# Revision

无计算题

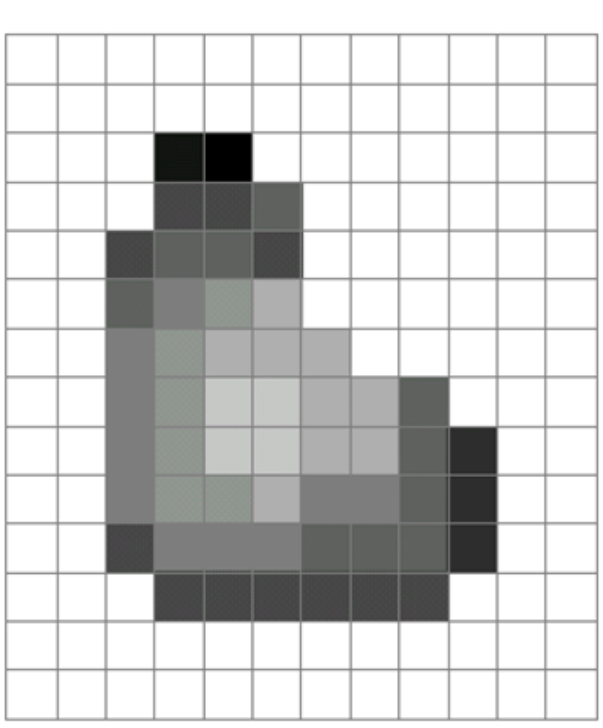
# Image Filtering 图像过滤

None-liner filtering covered 涵盖非线性过滤

Input an image, if the output is not an image, it can be feature extraction 输入图像，如果输出不是图像，则为特征提取

## Image

Image is a grid (matrix) of intensity values



=

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 255 | 255 | 255 | 20 | 0 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 255 | 255 | 255 | 75 | 75 | 75 | 255 | 255 | 255 | 255 | 255 | 255 |
| 255 | 255 | 75 | 95 | 95 | 75 | 255 | 255 | 255 | 255 | 255 | 255 |
| 255 | 255 | 96 | 127 | 145 | 175 | 255 | 255 | 255 | 255 | 255 | 255 |
| 255 | 255 | 127 | 145 | 175 | 175 | 175 | 255 | 255 | 255 | 255 | 255 |
| 255 | 255 | 127 | 145 | 200 | 200 | 175 | 175 | 95 | 255 | 255 | 255 |
| 255 | 255 | 127 | 145 | 200 | 200 | 175 | 175 | 95 | 47 | 255 | 255 |
| 255 | 255 | 127 | 145 | 145 | 175 | 127 | 127 | 95 | 47 | 255 | 255 |
| 255 | 255 | 74 | 127 | 127 | 127 | 95 | 95 | 95 | 47 | 255 | 255 |
| 255 | 255 | 255 | 74 | 74 | 74 | 74 | 74 | 74 | 255 | 255 | 255 |
| 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |

## Filters

Filtering: form a new image whose pixels are a combination of the original pixels 滤波：形成新图像，其像素由原始像素组合而成

We use filtering to:

- To get useful information from images 从图像中获取有用信息
  - E.g., extract edges or contours (to understand shape)
- To enhance the image 增强图像效果
  - E.g., to remove noise
  - E.g., to sharpen or to “enhance image”

Image filtering: Modify the pixels in an image, based on some function of a local neighborhood of each pixel

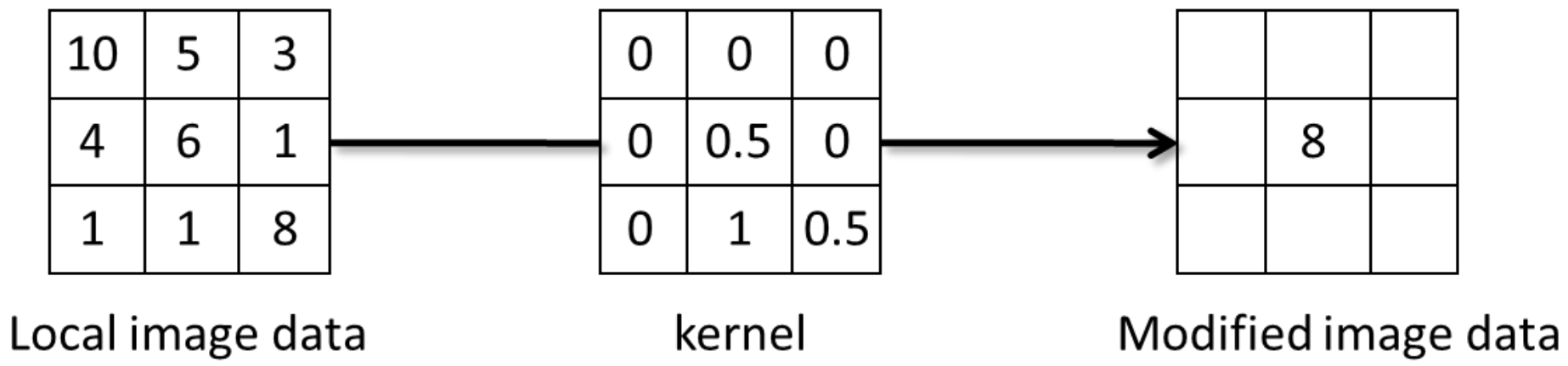
图像过滤： 根据每个像素的局部邻域的某些函数修改图像中的像素


## Linear filtering 线性滤波

One simple version of filtering, including cross-correlation, convolution. 一种简单的滤波方法，包括交叉相关、卷积。

It replaces each pixel by a linear combination (a weighted sum) of its neighbors 它将每个像素替换为其邻近像素的线性组合（加权和）。

•The prescription for the linear combination is called the “kernel” (or “mask”, “filter”) -线性组合的处方称为 "核"（或 "掩码"、"滤波器"）。




## Cross-correlation 交叉相关

Let F be the image, H be the kernel (of size 2*k*+1 x 2*k*+1), and G be the output image

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

Can be represented as :

$$G = H \otimes F$$

We could take it as a “dot product” between local neighborhood and kernel for each pixel. 我们可以把它看作是每个像素的局部邻域和内核之间的 "点积"。

## Convolution卷积

Very similar to Cross-correlation, but the kernel is “flipped” (**horizontally and vertically**) 与交叉相关非常相似，但内核被 "翻转"（**横向和纵向**）。

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

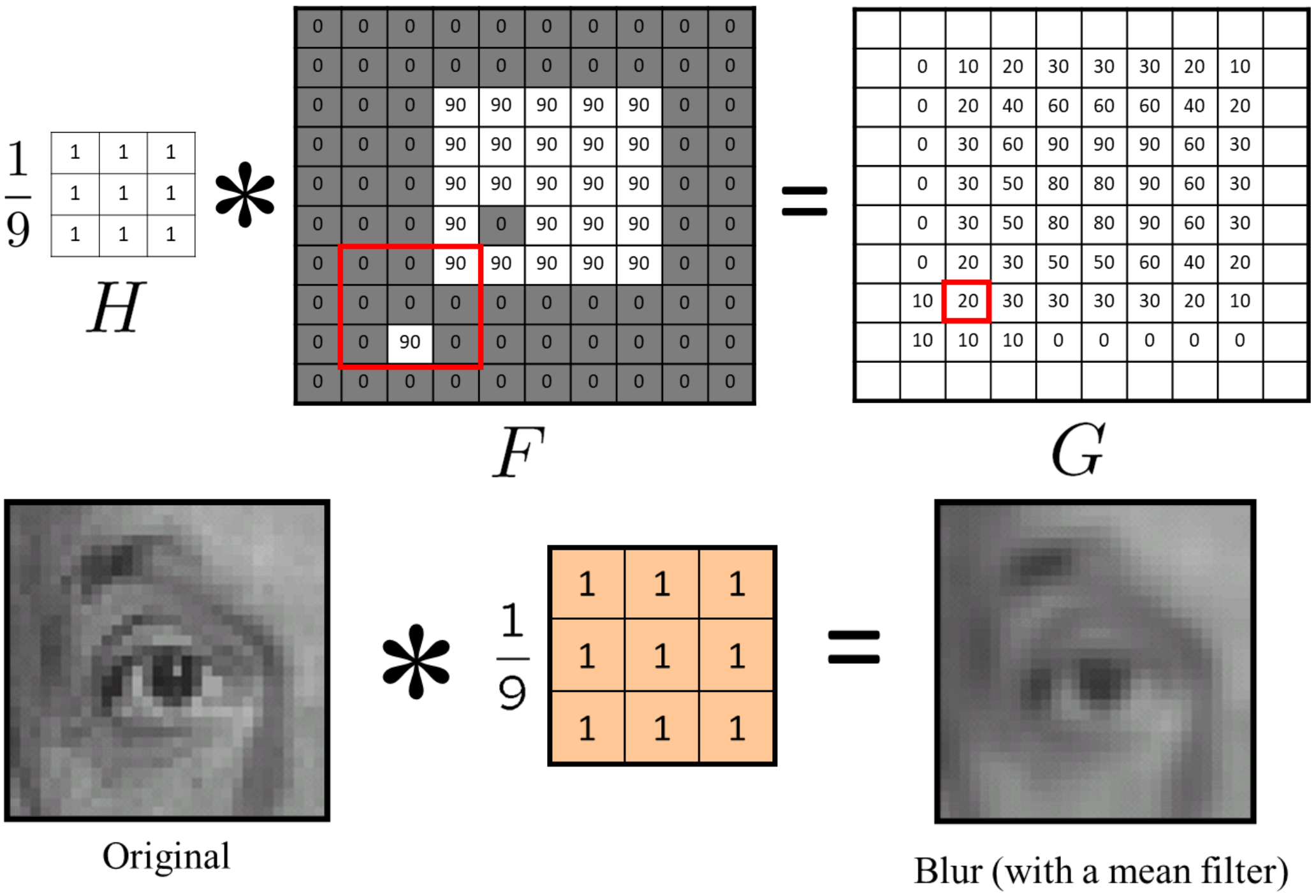
$$G = H * F$$

## Mean filtering 平均过滤

均值滤波器

It is a simple and commonly used image smoothing technique for **noise reduction**. It achieves the smoothing effect of an image by replacing the value of the center pixel by calculating the average value of the pixels in the neighborhood. This method is effective in reducing random noise, but it may also lead to blurring of image details, especially edge information.

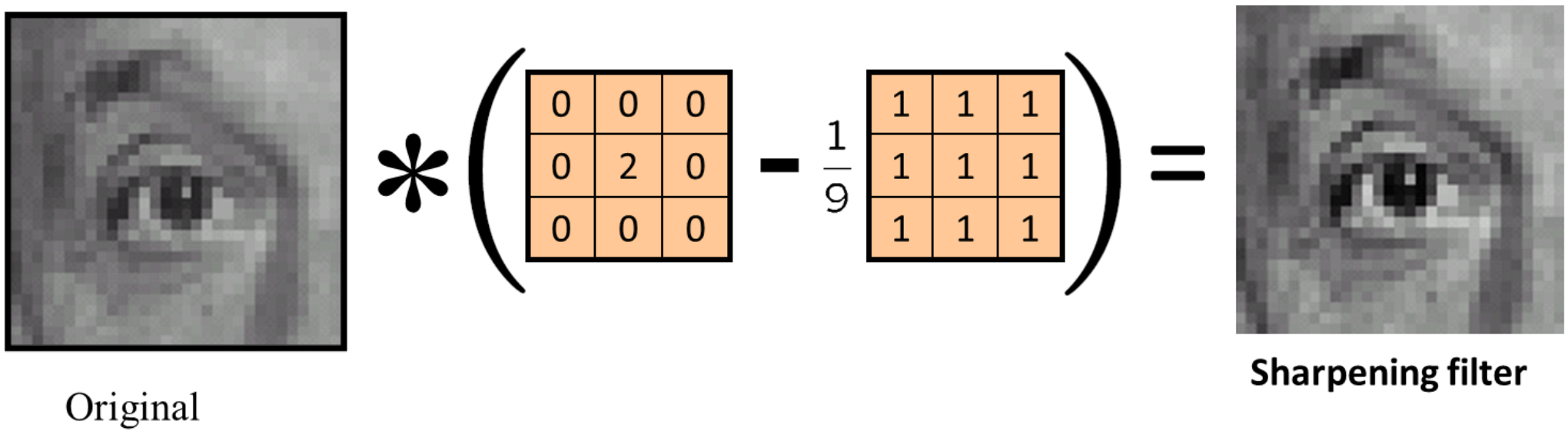
它是一种简单而常用的图像平滑技术，用于降低噪声。 它通过计算邻近像素的平均值来替换中心像素的值，从而达到平滑图像的效果。 这种方法能有效减少随机噪声，但也可能导致图像细节模糊，尤其是边缘信息。







Similarly, subtract smoothed image could be used for sharpening. 同样，减法平滑图像也可用于锐化。




## Gaussian Filter 高斯滤波器

### Gaussian Kernel

Gaussian kernel is a commonly used smoothing filter to reduce image noise and retain important features

高斯核是一种常用的平滑滤波器，可减少图像噪声并保留重要特征

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

It achieves the smoothing effect by assigning different weights to neighboring pixels, with the weights decreasing with increasing distance from the center point

它通过为相邻像素分配不同的权重来实现平滑效果，权重随中心点距离的增加而减小

| | | | | |
|---|---|---|---|---|
| 0.003 | 0.013 | 0.022 | 0.013 | 0.003 |
| 0.013 | 0.059 | 0.097 | 0.059 | 0.013 |
| 0.022 | 0.097 | 0.159 | 0.097 | 0.022 |
| 0.013 | 0.059 | 0.097 | 0.059 | 0.013 |
| 0.003 | 0.013 | 0.022 | 0.013 | 0.003 |

$5 \times 5, \sigma = 1$

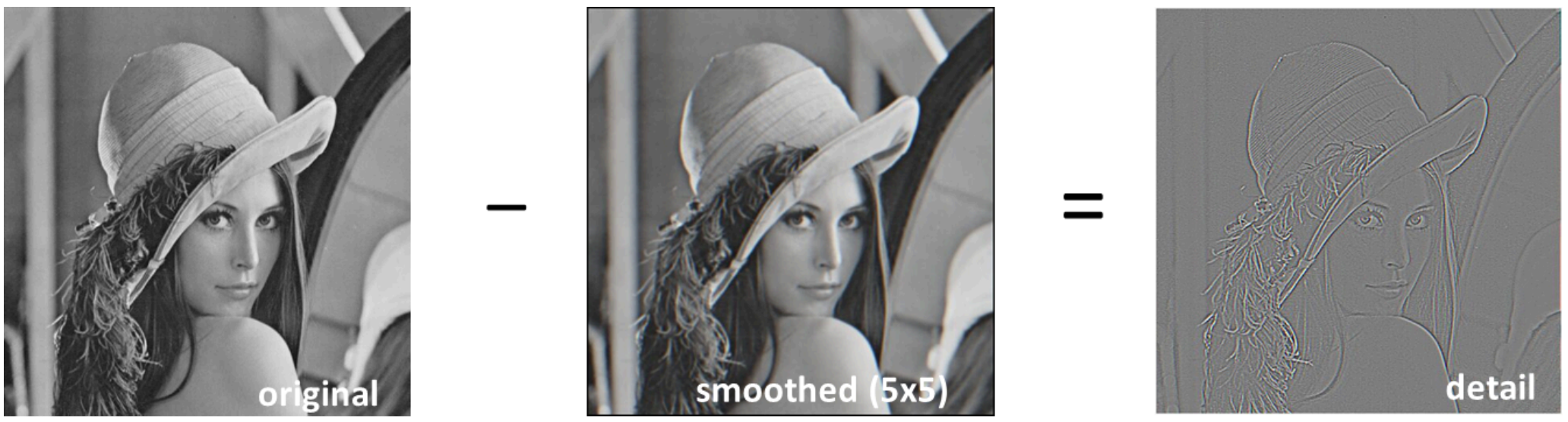
Constant factor at front makes volume sum to 1 (can be ignored, as we should re-normalize weights to sum to 1 in any case)

前部的常数因子使体积总和为 1（可以忽略，因为无论如何我们都应该将权重重新归一化，使其总和为 1）

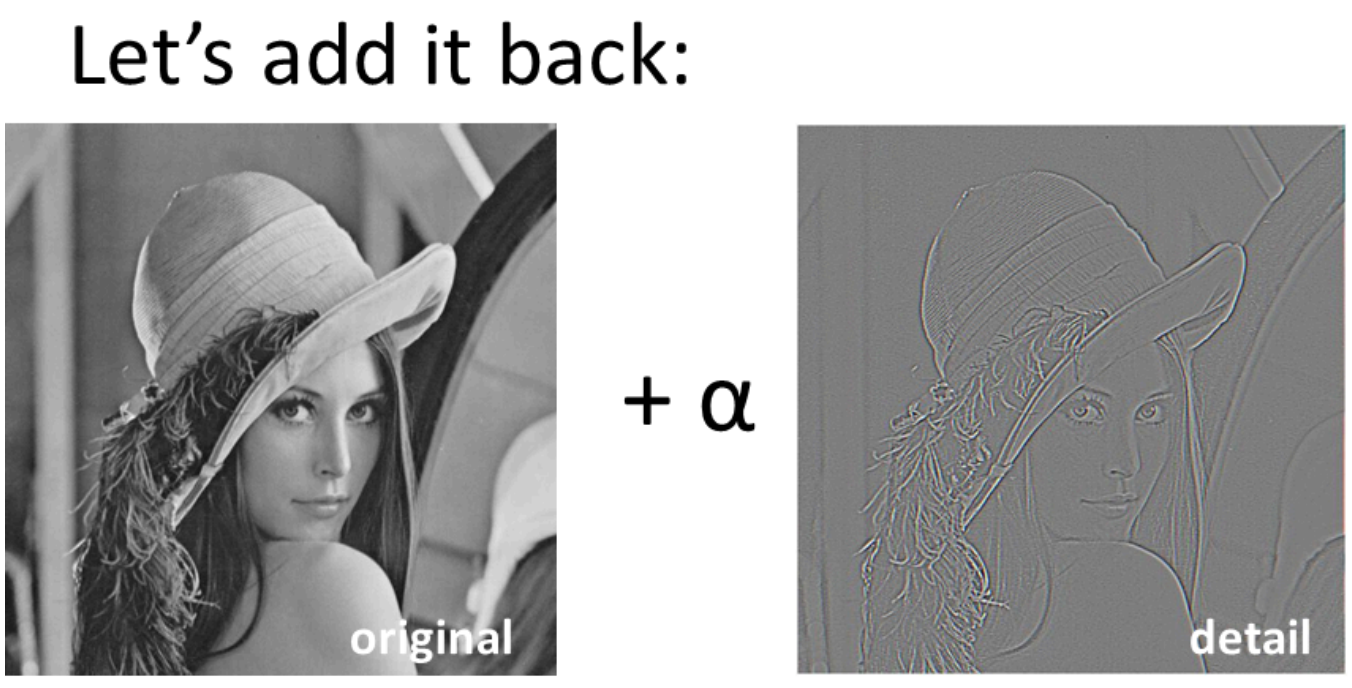
## Image Sharpening 图像锐化

We use original image to subtract smoothed image to get image detail

我们使用原始图像减去平滑图像，以获得图像细节




Then we add it back for sharpening, which emphasizes edge information 然后，我们再将其添加回去进行锐化，以强调边缘信息




# Edge Detection 边缘检测

## Image derivatives 图片导数

How can we differentiate a image and get its gradient? 如何微分图像并获取梯度?

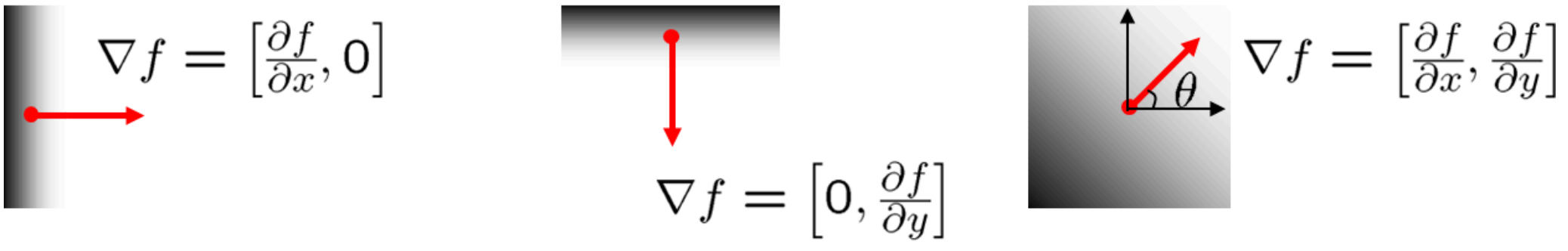
1. reconstruct a continuous image, *f,* then compute the derivative 重建连续图像 f，然后计算导数
2. take discrete derivative (finite difference) 取离散导数（有限差分）

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

## Image gradient 图像梯度

The gradient:

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



The **edge strength** is given by the **gradient magnitude**: 边缘强度由梯度大小决定：

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

The gradient magnitude reflects how much the brightness of that point in the image changes. The larger the value, the more drastic the change in brightness around the point, which usually corresponds to an edge or boundary in the image.

梯度大小反映了图像中该点亮度的变化程度。 数值越大，该点周围的亮度变化越剧烈，通常对应于图像中的边缘或边界。

## Gradient Direction 梯度方向

The **gradient direction** is given by: 梯度方向由以下公式给出：

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

## Gradient Direction

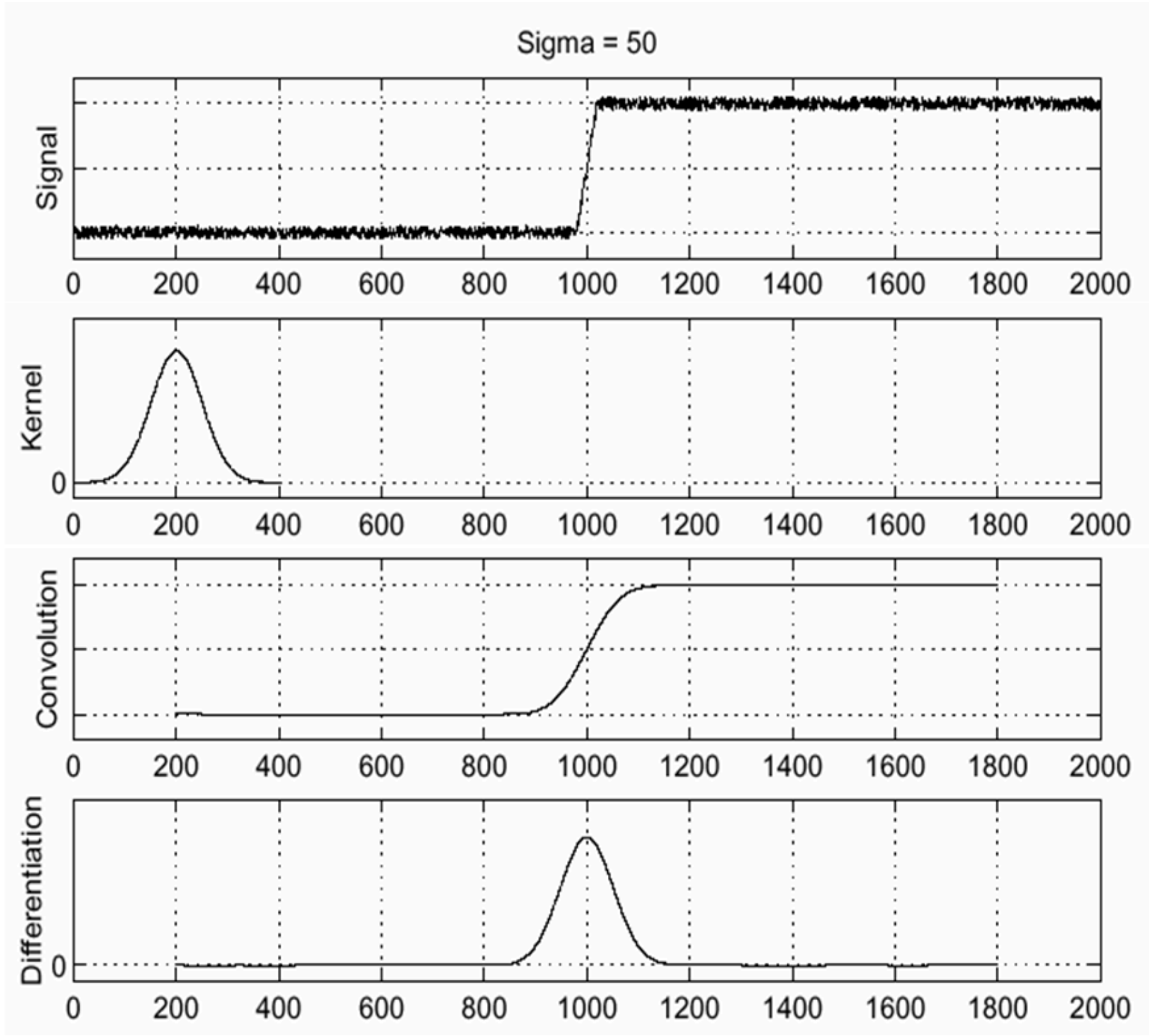
The **gradient direction** points in the direction of the fastest change in brightness. In other words, it points to the direction of the edge at that point in the image

梯度方向指向亮度变化最快的方向。 换句话说，它指向图像中该点的边缘方向

## smooth first 先平滑

One way to detect edge: smooth first.

