

# 信号与图像处理基础

---

## Spatial Filtering

中国科学技术大学 自动化系

曹 洋





# 空域滤波

---

$W_1$	$W_2$	$W_3$
$W_4$	$W_5$	$W_6$
$W_7$	$W_8$	$W_9$

◆空域滤波和空域滤波器的定义：

使用空域模板进行的图像处理，被称为空域滤波。模板本身被称为空域滤波器。

◆空域模板

空域模板是一个系数矩阵，也可看作是一副子图像。它是图像空域运算的基本处理单元，也被称之为核函数或滤波器。



# 空域滤波

---

在 $M \times N$ 的图像 $f$ 上，使用 $m \times n$ 的滤波器：

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

$$m = 2a + 1, n = 2b + 1$$

空间滤波的简化形式：

$$R = w_1 z_1 + w_2 z_2 + \cdots + w_{mn} z_{mn}$$

其中， $w$ 是滤波器系数， $z$ 是与该系数对应的图像灰度值， $mn$ 为滤波器中包含的像素点总数。



# 空域滤波

---

在空域滤波功能都是利用模板运算，主要步骤为：

- (1) 将模板在图中漫游，并将模板中心与图中某个像素位置重合；
- (2) 将模板上系数与模板下对应像素相乘；
- (3) 将所有乘积相加；
- (4) 将和（模板的输出响应）赋给图中对应模板中心位置的像素。

# 举例：线性空域滤波



Pixels of image

	$w(-1,-1)$ $f(x-1,y-1)$	$w(-1,0)$ $f(x-1,y)$	$w(-1,1)$ $f(x-1,y+1)$	
	$w(0,-1)$ $f(x,y-1)$	$w(0,0)$ $f(x,y)$	$w(0,1)$ $f(x,y+1)$	
	$w(1,-1)$ $f(x+1,y-1)$	$w(1,0)$ $f(x+1,y)$	$w(1,1)$ $f(x+1,y+1)$	

滤波器的输出是：将模板系数与对应像素相乘，之后再求和。类似于加权求和。

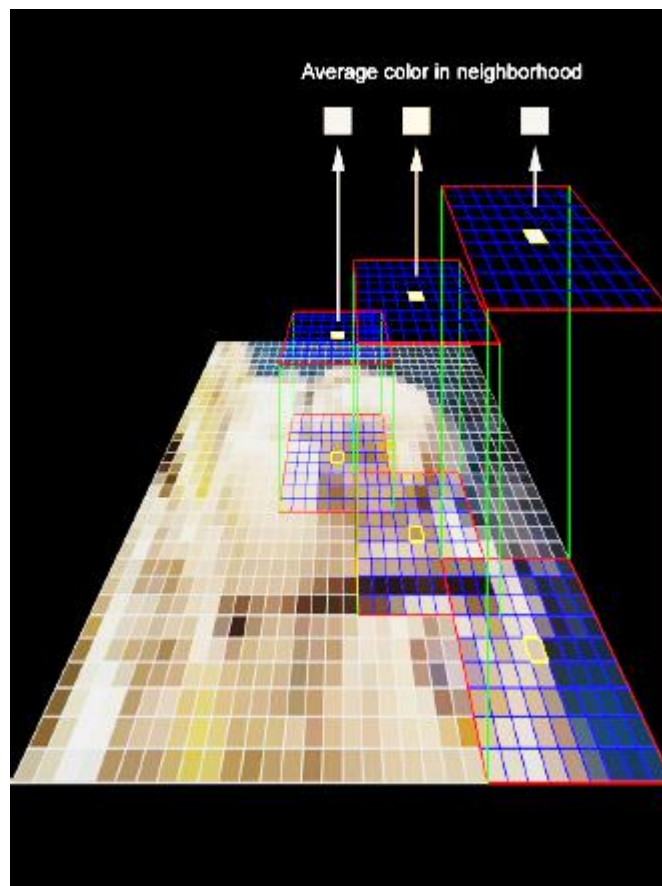
模板系数

$w(-1,-1)$	$w(-1,0)$	$w(-1,1)$
$w(0,-1)$	$w(0,0)$	$w(0,1)$
$w(1,-1)$	$w(1,0)$	$w(1,1)$

$$f(x, y) = w(-1,-1)f(x-1, y-1) + w(-1,0)f(x-1, y) + w(-1,1)f(x-1, y+1) + w(0,-1)f(x, y-1) + w(0,0)f(x, y) + w(0,1)f(x, y+1) + w(1,-1)f(x+1, y-1) + w(1,0)f(x+1, y) + w(1,1)f(x+1, y+1)$$

# 模板运算

- 对于一个 $m \times n$  的模板，通常会假设 $m=2a+1$ ， $n=2b+1$ ，其中 $a$ ， $b$  为非负整数。也就是说 $m$ 和 $n$ 为奇数。





# 线性空域滤波

---

- 线性空域滤波的数学表达式为：

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

- 这与图像卷积非常类似

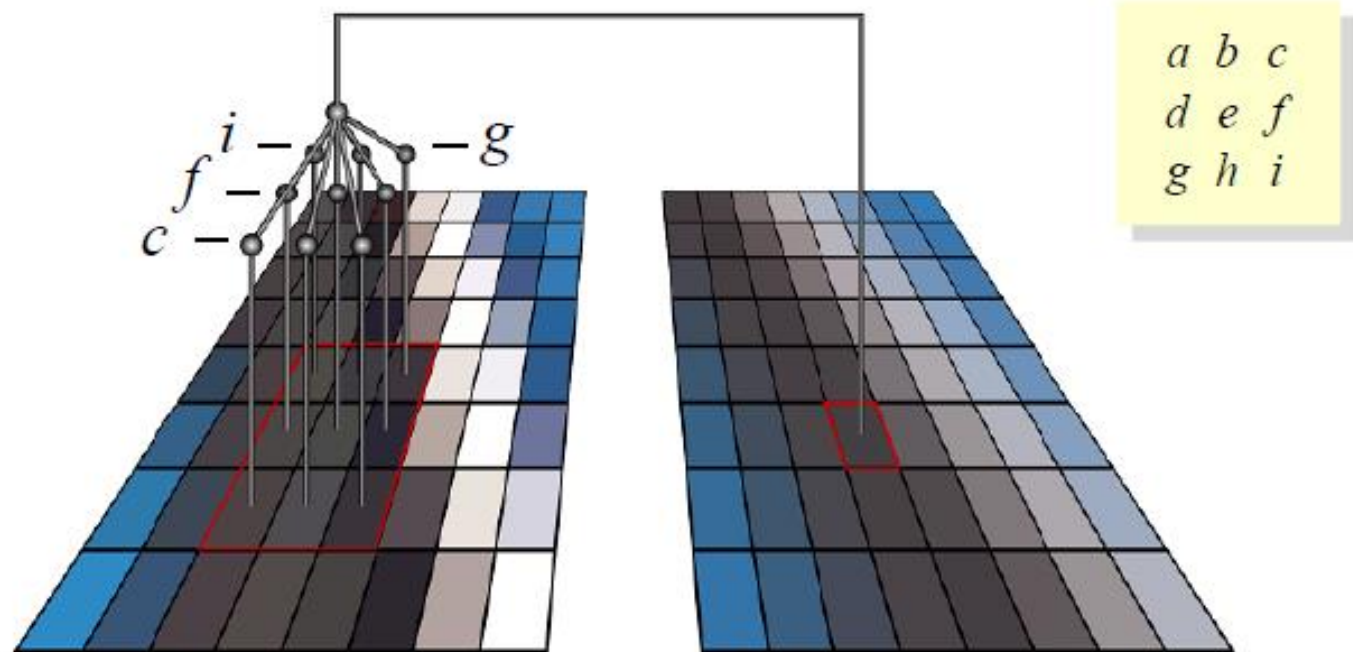
$$R = w_1 z_1 + w_2 z_2 + \dots + w_9 z_9 = \sum_{i=1}^9 w_i z_i$$

$w_1$	$w_2$	$w_3$
$w_4$	$w_5$	$w_6$
$w_7$	$w_8$	$w_9$





# 线性空域滤波

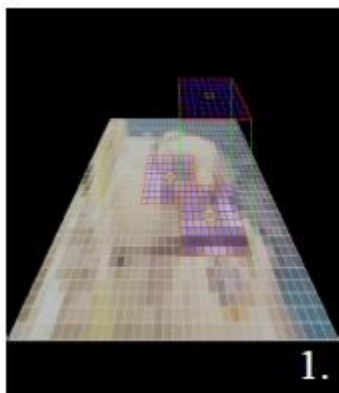






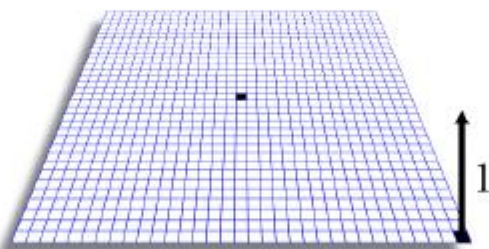
# 图像卷积

- 三种计算图像滤波的方法：
  - 模板运算
  - 平移+求积+叠加
  - 频域运算



# 图像卷积

- 平移+求积+叠加



$$\delta(r-16, c-16)$$

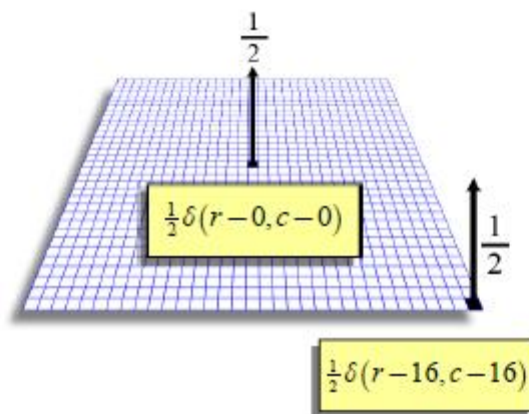
Shifted down and to the right by 16 pixels.





# 图像卷积

- 平移+求积+叠加



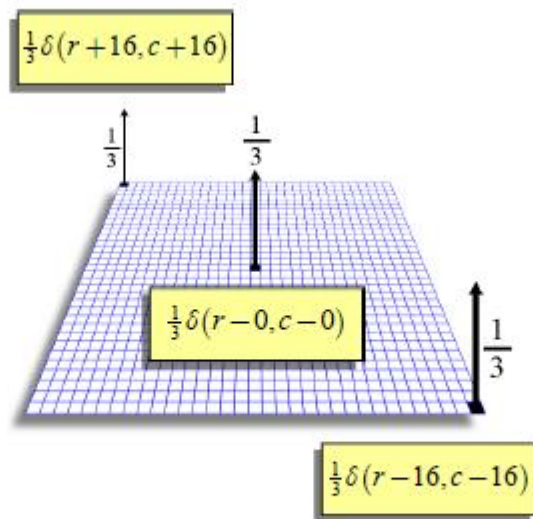
Two copies, one moved,  
one not moved, averaged.





