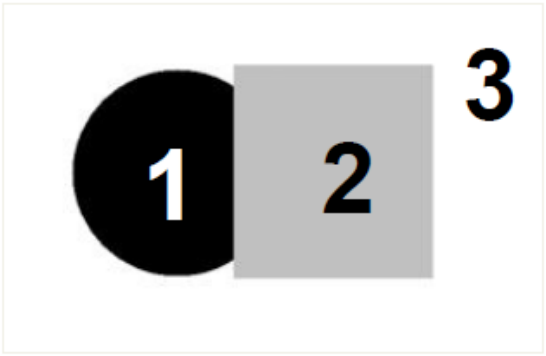
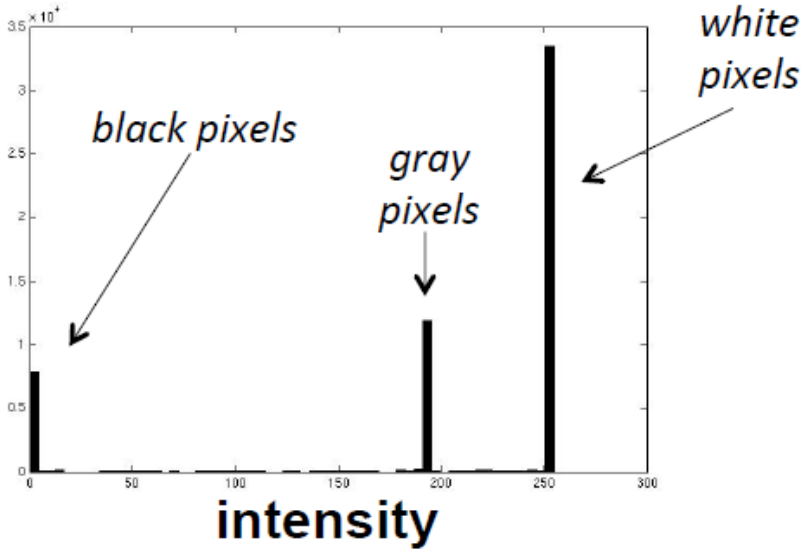
7 图像分割
IMAGE SEGMENTATION

图像分割：示例



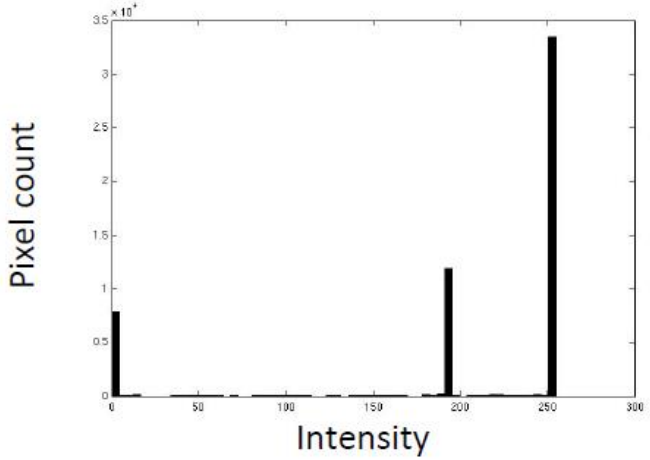
input image

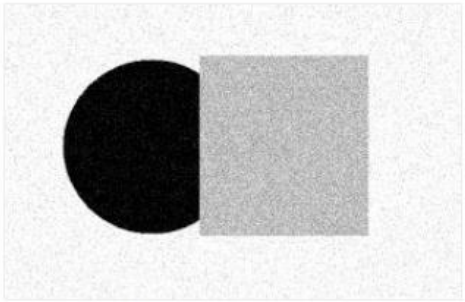


- 这些灰度定义了三个组。
- 我们可以根据这些主要强度中的哪个像素来标记图像中的每个像素。
  - 即，根据灰度特征分割图像。
- 如果图像不是那么简单怎么办？

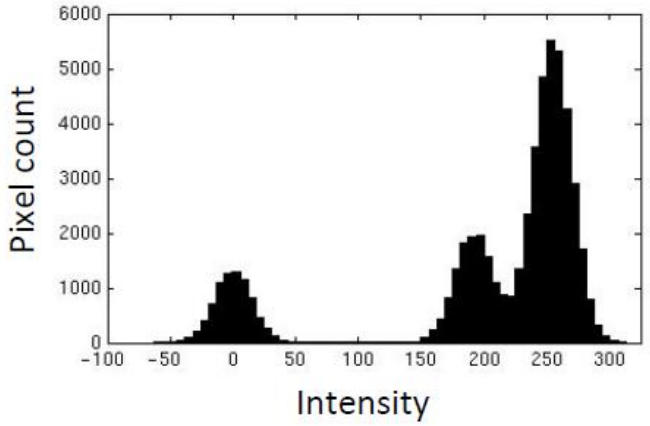
7.6.1 K均值聚类

Input image



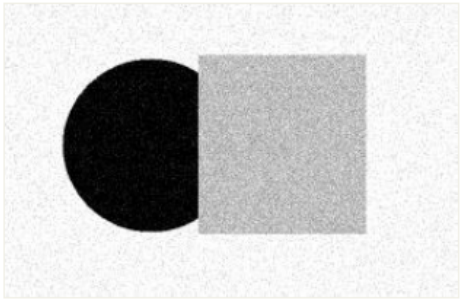


Input image

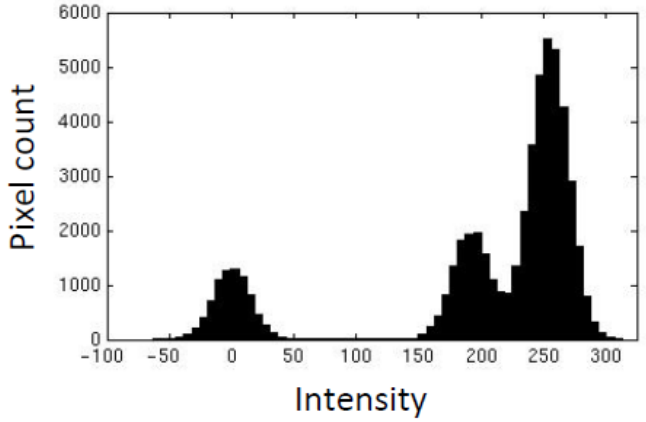


Slide credit: Kristen Grauman

7.6.1 K均值聚类

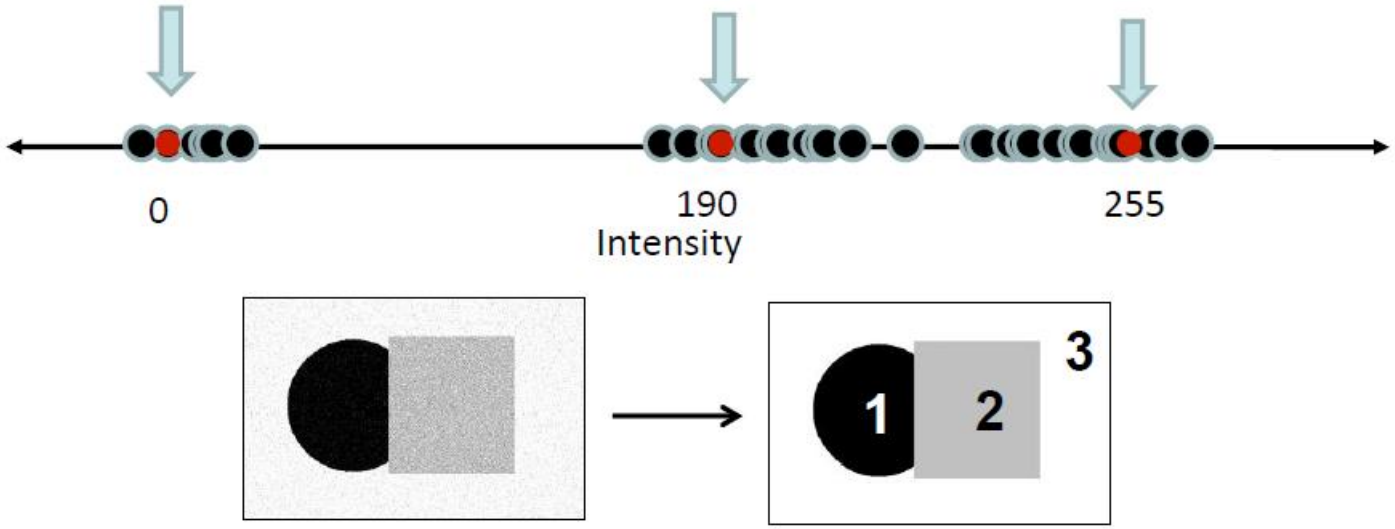


Input image



- 现在如何确定定义我们分组的三个主要强度？
- 我们需要聚类。

7.6.1 K均值聚类

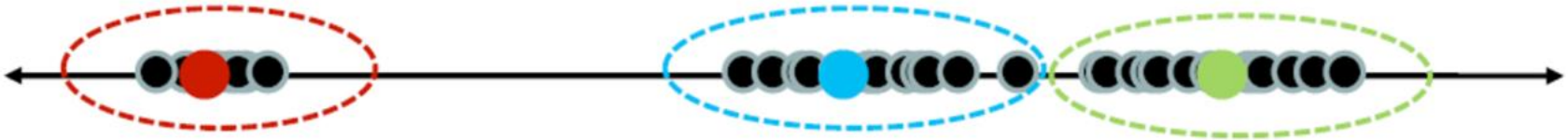


- 目标：选择三个“中心”作为代表性灰度，并根据每个像素中离它们最近的哪个中心进行标记。
- 最佳聚类中心是最小化所有点与其最近的聚类中心之间的平方距离之和 （SSD） 的中心 c_i

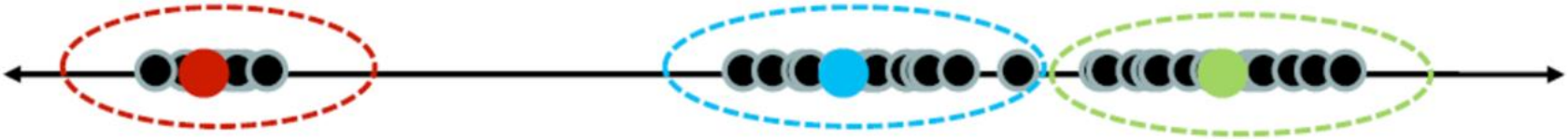
7.6.1 K均值聚类

下面这个事件，属于“鸡和鸡蛋”的问题：

如果我们知道簇的中心，我们可以将距离中心最近的点分配给这个簇



如果我们知道这个簇的成员，我们可以通过计算每个簇的均值的得到簇的中心



7.6.1 K均值聚类

- 基本思想：随机初始化 k 个聚类中心，并在我们刚刚看到的两个步骤之间迭代。

➢ 算法步骤：

1. 随机初始化簇的中心，$C_1, \ldots, C_k$
2. 给定聚类中心，确定每个聚类中的点
   - 对于每个点 p，找到最接近的 C_i。将 p 放入C_i中。
3. 每个聚类中的给定点，求解 C_i i
   - 将C_i设置为聚类 i 中点的平均值。
4. 如果C_i已更改，重复步骤2

➢ 属性

- 将始终收敛到某个解决方案
- 可以是“局部最小值”

聚类相关的总结

- 目标：聚类以最小化给定聚类中数据的差异

- 保留信息

簇的中心

数据

$$\mathbf{c}^*, \boldsymbol{\delta}^* = \underset{\mathbf{c},\boldsymbol{\delta}}{\operatorname{argmin}} \frac{1}{N} \sum_{j}^{N} \sum_{i}^{K} \delta_{ij} \left(\mathbf{c}_i - \mathbf{x}_j \right)^2$$