Smoothing before differentiating 先平滑再微分




$f$

$h$

$f * h$

$$\frac{d}{dx}(f * h)$$

The edges lie in peaks. 边缘在峰值处。

Gaussian filter could be used here. 这里可以使用高斯滤波器。

## Sobel

Sobel operator is also a way of edge detection Sobel 算子也是一种边缘检测方法

It includes 2 kernal:

$$\frac{1}{8}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

$s_x$

$$\frac{1}{8}\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

$s_y$

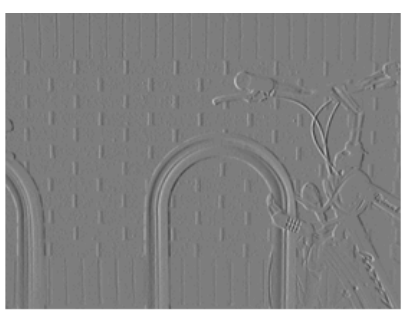
Using the original image to do convolution calculation with $s_x$ and $s_y$, we could get gradient on x and y directions

使用原始图像与 $s_x$ 和 $s_y$进行卷积计算，我们可以得到 x 和 y 方向上的梯度。

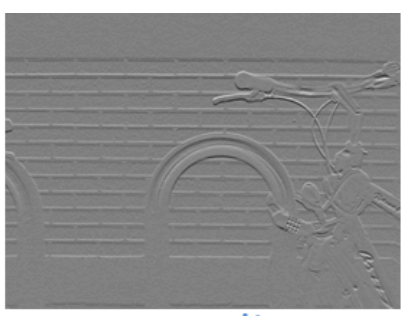
Then by calculating

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

We could get edge information



*x*-gradient



*y*-gradient

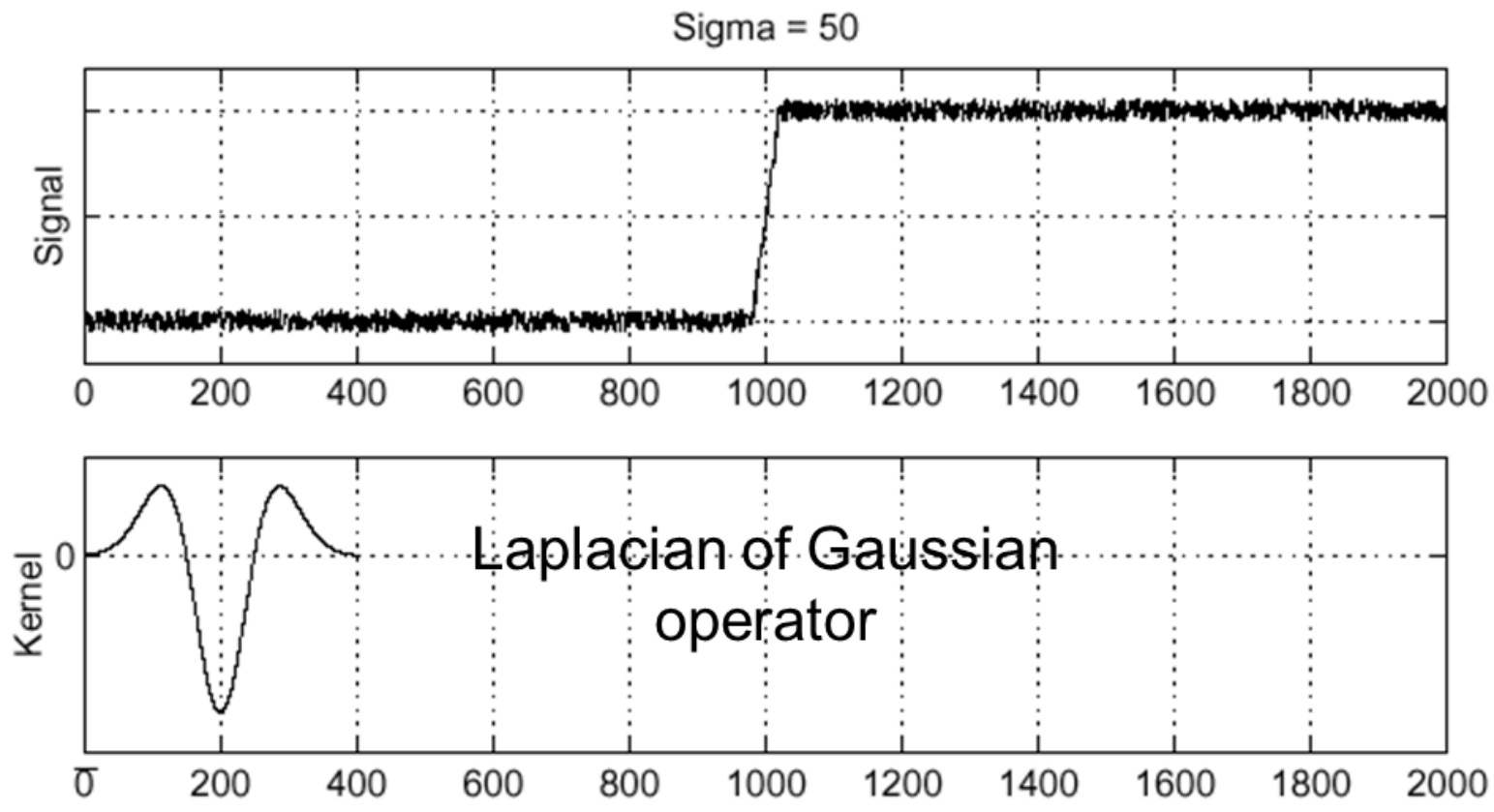
gradient magnitude

## Laplacian of Gaussian 高斯拉普拉斯

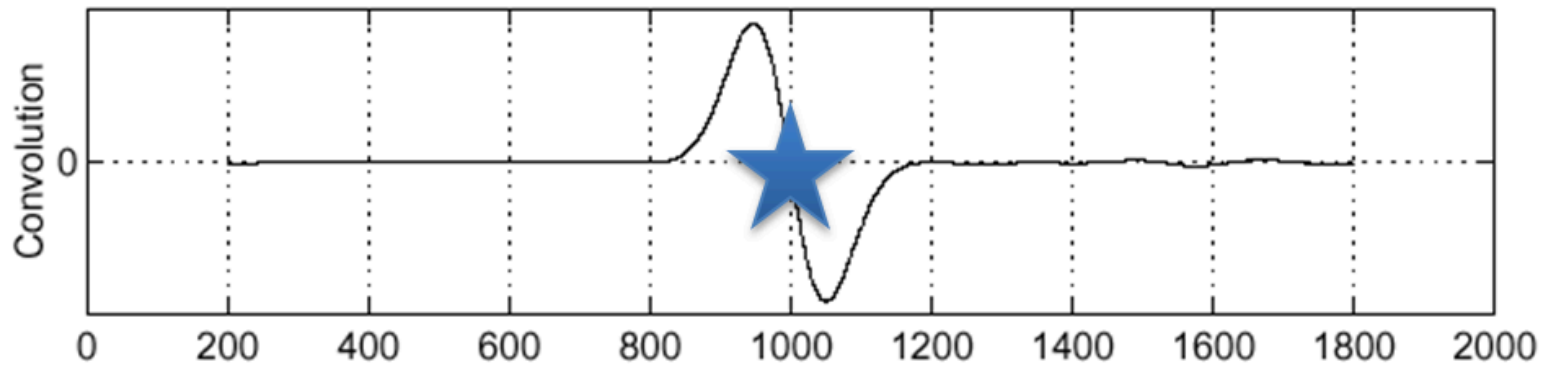
Laplacian of Gaussian, LoG is also useful in edge detection  LoG 也适用于边缘检测

$f$

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




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



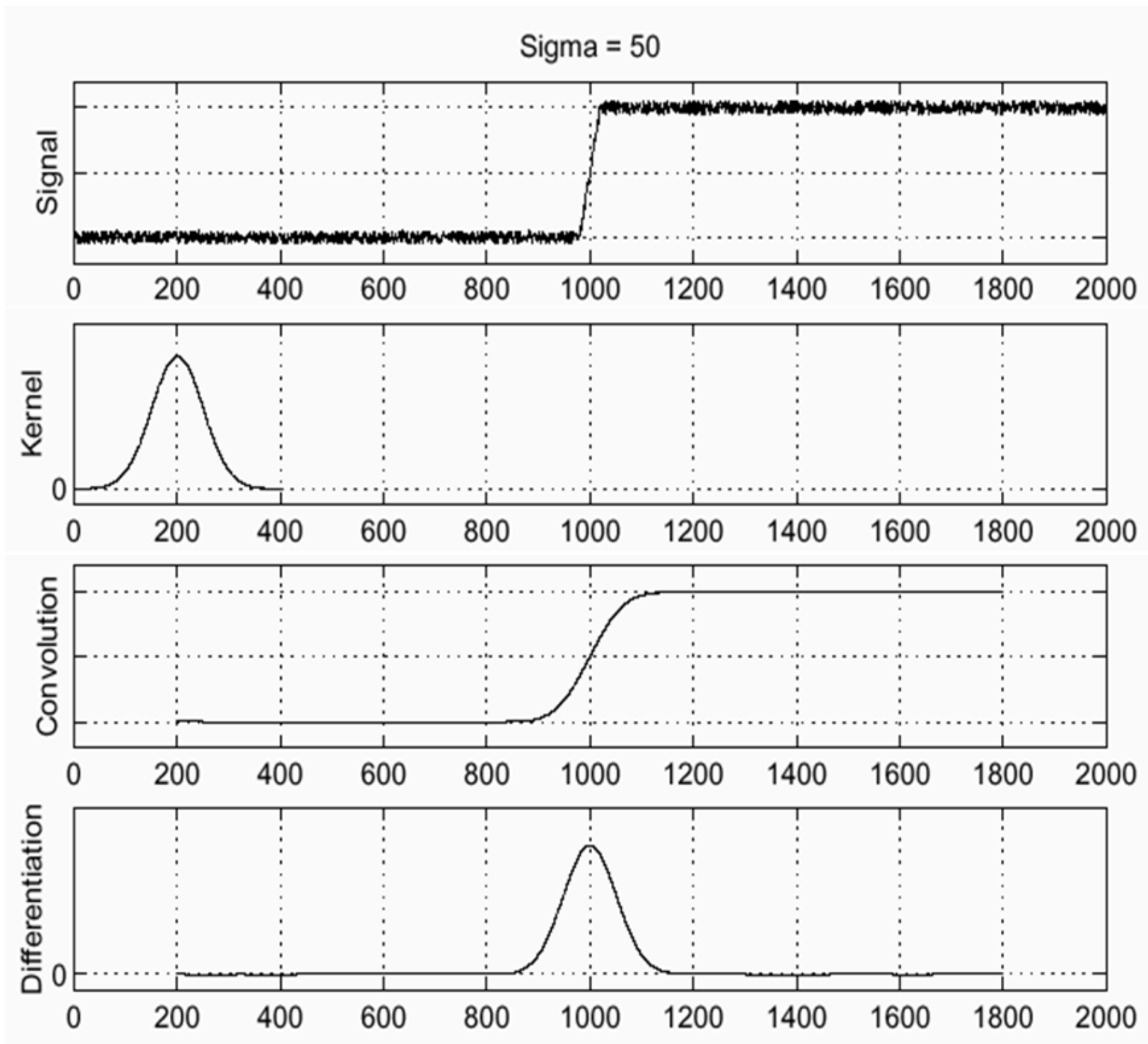

Convolute to smooth, second-order derivative to find the peak. 卷积平滑，二阶导数找到峰值。

$f$

$h$

$f * h$

$\frac{d}{dx}(f * h)$




**Attention**: Second-order derivative zero doesn't mean it's edge, it should be going across 0-line.

**注意**： 二阶导数为零并不意味着它是边缘，它应该穿过 0 线。

$\nabla^2$ is the **Laplacian** operator:

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

## Designing an edge detector:Canny edge detector

**Criteria for a good edge detector**: **良好边缘检测器的标准**:

- **Good detection:** the optimal detector should find all real edges, ignoring noise or other artifacts **良好检测**：最佳检测器应能找到所有真实边缘，忽略噪音或其他伪影
- **Good localization** 良好的定位
  - the edges detected must be as close as possible to the true edges 检测到的边缘必须尽可能接近真实边缘
  - the detector must return one point only for each true edge point 检测器必须为每个真正的边缘点只返回一个点

**Cues of edge detection 边缘检测线索**

- Differences in color, intensity, or texture across the boundary 边界的颜色、强度或纹理差异
- Continuity and closure 连续性和封闭
- High-level knowledge 高级知识

**Canny edge detector**

very widely used 广泛使用

1. Filter image with derivative of Gaussian (smooth) 用高斯导数（平滑）过滤图像
2. Find magnitude and orientation of gradient(find edge) 查找梯度的大小和方向（查找边缘）

3. Non-maximum suppression(remove redundant edge point) 非最大抑制（去除多余的边缘点）

4. Linking and thresholding (hysteresis):(ensure edge continuity and integrity) 链接和阈值（滞后）：（确保边缘的连续性和完整性）

   –Define two thresholds: low and high 定义两个阈值：低和高

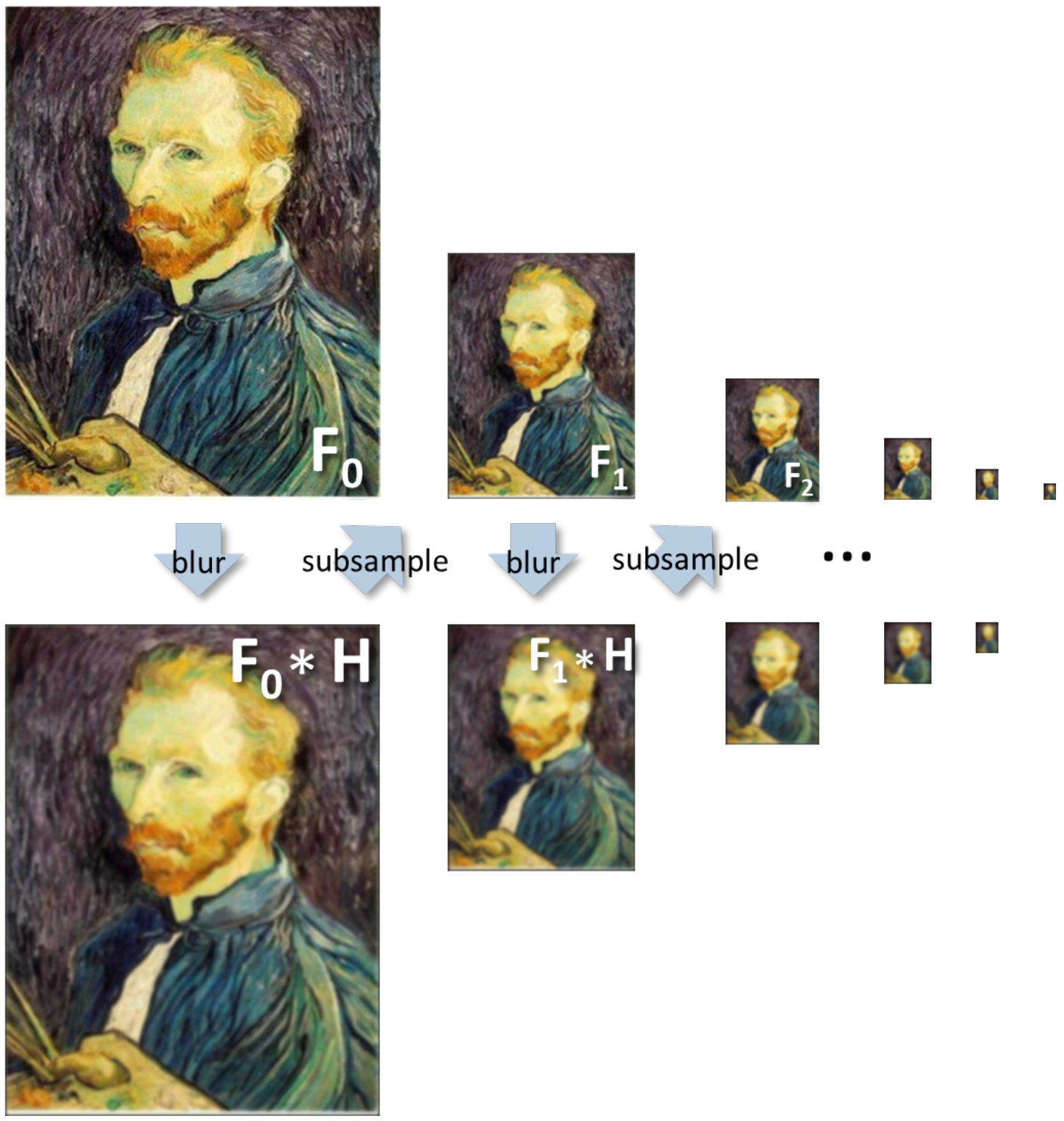
   –Use the high threshold to start edge curves and the low threshold to continue them 使用高阈值启动边缘曲线，使用低阈值延续边缘曲线

# Image Sampling 图像采样

## Gaussian pyramid 高斯金字塔

Repeat blurring and subsampling, finally get a pyramid-like structure 重复模糊和子采样，最终得到金字塔状结构

*Gaussian pyramid*




## Upsampling 升采样

The simplest way: 最简单方式

Nearest neighbor interpolation: repeat each row and column 10 times 近邻插值：每行每列重复 10 次

# Color and Texture 色彩与纹理

## Color Spaces 色彩空间

RGB: standard for cameras 相机标准

Red, Green, Blue, each component value is set from 0 to 255, which represents intensity 红、绿、蓝，每个分量的值设置为 0 至 255，代表强度

HSI/HSV: hue, saturation, intensity 色调、饱和度、强度

hue: represent the basic types of colors, such as red, green, blue, etc. 色调：表示颜色的基本类型，如红色、绿色、蓝色等。

saturation: Indicates the purity or intensity of the color, the higher the saturation, the more vibrant the color is 饱和度： 表示颜色的纯度或强度，饱和度越高，颜色越鲜艳

intensity or value: Indicates the brightness or darkness of a color 强度或值： 表示颜色的亮度或暗度

YIQ: mainly used in analog TV broadcasting systems, especially NTSC standard 主要用于模拟电视广播系统，尤其是 NTSC 标准

Y: indicates the brightness information of the image 表示图像的亮度信息

I and Q: Indicates color information I 和 Q：表示颜色信息

## Histograms 直方图

- A histogram of a gray-tone image is an array $H[*]$ of bins, one for each gray tone.

H with length of 256 means pixel value is from 0 to 255 长度为 256 的 H 表示像素值为 0 至 255

- H[i] gives the count of how many pixels of an image have gray tone i. H[i]表示图像中有多少像素的灰度为 i。

If H[100]=50, means there are 50 pixels in the image with a gray value of 100 如果 H[100]=50 表示图像中有 50 个灰度值为 100 的像素

- P[i] (the normalized histogram) gives the percentage of pixels that have gray tone i. P[i]（归一化直方图）表示灰阶为 i 的像素百分比。

This value comes from each H[i] divides total pixel value. Such as 10000 pixels, H[100]=50, P[100]=50/10000=0.005 该值来自每个 H[i] 除以总像素值。 例如 10000 像素，H[100]=50，P[100]=50/10000=0.005

It can be used to match images

## Two Edge-based Texture Measures 两种基于边缘的纹理测量方法

1. **Edgeness per unit area**：**单位面积**：

$$F_{edgeness} = |\{ p \mid gradient\_magnitude(p) \geq threshold\}| / N$$

gradient_magnitude(*p*) means gradient value of pixel p gradient_magnitude(*p*) 表示像素 p 的梯度值

Count the number of edge points in the unit area that satisfy the condition and divide by the size of the unit area N to get the edge degree in the unit area

计算单位区域内满足条件的边缘点数量，除以单位区域的大小 N，得到单位区域内的边缘度

2. **edge magnitude and direction histograms 边缘幅度和方向直方图**

$$F_{magdir} = ( H_{magnitude}, H_{direction} )$$

Normalized histogram of gradient magnitude and histogram of gradient direction 梯度大小的归一化直方图和梯度方向的直方图

$H_{magnitude}$：Normalized histogram of gradient magnitude 梯度大小的归一化直方图

$H_{direction}$：Normalized histogram of gradient direction 梯度方向的归一化直方图

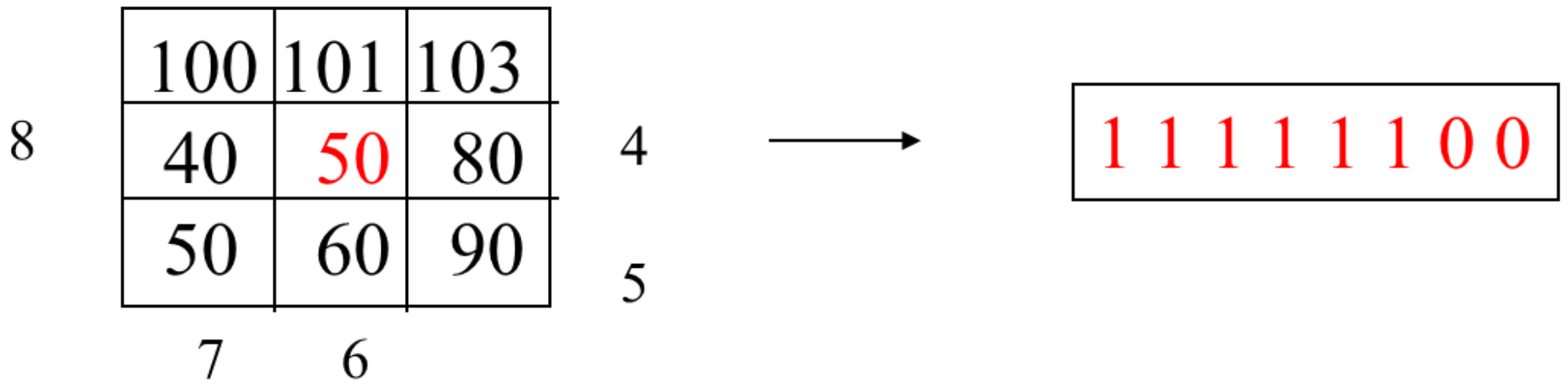
## Local Binary Pattern Measure 局部二进制模式测量

A technique for image texture analysis 图像纹理分析技术

For each pixel p, create an 8-bit number b1 b2 b3 b4 b5 b6 b7 b8 为每个像素 p 创建一个 8 位数字 b1 b2 b3 b4 b5 b6 b7 b8

if $b_i = 0$, neighbor i has value less than or equal to p's value and 1 otherwise. 若$b_i = 0$,邻居 i 的值小于或等于 p 的值，否则为 1。

Represent the texture in the image (or a region) by the histogram of these numbers. 用这些数字的直方图来表示图像（或区域）的纹理。




The following is math representation:

$$LBP_{p,r}(N_c) = \sum_{p=0}^{P-1} g\left(N_p - N_c\right)2^p$$

$N_c$: center pixel
$N_p$: neighbor pixel
$r$: radius (for 3x3 cell, it is 1).

binary threshold function $g(x)$ is,

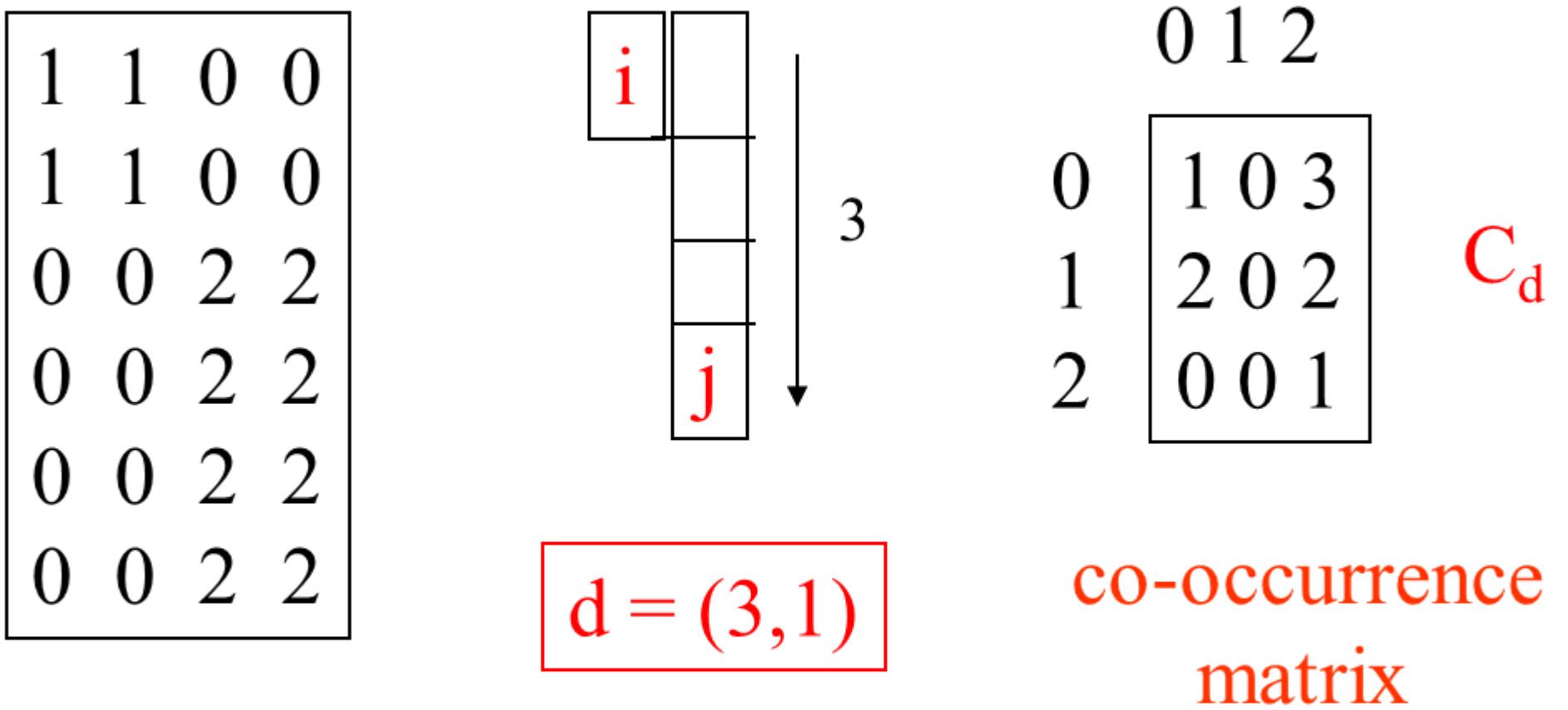
$$g(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

## Co-occurrence Matrix Features 共现矩阵特征

Used to describe texture characteristic 用于描述纹理特征

$C_d(i,j)$ indicates the co-occurrence number of i and j in particular space relationship $C_d(i,j)$表示 i 和 j 在特定空间关系中的共现次数

$d = (d_r, d_c)$ represents space relationship,indicates row and column offsets, respectively $d = (d_r, d_c)$ 表示空间关系，分别表示行和列的偏移量



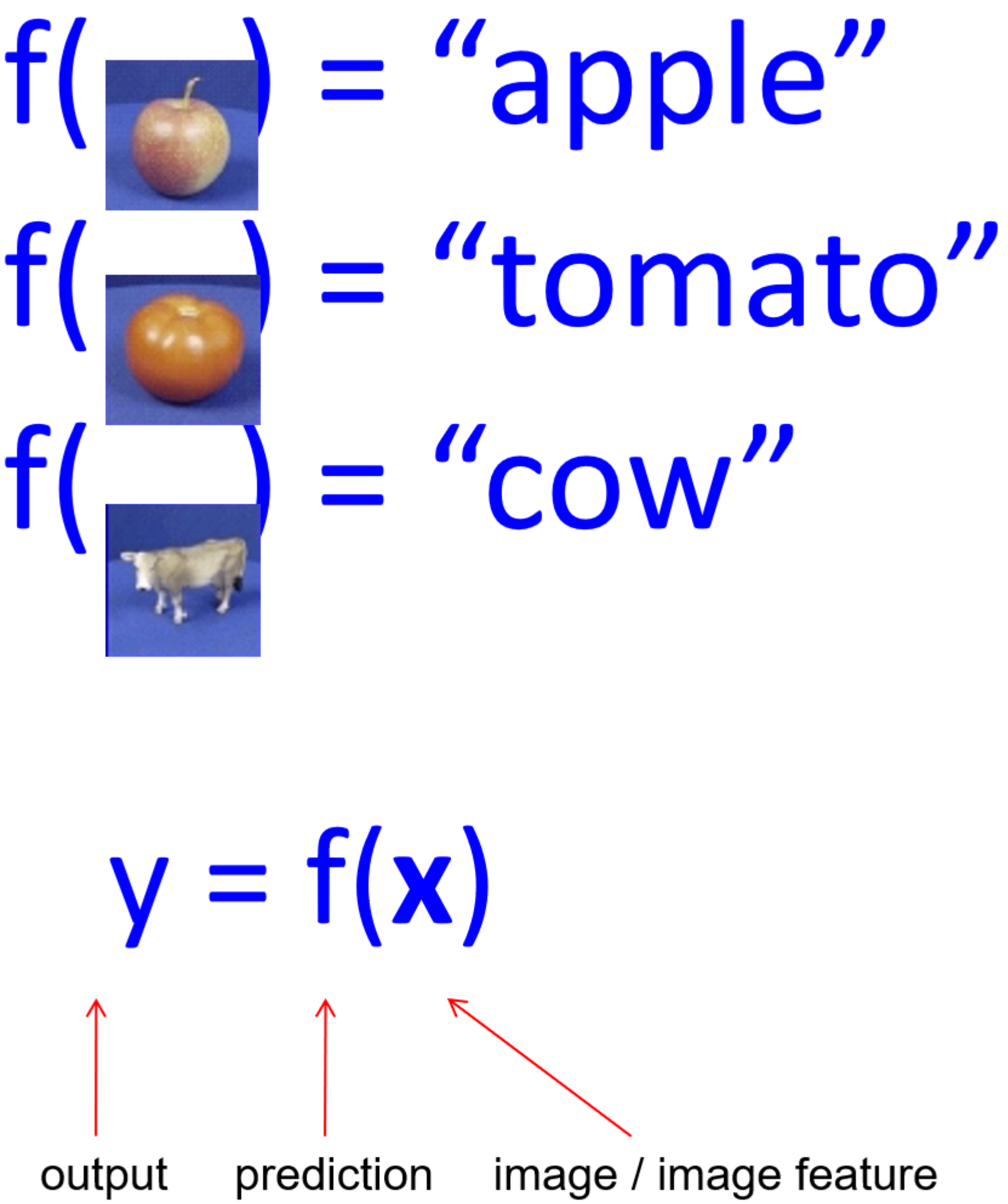

from 0 to 0, there is only 1 time, so $C_d[0,0] = 1$ 从 0 到 0，只有 1 次，所以 $C_d[0,0] = 1$

# Convolutional Neural Network 卷积神经网络

## Learning Framework 学习框架

Machine learning is to apply a prediction function to a feature representation of the image to get the desired output:

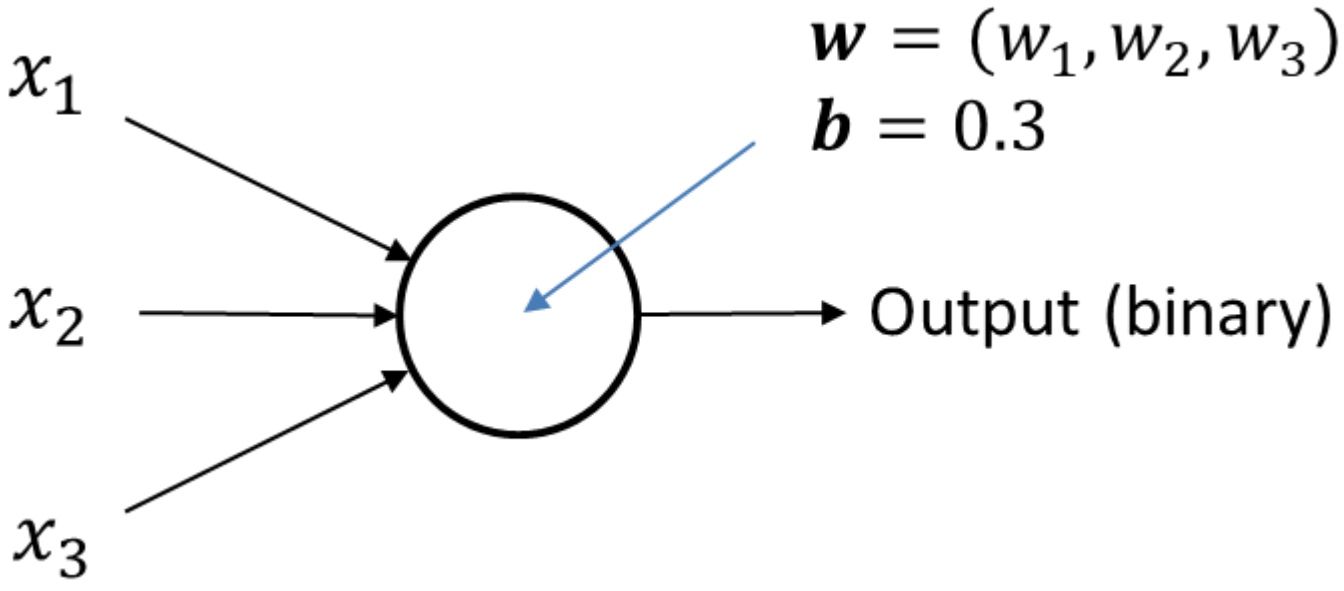
机器学习是将预测函数应用于图像的特征表示，以获得所需的输出：




- **Training:** given a *training set* of labeled examples $\{(x_1, y_1), \ldots, (x_N, y_N)\}$, estimate the prediction function $f$ by minimizing the prediction error on the training set **训练：** 给定标有示例的训练集 { $(x1,y1),\ldots,(xN,yN)$ }，通过最小化训练集上的预测误差来估计预测函数 $f$
- **Testing:** apply f to a never before seen *test example* **x** and output the predicted value y = f(**x**) **测试：** 将 f 应用于从未见过的测试示例 x，并输出预测值 y = f(x)