# 图像卷积

- 平移+求积+叠加



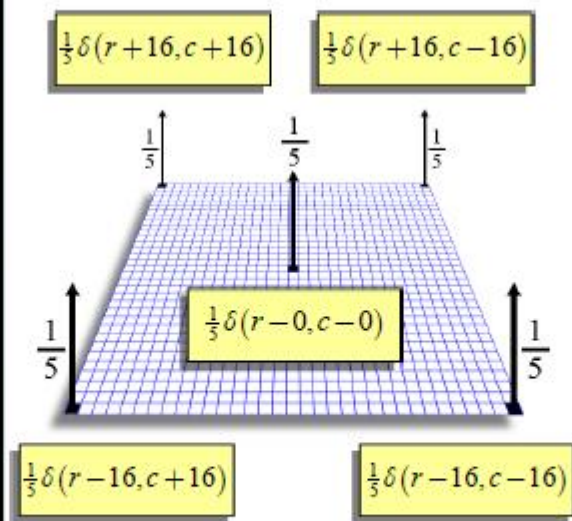
Three copies, two moved,  
one not moved, averaged.





# 图像卷积

- 平移+求积+叠加



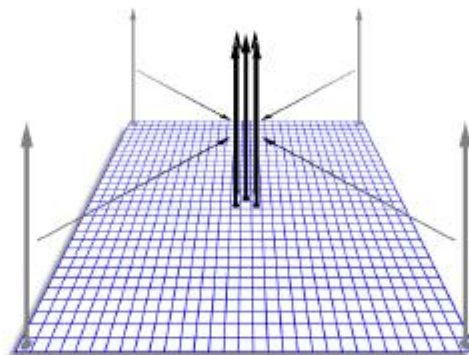
Five copies, four moved,  
one not moved, averaged.





# 图像卷积

- 平移+求积+叠加



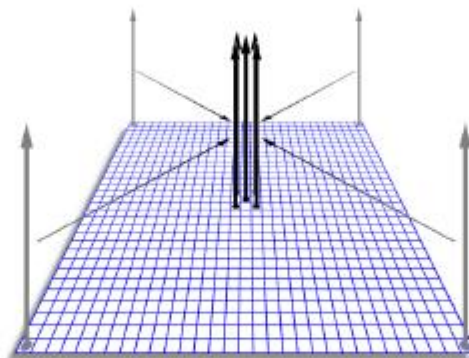
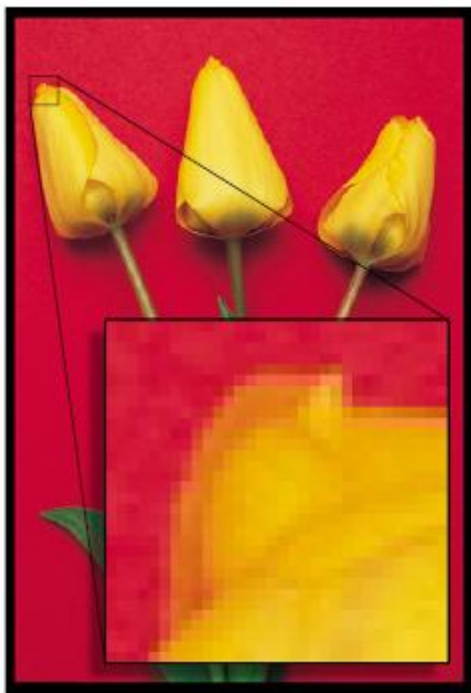
Moved adjacent to each other, the convolution becomes a blurring filter.





# 图像卷积

- 平移+求积+叠加



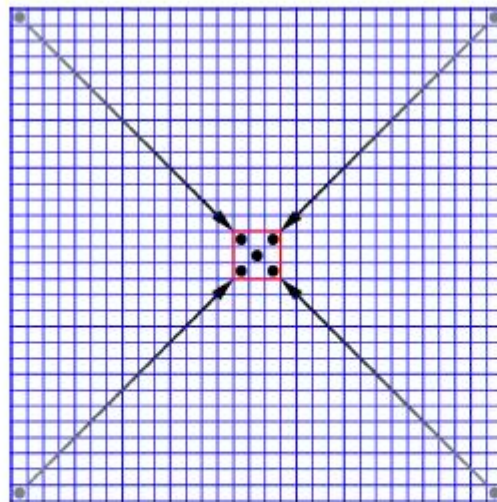
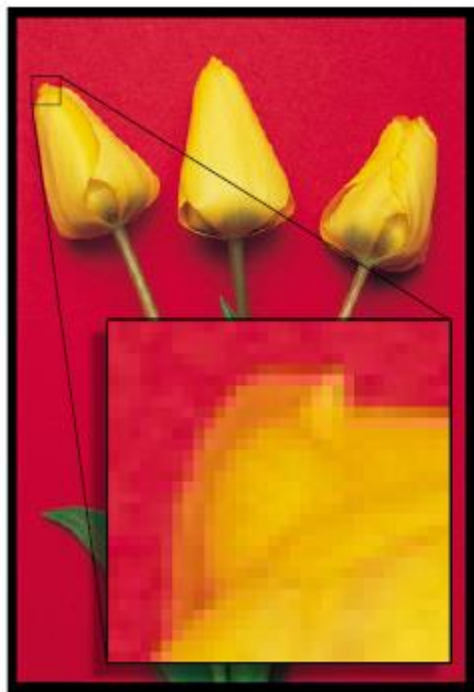
The impulses become values in a 3x3 neighborhood.



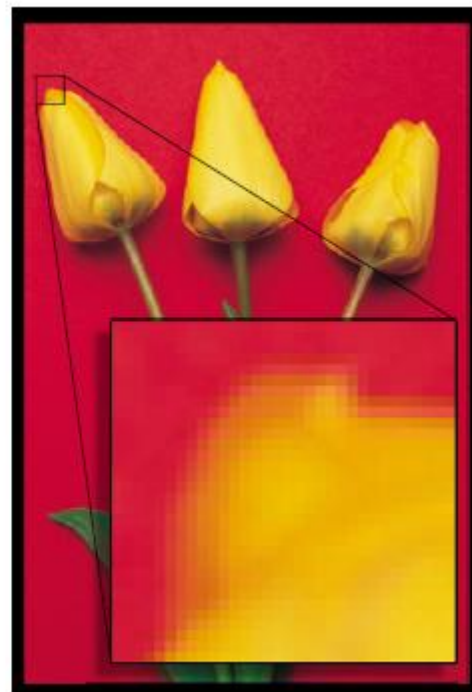


# 图像卷积

- 平移+求积+叠加



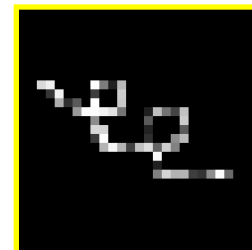
The convolution mask has five elements at  $1/5$  and four at  $0$ .



# 图像卷积



=

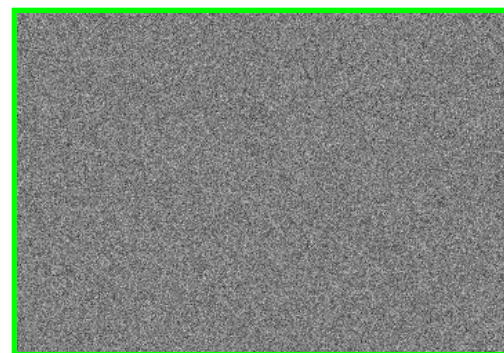


Blurred image  $I$

Sharp image  $L$

Blur kernel  $h$

+



Camera Noise  $n$



# 非线性空域滤波

---

- 非线性空域滤波也是作用在图像邻域上，采用滑窗式的模板运算，将运算输出赋给处理像素。
- 但是其运算并不是基于线性计算，而通常是基于启发式的规则，例如对于邻域范围内的像素进行排序等。



# 平滑空域滤波

---

- 作用

- (1) 模糊处理: 去除图像中一些不重要的细节
- (2) 减小噪声。

- 平滑空间滤波器的分类

- (1) 线性平滑滤波器: 如均值滤波器

- (2) 非线性平滑滤波器:

如最大值滤波器

中值滤波器

最小值滤波器



# 线性平滑空域滤波

---

- 也被称作邻域平均法，输出包含在滤波器邻域内像素的加权平均值。
- 作用
  - (1) 减小图像灰度的“尖锐”变化，减小噪声。
  - (2) 由于图像边缘是由图像灰度尖锐变化引起的，所以也存在边缘模糊的问题。

