计算机视觉

董秋雷
中国科学院自动化研究所
qldong@nlpr.ia.ac.cn

提纲

1 深度学习与卷积神经网络

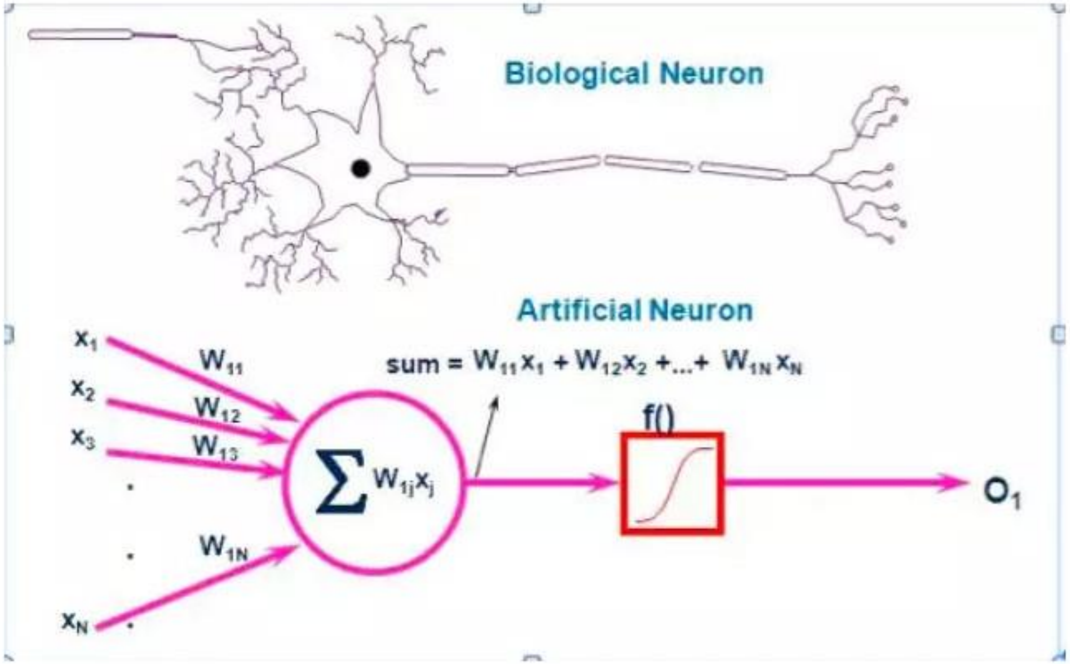
2 图像底层特征提取

深度学习的发展历史

人工神经网络　　浅层学习　　深度学习

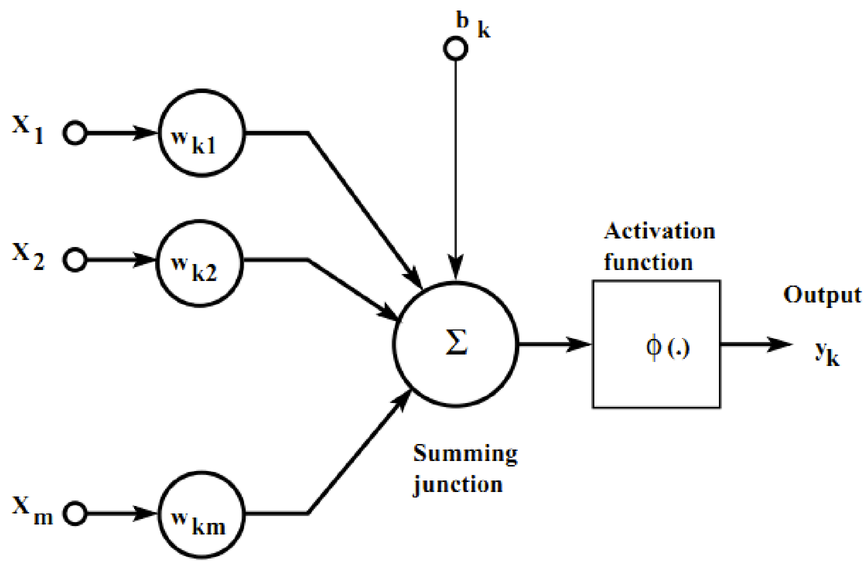
生物神经元与 MP 模型

生物神经元	神经元	输入信号	权值	输出	总和	膜电位	阈值
MP 模型	j	χ_i	ω_{ij}	o_j	$\sum$	$\sum_{i=1}^{n} \omega_{ij}\chi_i(t)$	T_j



深度学习的发展历史

人工神经网络　　浅层学习　　深度学习



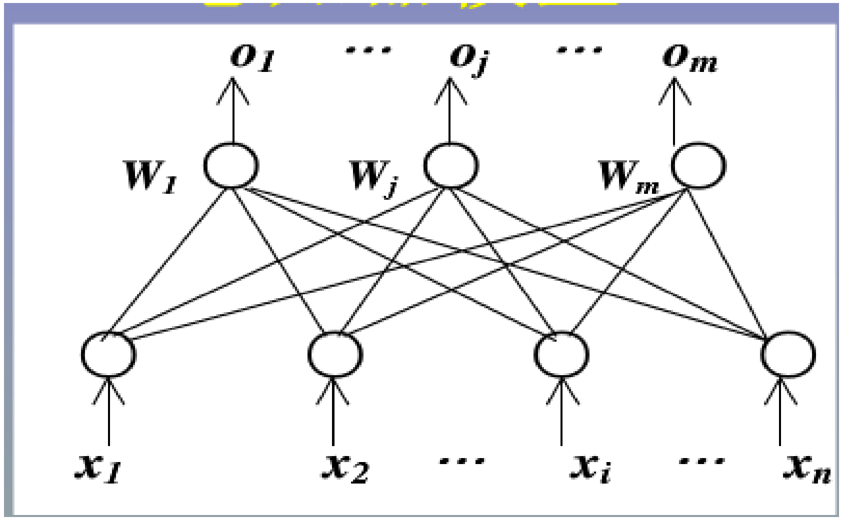
$$y = f[\sum W_i x_{Ei} - T] \qquad E = \sum W_i x_{Ei}$$

$$y = \begin{cases} 1 & E - T \geq 0, \text{且} I = 0 \\ 0 & E - T < 0, \text{或} I > 0 \end{cases}$$

1943年，Mcculloch 和 Pitts提出神经元的数学模型；

深度学习的发展历史

人工神经网络 → 浅层学习 → 深度学习



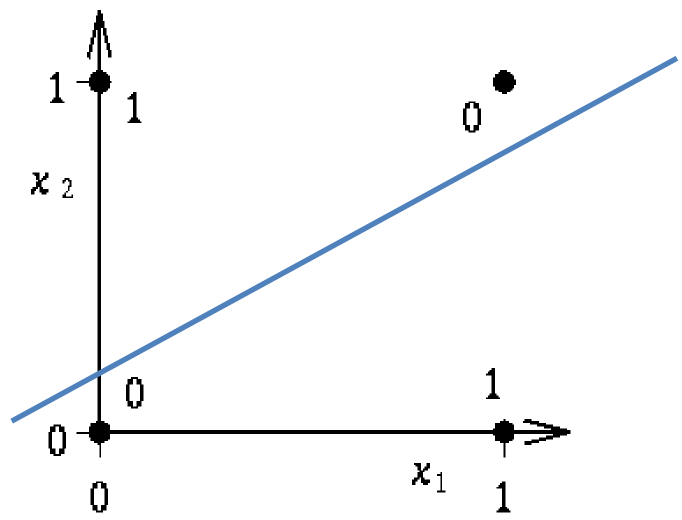
50 年代末，Rosenblatt 提出了感知机模型；

1943年，Mcculloch 和 Pitts提出神经元的数学模型；

深度学习的发展历史

人工神经网络 | 浅层学习 | 深度学习

1968 年，《感知机》中指出线性感知机的功能是有限的，它不能解决如异或这样的基本问题；



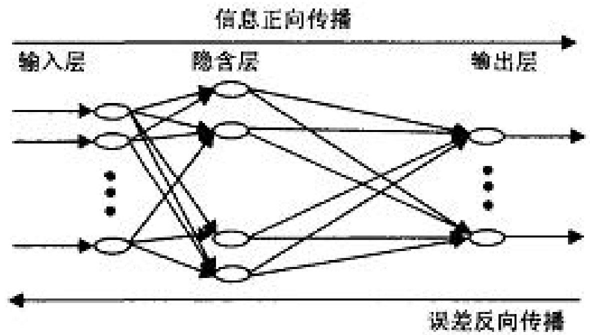
50 年代末，Rosenblatt 提出了感知机模型；

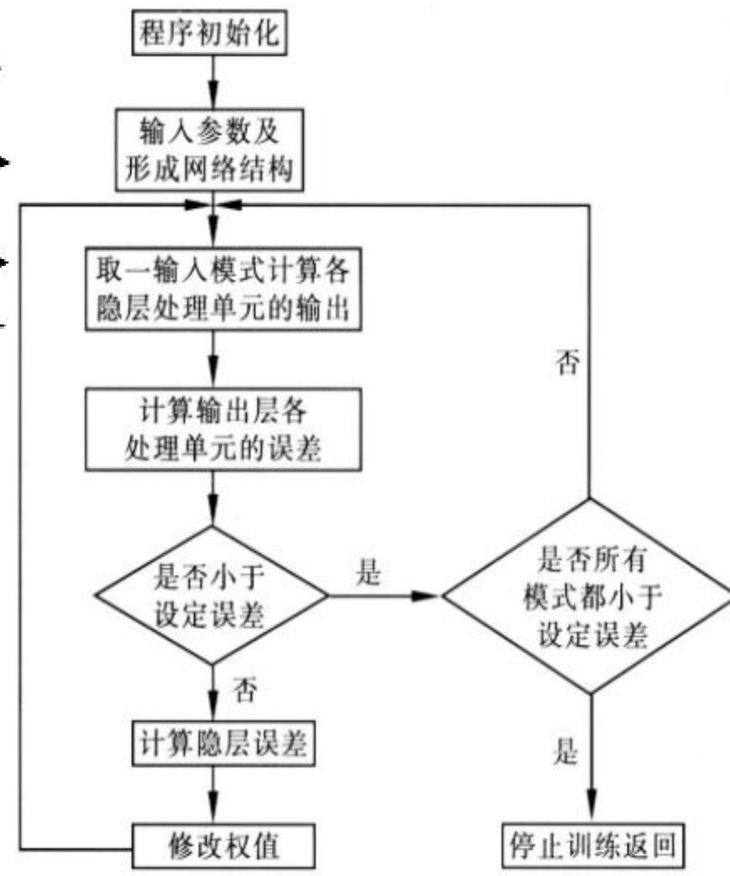
1943年，Mcculloch 和 Pitts提出神经元的数学模型；

深度学习的发展历史

人工神经网络　　浅层学习　　深度学习

80年代末期，BP算法提出。





1968 年，《感知机》中指出线性感知机的功能是有限的，它不能解决如异或这样的基本问题；

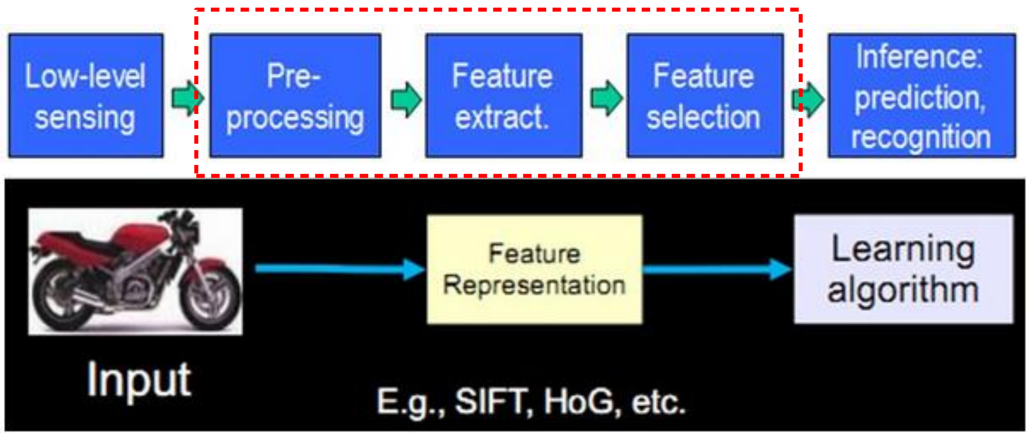
50 年代末，Rosenblatt 提出了感知机模型；

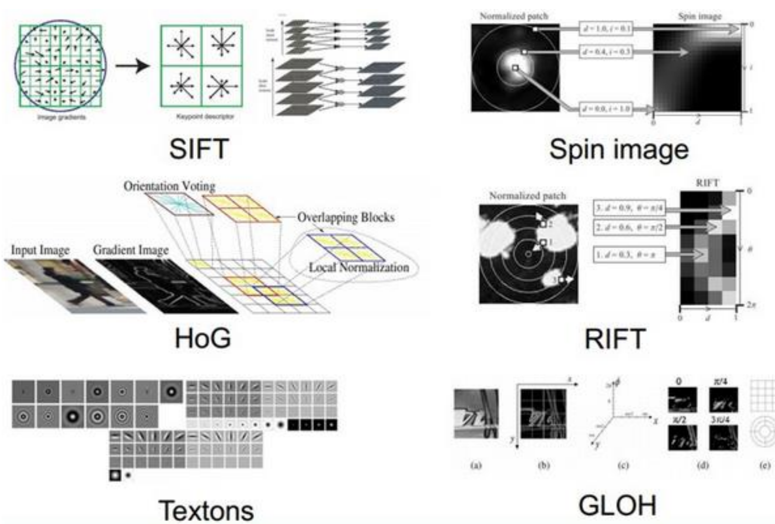
1943年，Mcculloch 和 Pitts提出神经元的数学模型；

深度学习的发展历史

人工神经网络　浅层学习　深度学习

90年代，SVM、 Boosting、最大熵方法等方法相继被提出



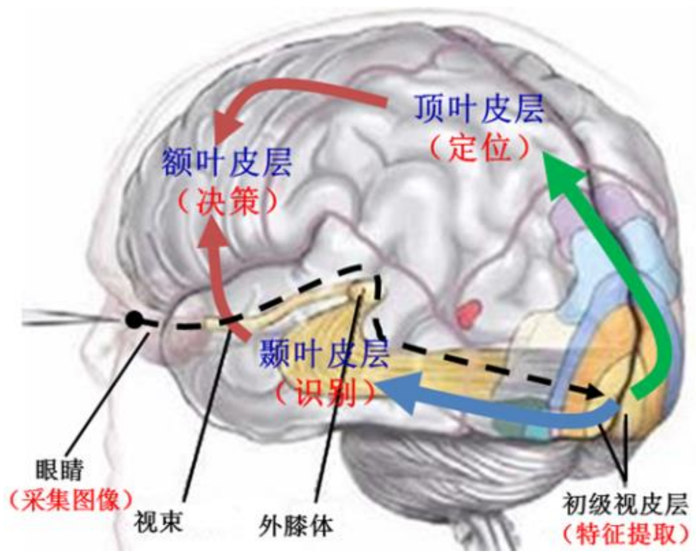


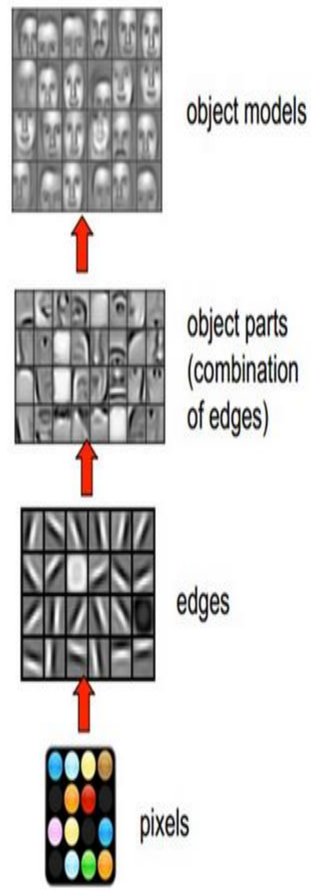
$$y = f(input)$$

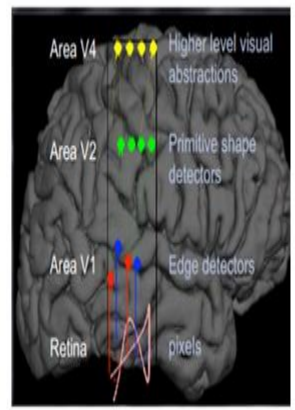
深度学习的发展历史

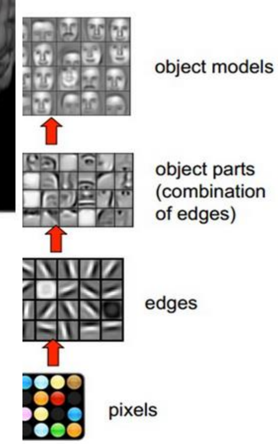
人工神经网络 　浅层学习 　深度学习

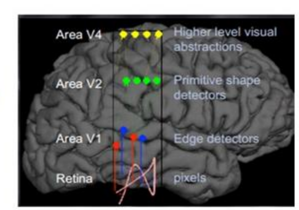
2006年，Hinton等人提出了深度学习的方法





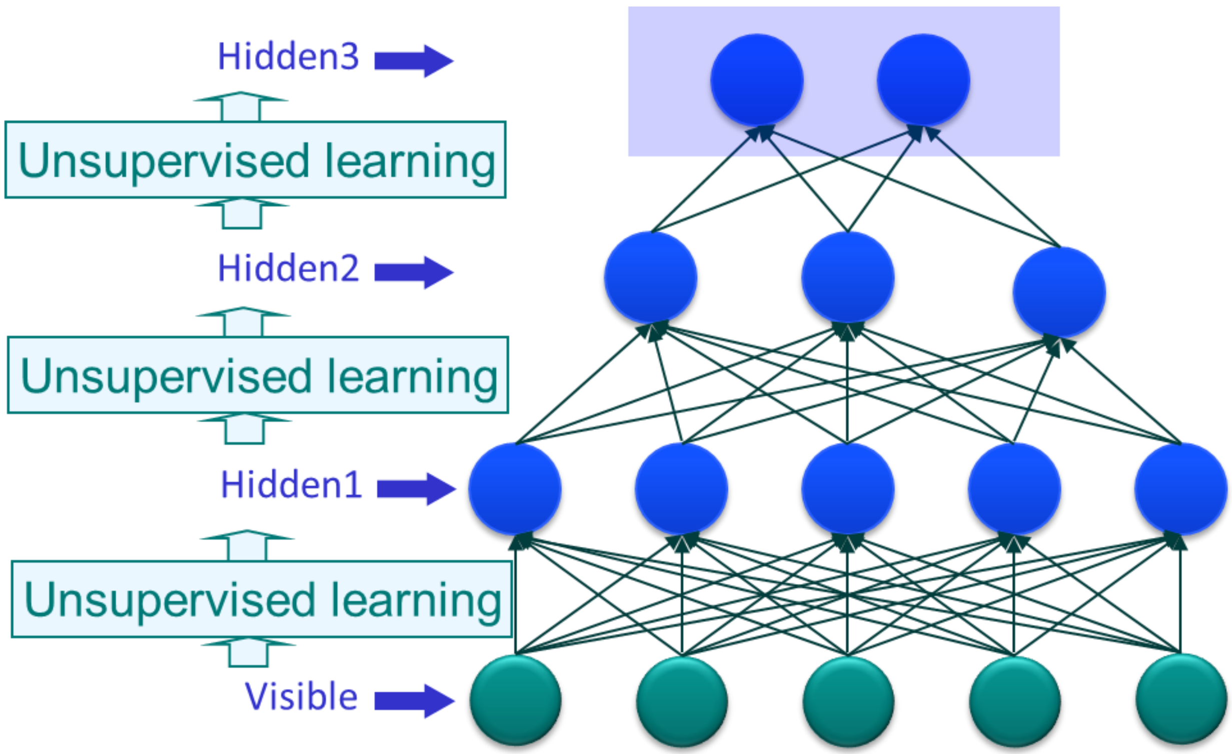






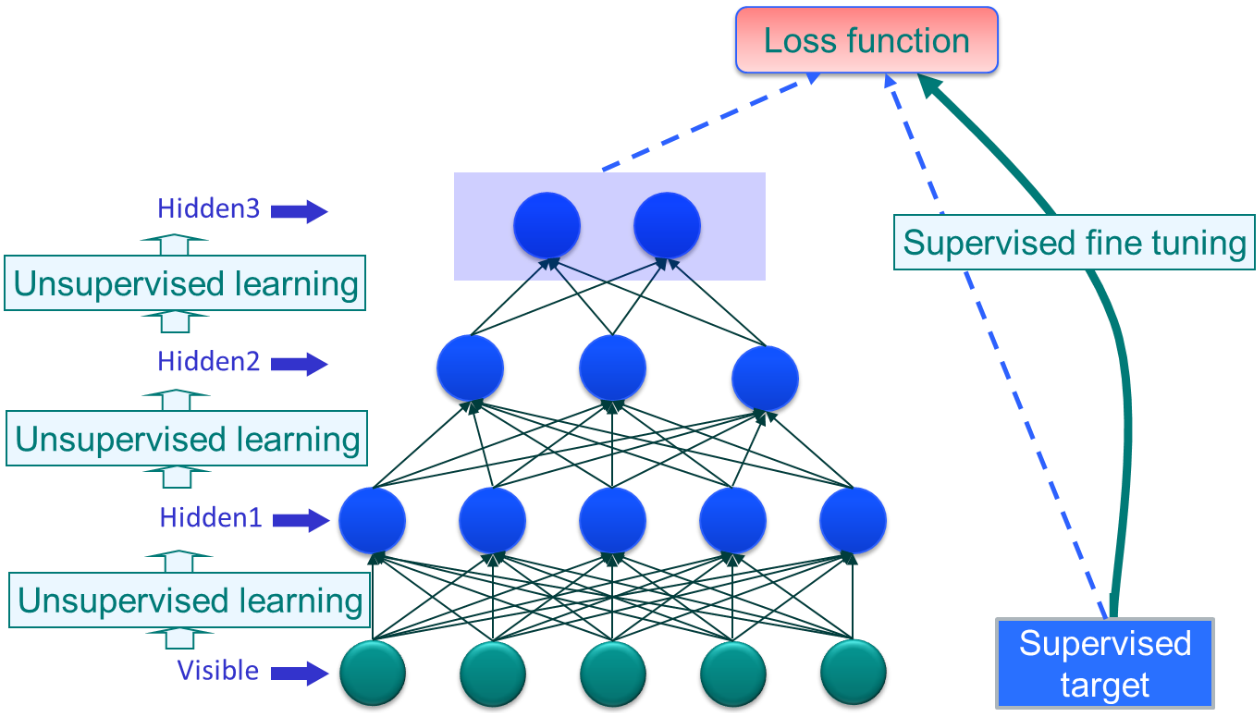
深度网络的基本设计思想

堆叠多层网络，即将低一层的输出作为高一层的输入。



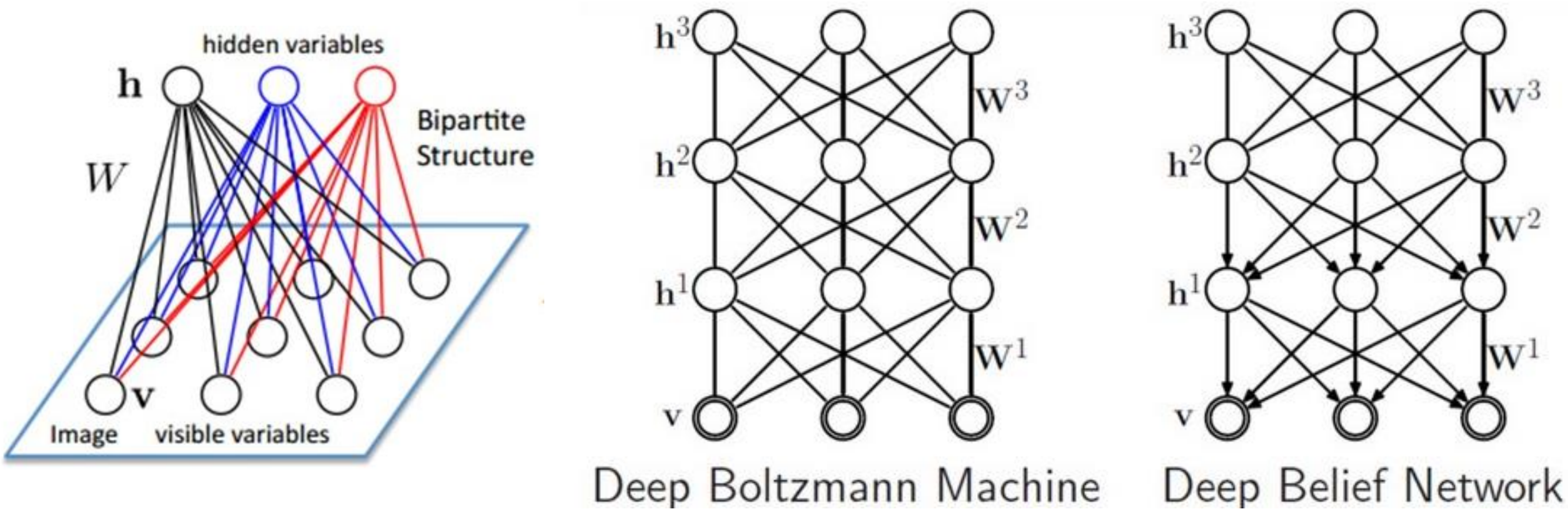
深度网络的训练

- 逐层的无监督训练：由最底层开始；
- fine-tune：基于第1步得到的各层参数，整体优化整个网络的参数



深度学习的常用模型

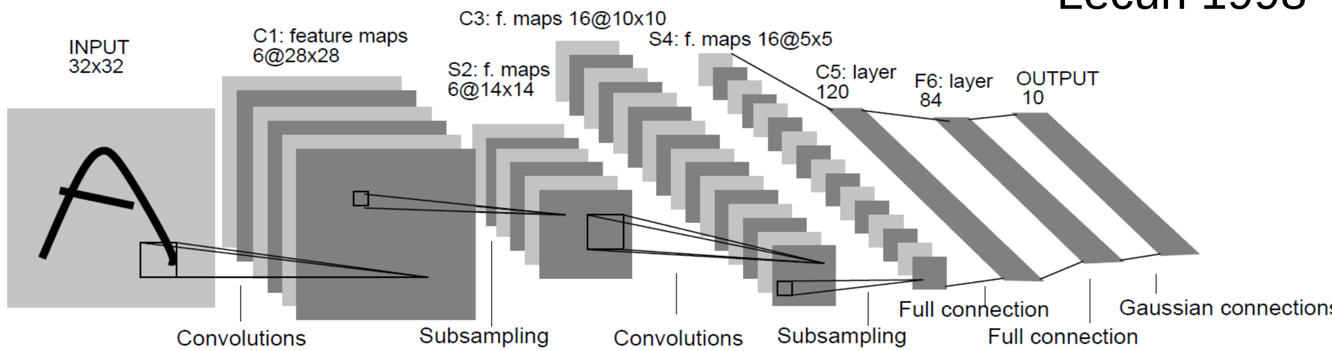
- 受限的玻尔兹曼机(RBM, Restricted Boltzmann Machine)