$\mathbf{x}_j$是否分配给$\mathbf{c}_i$

K均值

1.初始化簇的中心：$\mathbf{c}^0$；t=0

2.每个点都分配给离他最近的中心点

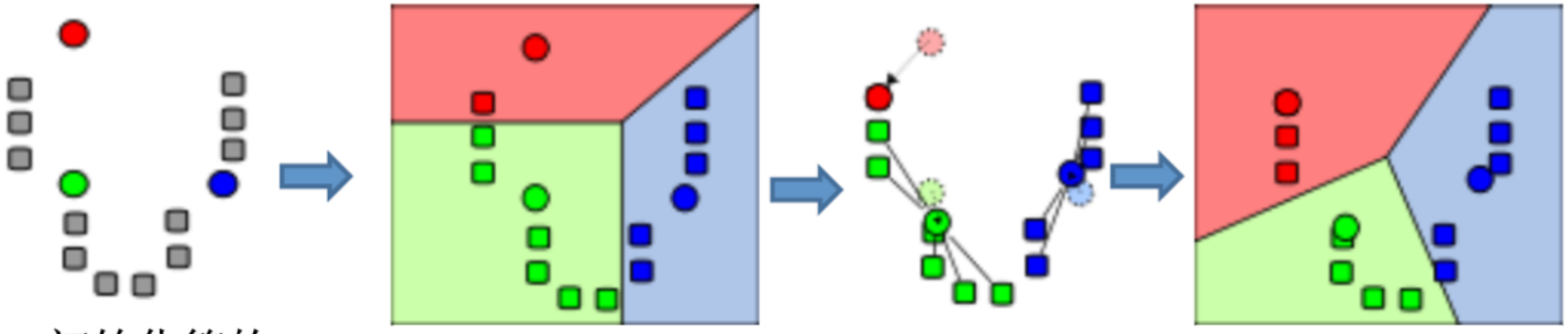
$$\boldsymbol{\delta}^t = \underset{\boldsymbol{\delta}}{\operatorname{argmin}} \frac{1}{N} \sum_j^N \sum_i^K \delta_{ij} \left(\mathbf{c}_i^{t-1} - \mathbf{x}_j \right)^2$$

3.用各个点的均值来更新簇的中心

$$\mathbf{c}^t = \underset{\mathbf{c}}{\operatorname{argmin}} \frac{1}{N} \sum_j^N \sum_i^K \delta_{ij}^t \left(\mathbf{c}_i - \mathbf{x}_j \right)^2$$

4.重复进行2-3步骤，直至没有点被重新分配

K均值



K均值（设计选择）

- 初始化
  - 随机选择 K 点作为初始聚类中心
  - 或贪心地选择 K 点以最小化残差

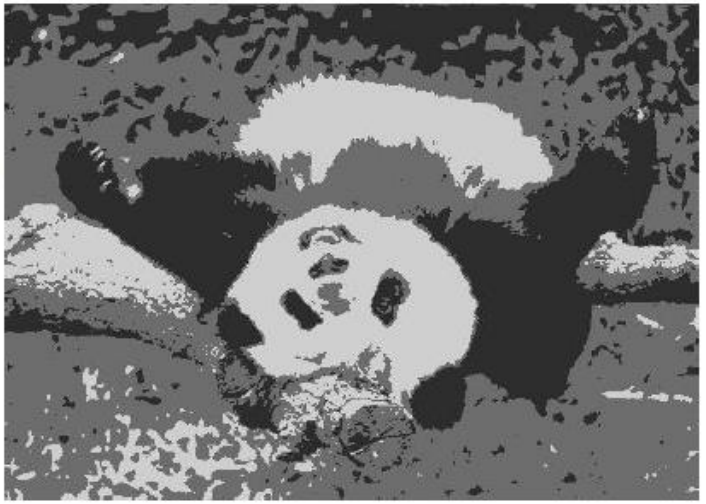
- 距离测量
  - 传统上Euclidean，可能是其他的

- 优化
  - 将收敛到局部最小值
  - 可能想要执行多次重新启动

用聚类分割

K=2

K=3



```
img_as_col = double(im(:));
cluster_membs = kmeans(img_as_col, K);

labelim = zeros(size(im));
for i=1:k
    inds = find(cluster_membs==i);
    meanval = mean(img_as_column(inds));
    labelim(inds) = meanval;
end
```