# 线性平滑空域滤波

- 线性平滑空域滤波的通用表达式

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

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

1	1	1
1	1	1
1	1	1

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

1	2	1
2	4	2
1	2	1





# 线性平滑空域滤波

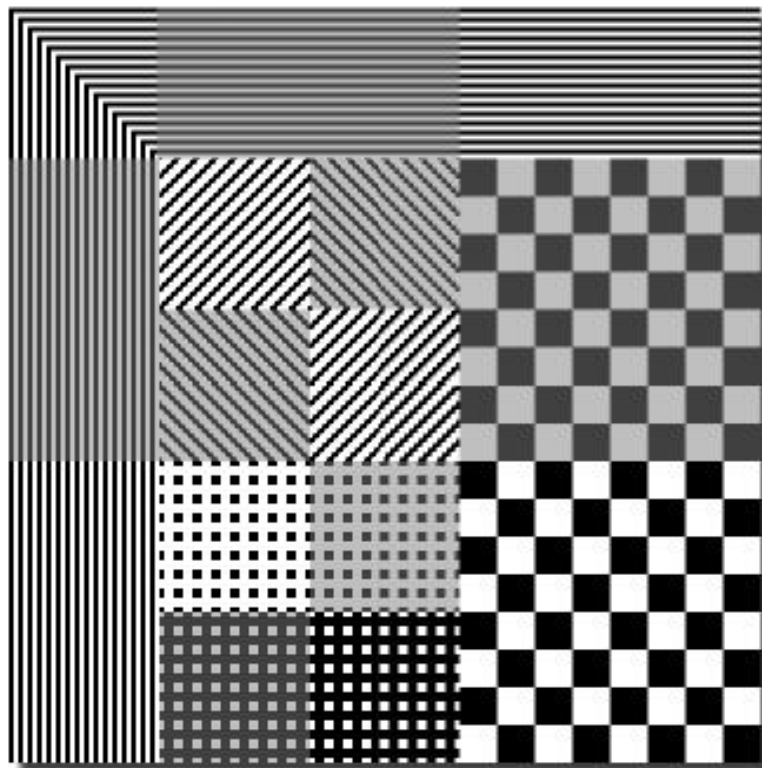
---

$$\frac{1}{25} \times \times$$

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1



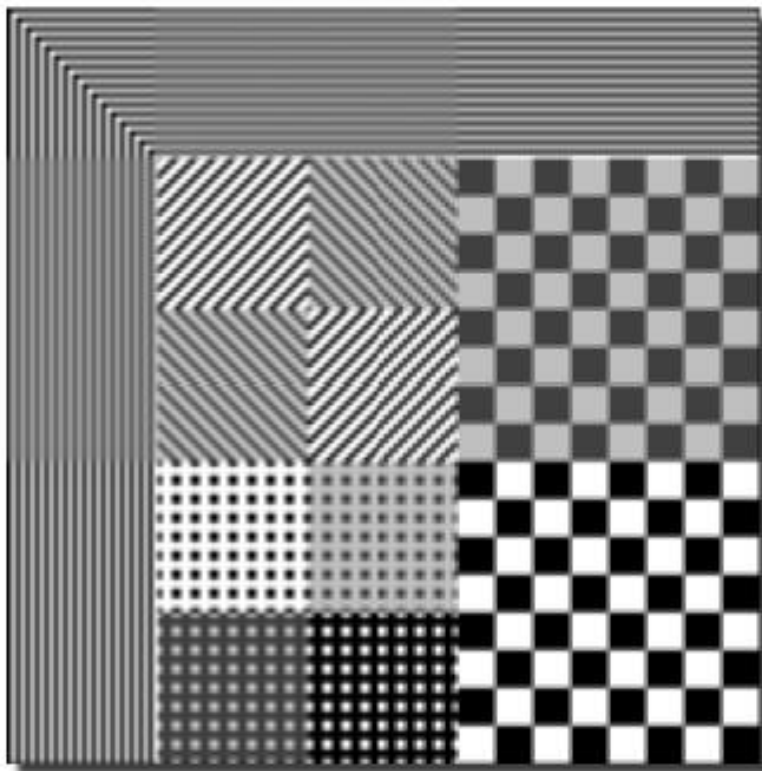
# 线性平滑空域滤波





# 线性平滑空域滤波

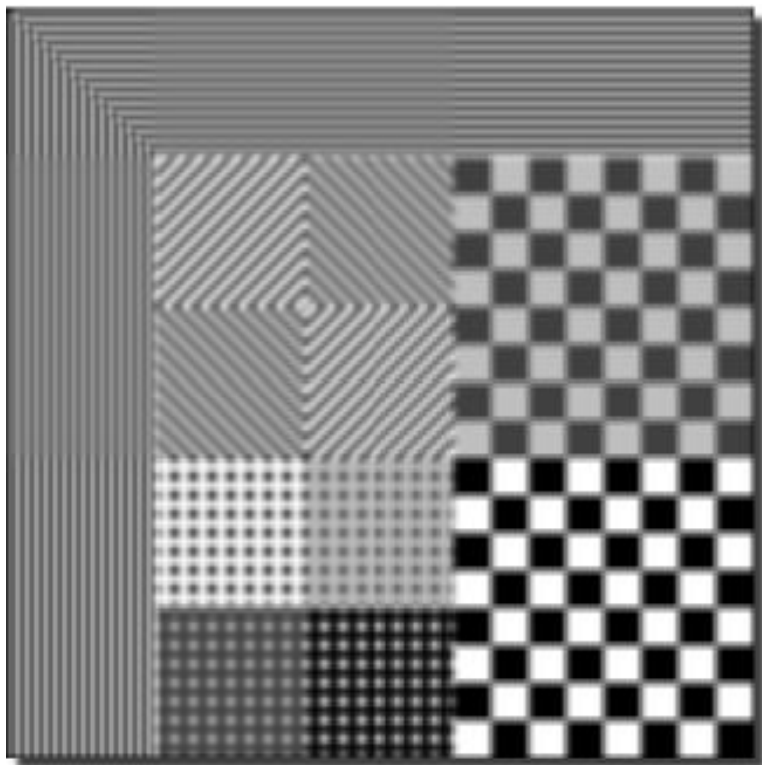
$$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$





# 线性平滑空域滤波

$$\frac{1}{25} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

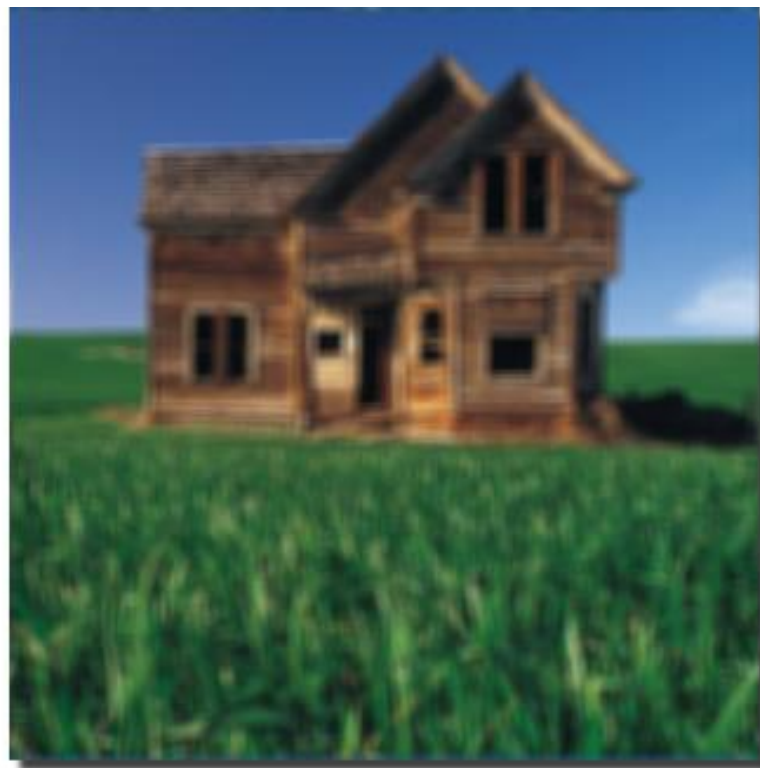
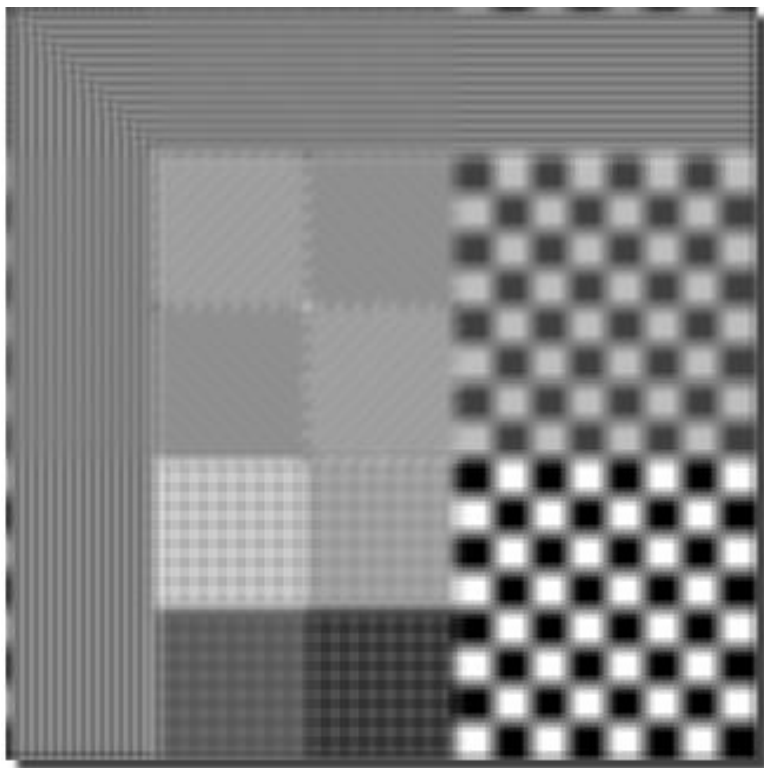






# 线性平滑空域滤波

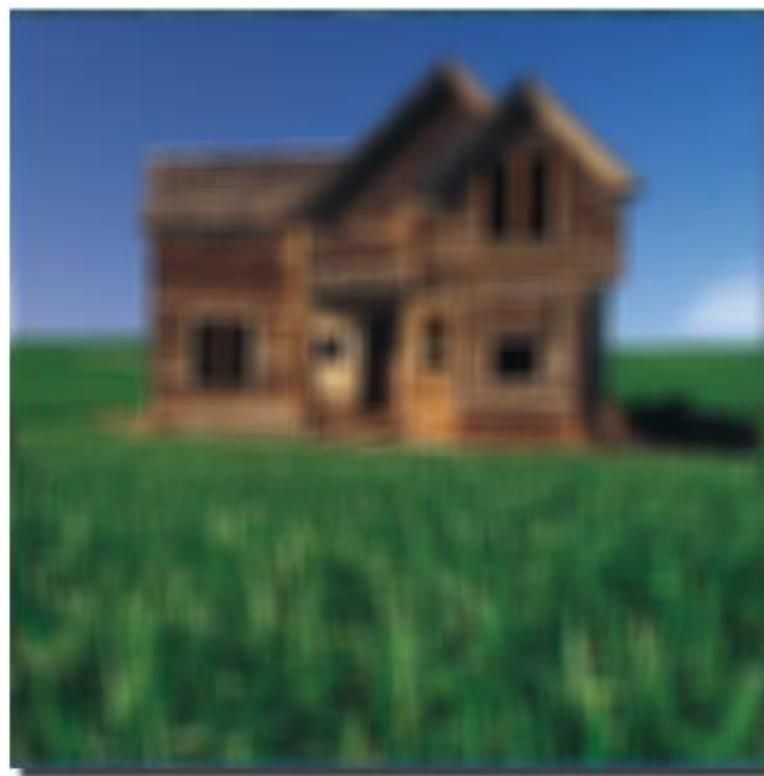
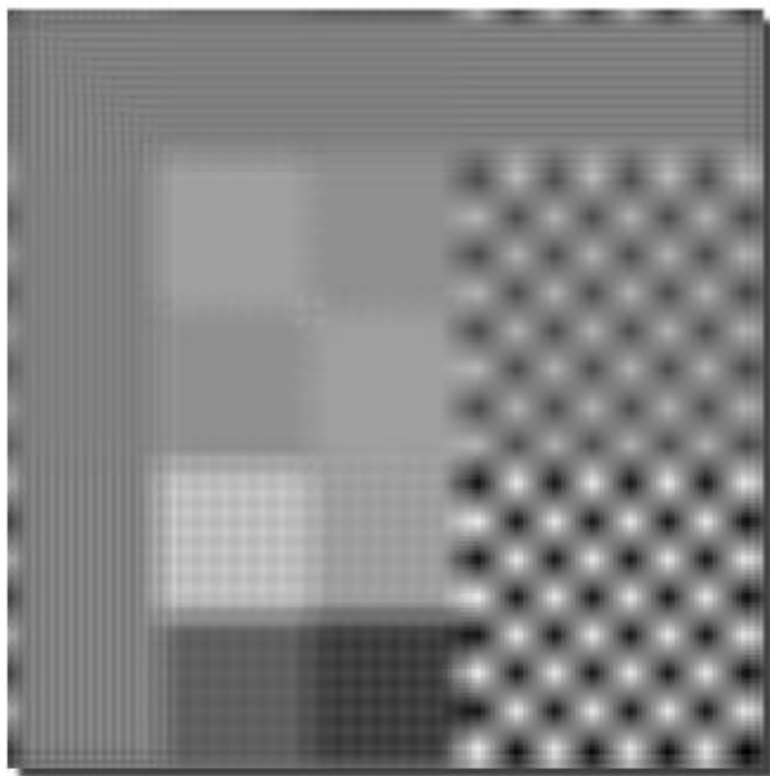
1111111111  
1111111111  
1111111111  
1111111111  
 $\frac{1}{81}$  1111111111  
1111111111  
1111111111  
1111111111  
1111111111