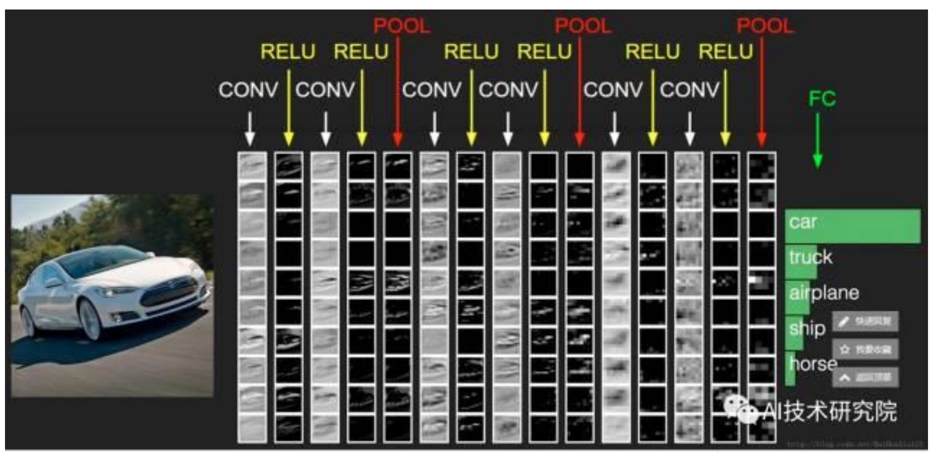


深度学习的常用模型

- 卷积神经网络(Convolutional Neural Networks)
  –第一个真正成功训练的多层网络结构

Lecun 1998

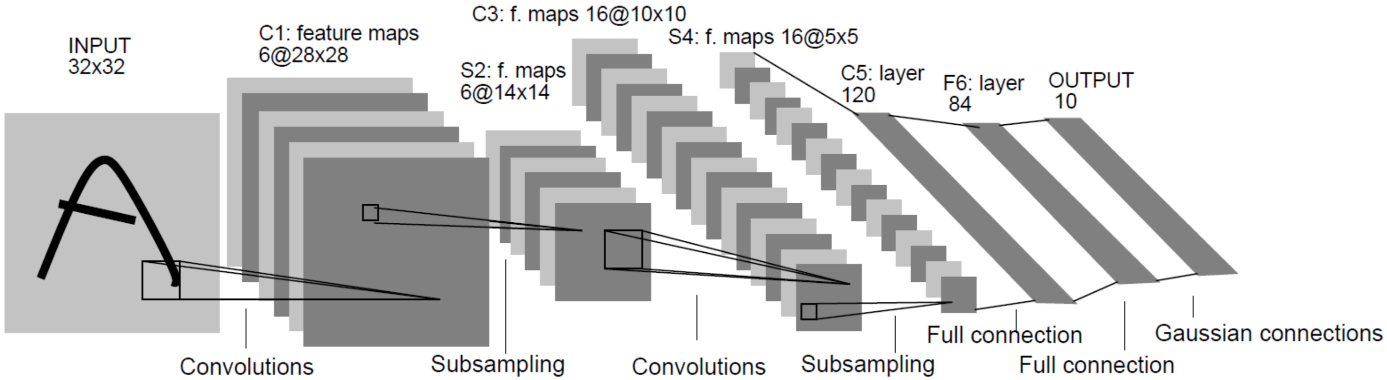


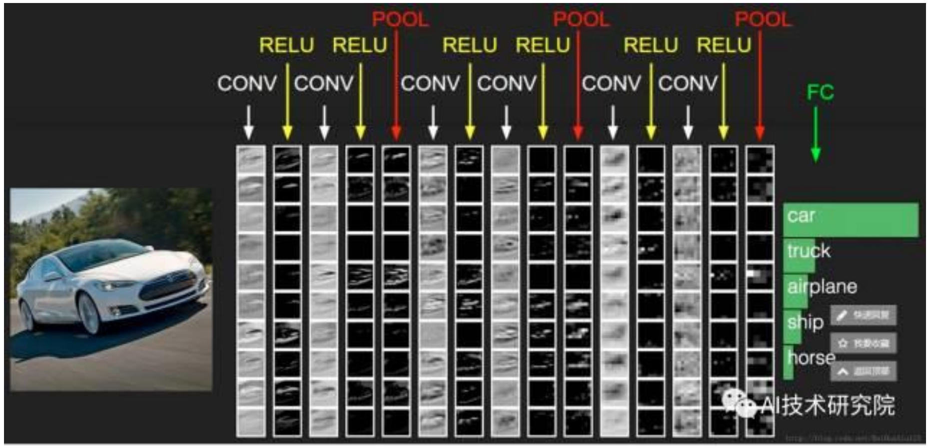


CNN基本结构

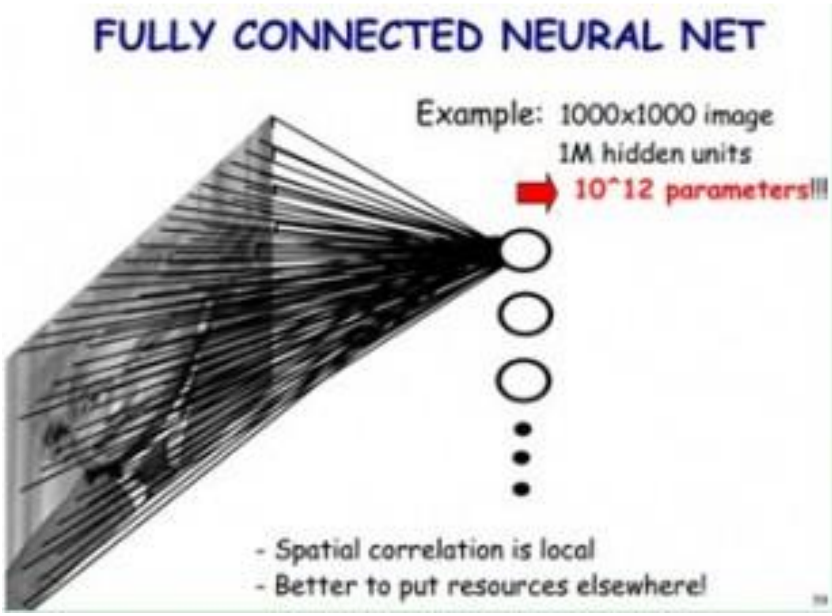
CNN基本网络：

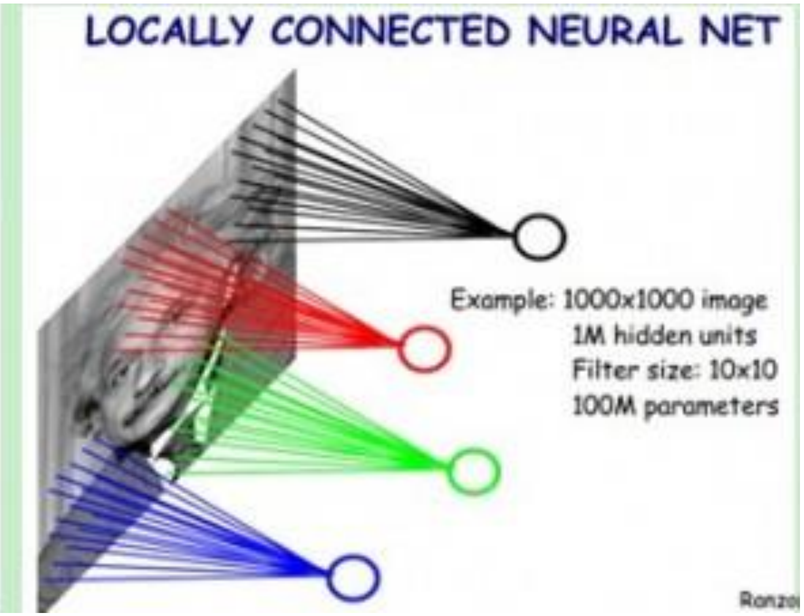
- 输入层、卷积层、激活层、池化层、全连接层





CNN—全连接与局部连接



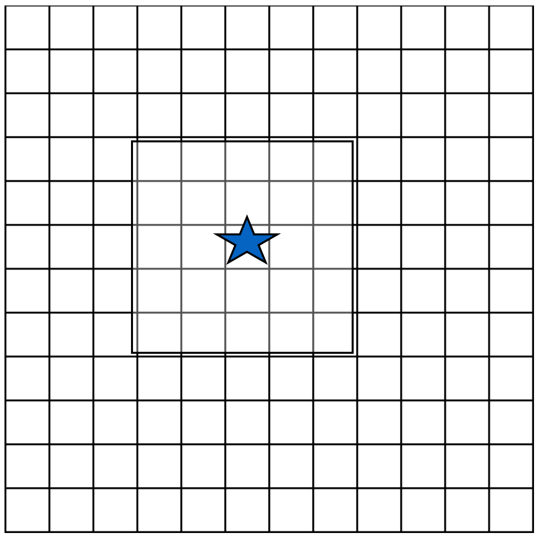


卷积运算

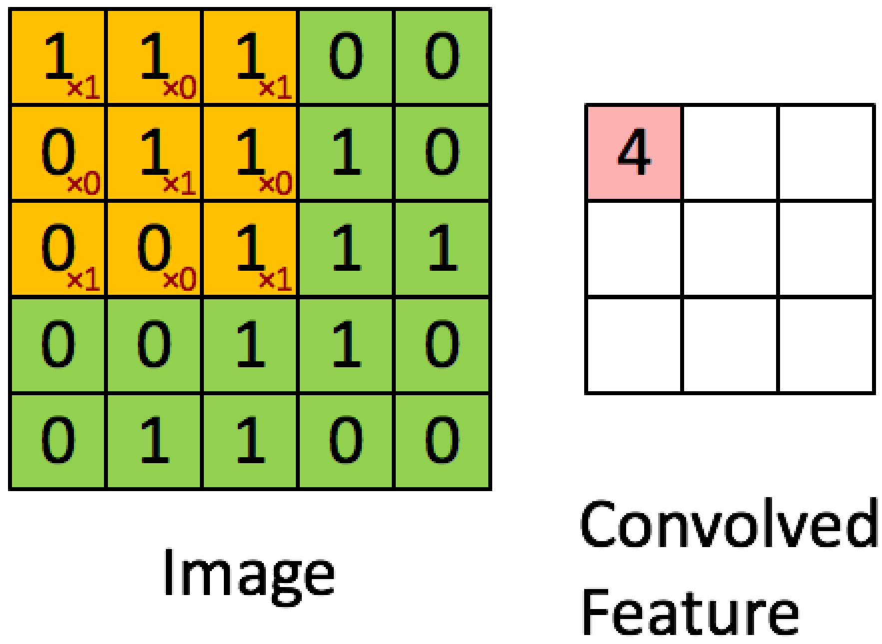
- 图像（二维数字信号的卷积运算）

$$G(x,y)*I(x,y)=\sum_{u=-M}^{M}\sum_{v=-N}^{N}G(u,v)I\,(x-u,y-v)$$

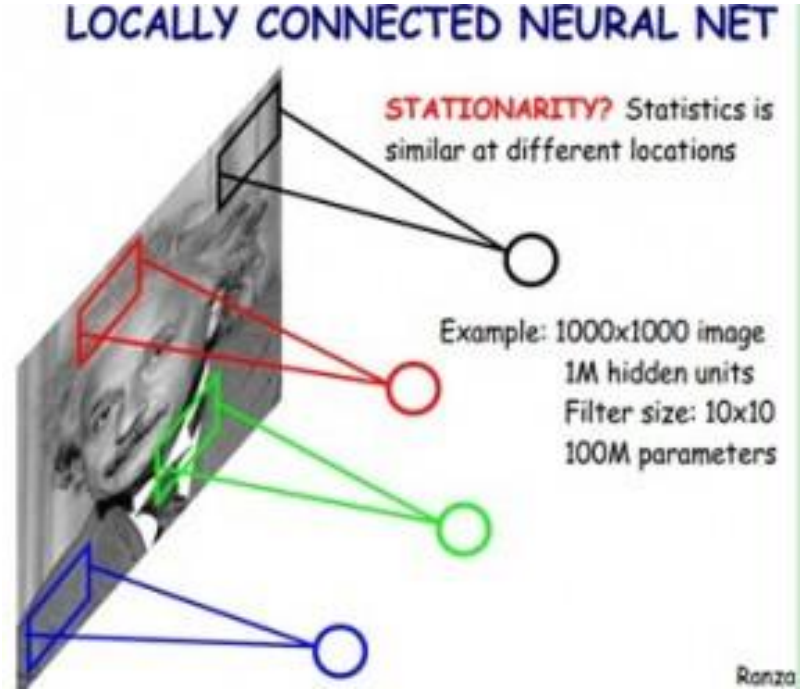
- 尺寸为（*2M+1*） ×（*2N+1*）的模版
- 是对连续卷积核函数的数字采样近似

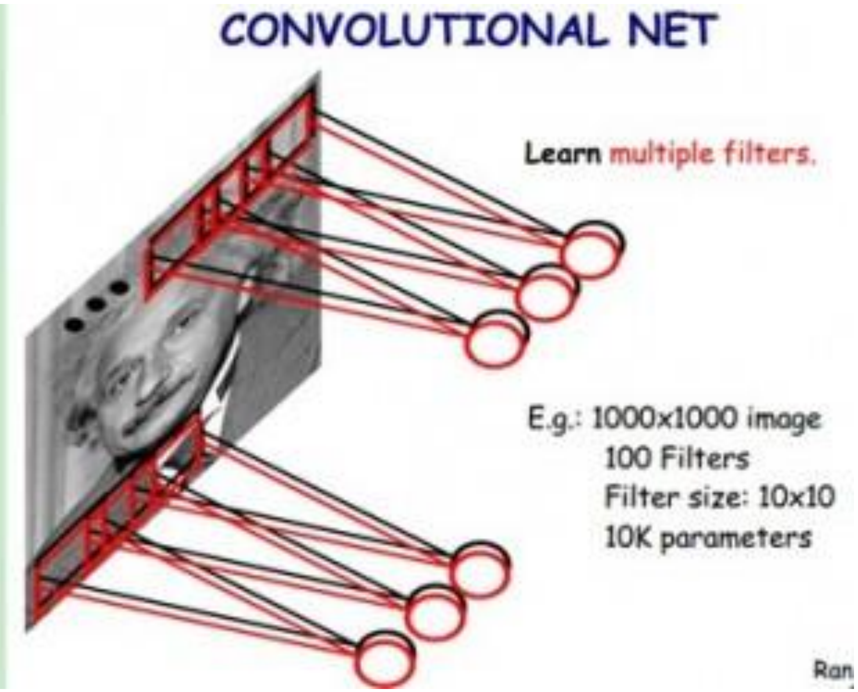


CNN—全连接与局部连接

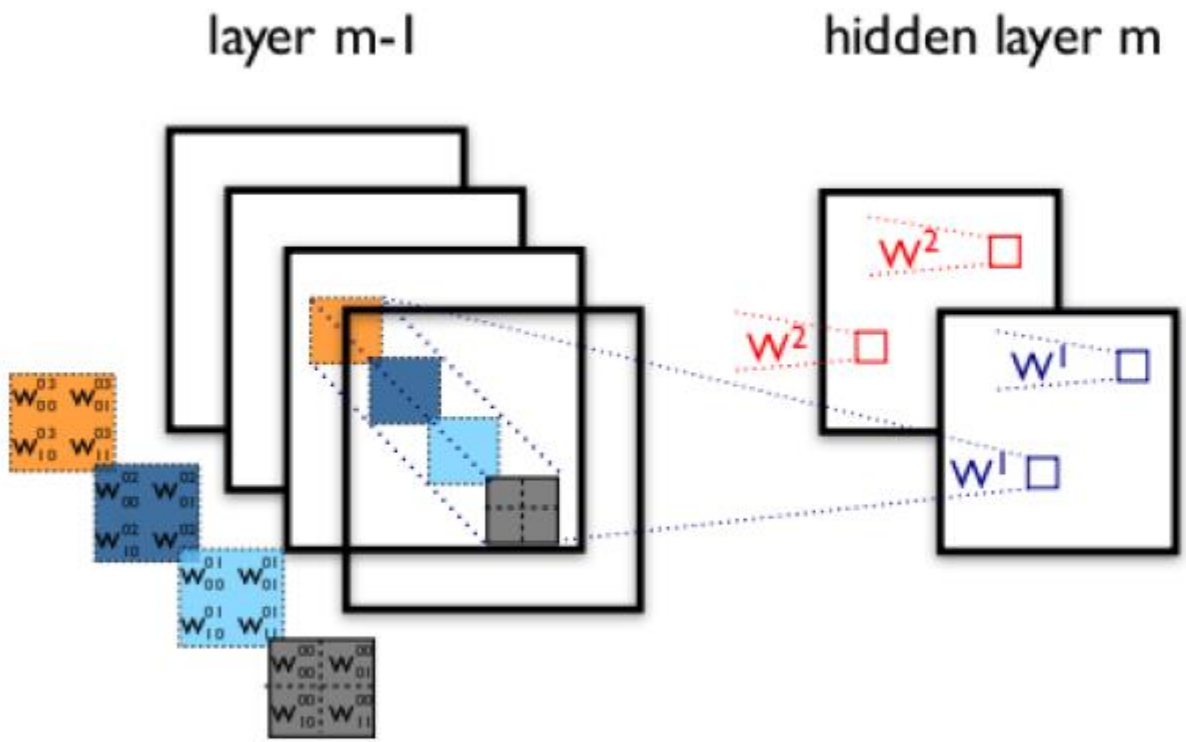


CNN—权值共享与多核卷积

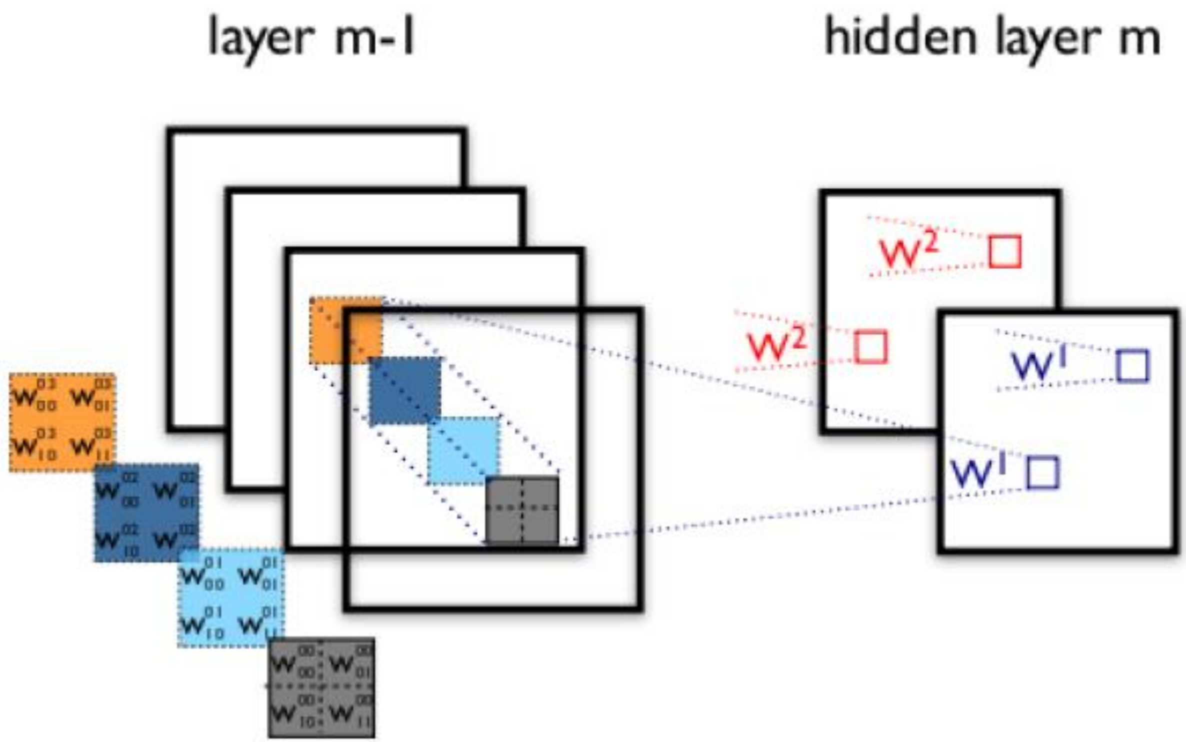




CNN—多核卷积



CNN—多核卷积



CNN—多核卷积

Input Volume (+pad 1) (7x7x3)

x[:,:,0]

0	0	0	0	0	0	0
0	0	1	1	2	2	0
0	0	1	1	0	0	0
0	1	1	0	1	0	0
0	1	0	1	1	1	0
0	0	2	0	1	0	0
0	0	0	0	0	0	0

x[:,:,1]

0	0	0	0	0	0	0
0	1	1	1	2	0	0
0	0	2	1	1	2	0
0	1	2	0	0	2	0
0	0	2	1	2	1	0
0	2	0	1	2	0	0
0	0	0	0	0	0	0

x[:,:,2]

0	0	0	0	0	0	0
0	2	0	2	0	2	0
0	0	0	1	2	1	0
0	1	0	2	2	1	0
0	2	0	2	0	0	0
0	0	0	1	1	2	0
0	0	0	0	0	0	0

Filter W0 (3x3x3)

w0[:,:,0]

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

w0[:,:,1]

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

w0[:,:,2]

0	1	0
1	0	1
0	-1	1

Bias b0 (1x1x1)

b0[:,:,0]

1

Filter W1 (3x3x3)

w1[:,:,0]

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

w1[:,:,1]

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

w1[:,:,2]

-1	0	1
1	0	1
0	-1	0

Bias b1 (1x1x1)

b1[:,:,0]

0

Output Volume (3x3x2)

o[:,:,0]

1	0	-3
-6	1	1
4	-3	1

o[:,:,1]

