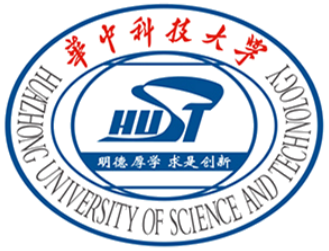


华中科技大学

HUAZHONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

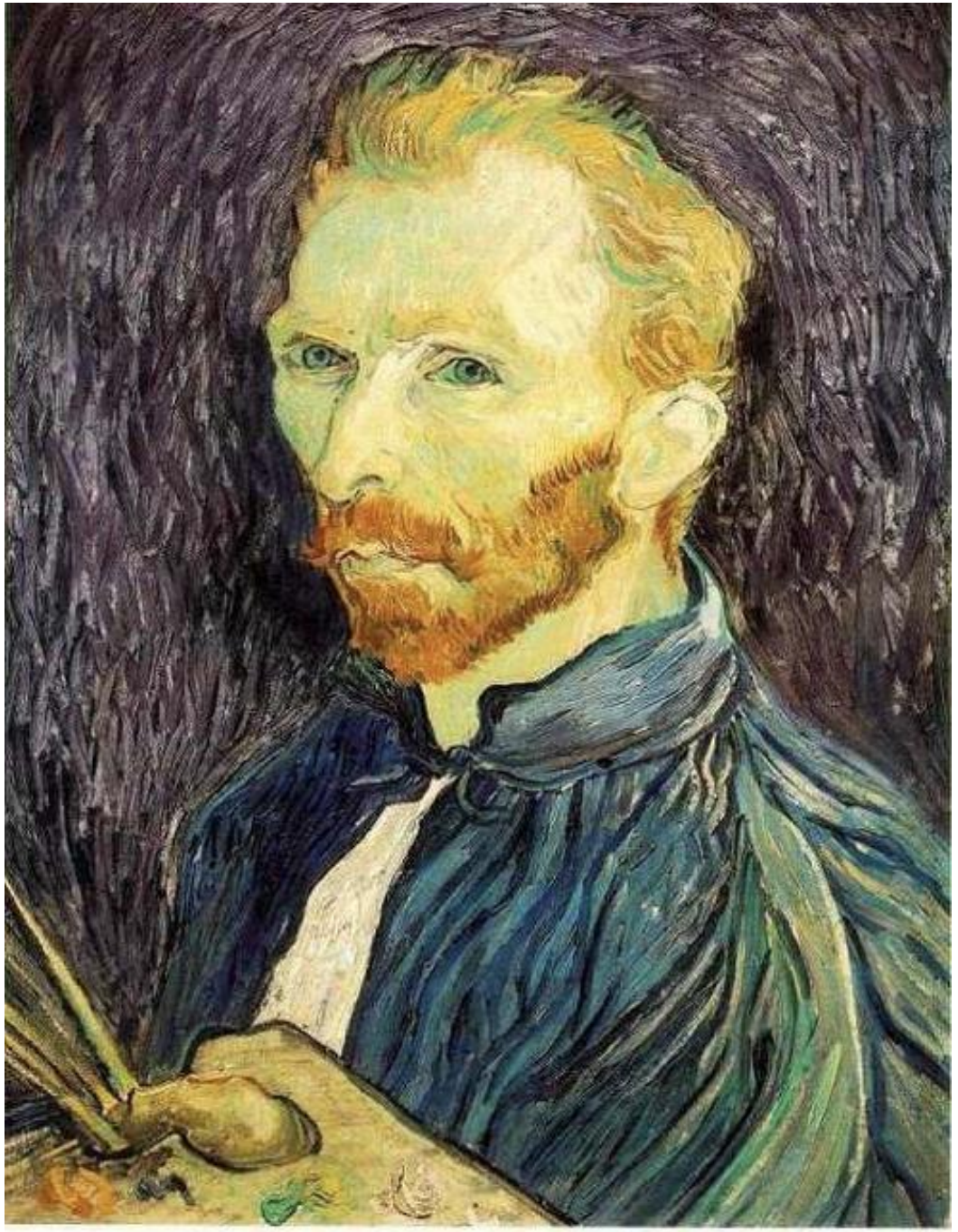
计算机视觉导论

任课教师：李贤芝

计算机科学与技术学院

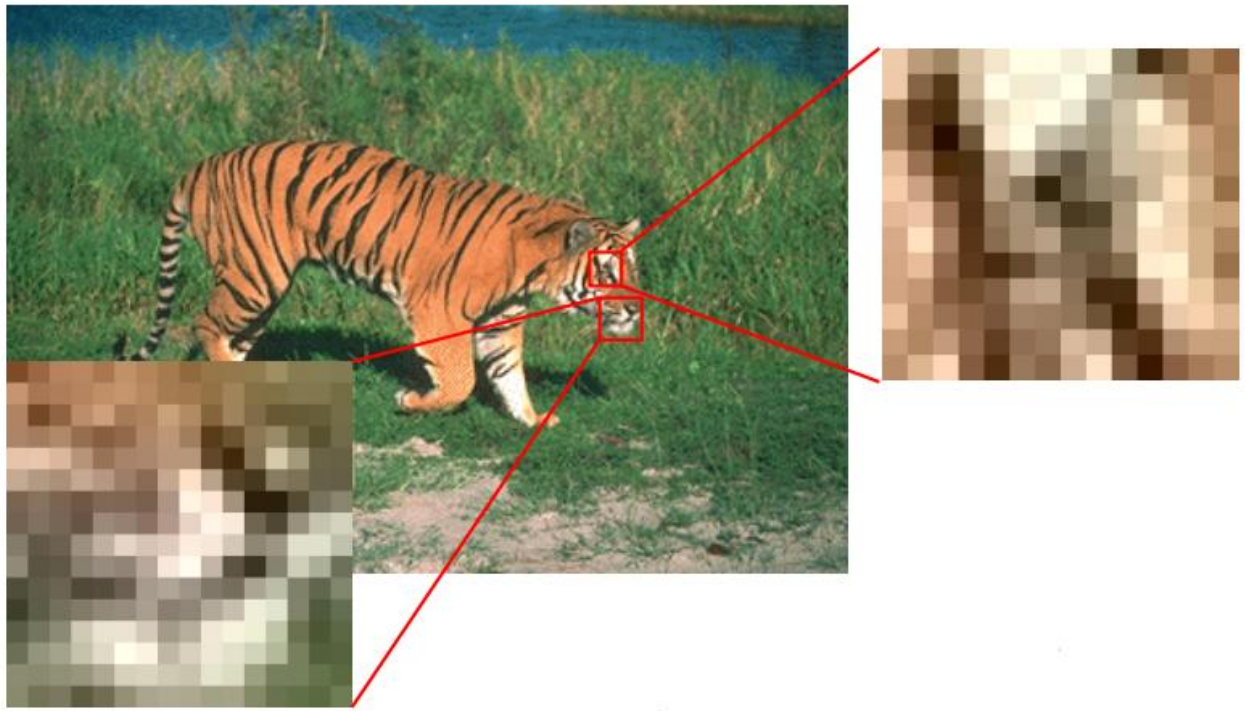
数字图像处理基础

- 图像采样与量化
- 图像滤波与卷积
- 图像放缩



计算机中的数字图像

- 数字图像是由有限数量的元素组成的，每个元素都有一个特定的位置和幅值，这些元素称为**像素**。可以定义一个二维函数$f(x,y)$，其中(x,y)是空间坐标，$f(x,y)$是点(x,y)的幅值。
  - 在灰度图像中，每个像素只用一个值表示。
  - 在彩色图像中，每个像素用多个值表示。



灰度图像

- 每个像素只有一个采样颜色的图像。通常有256个灰度等级，255代表全白，0代表全黑。

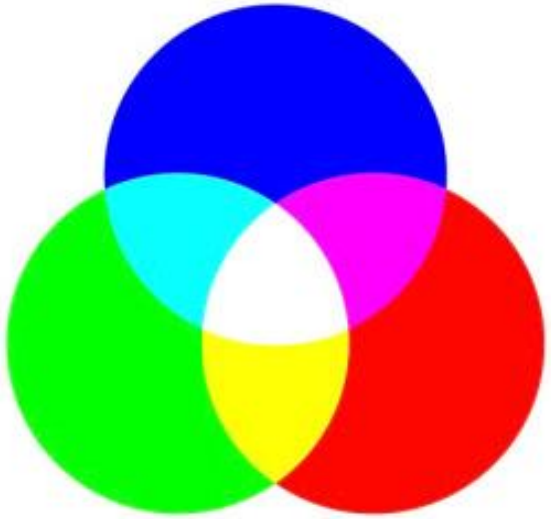
灰度图像

vs.

黑白图像

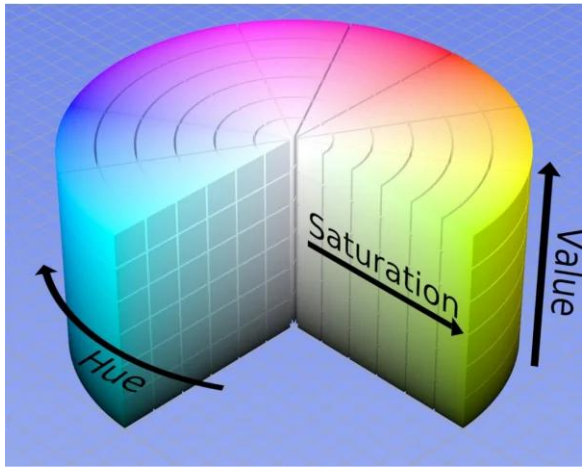
彩色图像的常用颜色模型

- 在彩色图像中，每个像素用多个值表示。



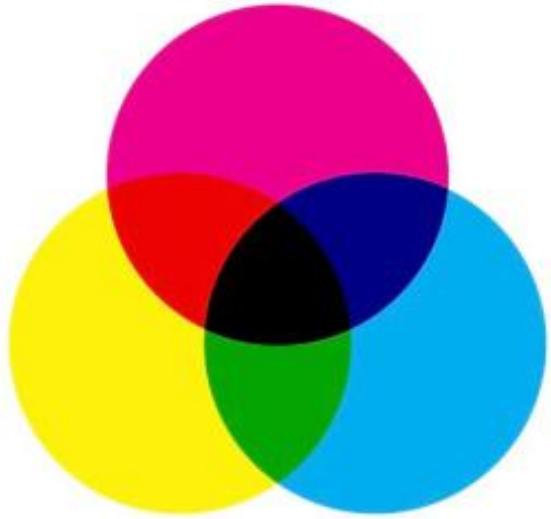
RGB颜色模型

- 最典型、最常用的三基色模型
- 电视、摄像机、彩色扫描仪都是根据RGB模型工作的
- 不直观，不符合人对颜色的认知



HSV颜色模型

- H：色调；S：饱和度；V：亮度值
- 与人类对颜色的感知更接近

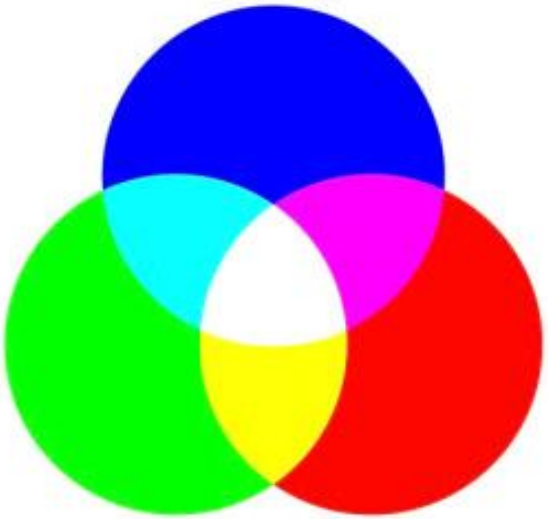


CMYK颜色模型

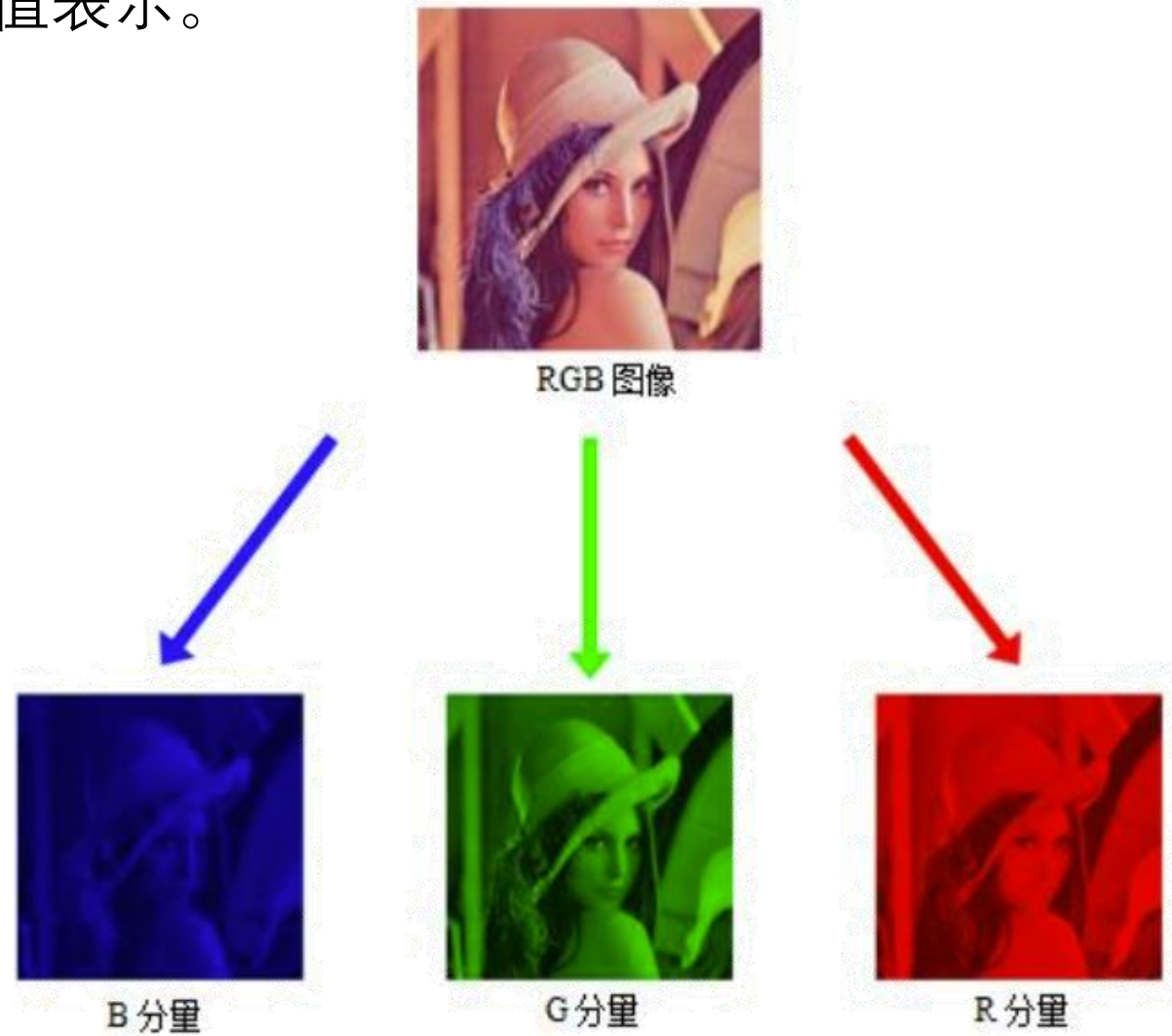
- 主要用于彩色打印，图像处理中几乎不用

彩色图像的常用颜色模型

- 在彩色图像中，每个像素用多个值表示。

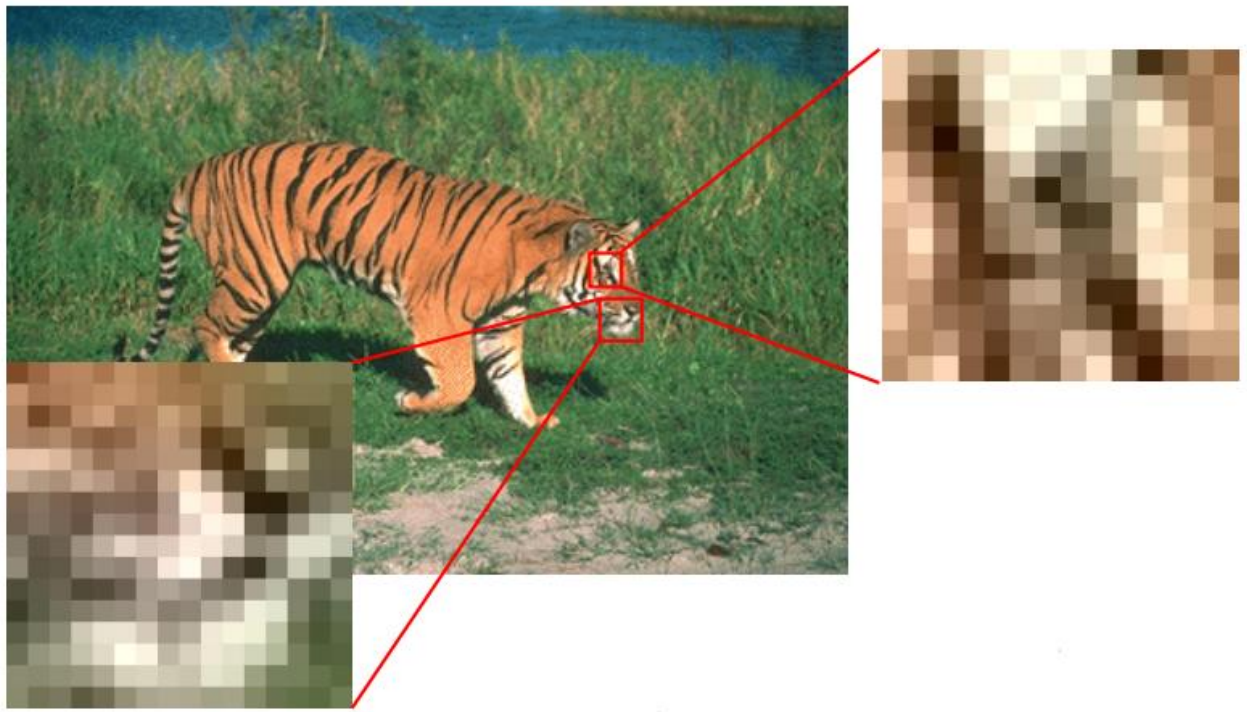


RGB颜色模型



计算机中的数字图像

- 数字图像是由有限数量的元素组成的，每个元素都有一个特定的位置和幅值，这些元素称为**像素**。可以定义一个二维函数$f(x,y)$，其中(x,y)是空间坐标，$f(x,y)$是点(x,y)的幅值。
  - 在灰度图像中，每个像素只用一个值表示。
  - 在彩色图像中，每个像素用多个值表示。