## Neural Networks 神经网络

Basic building block for composition is a *perceptron* 组成的基本构件是感知器



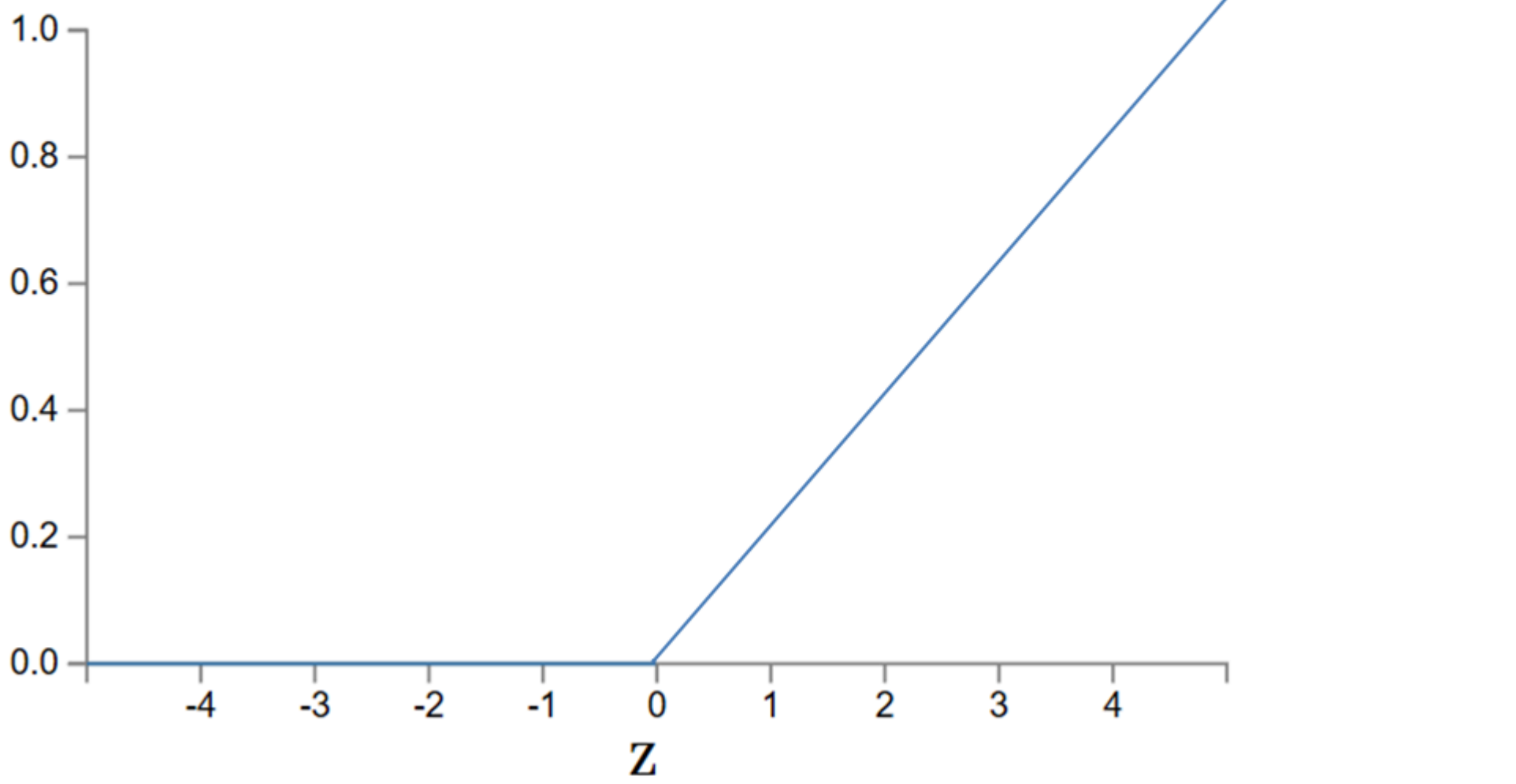

Input vector $x = (x_1, x_2, x_3)$, weight vector $w = (w_1, w_2, w_3)$, bias b=0.3

$$\text{output} = \begin{cases} 0 & \text{if } w \cdot x + b \leq 0 \\ 1 & \text{if } w \cdot x + b > 0 \end{cases}$$

$$w \cdot x \equiv \sum_j w_j x_j$$

**Rectified Linear Unit 线性单位**

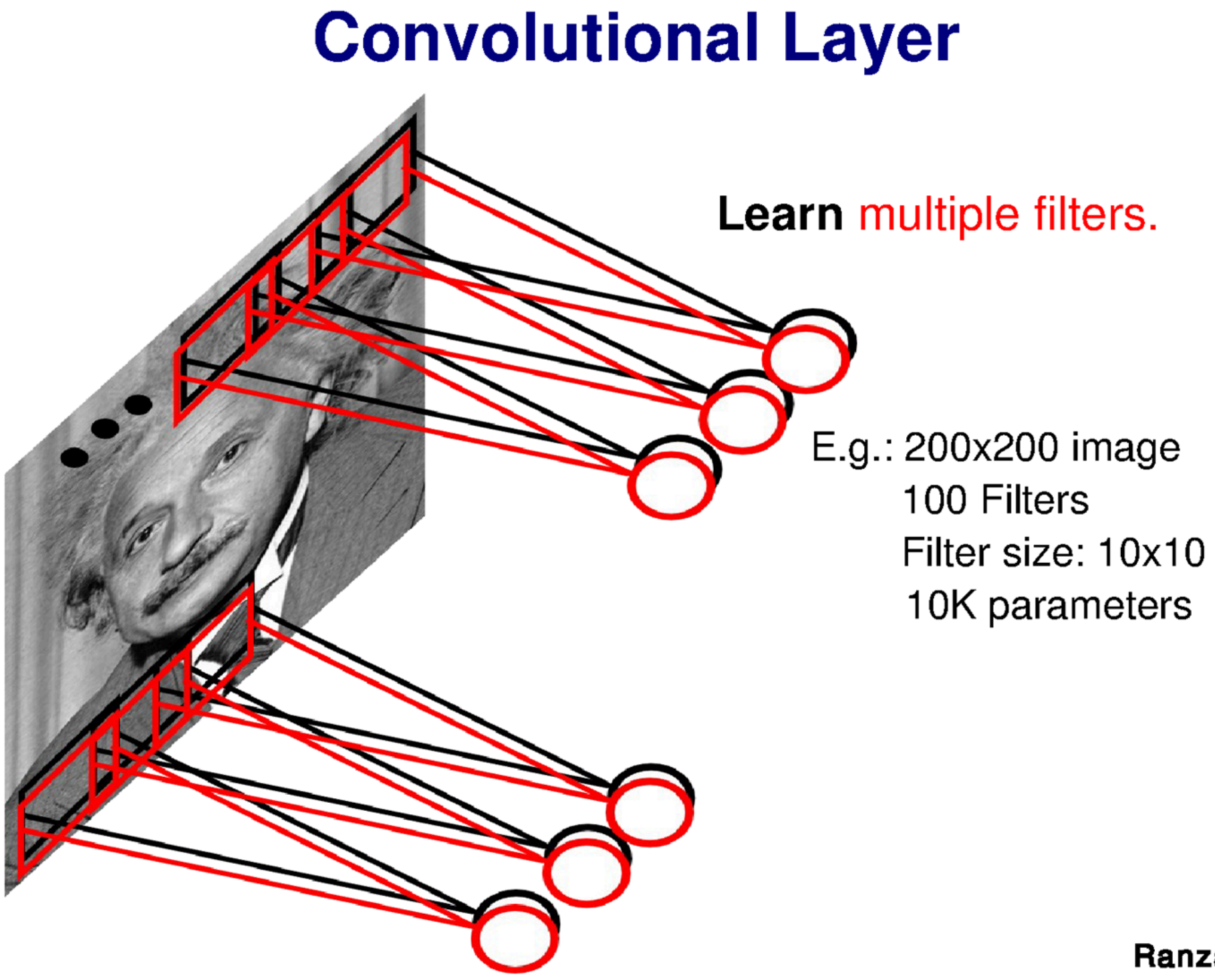
ReLU: a activating function. ReLU：一个激活函数。




Very efficient when dealing with large amounts of data and helps to avoid the problem of vanishing gradients

处理大量数据时非常高效，有助于避免梯度消失问题

## Convolutional Layer 卷积层



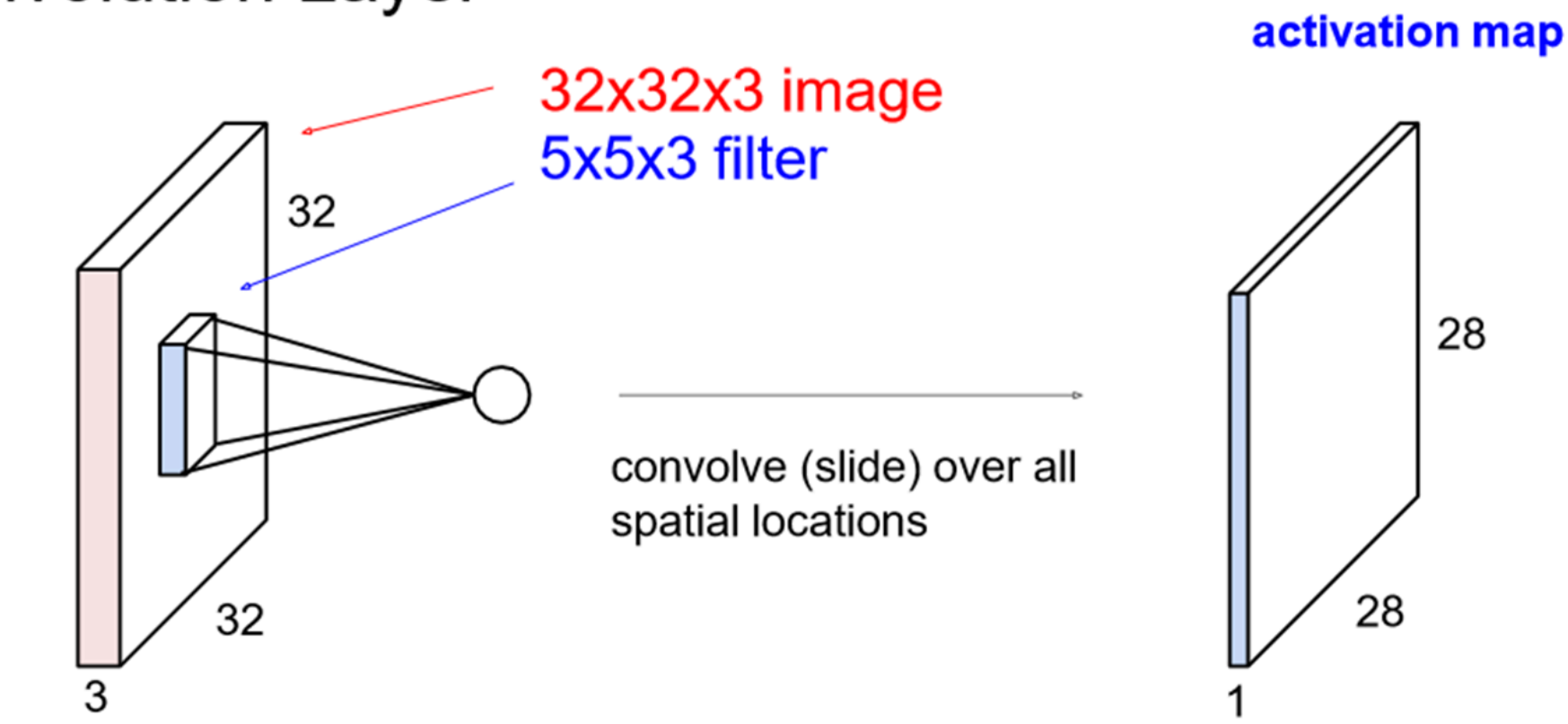

Core component in CNN, used to extract local features of an image CNN 的核心组件，用于提取图像的局部特征

Learning different features of an image through multiple filters 通过多重滤波器学习图像的不同特征

Each filter slides over the image to generate a feature map 每个滤镜在图像上滑动，生成特征图

# Convolutions: More detail

## Convolution Layer



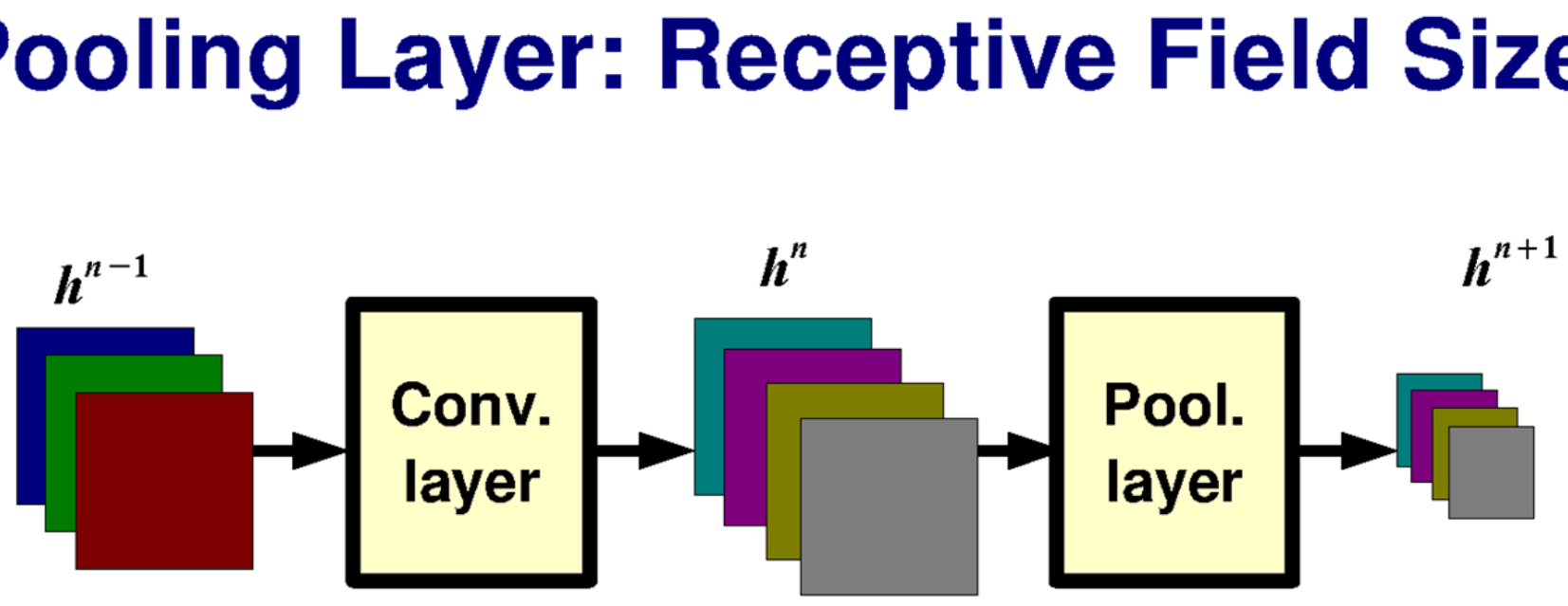

Andrej Karpathy

The convolutional layer slides over the input image through a filter to extract local features 卷积层通过滤波器在输入图像上滑动，以提取局部特征

The filter performs a dot product operation with each region on the image to generate a new 2D matrix called the activation map 滤波器对图像上的每个区域进行点乘运算，生成一个新的二维矩阵，称为激活图

With these steps, the convolutional layer is able to efficiently extract localized features from the image 通过这些步骤，卷积层能够有效地从图像中提取局部特征

## Pooling Layer 池化层



The pooling layer reduces computational complexity and reduces the risk of overfitting by **reducing the size of the feature map** and **retaining the most important information** 池化层通过**减少特征图的大小**和**保留最重要的信息**，降低了计算复杂度，并减少了过度拟合的风险。

## Pooling Layer: Examples

Max-pooling:

$$h_j^n(x, y) = max_{\bar{x} \in N(x),\, \bar{y} \in N(y)} h_j^{n-1}(\bar{x}, \bar{y})$$

Average-pooling:

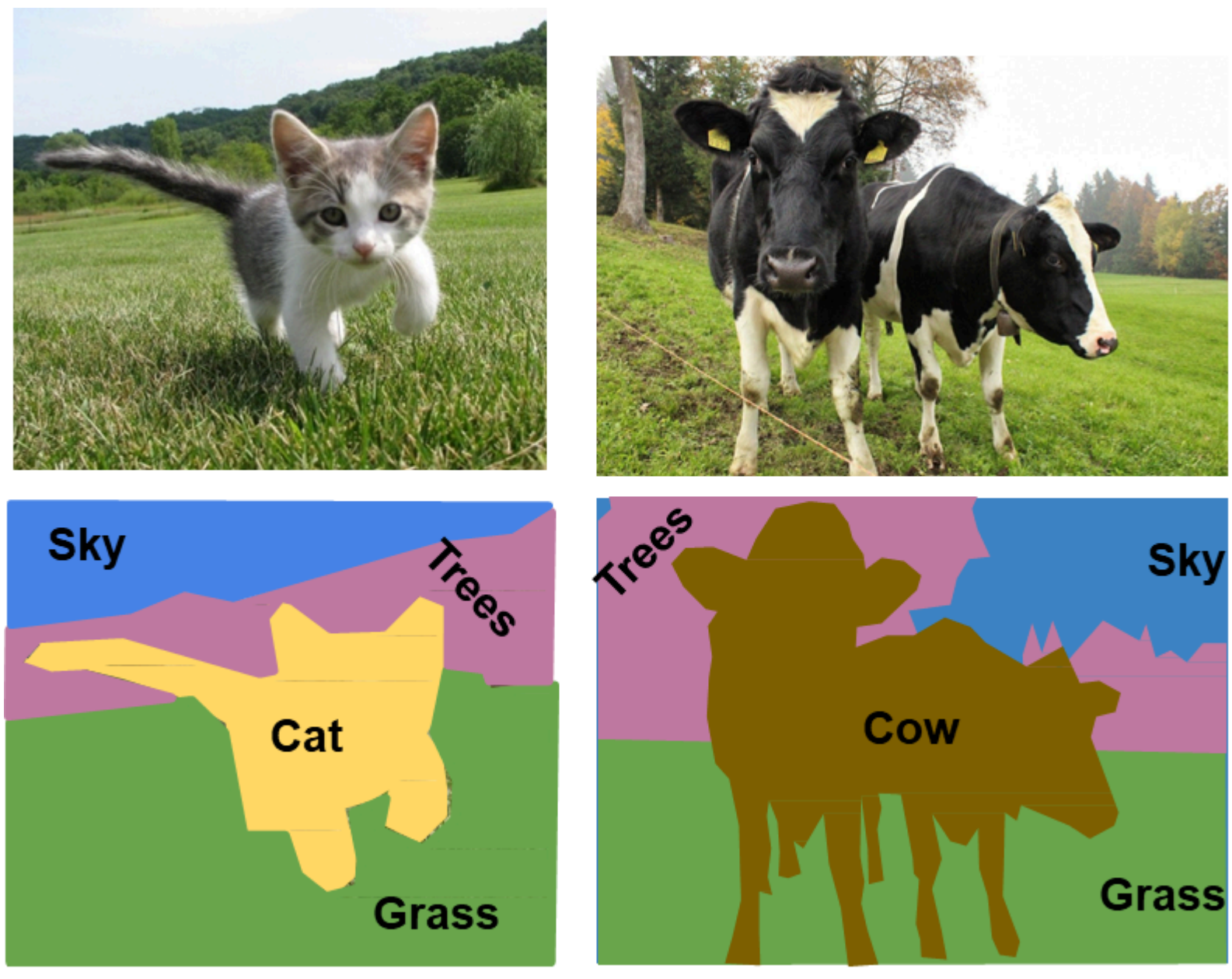
$$h_j^n(x, y) = 1/K \sum_{\bar{x} \in N(x),\, \bar{y} \in N(y)} h_j^{n-1}(\bar{x}, \bar{y})$$

2 common types

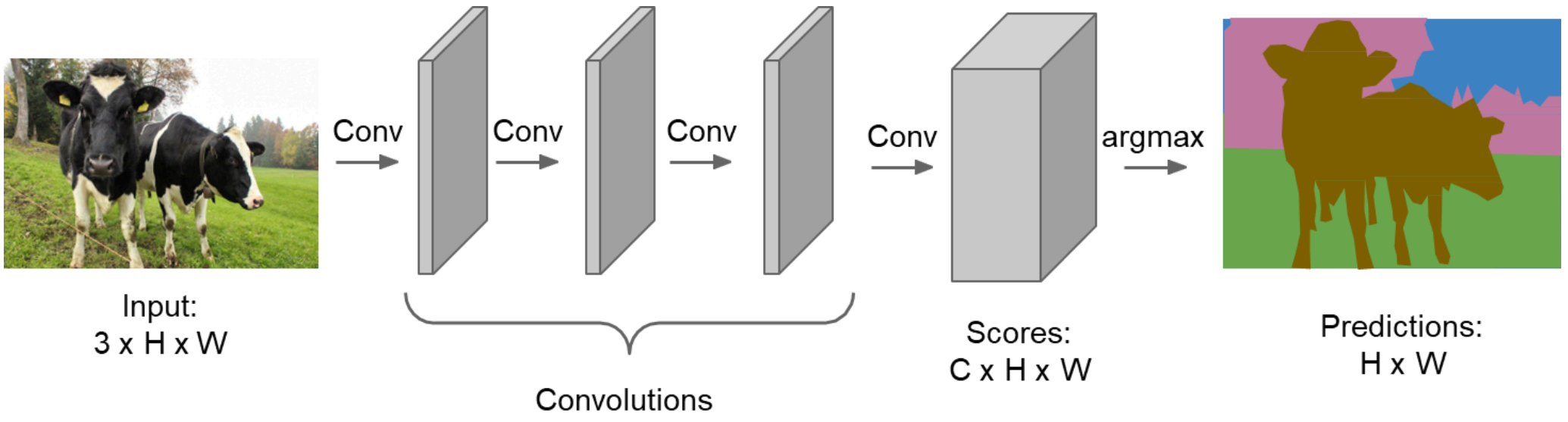
# Image Segmentation 图像分割

## Semantic Segmentation 语义分割

To label each pixel in the image with a category label 为图像中的每个像素标注类别标签




Don’t differentiate instances, only care about pixels 不区分实例，只关注像素



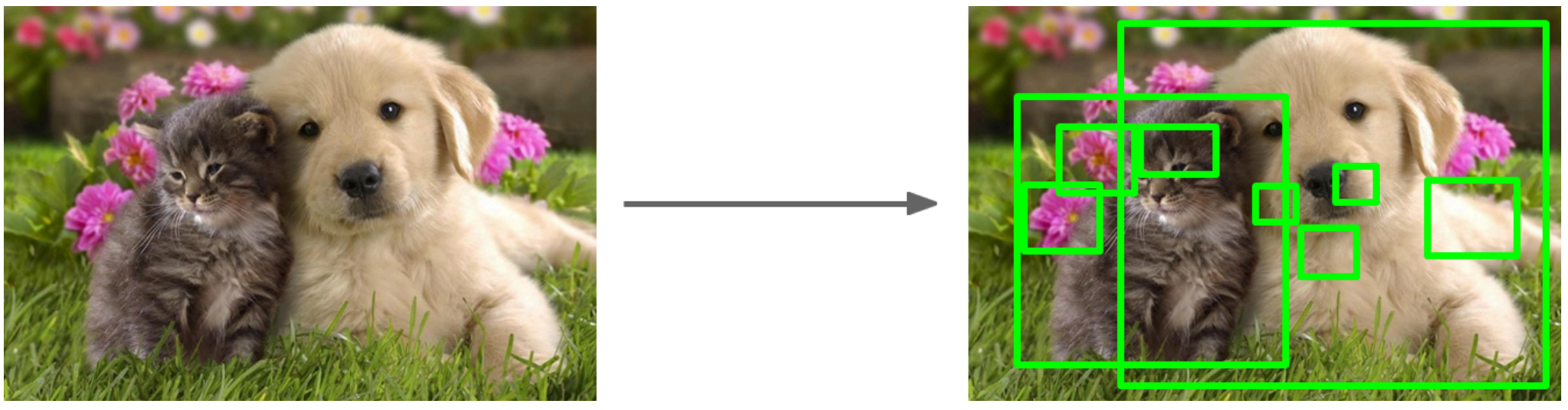

Input image-> multiple convolution layers to extract feature, each layer generates a new feature map. 输入图像-> 多个卷积层提取特征，每一层生成一个新的特征图。

Each convolution layer could generate a map with multiple channels, each channel is a class 每个卷积层都能生成一个包含多个通道的图，每个通道都是一个类

C channels->C classes　C 个通道->C个 类

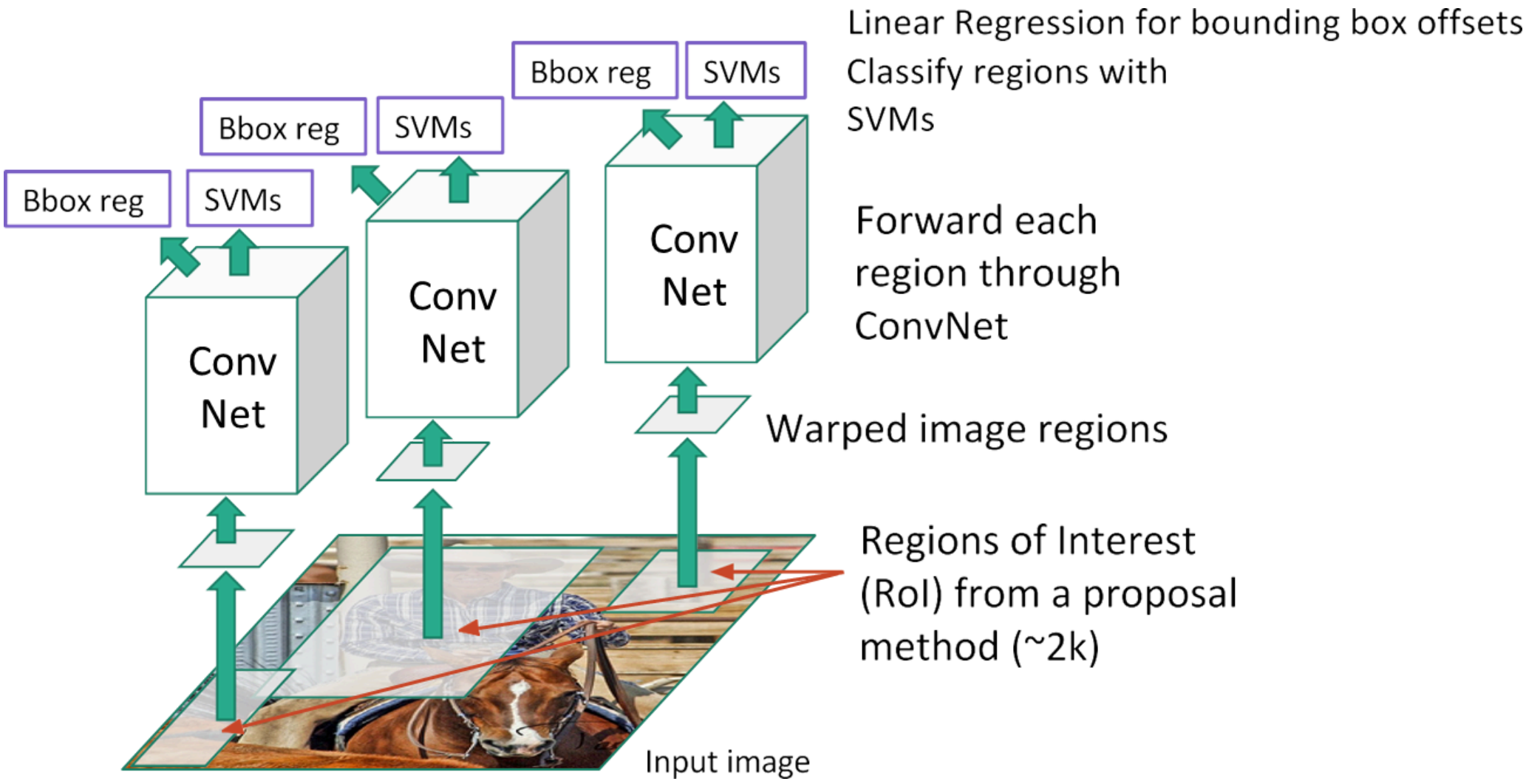
## Region Proposals 区域建议

Used to find image regions that are likely to contain objects 用于查找可能包含物体的图像区域



Relatively fast to run；e.g. Selective Search gives 1000 region proposals in a few seconds on CPU 运行速度相对较快；例如，选择性搜索只需几秒钟就能在 CPU 上给出 1000 个区域建议