-1	-6	-4
-2	-3	-4
-1	-3	-3

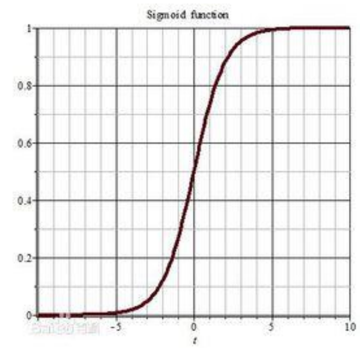
toggle movement

CNN中常用的激活函数

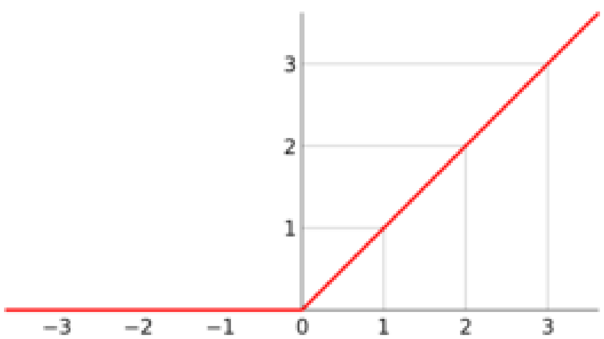
CNN中常用的激活函数

- Sigmoid函数
- Tanh函数
- ELU
- Maxout
- ReLU（Rectified Linear Unit）
- Leaky ReLU

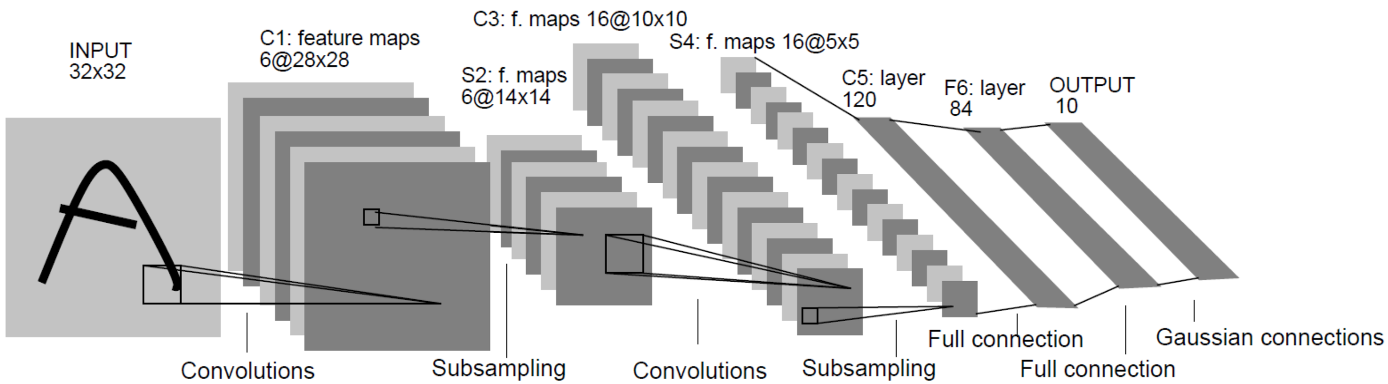
$$S(x) = \frac{1}{1+e^{-x}}$$

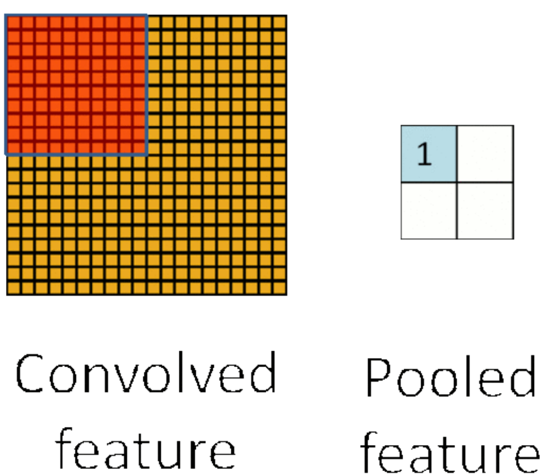


$$f(x) = \max(0, x),$$



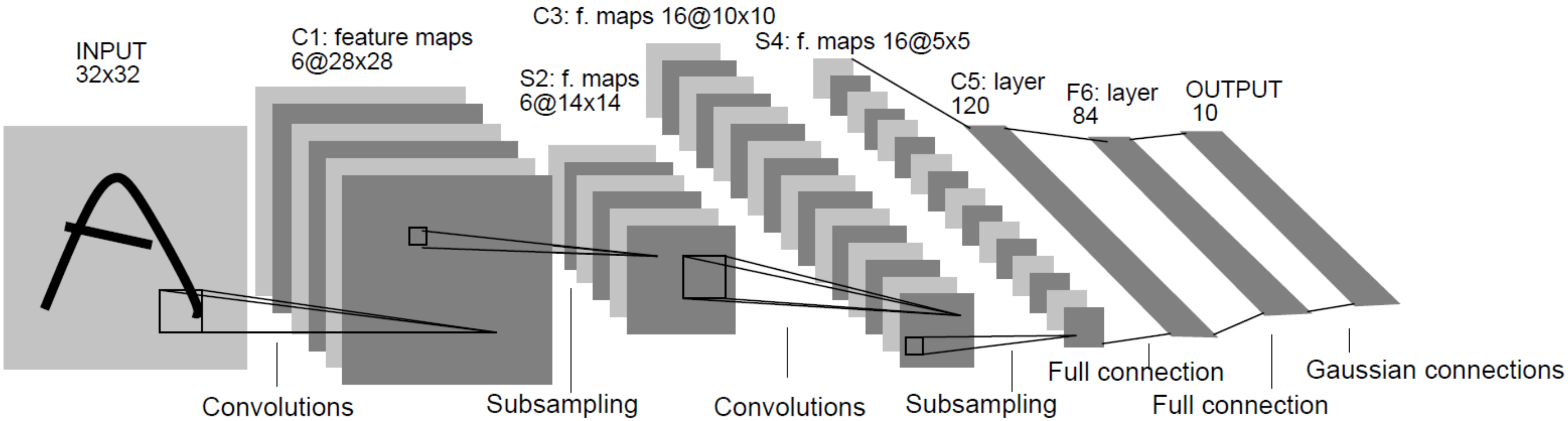
CNN—Pooling(池化)





池化主要作用：特征降维、避免过拟合

CNN再回顾



INPUT
32x32
C1: feature maps
6@28x28
S2: f. maps
6@14x14
C3: f. maps 16@10x10
S4: f. maps 16@5x5
C5: layer
120
F6: layer
84
OUTPUT
10
Convolutions
Subsampling
Convolutions
Subsampling
Full connection
Full connection
Gaussian connections

开发平台

- **Caffe：**
  - 开发者：Berkeley Vision and Learning Center
  - 底层语言：C++
  - 接口语言：命令行，Python，Matlab
  - 运行方式：CPU or GPU
  - 操作系统：Windows or Linux

开发平台

- Torch
  - 开发者：Facebook
  - 底层语言：脚本语言LuaJit，底层C/CUDA
  - 接口语言：Lua语言
  - 运行方式：CPU or GPU
  - 操作系统：Windows or Linux

torch

A SCIENTIFIC COMPUTING FRAMEWORK FOR LUAJIT

开发平台

- TensorFlow
  - 开发者：Google Brain Team
  - 底层语言：C++ Python
  - 接口语言：Python C/C++
  - 运行方式：CPU or GPU
  - 操作系统：Linux or Mac OS X

开发平台

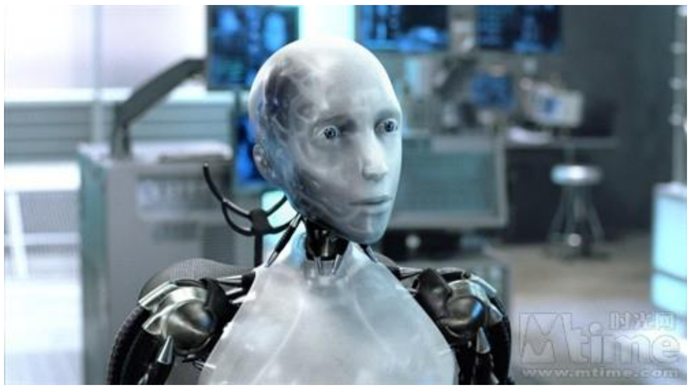
- MXNet
  - 开发者：分布式（深度）机器学习社区
  - 底层语言：C++
  - 接口语言：C++ Python Julia Matlab JavaScript R Scala
  - 运行方式：CPU or GPU
  - 操作系统：Linux OS X Windows

提纲

1 深度学习与卷积神经网络

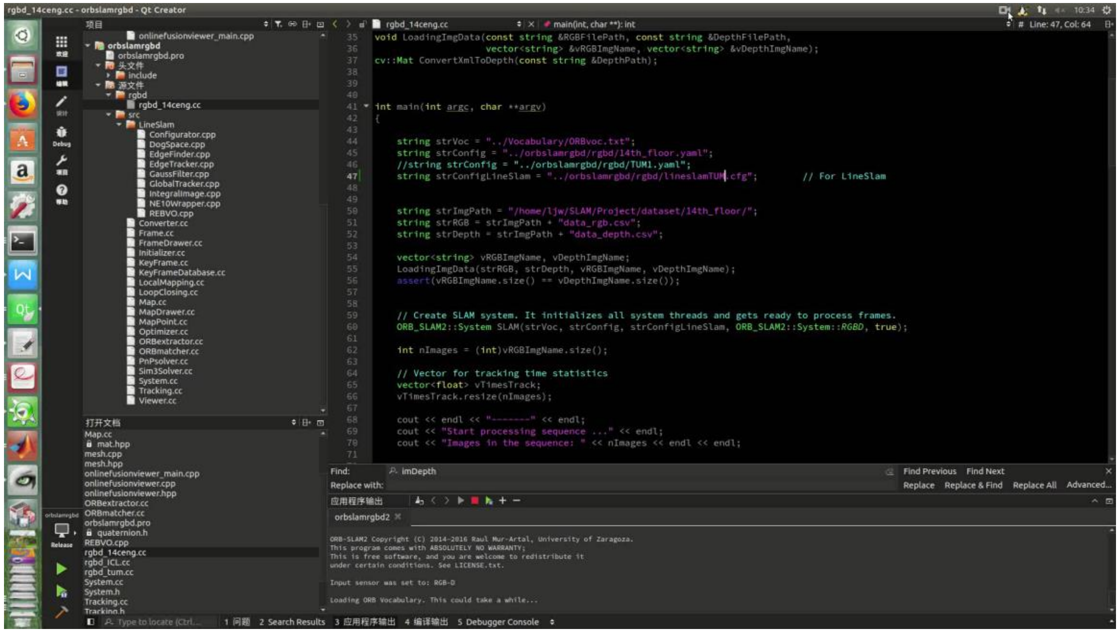
2 图像底层特征提取

计算机的“眼睛”在哪?

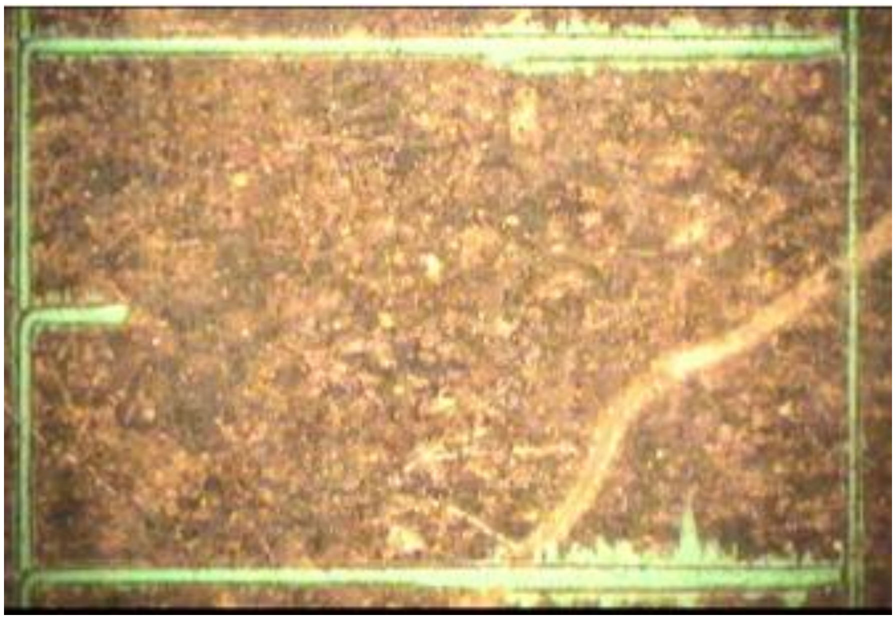


特征提取是计算机视觉的基本步骤

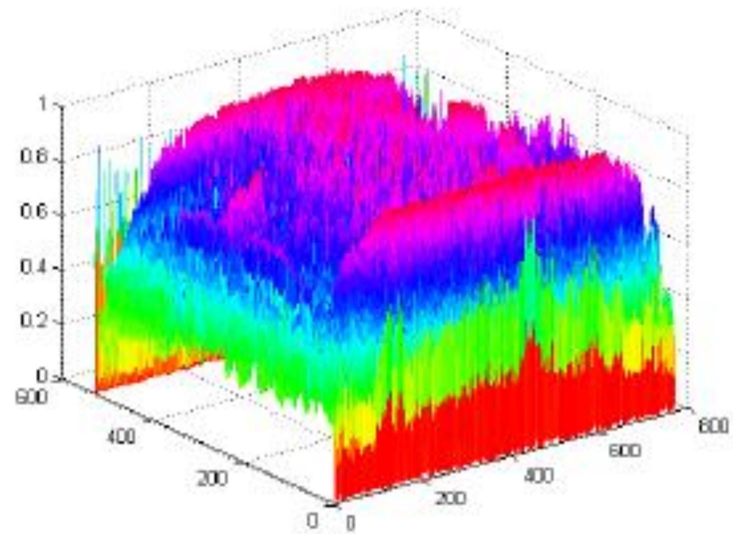
- 边缘和轮廓能反映图像内容；
- 如果能对边缘和关键点可靠提取的话，很多视觉问题就基本上得到了解决



灰度图像



真实的图像



图像数据的表示

灰度值范围：0~255

边缘的物理意义

图像边缘的产生

- **物体的边界、表面方向的改变、不同的颜色、光照明暗的变化**

边缘提取

- 边缘的定义（WHAT）
- 提取边缘的意义（WHY）
- 提取边缘的方法（HOW）

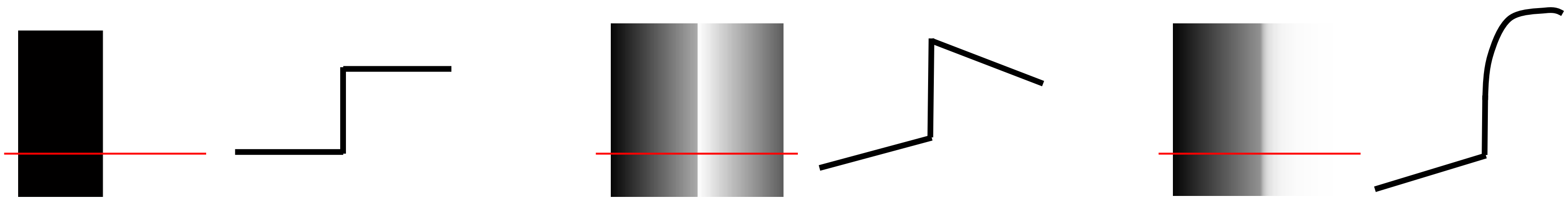
■ 使用微分滤波器提取边缘
  ■ 一阶微分滤波器：梯度算子
  ■ 二阶微分滤波器：LoG
■ Canny算子

边缘的定义

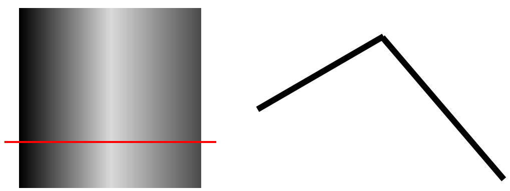
定义：

- “边缘是图像中亮度突然变化的区域。”
- “图像灰度构成的曲面上的陡峭区域。”
- “像素灰度存在阶跃变化或屋脊状变化的像素的集合。”

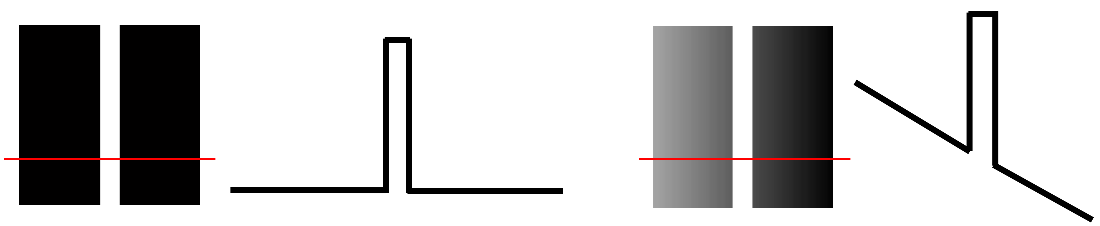
灰度图像中边缘的类型



阶梯状边缘



屋脊状边缘



线条状边缘

为什么要提取边缘？

边缘是最基本的图像特征之一：

- ■可以表达物体的特征
- ■边缘特征对于图像的变化不敏感
  - ●几何变化，灰度变化，光照方向变化
- ■可以为物体检测提供有用的信息
- ■是一种典型的图像预处理过程

如何提取边缘？（灰度图象）

灰度图象边缘提取思路：

■抑制噪声（低通滤波、平滑、去噪、模糊）

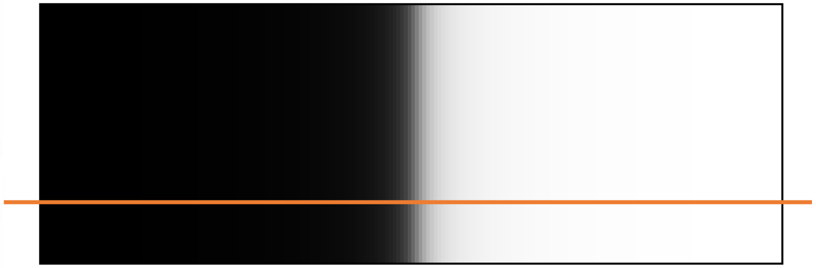
■边缘特征增强（高通滤波、锐化）

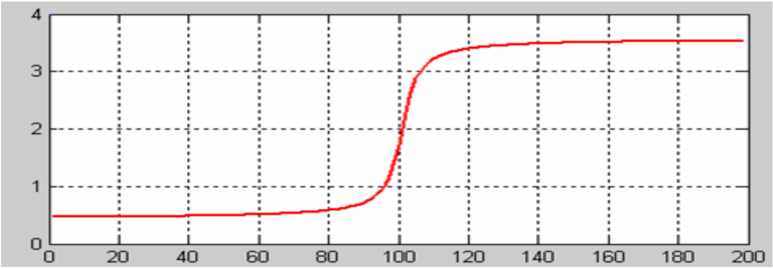
■边缘定位

图像微分算子

- ■一阶微分算子（梯度算子）：Prewitt、Sobel等
  - 检测最大值
- ■二阶微分算子：Laplacian
  - 检测过零点

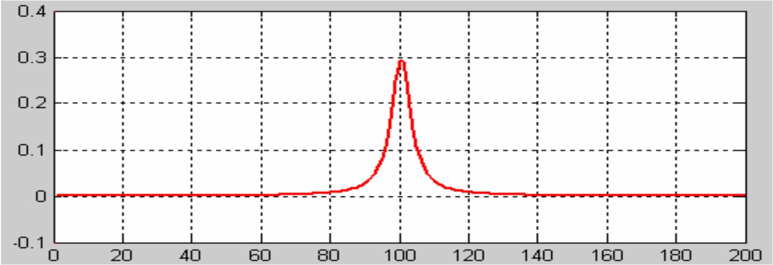
微分算子检测边缘：一维信号





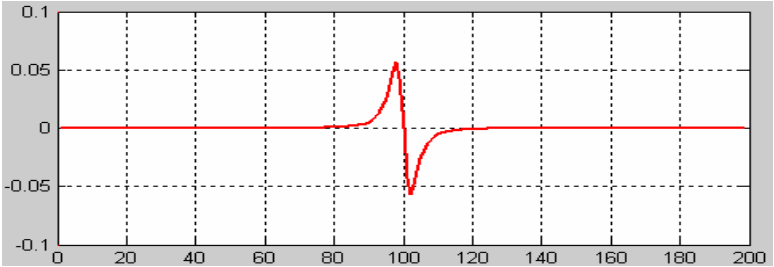
一阶导数的极大值点：

$$\text{Edge} = \{x \mid x = \text{argmax}(f'(x))\}$$



二阶导数的过零点：

$$\text{Edge} = \{x \mid f''(x) = 0, \text{zero crossing}\}$$



微分算子检测边缘：二维信号

一阶导数的极大值点：

$$\mathrm{Edge} = \{p = (x, y) | p = \operatorname{argmax}(|\nabla I(p)|)\}$$

其中，图像梯度向量： $\nabla I(x, y) = (\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y})$

梯度幅值表示边缘的强弱　　**梯度方向代表灰度变化最快的方向**

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

$$\Psi(x, y) = \arctan(\frac{\partial I}{\partial y} / \frac{\partial I}{\partial x})$$

微分算子检测边缘：二维信号

二阶导数的过零点：

$$\mathrm{Edge} = \{p = (x, y) | \Delta I(p) = 0, \text{zero crossing}\}$$

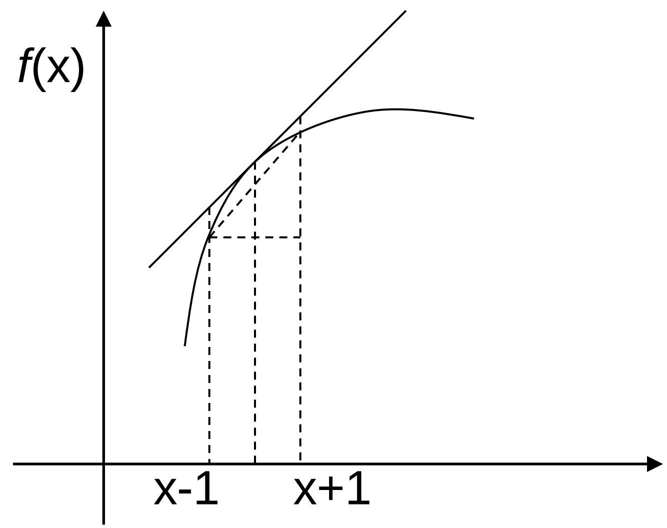
拉普拉斯算子：

$$\Delta I = \nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

在数字图像上计算梯度

- 一维的情况：

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



对于离散的数字信号，可以使用差分近似：

$$f'(x) \approx \frac{f(x+1) - f(x-1)}{2}$$

相当于与如下模版进行卷积运算：

-1	0	1

×0.5

在数字图像上计算梯度

- **使用差分运算在数值上近似一阶微分运算**

$$\frac{\partial I}{\partial x} \approx \frac{f(x+1,y)-f(x-1,y)}{2}$$

竖直边缘

$$\begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline \end{array} \times 0.5$$

$$\frac{\partial I}{\partial y} \approx \frac{f(x,y+1)-f(x,y-1)}{2}$$

水平边缘

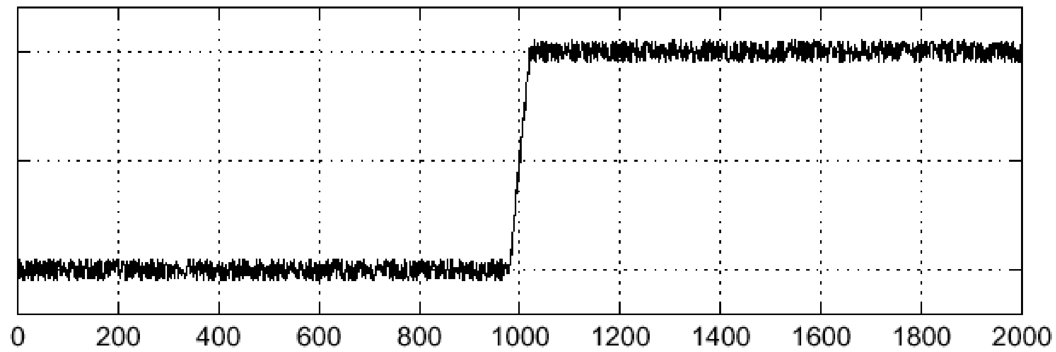
$$\begin{array}{|c|}\hline -1 \\ \hline 0 \\ \hline 1 \\ \hline \end{array} \times 0.5$$

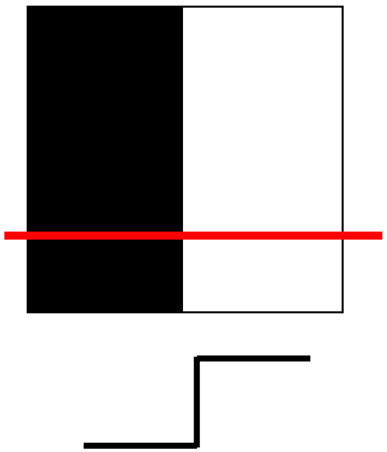
$$\nabla I(x,y) = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right)$$