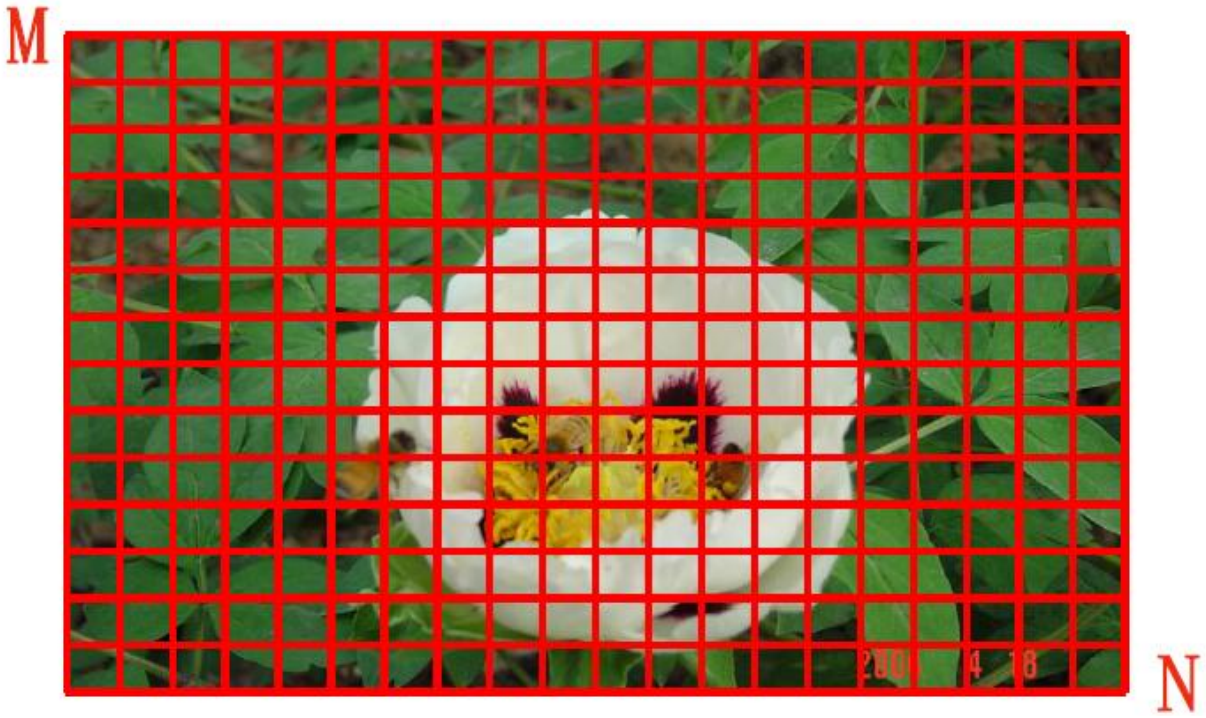


- 传感器获取的图像是平面上的**连续函数**。为了产生一副数字图像，需要把连续的感知数据转换为数字形式，这包括**采样 (Sampling)** 和**量化 (Quantization)**两个基本步骤。

图像的采样

- 图像采样：图像空间坐标的数字化 [**可以理解为坐标点的选取**]

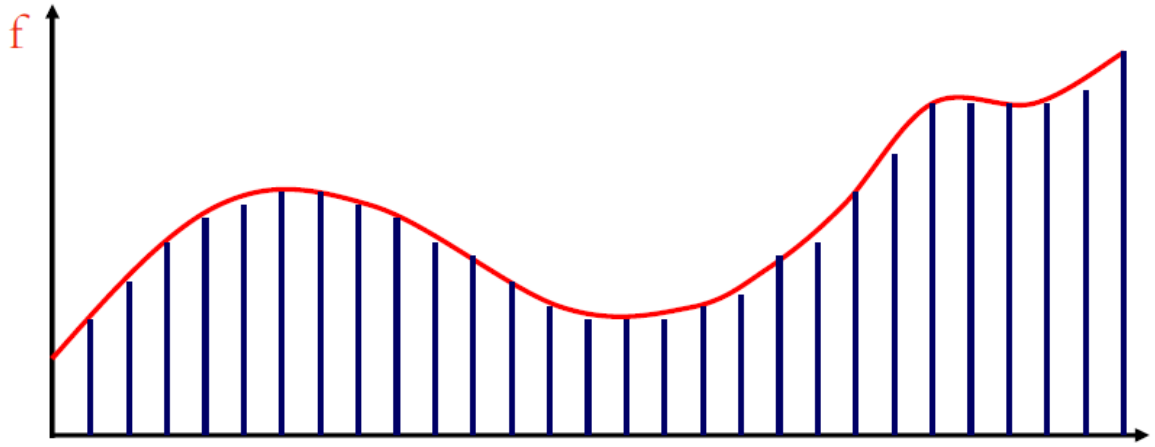
  如果对同一幅图像采样间隔越小：
  - 所得像素越多，图像的空间分辨率越高，细节越清晰
  - 文件尺寸大，处理时间长，对设备要求高

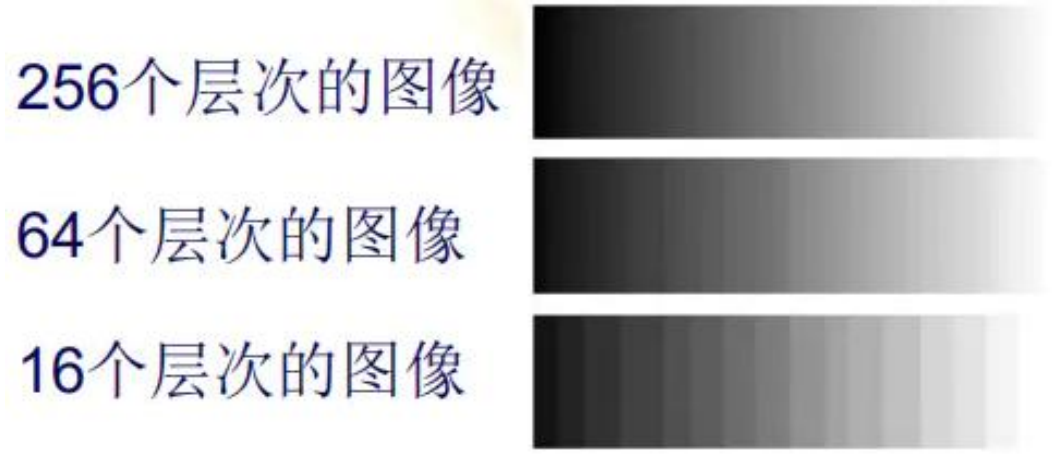


图像的量化

- 图像量化：图像函数值 (灰度值) 的数字化 [**可以理解为幅值的数字化**]

  - 均匀量化：将灰度级值等间隔分档取整
  - 随着灰度等级的减少，图像的细节信息在逐渐损失，视觉效果越差。





图像的采样与量化

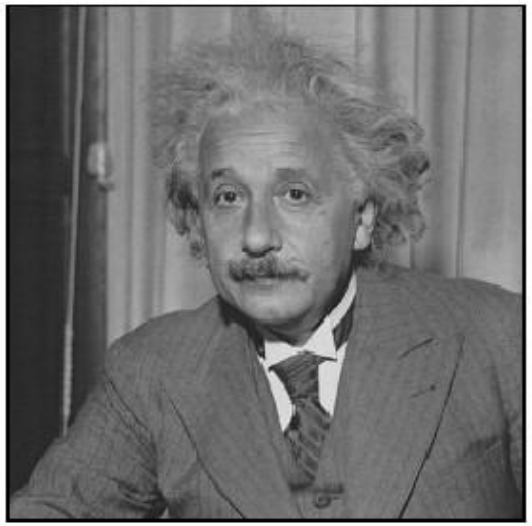
- 不同采样点数对图像质量的影响：

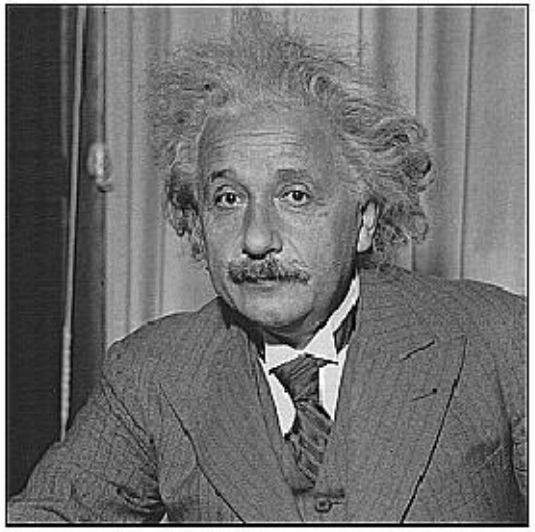
(a): 512×512　(b): 256×256　(c): 128×128
(d): 64×64　(e): 32×32　(f):16×16

- 不同量化级别对图像质量的影响：

(a): 256　(b): 64　(c): 16
(d): 8　(e): 4　(f): 2

图像滤波与卷积





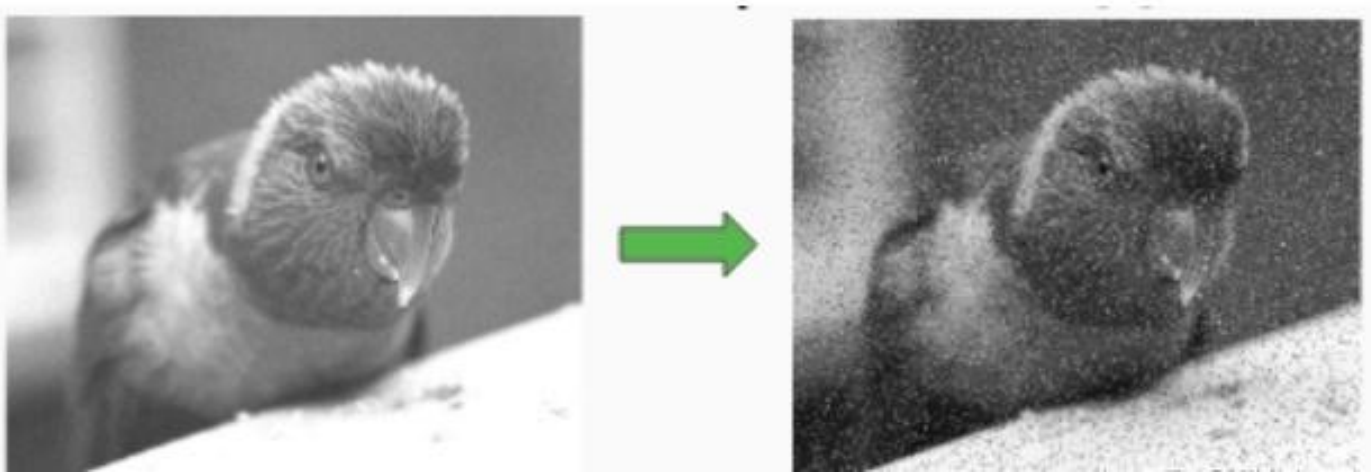
平滑/锐化图像

边缘检测

模板匹配

图像滤波与卷积的动机 – 降噪

- 在图像产生、传输和复制过程中，常常会因为多方面原因而被噪声干扰，降低了图像质量，这就需要对图像进行一定的处理以减小这些缺陷带来的影响。
- 噪声在图像上常表现为引起较强视觉效果的孤立像素点或者像素块。
- 对于数字图像，噪声表现为或大或小的极值，这些极值通过加减作用于图像像素的真实灰度值上，对图像造成亮、暗点干扰，降低了图像的视觉质量，影响图像复原、分割、特征提取、图像识别等后继工作的进行。

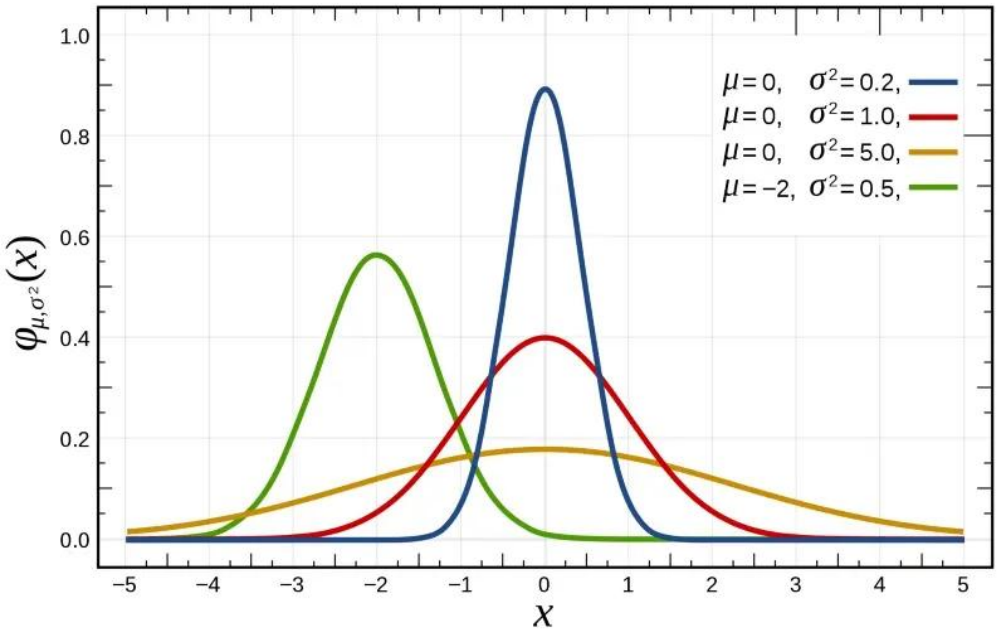


噪声类型

椒盐噪声：黑白像素的随机出现。

脉冲噪声：白色像素的随机出现。

高斯噪声：服从高斯分布的噪声。



Original

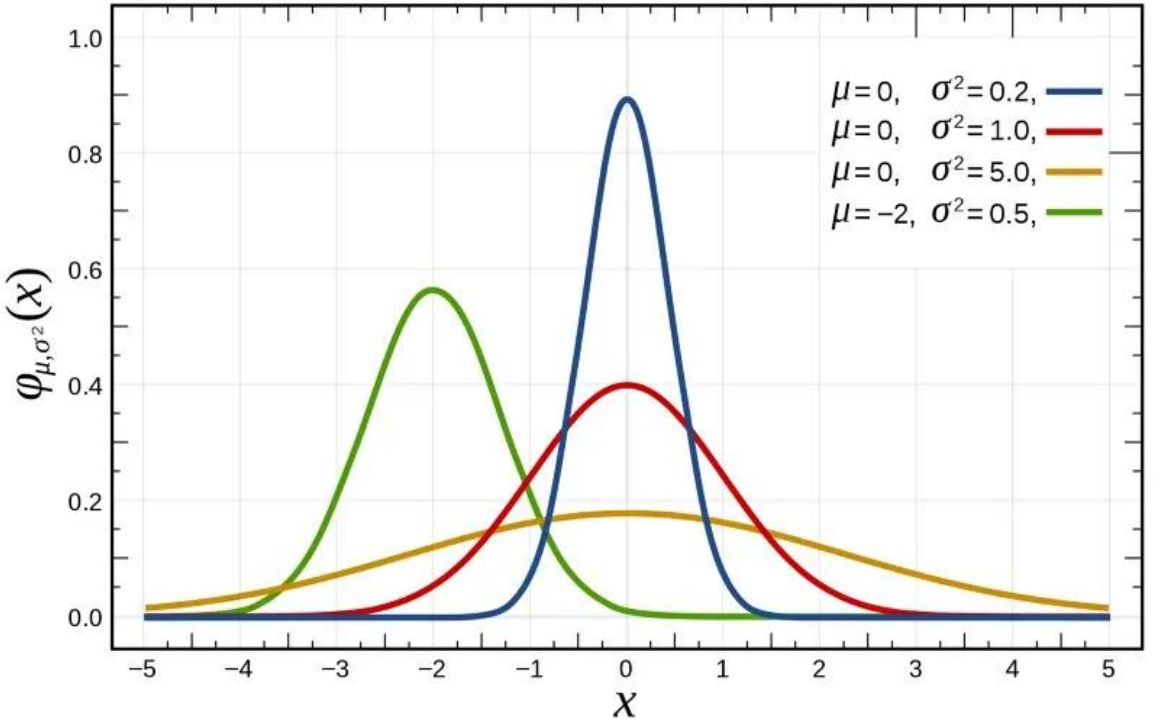
Salt and pepper noise

Impulse noise

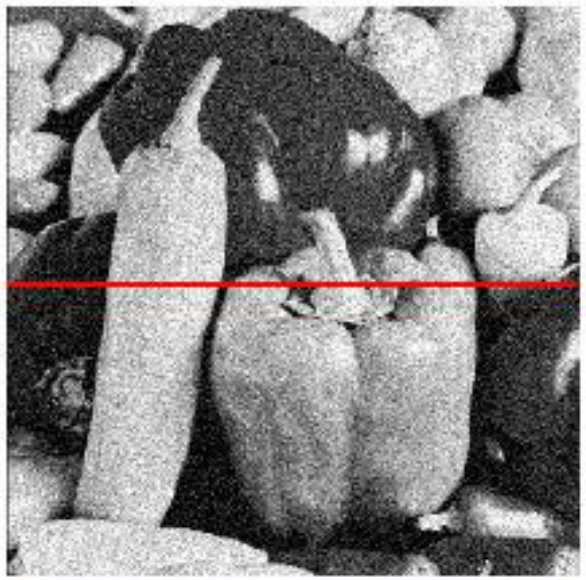
Gaussian noise

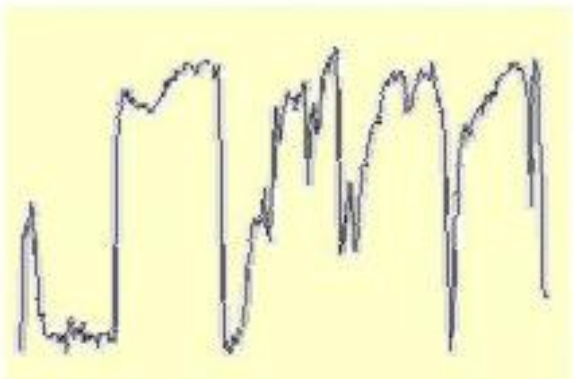
高斯分布：也叫正态分布（normal distribution），通常记作$X \sim N(\mu, \sigma^2)$。其中，μ是正态分布的均值，σ^2是正态分布的方差。$\mu = 0$，$\sigma^2 = 1$的正态分布被称作标准正态分布。

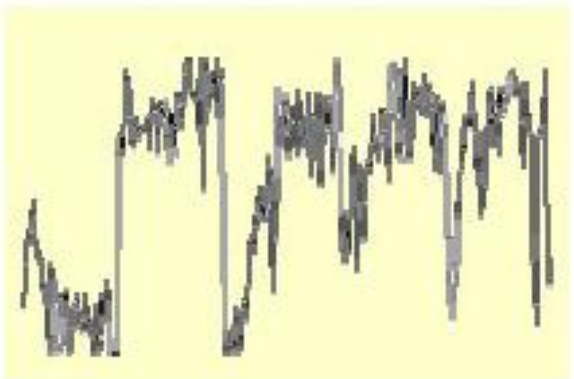
正态分布的概率密度函数显示为典型的钟形曲线。μ描述了分布的中心位置，即曲线对称轴所在位置；σ^2描述了随机变量相对于均值的偏离程度。



高斯噪声



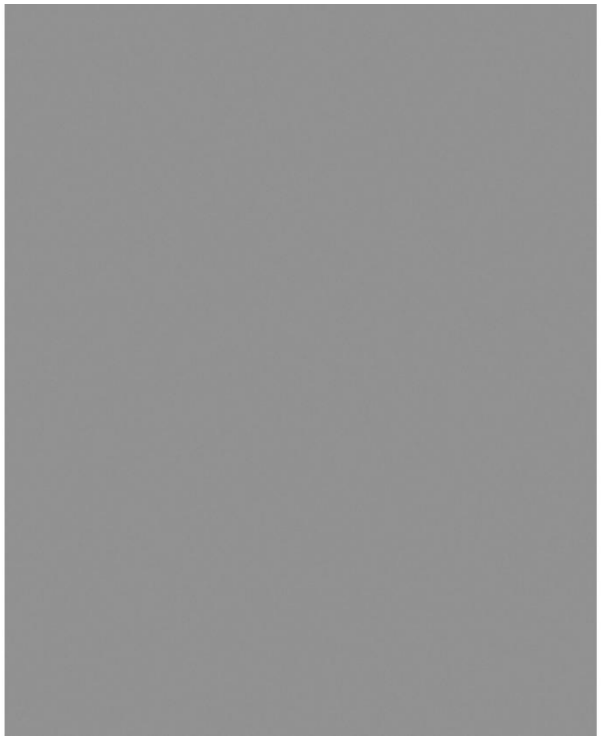




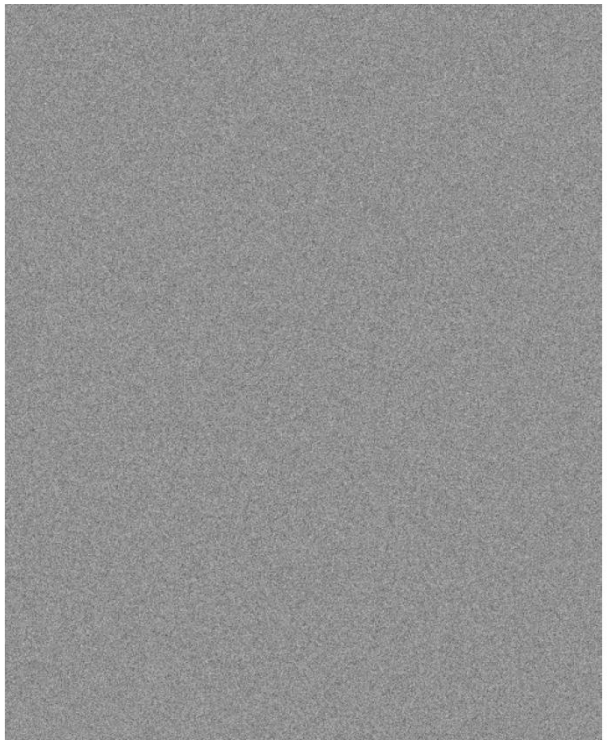
$$f(x,y) = \overbrace{\bar{f}(x,y)}^{\text{Ideal Image}} + \overbrace{\eta(x,y)}^{\text{Noise process}}$$

Gaussian i.i.d. ("white") noise:
$\eta(x,y) \sim \mathcal{N}(\mu,\sigma)$

高斯噪声图



VS.