# 线性平滑空域滤波

$\frac{1}{289}$



# 图像噪声



(a) 椒盐噪声



(b) 高斯噪声

**Pepper and Salt noise: similar intensity, random distribution**

**Gaussian Noise: locate at every pixel, random intensity**



# 基于邻域平滑滤波的图像去噪







# 线性平滑空域滤波

## 图像去噪：空域平滑法

现象：邻域平均法在去除噪声的同时，也可造成图像边缘等细节的模糊。而且，这种模糊效应将随所取模板尺寸的增大而愈趋严重。

0	1/5	0
1/5	1/5	1/5
0	1/5	0

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

$\frac{1}{25}$	1	1	1	1	1
	1	1	1	1	1
	1	1	1	1	1
	1	1	1	1	1
	1	1	1	1	1
	1	1	1	1	1

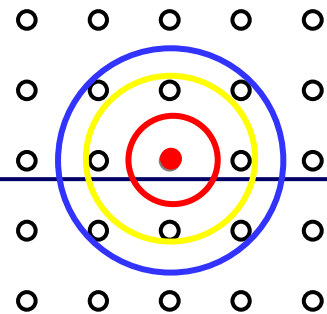
去噪能力

模糊程度

抑制模糊的对策(1)：合理选择模板的尺寸。为避免处理结果过于模糊的情况发生，应选择尺寸相对小一点的模板。



# 线性平滑空域滤波



## 图像去噪：空域平滑法

现象：在模板尺寸相同的情况下，选择不同的权值矩阵所造成的模糊程度也有差异。此时，模糊效应将随邻域半径的增大而愈趋严重。

差异权重

$\frac{1}{16}$	1	2	1
	2	4	2
	1	2	1

差异权重

$\frac{1}{10}$	1	1	1
	1	2	1
	1	1	1

等权重

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

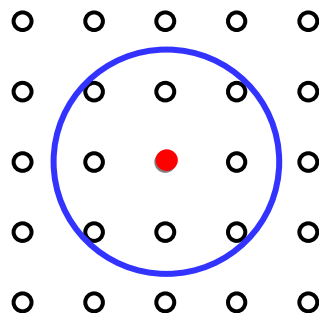
模糊程度

抑制模糊的对策(2)：合理选择各模板元素的权值。例如，使模板中心元素和距离中心较近的元素具有相对大的权值以突出待处理像素点、尽量抑制由平滑化处理引入的模糊。

# 线性平滑空域滤波

## 图像去噪：空域平滑法

抑制模糊的对策(3)：使用多模板技术，根据对当前点是否为噪声点的检测结果，选择使用具有不同权值的模板完成计算。



用于噪声点



1/8	1/8	1/8
1/8	0	1/8
1/8	1/8	1/8

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

0	0	0
0	1	0
0	0	0



用于非噪声点

# 线性平滑空域滤波

## 图像去噪：空域平滑法

### 是否孤立噪声点的检测

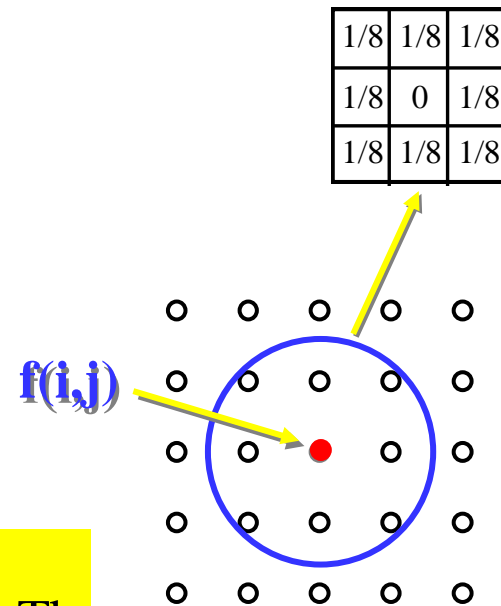
#### 检测规则

对 $f(i,j)$ , 若

$$\left| f(i,j) - \frac{1}{N} \sum_{(m,n) \in T} f(m,n) \right| > Th$$

则判 $f(i,j)$ 为噪声点, 否则判为非噪声点。

根据：孤立噪声点的灰度值与邻域平均灰度值相差悬殊。





# 线性平滑空域滤波

---

## 图像去噪：空域平滑法

### 是否孤立噪声点的检测

检测规则的检测效果在很大程度上取决于邻域的选择。一般而言，一种邻域的选择不可能同时适用于所有模式的图像边缘点。

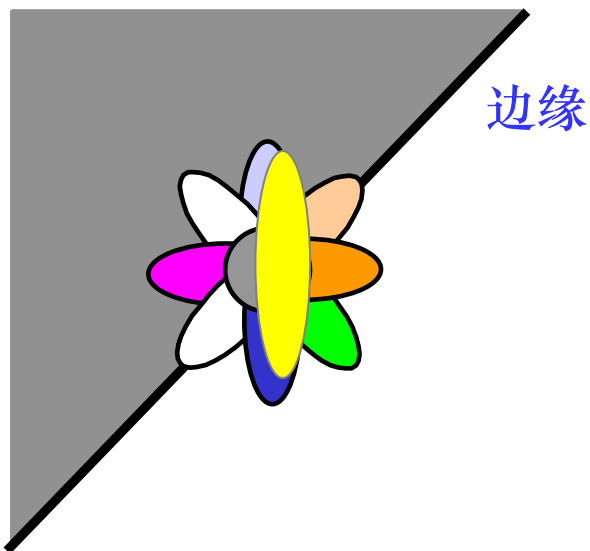


在某些情形下，上述方法给出的结果是有限的。例如，对于横跨两个灰度差异较大区域的点。

# 线性平滑空域滤波

## 图像去噪：空域平滑法

### 是否孤立噪声点的改进检测

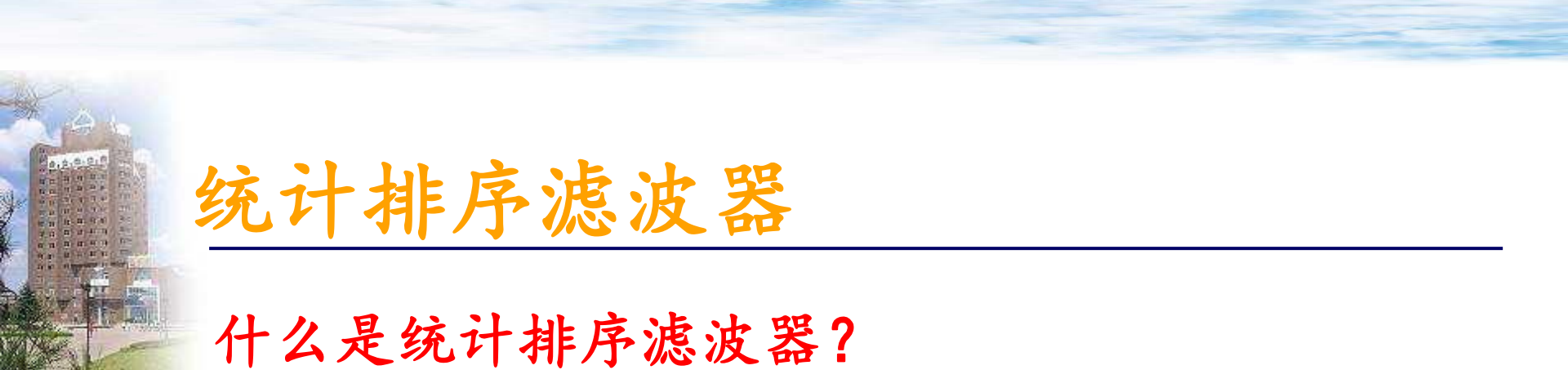


以待处理像素为中心，设置9个邻域模式。对每一种邻域模式 $k$ ，分别计算邻域内元素的均值 $m_k$ 和方差 $\sigma_k$ ， $k=1,2,\dots,9$ 。

噪声点判决规则如下：当

$$\sigma_k = \min_l \{ \sigma_l \} > Th$$

时，待处理像素点为噪声点。



# 统计排序滤波器

---

什么是统计排序滤波器？

是一种非线性滤波器，基于滤波器所在图像区域中像素的排序，由排序结果决定的值代替中心像素的值。

分类：

(1) 中值滤波器：用像素领域内的中间值代替该像素。

(2) 最大值滤波器：用像素领域内的最大值代替该像素。

(3) 最小值滤波器：用像素领域内的最小值代替该像素。





# 统计排序滤波器

---

- 中值滤波器

主要用途：去除噪声

计算公式： $R = \text{mid} \{z_k \mid k = 1, 2, \dots, n\}$

- 最大值滤波器

主要用途：寻找最亮点

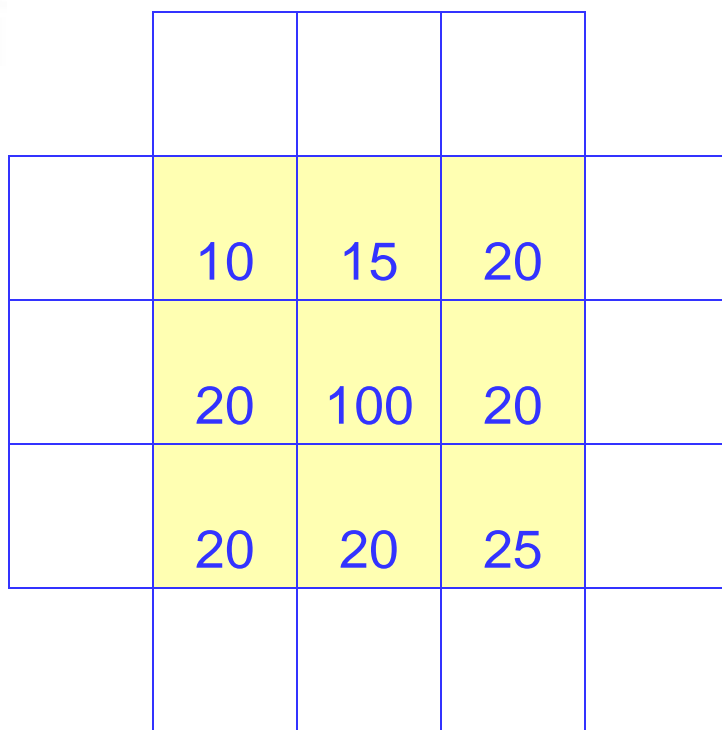
计算公式： $R = \max \{z_k \mid k = 1, 2, \dots, n\}$