噪声的影响：一维信号的例子

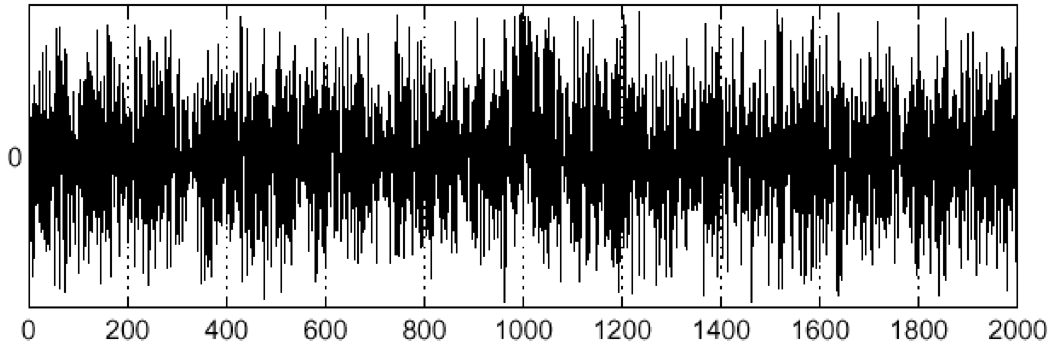
从图像中取出某行像素值：

$f(x)$





$\frac{d}{dx}f(x)$



边缘在哪里？

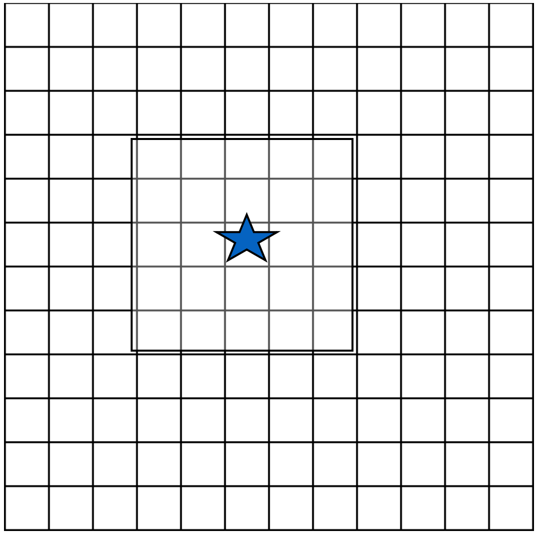
补充知识：卷积运算

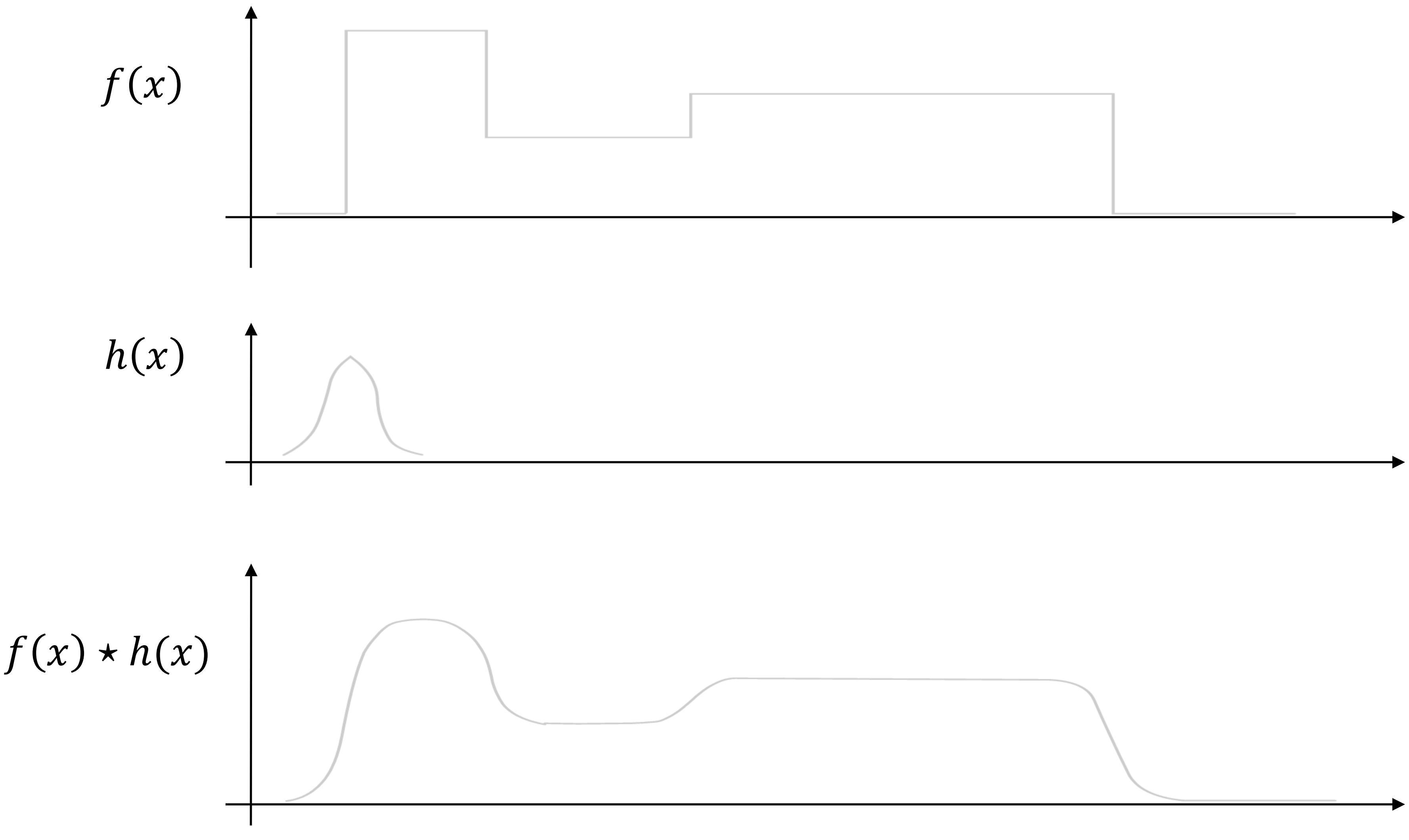
图像（二维数字信号的卷积运算）

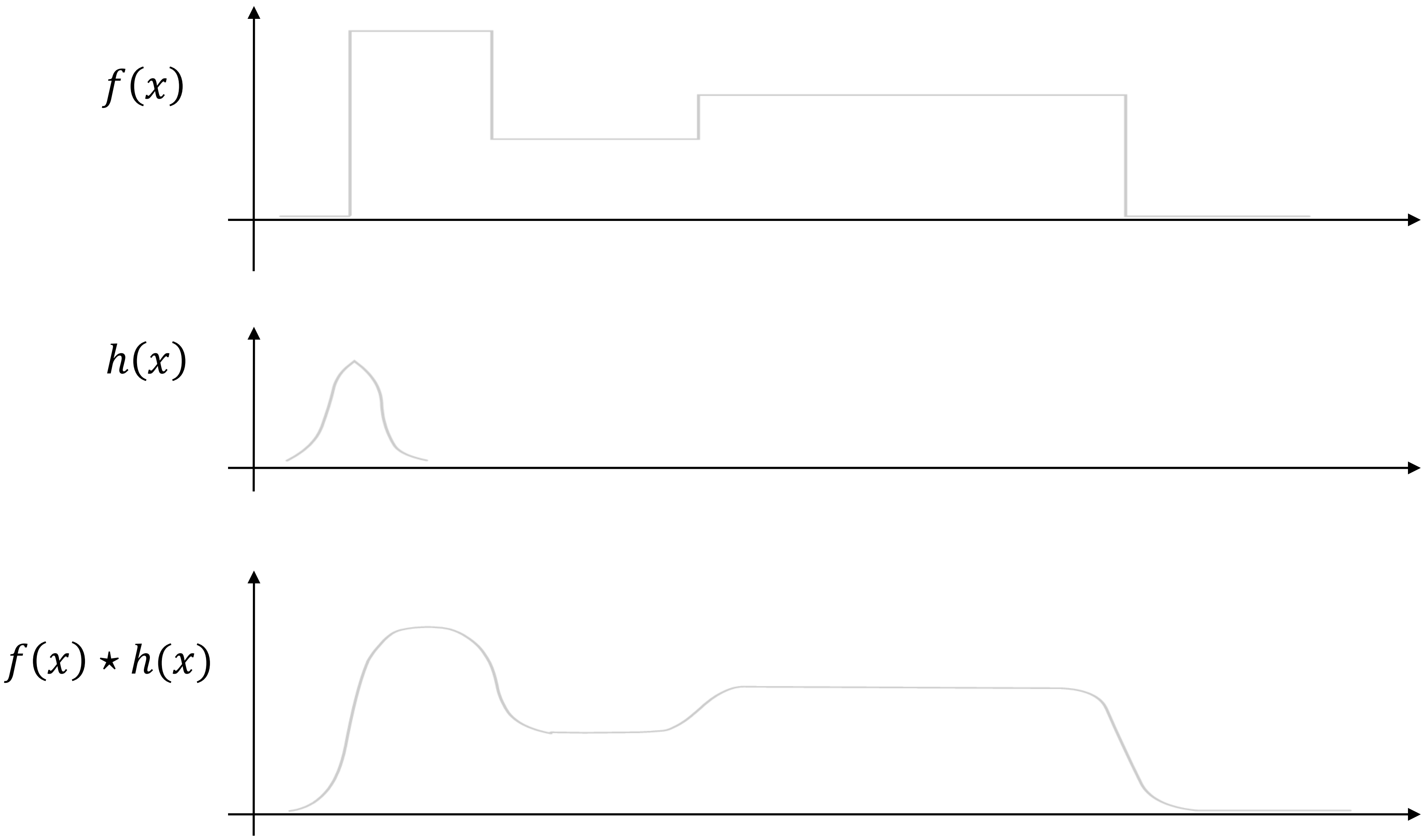
$$G(x,y)\star I(x,y)=\sum_{u=-M}^{M}\sum_{v=-N}^{N}G(u,v)I(x-u,y-v)$$