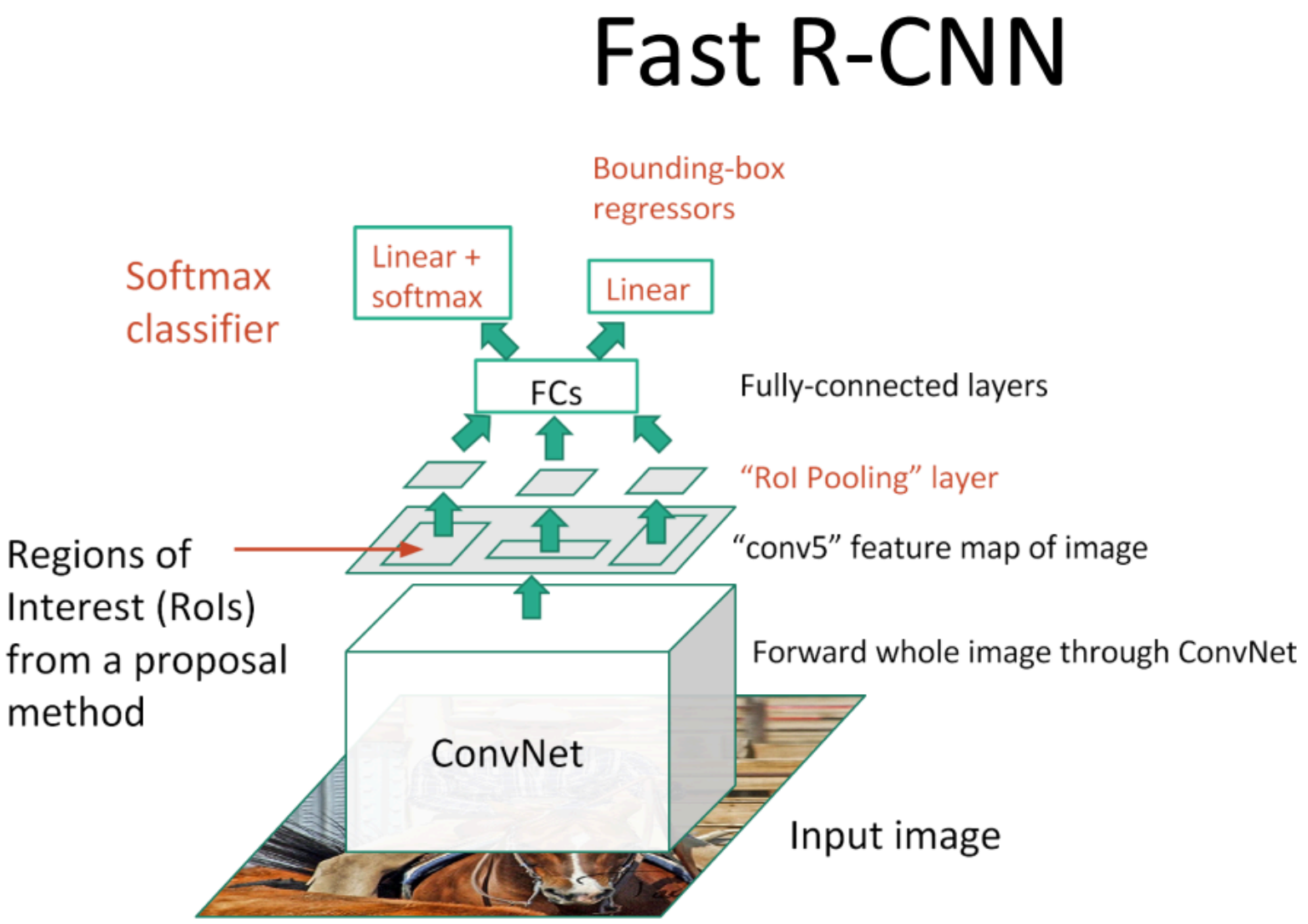
# R-CNN




A deep learning model for target detection. 目标检测深度学习模型

1. input image 输入图像
2. region proposal 区域建议
3. ConvNet to extract feature 卷积神经网络提取特征
4. Bounding Box Regression to trim bounding box, use SVMs to classify each region 边框回归修剪边框，使用 SVM 对每个区域进行分类

## Fast R-CNN

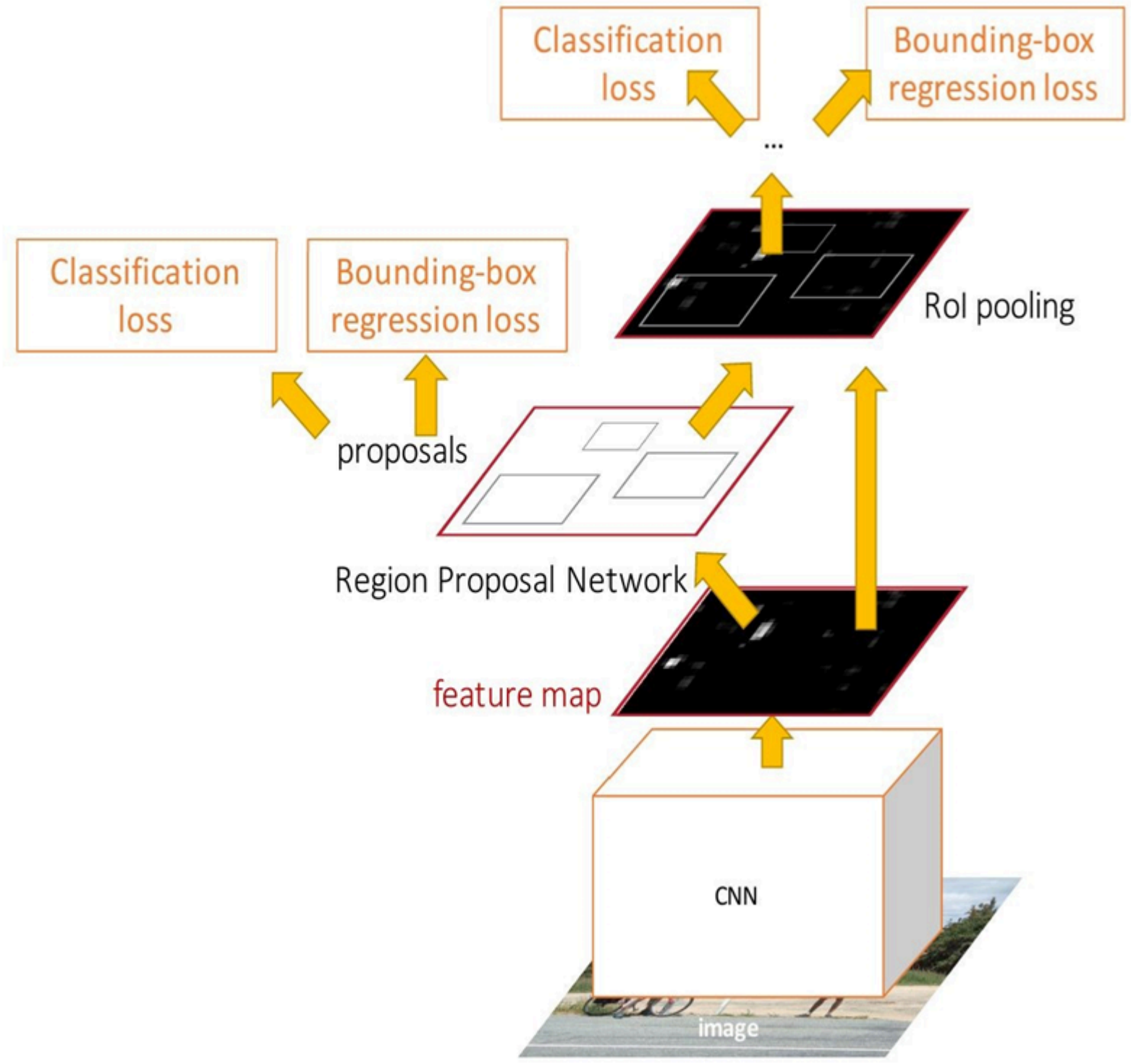



*Girshick, "Fast R-CNN", ICCV 2015*

1. Input 输入
2. ConvNet 卷积神经网络
3. Region proposal 区域建议
4. RoI Pooling layer to crop each region from the feature map to get feature map with fixed size RoI Pooling 图层用于裁剪特征图中的每个区域，以获得固定大小的特征图
5. Fully-connected layers: Further process croped feature map, including linear layer(Adjusting the position and size of the candidate area) and softmax classifier(classify) 完全连接层： 进一步处理裁剪后的特征图，包括线性层（调整候选区域的位置和大小）和 softmax 分类器（分类）

## Faster R-CNN

It allows CNN to do proposal jobs 它使 CNN 能够开展提案工作

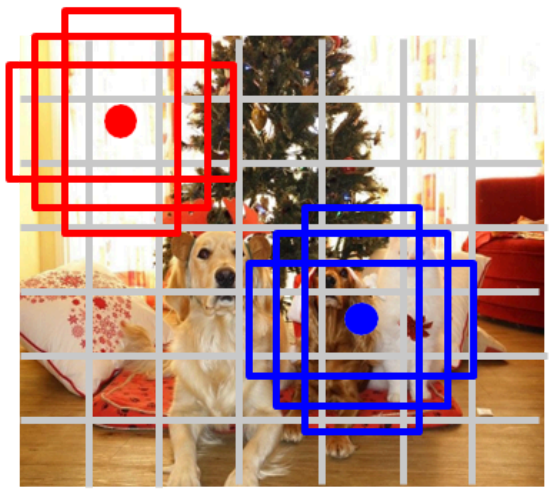



1. Input
2. CNN
3. RPN to predict candidate regions and output classification loss and Bounding-box regression loss RPN 预测候选区域，并输出分类损失和边界框回归损失
4. RoI pooling crosp candidate regions from feature map and convert to fixed size 从特征图中提取候选区域，并转换为固定大小的区域
5. Final loss function includes classification loss and bounding-box regression loss 最终损失函数包括分类损失和边界框回归损失

## YOLO

Detection without Proposals 无提议检测

Input image
3 x H x W



Divide image into grid 7 x 7

Image a set of **base boxes** centered at each grid cell Here B = 3

Within each grid cell:
•Regress from each of the B base boxes to a final box with 5 numbers:(dx, dy, dh, dw, confidence)
•Predict scores for each of C classes (including background as a class)

Output:
7 x 7 x (5 * B + C)

1. Input image 输入图片

2. divide image into grids, each grid has multiple base boxes 将图像划分为网格，每个网格有多个基框

3. For each base box, predict 5 value: $(dx, dy, dh, dw, confidence)$ 为每个基箱预测 5 个值

# Corners and blobs 边角和圆球

## Corner detection 边缘检测

$$E(u,v) \approx Au^2 + 2Buv + Cv^2$$

$$\approx \begin{bmatrix} u & v \end{bmatrix} \underbrace{\begin{bmatrix} A & B \\ B & C \end{bmatrix}}_{H} \begin{bmatrix} u \\ v \end{bmatrix}$$

$$A = \sum_{(x,y)\in W} I_x^2 \qquad B = \sum_{(x,y)\in W} I_x I_y \qquad C = \sum_{(x,y)\in W} I_y^2$$

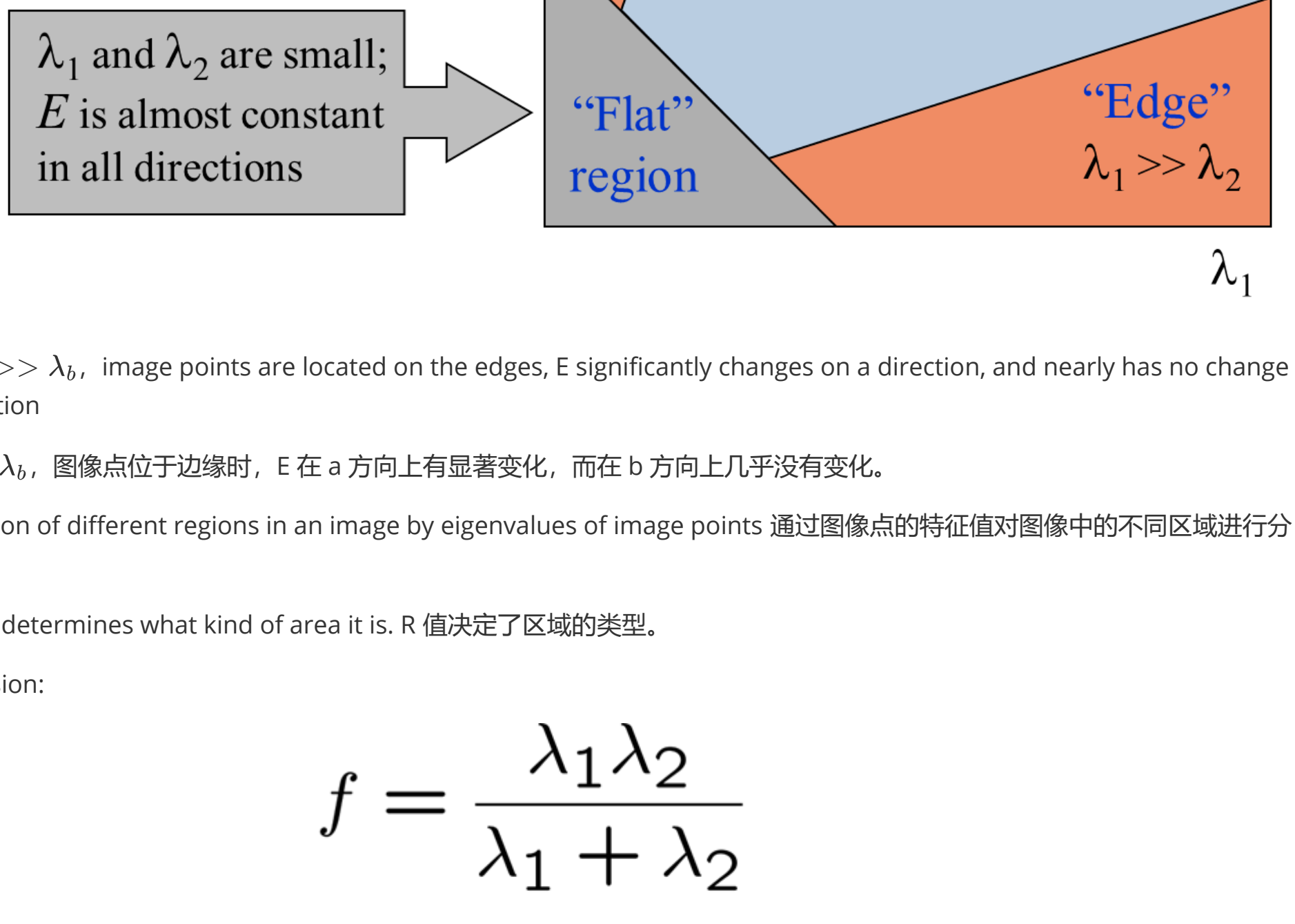
The surface $E(u, v)$ is locally approximated by a quadratic form. 曲面 $E(u,v)$局部近似于二次方程形式。

Efficiently recognizes corner points in an image and provides precise location information 有效识别图像中的角点并提供精确的位置信息

## Interpreting the eigenvalues

$$R = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2$$

R is large for corner
R is negative (with large magnitude) for edge
R is small for flat region




When $\lambda_a >> \lambda_b$，image points are located on the edges, E significantly changes on a direction, and nearly has no change on b direction

当$\lambda_a >> \lambda_b$，图像点位于边缘时，E 在 a 方向上有显著变化，而在 b 方向上几乎没有变化。

Classification of different regions in an image by eigenvalues of image points 通过图像点的特征值对图像中的不同区域进行分类

Value of R determines what kind of area it is. R 值决定了区域的类型。

Other version:

$$f = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2} = \frac{determinant(H)}{trace(H)}$$

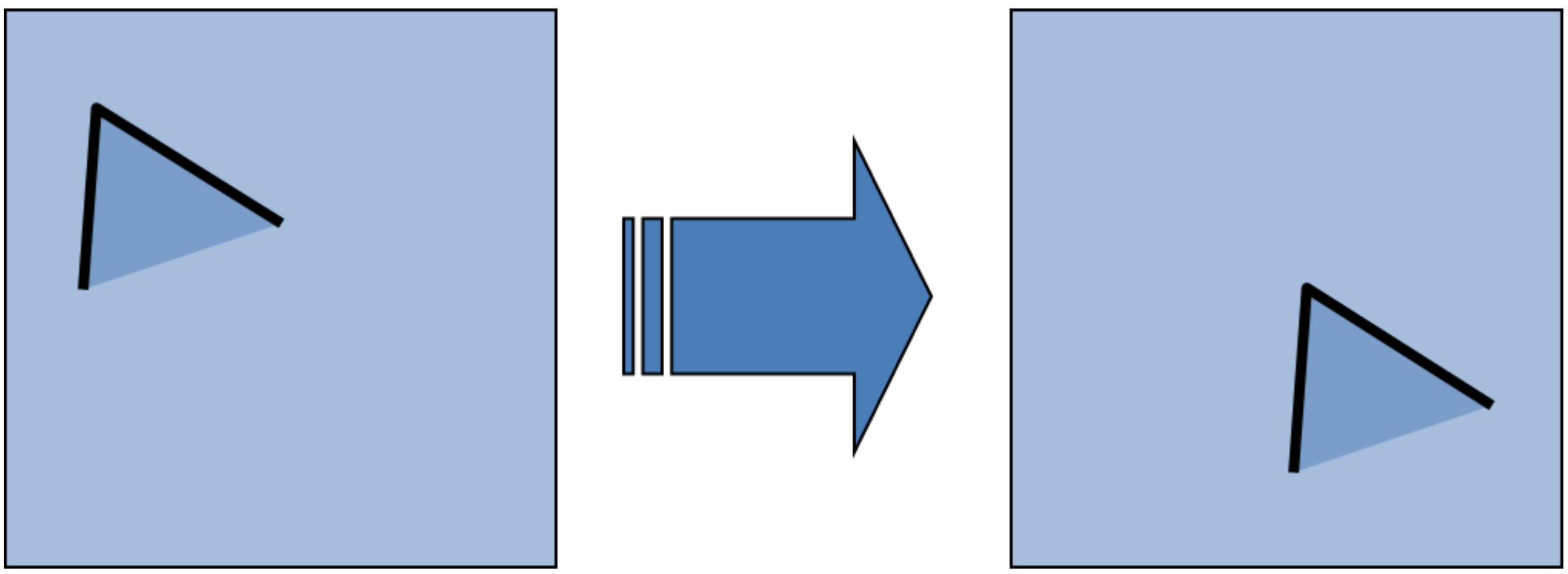
## Harris detector 哈里斯探测器

We want corner locations to be *invariant* to photometric transformations and *covariant* to geometric transformations 我们希望角位置与光度变换*不变*，*与几何变换*共变

- **Invariance:** image is transformed and corner locations do not change 不变性：图像经过转换后，边角位置不变
- **Covariance:** if we have two transformed versions of the same image, features should be detected in corresponding locations **共变：** 如果我们有同一图像的两个转换版本，则应在相应位置检测到特征

In Harris Detector:

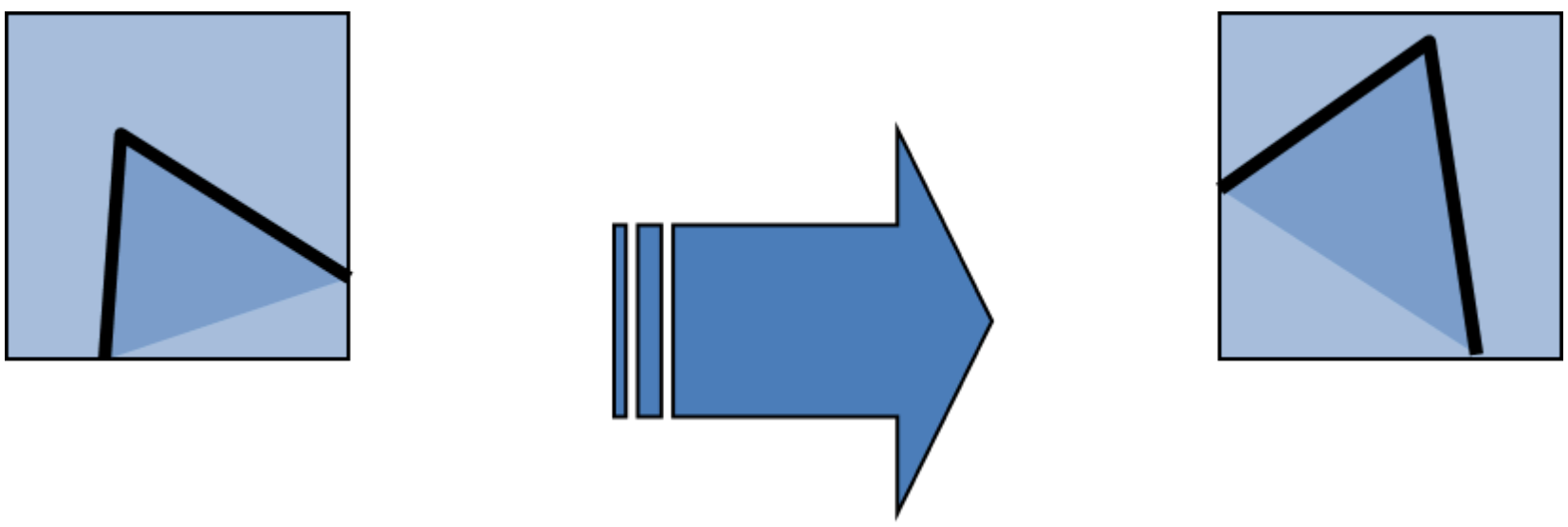
**Image translation 图片平移**



Derivatives and window function are shift-invariant. These feature values do not change after image shifting 导数和窗口函数是移位不变的。 这些特征值在图像移动后不会发生变化

The corner location is covariant with respect to translation. If the image is translated by a certain distance, the corner points will be translated by the same distance accordingly 角的位置是随平移而变化的。 如果图像平移了一定距离，角点也会相应平移相同的距离

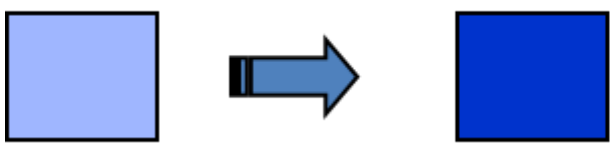
**Image rotation** 图像旋转



And corner location is covariant with respect to rotation 而角的位置与旋转是共变的

**Affine intensity change(brightness) 映射强度变化（亮度）**

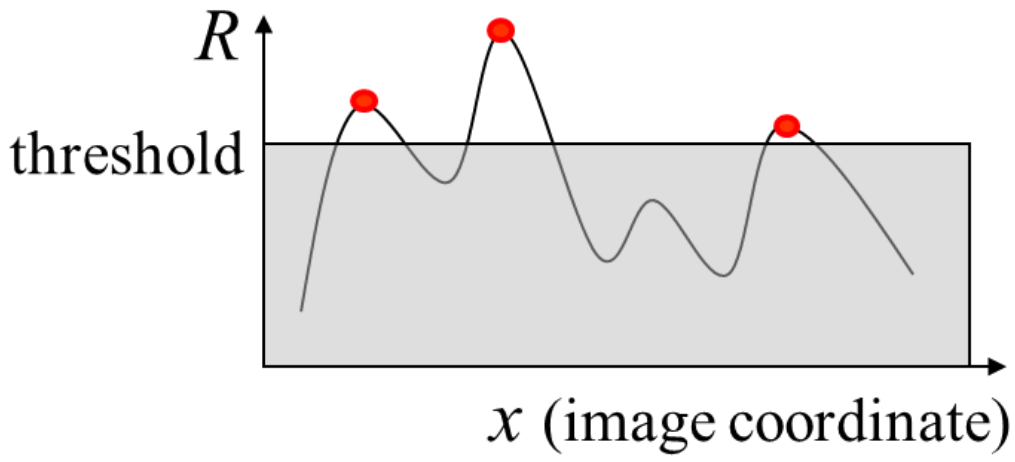
The brightness of an image can be transformed linearly by the following formula 图像亮度的线性变换公式如下

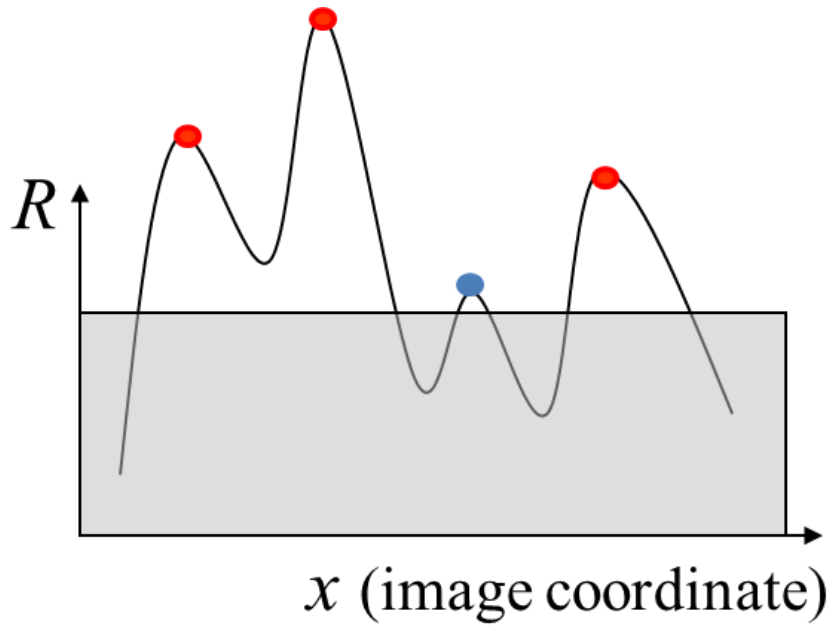


$$I \rightarrow a\,I + b$$

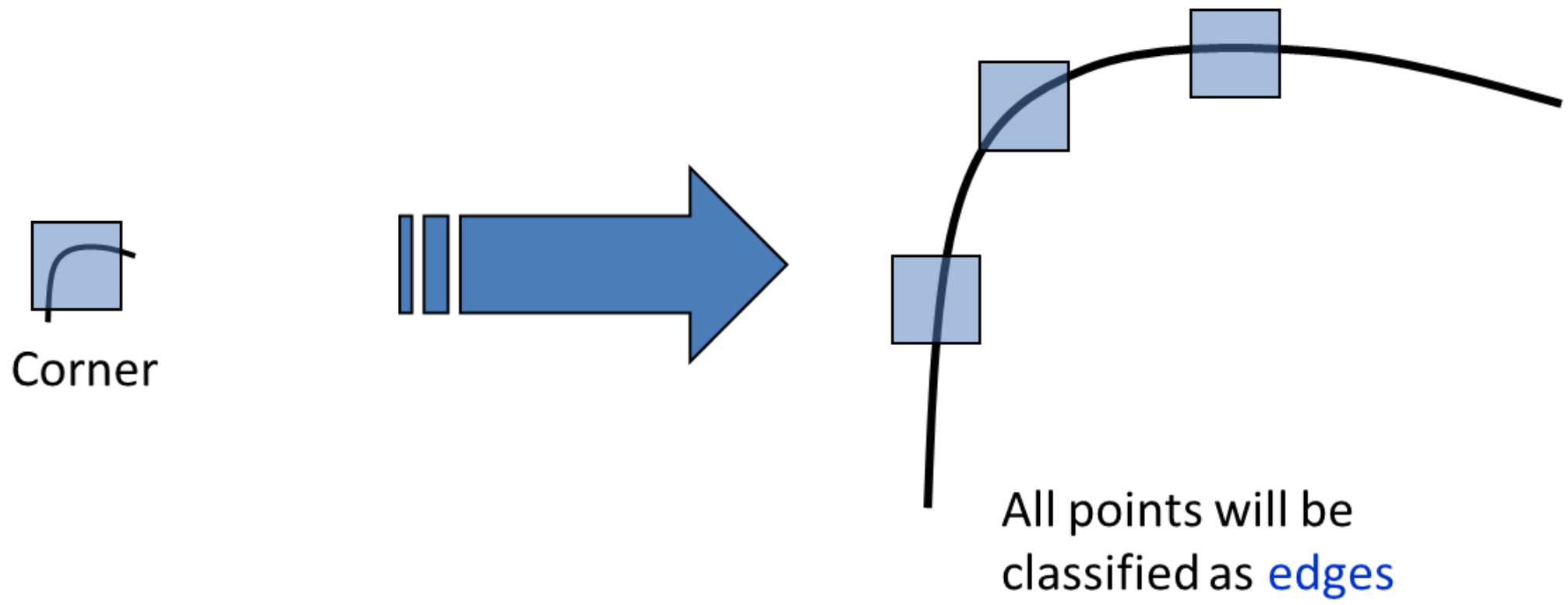
Only derivatives are used, so it's **invariant** to **intensity shift**: $I \rightarrow I + b$, but it's **partially invariant** to **intensity scaling**: $I \rightarrow aI$