- 最小值滤波器

主要用途：寻找最暗点

计算公式： $R = \min \{z_k \mid k = 1, 2, \dots, n\}$

# 中值滤波器



	10	15	20	
	20	100	20	
	20	20	25	

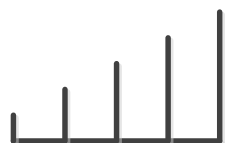
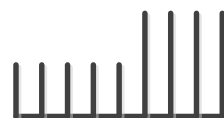
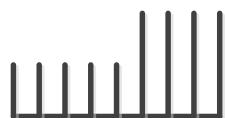
10, 15, 20, 20, 20, 20, 20, 25, 100

↑  
5<sup>th</sup>

- 用模板区域内像素的中间值，作为结果值
- $R = \text{mid} \{z_k \mid k = 1, 2, \dots, n\}$
- 强迫突出的亮点或暗点更象它周围的值，以消除孤立的亮点或暗点。



# 中值滤波的结果（一维）



原始信号

滤波结果

# 中值滤波的结果（二维）



椒盐噪声

# 中值滤波的结果（二维）



高斯噪声





# 中值滤波器

---

- 中值滤波算法的特点：
  - (1) 在去除噪音的同时，可以比较好地保留边的锐度和图像的细节（优于均值滤波器）
  - (2) 能够有效去除脉冲噪声：以黑白点（椒盐噪声）叠加在图像上中。

# 图像去噪的对比实验结果

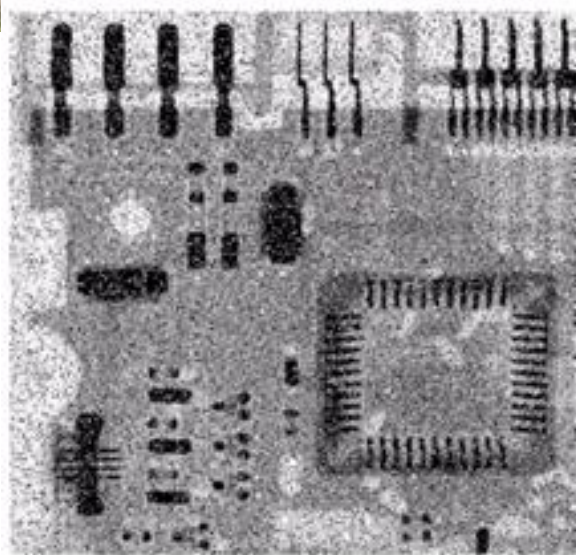


**Median filter**

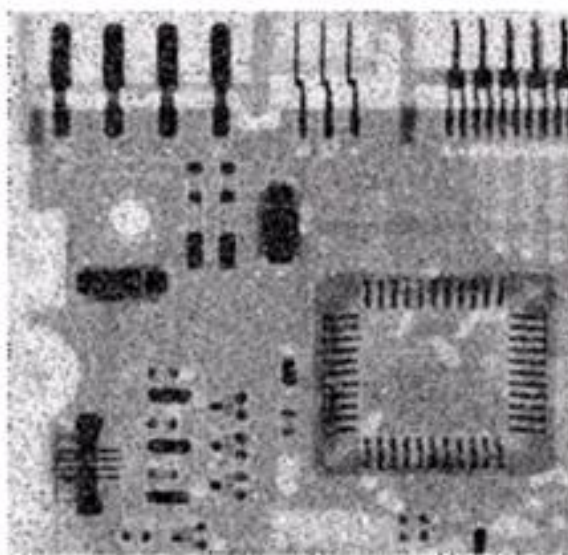


**Means filter**

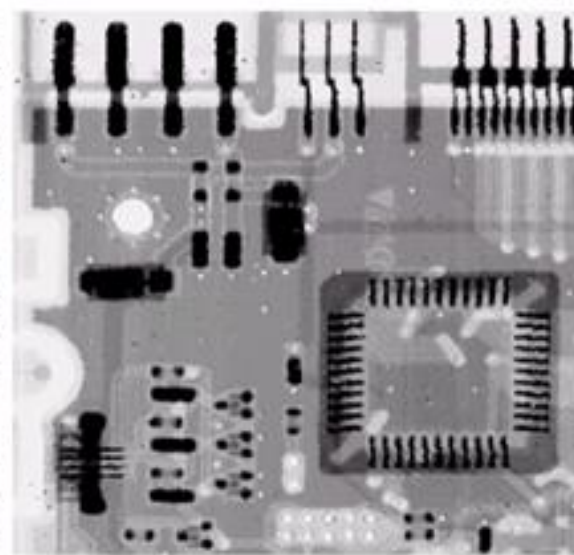
# 图像去噪的对比实验结果



Noisy image  
(Pepper and  
Salt noise)



Means  
filtering  
result



Median  
filtering  
result





# 最大值滤波器



Maxwell, P. (2001). The use of the maximum filter in the analysis of soccer match data. *Journal of Sports Sciences*, 19(1), 1-10.

# 最小值滤波器



**Man City's Richard Edgill tackles Newcastle's David Ginola at St James Park. City lost the match 3-1      Picture: REUTERS**