$\sigma = 1$ $\sigma = 16$

高斯噪声加到原始图像上

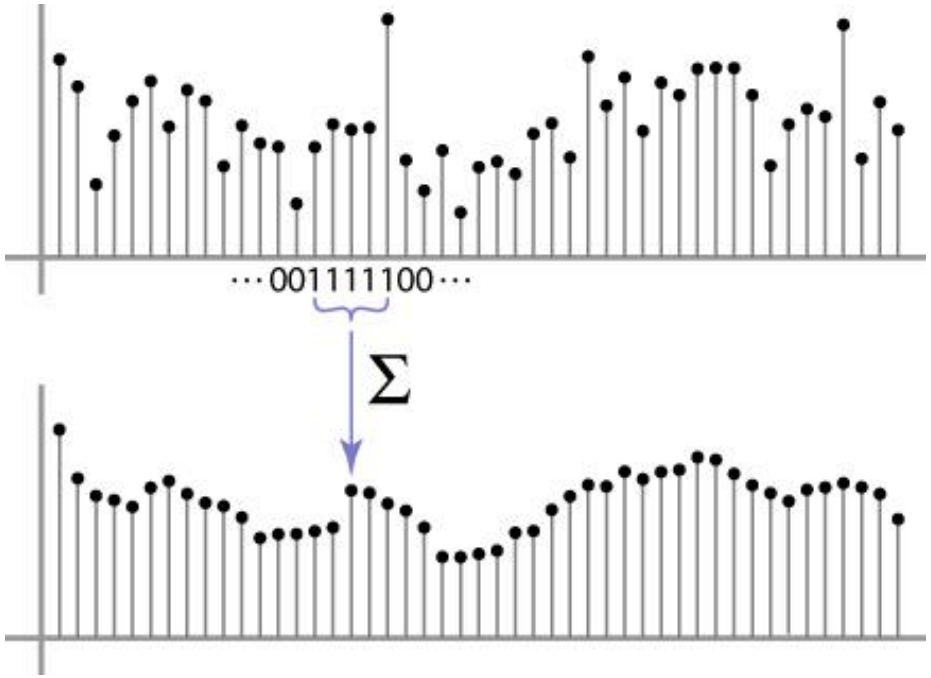
VS.

$\sigma = 1$ $\sigma = 16$

均值滤波

- 用每个像素和它周围像素计算出来的平均值替换图像中的每个像素。
- 消除尖锐噪声，实现图像平滑、模糊等功能。

一维



取周围五个位置的像素，
五个像素的权重为[1, 1, 1, 1, 1] / 5

二维：像素值相加后求平均

$F[x, y]$

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

$G[x, y]$

	0								

$F[x, y]$

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

$G[x, y]$

	0	10							

$F[x, y]$

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

$G[x, y]$

	0	10	20						

$F[x, y]$

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

$G[x, y]$

	0	10	20	30					

$F[x, y]$

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

$G[x, y]$

	0	10	20	30	30				

$F[x, y]$

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

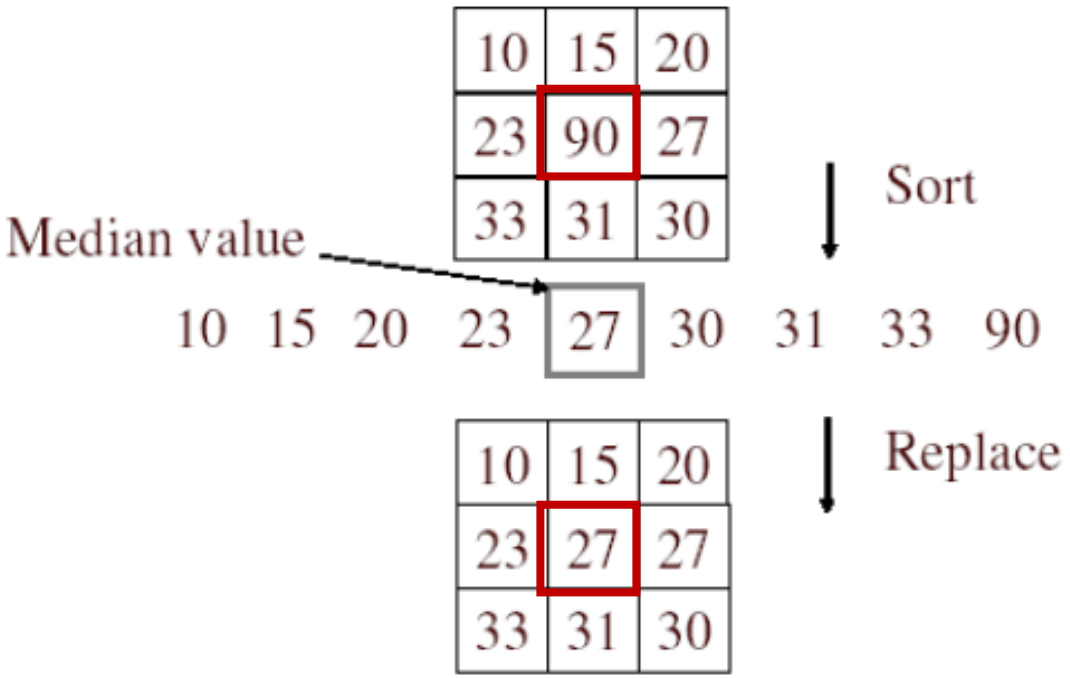
$G[x, y]$

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

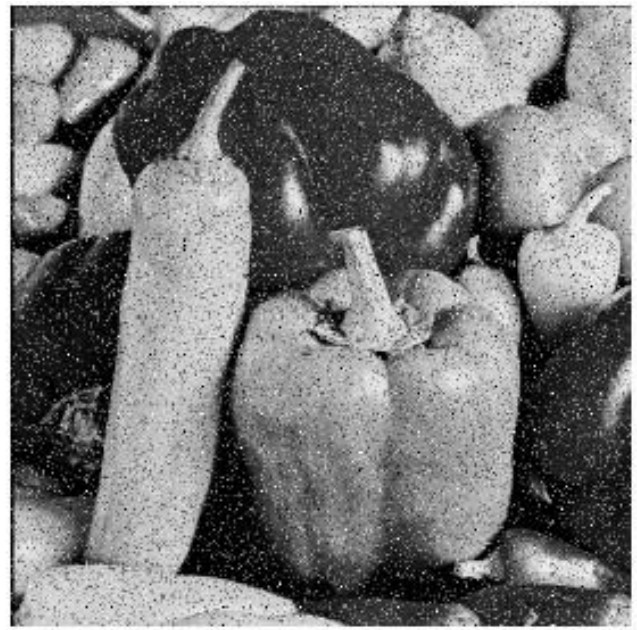
随着窗口的增大，图像越来越模糊!

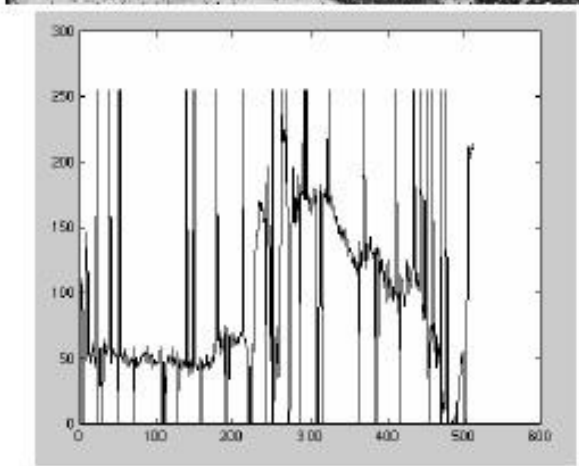
中值滤波

- 用每个像素和它周围像素的中值替换图像中的每个像素。
- 对脉冲噪声和椒盐噪声有良好的效果。

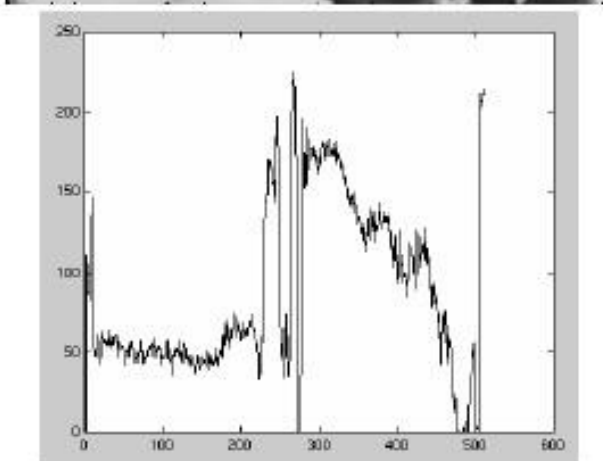


加了椒盐噪声的图像

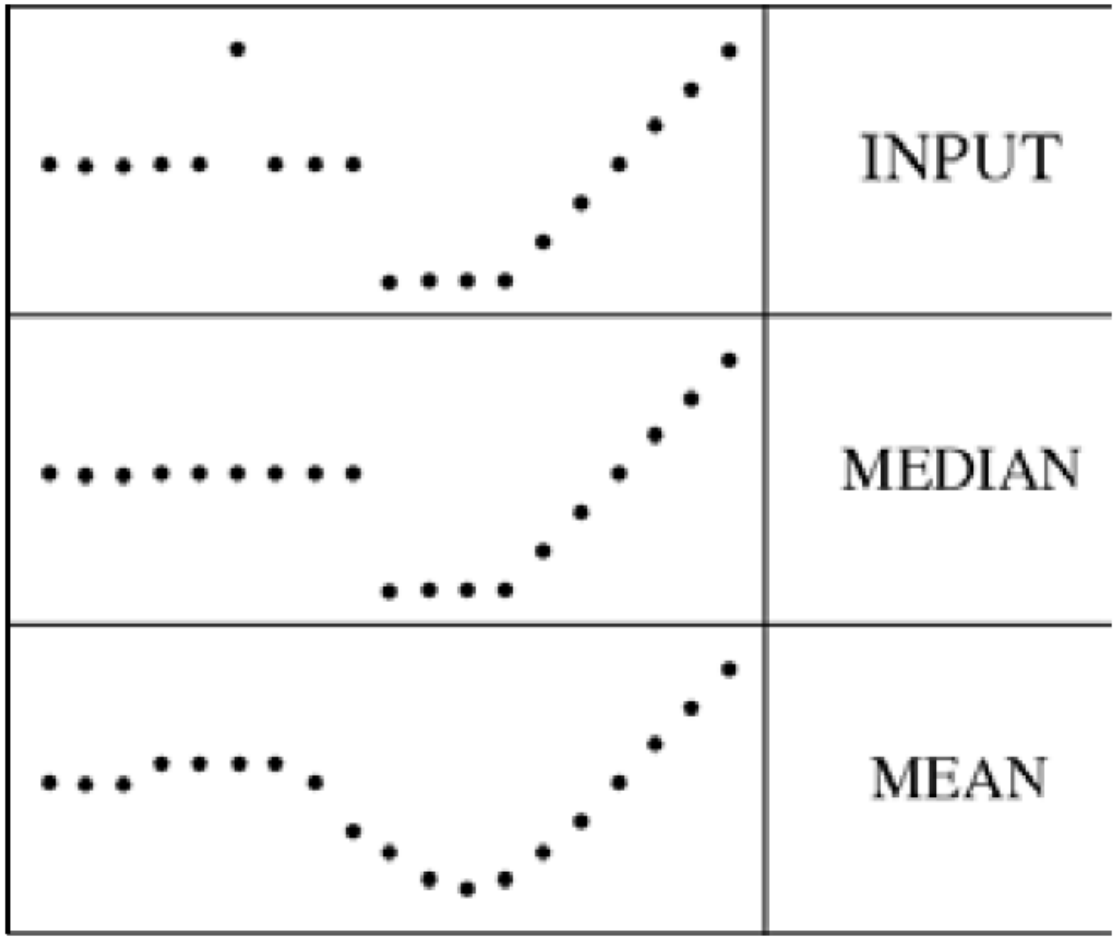




中值滤波后的图像



中值滤波可以有效保留边缘



高斯滤波

- 用每个像素和它周围像素的加权平均灰度值替换图像中的每个像素。距离中心像素越近，分配权重越大。
- 适用于消除高斯噪声。

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

$F[x, y]$

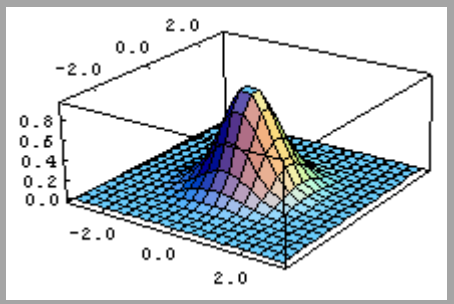
$$\frac{1}{16}$$

1	2	1
2	4	2
1	2	1

$H[u, v]$

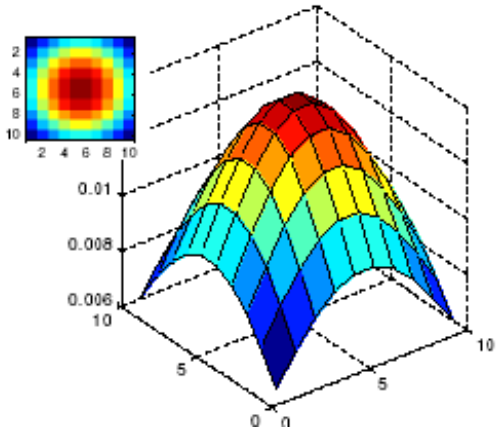
高斯滤波核是对高斯方法的模拟近似。

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

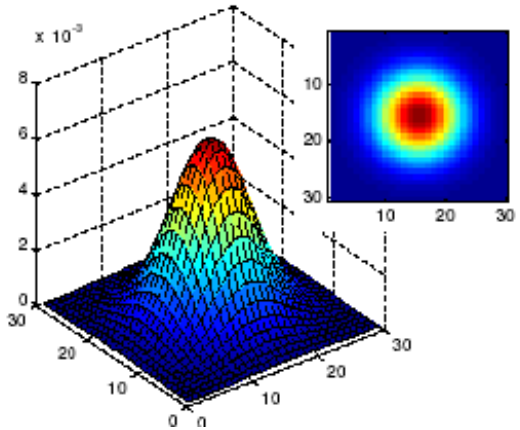


高斯滤波的参数

- 高斯滤波核的大小：
  σ固定，核越大图像越模糊。

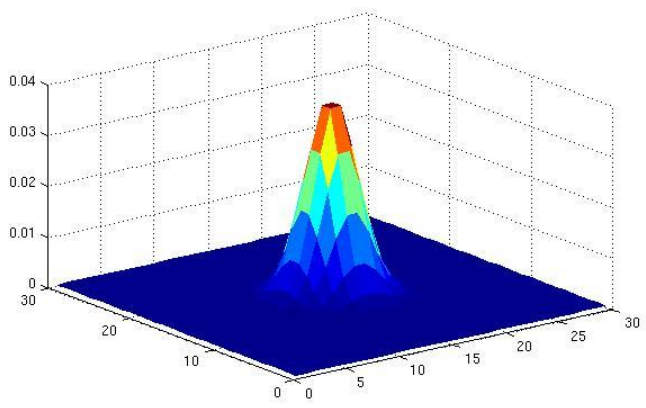


σ = 5
10 x 10

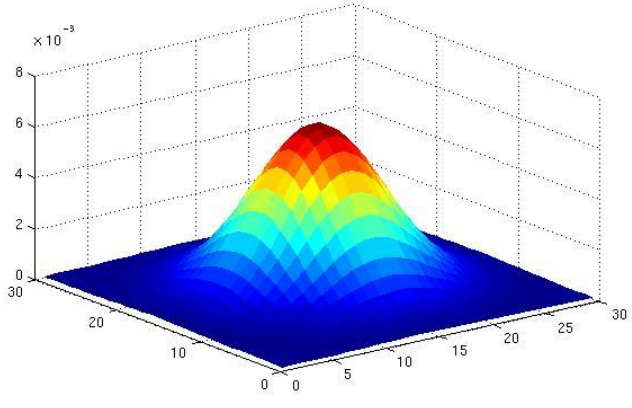


σ = 5
30 x 30

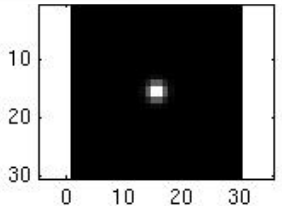
- 标准差σ的大小：
  核固定，σ越大图像越模糊。

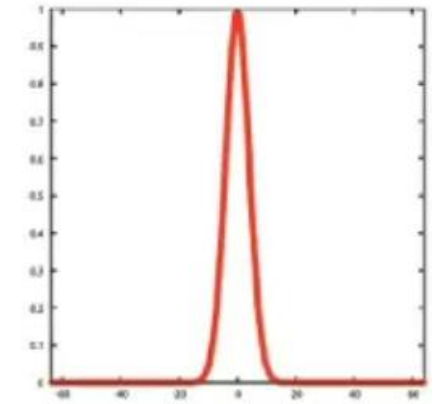


σ = 2
30 x 30

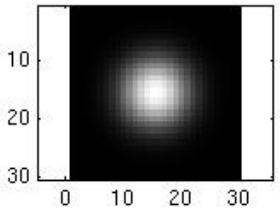


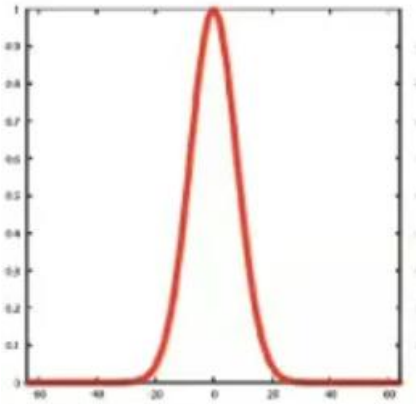
σ = 25
30 x 30





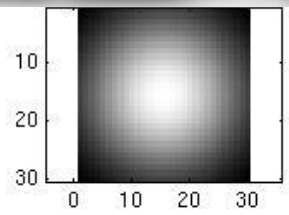
σ = 1
30 x 30

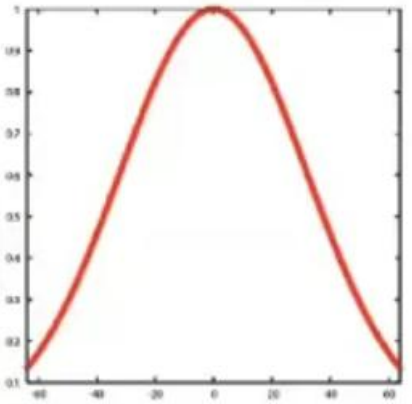




σ = 4
30 x 30

...