Whether it's invariant or not depends on threshold:




**Scaling 缩放**

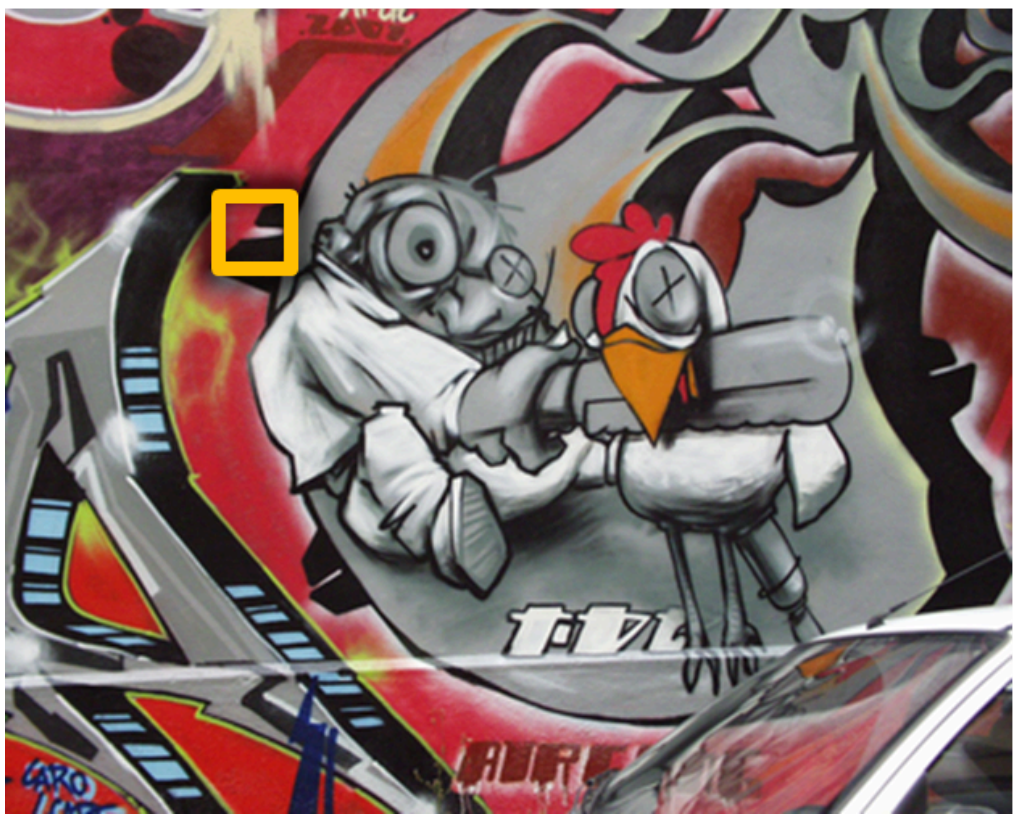


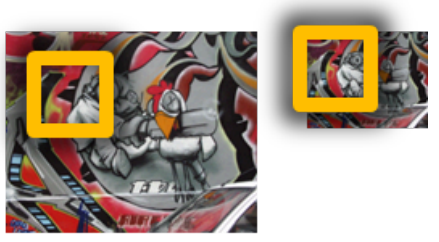

So it's *not invariant* to scaling 因此，*它不随缩放*而变化

## “Blob” detector "Blob "探测器

Instead of computing $f$ for larger and larger windows, we can implement using a fixed window size with a Gaussian pyramid

无需计算$f$我们可以使用固定窗口大小的高斯金字塔来实现

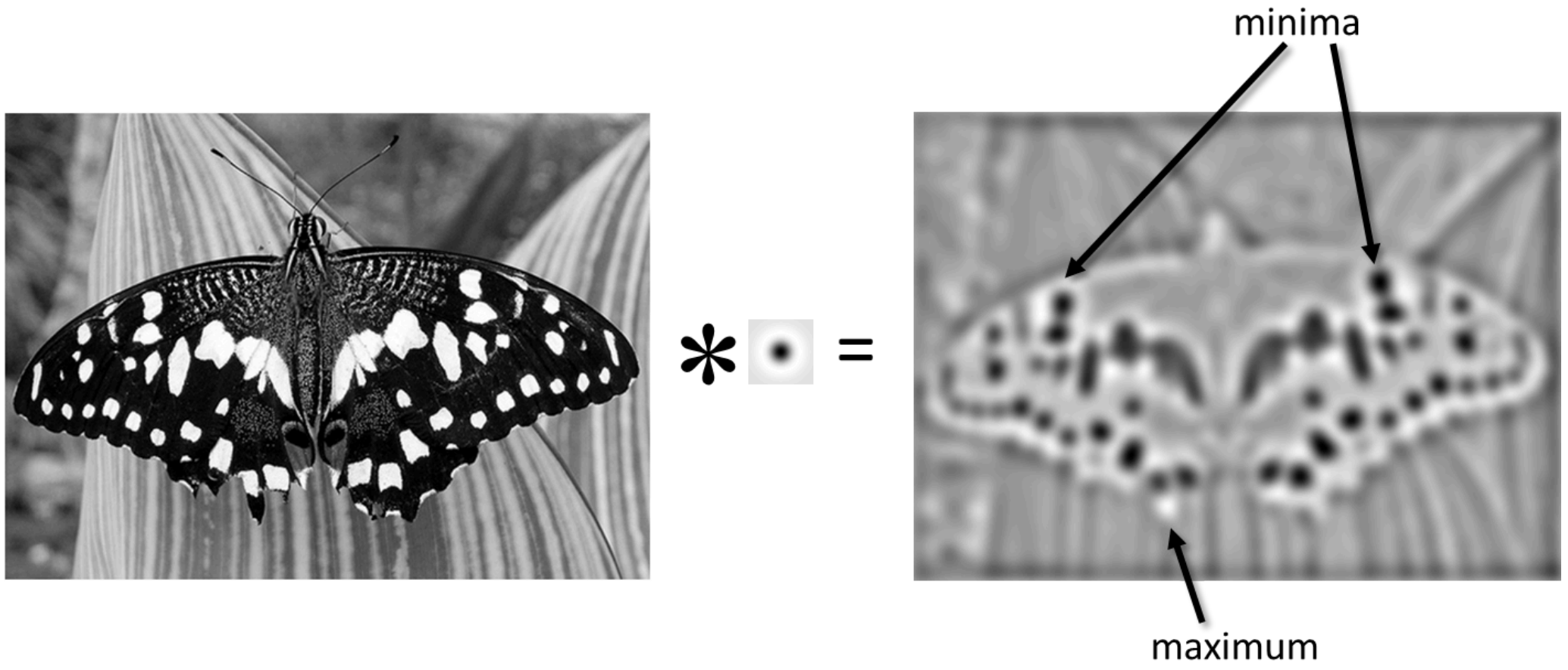




“blob” refers to a connected area of an image whose brightness or color is significantly different from that of the surrounding area, i.e., it appears as a "blob" or "speckle" in a two-dimensional image. "斑点 "是指图像中亮度或颜色与周围区域明显不同的连接区域，即在二维图像中显示为 "斑点 "或 "斑点"。

distinct from harris corner detector 有别于哈里斯角探测器

Laplacian of Gaussian (LoG) is able to do blob detection job 高斯的拉普拉斯（LoG）能够完成 Blob 检测工作

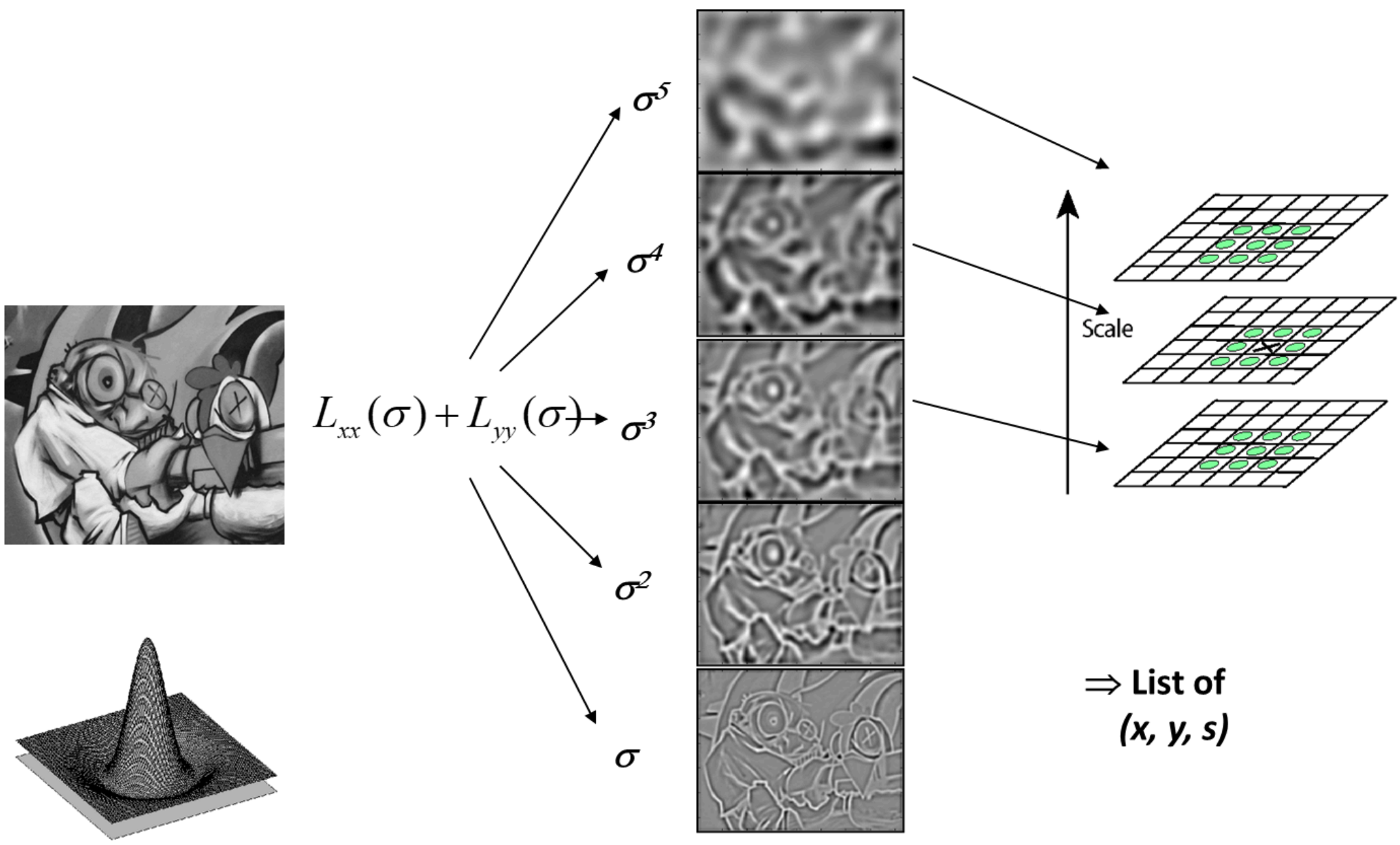



After LoG, Local maxima (maxima) and minima (minima) are labeled LoG 后，局部最大值（最大值）和最小值（最小值）被标注出来

maxima usually correspond to bright areas in the image or the centers of objects 最大值通常与图像中的明亮区域或物体中心相对应

minima usually correspond to dark areas in the image or the centers of objects 最小值通常与图像中的暗区或物体中心相对应

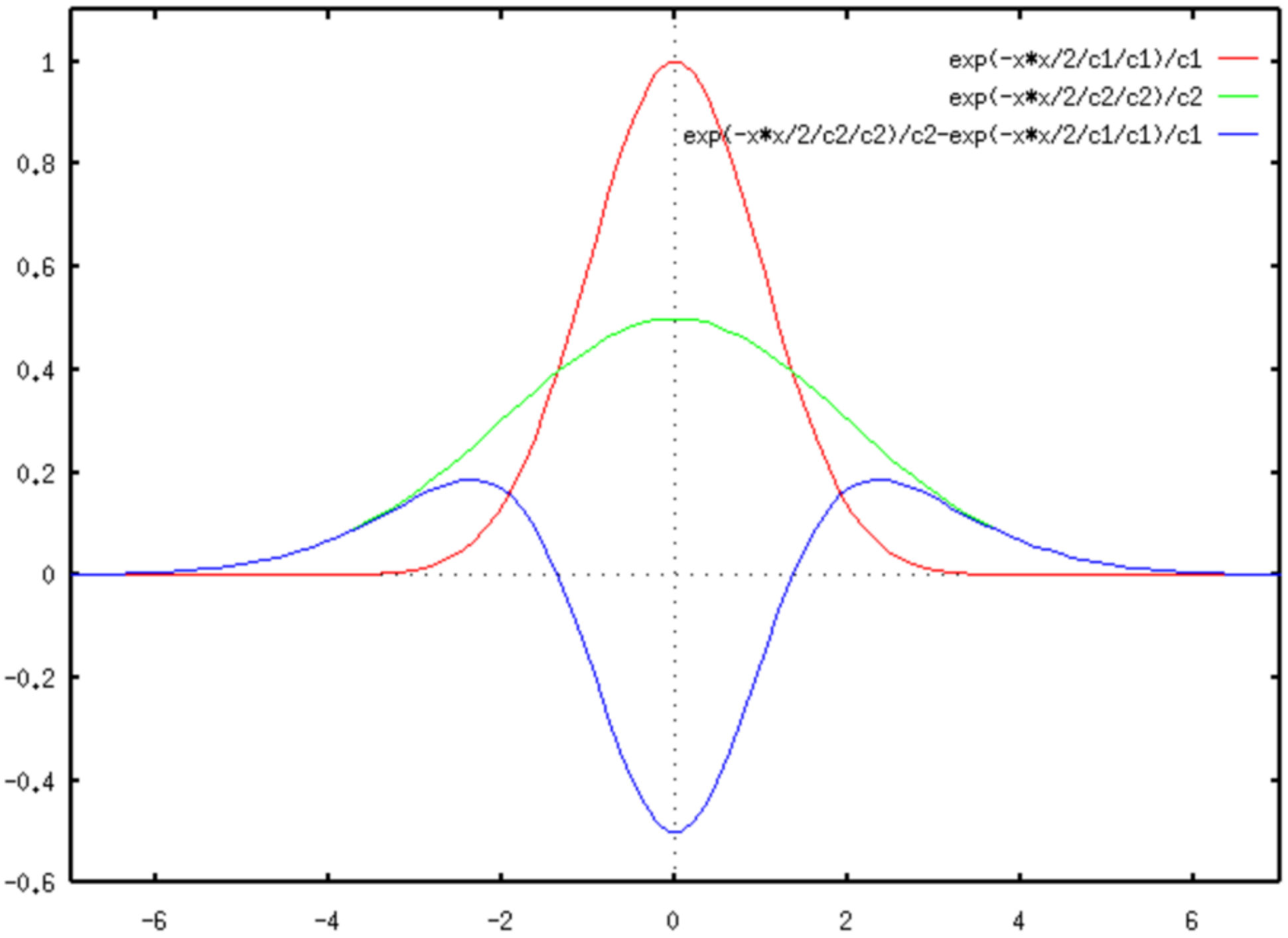
**Scale-space blob detector** 缩放空间斑点探测器




1. Input
2. Gaussian pyramid: subsampling
3. Apply LoG for every layer, detect corner and edge
4. Find maxima and minima
5. record blob feature and generate a list

# Scale Invariant Feature Transform(SIFT) 尺度不变特征变换（SIFT）

## LoG-DoG



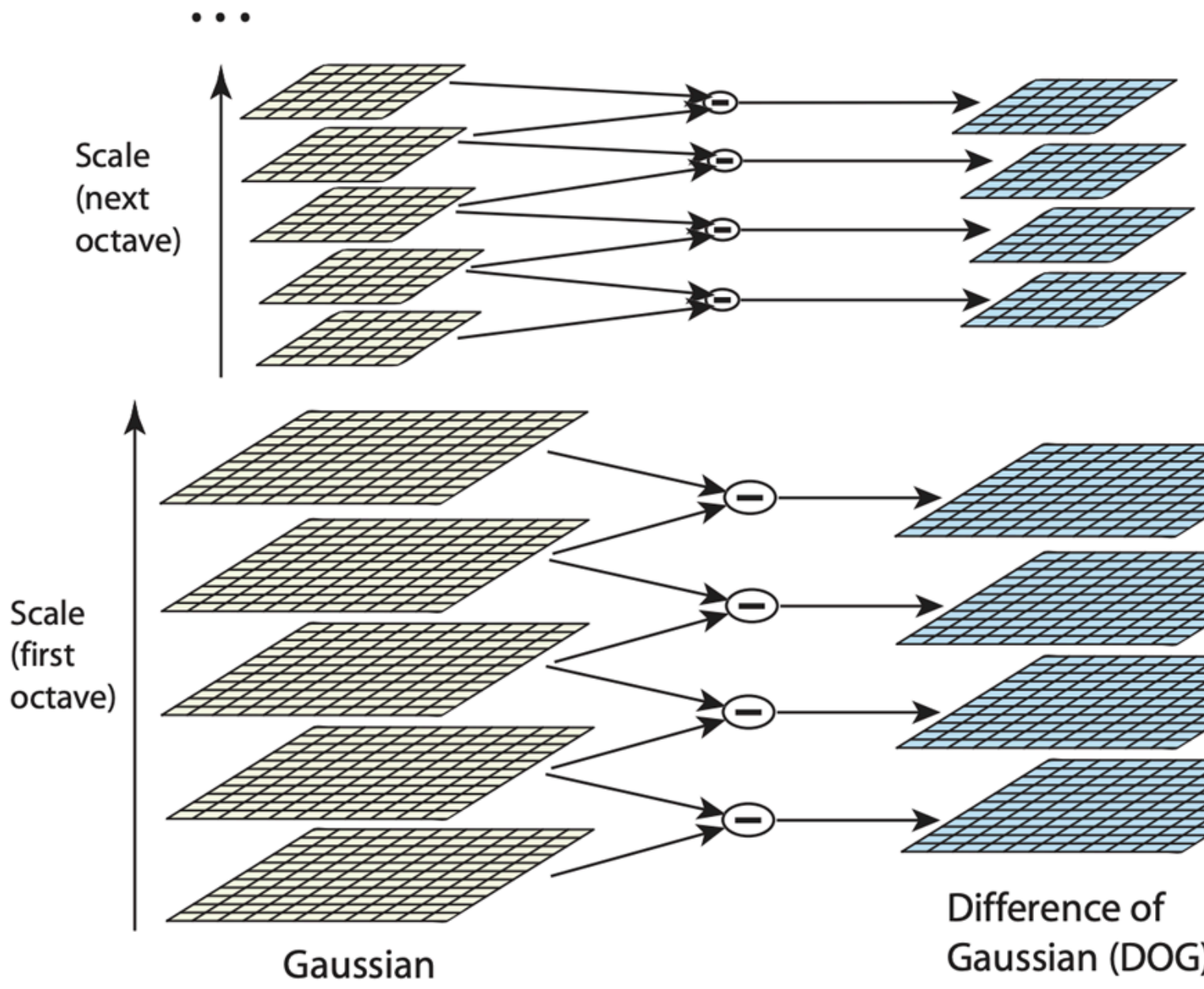

LoG can be approximated by a difference of two Gaussians (DoG) at different scales LoG 可以用不同尺度的两个高斯差（DoG）来近似表示

## SIFT process

SIFT contains 4 main steps:

1. Scale space peak selection: Finding potential feature point locations in space at different scales 比例空间峰值选择： 在不同尺度空间中寻找潜在特征点位置
2. Key point localization: Further pinpoint the location of these feature points 关键点定位： 进一步确定这些特征点的位置
3. Orientation assignment :Assigning orientation to the key points 方位分配 :为关键点分配方位
4. Key point descriptor: Describing the key point as a high dimensional vector (128) 要点描述： 将关键点描述为高维向量 (128)

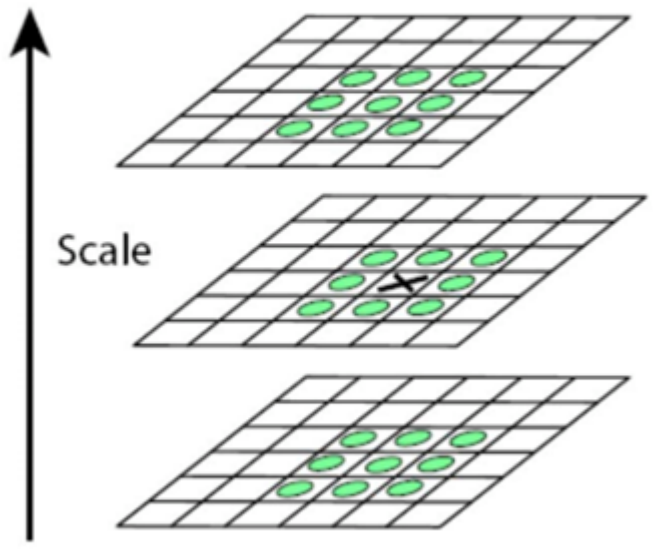
# Building the Scale Space 建立尺度空间




each scale corresponds to a Gaussian Pyramid consisting of multiple layers 每个刻度对应一个由多层组成的高斯金字塔

For neighboring scale images within each octave, calculate the difference between them, use difference to detect potential feature points. 对于每个八度内的相邻比例图像，计算它们之间的差值，利用差值检测潜在的特征点。

**Peak Detection** 峰值探测




Take a central pixel to compare with 26 neighboring pixel, if the value is greater or less than the value of all 26 neighboring pixels, the pixel is selected as the peak point. 取一个中心像素点与 26 个相邻像素点进行比较，如果其值大于或小于所有 26 个相邻像素点的值，则选取该像素点作为峰值点。

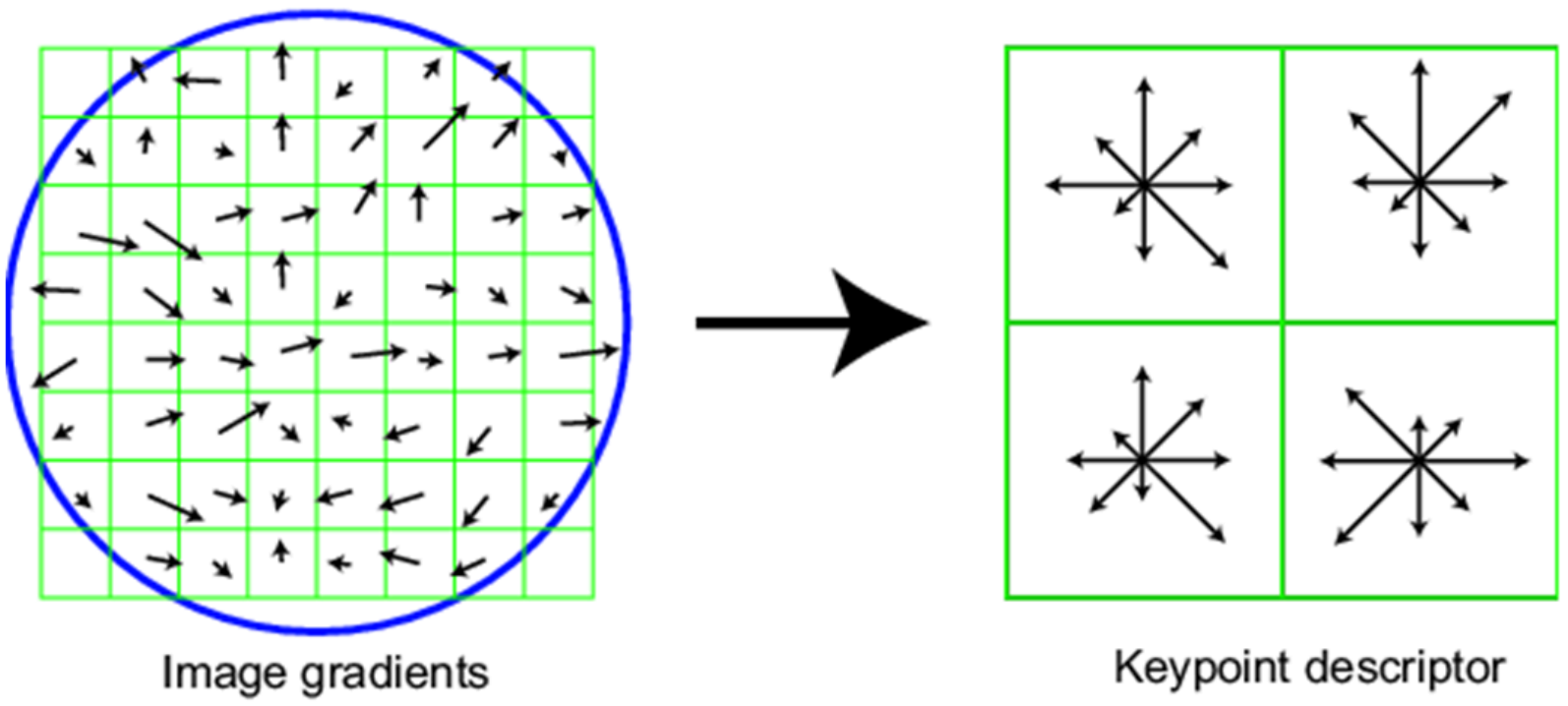
## Assignment of the Orientation 方向定向

- An orientation histogram is formed from the gradient orientations of sample points within a region around the keypoint. 方位直方图由关键点周围区域内样本点的梯度方位组成。
- The orientation histogram has 36 bins covering the 360 degree range of orientations. 方位直方图有 36 个分区，涵盖 360 度的方位范围。
- The samples added to the histogram is weighted by the **gradient magnitude**. 添加到直方图中的样本按梯度大小加权。
- The dominate direction is the **peak** in the histogram. 主导方向是直方图中的峰值。

## Keypoint Descriptor 关键点描述符

Full version

- Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
- Compute an orientation histogram (8 bin) for each cell (relative orientation and magnitude)
- 16 cells * 8 orientations = 128 dimensional descriptor




1. Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below) 将 16x16 的窗口划分为 4x4 的单元格（如下图所示为 2x2 的情况）
2. Compute an orientation histogram (8 bin) for each cell (relative orientation and magnitude) 计算每个单元格的方向直方图（8 个分区）（相对方向和幅度）
3. Weighting each directional interval according to the gradient magnitude 根据梯度大小对每个方向区间加权

•16 cells * 8 orientations = 128 dimensional descriptor

notice: SIFT is invariant to 2D rotation, translation and scaling, but it's not invariant to 3d rotation