Slide credit: Kristen Grauman

特征空间

- 每个令牌都由一组称为特征的显著视觉特征标识。例如：
  - 位置
  - 颜色
  - 纹理
  - 运动矢量
  - 大小，方向（如果令牌大于像素）

- 特征的选择及其量化方式意味着一个特征空间，其中每个令牌都由一个点表示

- 令牌相似性因此由特征空间中的点（“特征向量”）之间的距离来衡量

特征空间

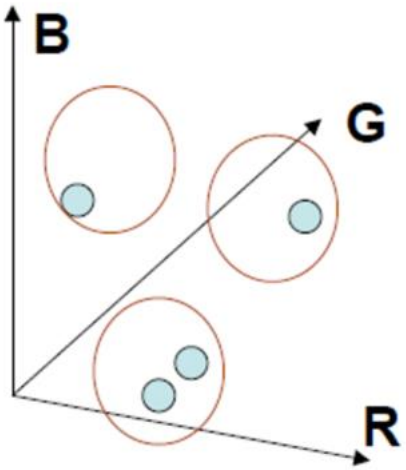
- 根据我们选择的功能空间，我们可以以不同的方式对像素进行分组。

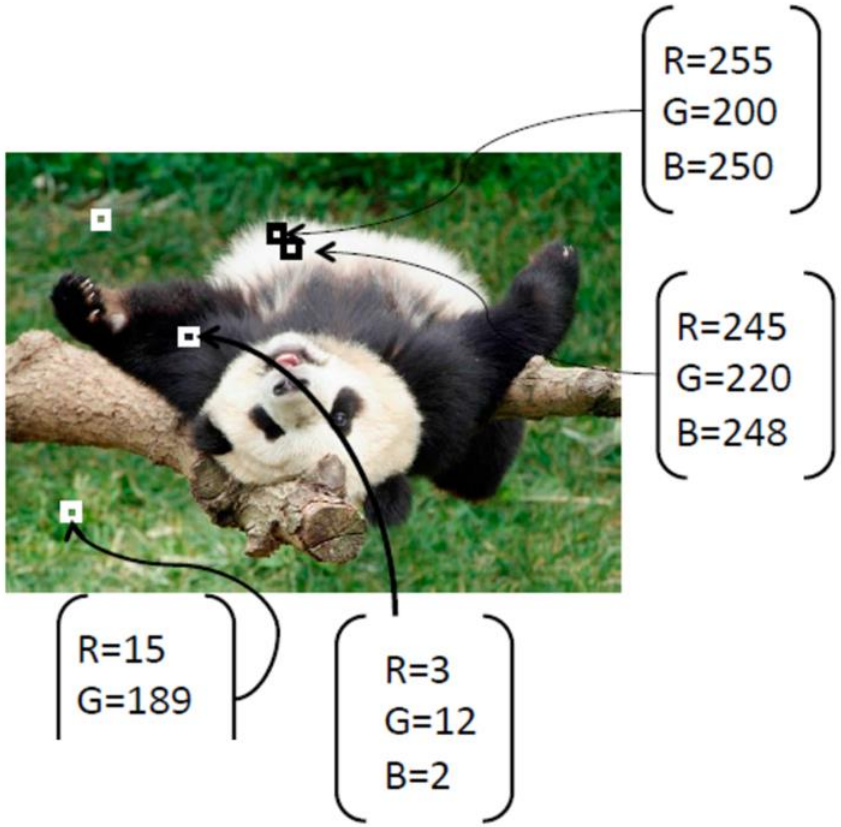
- 根据灰度相似性对像素进行分组

- 特征空间：灰度值（1D）

特征空间

- 根据我们选择的功能空间，我们可以以不同的方式对像素进行分组。
- 根据颜色相似性对像素进行分组



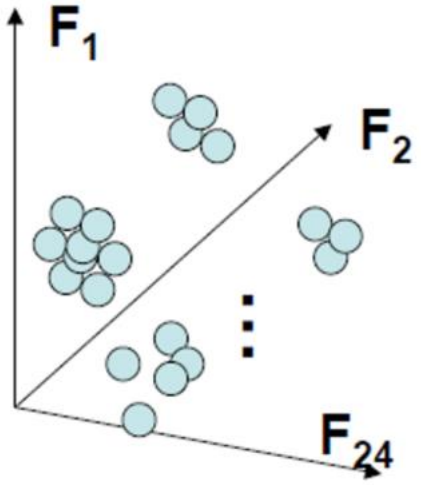


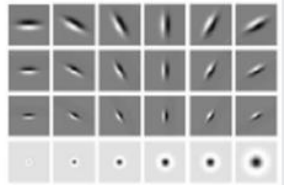
- 特征空间：灰度值（3D）

特征空间

- 根据我们选择的功能空间，我们可以以不同的方式对像素进行分组。

- 根据纹理相似性对像素进行分组





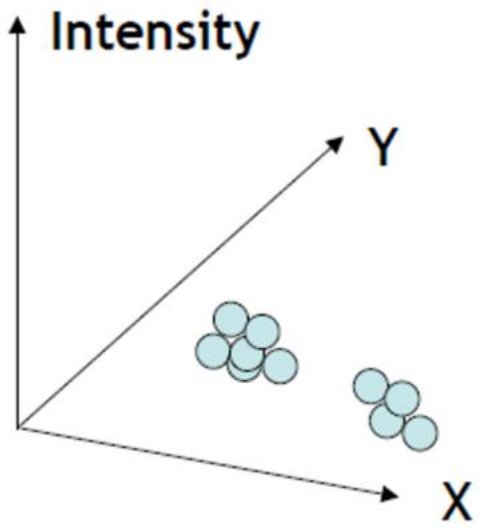
Filter bank of
24 filters

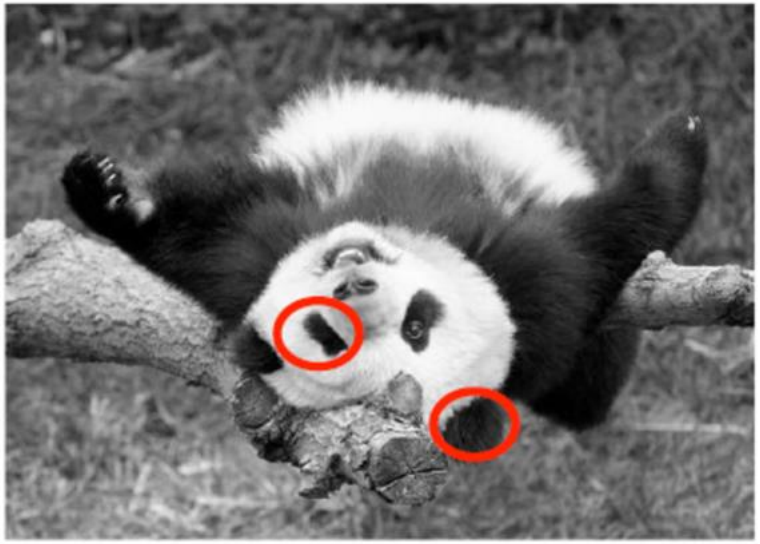
- 特征空间：过滤组响应（例如，24D）

特征空间-用聚类分割

- 根据我们选择的功能空间，我们可以以不同的方式对像素进行分组。

- 根据灰度+位置相似性对像素进行分组



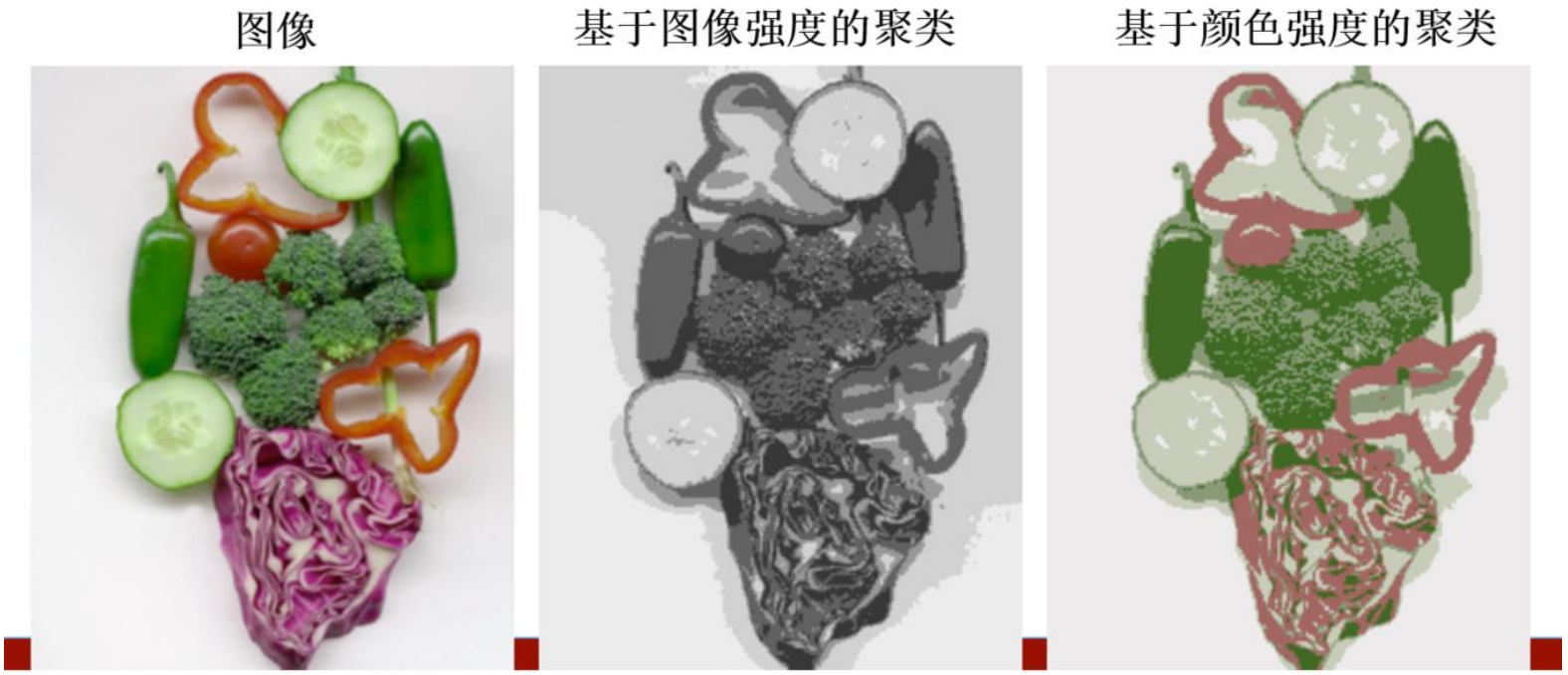


=>对相似性和邻近性进行编码的方法。

特征空间-用聚类分割

- 基于灰度或颜色的 K 均值聚类，本质上是图像属性的矢量量化。

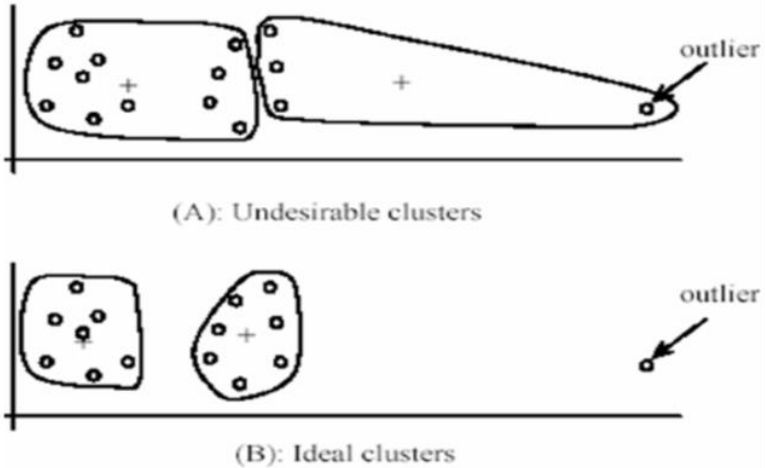
  - 聚类不必是空间相干的

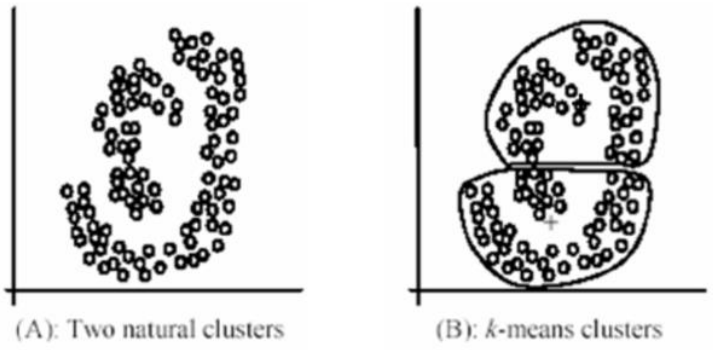


K均值总结

- 优点
  - 简单，快速计算
  - 收敛到聚类内平方误差的局部最小值

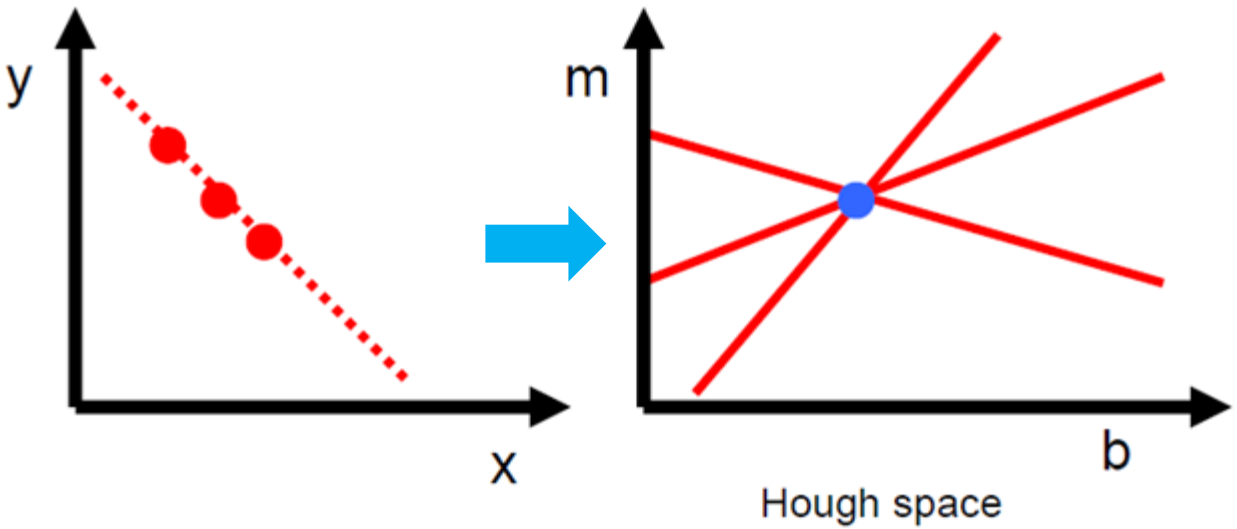
- 缺点/问题
  - 设置 k?
  - 对初始中心敏感
  - 对异常值敏感
  - 仅检测球形聚类
  - 假设可以计算均值





7.6.2：霍夫变换

■ 变换思想：给定一组待分类的数据点，找到最能解释数据点的曲线或直线。



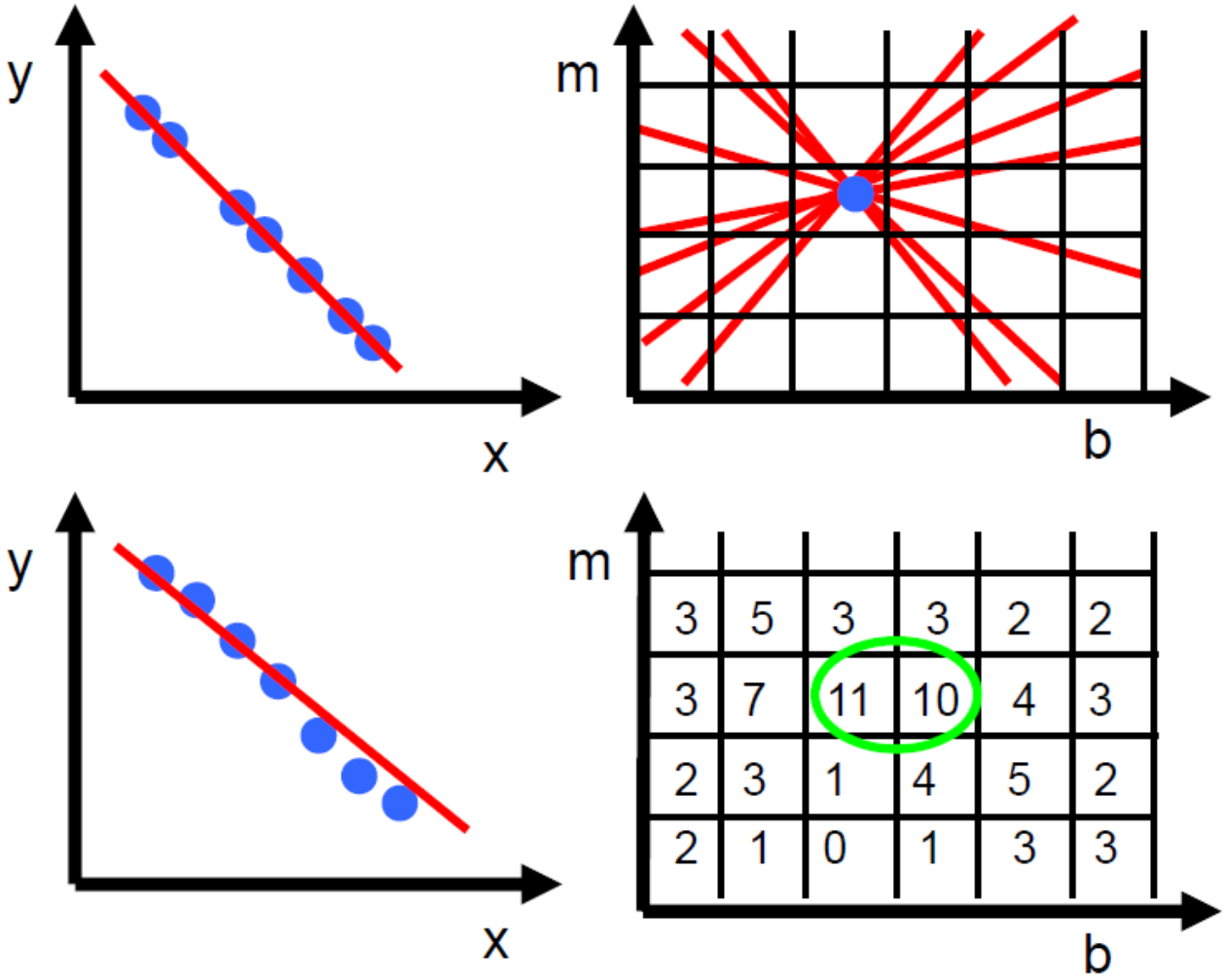
霍夫空间：

$q = -kX_1 + Y_1$

$q = -kX_2 + Y_2$

- 左图：两个点$A=(X_1,Y_1)$和$B=(X_2,Y_2)$确定直线：$y=kx+q$。
- 右图：$y=kx+q$也可以写成关于自变量为（k,q）的函数表达式。