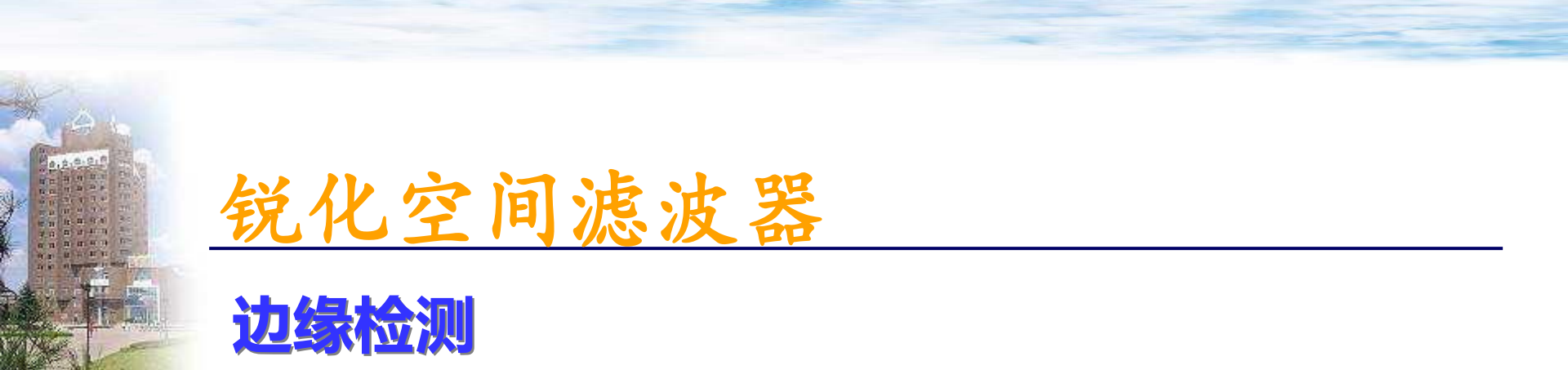
# 锐化空间滤波器

---

主要用于增强图像的边缘及灰度跳变部分

{ 邻域平均方法—积分过程—结果使图像的  
边缘模糊  
锐化方法—微分过程—结果使图像的边缘突出

边缘提取



# 锐化空间滤波器

---

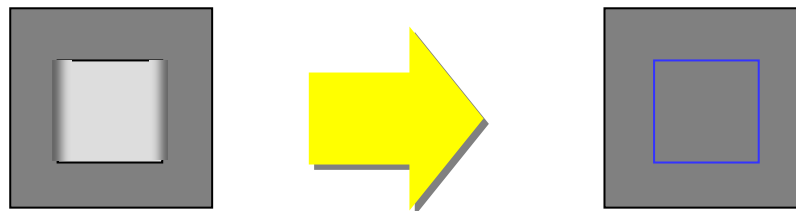
## 边缘检测

**什么是图像边缘?**

**图像灰度发生跳变的场所。**

**什么是图像边缘检测?**

**在灰度图像的情况下，基于图像像素的灰度值在空间的不连续性对图像作出的一种分割。**

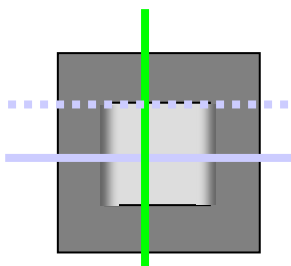




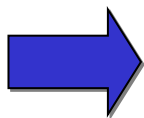
# 锐化空间滤波器

## 边缘检测

### 图像边缘的描述



**方向**  
**幅度**

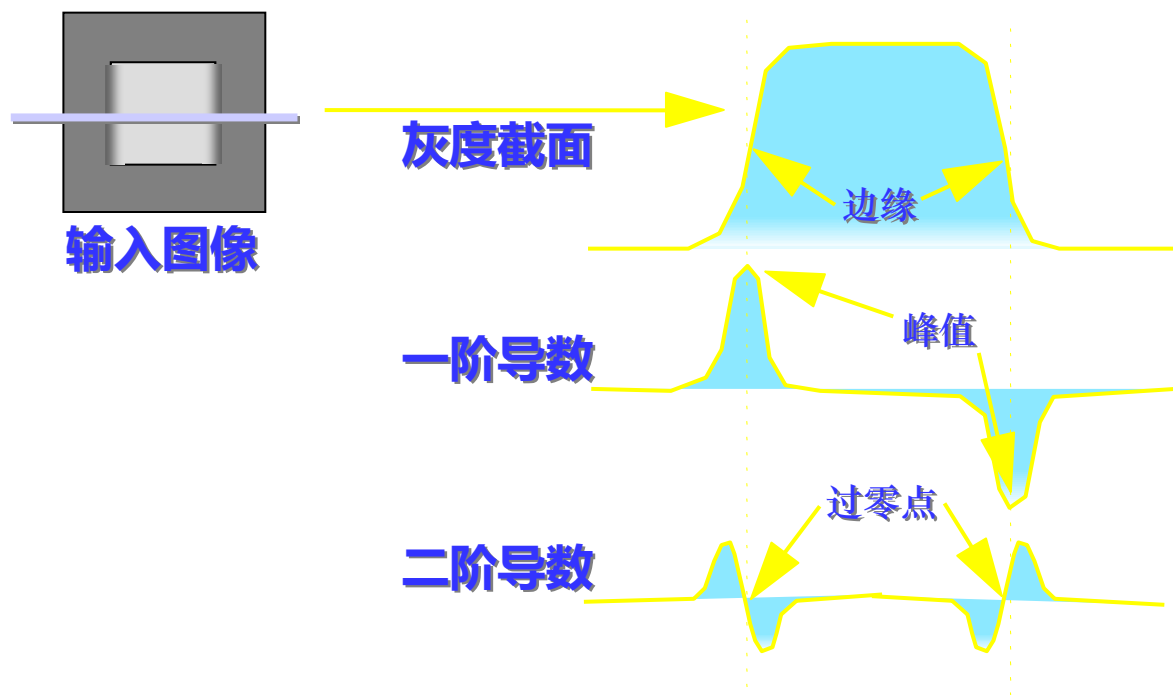


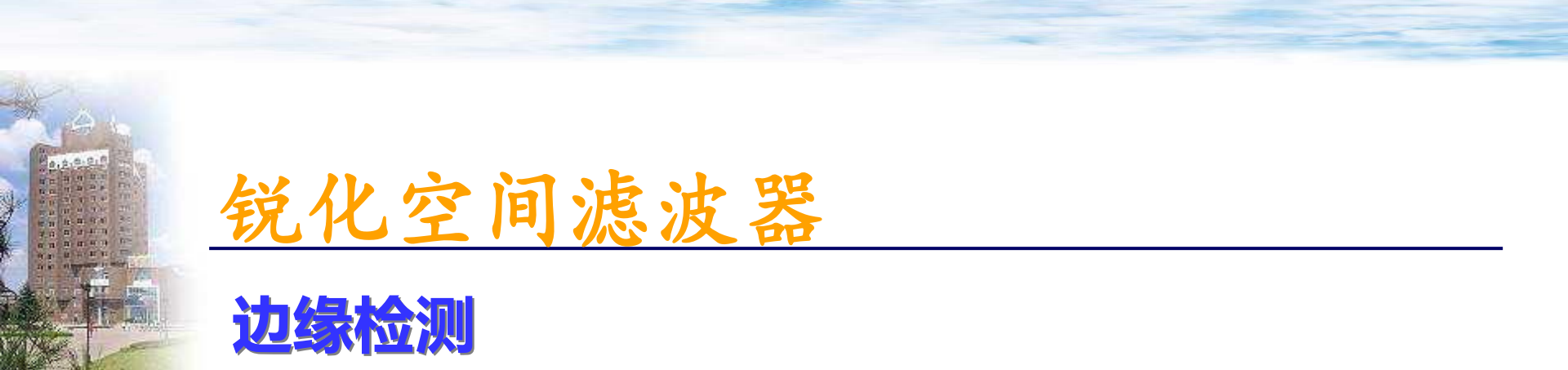
沿边缘走向方向其幅度值变化较平缓，  
而沿垂直于边缘走向方向其幅度值变化  
较剧烈

# 锐化空间滤波器

## 边缘检测

### 灰度截面与图像边缘





# 锐化空间滤波器

---

## 边缘检测

- 判据1

在边缘处，图像的一阶导数的幅度值较大，并在其附近形成一个峰值。其峰值的大小和极性反映了边缘点的强度和方向。

- 判据2

图像的二阶导数在边缘处取零值，并且，在该零值的左右两侧分别存在极性相反的两个波峰；其波峰的大小和走向反映了边缘点的强度和方向。





# 锐化空间滤波器

---

## 边缘检测 一阶微分算子

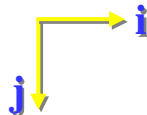
如果沿垂直于某个边缘走向方向求图像的一阶微分，则在相应的边缘点处，其微分的幅度值将较大；而在其它的图像点处，其微分的幅度值则取较小的值。这样，通过将各点处图像的一阶微分的幅度值同事先设定的阈值进行比较，可以很容易根据边缘检测的判据1判定该点是不是一个边缘点。

# 锐化空间滤波器

## 边缘检测 一阶微分算子

差分运算可以用卷积算子与图像的卷积来表示。

□ 常用的一阶微分运算:



(1) 沿X方向的一阶差分  $\Delta_x$ :  $\Delta_x f(i,j) = f(i,j) - f(i-1,j)$

-1	1
----	---

(2) 沿X方向的一阶差分  $\Delta_{2x}$ :  $\Delta_{2x} f(i,j) = f(i+1,j) - f(i-1,j)$

-1	0	1
----	---	---

(3) 沿Y方向的一阶差分  $\Delta_y$ :  $\Delta_y f(i,j) = f(i,j) - f(i,j+1)$

1
-1

(4) 沿Y方向的一阶差分  $\Delta_{2y}$ :  $\Delta_{2y} f(i,j) = f(i,j-1) - f(i,j+1)$

1
0
-1

(5) 沿45°对角线方向的一阶差分  $\Delta_+$ :

$$\Delta_+ f(i,j) = f(i-1,j+1) - f(i,j)$$

0	1
-1	0

(6) 沿135°对角线方向的一阶差分  $\Delta_-$ :

$$\Delta_- f(i,j) = f(i-1,j-1) - f(i,j)$$

1	0
0	-1

# 锐化空间滤波器

## 边缘检测 一阶微分算子

□ 其它常用的微分运算：

一阶差分运算因使用的数据少，对噪声比较敏感。为了克服这个缺点、提高算子本身的抗干扰能力，可以考虑使用较大尺寸的模板。

**Sobel算子**

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

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

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

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

**Priwitt算子**

-1	0	1
-1	0	1
-1	0	1

1	1	1
0	0	0
-1	-1	-1

0	1	1
-1	0	1
-1	-1	0

1	1	0
1	0	-1
0	-1	-1



# 锐化空间滤波器

---

## 边缘检测 一阶微分算子

### □ 关于差分运算的一些结论：

- 差分运算（也即微分运算）是一种有方向性的运算，其运算结果与所取的具体差分方向有关。
- 为了获得好的边缘检测性能，要求所进行差分运算的差分方向与边缘方向保持垂直。
- 为了检测出图像上所有的边缘点，必须沿不同方向进行相应的差分运算，并选择差分输出的绝对值大于给定阈值的点作为边缘点。
- 为了克服一阶差分算子存在的缺点，希望能找到一种各向同性的运算，并能根据该运算得到图像中的边缘。



# 锐化空间滤波器

## 边缘检测 梯度算子

图像灰度函数 $f(x,y)$ 的梯度

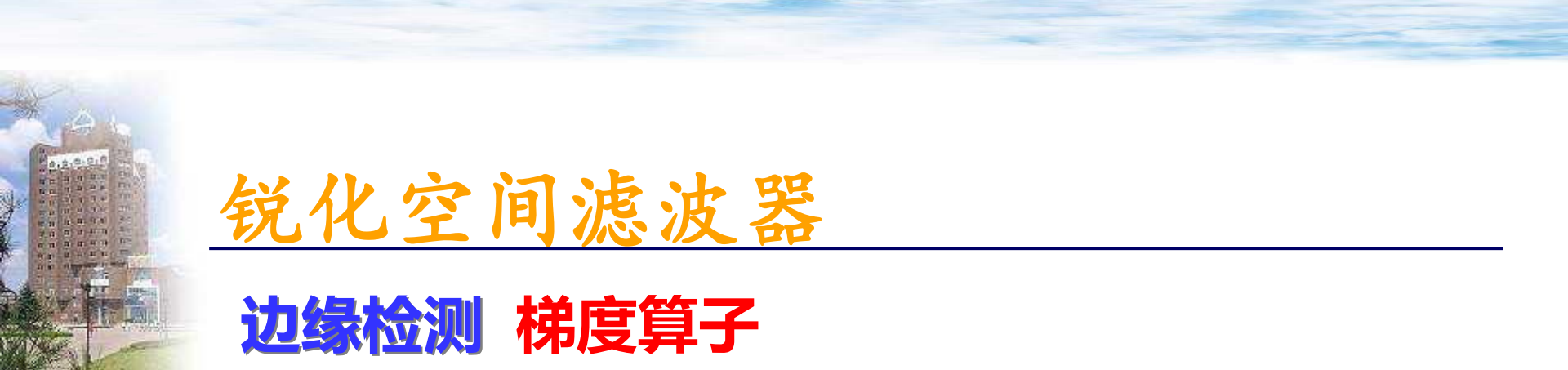
$$\nabla = \frac{\partial}{\partial x} \mathbf{i} + \frac{\partial}{\partial y} \mathbf{j}$$

$$\nabla f(x, y) = \frac{\partial f(x, y)}{\partial x} \mathbf{i} + \frac{\partial f(x, y)}{\partial y} \mathbf{j}$$



$$\left\{ \begin{array}{l} |\nabla f(x, y)| = \sqrt{\left(\frac{\partial f(x, y)}{\partial x}\right)^2 + \left(\frac{\partial f(x, y)}{\partial y}\right)^2} \\ \theta = \tan^{-1} \left( \frac{\partial f(x, y)}{\partial y} / \frac{\partial f(x, y)}{\partial x} \right) \end{array} \right.$$





# 锐化空间滤波器

---

## 边缘检测 梯度算子

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

### □ 关于梯度幅度的一些结论:

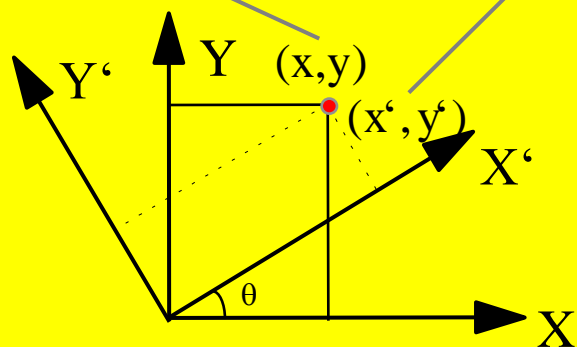
- 梯度幅度是一个各向同性的微分算子
- 梯度幅度等于 $f(x, y)$ 沿梯度方向上的最大变化率

# 锐化空间滤波器

## 边缘检测 梯度算子

□ 梯度幅度是一个各向同性的微分算子

$$|\nabla f(x, y)| = \sqrt{\left(\frac{\partial f(x, y)}{\partial x}\right)^2 + \left(\frac{\partial f(x, y)}{\partial y}\right)^2} \longleftrightarrow |\nabla f(x', y')| = \sqrt{\left(\frac{\partial f(x', y')}{\partial x'}\right)^2 + \left(\frac{\partial f(x', y')}{\partial y'}\right)^2}$$

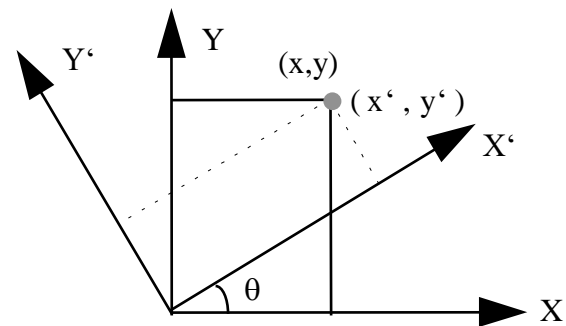


$$\begin{cases} x = x' \cos \theta - y' \sin \theta \\ y = x' \sin \theta + y' \cos \theta \end{cases}$$



# 锐化空间滤波器

## 边缘检测 梯度算子



$$\begin{cases} x = x' \cos \theta - y' \sin \theta \\ y = x' \sin \theta + y' \cos \theta \end{cases}$$

$$\frac{\partial f}{\partial x'} = \frac{\partial f}{\partial x} \frac{\partial x}{\partial x'} + \frac{\partial f}{\partial y} \frac{\partial y}{\partial x'} = \frac{\partial f}{\partial x} \cos \theta + \frac{\partial f}{\partial y} \sin \theta$$