注： SIFT 对二维旋转、平移和缩放不变，但对三维旋转变换

## Feature distance 特征距离

$I_1$

$I_2$

Simple approach: L2 distance $||f_1 - f_2||$

–can give good scores to ambiguous (incorrect) matches

a better way: $ratiodistance = ||f1 - f2||/||f1 - f2'||$

•f2 is best SSD match to f1 in I2

•f2’ is 2nd best SSD match to f1 in I2

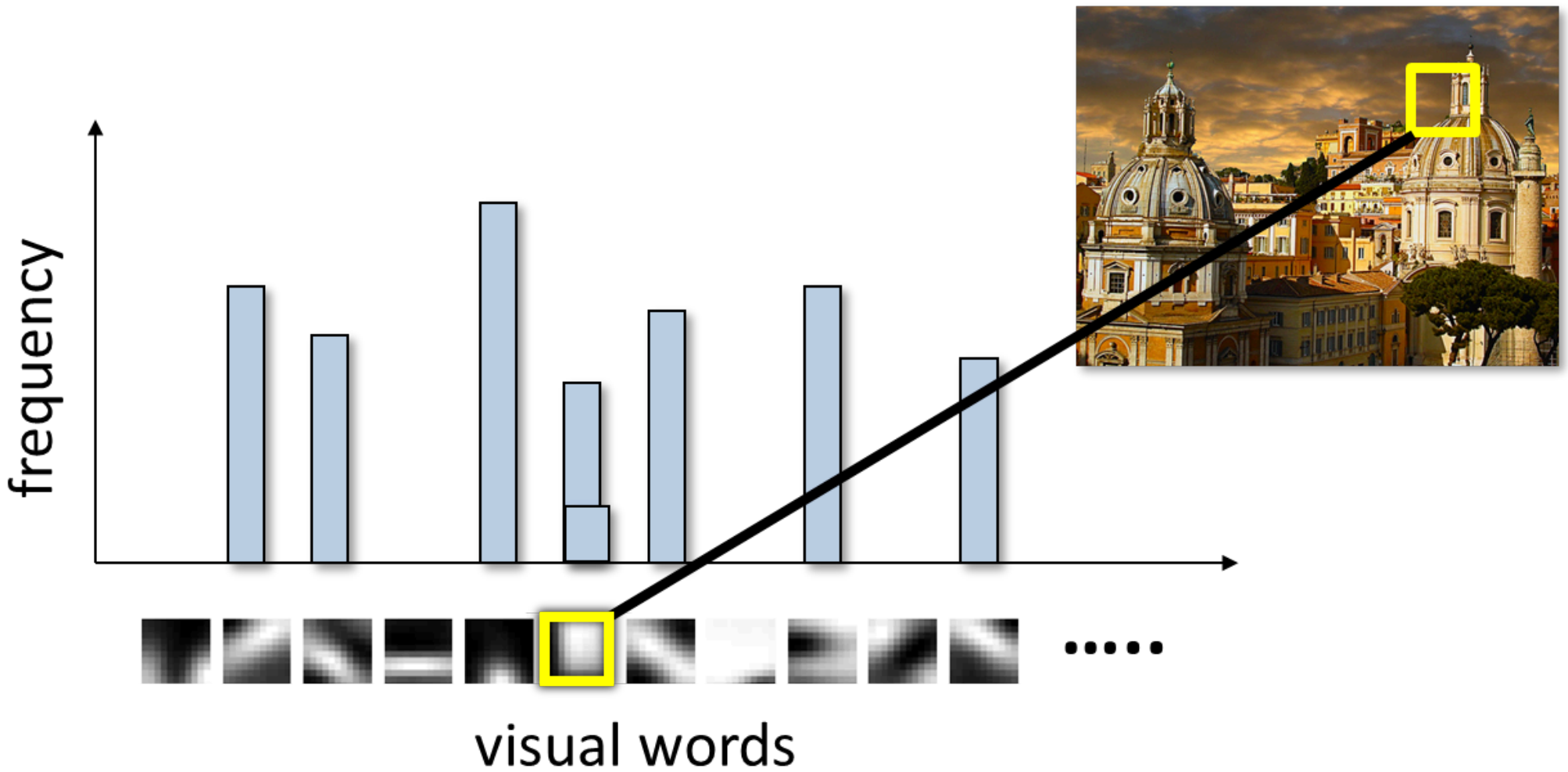
•gives large values for ambiguous matches

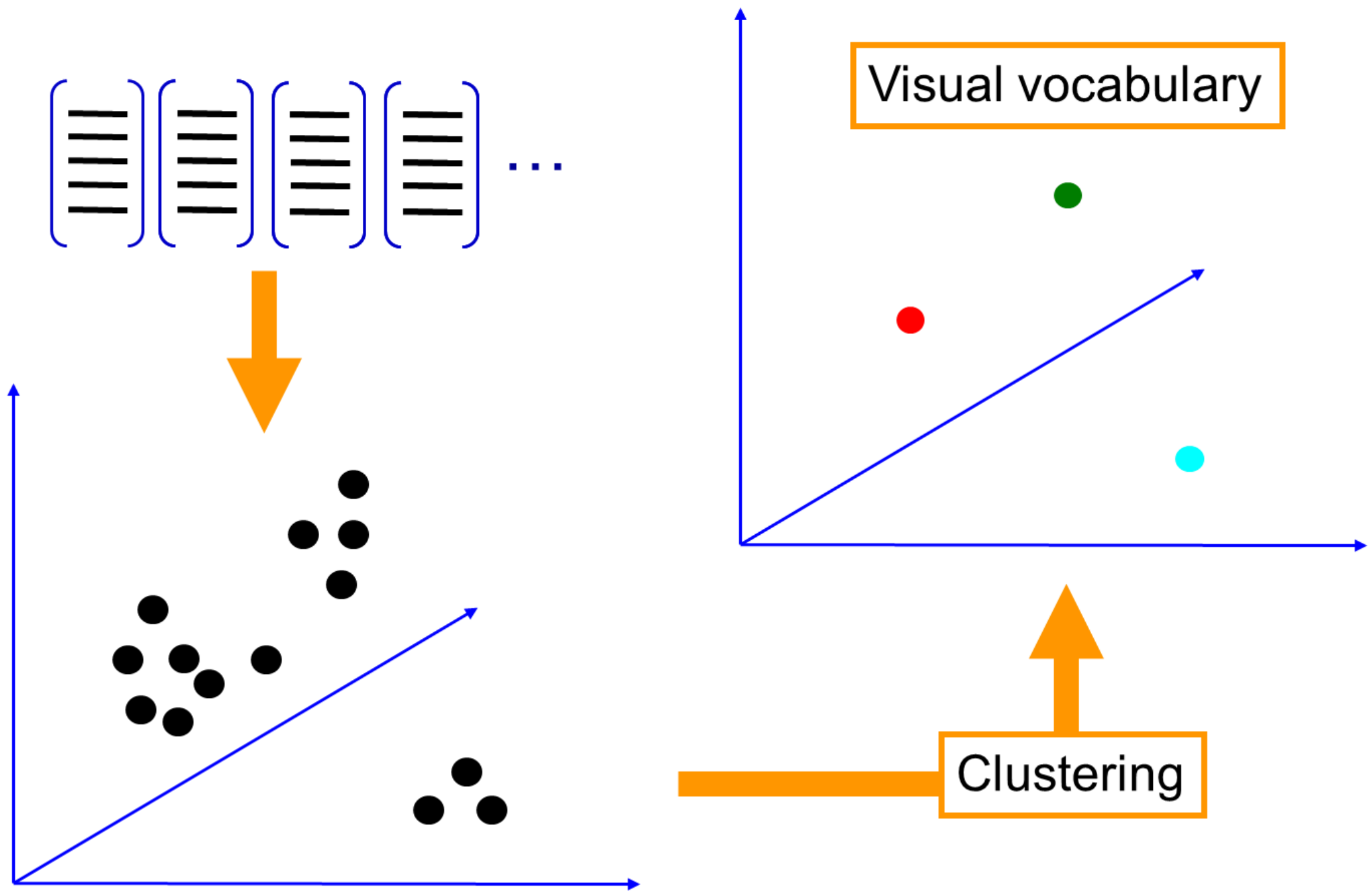
# Bag of Words 字袋

## Visual vocabulary 视觉词汇

Image can be represented as a histogram of a set of visual words, similar to word frequency statistics in a text document

图像可表示为一组视觉词的直方图，类似于文本文档中的词频统计




By using clustering, efficiently quantize local features in an image into a set of visual words for efficient image description and retrieval

通过使用聚类技术，有效地将图像中的局部特征量化为一组视觉词，以便高效地进行图像描述和检索

**K-means clustering K均值聚类**

$$D(X,M)=\sum_{\text{cluster } k}\ \sum_{\substack{\text{point } i \text{ in}\\ \text{cluster } k}}(x_i-m_k)^2$$

1. First Iteration 第一次迭代
   - Calculate the distance from each data point to the two cluster centers and assign it to the nearest cluster center. 计算每个数据点到两个聚类中心的距离，并将其分配到最近的聚类中心。
   - Update the cluster centers to be the average of all points in their respective clusters. 更新聚类中心，使其成为各自聚类中所有点的平均值。
2. Second Iteration 第二次迭代
   - The distance from each data point to the new cluster center is again calculated and redistributed. 再次计算每个数据点到新聚类中心的距离并重新分配。
   - The updated cluster center is the average of all points in the respective cluster. 更新后的聚类中心是相应聚类中所有点的平均值。
3. Repeat until the center doesn't change or reach the maximum iteration number. 重复上述步骤，直到中心不变或达到最大迭代次数。

A simple but effective unsupervised learning method for grouping data points into distinct clusters. By minimizing the sum of squares of the distances from the points to the center of the clusters, K-means is able to efficiently discover structures in the data. 一种简单而有效的无监督学习方法，可将数据点归入不同的群组。 通过最小化各点到聚类中心的距离平方和，K-means 能够有效地发现数据中的结构。

## TF-IDF

Instead of computing a regular histogram distance, we'll weight each word by it's **inverse document frequency**我们不计算常规的直方图距离，而是根据每个词的**反文档频率**对其进行加权：

IDF(j)=

$$\log\ \frac{\text{number of documents}}{\text{number of documents in which } j \text{ appears}}$$

To compute the value of bin *j* in image *I*: 计算图像 *I* 中 bin *j* 的值：

$$\textit{term frequency of } j \text{ in } I\ \ \mathbf{X}\ \ \textit{inverse document frequency of } j$$

which is actually is: $TF-IDF(j,I)=TF(j,I)\times IDF(j)$

By combining word frequency (TF) with inverse document frequency (IDF), the importance of a visual word in an image can be more accurately measured. Visual words that appear with high frequency in a few images are given higher weights, while visual words that appear with high frequency in most images are given lower weights.

通过将词频（TF）与反向文档频率（IDF）相结合，可以更准确地衡量图像中视觉词的重要性。 在少数图像中出现频率较高的视觉词会被赋予较高的权重，而在大多数图像中出现频率较高的视觉词则会被赋予较低的权重。

### Inverted file 反转文件

Able to effectively manage and retrieve information from large-scale image datasets 能够有效管理和检索大规模图像数据集的信息

If each image contains about 1000 features, and the total number of visual words is 1000000, then each histogram is extremely sparse (mostly zeros) 如果每幅图像包含约 1000 个特征，而视觉词的总数为 1000000，那么每个直方图都非常稀疏（大部分为零）。

we can try to map from words to documents 我们可以尝试将单词映射到文件

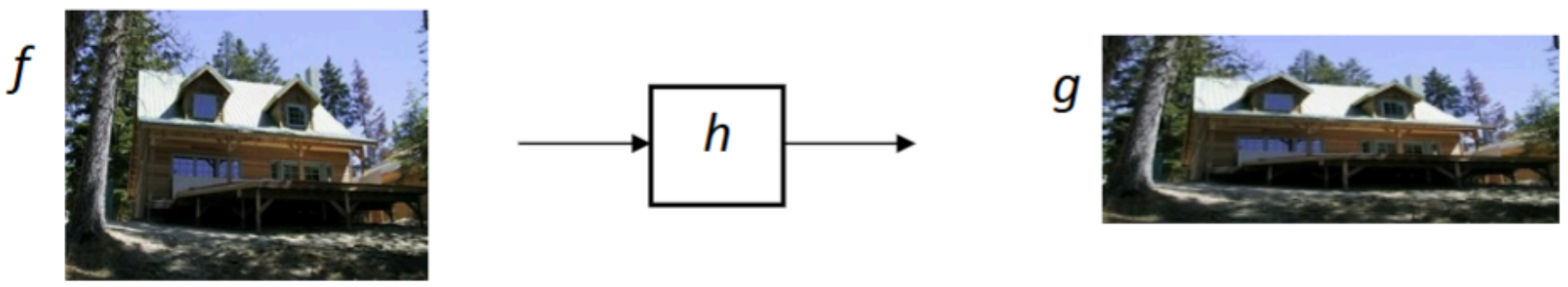
```
"a":      {2}
"banana": {2}
"is":     {0, 1, 2}
"it":     {0, 1, 2}
"what":   {0, 1}
```

## Transformation and Alignment 转型和调整

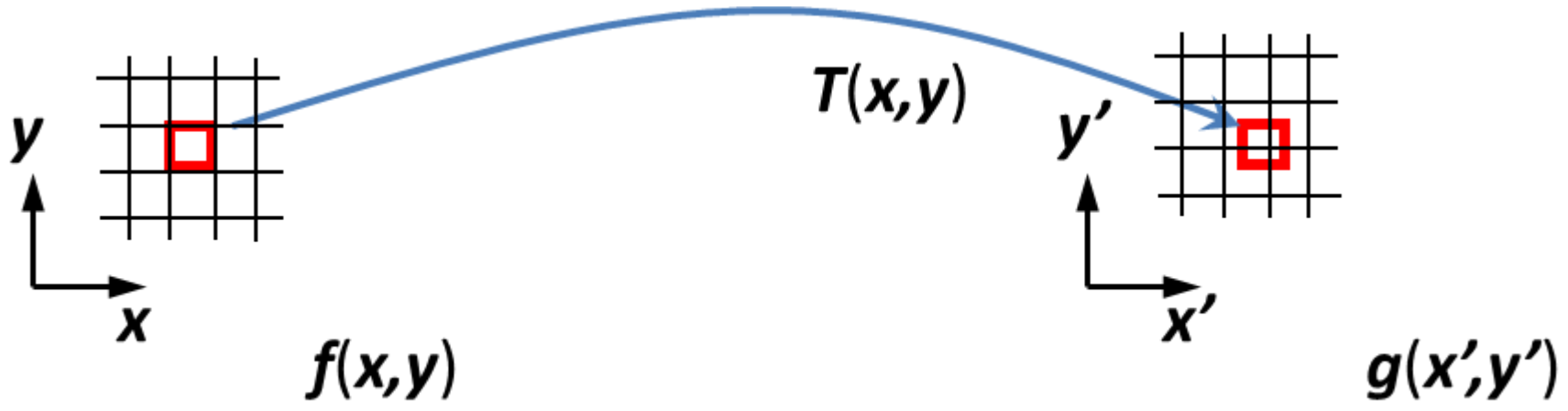
### Forward Warping 前向翘曲

Consider changing position of each point in an image 考虑改变图像中每个点的位置

- $g(x) = f(h(x))$




When $f(x, y)$ has a new position $(x', y')$ by mapping function T in a new image:



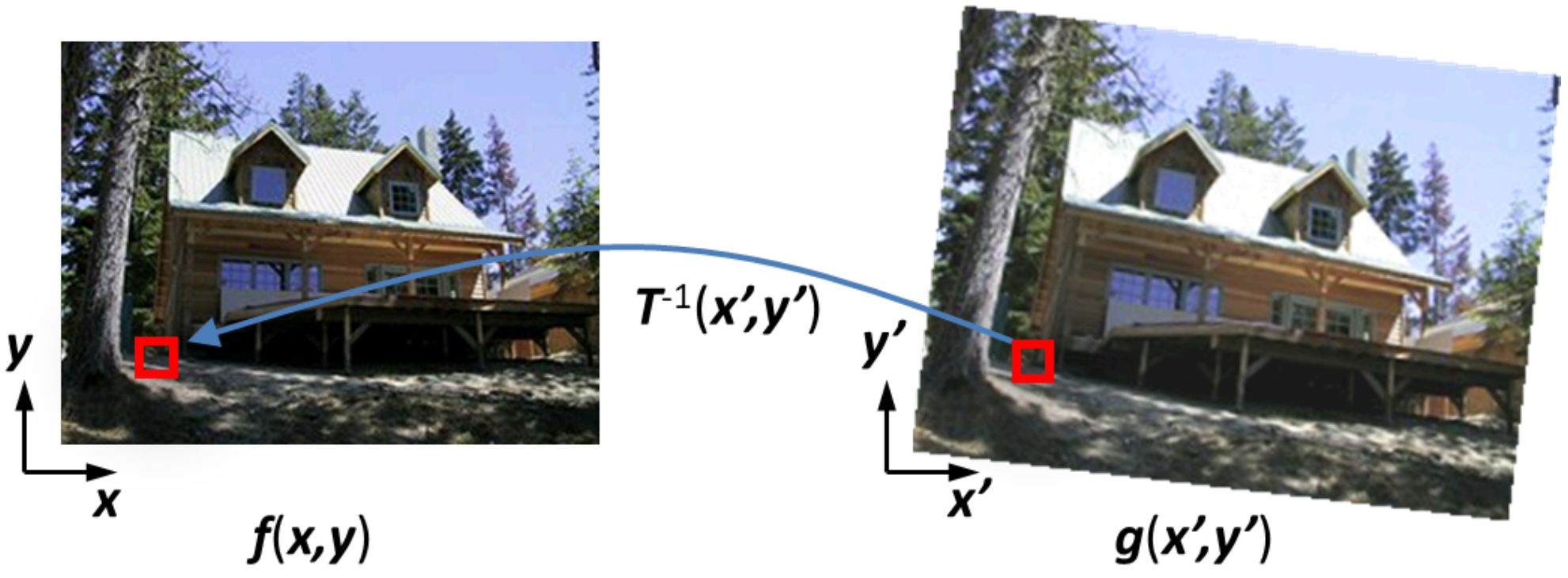

What if pixel lands “between” two pixels? 如果像素落在两个像素之间怎么办?

Solution: add “contribution” to several pixels, normalize later 解决方法：为几个像素添加 "贡献值"，然后进行归一化处理

But this method can still result in **holes** 但这种方法仍可能导致**孔**

# Inverse Warping 逆扭曲

How about giving $g(x', y')$ , to get its corresponding origin location $(x, y) = T^{-1}(x', y')$ in $f(x, y)$ ?




We need to find the inverse of mapping $T^{-1}$

And if pixel comes from “between” two pixels, there might still be holes

# All 2D Linear Transformations 所有二维线性变换

Linear transformations includes: 线性变换包括

- Scale 缩放
- Rotation 选择
- Shear 剪切
- Mirror 镜像

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} e & f \\ g & h \end{bmatrix} \begin{bmatrix} i & j \\ k & l \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

# Homogeneous coordinates

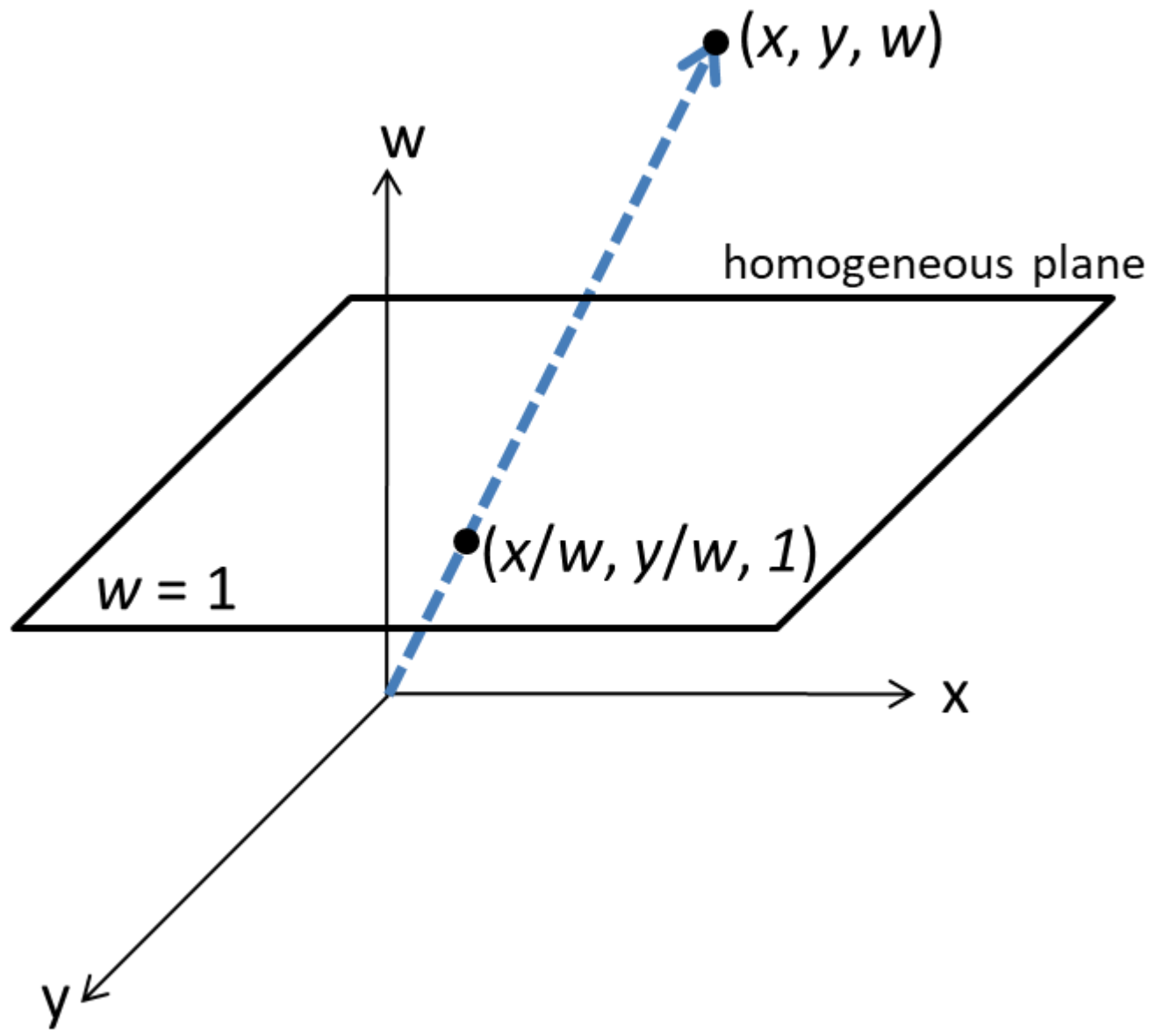
齐次坐标

Homogeneous coordinates allow translation, rotation, scaling, and other linear transformations to be represented in a uniform way, i.e., through matrix multiplication 齐次坐标允许以统一的方式（即通过矩阵乘法）表示平移、旋转、缩放和其他线性变换

By using Homogeneous coordinates, we can represent transformations into 3×3 matrixes 通过使用均质坐标，我们可以将变换表示为 3×3 矩阵

We need to add one more coordinate $w$ to create homogeneous coordinate: 我们需要增加一个坐标$w$来创建均质坐标：

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$




If we want to get normal coordinate back: 如果我们想恢复正常的坐标

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w)$$

## Affine Transformations 仿射变换

Affine Transformation is set of linear transformations and translations 仿射变换是线性变换和平移的集合(平移)

This transformations can be represented by matrixes: 这种变换可以用矩阵来表示：

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Translate

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Scale

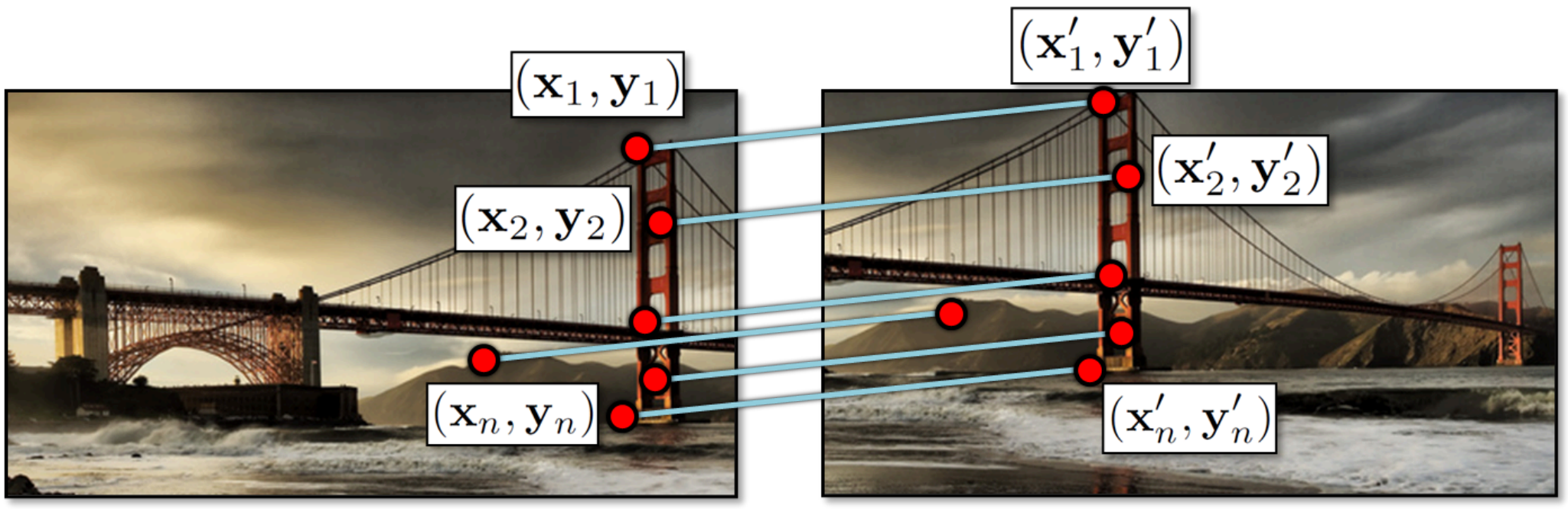
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

2D *in-plane* rotation

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & sh_x & 0 \\ sh_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Shear

Consider a **translation** case:




For each i, it's displacement(位移):

$$(\mathbf{x}'_i - \mathbf{x}_i, \mathbf{y}'_i - \mathbf{y}_i)$$

The average displacement is: 平均位移

$$\text{Mean displacement} = \left( \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}'_i - \mathbf{x}_i, \frac{1}{n} \sum_{i=1}^{n} \mathbf{y}'_i - \mathbf{y}_i \right)$$

So the displacement estimate is: 位移估计值

$$I(\mathbf{x}_t, \mathbf{y}_t) = = \left( \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}'_i - \mathbf{x}_i, \frac{1}{n} \sum_{i=1}^{n} \mathbf{y}'_i - \mathbf{y}_i \right)$$

Since there are a lot of matching points, equations are more than unknows: 由于匹配点很多，方程比未知数多：

It's "overdetermined" system of equations, we need to find the least squares solution 这是一个 "超定 "方程组，我们需要找到最小二乘法解

For each point, the estimate displaced position is: 每个点的估计位移位置为

$$\mathbf{x}_i + \mathbf{x_t} = \mathbf{x}'_i$$

$$\mathbf{y}_i + \mathbf{y_t} = \mathbf{y}'_i$$

The residual(残差) is :

$$r_{\mathbf{x}_i}(\mathbf{x}_t) = (\mathbf{x}_i + \mathbf{x}_t) - \mathbf{x}'_i$$

$$r_{\mathbf{y}_i}(\mathbf{y}_t) = (\mathbf{y}_i + \mathbf{y}_t) - \mathbf{y}'_i$$

We need to **minimize** sum of squared residuals 我们需要**最小化**残差平方和

$$C(\mathbf{x}_t, \mathbf{y}_t) = \sum_{i=1}^{n} \left( r_{\mathbf{x}_i}(\mathbf{x}_t)^2 + r_{\mathbf{y}_i}(\mathbf{y}_t)^2 \right)$$

It's “Least squares” solution. For translations, is equal to mean (average) displacement 这是 "最小平方 "解法。 对于平移，等于平均（平均）位移

# Least squares formulation 最小二乘法公式

This problem can be written as matrix equation: 这个问题可以写成矩阵方程：

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ & \vdots \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_t \\ y_t \end{bmatrix} \approx \begin{bmatrix} x'_1 - x_1 \\ y'_1 - y_1 \\ x'_2 - x_2 \\ y'_2 - y_2 \\ \vdots \\ x'_n - x_n \\ y'_n - y_n \end{bmatrix}$$

$$\underset{2n \times 2}{\mathbf{A}} \quad \underset{2 \times 1}{\mathbf{t}} \quad \underset{2n \times 1}{\mathbf{b}}$$

we need to find $t$ that minimizes:

$$||\mathbf{At} - \mathbf{b}||^2$$

To solve it, we build:

$$\mathbf{A}^{\mathrm{T}}\mathbf{At} = \mathbf{A}^{\mathrm{T}}\mathbf{b}$$

The t can be calculated through:

$$\mathbf{t} = \left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\right)^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{b}$$

## RANSAC

A robust statistical method for **estimating** model parameters in a dataset, with particular application to the presence of a large number of **outliers** 在数据集中**估计**模型参数的稳健统计方法，特别适用于存在大量**异常值的情况**