σ = 10
30 x 30

卷积

- 互相关：对图像矩阵和滤波矩阵按位进行逐个元素相乘再求和的操作。
- 卷积：先对滤波矩阵顺时针旋转180° (由下到上，由右到左)，而后再按位点乘求和。

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i-u,j-v]$$

↑

卷积核

注：深度学习中的卷积可以看作互相关运算，因为深度学习中的卷积核的参数是可学习的，无所谓翻转与否。

卷积

- 互相关：对图像矩阵和滤波矩阵按位进行逐个元素相乘再求和的操作。
- 卷积：先对滤波矩阵顺时针旋转180° (由下到上，由右到左)，而后再按位点乘求和。

$F[x,y]$

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

$*$

卷积核

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

$H[u,v]$

$=$

$G[x,y]$

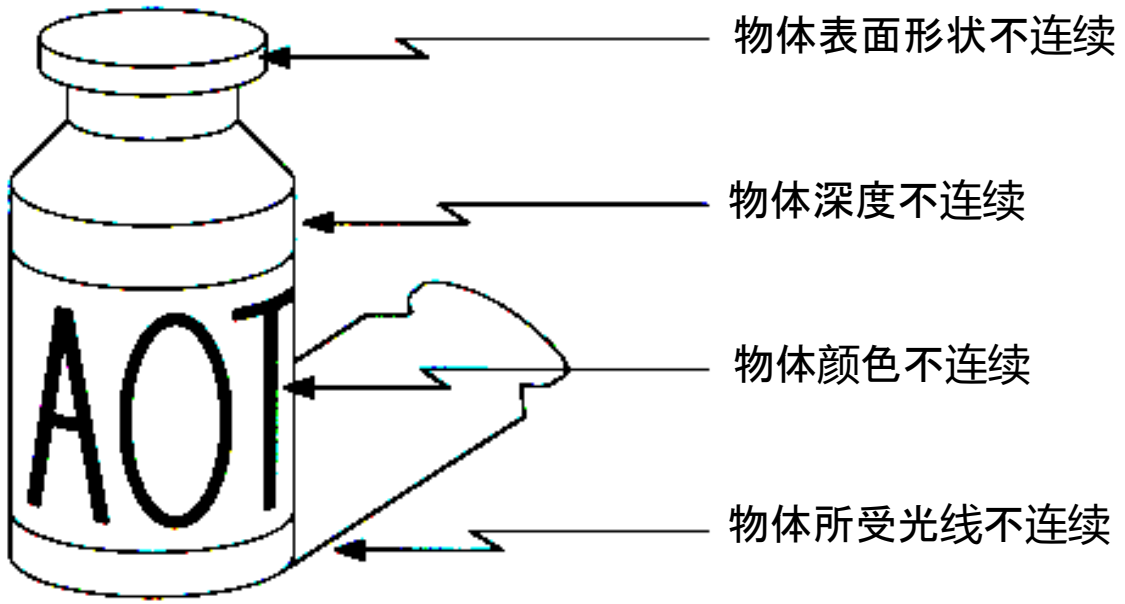
	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

图像滤波与卷积的动机 – 提取特征

- 一开始，是将滤波看作一种去除噪声的方式。
- 现在，考虑滤波器如何提取高层特征。
  - 将原始像素映射到某种表示，以便后续处理;
  - 减少数据量，丢弃冗余，保留有用的内容（如边缘信息）。

图像边缘检测

- 边缘是像素值发生突变的地方，也是图像中信息最集中的地方。图像边缘通常由多种因素引起:



如何判断一个像素是否属于边?

图像梯度

导数:

对于函数$f(x)$，它的一阶导数记为$f'(x)$，二阶导数记为$f''(x)$

$$f'(x) = \frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{f(x+\Delta x) - f(x)}{\Delta x}$$

$$f''(x) = \frac{df'(x)}{dx} = \lim_{\Delta x \to 0} \frac{f'(x+\Delta x) - f'(x)}{\Delta x}$$

$f'(x)$表示$f(x)$沿x方向的变化速率

$f''(x)$表示$f'(x)$沿x方向的变化速率

图像梯度

偏导数:

对于二元函数$f(x,y)$，对变量x的偏导数记为$f_x'(x,y)$，对变量y的偏导数记为$f_y'(x,y)$

$$f_x'(x,y)=\frac{\partial f(x,y)}{\partial x}=\lim_{\Delta x\to 0}\frac{f(x+\Delta x,y)-f(x,y)}{\Delta x}$$
$$f_y'(x,y)=\frac{\partial f(x,y)}{\partial y}=\lim_{\Delta y\to 0}\frac{f(x,y+\Delta y)-f(x,y)}{\Delta y}$$

梯度:

对于二元函数$f(x,y)$，梯度 $gradient=\nabla f(x,y)=[f_x'(x,y),f_y'(x,y)]$

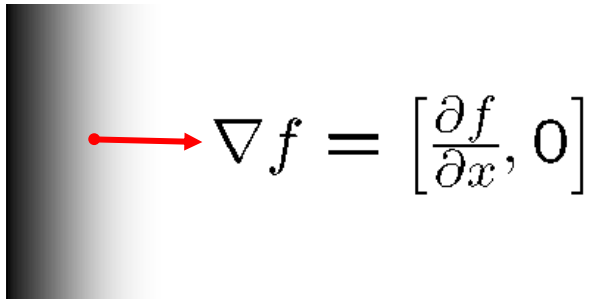
梯度是一个向量，表示函数$f(x,y)$在点(x,y)处沿着该方向（梯度方向）变化最快，变化率最大（梯度的模）

图像梯度

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

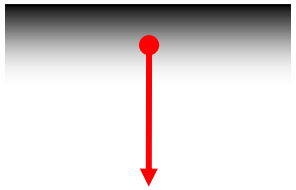
梯度指向强度最快速变化的方向，图像梯度计算的是图像像素值变化的速度:

- 对于图像的边缘部分，因为其灰度值变化较大，所以梯度值也较大
- 对于图像中比较平滑的部分，因为其灰度值变化较小，所以梯度值也较小



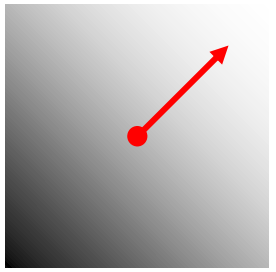
$$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]$$

水平方向强度变化最快



$$\nabla f = \left[0, \frac{\partial f}{\partial y}\right]$$

垂直方向强度变化最快



$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

45°方向强度变化最快

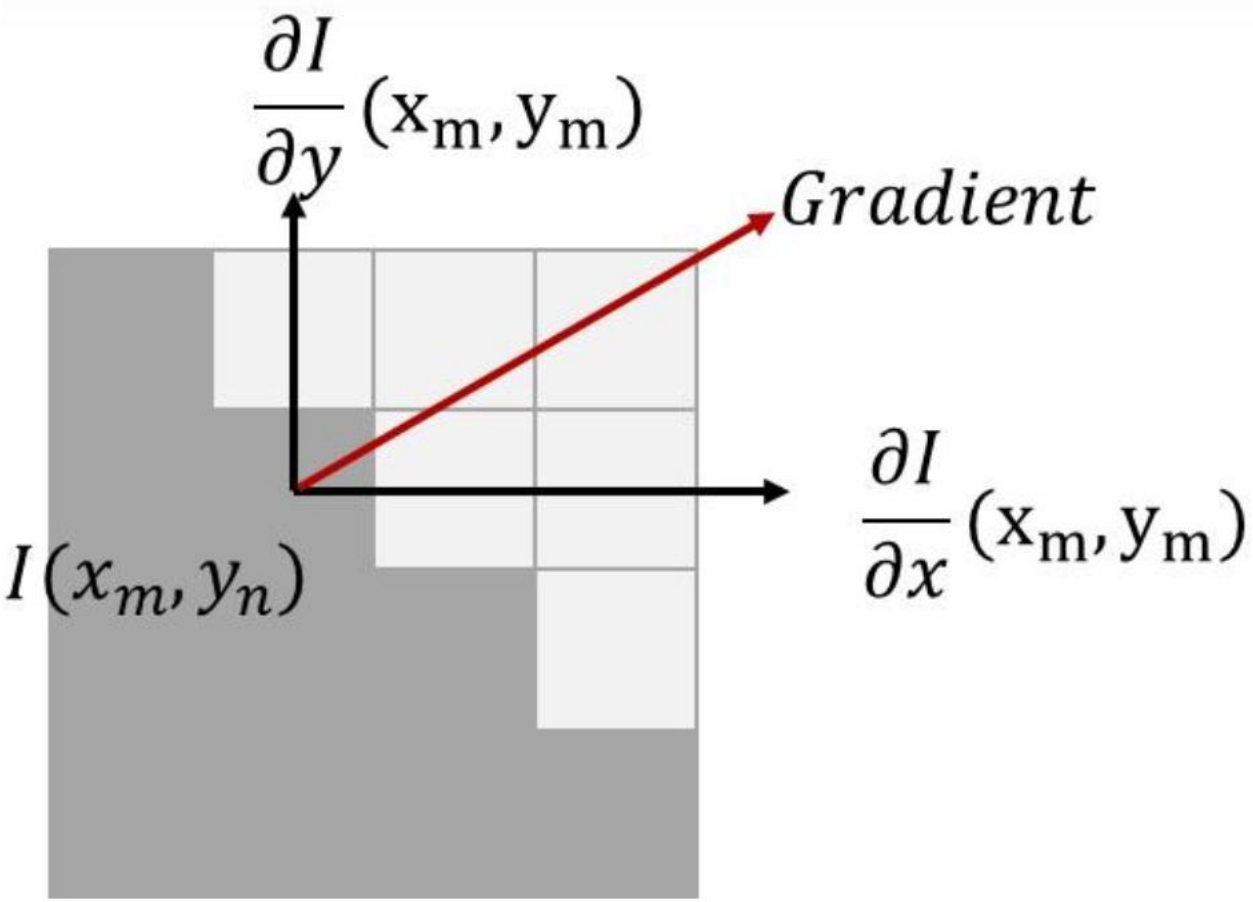
如何对一个数字图像求梯度?

离散梯度

- 有限差分:

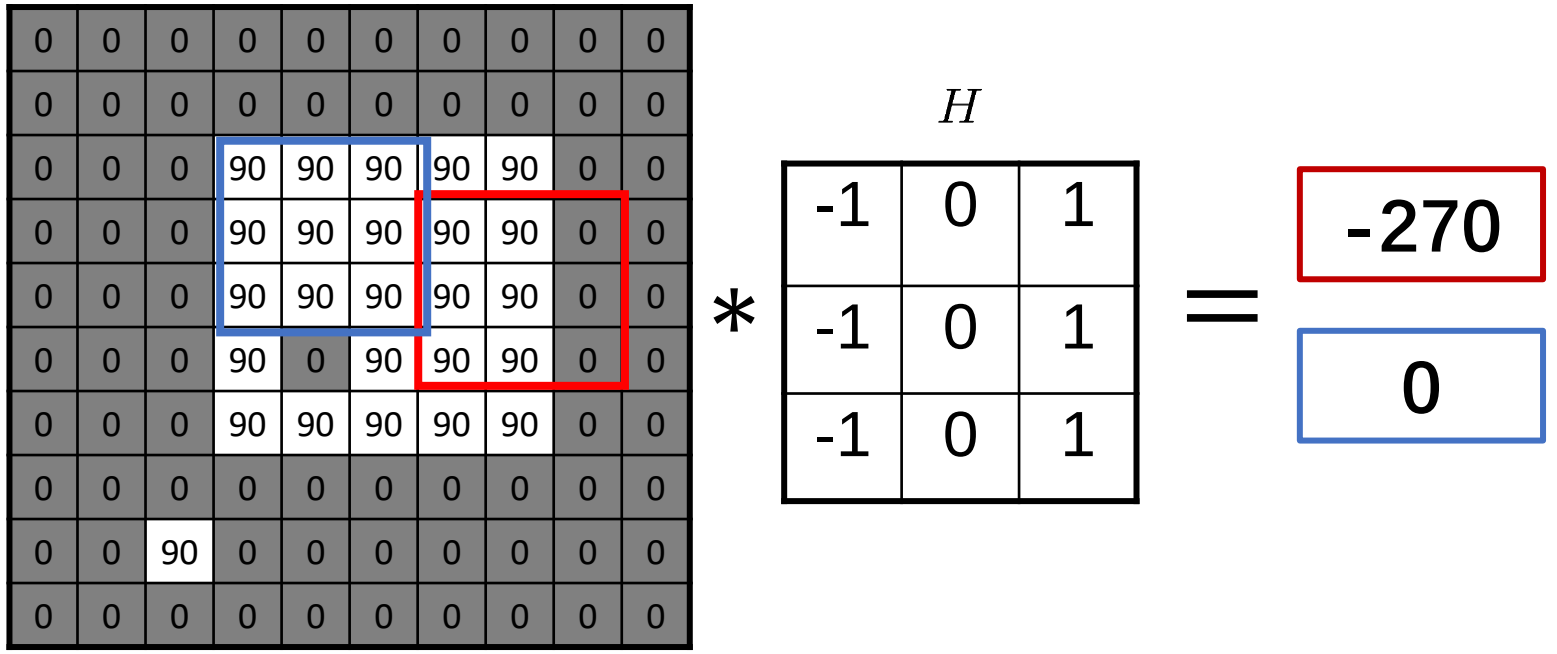
$$\frac{\partial f}{\partial x}[x,y] \approx F[x+1,y] - F[x-1,y]$$

$$\frac{\partial f}{\partial y}[x,y] \approx F[x,y+1] - F[x,y-1]$$



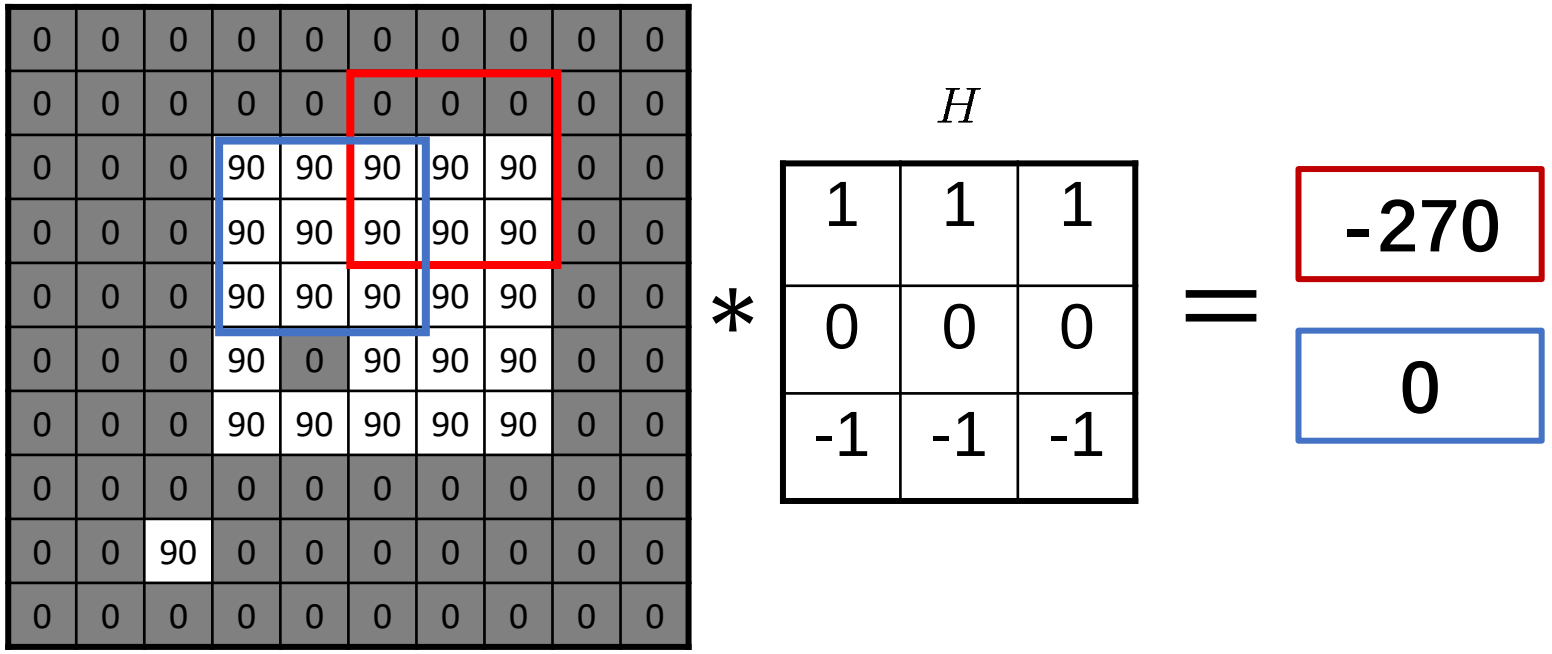
离散梯度

- 例：计算水平方向的差分



离散梯度

- 例：计算垂直方向的差分



离散梯度

- 有限差分:

$$\frac{\partial f}{\partial x}[x,y] \approx F[x+1,y] - F[x-1,y] \qquad \frac{\partial f}{\partial y}[x,y] \approx F[x,y+1] - F[x,y-1]$$

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

$*$

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

水平差分

or

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

垂直差分

边缘检测本质上就是一种滤波，区别在于滤波器的选择，滤波的规则是完全一样的!