- **尺寸为（*2M+1*）×（*2N+1*）的模版**
- **是对连续卷积核函数的数字采样近似**
- **不同模版形式决定了卷积的不同功能**

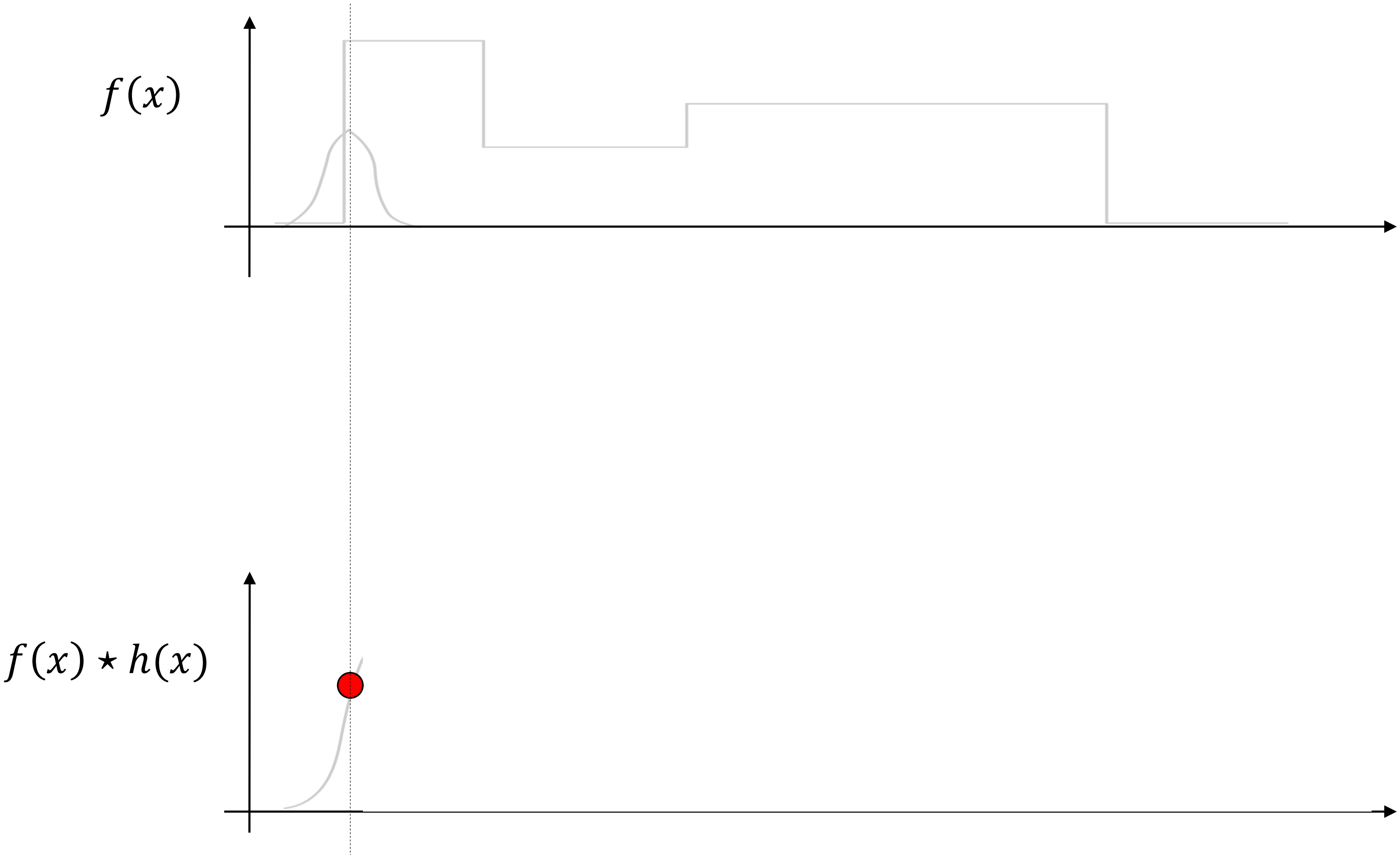
--滤波、增强、匹配……



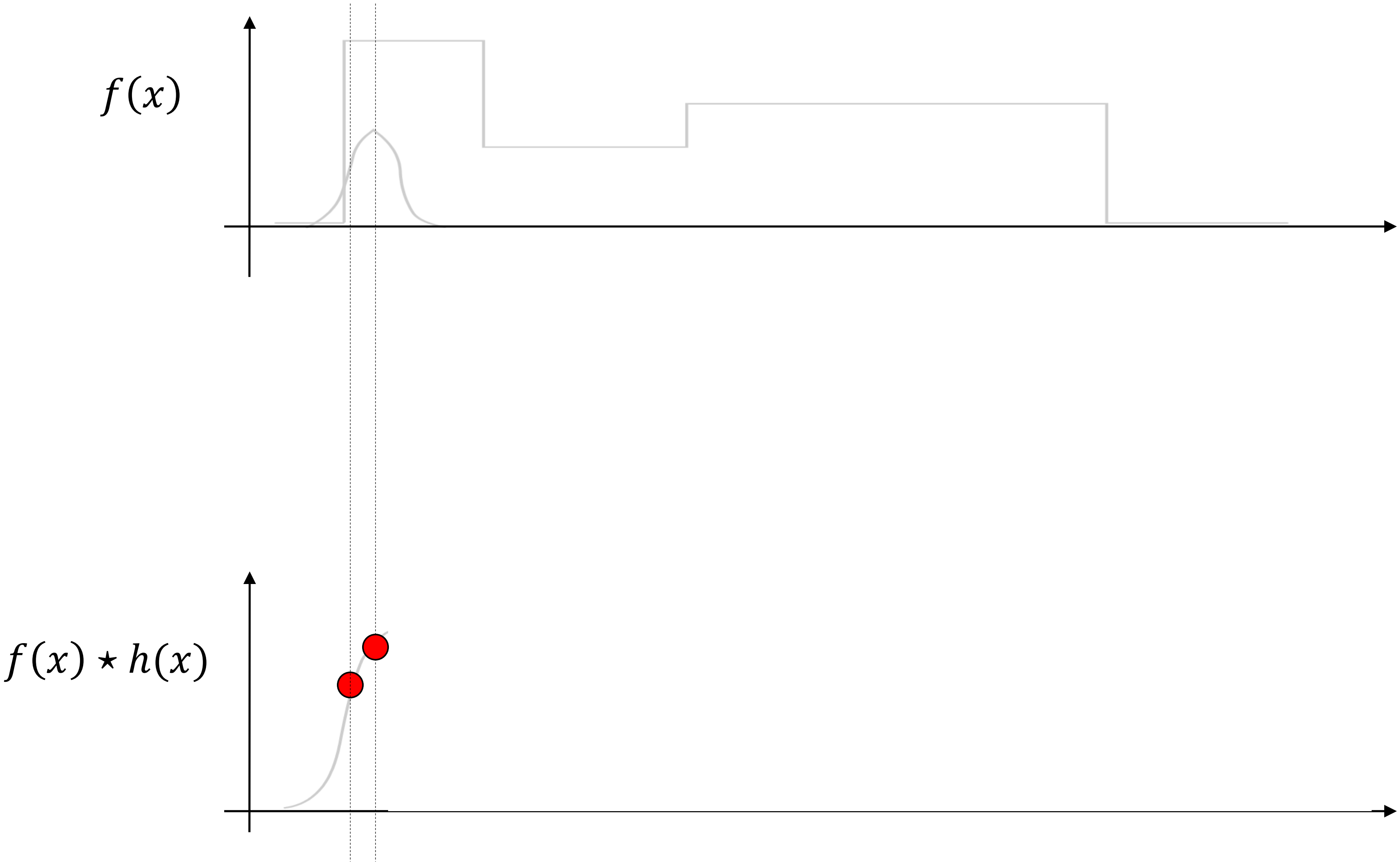

$f(x)$
$h(x)$
$f(x) \star h(x)$



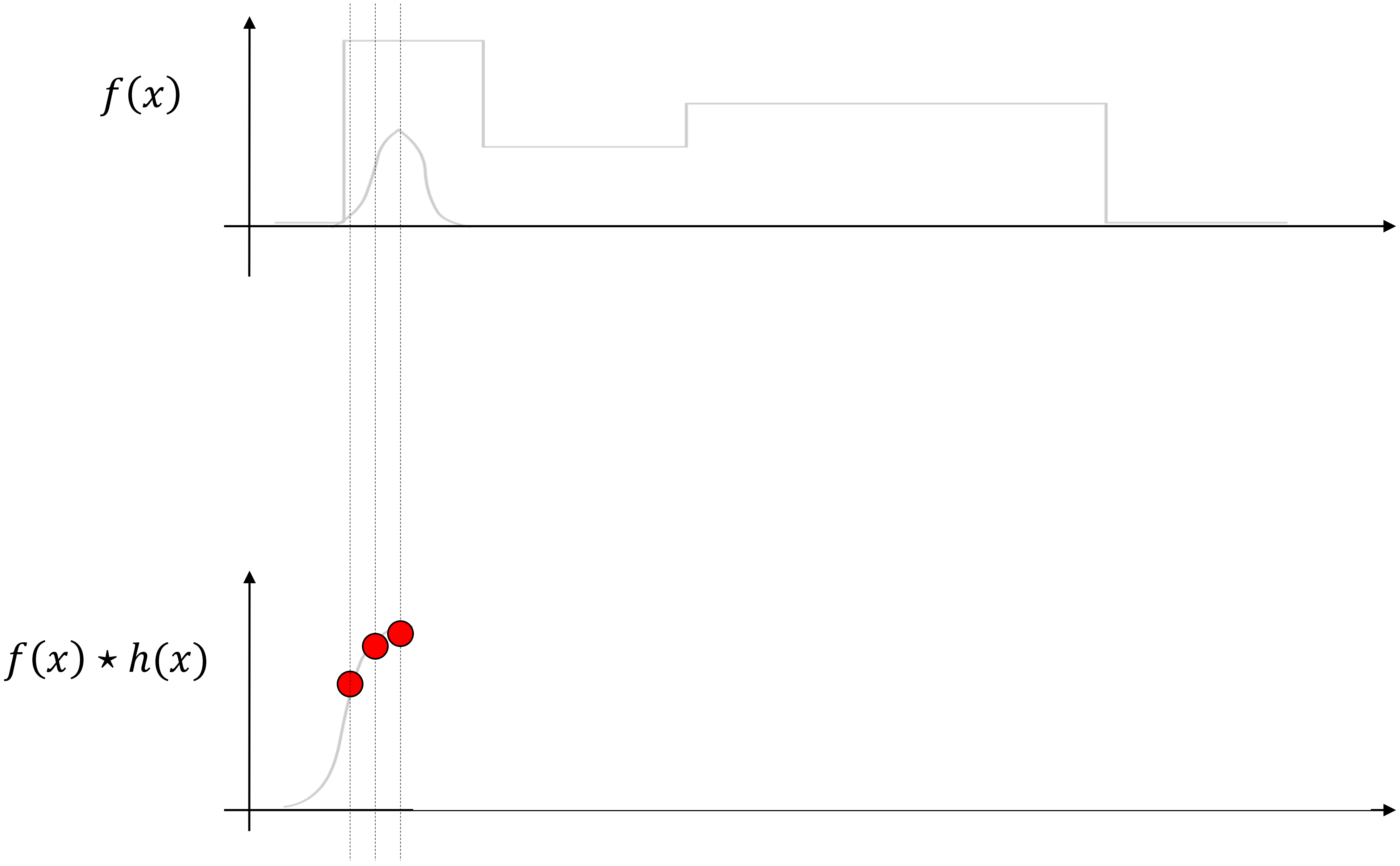
$f(x)$
$h(x)$
$f(x) \star h(x)$



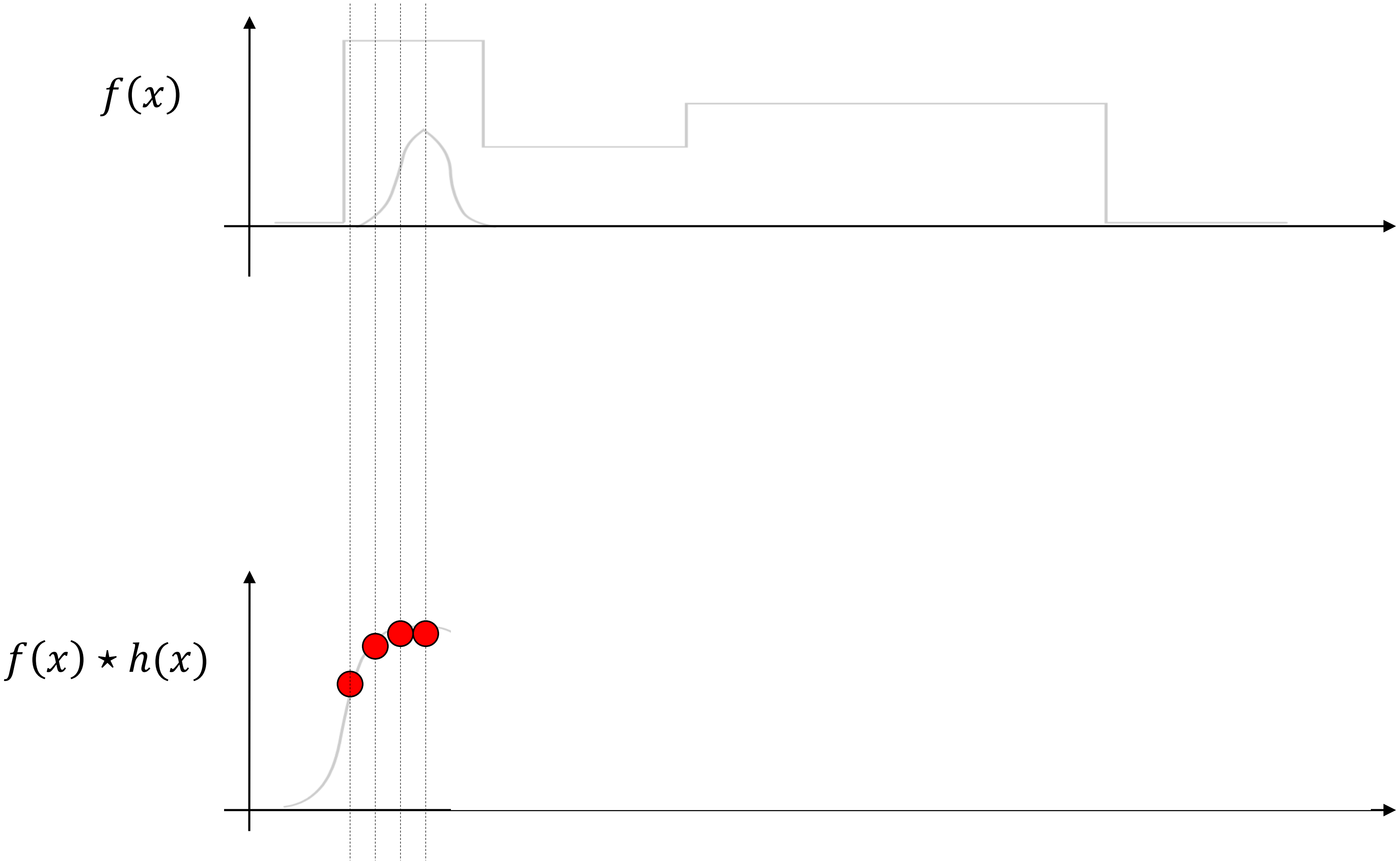
$f(x)$
$f(x) \star h(x)$

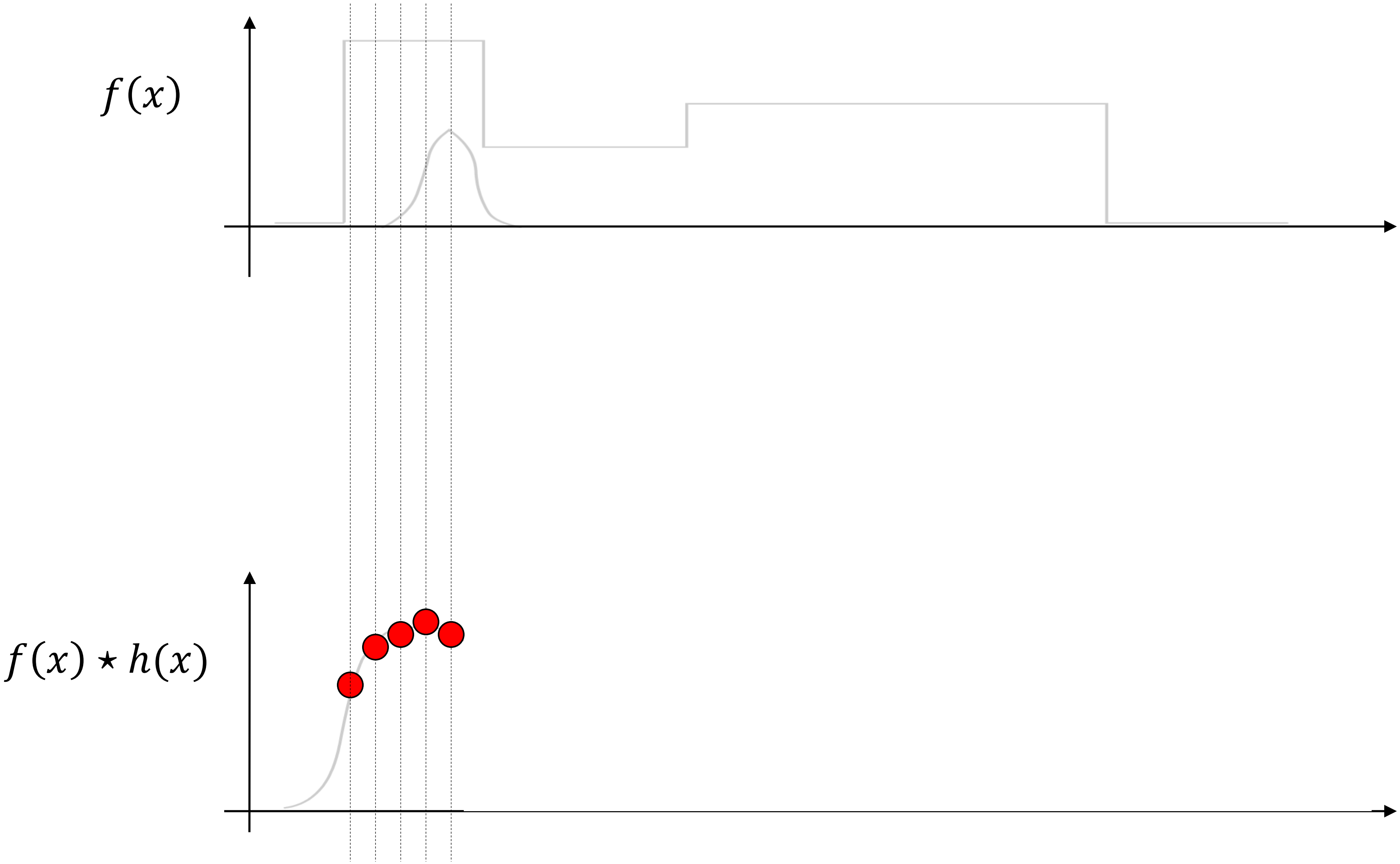


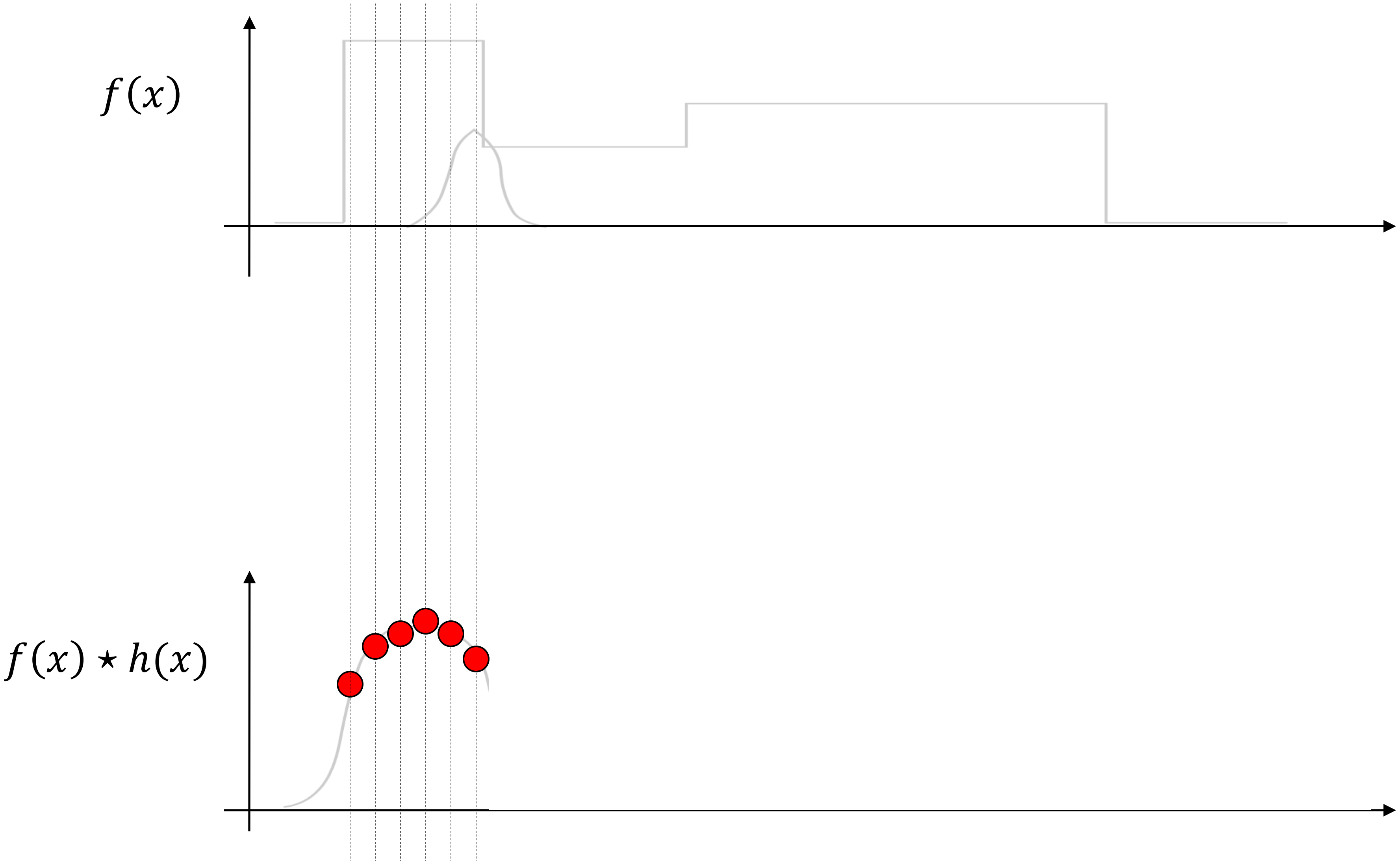
$f(x)$
$f(x) \star h(x)$

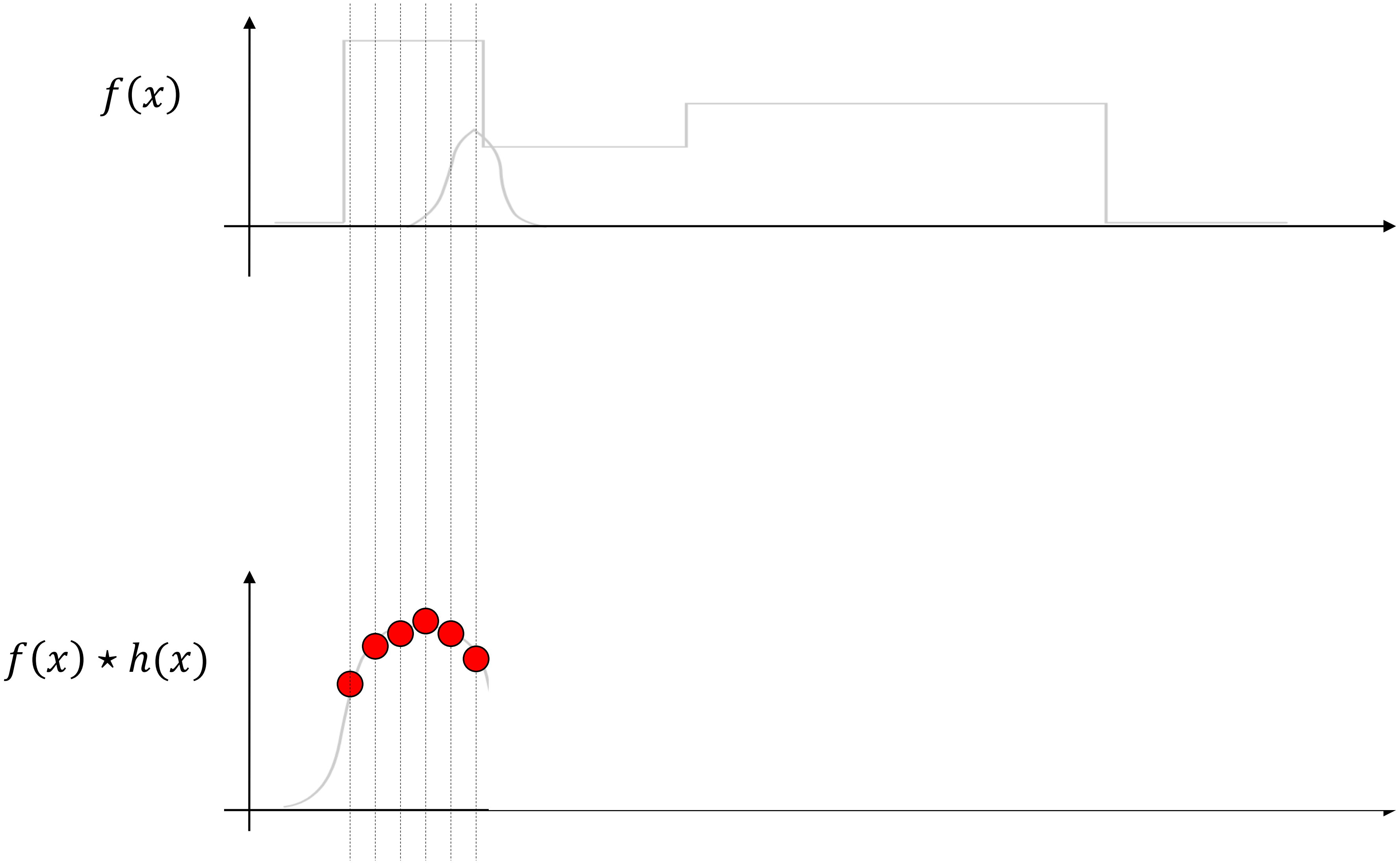


$f(x)$
$f(x) \star h(x)$


$f(x)$
$f(x) \star h(x)$


$f(x)$
$f(x) \star h(x)$


$f(x)$
$f(x) \star h(x)$



$f(x)$
$f(x) \star h(x)$

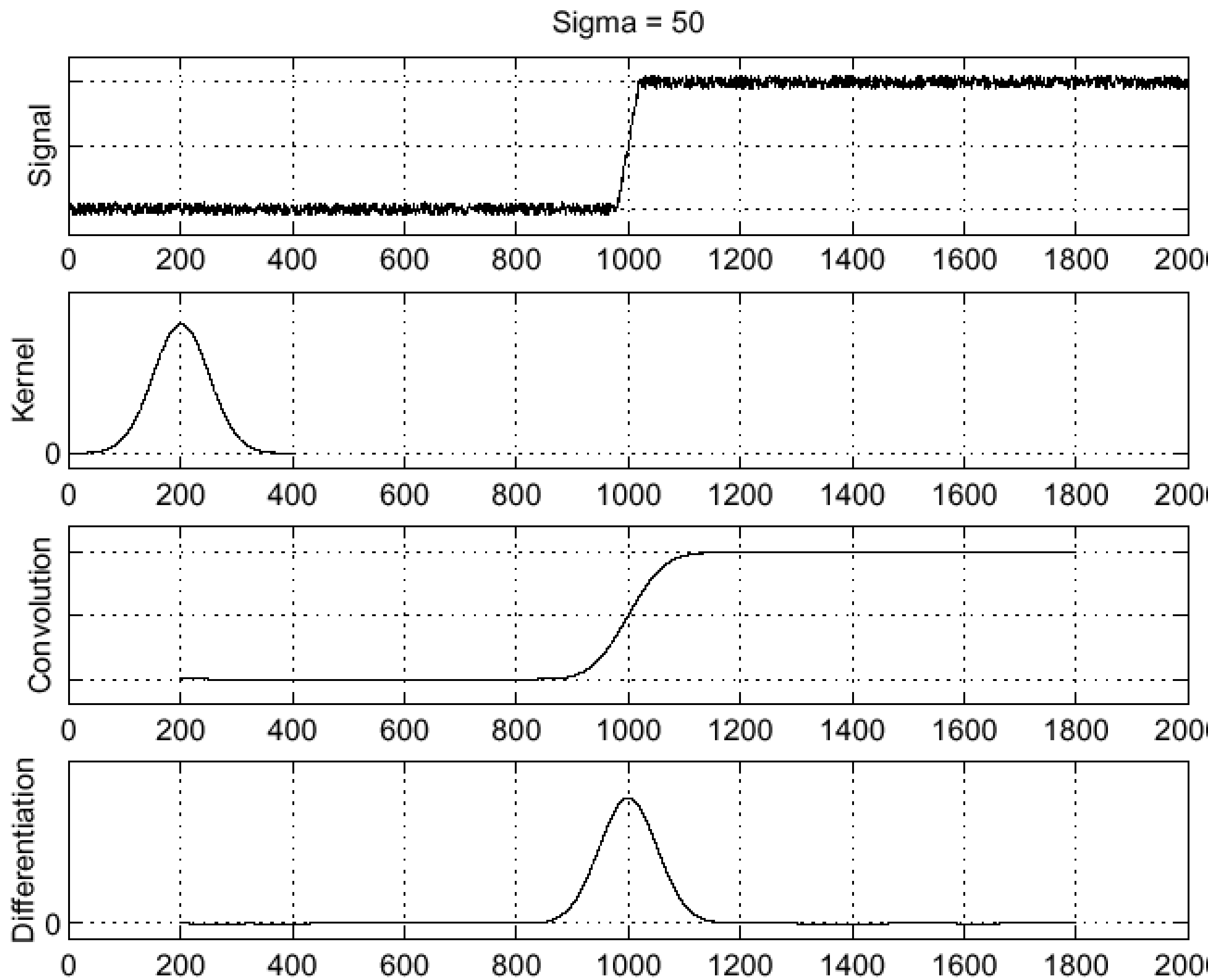
解决方法之一，首先进行滤波

$$f$$

$$h$$

$$h \star f$$

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



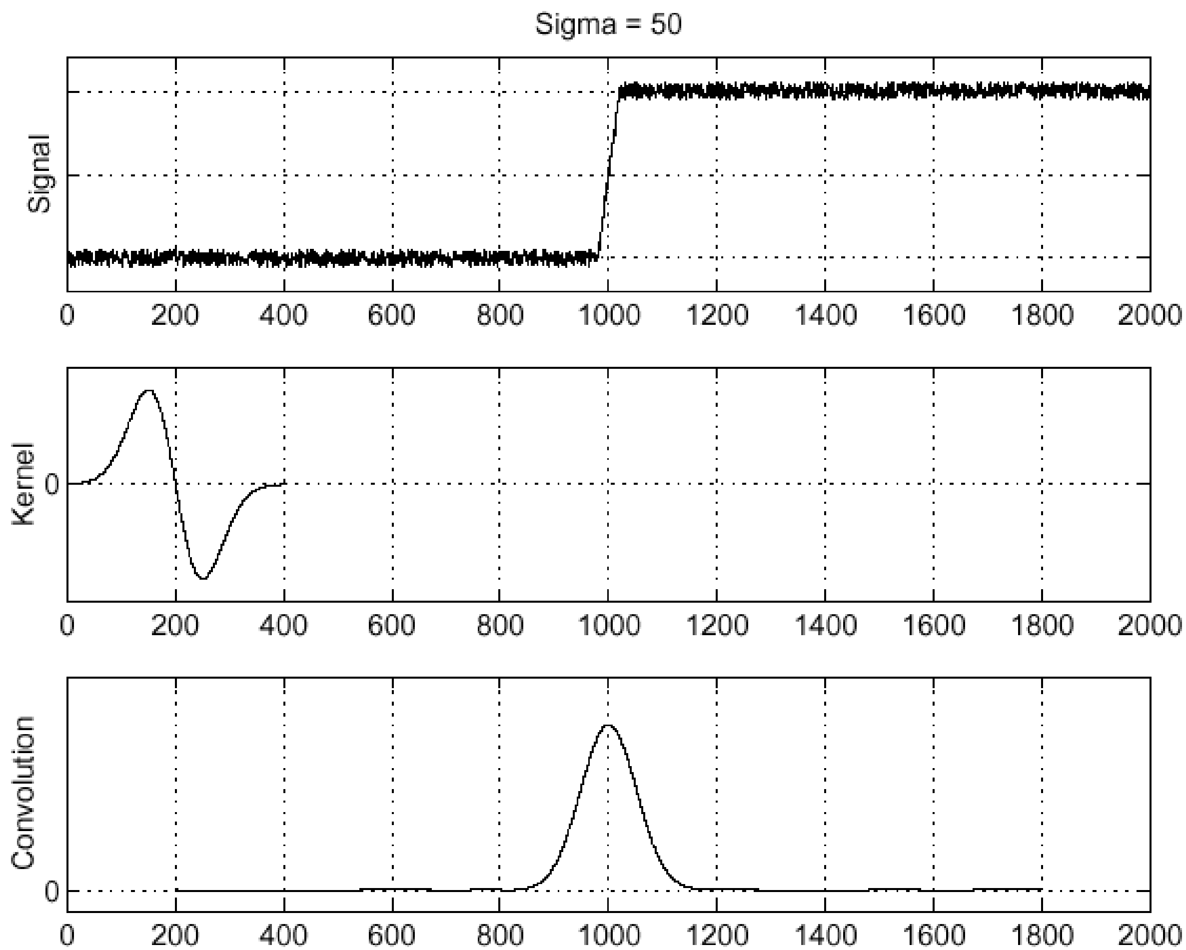
利用卷积运算的性质：

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

$$f$$

$$\frac{\partial}{\partial x}h$$

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



图像梯度算子的近似

- Sober算子
- Prewitt算子
- Roberts算子

Prewitt算子

- Prewitt算子，近似一阶微分
- 卷积模版：去噪 + 增强边缘

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

计算均值，
平滑噪声

检测竖直边缘

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

检测水
平边缘

计算均值，
平滑噪声

Sobel算子

- **Sobel**算子，近似一阶微分
- 去噪 + 增强边缘，给四邻域更大的权重

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

计算均值，
平滑噪声

检测竖直边缘

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

检测水平边缘

计算均值，
平滑噪声

常见的梯度算子

$$\Delta_1 \quad \begin{matrix} 0 & 1 \\ -1 & 0 \end{matrix} \qquad \Delta_2 \quad \begin{matrix} 1 & 0 \\ 0 & -1 \end{matrix}$$

(a)

$$\Delta_1 \quad \begin{matrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{matrix} \qquad \Delta_2 \quad \begin{matrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{matrix}$$

(b)

$$\Delta_1 \quad \begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix} \qquad \Delta_2 \quad \begin{matrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{matrix}$$

(c)

$$\Delta_1 \quad \begin{matrix} -3 & -1 & 1 & 3 \\ -3 & -1 & 1 & 3 \\ -3 & -1 & 1 & 3 \\ -3 & -1 & 1 & 3 \end{matrix} \qquad \Delta_2 \quad \begin{matrix} 3 & 3 & 3 & 3 \\ 1 & 1 & 1 & 1 \\ -1 & -1 & -1 & -1 \\ -3 & -3 & -3 & -3 \end{matrix}$$

(d)

Sobel与Prewitt边缘提取示例

Prewitt

Sobel

在数字图像上计算二阶微分

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

- **拉普拉斯算子的数字近似**
  - **3*3卷积模版**

0	-1	0
-1	4	-1
0	-1	0

-1	-1	-1
-1	8	-1
-1	-1	-1

拉普拉斯算子的特点

- 拉普拉斯算子的运算结果是标量
  - 只有幅值，只使用一个模版便可计算得到
  - 方向属性丢失
- 实际中几乎不单独使用拉普拉斯算子：
  - 二次求导数，对噪声非常敏感
  - 通常配合滤波器同时使用

Laplacian of Gaussian (LoG)

- 首先用Gauss函数对图像进行平滑，抑制噪声
- 然后对经过平滑的图像使用Laplacian算子
- 利用卷积的性质LoG算子等效于：

Gaussian平滑 + Laplacian 二阶微分

$$\nabla^2\big(G(x,y) \star I(x,y)\big) = (\nabla^2 G(x,y)) \star I(x,y)$$

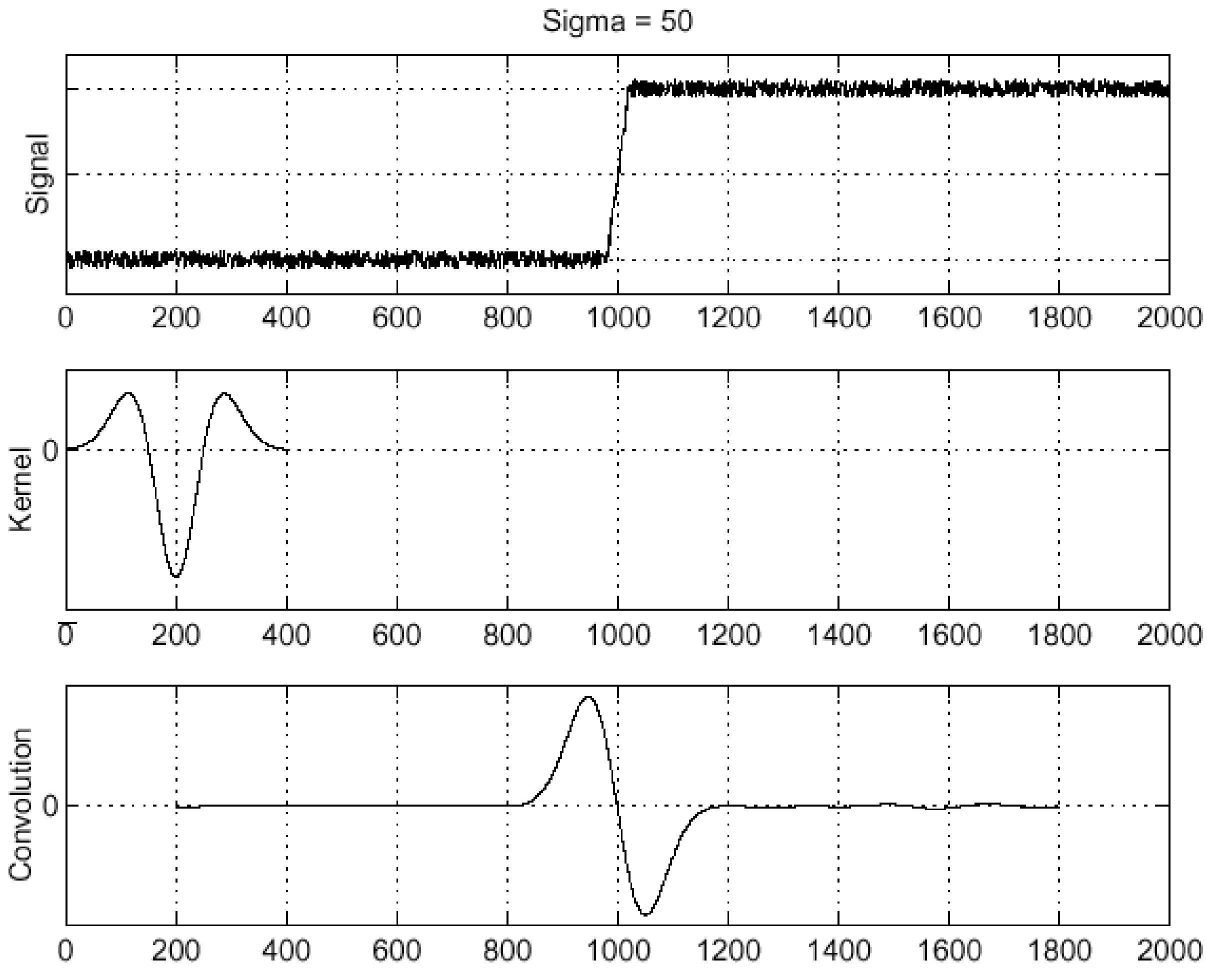
高斯拉普拉斯

$$\frac{\partial^2}{\partial x^2}(h \star f)$$

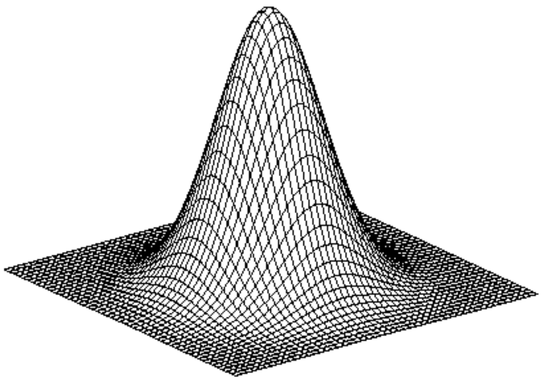
$$f$$

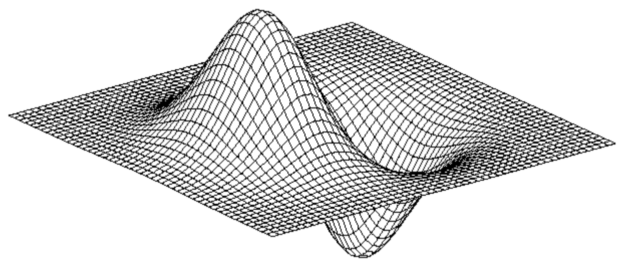
$$\frac{\partial^2}{\partial x^2}h$$

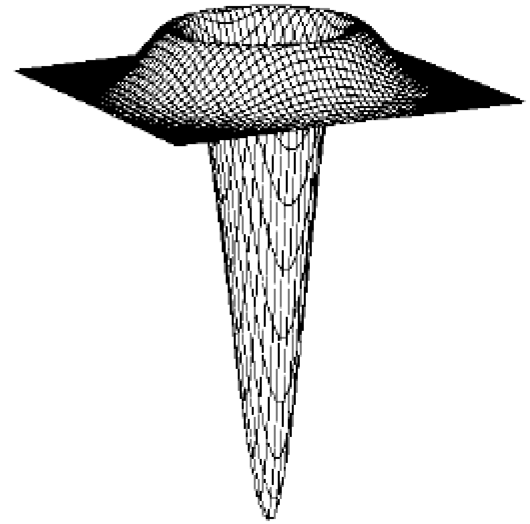
$$(\frac{\partial^2}{\partial x^2}h) \star f$$