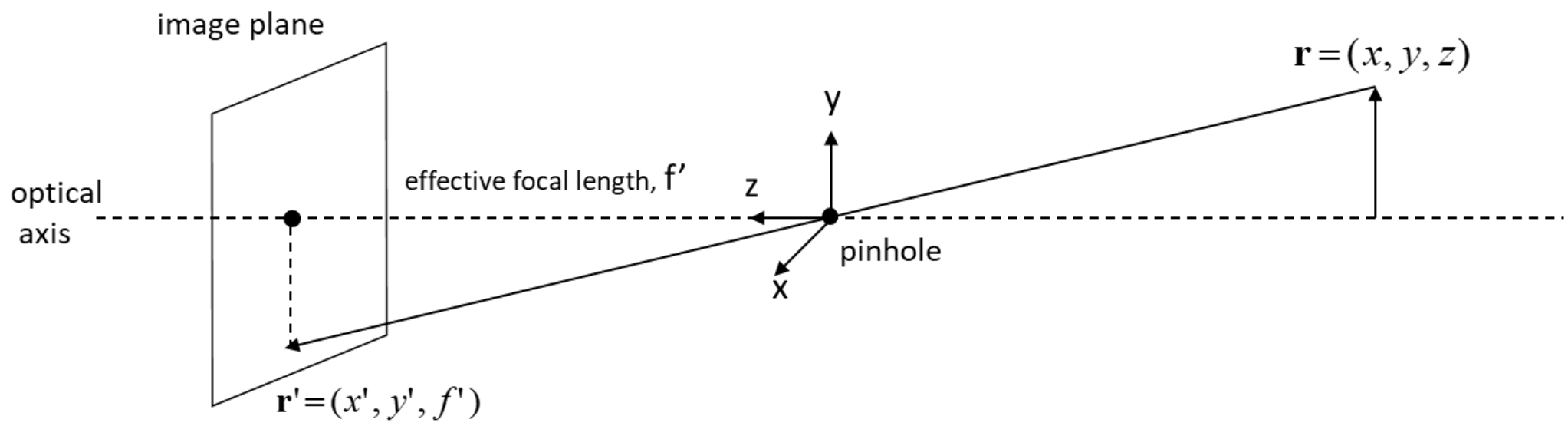
1. **Randomly** choose $s$ samples, Typically $s$ = minimum sample size that lets you fit a model **随机**选择 $s$ 样本，通常 $s$ = 可以拟合模型的最小样本量
2. Fit a model (e.g., line) to those samples 为这些样本拟合模型（如线性模型
3. Count the number of inliers that approximately fit the model 计算近似符合模型的离群值数量
4. Repeat *N* times(new samples, new model) 重复 *N* 次（新样本、新模型）
5. Choose the model that has the largest set of inliers 选择群内值最大的模型

the process is like voting for a candidate, inlier number is like vote 这个过程就像给候选人投票，群内值就像选票

# Cameras

## Pinhole camera 针孔摄像机

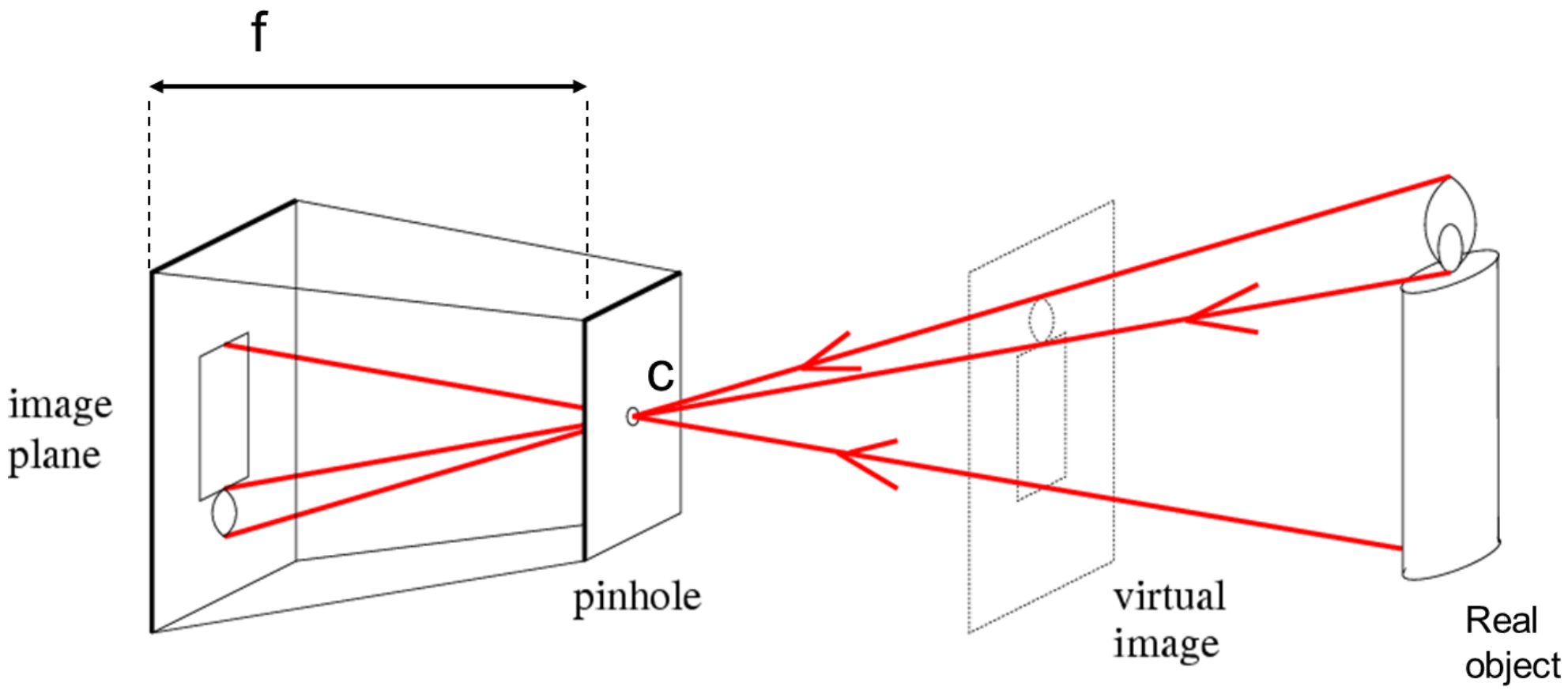
If a object is projected to screen through a pinpoint: 如果通过针尖将物体投射到屏幕上：




$$\frac{\mathbf{r}'}{f'} = \frac{\mathbf{r}}{z} \quad \Rightarrow \quad \frac{x'}{f'} = \frac{x}{z} \qquad \frac{y'}{f'} = \frac{y}{z}$$

The projected coordinate  can be calculate. The value depends on focal length: $f'$可以计算出投影坐标。 该值取决于焦距

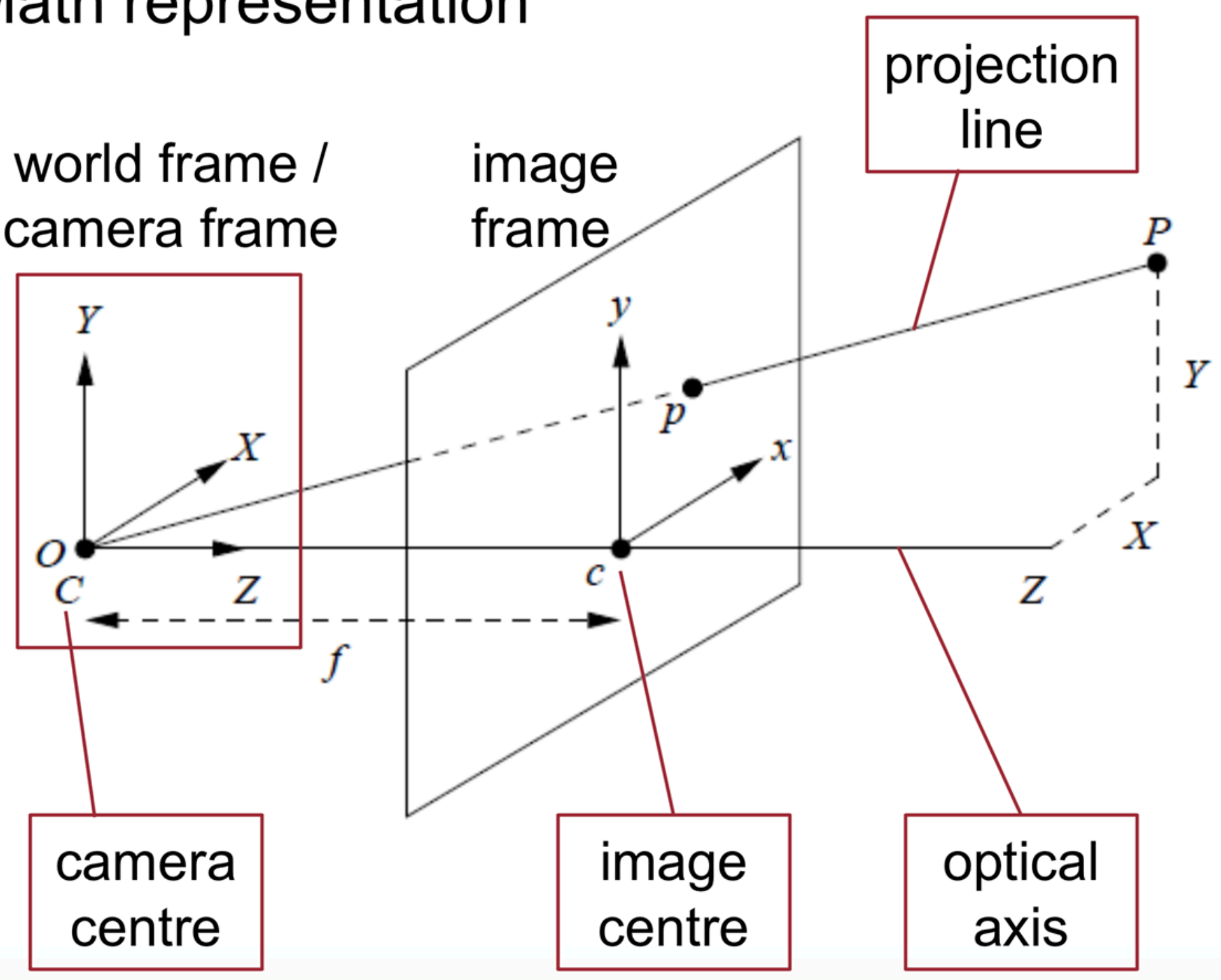
Pinhole camera model:




f: Focal length 焦距

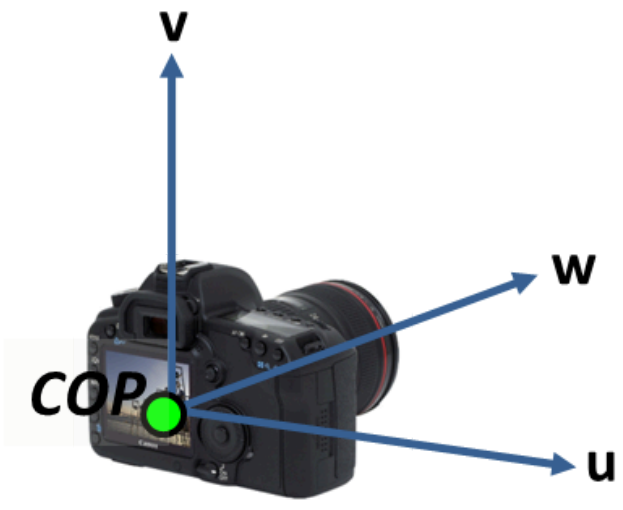
c: Optical center of the camera 摄像机的光学中心

## ⊙ Math representation




# Camera parameters 相机参数

How can we model the geometry of a camera? 如何建立摄像机的几何模型?





"The World"

We need to connect *World* coordinate system and *Camera* coordinate system 我们需要连接*世界*坐标系和*摄像机*坐标系

To project a point (*x*,*y*,*z*) in *world* coordinates into a camera 将*世界*坐标中的点（*x*,*y*,*z*）投射到摄像机中

we need to know: 我们需要知道

- Camera position (in world coordinates) 摄像机位置（世界坐标）
- Camera orientation (in world coordinates) 摄像机方向（世界坐标）
- The formation of image frame(camera *intrinsics*) 图像帧的形成（摄像机固有属性）

## Intrinsic Parameters 内在参数

Intrinsic Parameters describe properties of the camera itself, such as focal length, principal point position, etc., that are used to project points in the camera's coordinate system onto the image plane.(camera to image) 固有参数描述摄像机本身的属性，如焦距、主点位置等，用于将摄像机坐标系中的点投影到图像平面上。

Denote location of c (principle point) image plane as $c_x$ and $c_y$ 用 $c_x$ 和 $c_y$ 表示 c（原理点）图像平面的位置

principle point: Intersection between the camera optical axis and image plane 摄像机光轴与图像平面的交点

then:

$$P' = (x', y') = (f\frac{x}{z} + c_x, f\frac{y}{z} + c_y)$$

Further consider mapping relationship between **digit plan and image plan** 进一步考虑**数字计划和图像计划之间的映射关系**

In digit plan, expressed as in pixels. 在数字平面图中，用像素表示。

in image plan, represented in physical measurement (e.g., centimeter) 在图像平面中，用物理测量（如厘米）表示

The mapping can be denoted as $\frac{pixels}{cm}$

use two parameters, *k* and *l*, to describe the mapping. If k=l, then the camera has “square pixels”. 使用两个参数 *k* 和 *l* 来描述映射。 如果 k=l，则摄像机具有 "方形像素"。

Now the equation is:

$$P' = (x', y') = (fk\frac{x}{z} + c_x, fl\frac{y}{z} + c_y)$$
$$= (\alpha\frac{x}{z} + c_x, \beta\frac{y}{z} + c_y)$$

In matrix form:

$$P' = \begin{bmatrix} \alpha & 0 & c_x & 0 \\ 0 & \beta & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = MP$$

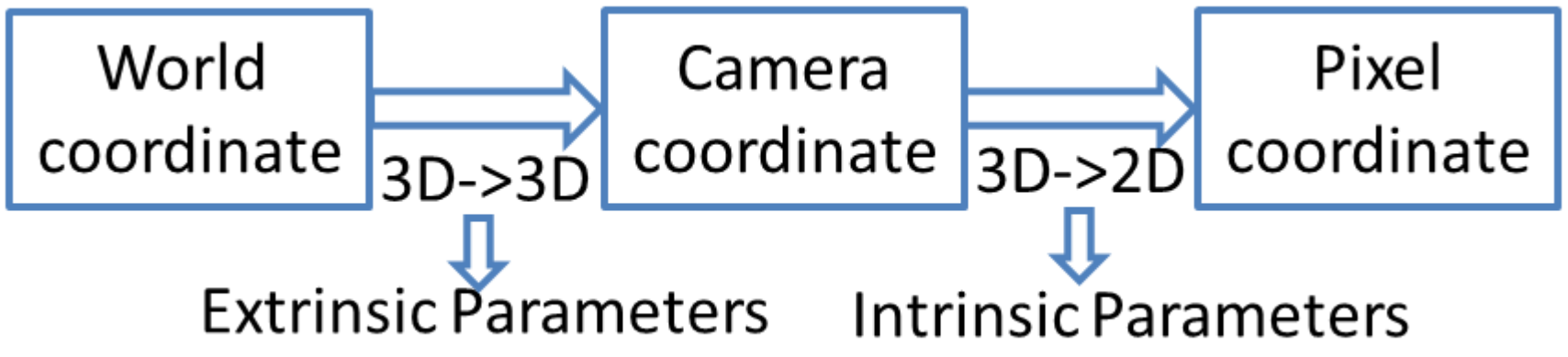
$$P' = MP = \begin{bmatrix} \alpha & 0 & c_x \\ 0 & \beta & c_y \\ 0 & 0 & 1 \end{bmatrix} [I \quad 0]P = K[I \quad 0]P$$

*K: Camera matrix (or calibration matrix)*

**Extrinsic Parameters 外在参数**

(world to camera)

We need to relate the points from world reference system to the camera reference system 我们需要将世界参照系中的点与摄像机参照系联系起来




Given a point in world reference system $P_w$, the camera coordinate is computed as 给定世界参照系中的一点$P_w$，摄像机坐标的计算公式为

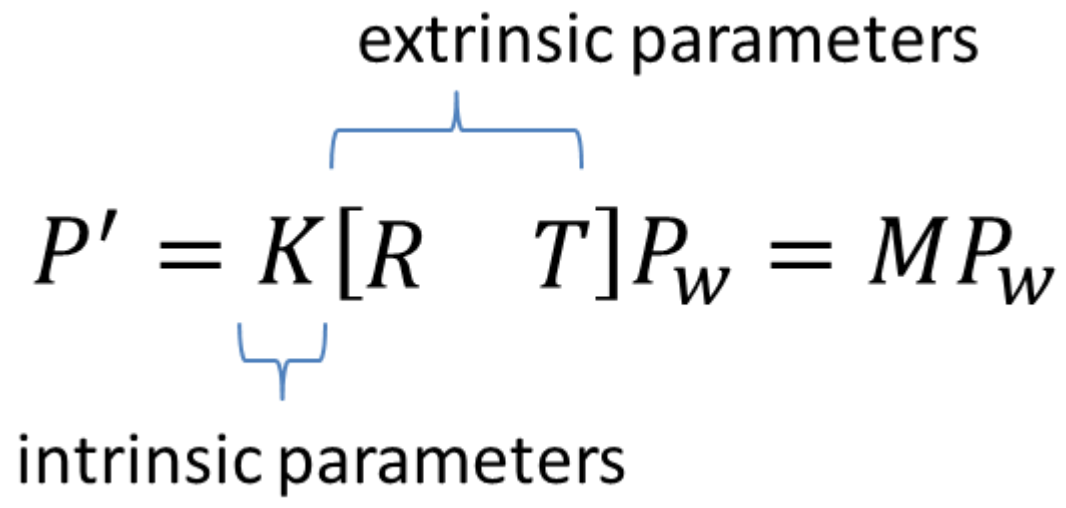
$$P = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} P_w$$

R: Rotation matrix R： 旋转矩阵

T: translation vector T：平移矢量

## Projection Matrix 投影矩阵

Combining equations above, we have: 综合上述方程，我们可以得出

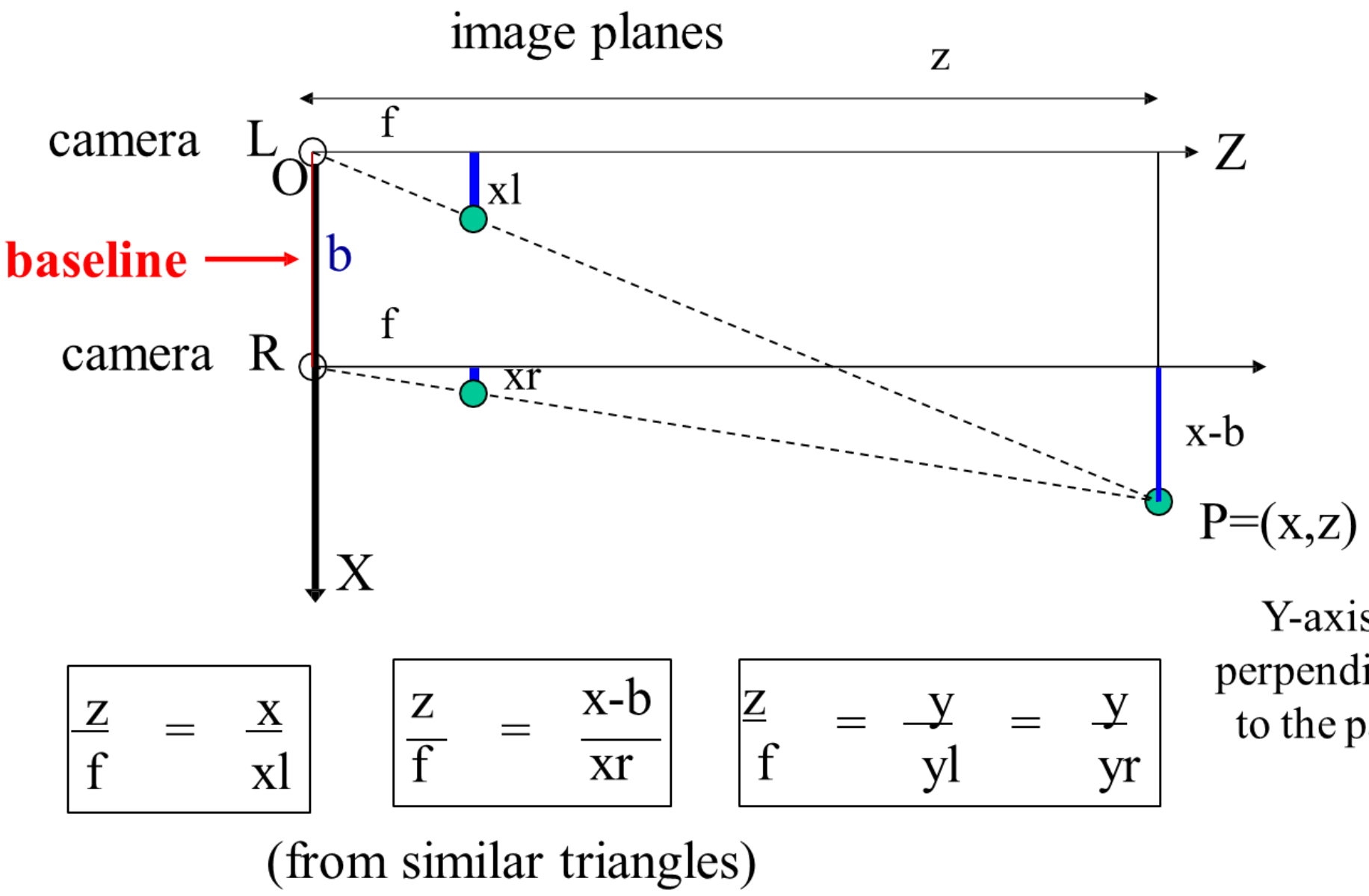



- *K* changes as the type of camera changes *K* 随着摄像机类型的变化而变化
- Extrinsic parameters are independent of camera 外在参数与摄像机无关

# Stereo Vision and Structure from Motion 立体视觉和运动结构

## Depth and Disparity 深度和视差

Acquiring Depth Information with Two Cameras 用两台相机获取深度信息




For stereo cameras with parallel optical axes, **focal length f**, **baseline b**, corresponding image **points (xl,yl) and (xr,yr)**, the **location** of the 3D point can be derived from previous slide's equations: 对于光轴平行、焦距为 f、基线为 b、对应图像点为 (xl,yl) 和 (xr,yr) 的立体相机，三维点的位置可以通过前面的幻灯片方程推导出来：

$$\textbf{Depth } z = f*b / (xl - xr) = f*b/d$$

$$x = xl*z/f \quad \text{or} \quad b + xr*z/f$$

$$y = yl*z/f \quad \text{or} \quad yr*z/f$$

z is depth, means the distance from object to the camera z 是深度，指物体到摄像机的距离

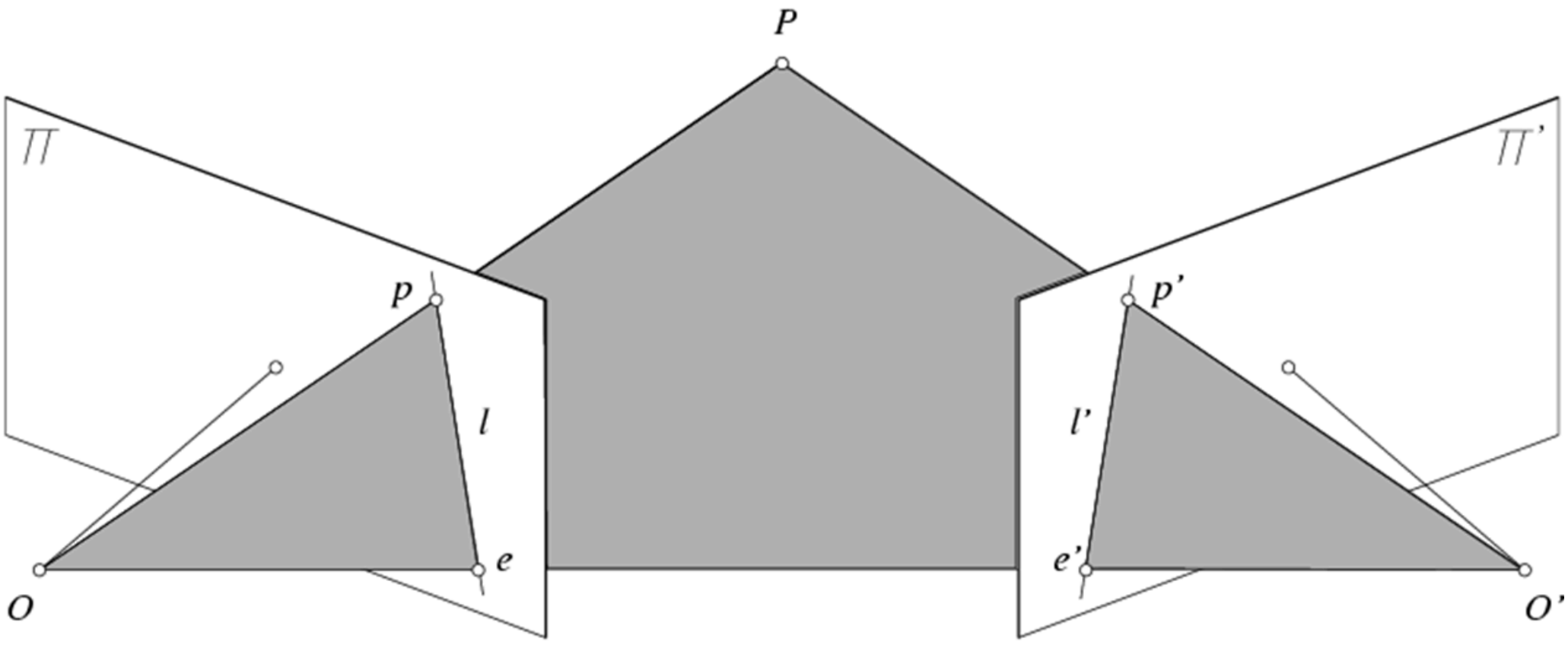
f is focal length, is a intrinsic parameter f 是焦距，是固有参数

b is baseline, is the distance between cameras b 是基线，是摄像机之间的距离

d is disparity, is the horizontal displacement of the same point in the left and right images d 是视差，是同一点在左右图像中的水平位移

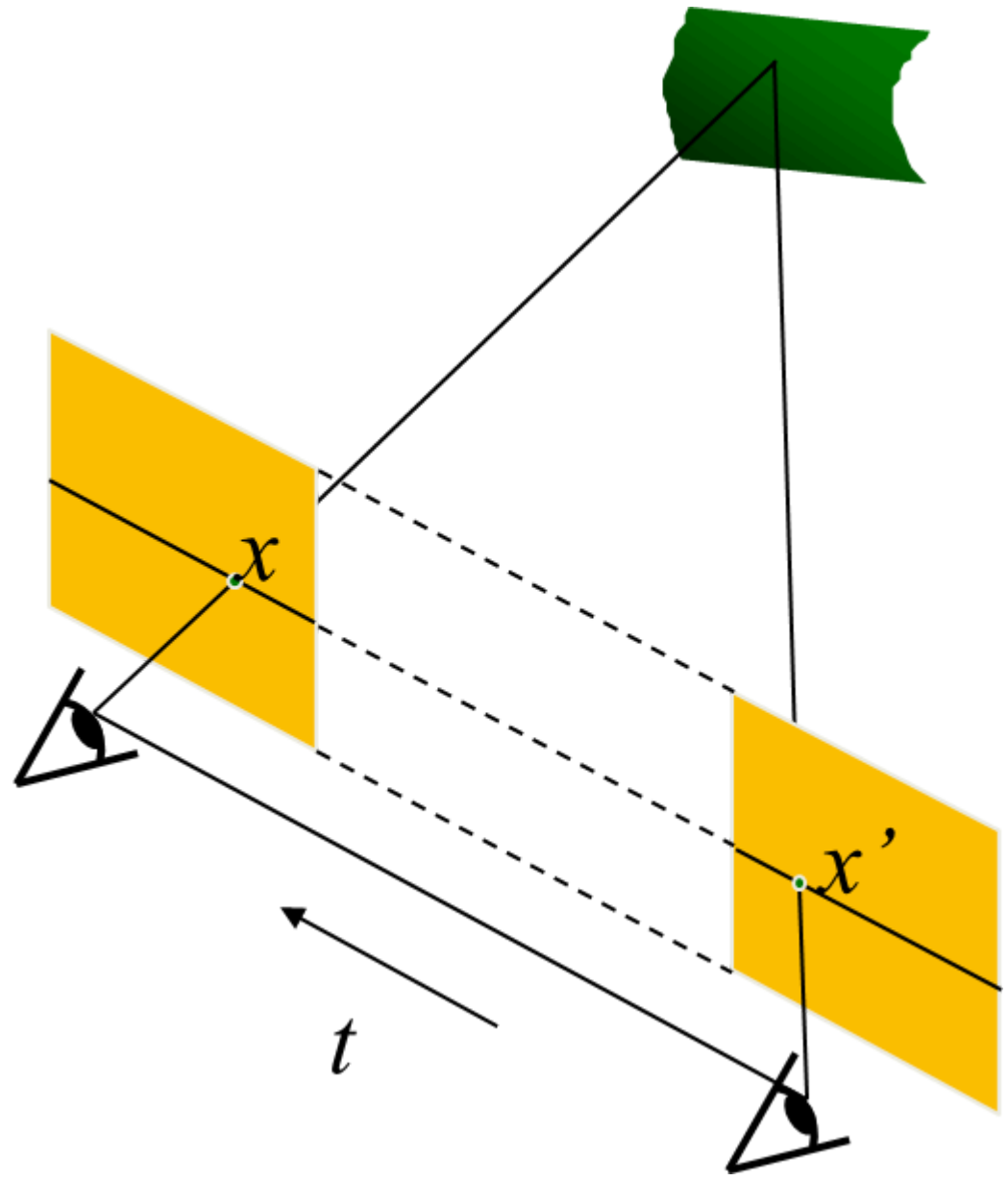
**depth is inversely proportional to disparity 深度与视差成反比**