对不同方向进行求导的滤波器

Prewitt: $M_x =$

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

; $M_y =$

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

Sobel: $M_x =$

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

; $M_y =$

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

Roberts: $M_x =$

0	1
-1	0

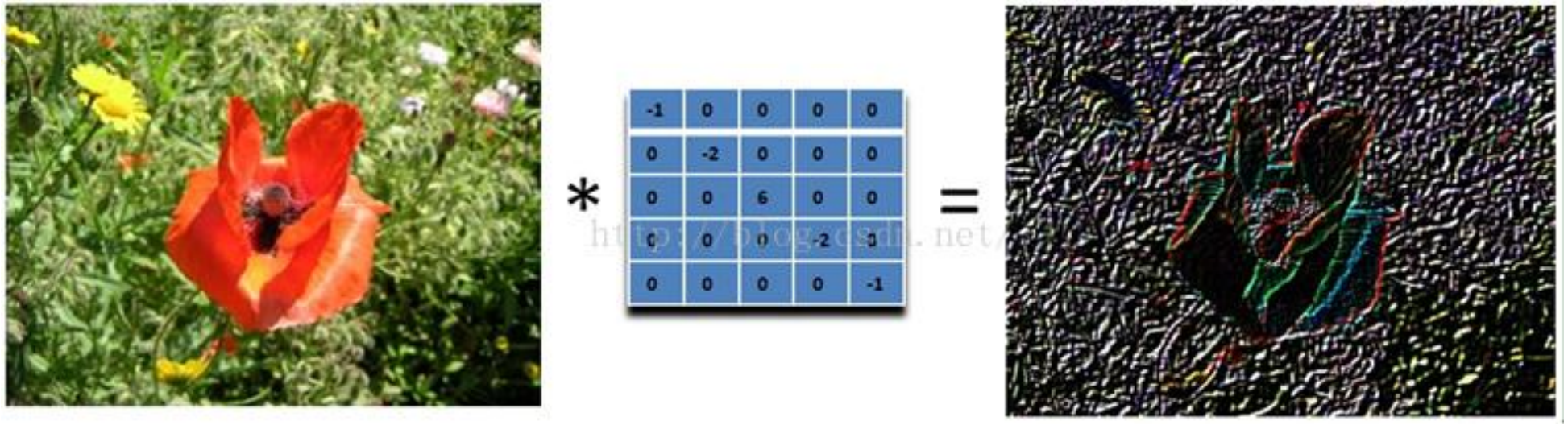
; $M_y =$

1	0
0	-1

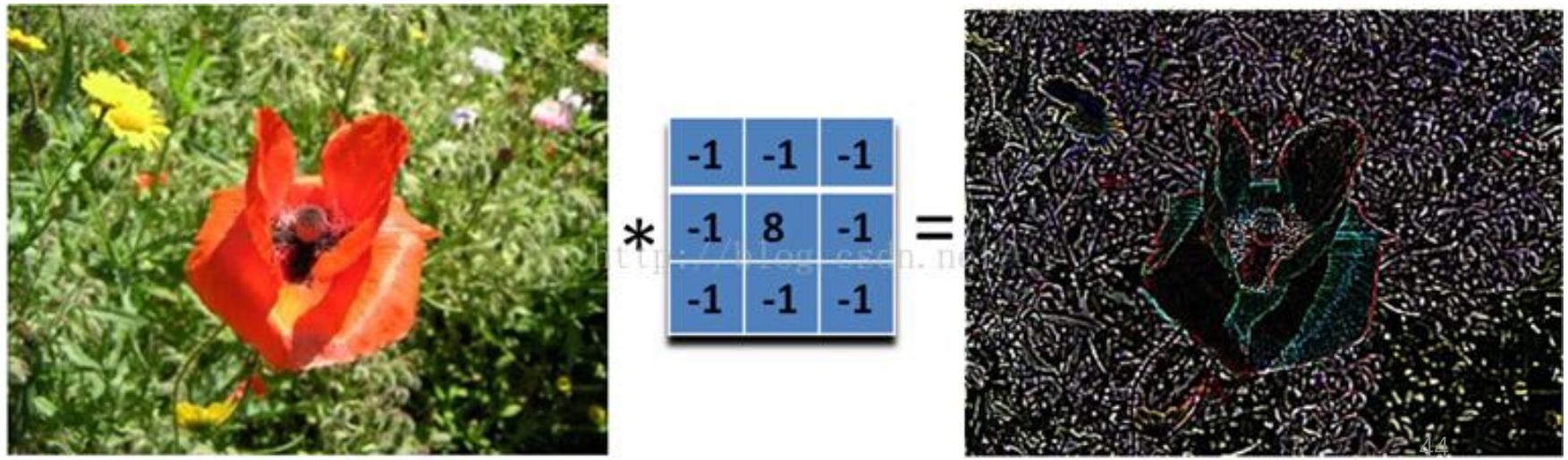
Sobel算子比Prewitt算子边缘检测性能更好，
Roberts算子计算对角线上而不是相邻两个像素的差分。

图像边缘检测

- 寻找45°边缘:



- 寻找所有方向的边缘:



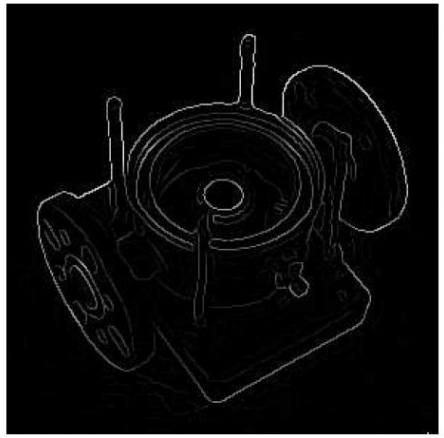
Canny边缘检测器

1. 图像降噪
2. 计算图像梯度
3. 非极大值抑制
   - 将多像素宽的边缘减少至一个像素宽度。
   - 规则：对每个“边缘”像素，沿着它的梯度方向，跟邻域像素的梯度强度进行比较，如果是局部极大值，则视为边缘像素，否则丢弃。
4. 双阈值筛选
   - 使用上下两个阈值来筛选。
   - 大于上阈值的认为是边缘点，小于下阈值的认为是非边缘点，在两个阈值之间的，如果与确定为边缘的点相邻接，认为是边缘点，否则剔除掉。

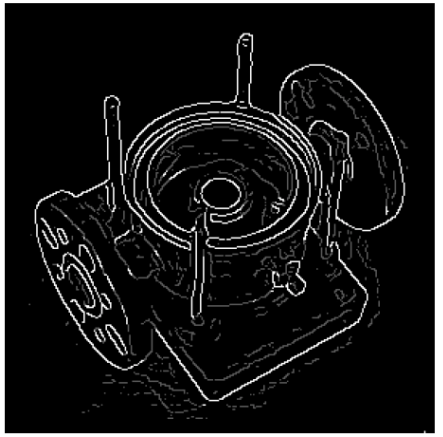
(a) Original

(b) Smoothed

(b) Gradient magnitudes



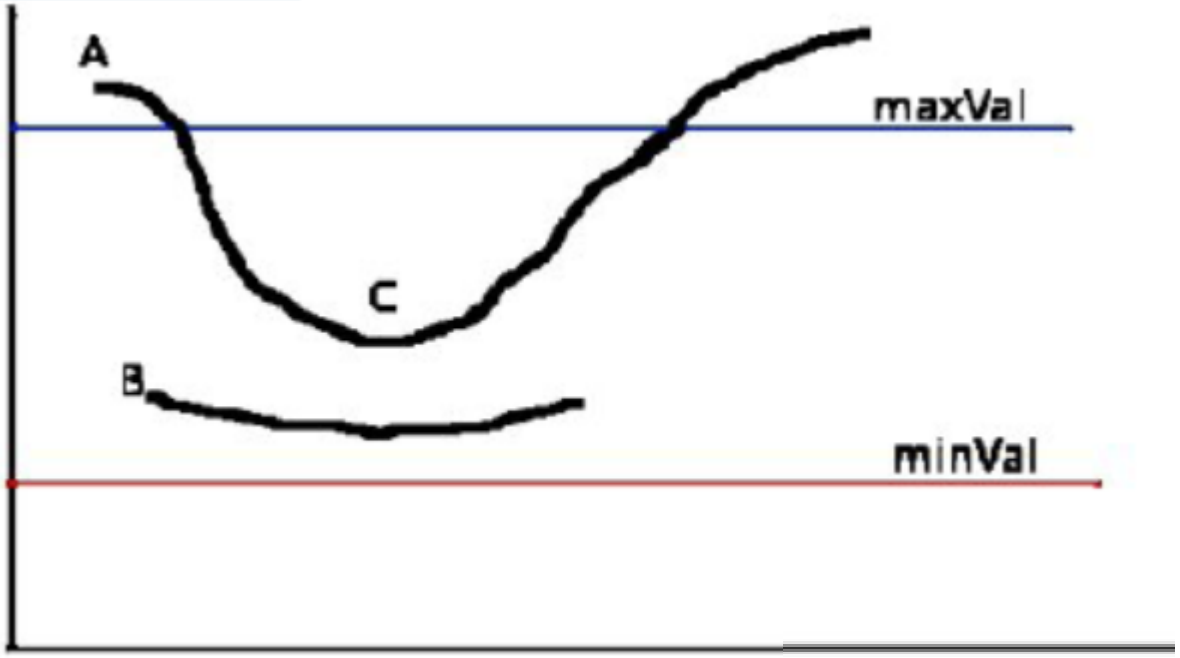
(b) Edges after non-maximum suppression



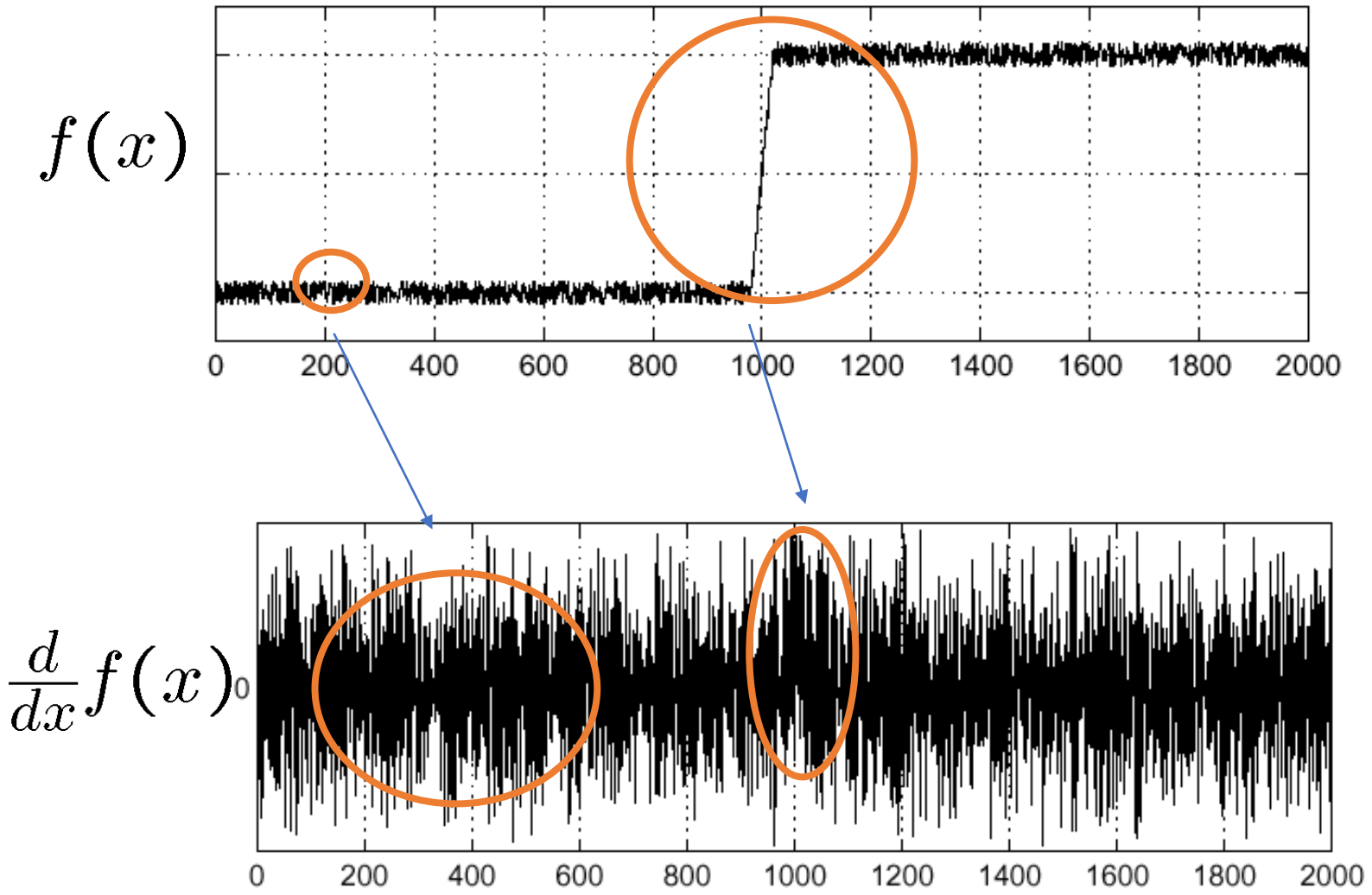
(b) Double thresholding

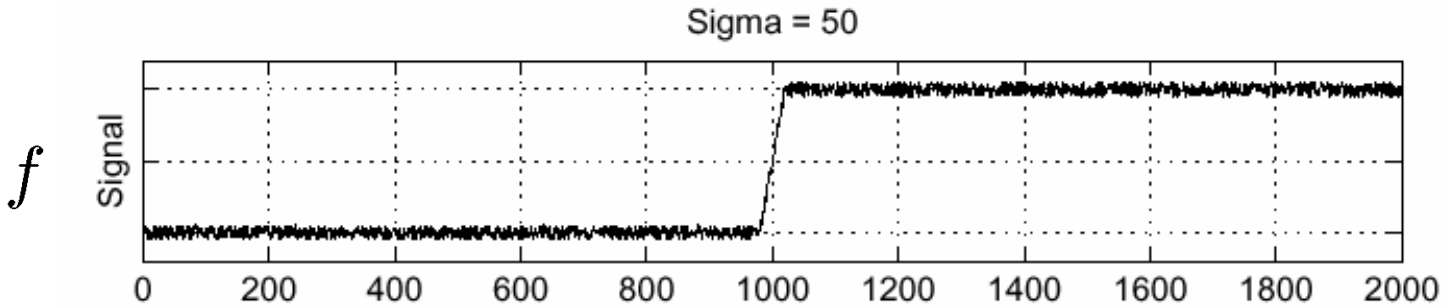
Canny边缘检测器

1. 图像降噪
2. 计算图像梯度
3. 非极大值抑制
   - 将多像素宽的边缘减少至一个像素宽度。
   - 规则：对每个“边缘”像素，沿着它的梯度方向，跟邻域像素的梯度强度进行比较，如果是局部极大值，则视为边缘像素，否则丢弃。
4. 双阈值筛选
   - 使用上下两个阈值来筛选。
   - 大于上阈值的认为是边缘点，小于下阈值的认为是非边缘点，在两个阈值之间的，如果与确定为边缘的点相邻接，认为是边缘点，否则剔除掉。

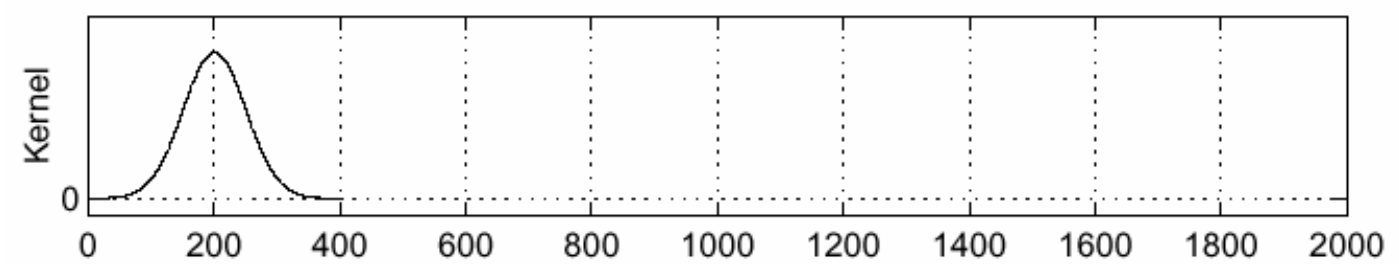


边缘检测前为什么要降噪?



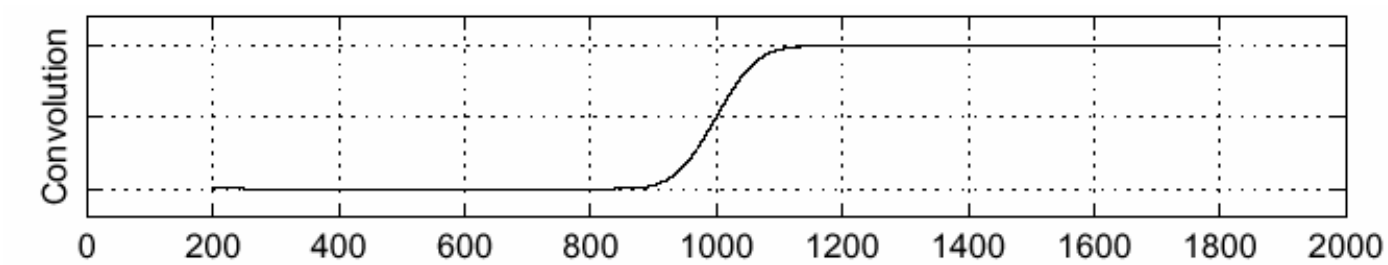


f



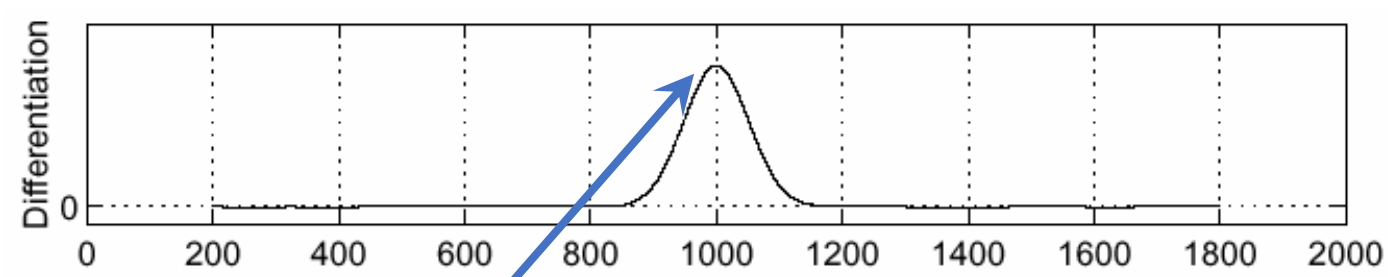
h

高斯函数



$h \star f$

高斯平滑



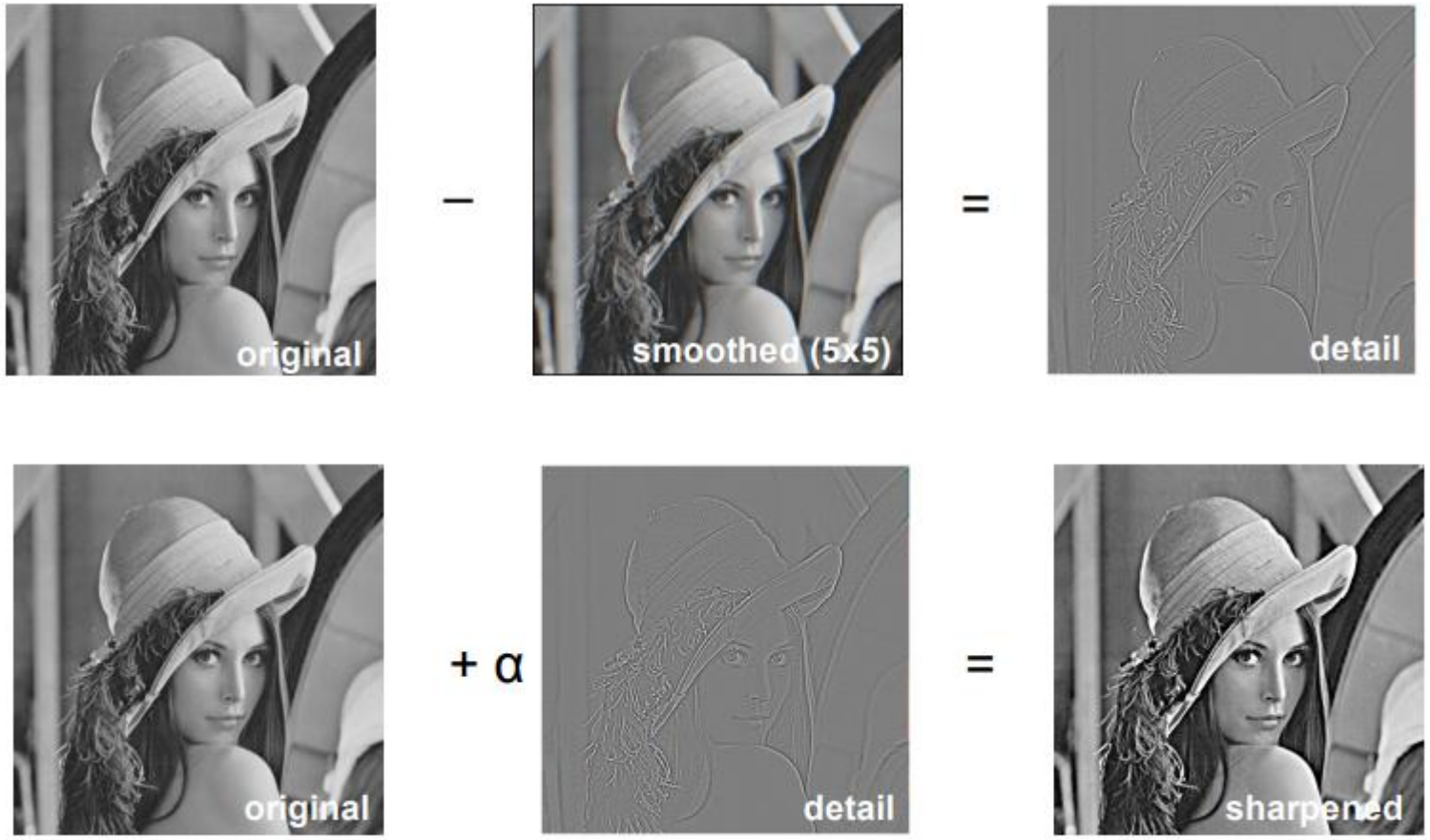
$\frac{\partial}{\partial x}(h \star f)$