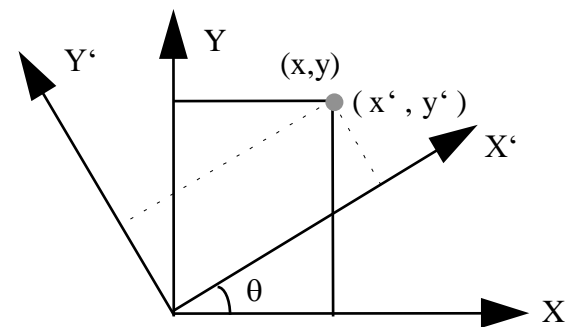
$$\frac{\partial f}{\partial y'} = \frac{\partial f}{\partial x} \frac{\partial x}{\partial y'} + \frac{\partial f}{\partial y} \frac{\partial y}{\partial y'} = \frac{\partial f}{\partial x} (-\sin \theta) + \frac{\partial f}{\partial y} \cos \theta$$

$$\begin{aligned} & \left( \frac{\partial f}{\partial x'} \right)^2 + \left( \frac{\partial f}{\partial y'} \right)^2 \\ &= \left( \frac{\partial f}{\partial x} \cos \theta + \frac{\partial f}{\partial y} \sin \theta \right)^2 + \left( \frac{\partial f}{\partial x} (-\sin \theta) + \frac{\partial f}{\partial y} \cos \theta \right)^2 \\ &= \left( \frac{\partial f}{\partial x} \right)^2 + \left( \frac{\partial f}{\partial y} \right)^2 \quad \Rightarrow \quad |\nabla f(x', y')| = |\nabla f(x, y)| \end{aligned}$$

# 锐化空间滤波器

## 边缘检测 梯度算子

□ 梯度幅度等于 $f(x,y)$ 沿梯度方向上的最大变化率



$$\begin{cases} x = x' \cos \theta - y' \sin \theta \\ y = x' \sin \theta + y' \cos \theta \end{cases}$$

$f(x,y)$ 在任意方向 $\theta$ 上的变化率

$$\frac{\partial f}{\partial x'} = \frac{\partial f}{\partial x} \frac{\partial x}{\partial x'} + \frac{\partial f}{\partial y} \frac{\partial y}{\partial x'} = \frac{\partial f}{\partial x} \cos \theta + \frac{\partial f}{\partial y} \sin \theta$$

记 $f(x,y)$ 最大变化率所在的方向为 $\theta_M$ ，则 $\theta_M$ 可用求极值的方法如下得到:

$$\frac{d}{d\theta} \left( \frac{\partial f}{\partial x'} \right) = -\frac{\partial f}{\partial x} \sin \theta + \frac{\partial f}{\partial y} \cos \theta = 0$$

➡  $f(x,y)$ 最大变化率所在方向为梯度方向

$$\theta_M = \tan^{-1} \left( \frac{\partial f(x,y)}{\partial y} / \frac{\partial f(x,y)}{\partial x} \right)$$

# 锐化空间滤波器

## 边缘检测 梯度算子

□ 梯度幅度等于 $f(x,y)$ 沿梯度方向上的最大变化率

$$\cos \theta_M = \frac{1}{\sqrt{1 + \tan^2 \theta_M}} = \frac{\frac{\partial f}{\partial x}}{\sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}}$$

$$\sin \theta_M = \frac{1}{\sqrt{1 - \cos^2 \theta_M}} = \frac{\frac{\partial f}{\partial y}}{\sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}}$$

$$\Rightarrow \left(\frac{\mathcal{F}}{\partial x}\right)_{\text{Max}} = \frac{\mathcal{F}}{\partial x} \Big|_{\theta=\theta_M} = \frac{\mathcal{F}}{\partial x} \cos \theta_M + \frac{\mathcal{F}}{\partial y} \sin \theta_M = \sqrt{\left(\frac{\mathcal{F}}{\partial x}\right)^2 + \left(\frac{\mathcal{F}}{\partial y}\right)^2}$$



# 锐化空间滤波器

## 边缘检测 梯度算子

□ 对于数字图像，可用一阶差分运算代替相应的一阶微分运算。设沿水平和垂直两个方向的一阶差分分别由 $\Delta u$ 和 $\Delta v$ 表示，则可以如下定义三种数字梯度幅度

$$|\nabla_1 f| = \sqrt{\Delta_u f^2 + \Delta_v f^2}$$

$$|\nabla_2 f| = |\Delta_u f| + |\Delta_v f|$$

$$|\nabla_3 f| = \text{Max}\{|\Delta_u f|, |\Delta_v f|\}$$

□ 将以上三种数字梯度幅度和各种一阶差分算子进行组合，可得到不同的边缘梯度算子。

{ Sobel梯度算子  
Priwitt梯度算子





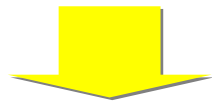
# 锐化空间滤波器

## 边缘检测 Canny算子

**问题：**边缘检测思想简单、易行，但在图像中存在噪声的情况下上述简单方法经常遇到困难。

**原因：**图像中通常混有高频噪声。这些高频噪声和边缘类似，均表现为图像灰度发生跳变，导致检测到的边缘点集中包含大量由噪声引起的虚假边缘点。

**解决方案：**平滑滤波 + 边缘检测



**怎样设计相关的滤波器和边缘检测算法以实现的边缘检测？**

**一个解答：CANNY算子**



# 锐化空间滤波器

---

## 边缘检测 Canny算子

### 最佳边缘检测的三个评价指标

- **好的信噪比** 一个好的边缘检测算子将非边缘点判为边缘点，或者反过来将边缘点判为非边缘点的概率要尽可能低。
- **好的定位性能** 一个好的边缘检测算子检测出的边缘点应尽可能在实际边缘的中心。
- **对单一边缘有唯一的响应** 一个好的边缘检测算子对图像中的单一边缘产生多个响应的概率要尽可能低。



# 锐化空间滤波器

---

## 边缘检测 Canny算子

### ● 平滑滤波 + 边缘检测

设输入图像为 $f(x,y)$ ，所使用的平滑滤波器为 $h(x,y)$ ，则平滑滤波后的输出图像可表为

$$s(x,y) = f(x,y) * h(x,y)$$

计算输出图像的梯度，有：

$$\begin{aligned}\nabla s(x,y) &= \nabla (f(x,y) * h(x,y)) = \nabla \left( \iint f(s,t)h(x-s,y-t)dsdt \right) \\ &= \iint f(s,t)\nabla h(x-s,y-t)dsdt = f(x,y) * \nabla h(x,y)\end{aligned}$$

计算其幅值，并同事先设定的阈值进行比较，即可得到所求边缘。



# 锐化空间滤波器

## 边缘检测 Canny算子

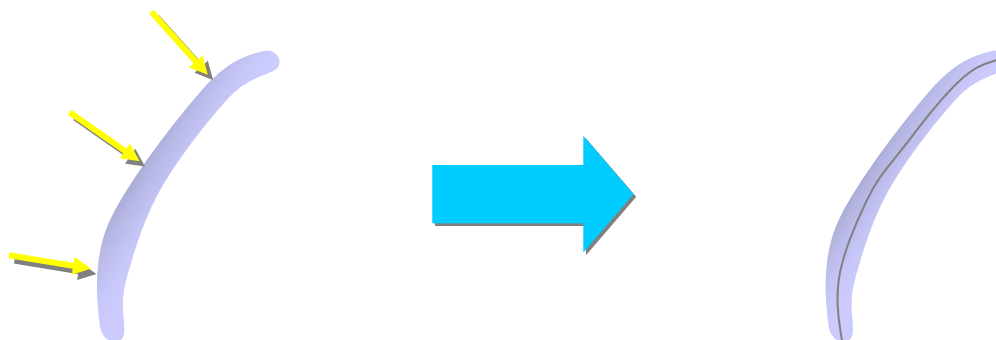
存在的问题(1):

高斯平滑滤波后 → 边缘变模糊

→ 得到的边缘具有一定宽度

对策：非极大点抑制

在垂直于边缘的方向（即梯度方向）上，对相邻点的梯度幅度进行比较，仅保留具有较大梯度幅度的点。





# 锐化空间滤波器

## 边缘检测 Canny算子

存在的问题(2):

高斯平滑滤波后 ———→ 剩余噪声, 图像的细纹理

—————→ 虚假边缘

对策: 非极大点抑制后采用双阈值法予以消除

设置两个阈值:  $T_1$ ,  $T_2 = 2 T_1$

采用非极大点抑制后, 分别得到两幅边缘图像:  $e_1$ ,  $e_2$ .

其中,  $e_1$  真边缘较全, 但虚假边缘也多;

$e_2$  虚假边缘少, 但真边缘不全, 出现断裂现象.

以 $e_2$ 图像为基础, 寻找连续的图像边缘; 当到达连续边缘的端点处时, 在 $e_1$ 图像相应点的邻域内寻找可连接的边缘点, 直到填满断裂处的间隙为止。

# 锐化空间滤波器

## 边缘检测 二阶微分算子

拉普拉斯算子  $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$

图像的二阶微分

$$\nabla^2 f(x, y) = \frac{\partial^2 f(x, y)}{\partial x^2} + \frac{\partial^2 f(x, y)}{\partial y^2}$$

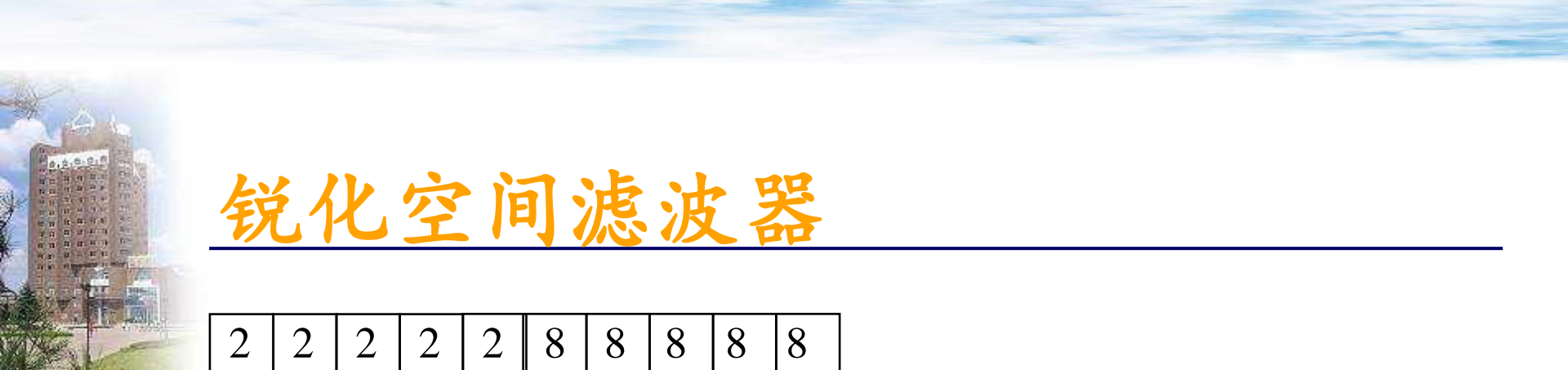
图像二阶微分的差分近似

$$\begin{aligned} \nabla^2 f(i, j) &= \{[f(i+1, j) - f(i, j)] - [f(i, j) - f(i-1, j)]\} \\ &\quad + \{[f(i, j+1) - f(i, j)] - [f(i, j) - f(i, j-1)]\} \\ &= f(i+1, j) + f(i-1, j) + f(i, j+1) + f(i, j-1) - 4f(i, j) \end{aligned}$$