二维边缘微分滤波器







高斯

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

高斯的导数

$$\frac{\partial}{\partial x}h_\sigma(u,v)$$

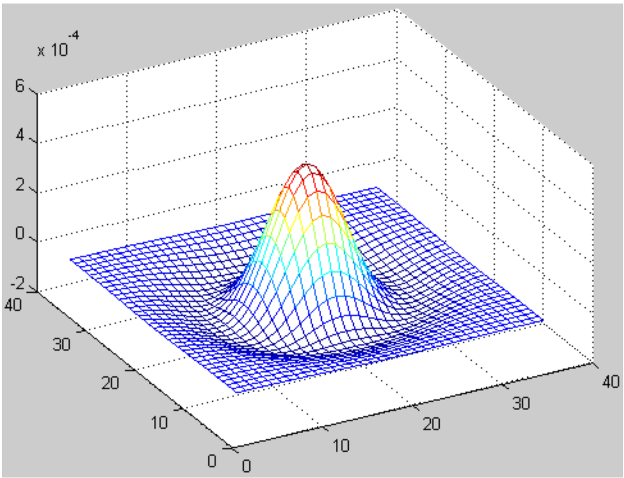
高斯拉普拉斯

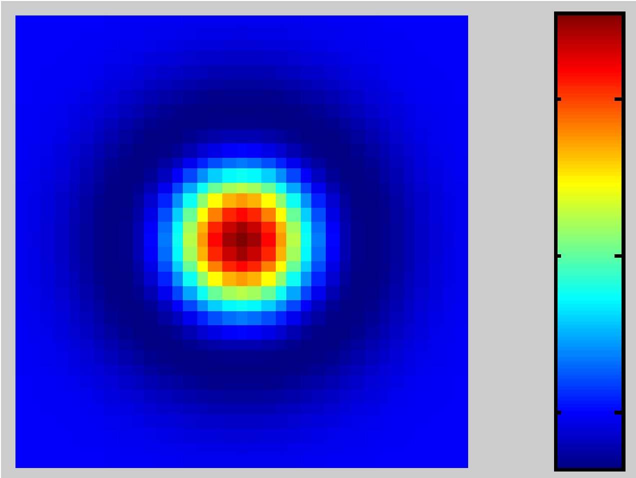
$$\nabla^2 h_\sigma(u,v)$$

LoG算子:

$$\nabla^2 G(x,y)=-\left[\frac{x^2+y^2-\sigma^2}{\sigma^4}\right]e^{\frac{x^2+y^2}{2\sigma^2}}$$

在数字图像上实现LoG





0	0	-1	0	0
0	-1	-2	-1	0
-1	-2	16	-2	-1
0	-1	-2	-1	0
0	0	-1	0	0

- LoG 因其形状，也称为Mexican hat
- 求和为0，即对平坦图像区域的响应为0
- 一个近似的卷积模版：体现主要的形状

$$\iint \nabla^2 G(x,y)dxdy = 0$$

LoG：例子

（a）

（b）

（c）

（a）Lenna Image
（b）Gaussian模版卷积（15*15）
（c）Laplacian模版卷积（3*3）

分两步实现LoG，可以提供更大的灵活性更小的模版尺寸

LoG：例子

（d）

（e）

（f）

（d）将(c)中大于0的像素置1，其余置0
（e）在二值图像（d）上检测边缘，使用形态学膨胀方法
（f）结果显示

几个特点：
（1）正确检测到的边缘：单像素宽度，定位准确
（2）形成许多封闭的轮廓（spaghetti，意大利面条），这是一个主要的问题
（3） 需要更加复杂的算法检测过零点

Canny边缘检测器

边缘检测最优准则

- 边缘检测最优准则：
  - 好的边缘检测性能：Good detection
    对边缘的响应大于对噪声的响应
  - 好的定位性能：Good localization
    其最大值应接近边缘的实际位置
  - 低的错误检测率：Low false positives
    在边缘附近只有一个极值点

Canny边缘检测器

算法基本过程：

计算图像梯度

↓

梯度非极大值抑制

↓

双阈值提取边缘点

幅值大小M(x, y)

方向Theta(x, y)

NMS：

Non-Maxima Suppression

计算图像梯度：高斯函数的一阶导数

（1）求图像与高斯平滑滤波器卷积:

$$S(x,y) = G(x,y,\sigma) * I(x,y)$$

σ代表对图像的平滑程度

（2）使用一阶有限差分计算偏导数的两个阵列：

$$D_x(x,y) \approx (S(x,y+1) - S(x,y) + S(x+1,y+1) - S(x+1,y))/2$$

$$D_y(x,y) \approx (S(x,y) - S(x+1,y) + S(x,y+1) - S(x+1,y+1))/2$$

相当于与模版进行卷积运算：

-1	1
-1	1

1	1
-1	-1

计算图像梯度：高斯函数的一阶导数

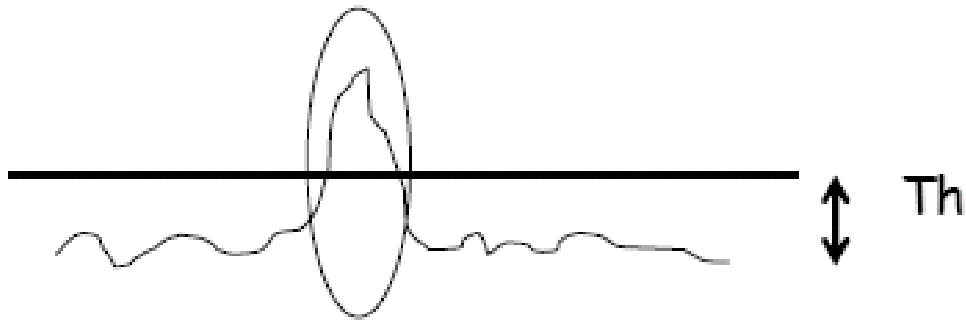
（3）幅值和方位角:

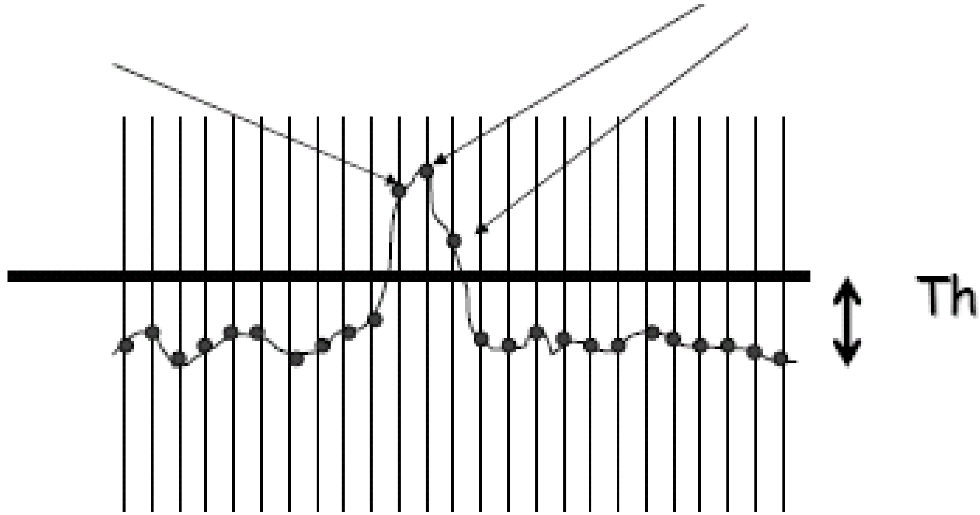
$$M(x,y)=\sqrt{D_x(x,y)^2+D_y(x,y)^2}$$

$$\theta(x,y)=\arctan(D_y(x,y)/D_x(x,y))$$

M代表梯度幅值的大小，在存在边缘的图像位置处，M的值变大，图像的边缘特征被“增强”。

如何检测边缘？





局部极值周围存在相近数值的点

非极大值抑制 NMS

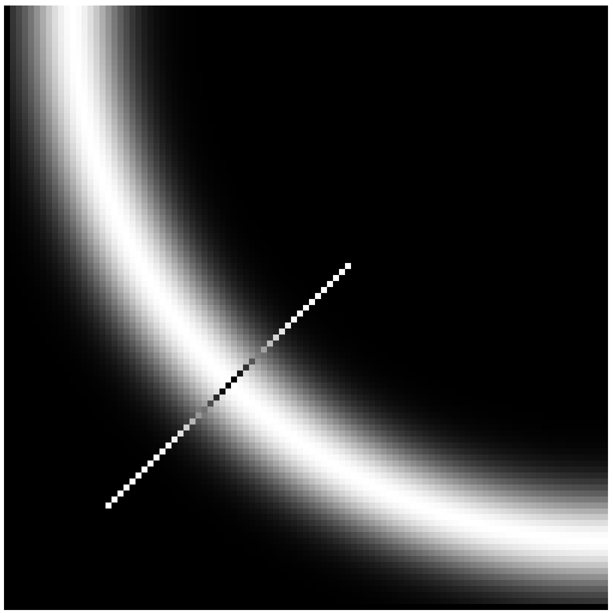
- 非极大值抑制（NMS：Non-Maxima Suppression ）
- 主要思想：由梯度幅值图像M(x,y)，仅保留极大值。

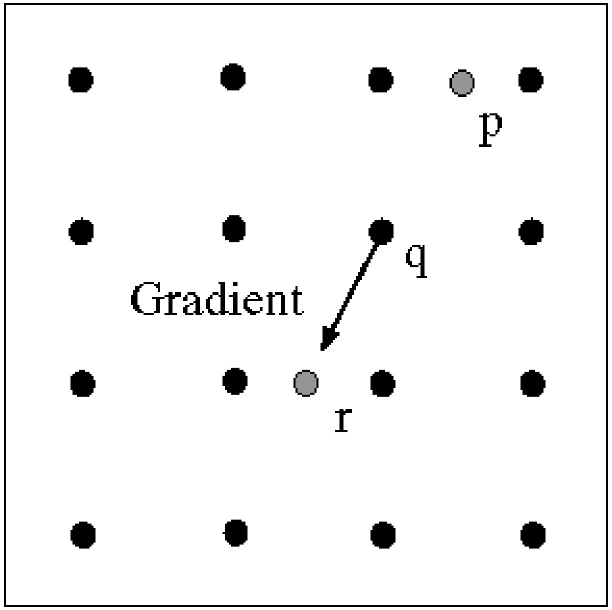
（严格地说，保留梯度方向上的极大值点。）

得到的结果为N(x,y)，具体过程：

  - 初始化N(x,y) = M(x,y)
  - 对于每个点，在梯度方向和反梯度方向各找n个像素点。若M(x,y)不是这些点中的最大点，则将N(x,y)置零，否则保持N(x,y)不变。
- N(x,y)单像素宽度：
  - 问题：额外的边缘点，丢失的边缘点

非极大值抑制 NMS





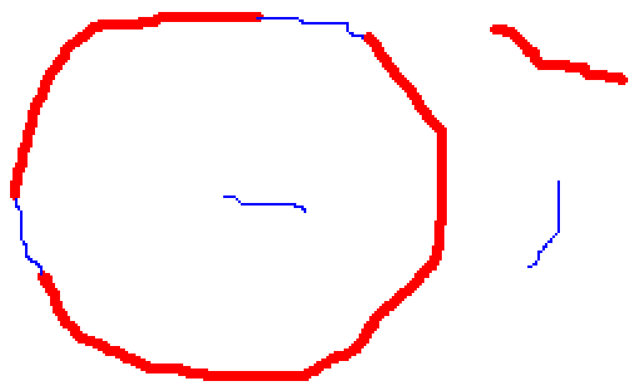
- 在梯度方向的沿线上检测该点是否为局部极大值
- 简化的情形，只使用4个方向：{0，45，90，135}
- 得到的结果N（*x*，*y*）包含边缘的宽度为1个像素

对NMS结果进行二值化

- 对上述得到的N(x,y)使用阈值进行二值化
- 使用大的阈值，得到：
  - 少量的边缘点
  - 许多空隙
- 使用小的阈值，得到：
  - 大量的边缘点
  - 大量的错误检测

使用双阈值检测边缘

- 两个阈值T1，T2： T2 >> T1
  - 由T1得到 E1(x, y)，低阈值边缘图：更大的误检测率
  - 由T2得到 E2(x, y)，高阈值边缘图：更加可靠
- 边缘连接：



边缘连接

1. 将E2(x, y)中相连的边缘点输出为一幅边缘图像E(x, y)
2. 对于E(x, y)中每条边，从端点出发在E1(x, y)中寻找其延长的部分，直至与E(x, y)中另外一条边的端点相连，否则认为E1(x, y)中没有它延长的部分
3. 将E(x, y)作为结果输出

Canny算子：流程

原始图像

原始图像经过Gauss平滑

Canny算子：流程

梯度幅值图像

梯度幅值经过非极大值抑制

Canny算子：流程

低阈值边缘图像

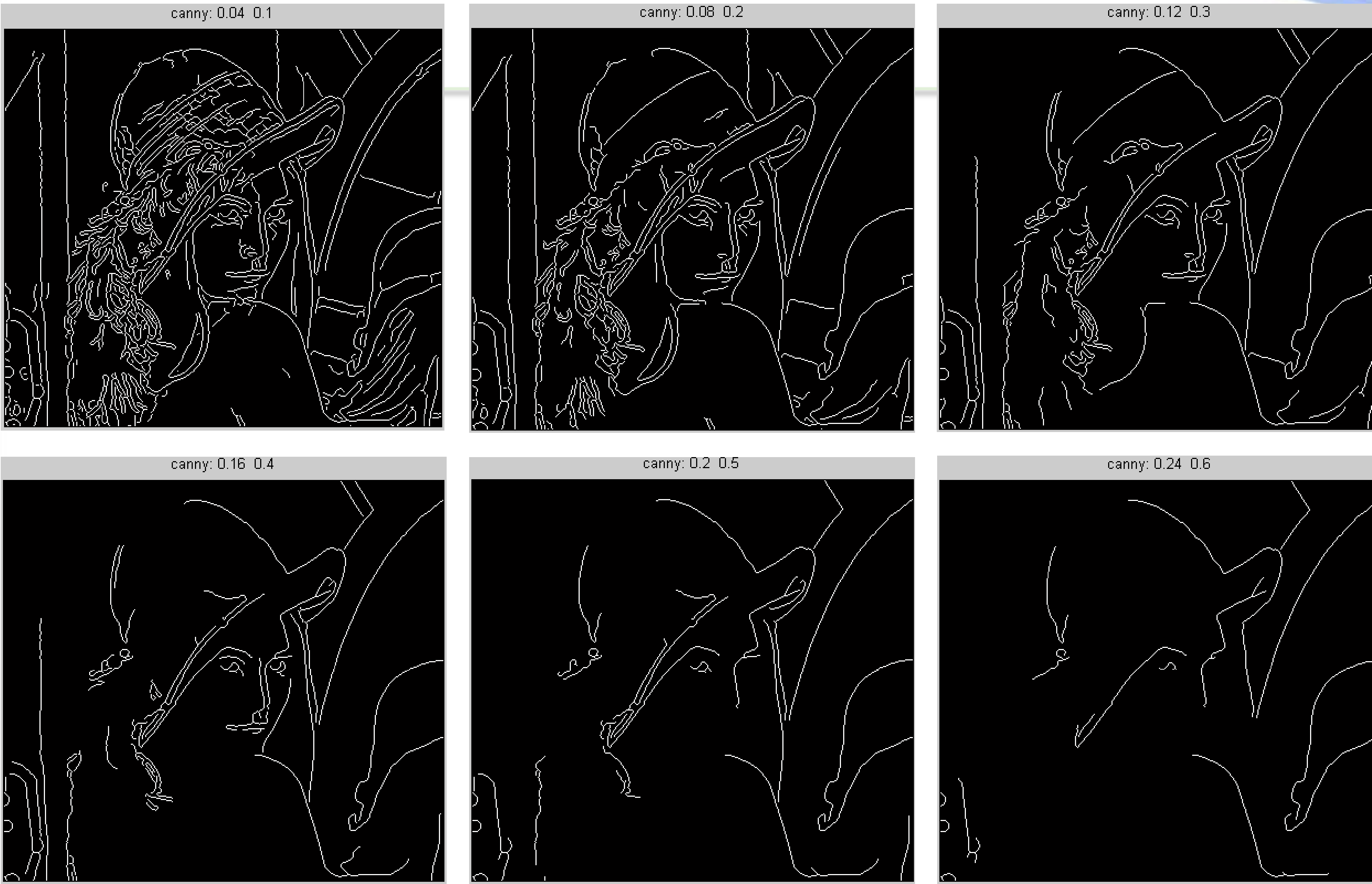
高阈值边缘图像

Canny输出边缘图像

使用Canny算子需要注意的问题

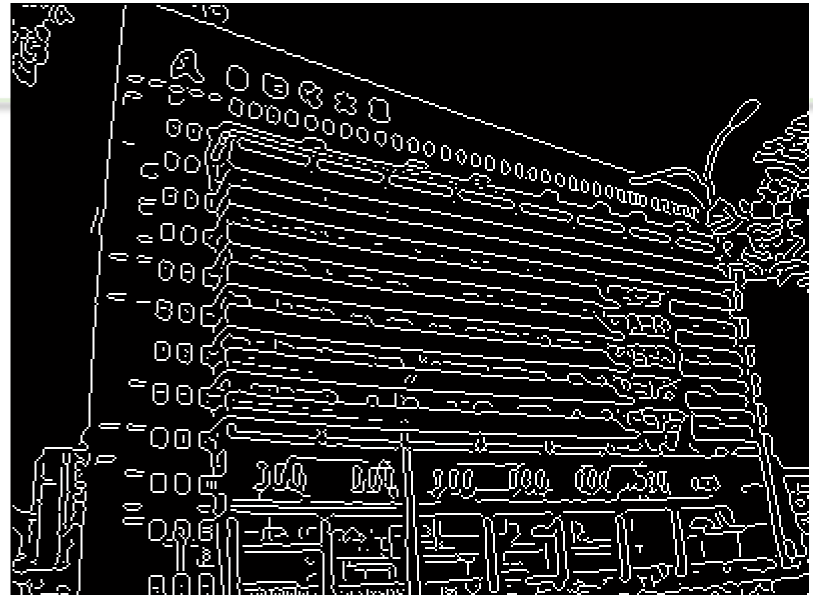
- Canny算子的优点:
  - 参数较少
  - 计算效率
  - 得到的边缘连续完整
- 参数的选择:
  - Gauss滤波的尺度
  - 双阈值的选择(LOW=HIGH*0.4)

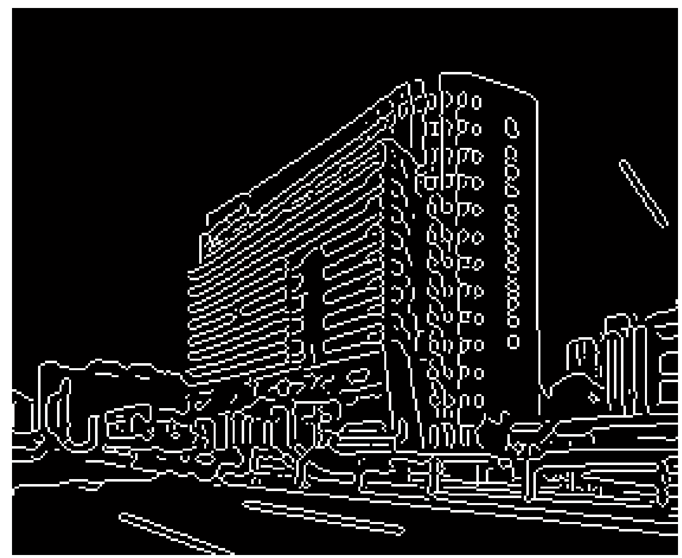
渐增高斯滤波模版的尺寸



渐增双阈值的大小，保持low = high*0.4

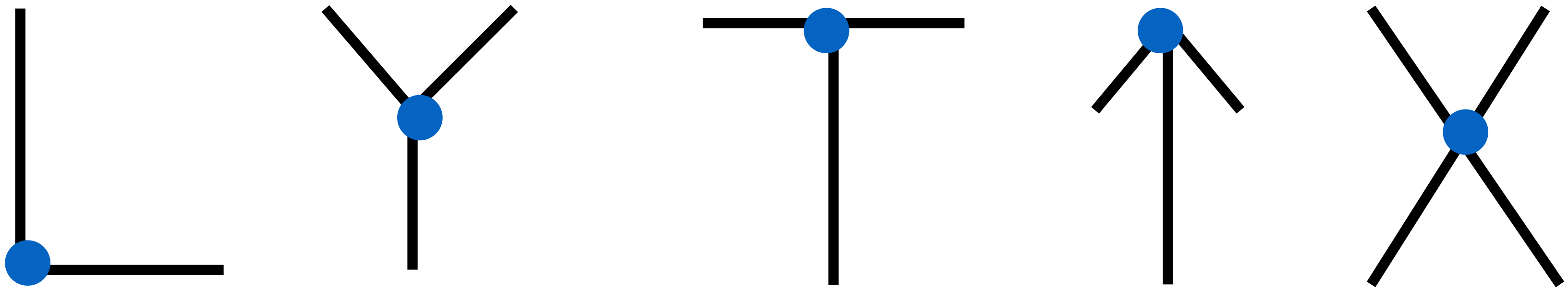

自动化大厦

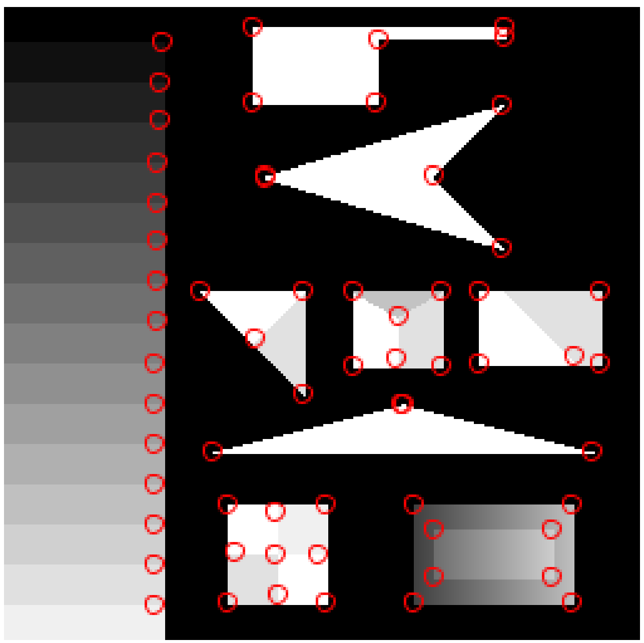


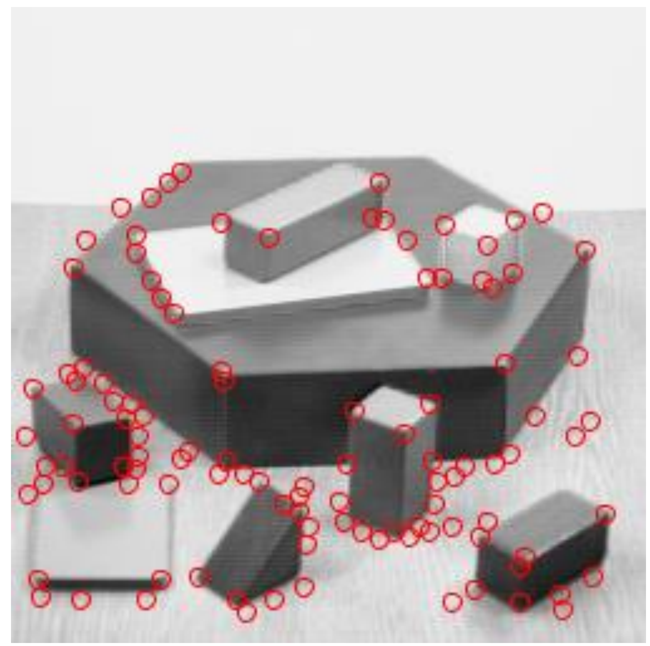


图像特征点提取

不同类型的角点







提取点特征的作用

- 图像的点特征是许多计算机视觉算法的基础：使用特征点来代表图像的内容
  - 运动目标跟踪
  - 物体识别
  - 图像配准
  - 全景图像拼接
  - 三维重建