求梯度

边缘点

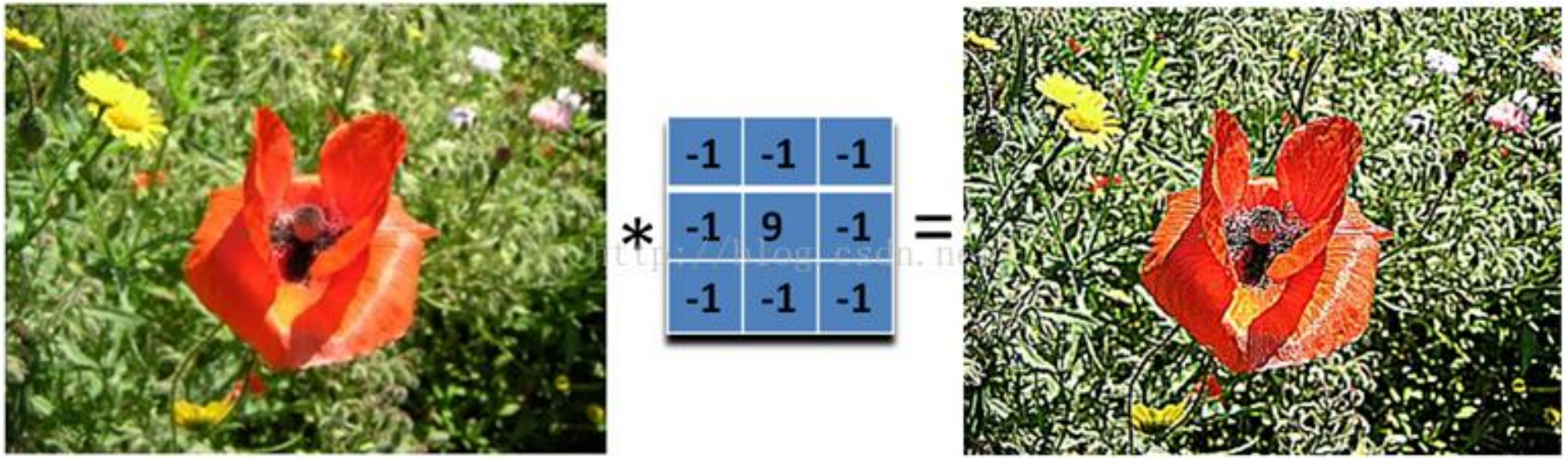
图像锐化

- 找到图像的边缘，然后把边缘加到原本的图像上，这样就强化了图像的边缘。

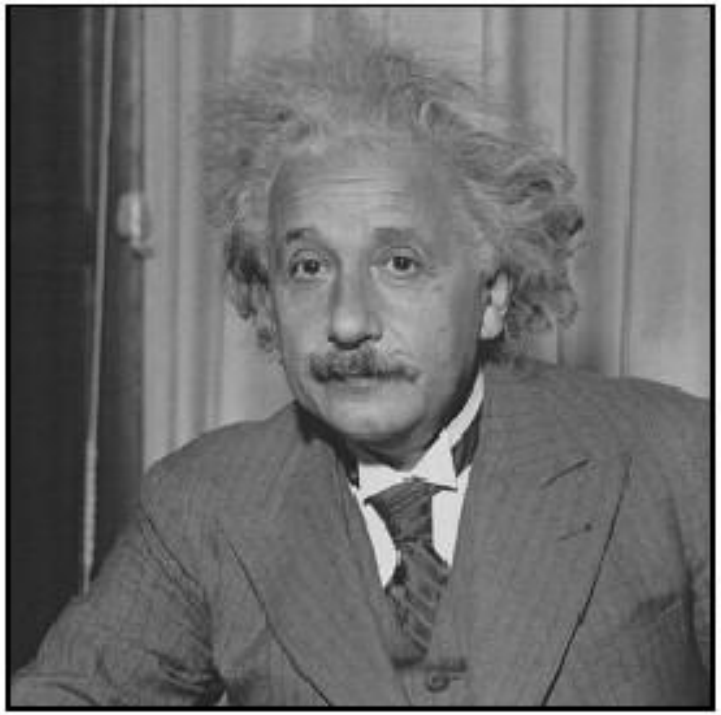


图像锐化

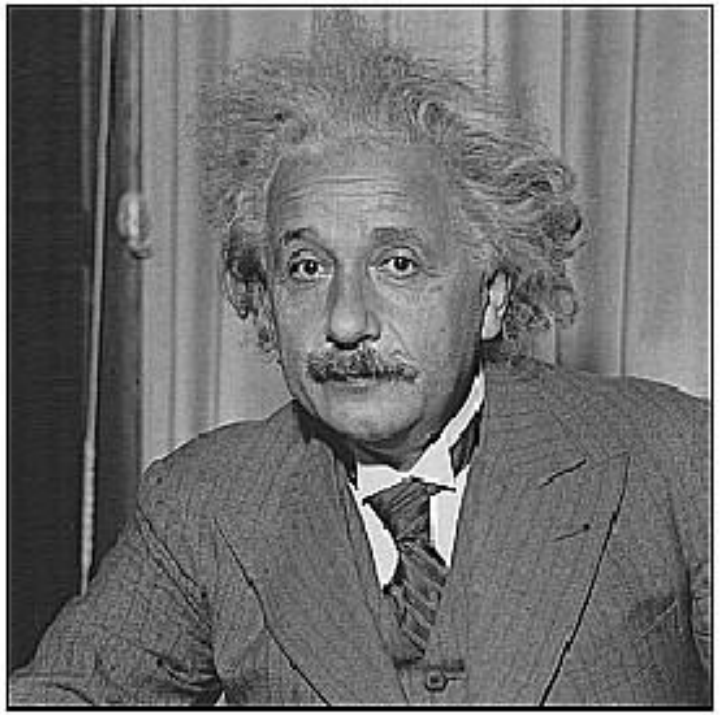
- 找到图像的边缘，然后把边缘加到原本的图像上，这样就强化了图像的边缘。



图像锐化



before



after

边界处理

选项一: 不做任何处理。（图像滤波（卷积）后会变小）

$F[x, y]$

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

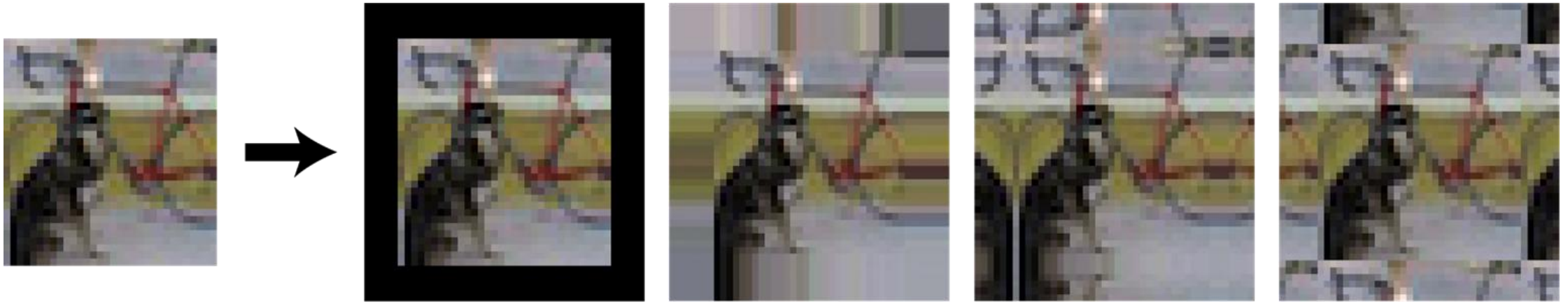
$G[x, y]$

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

边界处理

选项一: 不做任何处理。（图像滤波（卷积）后会变小）

选项二: 对原图像四周填充额外的行或列，使用不同的策略填充。（可保持图像大小不变）



Zero　　Clamp　　Mirror　　Wrap

填充0值像素　　填充对应边缘像素　　与边缘区域对称填充　　填充对边区域像素

案例分析-1

Original

0	0	0
0	1	0
0	0	0

?

案例分析-1

Original

0	0	0
0	1	0
0	0	0

滤波后
（没有变化）

案例分析-2

Original

0	0	0
0	0	1
0	0	0

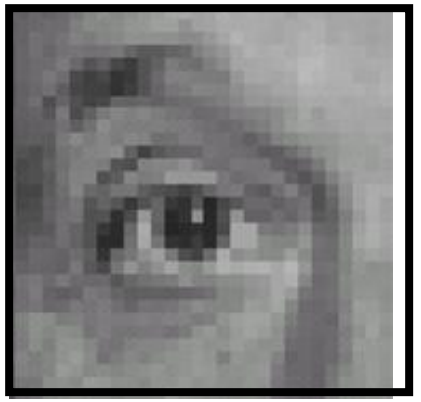
?

（假定卷积核已经旋转）

案例分析-2

Original

0	0	0
0	0	1
0	0	0



左移一个像素

（假定卷积核已经旋转）

案例分析-3

Original

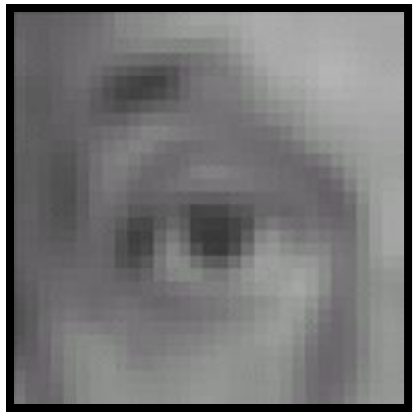
$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

?

案例分析-3

Original

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



模糊

案例分析-4

Original

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

案例分析-4

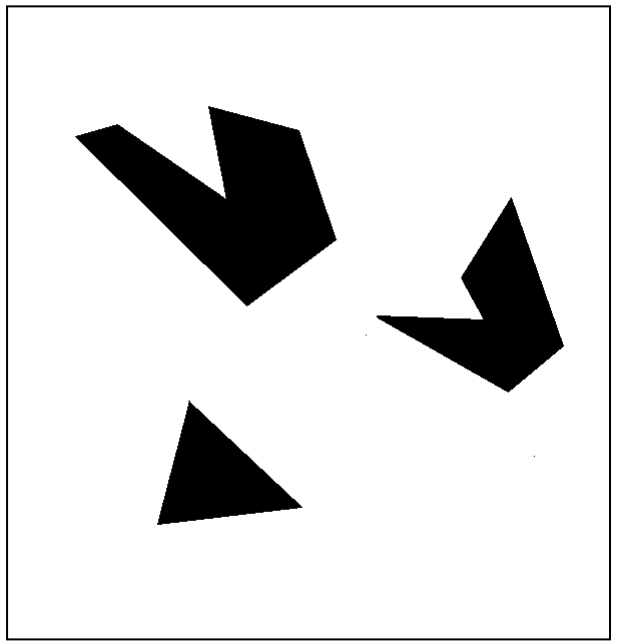
Original

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

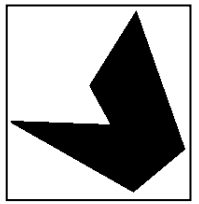
锐化滤波，注重与平均值的差异

模板匹配

- 模板是我们已知的小图像，模板匹配就是在一副大图像中搜寻目标。
- 解决方案：滤波器的值就是模板的像素值!



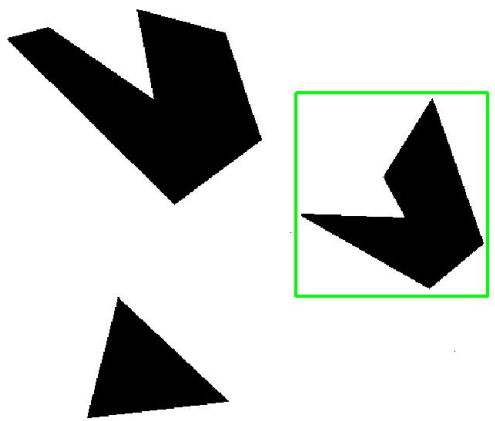
图像



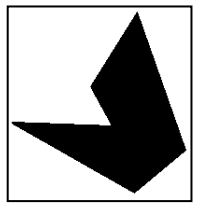
模板

模板匹配

- 在要检测的图像上，用滤波器从左到右、从上到下遍历图像，计算滤波器（模板）与重叠子图像的像素匹配度。如果匹配程度高，说明相似度高。



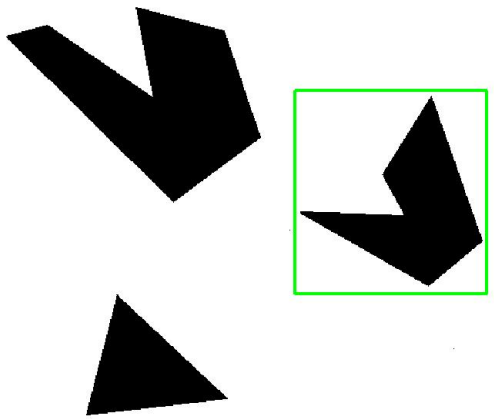
检测到的模板



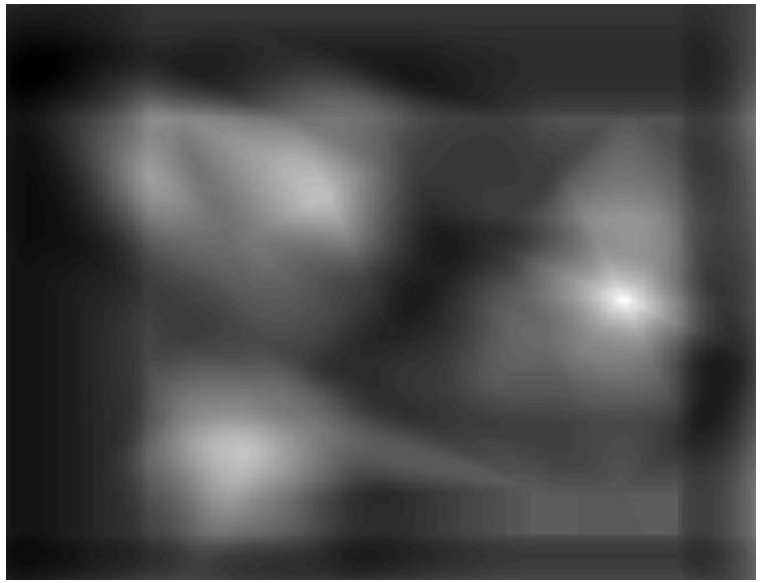
模板

模板匹配

- 互相关图中亮度最高的地方就是匹配度最大的位置。



检测到的模板



互相关图

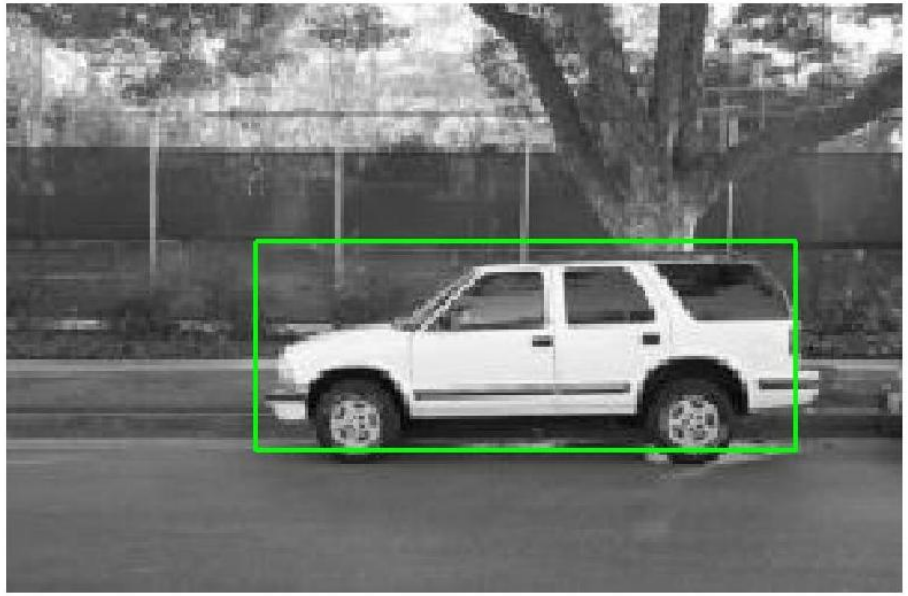
模板匹配

图像

模板

如果模板不属于原图像的子图，结果会怎么样?