霍夫变换



Slide from S. Savarese

霍夫变换

- 问题：参数空间 [k, q] 是无界的。

✓ 对参数空间使用极坐标表示

$$X_1 = \rho \cos\theta \quad Y_1 = \rho \sin\theta$$

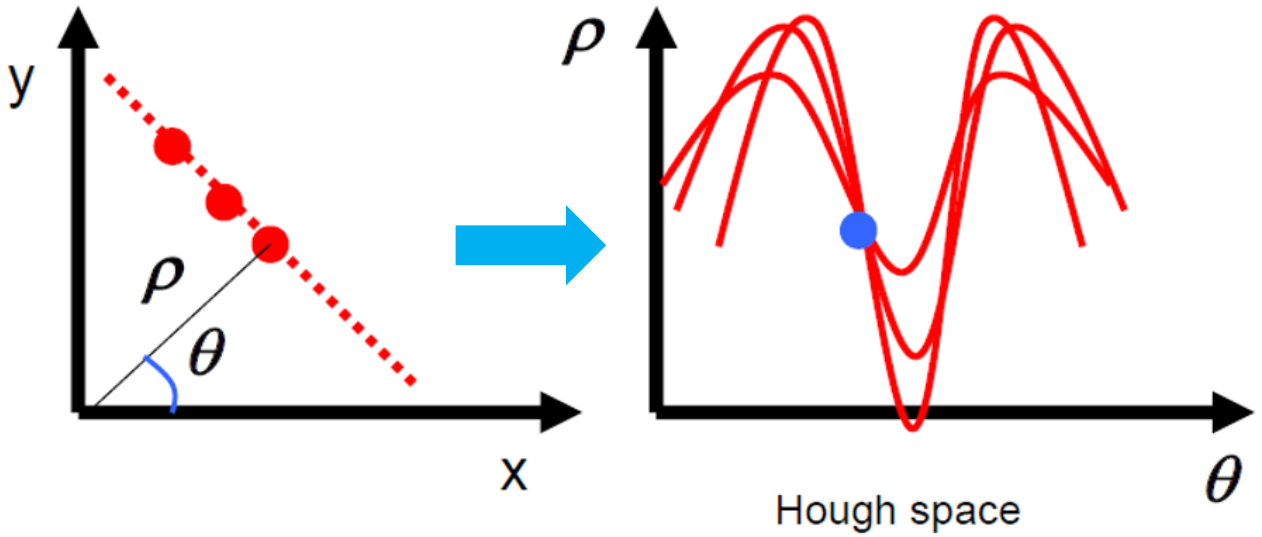
$$X_1 \cos\theta = \rho \cos^2\theta \quad Y_1 \sin\theta = \rho \sin^2\theta$$

$$X_1 \cos\theta + Y_1 \sin\theta = \rho$$

霍夫变换

■ 问题：参数空间 [k, q] 是无界的。

✓ 对参数空间使用极坐标表示



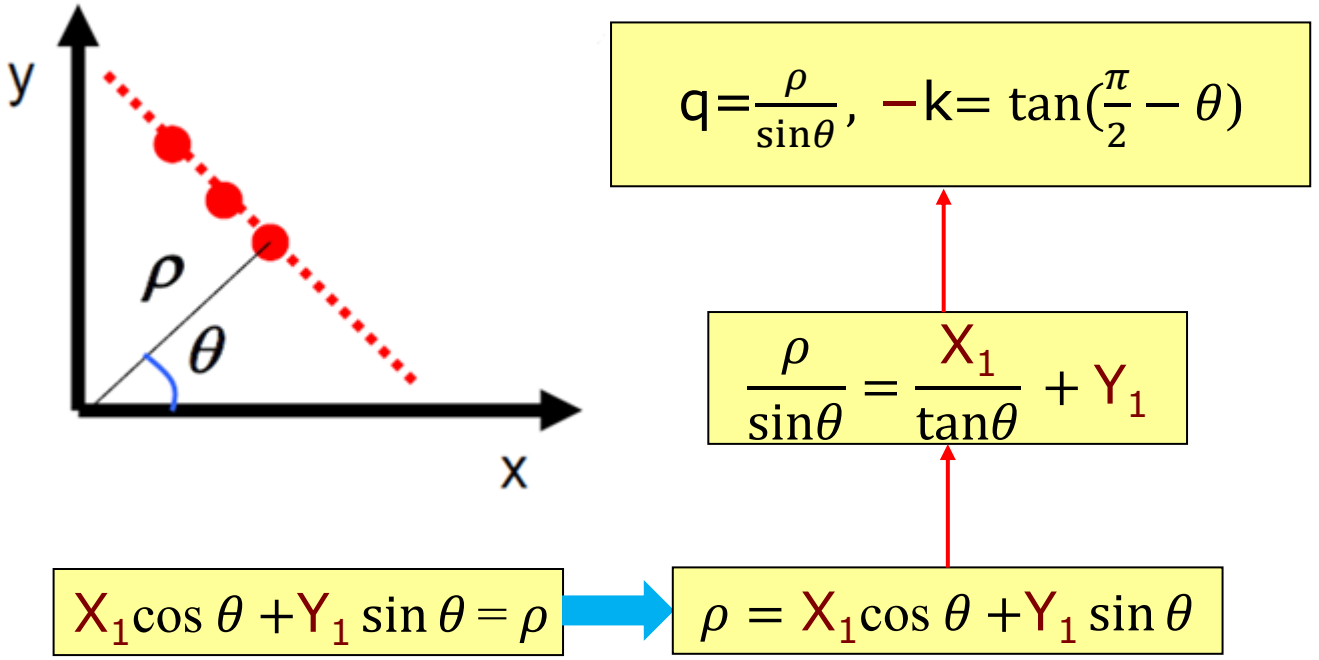
$X_1 \cos\theta + Y_1 \sin\theta = \rho$

$\rho = X_1 \cos\theta + Y_1 \sin\theta$

霍夫变换

■ 问题：参数空间 [k, q] 是无界的。

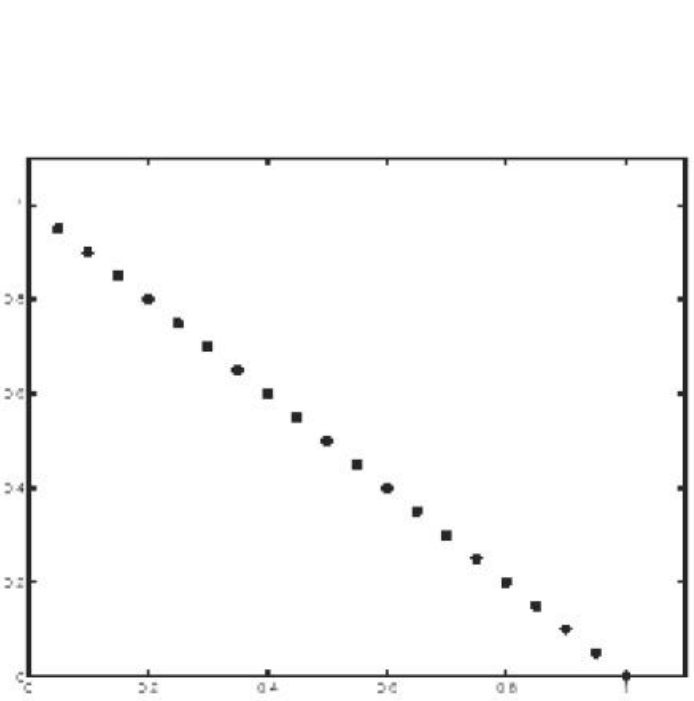
✓ 对参数空间使用极坐标表示



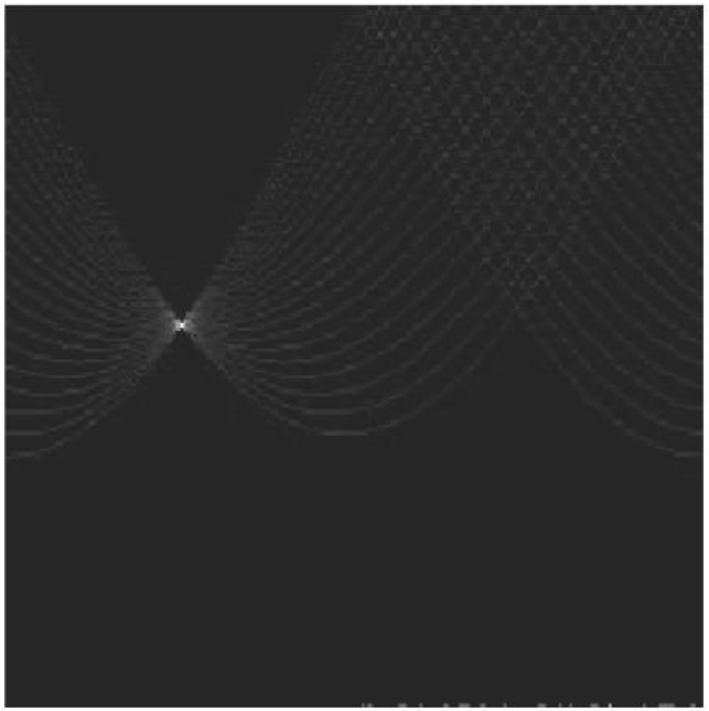
霍夫变换

- **霍夫变换的离散算法（大纲）：**
  - 创建参数值网格
  - 每个点对一组参数进行**投票**，在网格中递增这些值
  - 在网格中查找最大值或局部最大值