## Epipolar geometry 外极几何




- Baseline $OO'$:line connecting the two camera centers 基线 连接两个摄影中心的直线
- Epipoles $e$ and $e'$: intersections of baseline with image planes= projections of the other camera center 外极点：基线与图像平面的交点=另一摄像机中心的投影
- Epipolar Plane $OO'P$: plane containing baseline (1D family) 外极平面 含基线的平面（一维族）
- Epipolar Lines $l$ and $l'$: intersections of epipolar plane with image planes (always come in corresponding pairs) 外极线 外极面与图像平面的交点（总是成对出现）

For the simples case: Parallel images 对于简单的情况： 平行图像




We have the Epipolar constraint 极几何约束公式

$$x^T E x' = 0, \quad E = [t_\times]R$$

$$R = I \qquad t = (T, 0, 0)$$

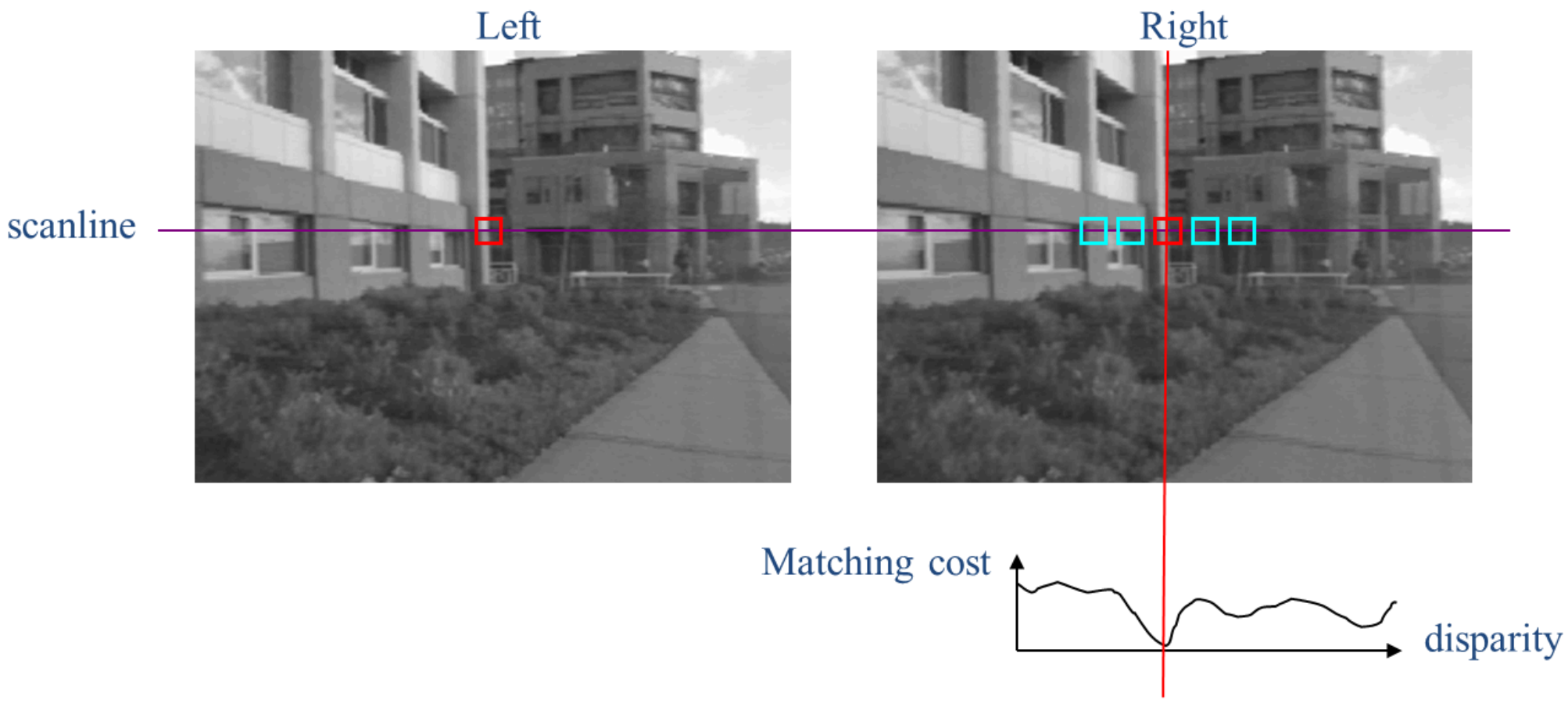
$$E = [t_\times]R = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -T \\ 0 & T & 0 \end{bmatrix}$$

$$\begin{pmatrix} u & v & 1 \end{pmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -T \\ 0 & T & 0 \end{bmatrix} \begin{pmatrix} u' \\ v' \\ 1 \end{pmatrix} = 0$$

The point in the left image $x$ and the point in the right image $x'$ satisfy a certain linear relationship $X^T E x' = 0$ 左右点满足一个确定线性关系

E includes rotation and translation information (R is I here) E 包括旋转和平移信息（此处 R 为 I）

# Stereo matching 立体匹配




Slide a window along the right scanline and compare contents of that window with the reference window in the left image
沿右侧扫描线滑动窗口，将窗口内容与左侧图像中的参考窗口进行比较

Matching cost is used to estimate the similarity between two windows, usually use SSD, SAD, or normalized cross correlation 匹配成本用于估算两个窗口之间的相似度，通常使用 SSD、SAD 或归一化交叉相关性

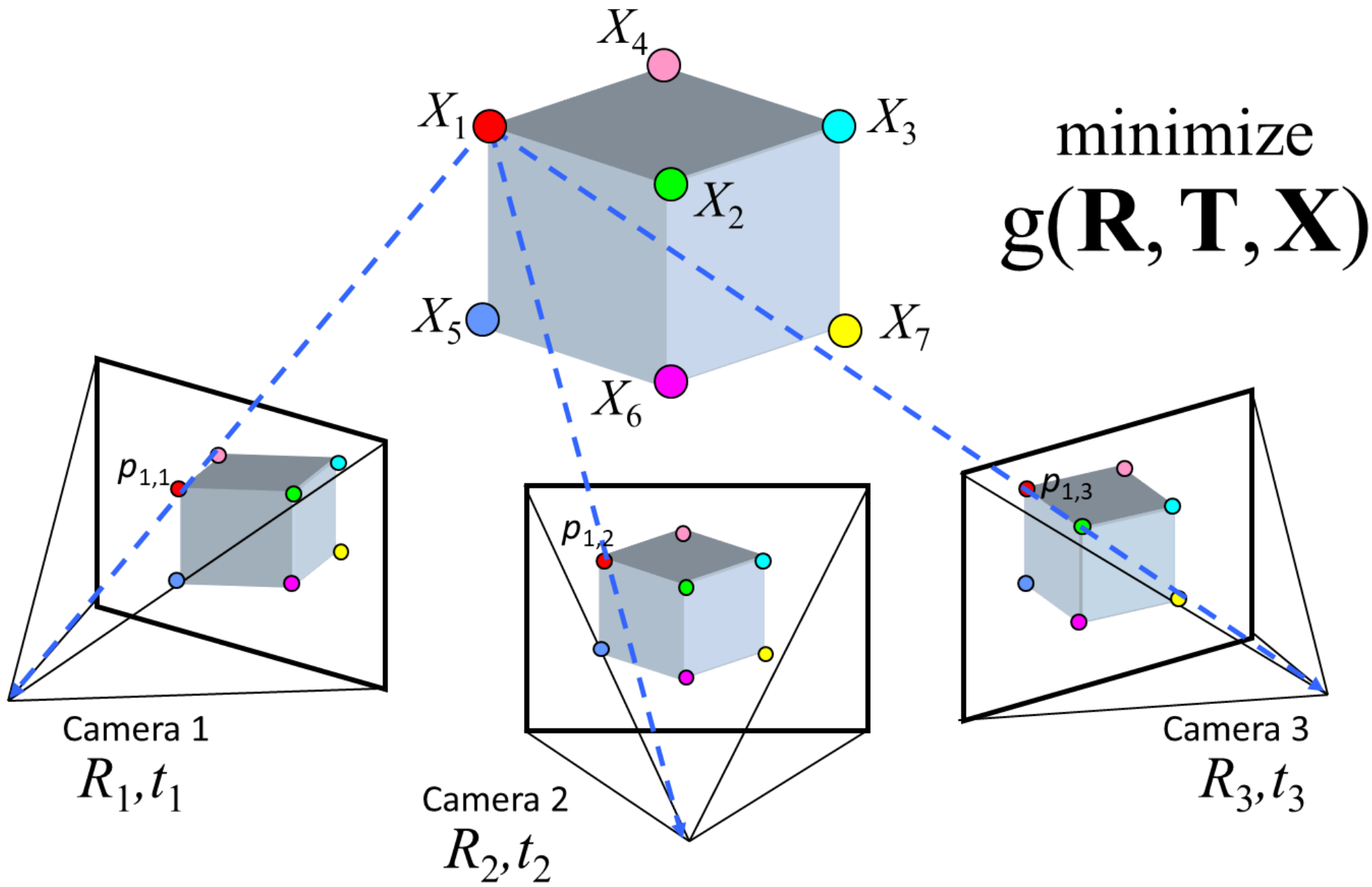
The horizontal displacement corresponding to the position of the sliding window with **the lowest matching cost** is the **disparity** at that point 与匹配成本最低的滑动窗口位置相对应的水平位移即为该点的视差。

smaller window: more detail, more noise 窗口更小：细节更多，噪点更多

bigger window: less noise, less detail 窗口更大：噪点更少，细节更少

| Similarity Measure | Formula |
| --- | --- |
| Sum of Absolute Differences (SAD) | $\sum_{(i,j)\in W} \lvert I_1(i,j) - I_2(x+i,y+j)\rvert$ |
| Sum of Squared Differences (SSD) | $\sum_{(i,j)\in W} \left(I_1(i,j) - I_2(x+i,y+j)\right)^2$ |
| Zero-mean SAD | $\sum_{(i,j)\in W} \lvert I_1(i,j) - \bar{I}_1(i,j) - I_2(x+i,y+j) + \bar{I}_2(x+i,y+j)\rvert$ |
| Normalized Cross Correlation (NCC) | $\frac{\sum_{(i,j)\in W} I_1(i,j).I_2(x+i,y+j)}{\sqrt[2]{\sum_{(i,j)\in W} I_1^2(i,j).\sum_{(i,j)\in W} I_2^2(x+i,y+j)}}$ |

# Structure from motion 运动结构

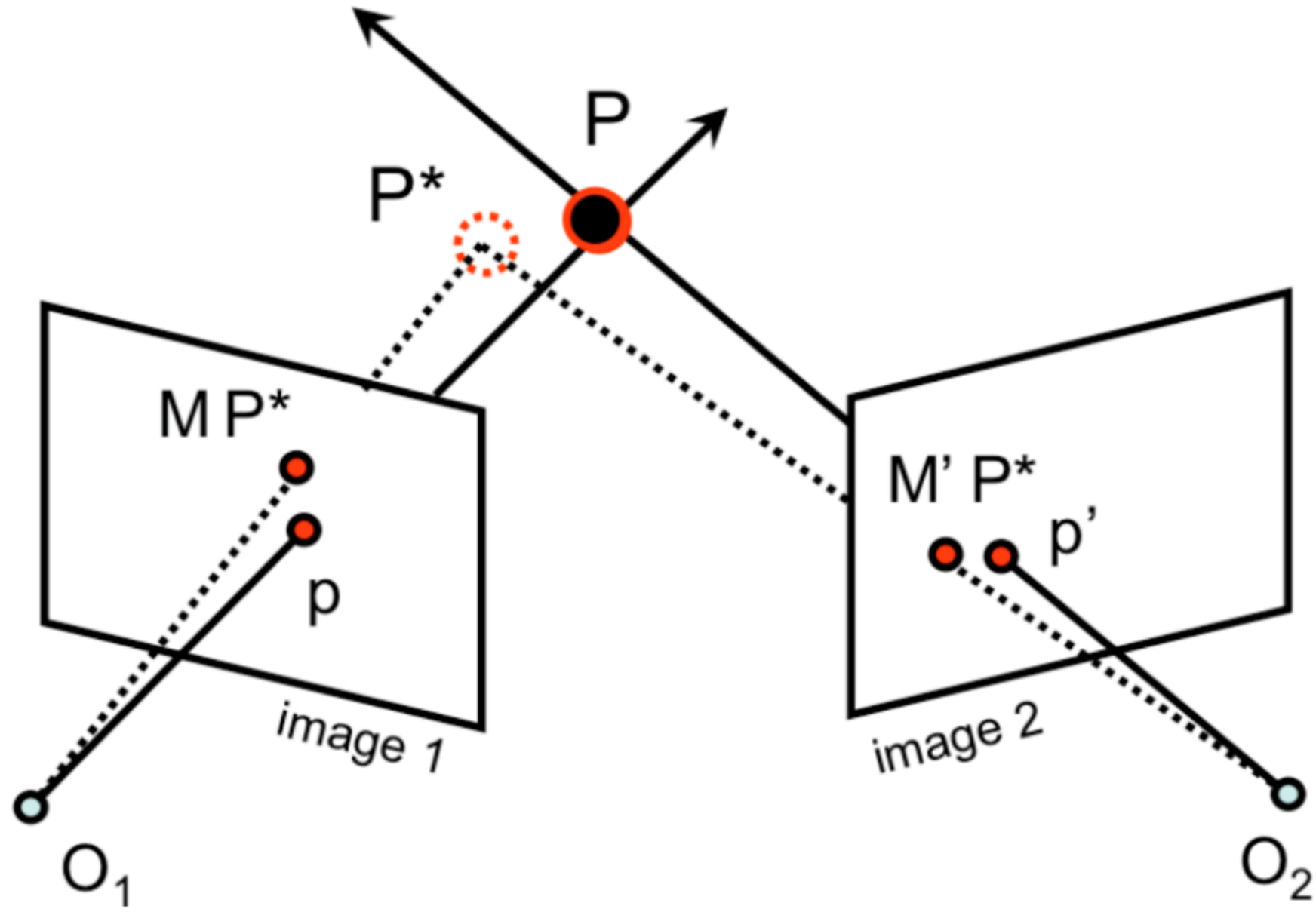



Reconstruct of a 3D scene and camera motion parameters from a series of images 从一系列图像中重建三维场景和摄像机运动参数

our goal is to minimize the function g, which represents error between all projected and actual points 我们的目标是最小化函数 g，它表示所有预测点和实际点之间的误差

**Re-projection Error 重投影错误**

considering 2 image plans 考虑 2 个图像计划



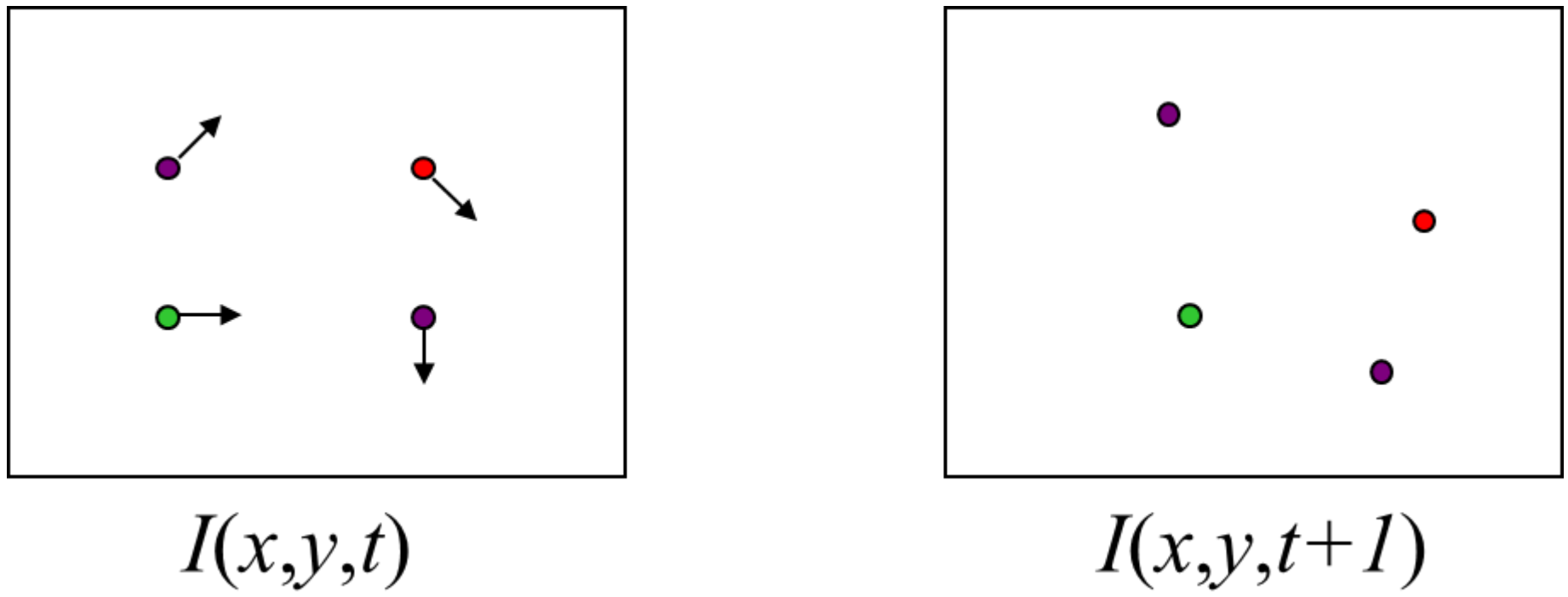

we wan to minimize error 我们要尽量减少误差

$$\min_{\hat{P}} \|M\hat{P} - p\|^2 + \|M'\hat{P} - p'\|^2$$

# Optical flow 光流

## Assumptions 假设

Given two subsequent frames, estimate the point translation 给定两个后续帧，估计点平移量

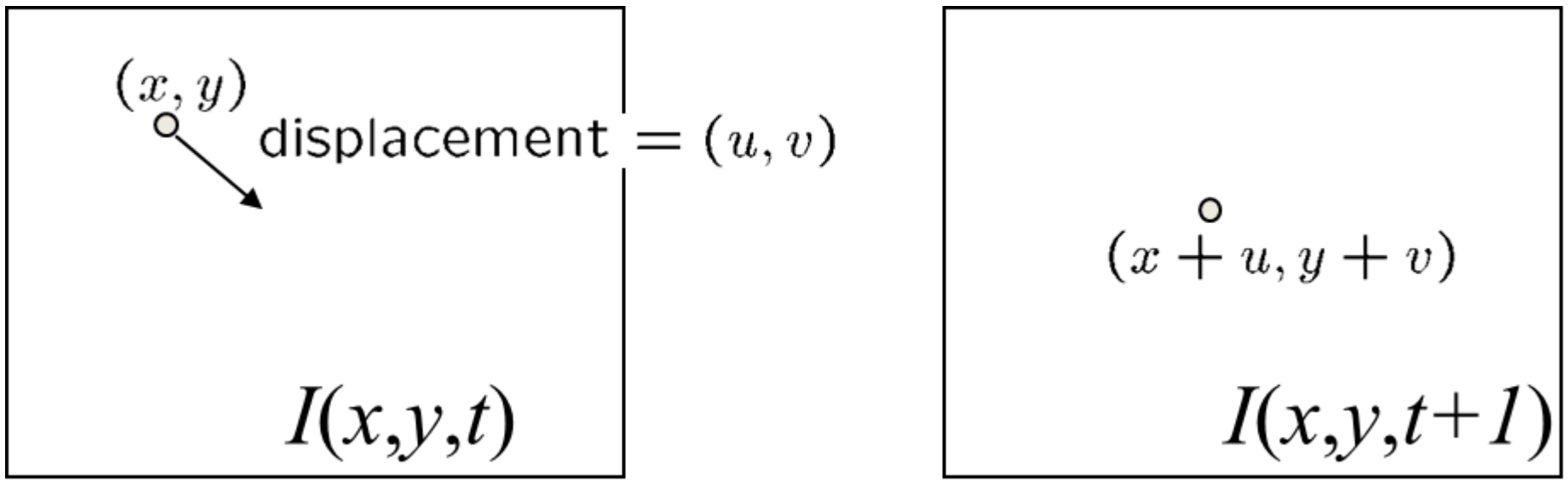



Key assumptions of **Lucas-Kanade** Tracker: **卢卡斯-卡纳德**跟踪器的主要假设：

- **Brightness constancy:** projection of the same point looks the same in every frame **亮度恒定性：** 同一点的投影在每个帧中看起来都一样

- **Small motion:** points do not move very far **小运动：**点不会移动太远
- **Spatial coherence:** points move like their neighbors **空间一致性：**点的移动与其邻近点相同

**The brightness constancy constraint 亮度恒定约束**




brightness constancy equation: 亮度恒等式

$$I(x,y,t) = I(x+u, y+v, t+1)$$

Take Taylor expansion of $I(x+u, y+v, t+1)$ at $(x,y,t)$ to linearize the right side:

Image derivative along x

$$I(x+u,y+v,t+1) \approx I(x,y,t) + I_x \cdot u + I_y \cdot v + I_t$$

$$I(x+u,y+v,t+1) - I(x,y,t) = +I_x \cdot u + I_y \cdot v + I_t$$

So: $$I_x \cdot u + I_y \cdot v + I_t \approx 0 \quad \rightarrow \nabla \mathrm{I} \cdot \left[\mathrm{u} \;\; \mathrm{v}\right]^{\mathrm{T}} + \mathrm{I_t} = 0$$

By using this method, we could estimate object's movement 利用这种方法，我们可以估算出物体的运动轨迹

It's similar to minimizing the squared error of I(t) and I(t+1) 这类似于最小化 I(t) 和 I(t+1) 的平方误差

**Spatial coherence constraint 空间一致性约束**

Another useful equation 另一个有用的等式

Assume the pixel’s neighbors have the same (u,v) 假设像素的邻居具有相同的 (u,v)

If we use a 5x5 window, that gives us 25 equations per pixel 如果使用 5x5 窗口，每个像素就有 25 个方程

$$0 = I_t(\mathbf{p_i}) + \nabla I(\mathbf{p_i}) \cdot [u\ v]$$

$$\begin{bmatrix} I_x(\mathbf{p_1}) & I_y(\mathbf{p_1}) \\ I_x(\mathbf{p_2}) & I_y(\mathbf{p_2}) \\ \vdots & \vdots \\ I_x(\mathbf{p_{25}}) & I_y(\mathbf{p_{25}}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p_1}) \\ I_t(\mathbf{p_2}) \\ \vdots \\ I_t(\mathbf{p_{25}}) \end{bmatrix}$$

extension of brightness constancy equation 亮度恒等式的扩展

**Matching patches across images 跨图像匹配补丁**

first, Overconstrained linear system 首先，超约束线性系统

$$\begin{bmatrix} I_x(\mathbf{p_1}) & I_y(\mathbf{p_1}) \\ I_x(\mathbf{p_2}) & I_y(\mathbf{p_2}) \\ \vdots & \vdots \\ I_x(\mathbf{p_{25}}) & I_y(\mathbf{p_{25}}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p_1}) \\ I_t(\mathbf{p_2}) \\ \vdots \\ I_t(\mathbf{p_{25}}) \end{bmatrix} \qquad \underset{25\text{x}2}{A} \quad \underset{2\text{x}1}{d} = \underset{25\text{x}1}{b}$$

To get the translation vector d: 要得到平移矢量 d

Least squares solution for *d* given by $(A^T A)\ d = A^T b$

$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

$$A^T A \qquad\qquad A^T b$$

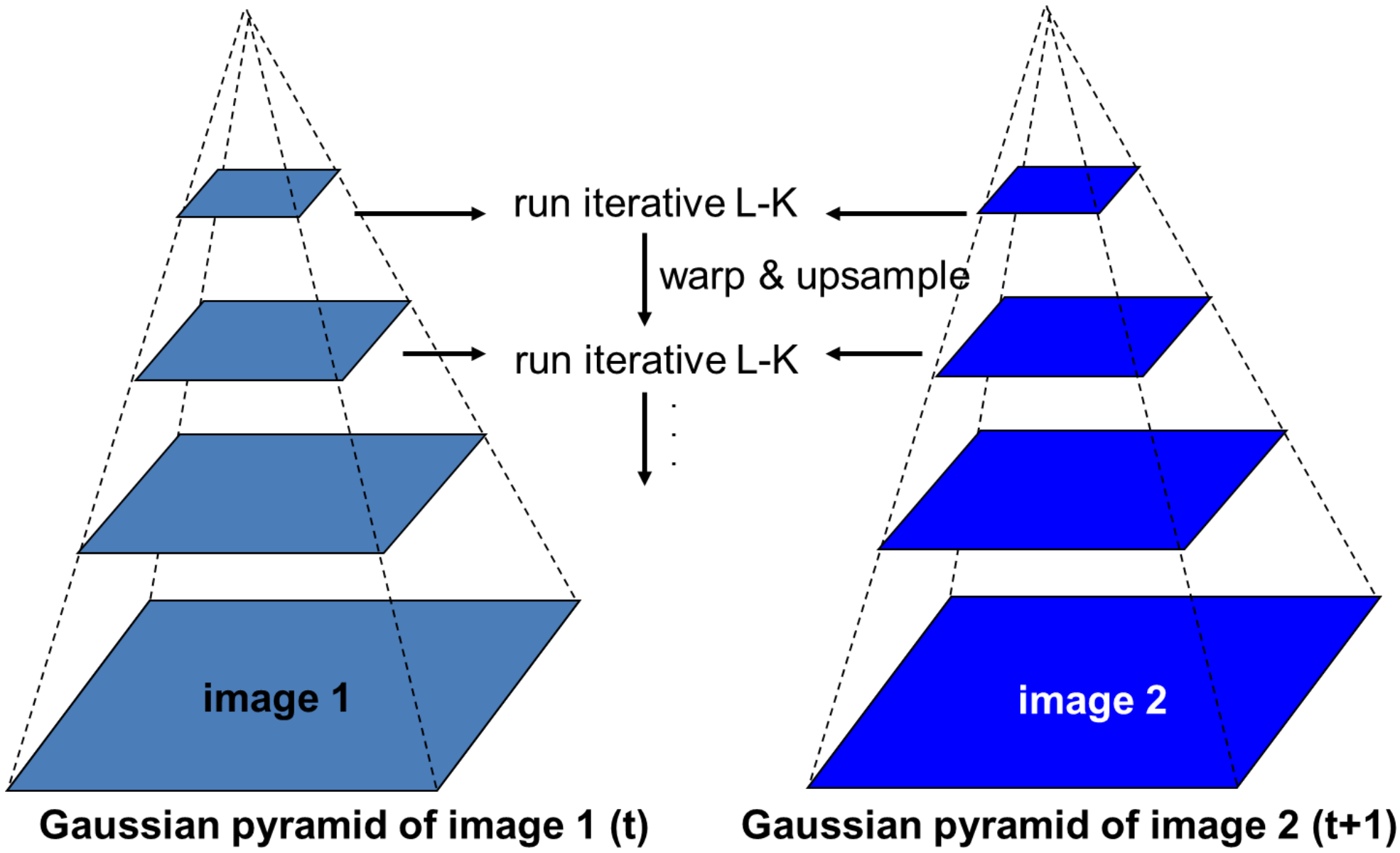
The summations are over all pixels in the K x K window 求和是对 K x K 窗口中所有像素的求和

Criteria for Harris corner detector

# Iterative Refinement 迭代改进

Iterative Lucas-Kanade Algorithm: 迭代卢卡斯-卡纳德算法：

1. Estimate velocity at each pixel by solving Lucas-Kanade equations 通过求解卢卡斯-卡纳德方程，估算每个像素的速度
2. Warp I(t-1) towards I(t) using the estimated flow field(*use image warping techniques*)(like **Affine Transformations**) 利用估计流场（使用图像扭曲技术）（如仿射变换）将 I（t-1）向 I（t）方向扭曲
3. Repeat until convergence 重复直至收敛




TOP level: 最高级别

1. Apply L-K to get a flow field representing the flow from the first frame to the second frame. 应用 L-K 得到一个流场，代表从第一帧到第二帧的流动。
2. Apply this flow field to warp the first frame toward the second frame. 应用该流场将第一帧向第二帧翘曲。
3. Return L-K on the new warped image to get a flow field from it to the second frame 在新的扭曲图像上返回 L-K，以获得从它到第二帧的流场
4. Repeat till convergence 重复直至收敛

Next level: 下一层

1. Upsample the flow field to the next level as the first guess of the flow at that level. 将流场上样到下一级，作为对该级流量的首次猜测。
2. Apply this flow field to warp the first frame toward the second frame. 应用该流场将第一帧向第二帧翘曲。
3. Rerun L-K and warping till convergence as above. 按上述方法重新运行 L-K 和翘曲直到收敛。