模板匹配

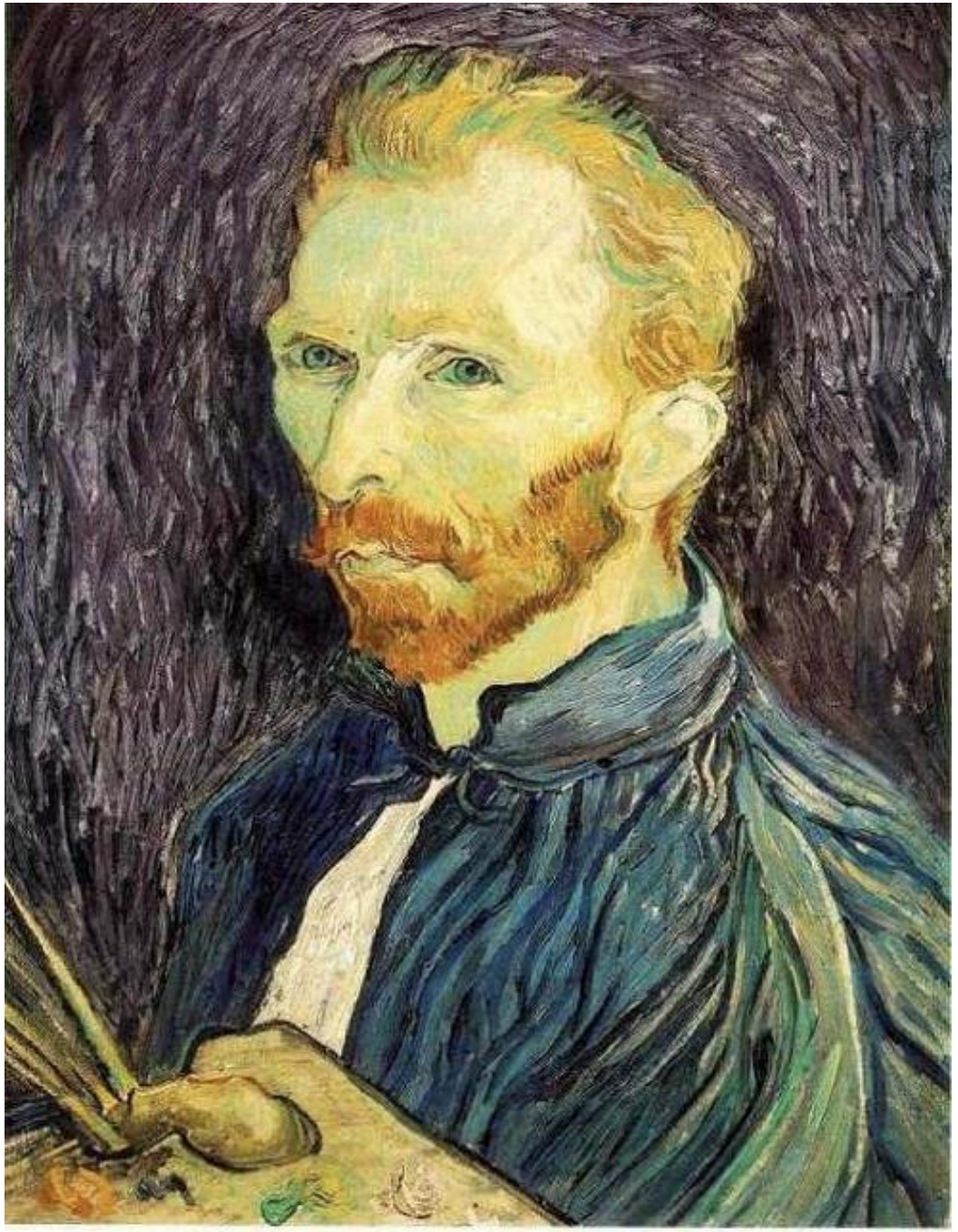


图像

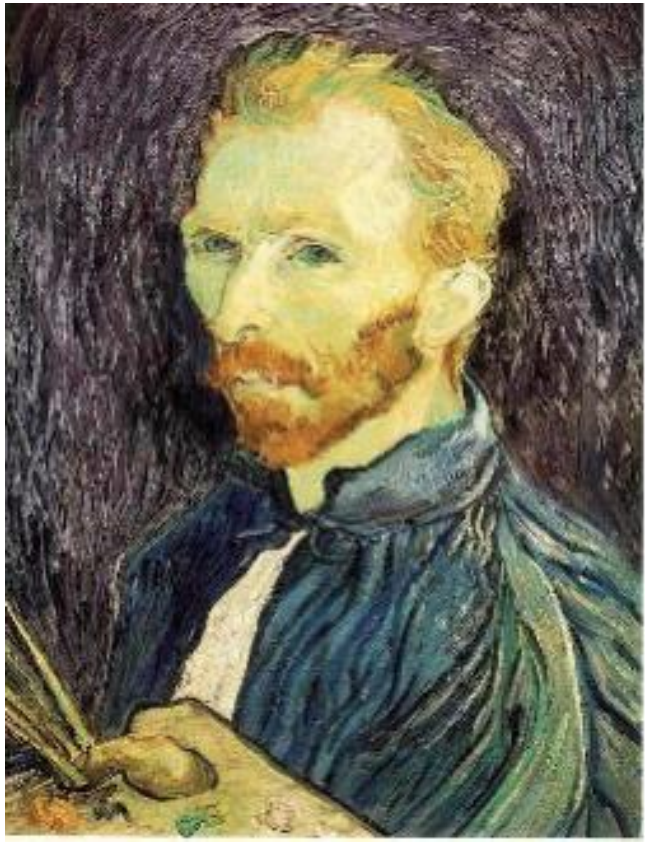
模板

如果比例、方向和整体外观相近，那么极大可能会匹配成功。

如何将图像缩小?



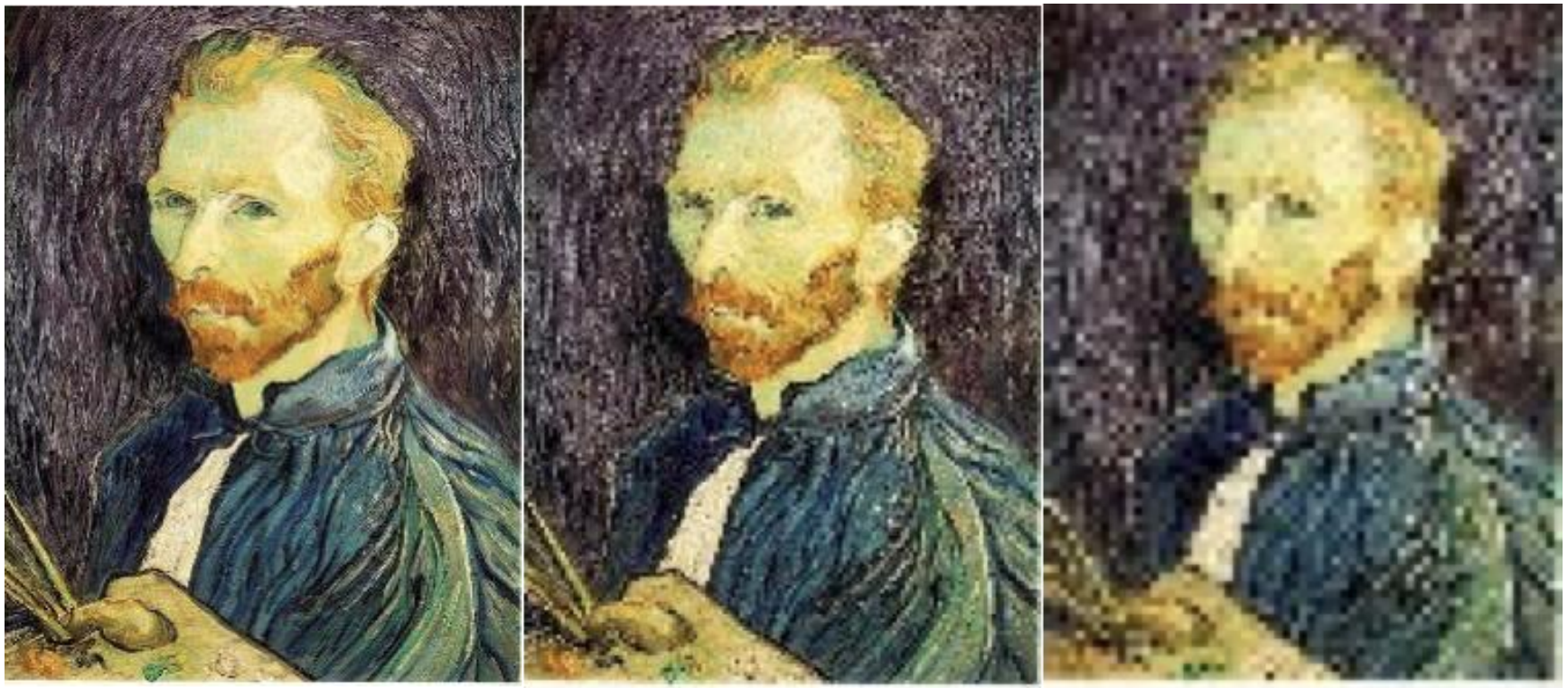
交替去除一些行和列



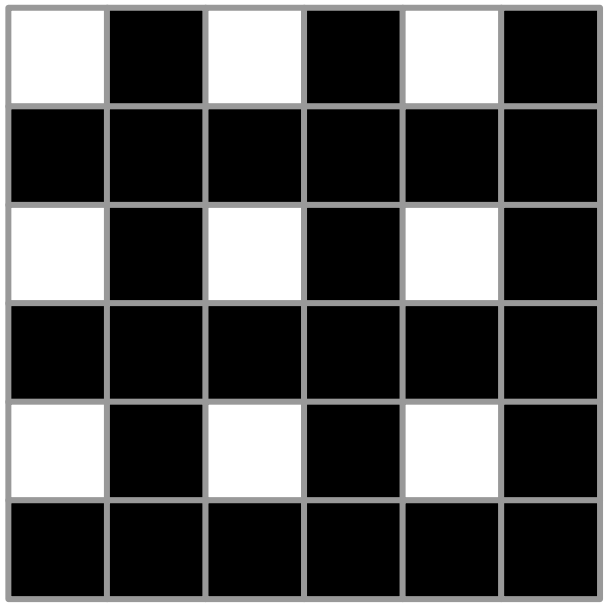
1/4

1/8

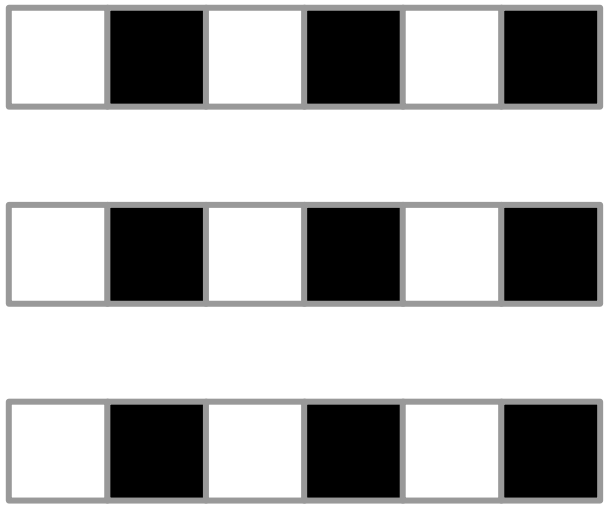
对图像缩小后再放大，图像失真



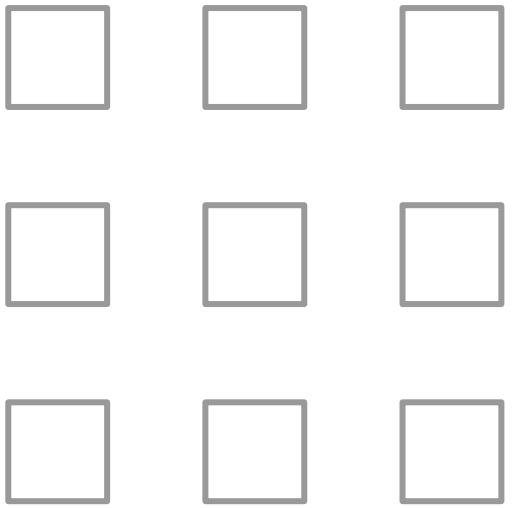
原始数据



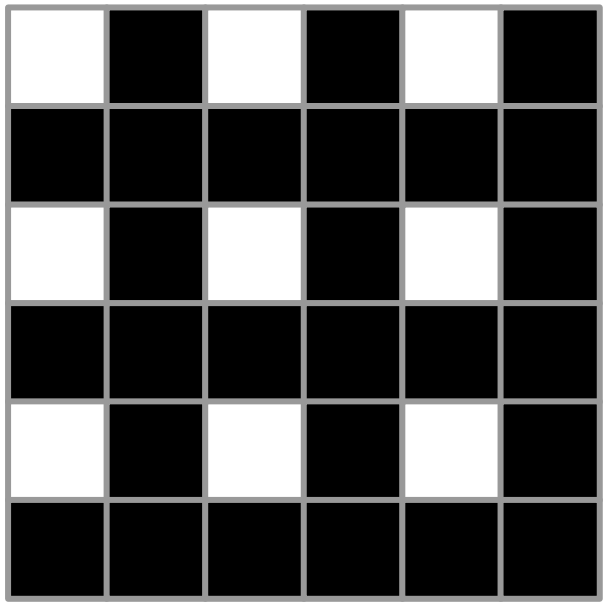
对原始数据缩小1/2

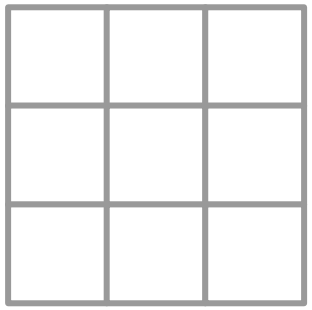


Step 1: 去除行后



Step2: 去除列后

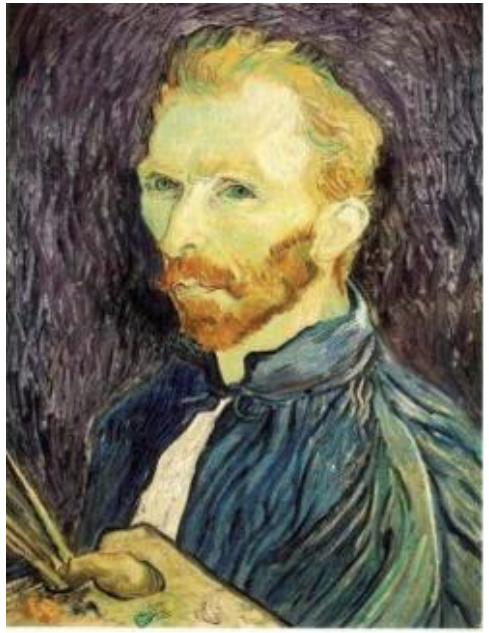




1/2

信息严重丢失!