什么是好的角点检测算法？

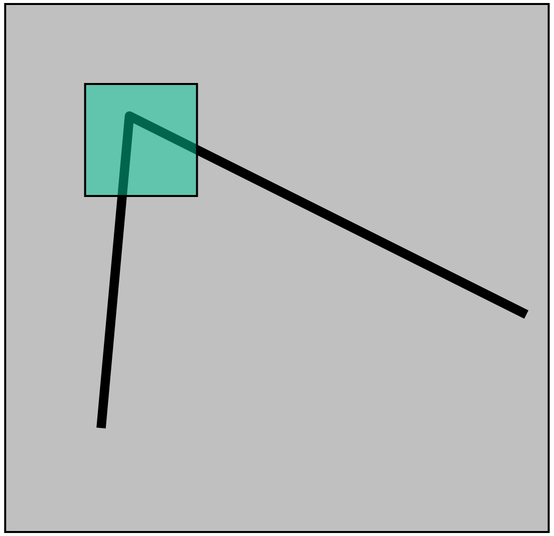
- 检测出图像中“真实的”角点
- 准确的定位性能
- 很高的重复检测率（稳定性好）
- 具有对噪声的鲁棒性
- 具有较高的计算效率

角点检测算法

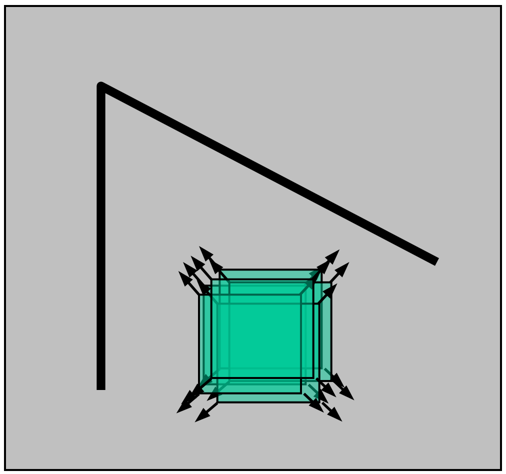
- Harris角点
- ORB特征

Harris角点检测基本思想

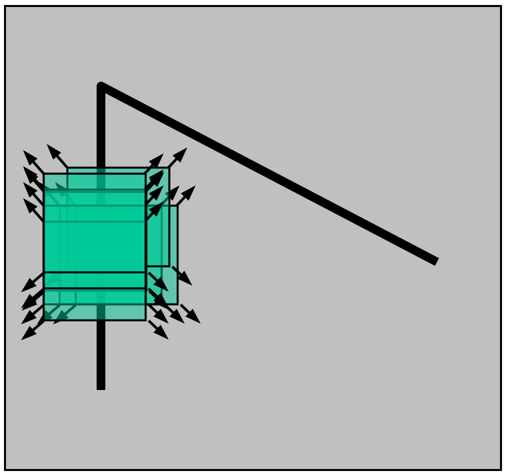
- 从图像局部的小窗口观察图像特征
- 角点定义 ← 窗口向任意方向的移动都导致图像灰度的明显变化



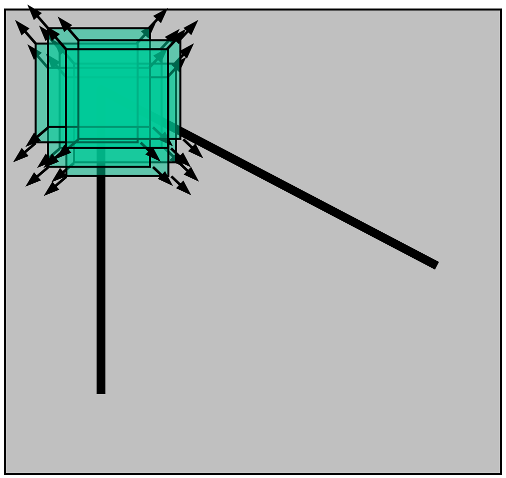
Harris角点检测基本思想



平坦区域：
任意方向移动，
无灰度变化



边缘：
沿着边缘方向移动，
无灰度变化



角点：
沿任意方向移动，
明显灰度变化

Harris检测：数学表达

将图像窗口平移$[u,v]$产生灰度变化$E(u,v)$

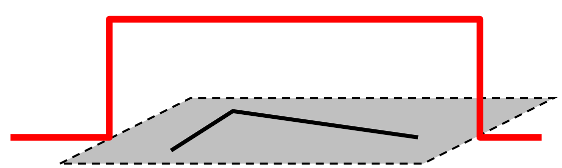
$$E(u,v)=\sum_{x,y} w(x,y)[I(x+u,y+v)-I(x,y)]^2$$

窗口函数

平移后的图像灰度

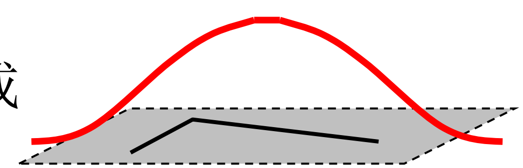
图像灰度

窗口函数 $w(x,y)=$



1 in window, 0 outside

或



Gaussian

Harris检测：数学表达

$$E(u,v)=\sum_{x,y} w(x,y)[I(x+u,y+v)-I(x,y)]^2$$

$$I(x+u,y+v)=I(x,y)+I_x u+I_y v+O(u^2+v^2)$$

$$\Rightarrow\quad E(u,v)=\sum_{x,y} w(x,y)[I_x u+I_y v+O(u^2+v^2)]^2$$

$$(I_x u+I_y v)^2=I_x^2u^2+2I_xI_yuv+I_y^2v^2$$

$$=(u,v)\begin{pmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{pmatrix}\begin{pmatrix} u \\ v \end{pmatrix}$$

Harris检测：数学表达

于是对于局部微小的移动量$[u,v]$，可以近似得到下面的表达：

$$E(u,v) \cong (u,v)M\begin{pmatrix} u \\ v \end{pmatrix}$$

其中M是2×2矩阵，可由图像的导数求得：

$$M = \sum_{x,y} w(x,y)\begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix}$$

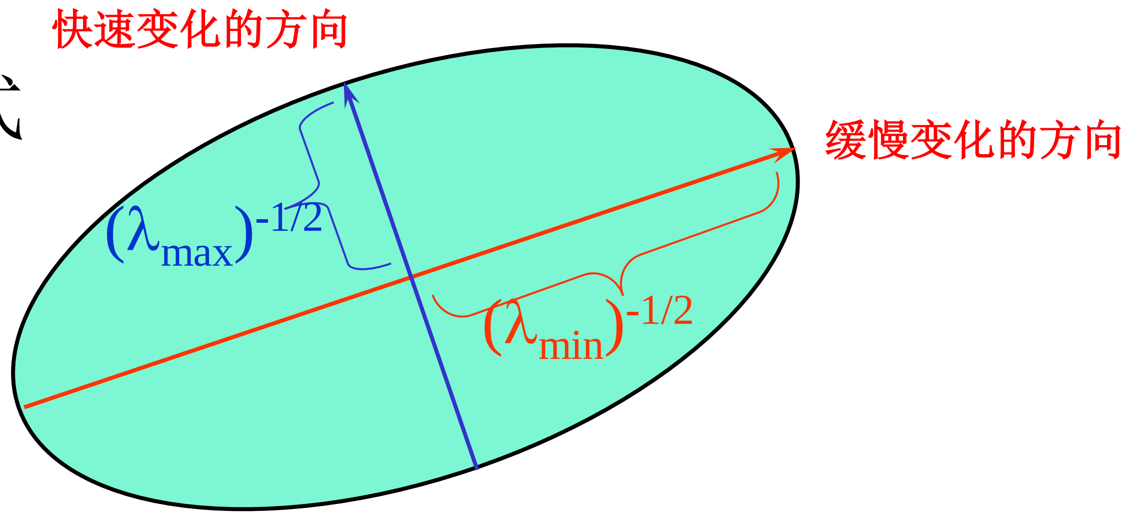
Harris检测：数学表达

窗口移动导致的图像变化：实对称矩阵M的特征值分析

$$E(u,v) \cong (u,v)M\binom{u}{v}$$

$$\lambda_{max}, \lambda_{min} \leftarrow M\text{的特征值}$$

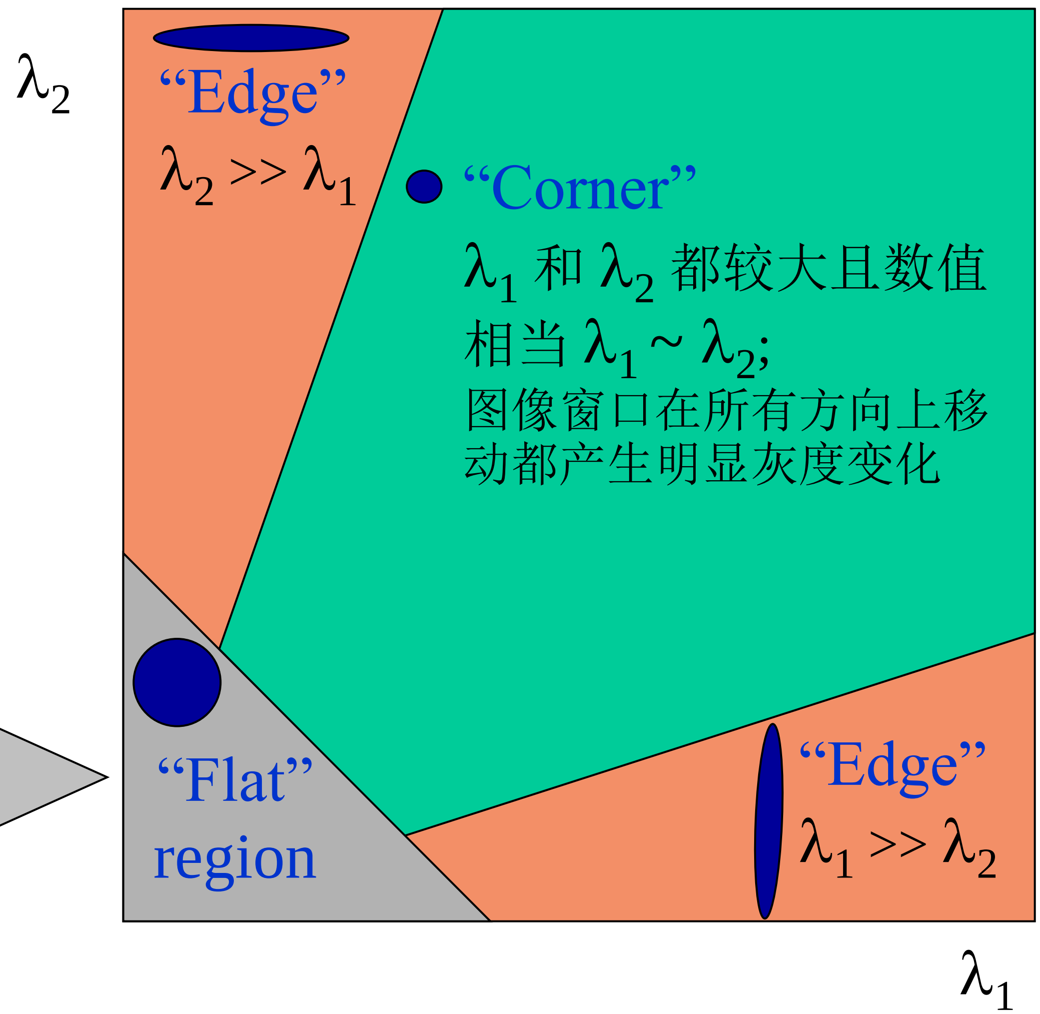
$E(u,v)$的椭圆形式



Harris检测：数学表达

通过M的两个特征值的大小对图像点进行分类:

如果λ_1 和 λ_2 都很小，图像窗口在所有方向上移动都无明显灰度变化



Harris检测：数学表达

定义：角点响应函数R

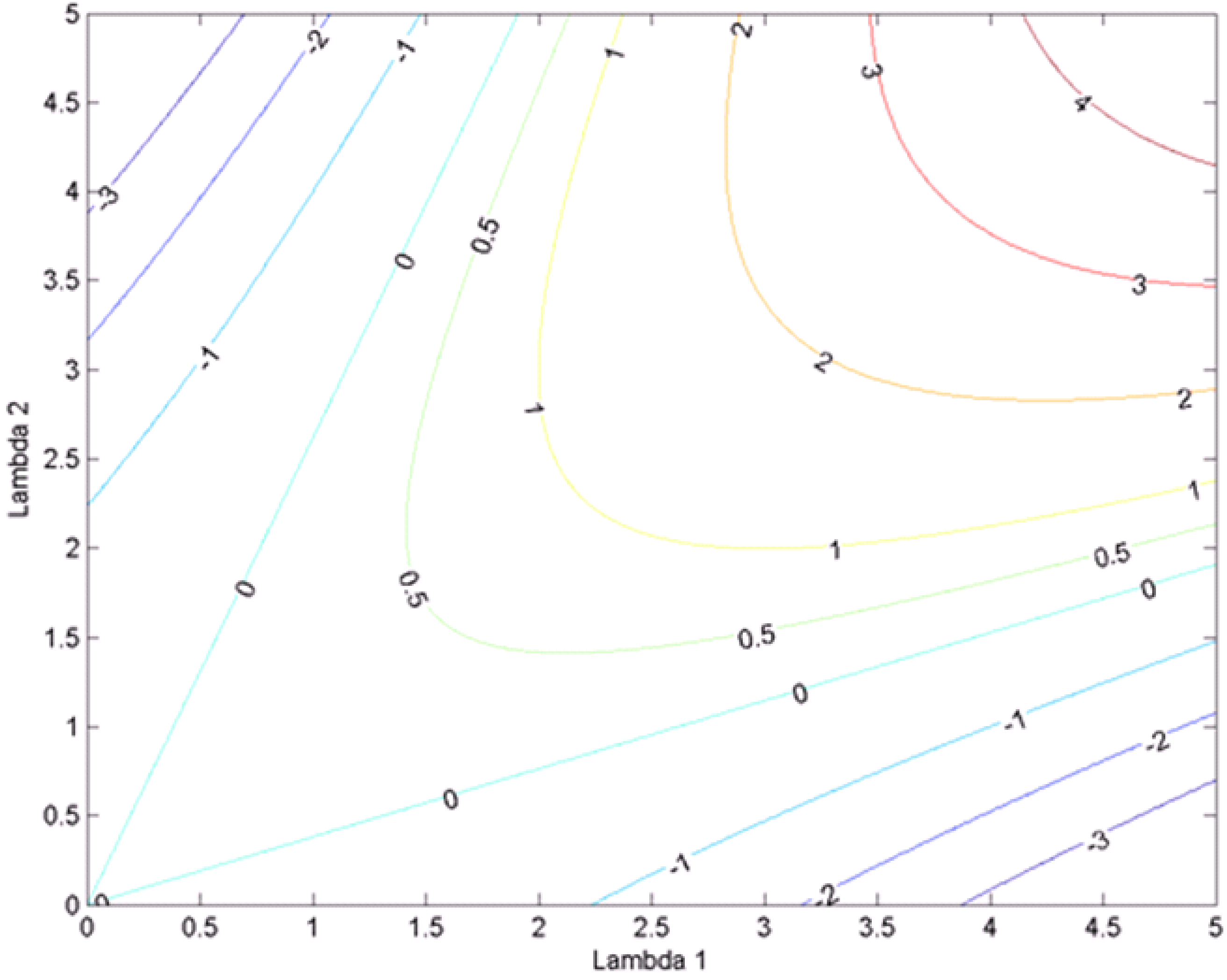
$$R = \det M - k(\operatorname{trace} M)^2$$

$$\det M = \lambda_1 \lambda_2$$

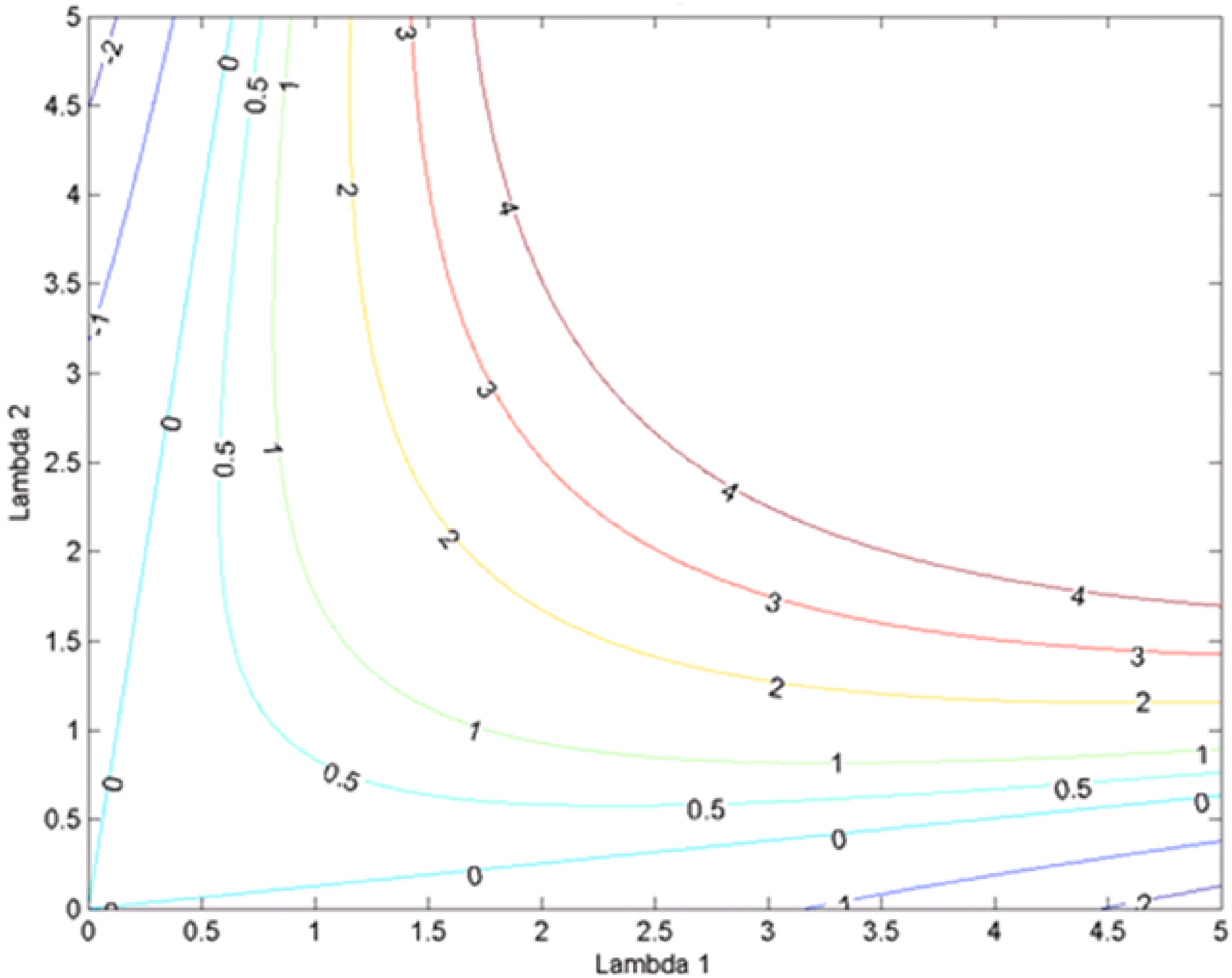
$$\operatorname{trace} M = \lambda_1 + \lambda_2$$

(k – empirical constant, k = 0.04-0.06)