霍夫变换-实验



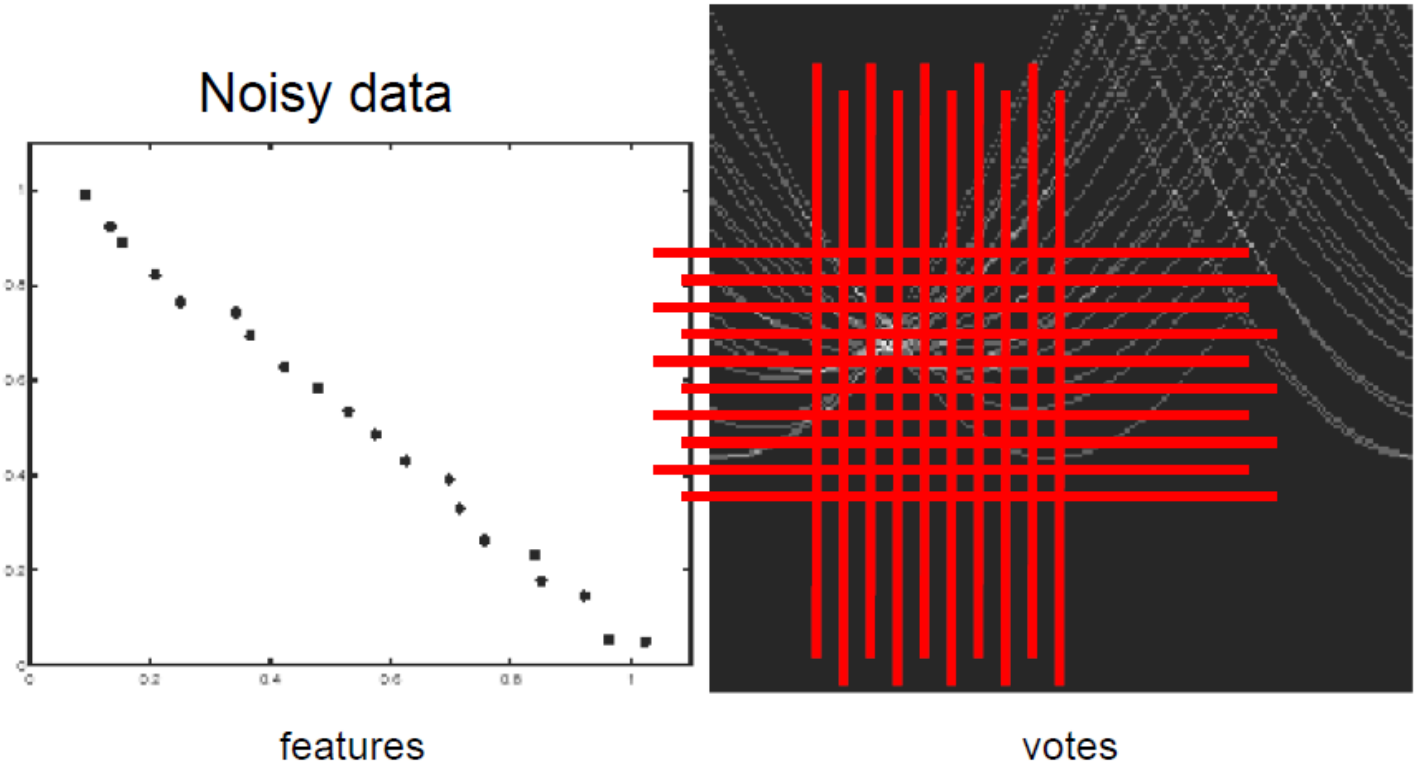
features



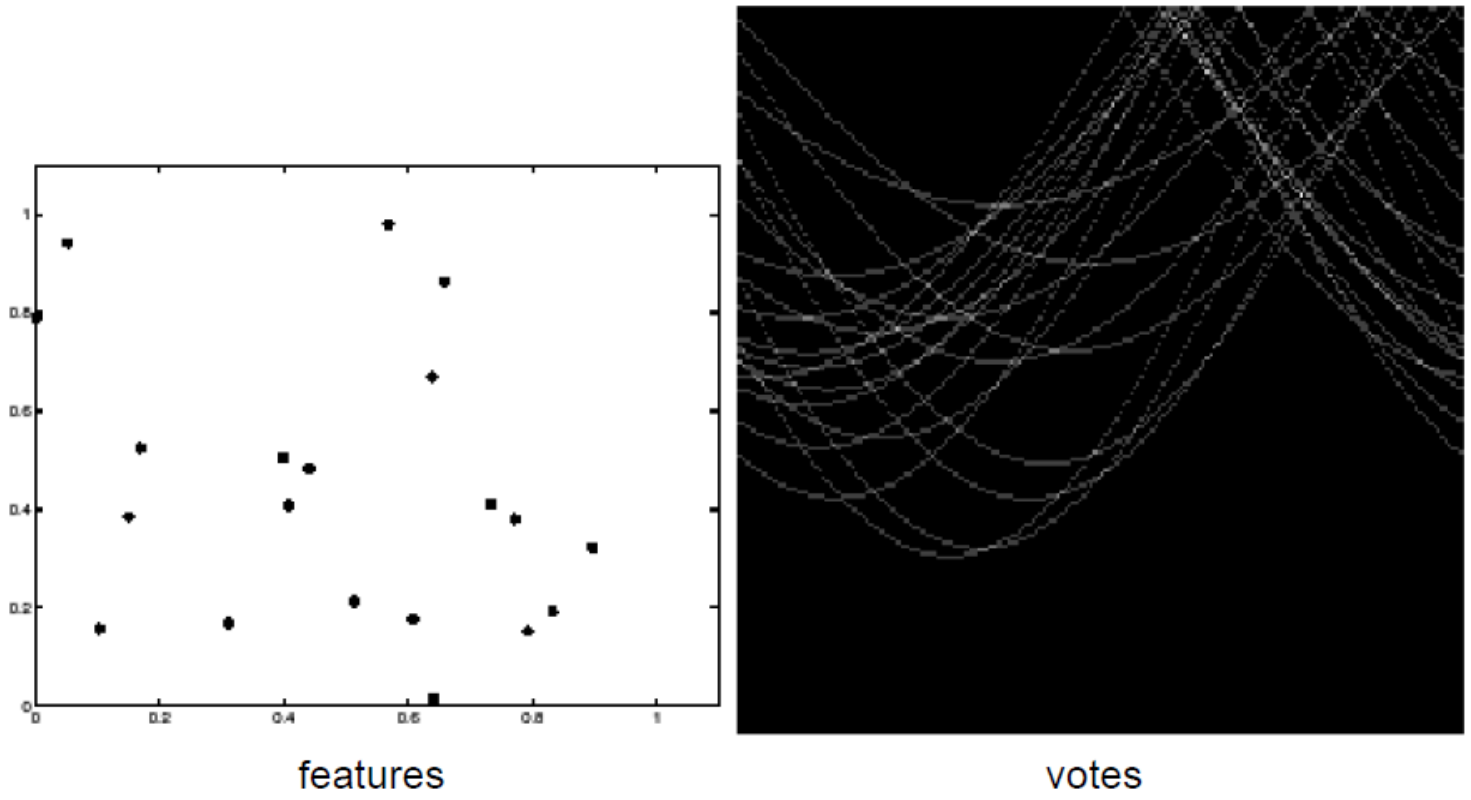
votes

Slide from S. Savarese

霍夫变换-实验

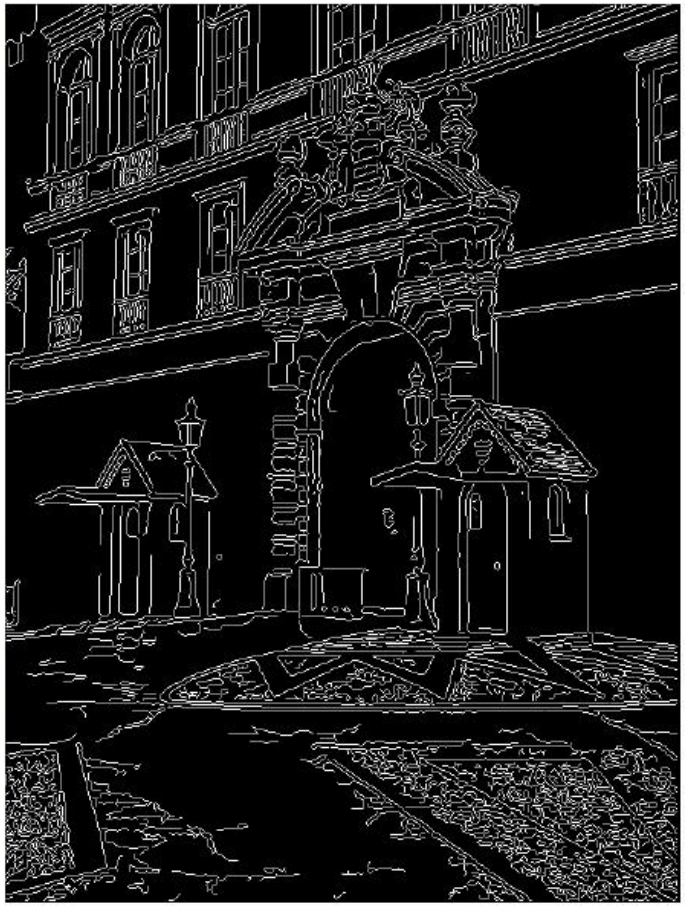


霍夫变换-实验

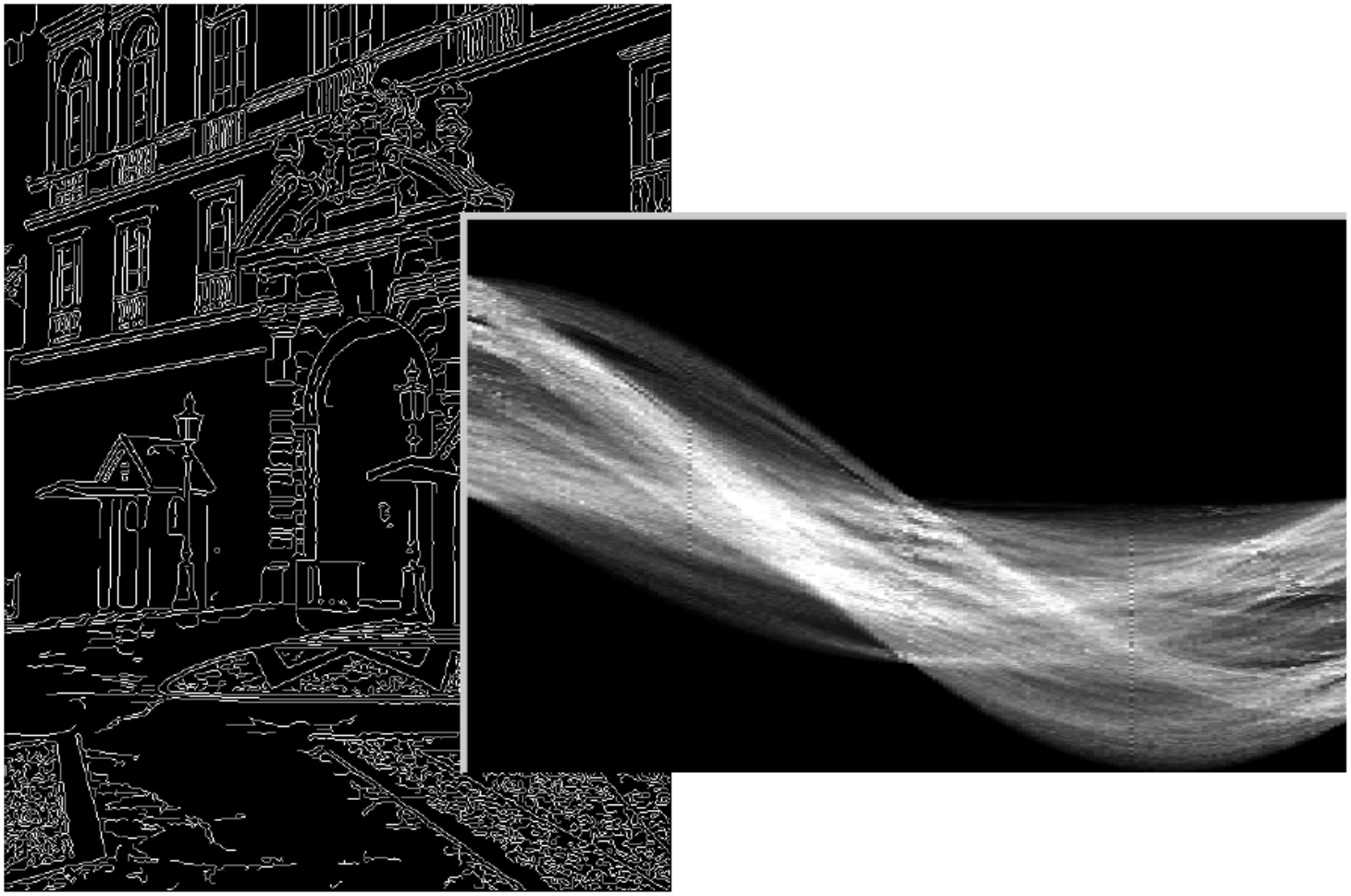


均匀噪声引起的虚假峰值

图像边缘

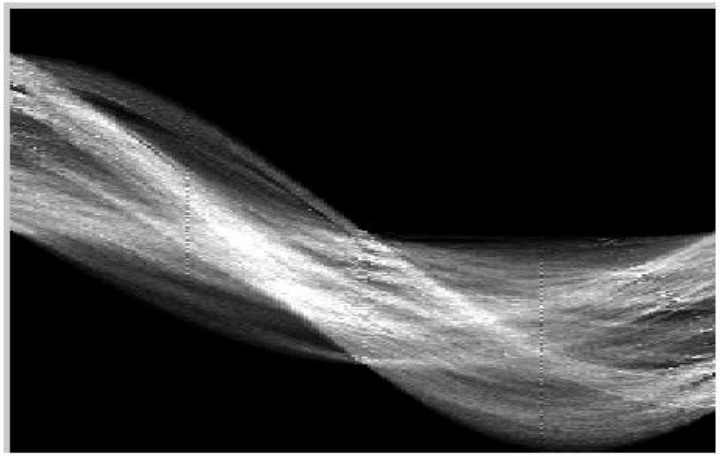


图像边缘——霍夫参数空间

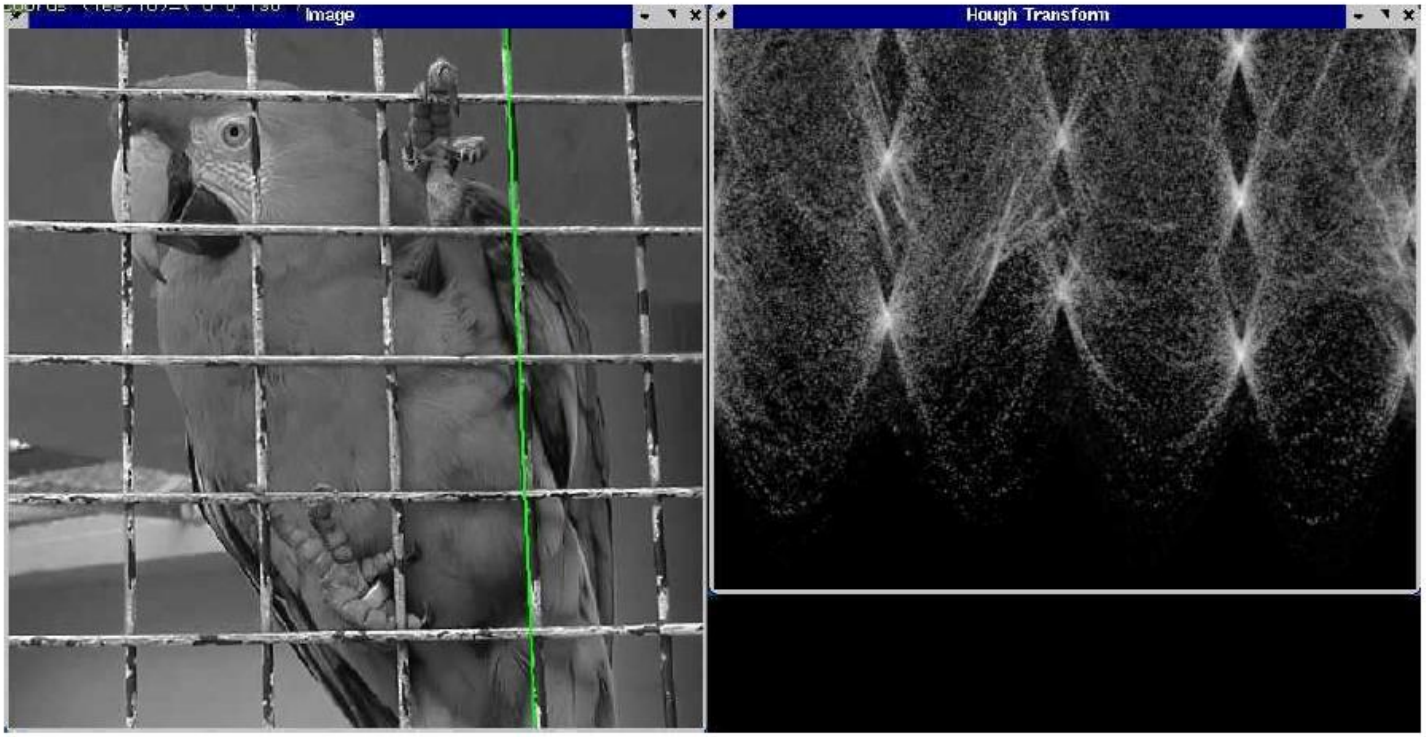


图像边缘——后处理

Find peaks and post-process



霍夫变换示例



http://ostatic.com/files/images/ss_hough.jpg

霍夫变换总结

优点

- 对异常值的适应性：每个点单独投票
- 相当高效（比尝试所有参数集快得多）
- 提供多个好拟合

缺点

- 对噪声敏感
- 容量大小受噪声容限、精度和速度/内存的限制
- 很难找到最佳点
- 不适合多个参数