滤波后下采样



Gaussian 1/2

G 1/4

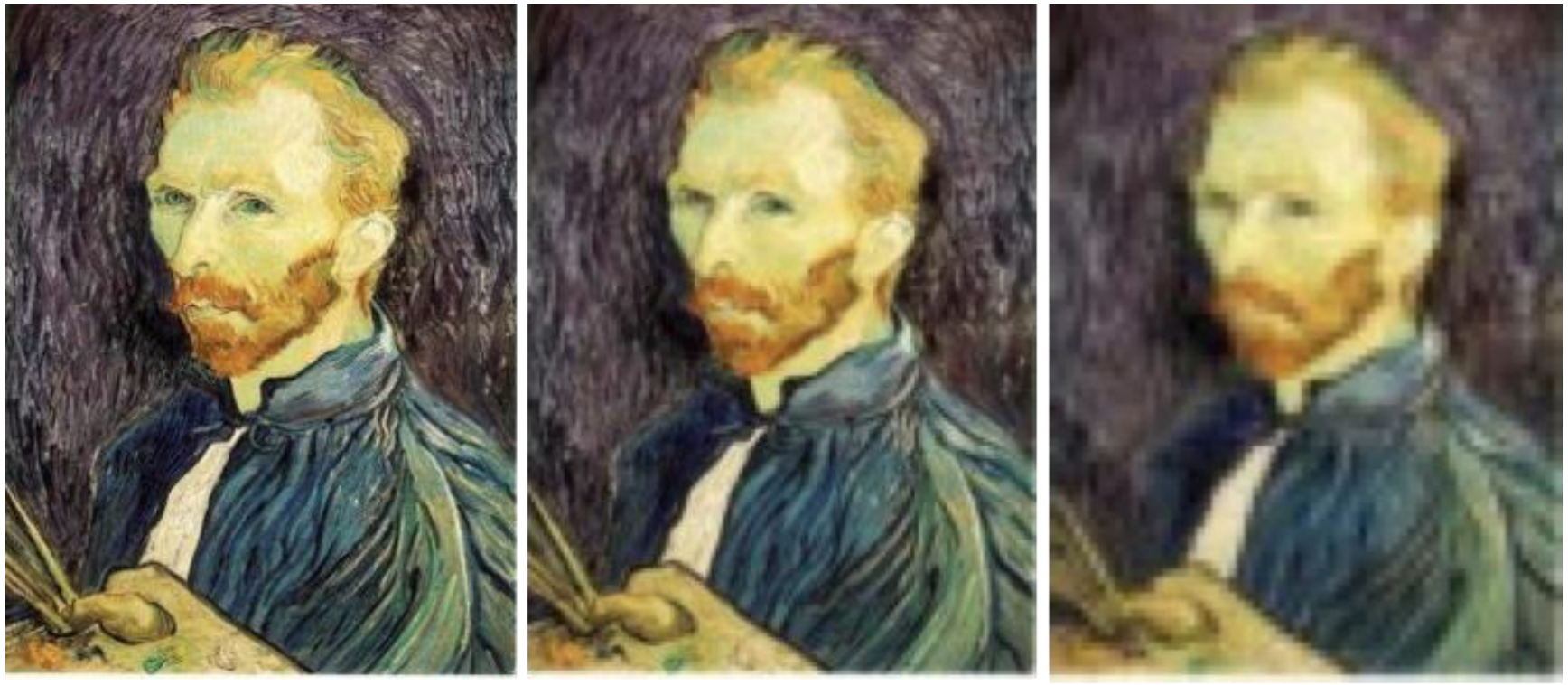
G 1/8

先对图像进行高斯滤波，然后进行下（子）采样。

- 每减少1/2尺寸，滤波核大小应增加一倍。为什么？

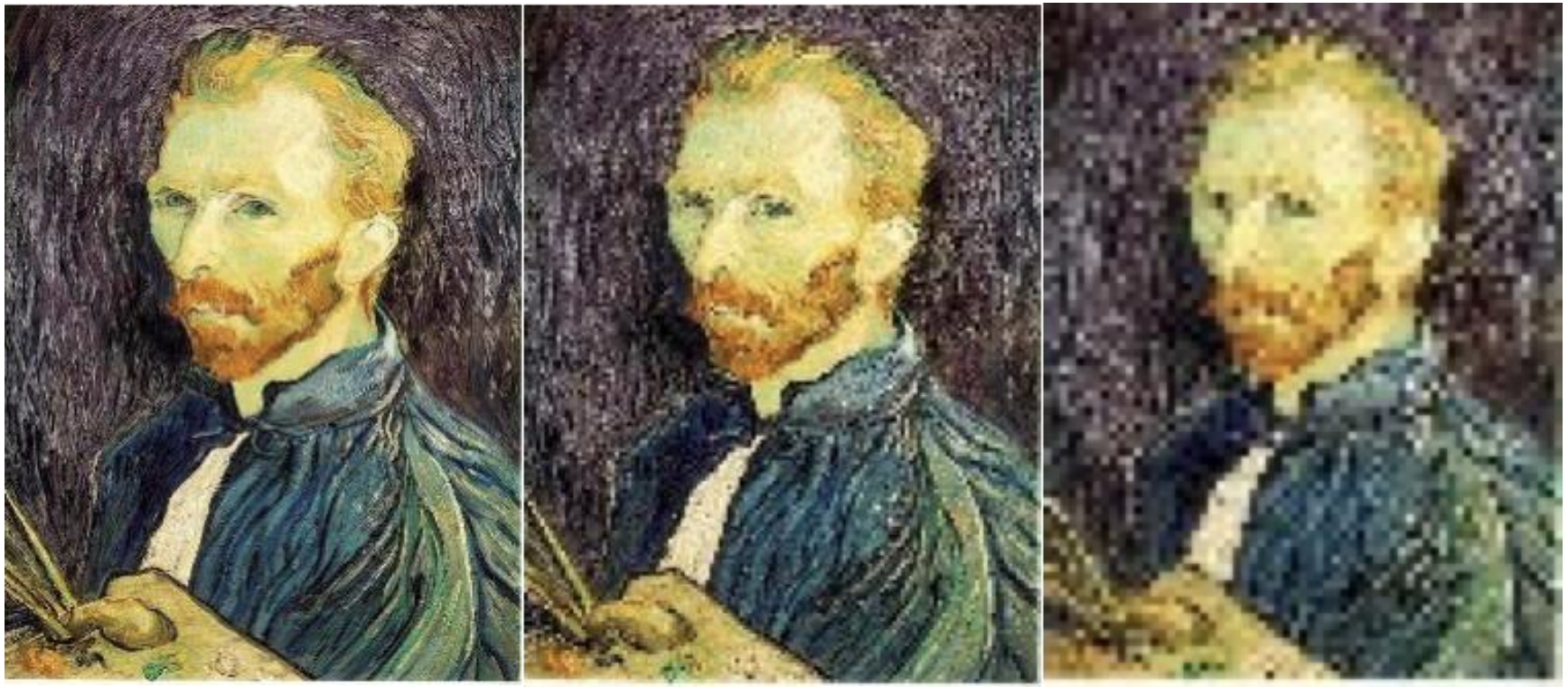
滤波核尺寸增加一倍，每个缩小后的图像对应的像素就考虑了多一倍的像素信息。

过滤后下采样



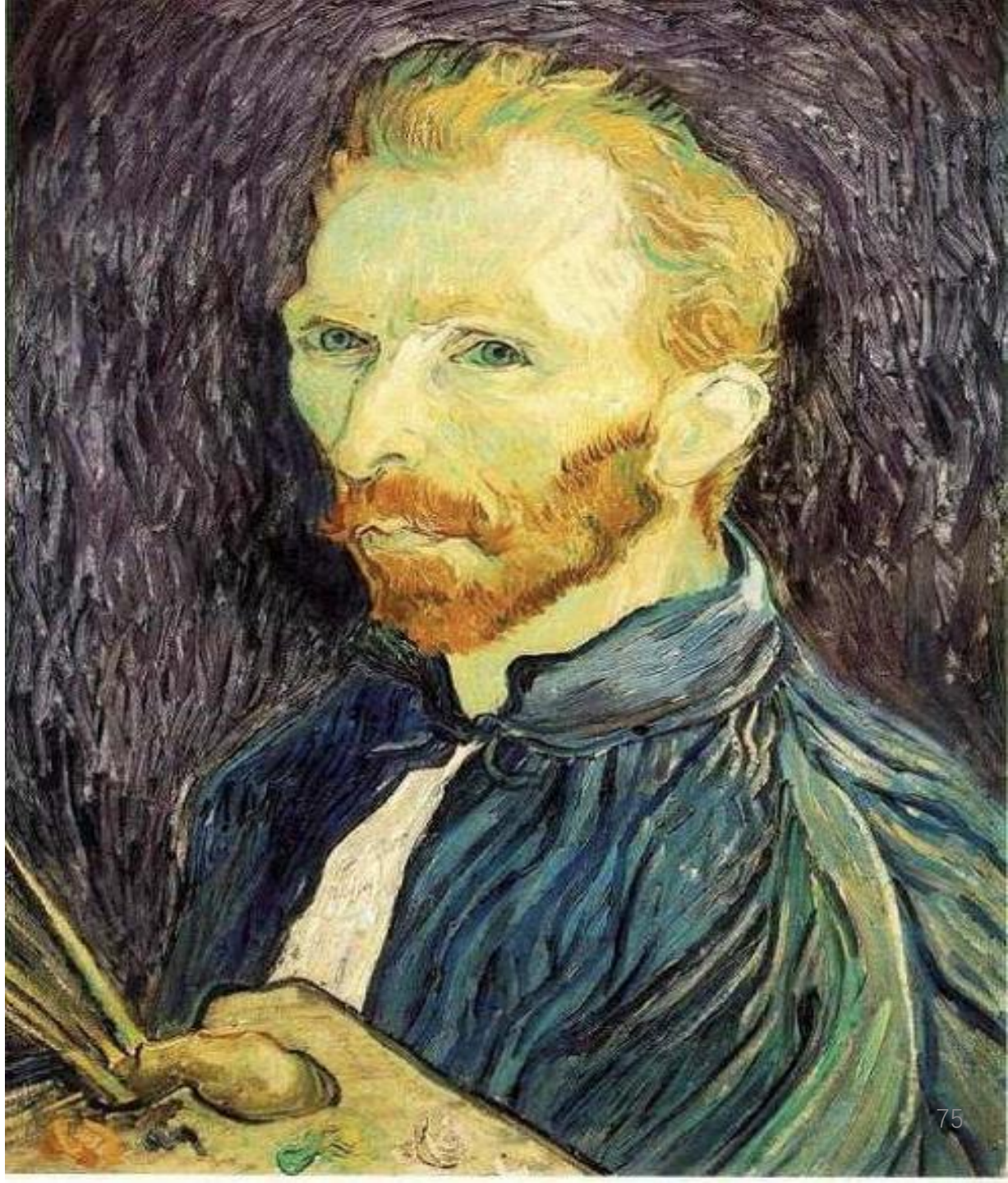
Gaussian 1/2　　G 1/4　　G 1/8

无过滤下采样



1/2 1/4 (2x zoom) 1/8 (4x zoom)

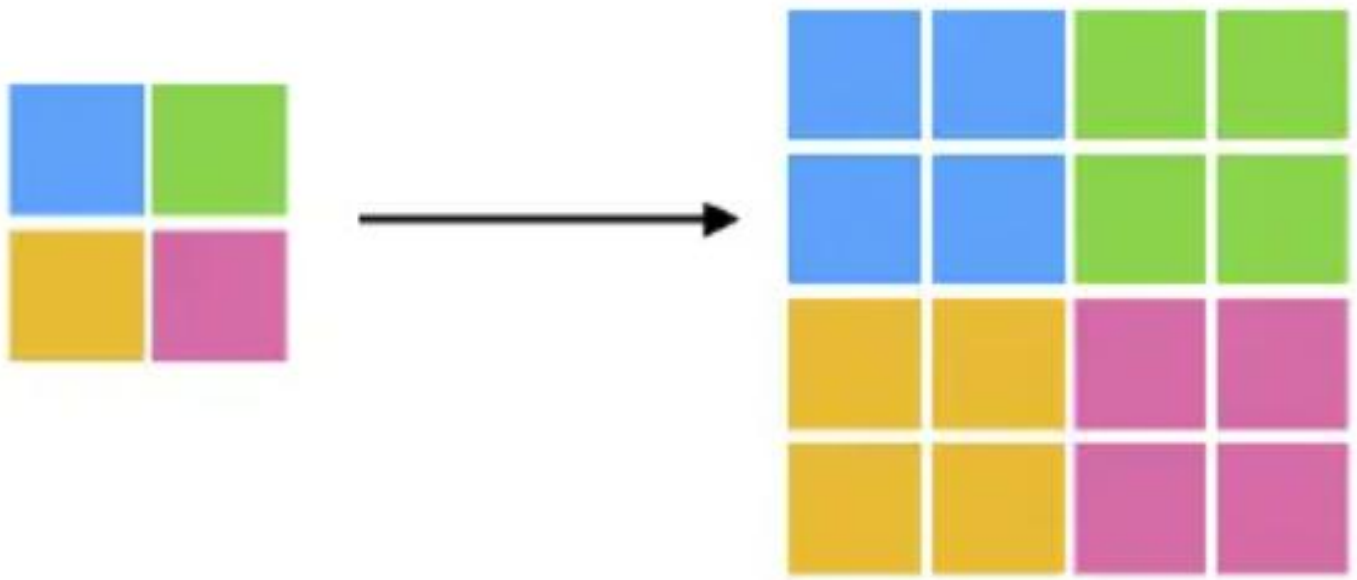
如何将图像放大？



最近邻

算法基本思想：令新增像素的灰度值等于距它最近的输入像素灰度值

- 计算简单，但不够精确，有锯齿现象



最近邻

算法基本思想：令新增像素的灰度值等于距它最近的输入像素灰度值

- 计算简单，但不够精确，有锯齿现象

双线性(Bilinear)插值

算法基本思想:

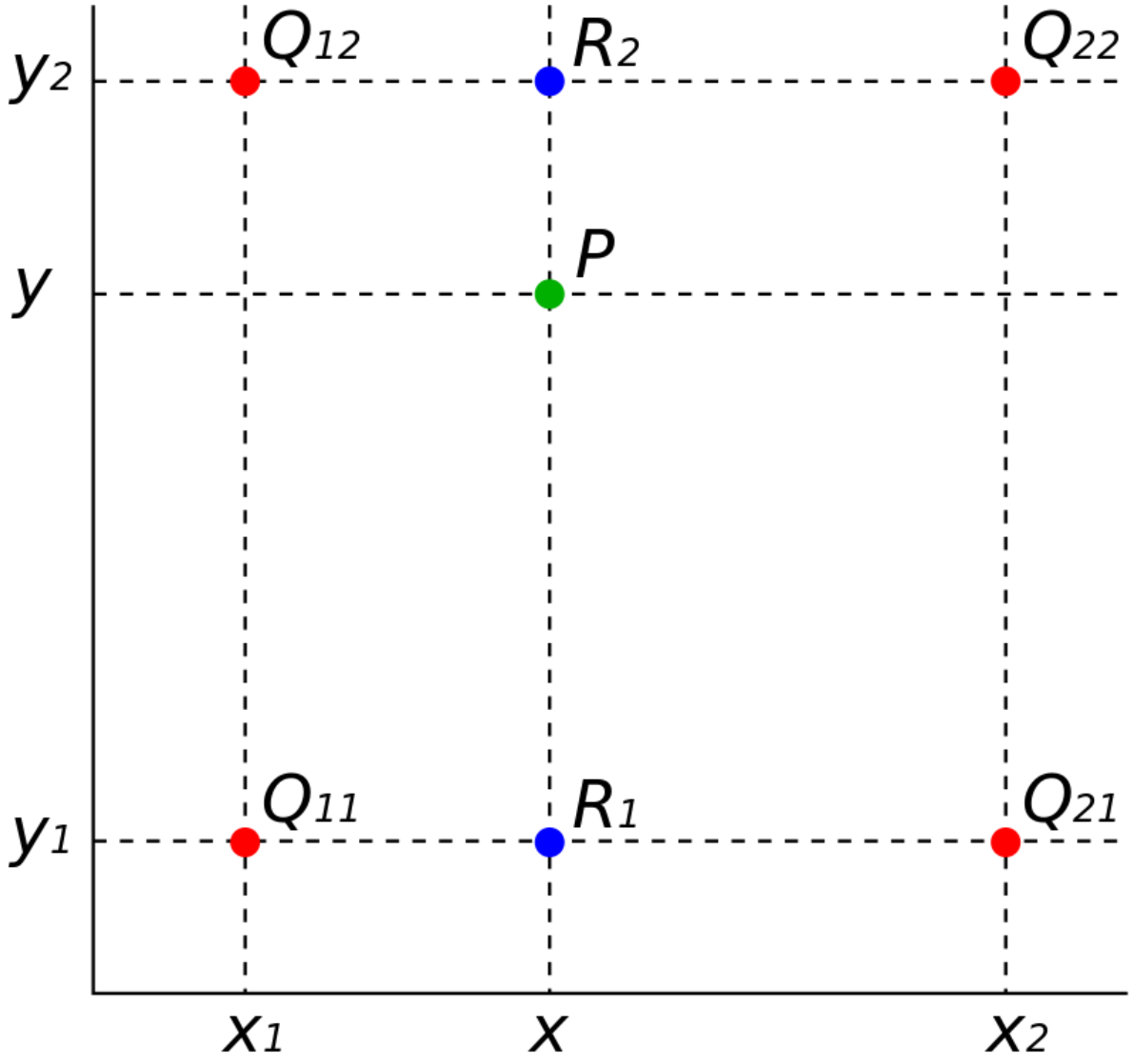
1. 沿X方向线性插值，在Q_{12}和Q_{22}之间插入蓝色点R_2，在Q_{11}和Q_{21}之间插入蓝色点R_1

$$f(x, y_1) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21}),$$

$$f(x, y_2) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22}).$$

2. 沿Y方向线性插值，通过第一步计算出的R_1和R_2在Y方向插值计算出P

$$f(P) \approx \frac{y_2 - y}{y_2 - y_1} f(R_1) + \frac{y - y_1}{y_2 - y_1} f(R_2).$$



$$f(x, y_1) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21}),$$

推导过程:

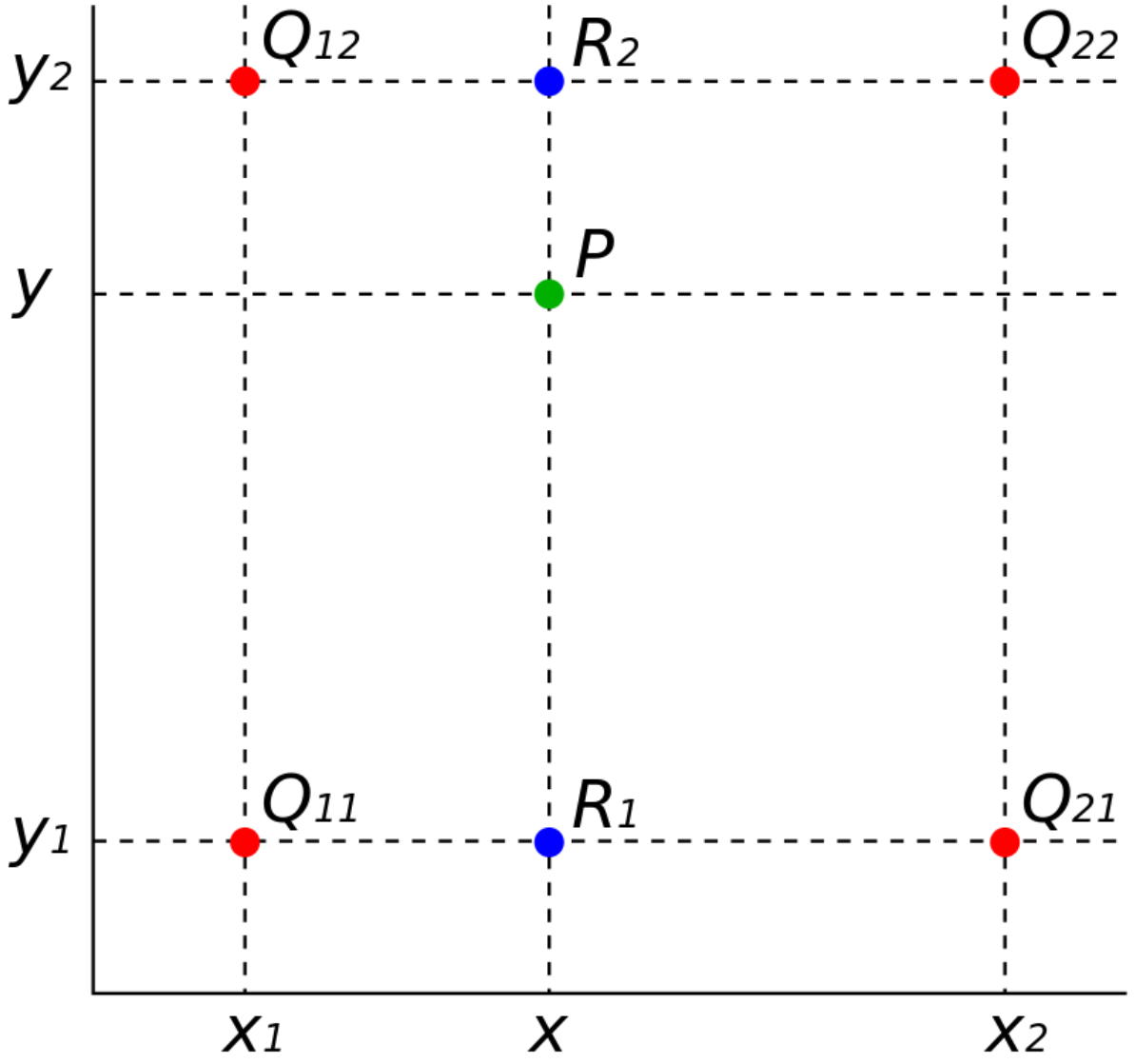
令$f(R_1) = f(x, y_1)$

$$\frac{f(Q_{21}) - f(Q_{11})}{x_2 - x_1} = \frac{f(R_1) - f(Q_{11})}{x - x_1}$$

$$f(R_1) - f(Q_{11}) = (x - x_1)\frac{f(Q_{21}) - f(Q_{11})}{x_2 - x_1}$$

$$f(R_1) = f(Q_{11}) + (x - x_1)\frac{f(Q_{21}) - f(Q_{11})}{x_2 - x_1}$$

$$= \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21})$$



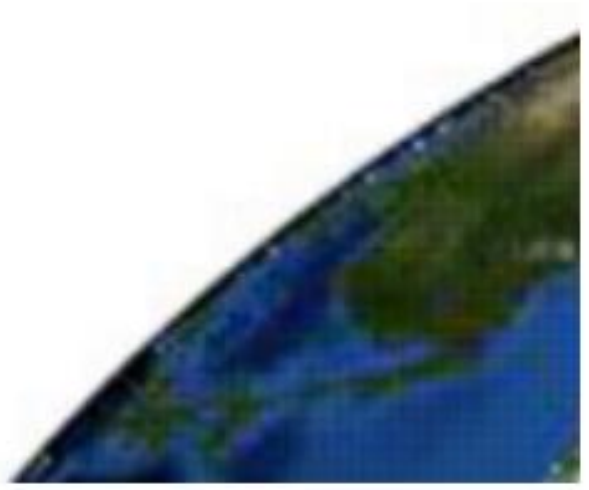
双三次(Bicubic)插值

算法基本思想:

$f(x,y)$的值可以通过矩形网格中最近的十六个采样点的加权平均得到，需要使用两个多项式插值三次函数，每个方向使用一个。

最近邻插值

双线性内插



双三次内插

图像超分辨率重建

超分辨率重建开山之作SRCNN：

http://mmlab.ie.cuhk.edu.hk/projects/SRCNN.html

Bicubic / 33.91 dB

VS.

SRCNN / 35.01 dB

Bicubic / 32.39 dB

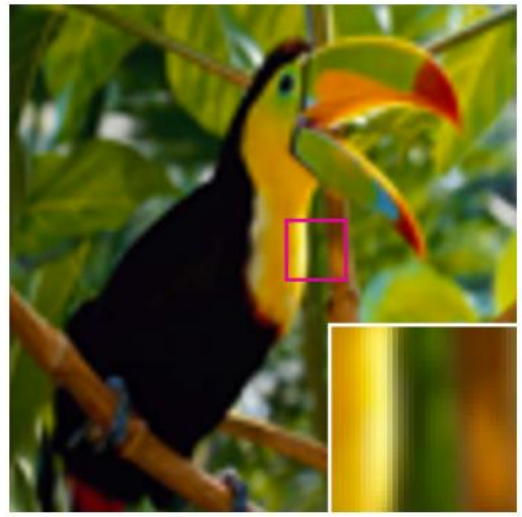
VS.

SRCNN / **34.35** dB

Bicubic / 24.04 dB

VS.

SRCNN / **27.58** dB



Bicubic / 32.58 dB

VS.

SRCNN / **34.91** dB

谢谢!