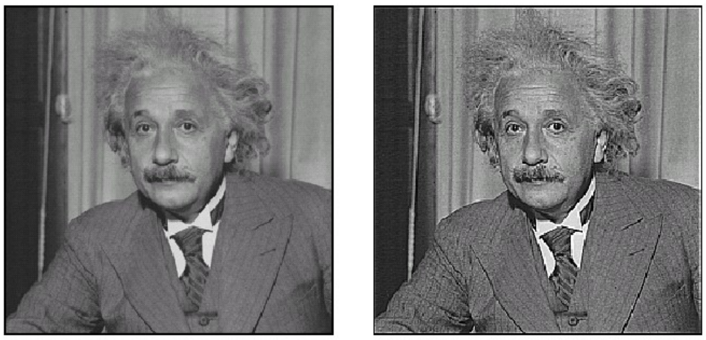
3x3 5x5 7x7

Gaussian

Median

Source: S. Lazebnik

Sharpening revisited



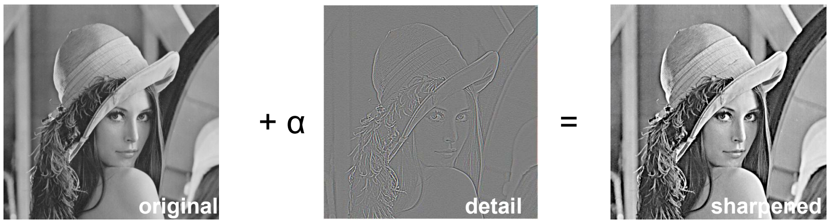
before after

Source: D. Lowe

Sharpening revisited

What does blurring take away?

Let's add it back:



Source: S. Lazebnik

Unsharp mask filter

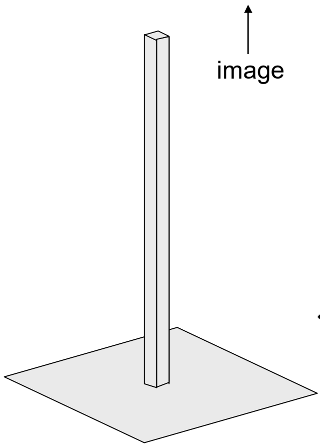
控制锐化的强度

$$f + \alpha(f - f * g) = (1+\alpha)f - \alpha f * g = f * ((1+\alpha)e - \alpha g)$$

image

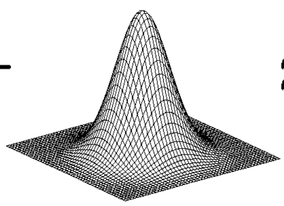
blurred image

unit impulse (identity)



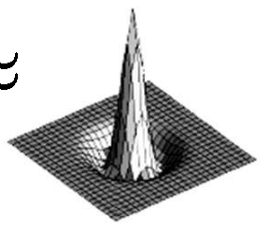
unit impulse

—



Gaussian

≈



Laplacian of Gaussian

Source: S. Lazebnik