- 网格大小呈指数级增长

常见应用

- 线拟合（**也包括圆、椭圆等**）
- 对象实例识别（参数为位置/刻度/方向）
- 对象类别识别（参数为位置/刻度）

7.6.3 SIFT特征

- 尺度不变的特征变换：

  SIFT — Scale Invariant Feature Transform

- 该算法提取的特征点具有：
  1. 尺度不变性
  2. 平移不变性
  3. 旋转不变性
  4. 亮度不变性

✓ 应用广泛

7.6.3 SIFT特征

- **算法步骤：**

第1步：尺度空间的**极值点检测**：

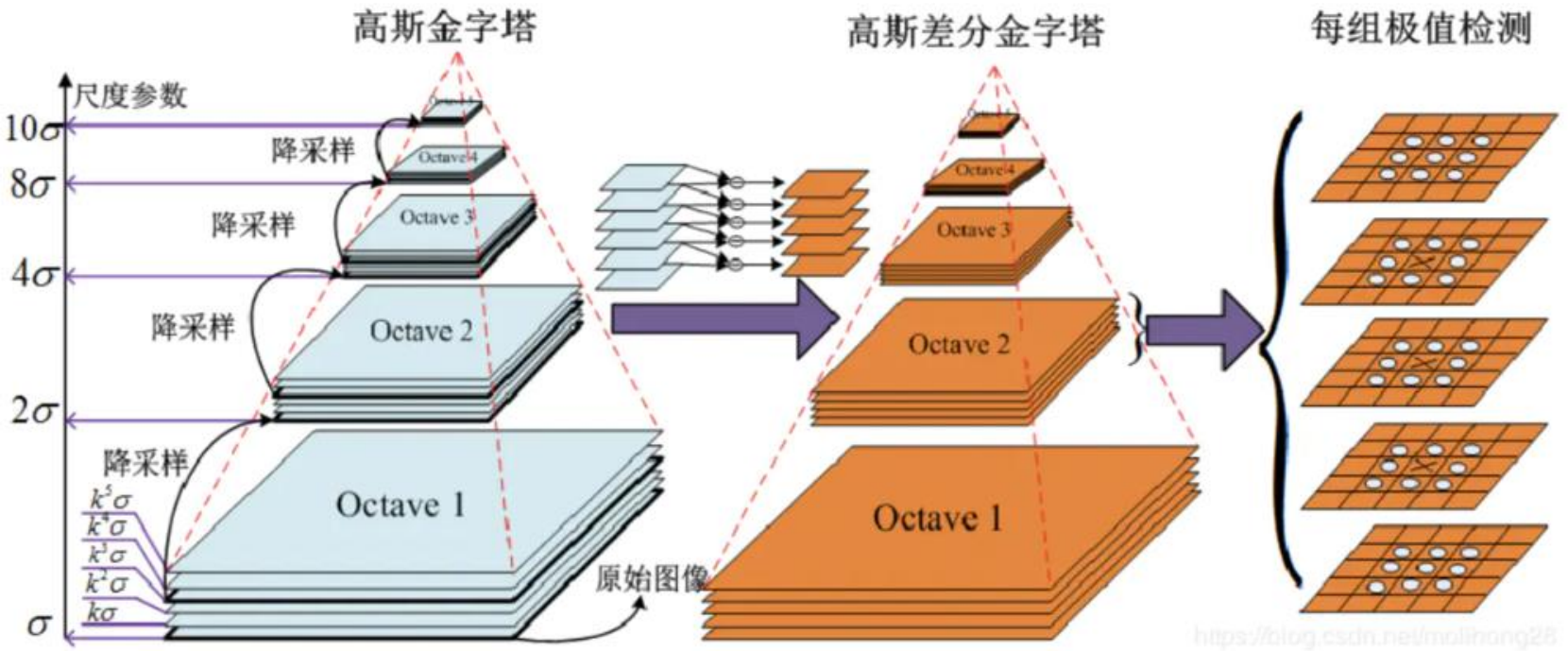
1）通过降采样，构建图像的分辨率金字塔

2）对金字塔的每一层，按照一系列递增的σ值进行高斯模糊

3）由LoG=DoG，使用高斯差分来近似计算拉普拉斯

4）提取极值点

7.6.3 SIFT特征


高斯金字塔
高斯差分金字塔
每组极值检测
尺度参数
10σ
8σ
4σ
2σ
σ
降采样
降采样
降采样
降采样
Octave4
Octave 3
Octave 2
Octave 1
$k^5\sigma$
$k^4\sigma$
$k^3\sigma$
$k^2\sigma$
$k\sigma$
原始图像