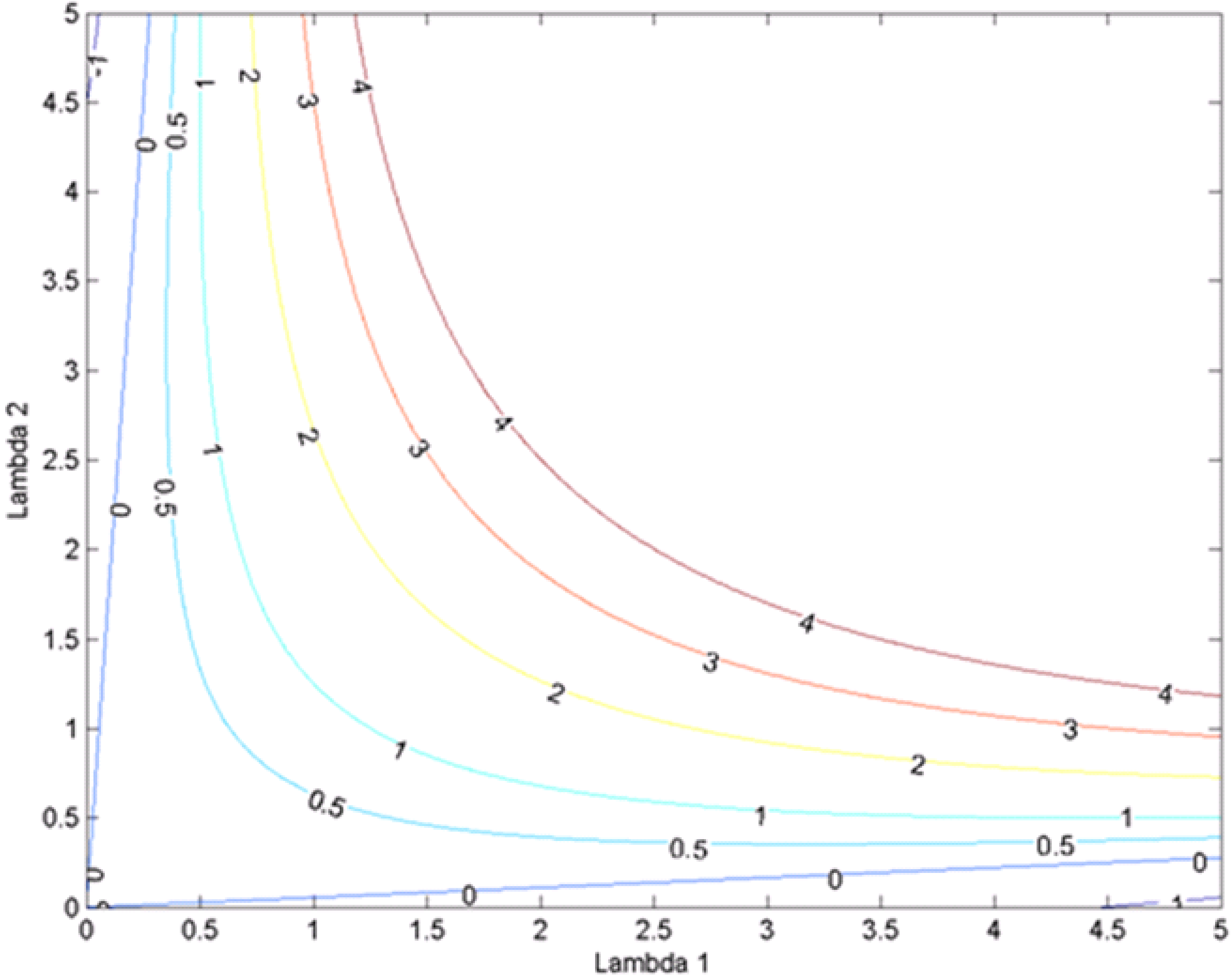
R的等高线图（$k = 0.2$）



R的等高线图（$k = 0.1$）

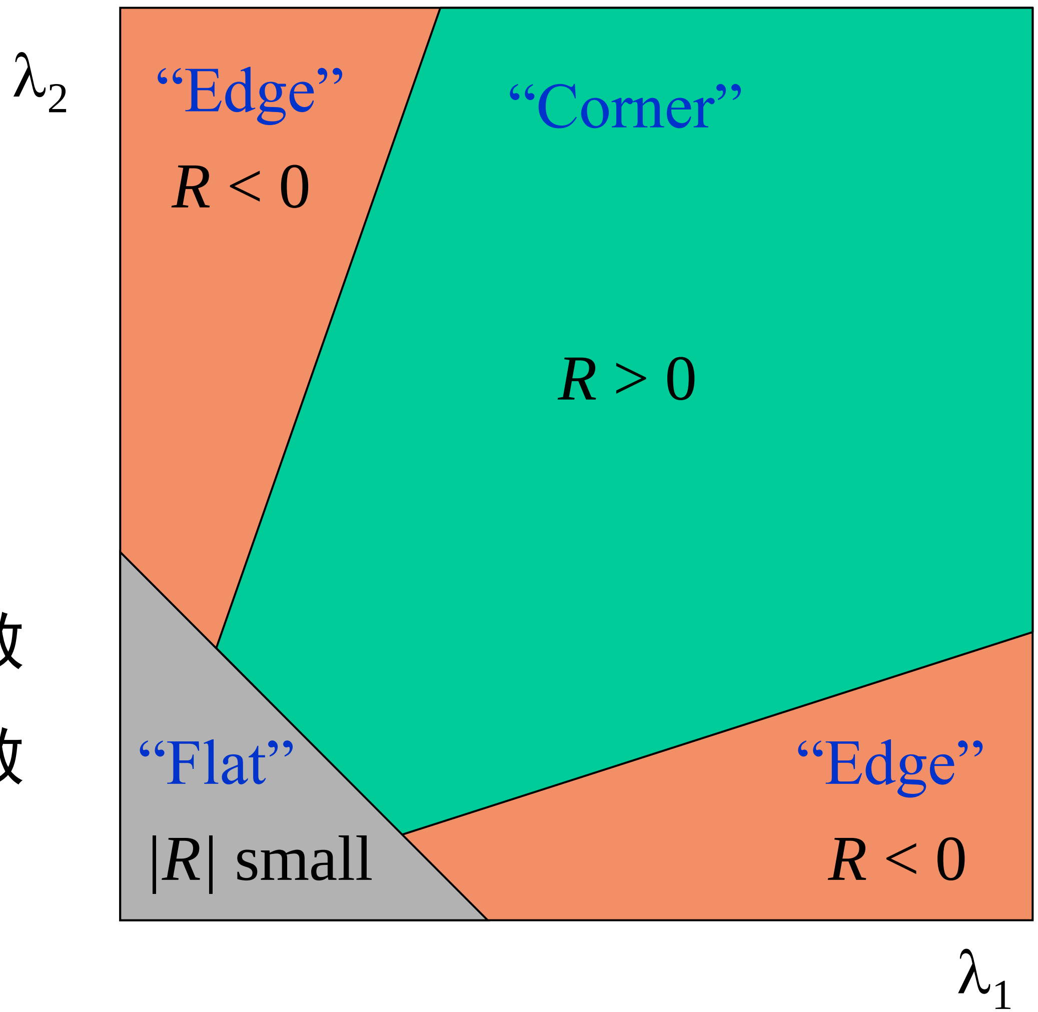


R的等高线图（$k = 0.05$ ）



Harris检测：数学表达

- R只与M的特征值有关
- 角点：R为大数值正数
- 边缘：R为大数值负数
- 平坦区：R为小数值



Harris角点检测

- 算法:
  - 对角点响应函数R进行阈值处理:

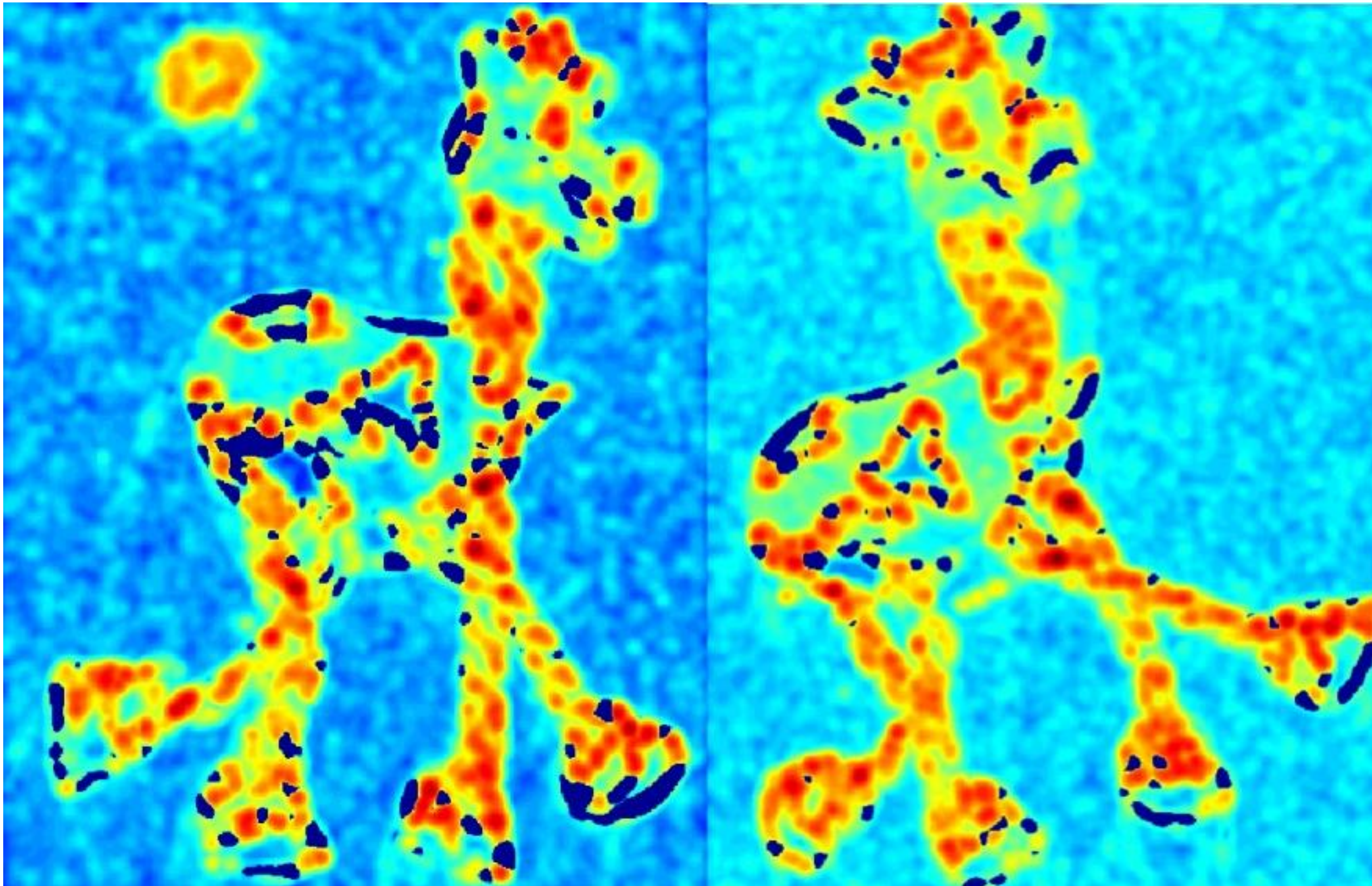
$$R > \text{threshold}$$

  - 提取R的局部极大值

Harris角点检测：流程

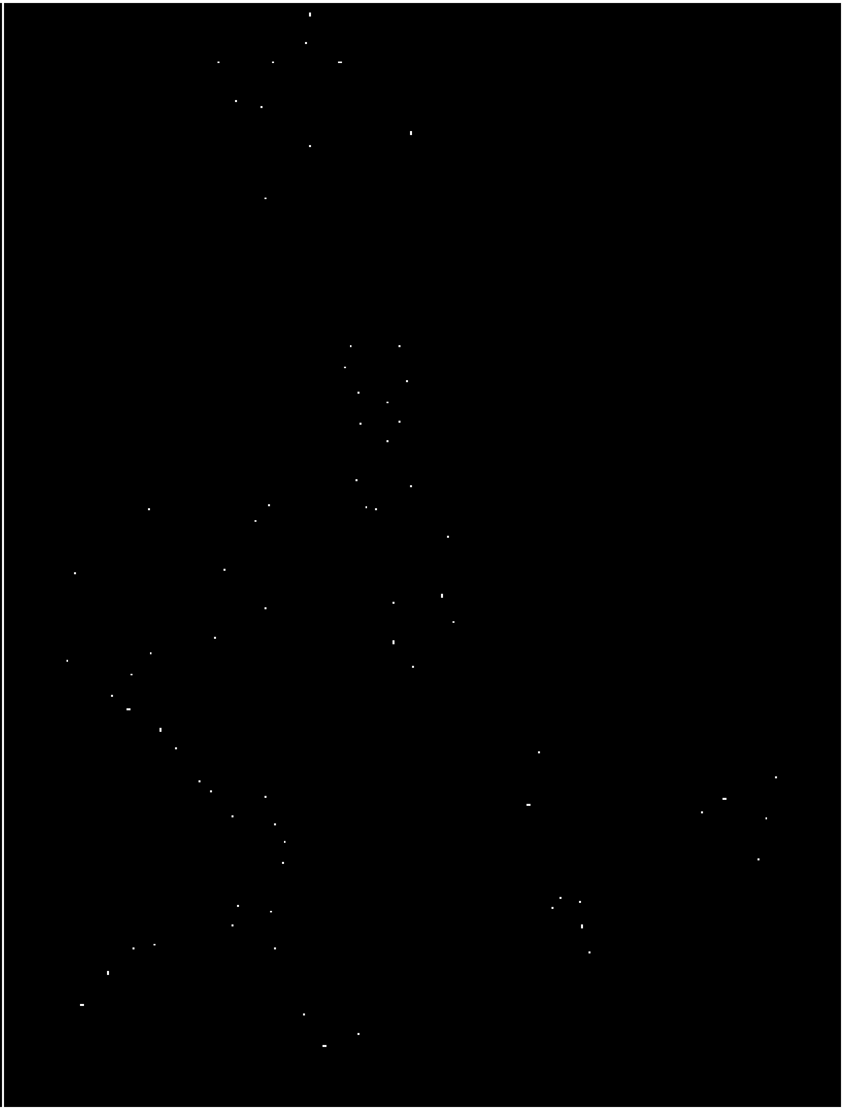
Harris角点检测：流程

角点响应函数R



Harris角点检测：流程

提取R的局部极值



Harris角点检测：流程

Harris角点检测：小结

- 沿方向$[u,v]$的平均灰度变化可以表达成双线性形式：

$$E(u,v) \cong (u,v)M\begin{pmatrix}u\\v\end{pmatrix}$$

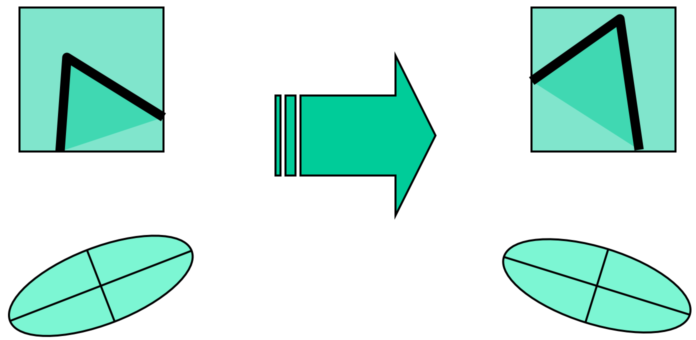
- 使用M的特征值表达图像点局部灰度变化的情况，定义角点响应函数：

$$R = \det M - k(\operatorname{trace} M)^2$$

- 一个好的角点沿着任意方向移动都将导致明显的图像灰度变化，即：R具有大的正数值。

Harris角点的性质

- 旋转不变性：



椭圆转过一定角度但是其形状保持不变（特征值保持不变）

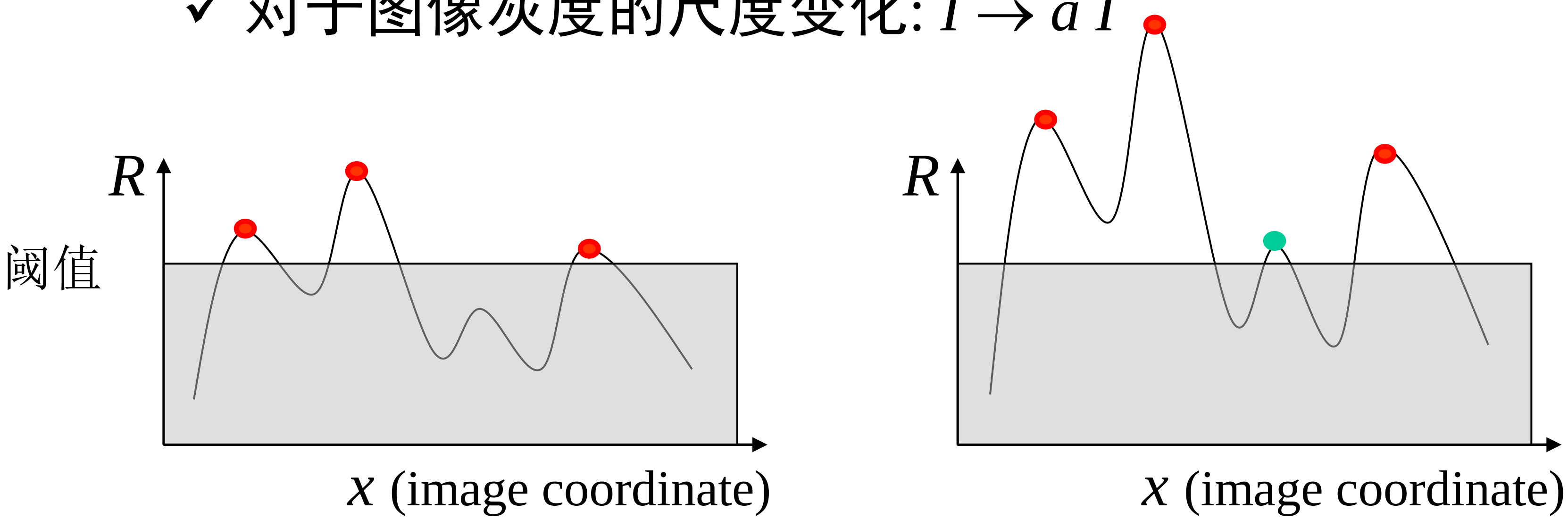
角点响应函数R对于图像的旋转具有不变性

Harris角点的性质

- 对于图像灰度的仿射变化具有部分的不变性
  - ✓ 只使用了图像导数 => 对于灰度平移变化不变

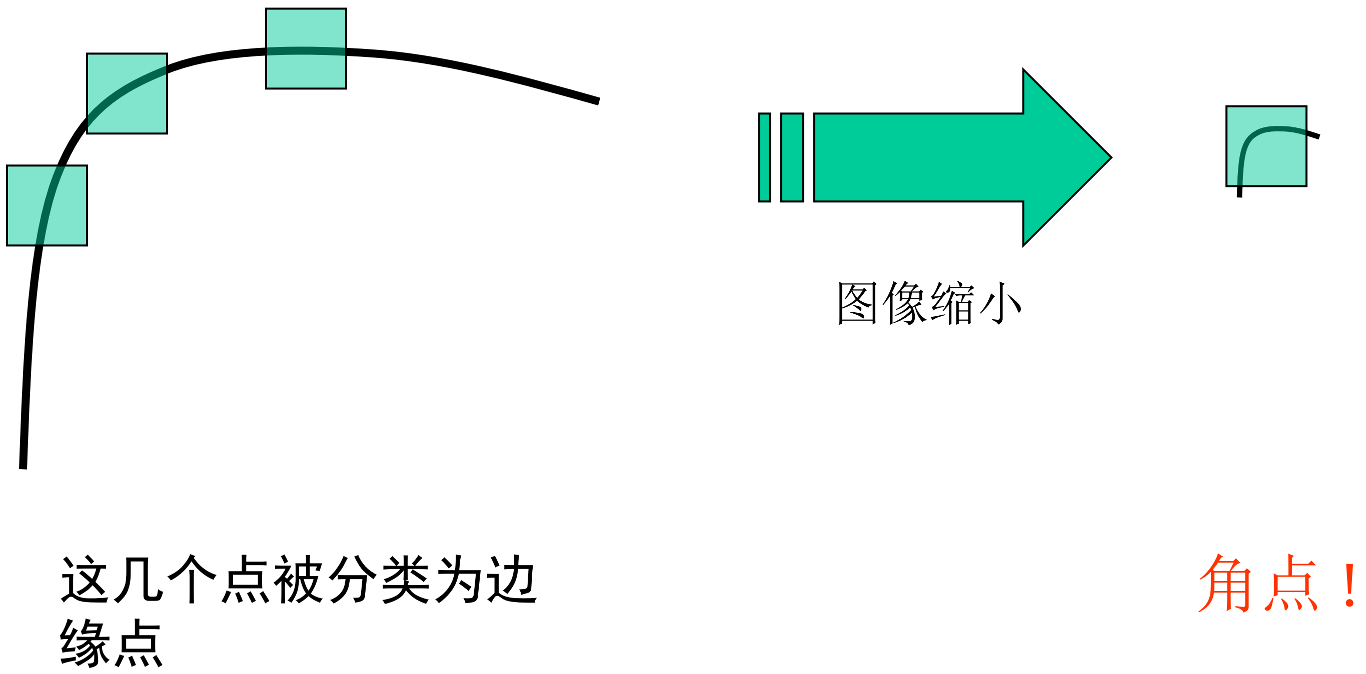
$$I \rightarrow I + b$$

  - ✓ 对于图像灰度的尺度变化: $I \rightarrow a\,I$



Harris角点的性质

- 对于图像几何尺度变化不具有不变性：

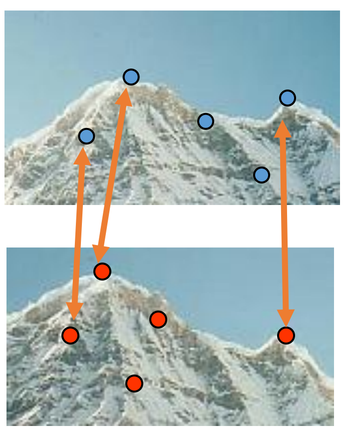


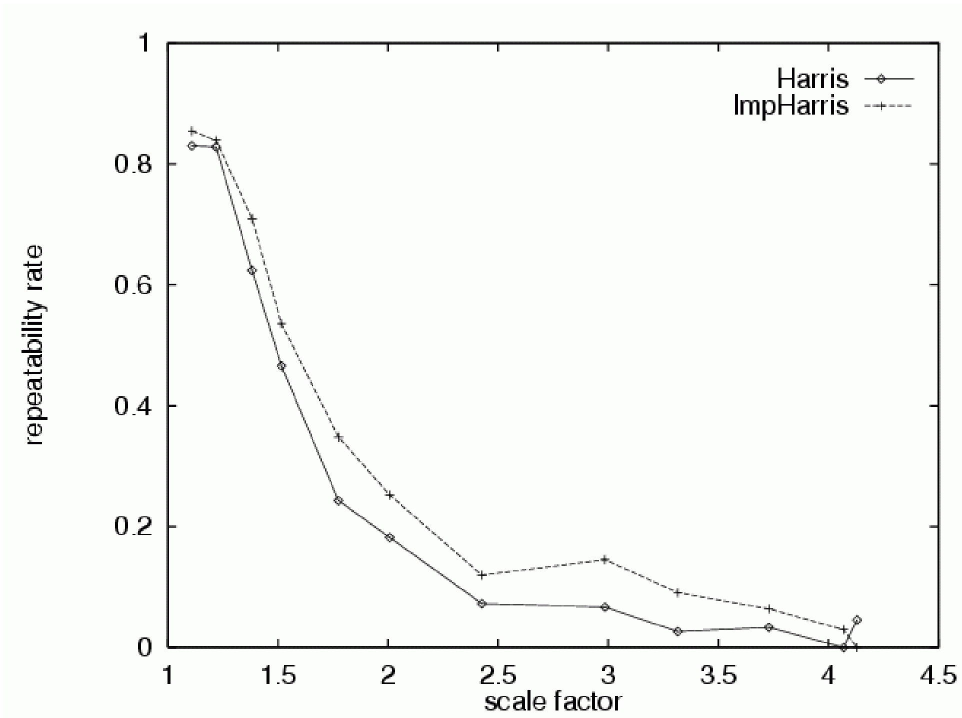
Harris角点的性质

- 随尺度变化，Harris角点检测的性能下降

Repeatability rate:

$$\frac{\text{\# correspondences}}{\text{\# possible correspondences}}$$





C.Schmid et.al. “Evaluation of Interest Point Detectors”. IJCV 2000

ORB特征检测

（Oriented FAST and Rotated BRIEF）

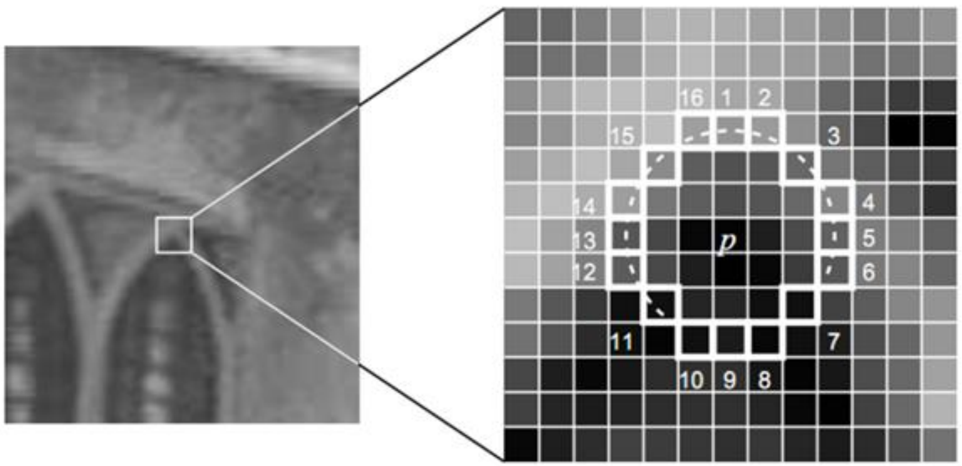
ORB特征

- ORB（Oriented FAST and Rotated BRIEF）
- ORB = FAST + BRIEF（改进的版本）
- 研究动机：快速性、兼顾准确性
- FAST（features from accelerated segment test）
  - 假设：若该点的灰度值比其周围领域内足够多的像素点的灰度值大或者小，则该点可能为角点。

Edward Rosten, Reid Porter, and Tom Drummond, "Faster and better: a machine learning approach to corner detection" in IEEE Trans. Pattern Analysis and Machine Intelligence, 2010, vol 32, pp. 105-119.

ORB特征

- Fast-N: $N = \sum_{x \forall (circle(p))} \left| I(x) - I(p) \right| > \varepsilon_d$



参考

- Harris角点：

  C.Harris, M.Stephens. “A Combined Corner and Edge Detector”. Proc. of 4th Alvey Vision Conference, 1988

- 一个介绍角点检测的网站：http://www.cim.mcgill.ca/~dparks/CornerDetector/index.htm

- 一个PPT讲义： Darya Frolova, Denis Simakov, Matching with Invariant Features, The Weizmann Institute of Science, March 2004

小节

- 边缘检测、特征点检测是计算机视觉中最基本的问题
  - 微分算子
  - Canny
  - Harris
  - ORB->Fast
- 没有一种统一的方法可以解决这两个问题：
  - 抑制噪声的能力
  - 定位精度
  - 计算的复杂程度

课后练习

1 试选用一个深度学习框架实现LeNet

2 试使用C++语言实现Harris角点检测算法

谢谢