根据边缘检测判据2，  
图像中的边缘点可通过  
检测图像二阶微分的  
过零点得到。

0	1	0
1	-4	1
0	1	0





# 锐化空间滤波器

2	2	2	2	2	8	8	8	8	8
2	2	2	2	2	8	8	8	8	8
2	2	2	2	2	8	8	8	8	8
2	2	2	2	2	8	8	8	8	8
2	2	2	2	2	8	8	8	8	8
2	2	2	2	2	8	8	8	8	8

0	0	0	6	-6	0	0	0
---	---	---	---	----	---	---	---

2	2	2	2	2	5	8	8	8	8
2	2	2	2	2	5	8	8	8	8
2	2	2	2	2	5	8	8	8	8
2	2	2	2	2	5	8	8	8	8
2	2	2	2	2	5	8	8	8	8
2	2	2	2	2	5	8	8	8	8

0	0	0	3	0	-3	0	0
---	---	---	---	---	----	---	---



# 锐化空间滤波器

---

## 边缘检测 二阶微分算子

采用拉普拉斯算子的差分形式时会遇到以下问题:

由于数字图像的离散特性, 使得实际图像边缘处未必出现所期望的零值。

对策: 可利用边缘点的过零特性, 通过检测其两侧的正、负峰值来定出实际的边缘位置, 但是处理起来显得较烦琐。

实际做法: 采用将  $\nabla^2 f$  的绝对值同给定阈值  $T$  进行比较的方法定出图像中的边缘点, 即把  $|\nabla^2 f| \geq T$  的像素看作是边缘点。

代价: 不得不经常接受双像素宽的边缘。同时由于  $|\nabla^2 f|$  对孤立噪声点等高频噪声很敏感, 使得方法的抗干扰能力较差。



# 锐化空间滤波器

---

## 边缘检测 二阶微分算子

提高拉普拉斯算子抗高频噪声能力的措施

在实际进行边缘检测之前，首先对图像进行平滑滤波处理以抑制掉图像中的高频噪声。

设输入图像为 $f(x,y)$ ，所使用的平滑滤波器为 $h(x,y)$ ，则输入图像经该平滑滤波器滤波后的输出可表为

$$s(x,y) = f(x,y) * h(x,y)$$

从而，

$$\begin{aligned}\nabla^2 s(x,y) &= \nabla^2 (f(x,y) * h(x,y)) = \nabla^2 \left( \iint f(s,t) h(x-s, y-t) ds dt \right) \\ &= \iint f(s,t) \nabla^2 h(x-s, y-t) ds dt = f(x,y) * \nabla^2 h(x,y)\end{aligned}$$



# 锐化空间滤波器

## 边缘检测 LOG算子

### 平滑滤波器的选择原则:

- 和引入平滑滤波的初衷一致，所选用的平滑滤波函数  $h(x,y)$  应该能够起到抑制高频噪声的作用。
- 所选用的平滑滤波函数  $h(x,y)$  应该至少是二次可微的。
- 平滑滤波器的滤波功能是可调的，并能保证灰度图像经滤波后，其均值保持不变。

可以证明，平滑滤波函数取高斯函数形式时满足要求。

$$h(x,y) = G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{1}{2\sigma^2} (x^2 + y^2)\right)$$



# 锐化空间滤波器

---

边缘检测

LOG算子

LOG算子

→  $\nabla^2 s(x, y) = f(x, y) * \nabla^2 G(x, y, \sigma)$

LOG滤波器的显著特点:

- 其中的平滑滤波函数  $G$  具有将图像模糊化的功能，可有效地消除图像中尺度小于空间参数  $\sigma$  的强度变化。
- 由于平滑滤波函数  $G$  在空域和空频域都是平滑和定域的，使得出现由它而引入的在原图像中没有的变化的可能性最小。
- 由于  $\nabla^2$  是各向同性的微分算子，用它可避免沿各个方向的方向导数的计算问题，从而可减少计算开销。



# 锐化空间滤波器

## 边缘检测 LOG算子

$$\Rightarrow \nabla^2 s(x, y) = f(x, y) * \nabla^2 G(x, y, \sigma)$$

图像中的边缘点集 $E(x, y)$ 可表为

$$\begin{aligned} E(x, y) &= \{(x, y, \sigma) \mid \nabla^2 (f(x, y) * G(x, y, \sigma)) = 0\} \\ &= \{(x, y, \sigma) \mid (f(x, y) * \nabla^2 G(x, y, \sigma)) = 0\} \end{aligned}$$

$\sigma$ 是可调因子。选用不同的 $\sigma$ ，可检测图像中以不同尺度出现的强度变化。 $\sigma$ 选得越大，检测到的边缘点越少，定位精度也越差。反之， $\sigma$ 选得越小，检测到的边缘点越多，定位精度也越高。



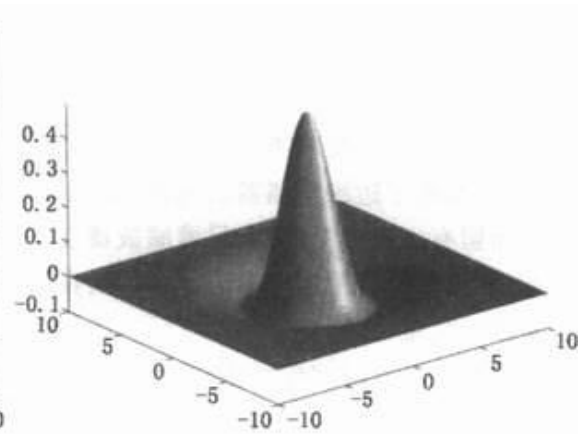
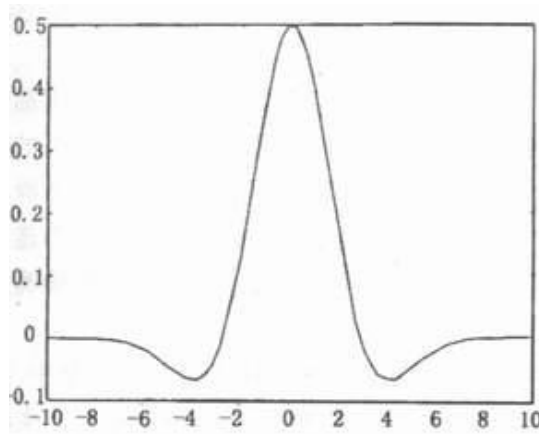


# 锐化空间滤波器

边缘检测

LOG算子

墨西哥草帽算子



与人类视觉视网膜感受野的  
响应模型相符合

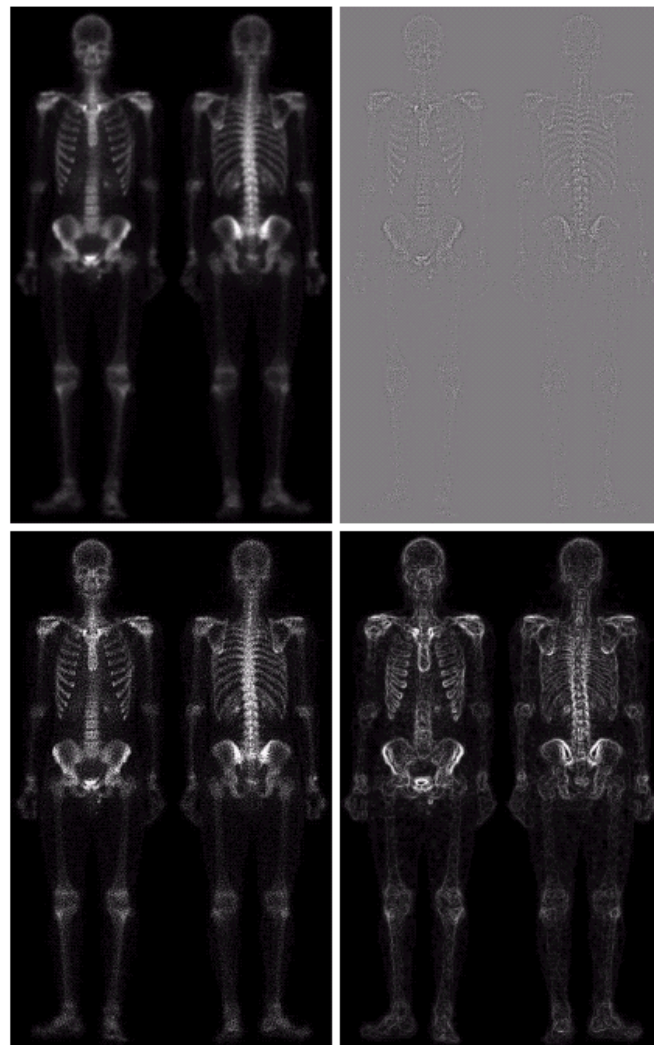
# 锐化空间滤波器

## 图像锐化

把各种互补的图像增强技术结合起来，实践复杂的增强任务  
举例：骨骼图像处理：

目的：突出骨骼的更多细节

策略：先用拉氏算子突出图像中的小细节，然后用梯度法突出其边缘，平滑过的梯度图像用于屏蔽拉氏图像，最后用灰度变换扩展灰度动态范围。





# 锐化空间滤波器

## 图像锐化

原图:月球北极 拉普拉斯滤波后的图像

a b  
c d

FIGURE 3.40

(a) Image of the North Pole of the moon.

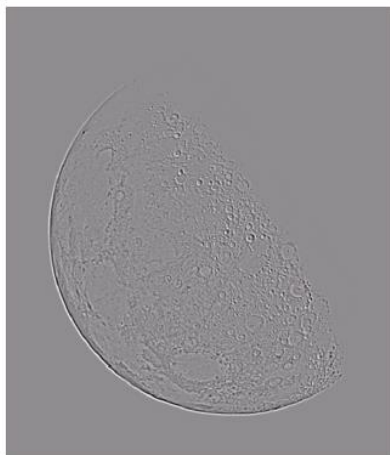
(b) Laplacian-filtered image.

(c) Laplacian image scaled for display purposes.

(d) Image enhanced by using Eq. (3.7-5). (Original image courtesy of NASA.)



3×3, 中心点为-8的掩膜



原始图像 + 拉普拉斯的结果

标定的图像