- **算法步骤：**

第2步：关键点的精准定位：

> ➢ **分析：**第1步找到的极值点是离散的——即，并不一定是的连续空间中的“真”极值点。因此：

1）根据已有的离散极值点，并使用二阶泰勒展开拟合出函数曲线，从而找到连续空间中的极值点。

2）SIFT认为图像中边缘上的点容易受到噪声的影响，因此使用以下不等式来剔除：

$$\frac{Tr(H)^2}{Det(H)} > \frac{(T_\gamma + 1)^2}{T_\gamma}$$

3）经过以上两步处理后的极值点，即关键点。

- **算法步骤：**

第3步：确定关键点的主方向：

1）主方向：即关键点处的梯度方向。
2）。

- **算法步骤：**

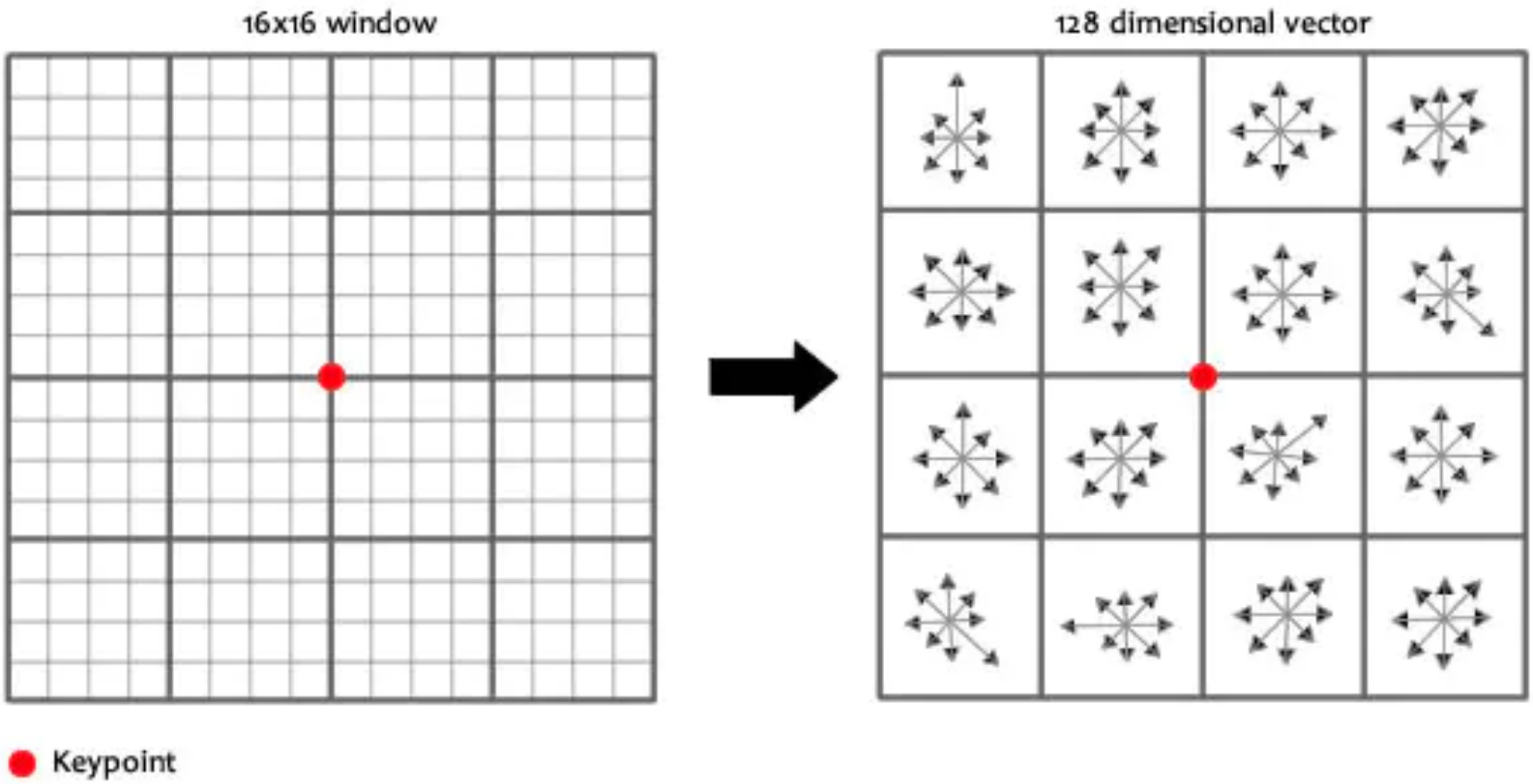
第4步：为每个关键点生成它的描述子（即，SIFT特征）

1）将关键点为中心4*4的局部邻域的X轴正方向，旋转到该关键点的主方向。

2） 为4*4邻域内的每个点，计算其8邻域方向上的梯度，从而构成4*4*8=128维的向量。

3）最后，生成描述子：（关键点，梯度方向，128维的向量）

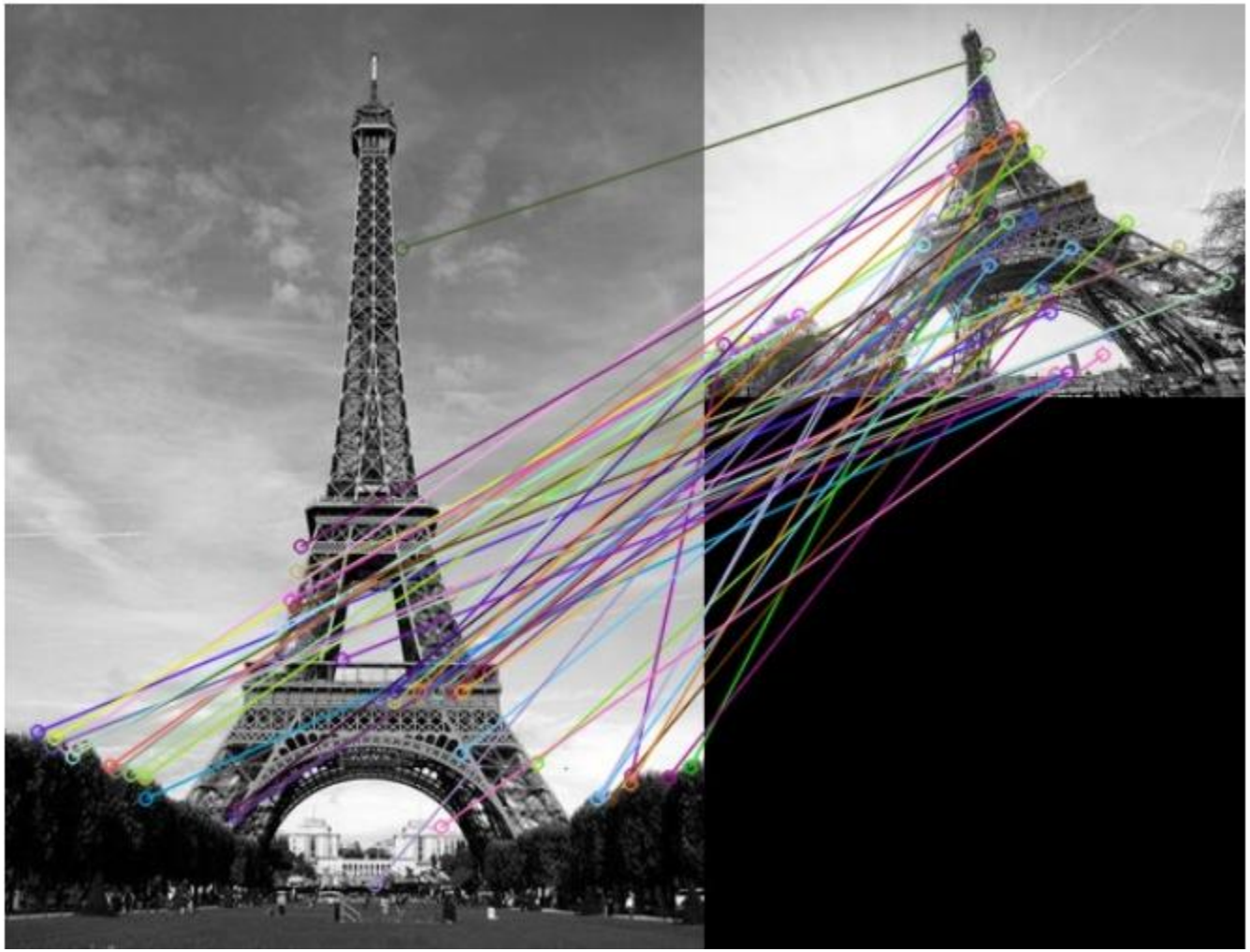
7.6.3 SIFT特征



- 算法步骤：

第5步：描述子的归一化

7.6.3 SIFT特征

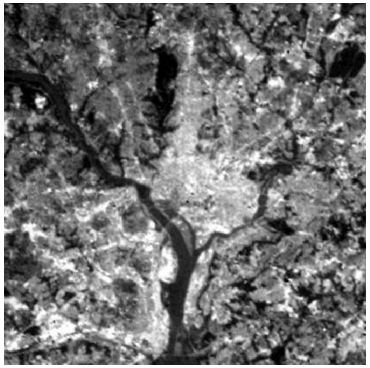


主成分分析

颜色 $= \begin{bmatrix} R \\ G \\ B \end{bmatrix}$ → 多光谱图像像素 $= \begin{bmatrix} a \\ \vdots \\ f \end{bmatrix}$ → n维向量$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$

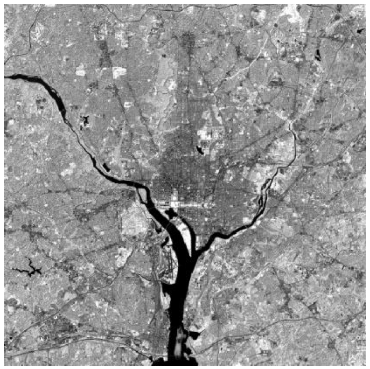
- 应用：多光谱图像

（a）蓝光

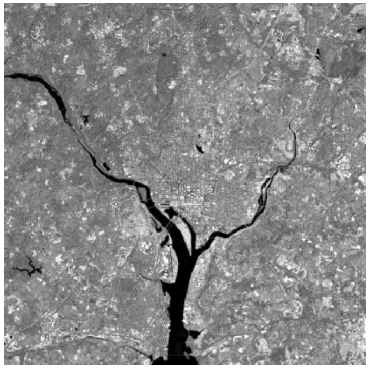


（b）绿光

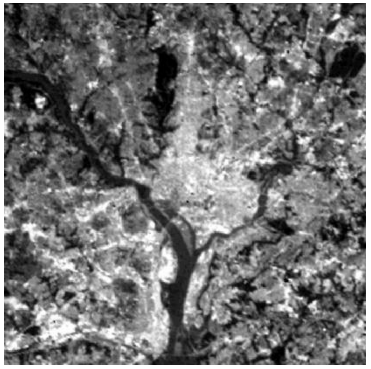
（c）红光



（d）近红外



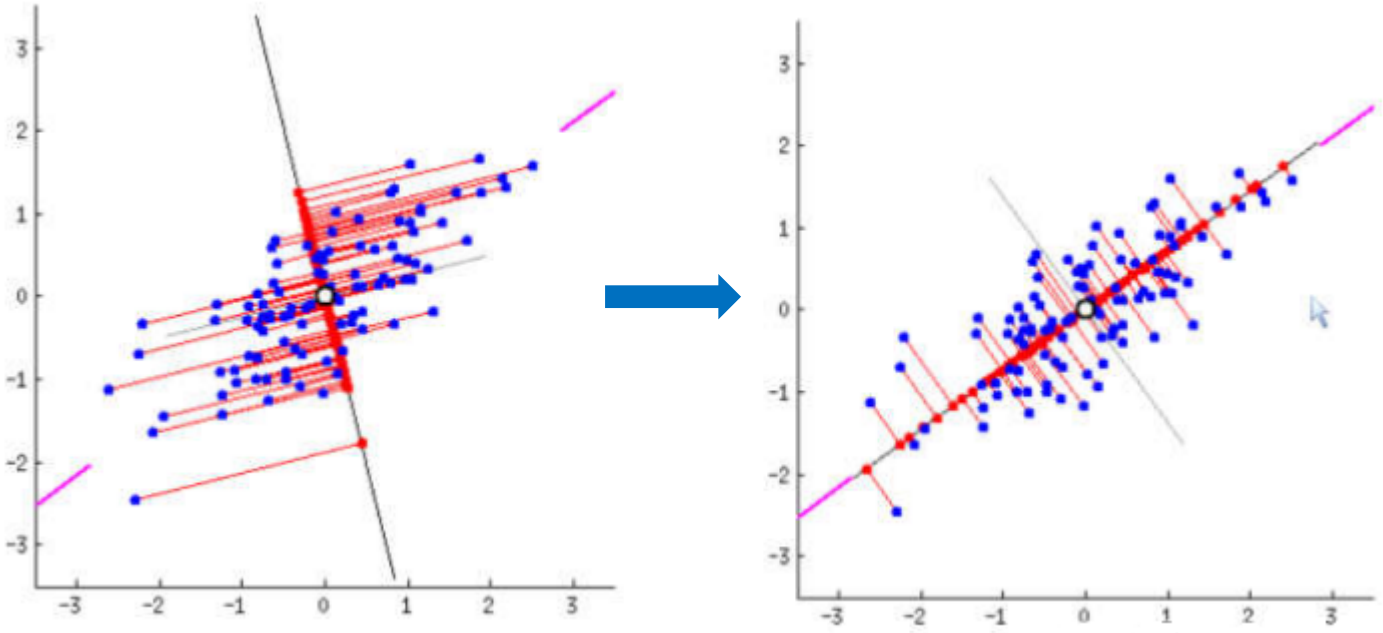
（e）中红外



（f）热红外

主成分分析

- **思想**：降低数据的维度
  - 为什么需要降维？



主成分分析

- 对降维后的数据有哪些要求：
  - 数据损失小 —> 降维前后的误差小
  - 线性不相关 —> 协方差尽可能的小
  - 位置更加分散 —> 每个向量的方差尽可能的大

向量$\boldsymbol{x}$的均值：$\boldsymbol{m_x}=\mathrm{E}\{\boldsymbol{x}\}$

协方差矩阵：$C_{\boldsymbol{x}}=\mathrm{E}\{(\boldsymbol{x}-\boldsymbol{m_x})\,(\boldsymbol{x}-\boldsymbol{m_x})^{\mathrm{T}}\}$

$$=\begin{bmatrix} \mathrm{E}\{(x_1-m_{x_1})(x_1-m_{x_1})\} & \cdots & \mathrm{E}\{(x_1-m_{x_1})(x_n-m_{x_n})\} \\ \vdots & \ddots & \vdots \\ \mathrm{E}\{(x_n-m_{x_n})(x_1-m_{x_1})\} & \cdots & \mathrm{E}\{(x_n-m_{x_n})(x_n-m_{x_n})\} \end{bmatrix}$$

主成分分析

- 坐标变换的目标：

  - 降维前后的误差小
  - ✓ 协方差尽可能的小
  - ✓ 每个向量的方差尽可能的大

✓ 由于C_x是实对称矩阵，因此它有：

1）按降序排列的特征值 $\{\lambda_i \mid i = 1, \cdots, n\}$

2）单位正交的特征向量 $\{e_i \mid i = 1, \cdots, n\}$

$$[e_i]_n^{\mathrm{T}} \cdot C_x \cdot [e_i]_n = \mathrm{A} \cdot C_x \cdot \mathrm{A}^{\mathrm{T}} = C_y = \begin{bmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_n \end{bmatrix}$$

- 将特征值作为对角元，组成矩阵 C_y。
- 将每个特征向量作为一行，组成矩阵A，显然：$\mathrm{A}^{\mathrm{T}} = \mathrm{A}^{-1}$。

- 坐标变换的公式：$\boldsymbol{y} = \mathrm{A}(\boldsymbol{x} - \boldsymbol{m}_x)$

证明：

首先，$\boldsymbol{m}_y = \mathrm{E}\{\boldsymbol{y}\} = \mathrm{E}\{\mathrm{A}\boldsymbol{x}\} - \mathrm{E}\{\mathrm{A}\boldsymbol{m}_x\} = \mathrm{A}\boldsymbol{m}_x - \mathrm{A}\boldsymbol{m}_x = 0$ 性质①

然后，协方差矩阵：

$$C_x = \mathrm{E}\{(\boldsymbol{x} - \boldsymbol{m}_x)(\boldsymbol{x} - \boldsymbol{m}_x)^{\mathrm{T}}\}$$
$$= \mathrm{E}\{\boldsymbol{x}\boldsymbol{x}^T - \boldsymbol{x}\boldsymbol{m}_x^T - \boldsymbol{m}_x\boldsymbol{x}^T - \boldsymbol{m}_x\boldsymbol{m}_x^T\}$$
$$= \mathrm{E}\{\boldsymbol{x}\boldsymbol{x}^T\} - \mathrm{E}\{\boldsymbol{x}\boldsymbol{m}_x^T\} - \mathrm{E}\{\boldsymbol{m}_x\boldsymbol{x}^T\} + \mathrm{E}\{\boldsymbol{m}_x\boldsymbol{m}_x^T\}$$
$$= \mathrm{E}\{\boldsymbol{x}\boldsymbol{x}^T\} - \boldsymbol{m}_x\boldsymbol{m}_x^T - \boldsymbol{m}_x\boldsymbol{m}_x^T + \boldsymbol{m}_x\boldsymbol{m}_x^T$$
$$= \mathrm{E}\{\boldsymbol{x}\boldsymbol{x}^T\} - \boldsymbol{m}_x\boldsymbol{m}_x^T$$

性质②

- 坐标变换的公式：$\boldsymbol{y} = \mathrm{A}(\boldsymbol{x} - \boldsymbol{m_x})$

证明：

$$C_y = \mathrm{E}\{(\boldsymbol{y} - m_y)(\boldsymbol{y} - m_y)^{\mathrm{T}}\} = \mathrm{E}\{\boldsymbol{y}\boldsymbol{y}^T\} - \boldsymbol{m_y}\boldsymbol{m_y^T} = \mathrm{E}\{\boldsymbol{y}\boldsymbol{y}^T\}$$

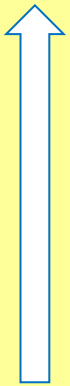
$$= \mathrm{E}\{(\mathrm{A}(\boldsymbol{x} - \boldsymbol{m_x}))(\mathrm{A}(\boldsymbol{x} - \boldsymbol{m_x}))^{\mathrm{T}}\}$$

$$= \mathrm{E}\{\mathrm{A}(\boldsymbol{x} - \boldsymbol{m_x})(\boldsymbol{x} - \boldsymbol{m_x})^{\mathrm{T}}\mathrm{A}^{\mathrm{T}}\}$$

$$= \mathrm{A} \cdot \mathrm{E}\{(\boldsymbol{x} - \boldsymbol{m_x})(\boldsymbol{x} - \boldsymbol{m_x})^{\mathrm{T}}\} \cdot \mathrm{A}^{\mathrm{T}}$$

$$= \mathrm{A} \cdot C_x \cdot \mathrm{A}^{\mathrm{T}}$$

证毕。



性质①②

- 误差分析：
  - 直接使用矩阵$\boldsymbol{A}$恢复的向量：$\boldsymbol{x} = \mathrm{A}^{\mathrm{T}}\boldsymbol{y} + m_{\boldsymbol{x}}$
  - 使用矩阵A的前K行$\mathrm{A}_{\boldsymbol{K}}^{\boldsymbol{T}}$恢复的向量：$\hat{\boldsymbol{x}} = \mathrm{A}_K^{\mathrm{T}}\boldsymbol{y} + m_{\boldsymbol{x}}$

$\boldsymbol{x}$ 和$\hat{\boldsymbol{x}}$之间的均方误差MSE（mean-square error）：

$$\mathrm{E}\{|\hat{\boldsymbol{x}} - \boldsymbol{x}|^2\} = \sum_{i=K+1}^{n} \lambda_i$$

主成分分析

■ 算法步骤

1. 构造原始数据的协方差矩阵 C_x；
2. 求出C_x的特征值λ_i及对应的特征向量e_i；
3. 将特征向量 按对应特征值大小从上到下按行 排列成矩阵，取前K行组成矩阵A；
4. $\boldsymbol{y}=A(\boldsymbol{x}-\mathbf{m}_{\boldsymbol{x}})$即为降维到$K$维后的数据。 A_K^T

- 向量均值的**离散形式**：$\boldsymbol{m}_{\boldsymbol{x}}=\frac{1}{K}\sum_{k=1}^{K}\boldsymbol{x}_k$

- 协方差矩阵的**离散形式**：$C_x=\frac{1}{K}\sum_{k=1}^{K}\boldsymbol{x}_k\boldsymbol{x}_k^T-\boldsymbol{m}_{\boldsymbol{x}}\boldsymbol{m}_{\boldsymbol{x}}^T$

主成分分析

■ 证明：$\mathrm{E}\{|\hat{\boldsymbol{x}}-\boldsymbol{x}|^2\}=\sum_{i=K+1}^{n}\lambda_i$

➢ 记$\boldsymbol{y}=(y^1;\ y^2;\ \cdots;\ y^n)$，特征向量$e_i=\left(e_{\mathrm{i}}^1;\ e_{\mathrm{i}}^2;\cdots;e_{\mathrm{i}}^n\right)$，则有：

$$\mathrm{E}\{|\hat{\boldsymbol{x}}-\boldsymbol{x}|^2\}=\mathrm{E}\left\{\left|\left(\mathbf{A}_K^{\mathrm{T}}\boldsymbol{y}+m_{\boldsymbol{x}}\right)-\left(\mathbf{A}^{\mathrm{T}}\boldsymbol{y}+m_{\boldsymbol{x}}\right)\right|^2\right\}=\mathrm{E}\{|\mathbf{A}_K^{\mathrm{T}}\boldsymbol{y}-\mathbf{A}^{\mathrm{T}}\boldsymbol{y}|^2\}$$

$$=\sum_{i=1}^{n}\mathrm{E}(e_{\mathrm{i}}^{K+1}y^{K+1}+e_{\mathrm{i}}^{K+2}y^{K+2}+\cdots e_{\mathrm{i}}^{K+n}y^{K+n})^2$$

$$=\sum_{i=1}^{n}\mathrm{E}\left\{\begin{array}{c}(e_{\mathrm{i}}^{K+1}y^{K+1})(e_{\mathrm{i}}^{K+1}y^{K+1})+\cdots+(e_{\mathrm{i}}^{K+1}y^{K+1})(e_{\mathrm{i}}^{n}y^{n})\\+\\(e_{\mathrm{i}}^{K+2}y^{K+2})(e_{\mathrm{i}}^{K+1}y^{K+1})+\cdots+(e_{\mathrm{i}}^{K+2}\,y^{K+2})(e_{\mathrm{i}}^{n}y^{n})\\+\\\cdots\\+\\(e_{\mathrm{i}}^{n}y^{n})(e_{\mathrm{i}}^{K+1}y^{K+1})+\cdots+(e_{\mathrm{i}}^{n}y^{n})(e_{\mathrm{i}}^{n}y^{n})\end{array}\right\}$$

主成分分析

■ 证明：由性质①②的结论：$\mathrm{E}\{\boldsymbol{y}\boldsymbol{y}^T\} = C_{\boldsymbol{y}} = \begin{bmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_n \end{bmatrix}$

$$\sum_{i=1}^{n} \mathrm{E}\left\{\begin{array}{c} (e_i^{K+1}y^{K+1})(e_i^{K+1}y^{K+1}) + \cdots + (e_i^{K+1}y^{K+1})(e_i^{n}y^{n}) \\ + \\ (e_i^{K+2}y^{K+2})(e_i^{K+1}y^{K+1}) + \cdots + (e_i^{K+2}y^{K+2})(e_i^{n}y^{n}) \\ + \\ \cdots \\ + \\ (e_i^{n}y^{n})(e_i^{K+1}y^{K+1}) + \cdots + (e_i^{n}y^{n})(e_i^{n}y^{n}) \end{array}\right\}$$

$$= \sum_{i=1}^{n}\left\{\mathrm{E}(e_i^{K+1}y^{K+1})^2 + \cdots + \mathrm{E}(e_i^{n}y^{n})^2\right\}$$

$$= \sum_{i=1}^{n}\left\{(e_i^{K+1})^2\mathrm{E}(y^{K+1})^2 + \cdots + (e_i^{n})^2\mathrm{E}(y^{n})^2\right\}$$

$$= \sum_{i=1}^{n}(e_i^{K+1})^2\lambda_{K+1} + \cdots + \sum_{i=1}^{n}(e_i^{n})^2\lambda_n$$

主成分分析

■ 证明：

构造相似变换：$A\left(A^{T}\begin{bmatrix}\lambda_1 & \cdots & 0\\ \vdots & \ddots & \vdots\\ 0 & \cdots & \lambda_n\end{bmatrix}A\right)A^{T}$

$$\begin{bmatrix}\lambda_1 & \cdots & 0\end{bmatrix}$$

$$A\left([e_i]_n\begin{bmatrix}\lambda_1 & \cdots & 0\\ \vdots & \ddots & \vdots\\ 0 & \cdots & \lambda_n\end{bmatrix}[e_i]_n^{T}\right)A^{T} \sim \begin{bmatrix}\lambda_1 & \cdots & 0\\ \vdots & \ddots & \vdots\\ 0 & \cdots & \lambda_n\end{bmatrix}$$

- 上式中，圆括号内矩阵的对角元依次为：

$$\sum_{i=1}^{n}(e_i^1)^2\lambda_i, \sum_{i=1}^{n}(e_i^2)^2\lambda_i, \cdots, \sum_{i=1}^{n}(e_i^n)^2\lambda_i$$

- 由于相似变换不改变矩阵的迹，则有：

$$\sum_{i=1}^{n}(e_i^1)^2\lambda_1 + \cdots + \sum_{i=1}^{n}(e_i^n)^2\lambda_n = \sum_{i=1}^{n}\lambda_i$$

- 同理，在构造相似变换时，～两侧的$\lambda_1, \ldots, \lambda_K$都替换为0，即得：

$$\sum_{i=1}^{n}(e_i^{K+1})^2\lambda_{K+1} + \cdots + \sum_{i=1}^{n}(e_i^{K+n})^2\lambda_n = \sum_{i=K+1}^{n}\lambda_i$$

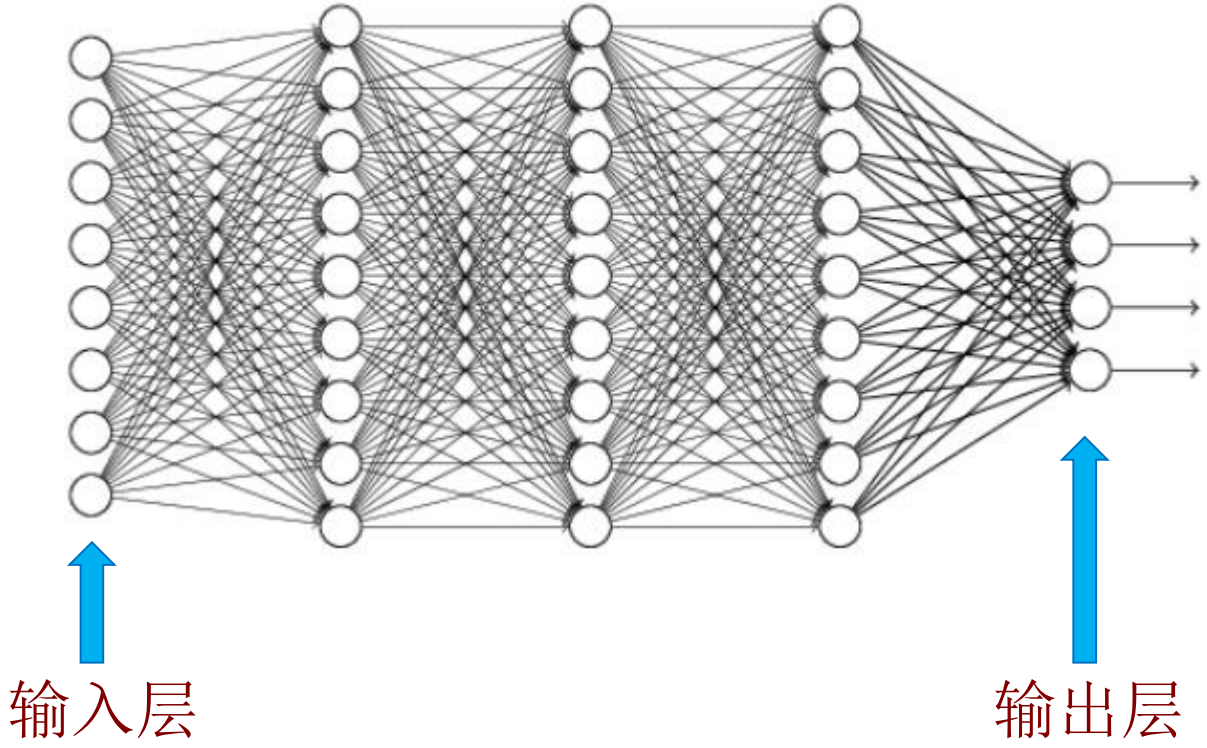
主成分分析

- 应用举例

7.7 计算机视觉—简介

一、传统方法：数字图像处理，模式识别

二、现代方法：基于深度神经网络的机器学习



7.7 计算机视觉—简介

- 传统与现代方法的思想类似：

- 问题建模：

  目标函数（Objective Function, in Math）

  能量函数（Energy Function, in Image Segmentation）

  损失函数（Loss Function, in Computer Vision）

- 求解算法

  解析表达式：例如，维纳滤波器模板

  迭代算法：例如，主动轮廓模型

  深度学习：训练网络参数

7.7 计算机视觉——简介

- 经典图像处理算法**在计算机视觉中的作用**：
  - 设计损失函数
  - 神经网络结构的可解释性
  - 数据集的构造、预处理、训练算法

7.7 计算机视觉——简介

- 计算机视觉的应用举例：