中国科学技术大学六系研究生课程《数字图像分析》

第六章：图像分割

授课老师：李厚强，胡洋，周文罡，李礼

图像分割的定义

- 令集合 R 代表整个图像区域，对 R 的分割可看做将 R分成若干个满足下述条件的非空的子集（子区域）$R_1, R_2, \cdots, R_n$ ：
  - $\cup_{i=1}^{n} R_i = R$
  - 对$\forall i \neq j$,有$R_i \cap R_j = \emptyset$
  - 每个子区域R_i都是联通的；
  - 对于各个子区域，有均匀性测度度量$P(\cdot)$为真；但对其中任意两个和两个以上相邻子区域之并，其均匀性测度度量P为假（避免过度分割），即：

$$P(R_i) = \text{TRUE}, \text{且 } P(R_i \cup R_j) = \text{FALSE}$$

图像分割

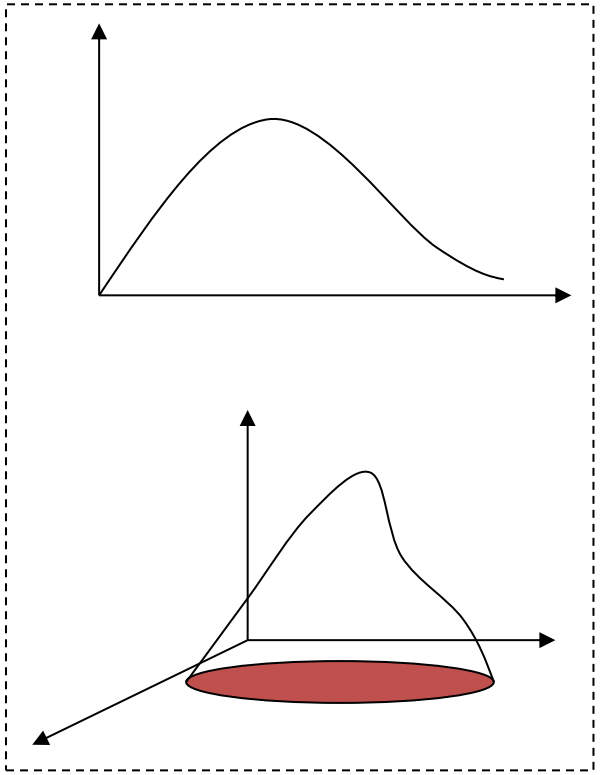
- 阈值分割
- 区域生长法
- 分裂合并方法
- 分水岭算法
- 聚类分割算法
- 主动轮廓分割
- Graph Cut
- Normalized Cut
- Mean Shift

阈值分割 (Thresholding)

- 取阈值法是以图像**直方图**为依据，选定阈值，再逐个对像素作判决

- 图像直方图可以是：
  - 单个特征的一维直方图
    - ✓ 如灰度直方图
  - 多个特征的多维直方图
    - ✓ 如两个波段组成的二维直方图

- **特征**可以是灰度，也可以是其他值



原理和分类

- 图像模型：
  - 图像由具有单峰测度(灰度, 颜色, 纹理)分布的目标和背景组成
    - ✓ 目标：Object
  - 目标或背景内部的相邻像素间的测度值高度相关
    - ✓ 均匀性测度度量为真
  - 目标和背景交界处两边的像素测度值差异大



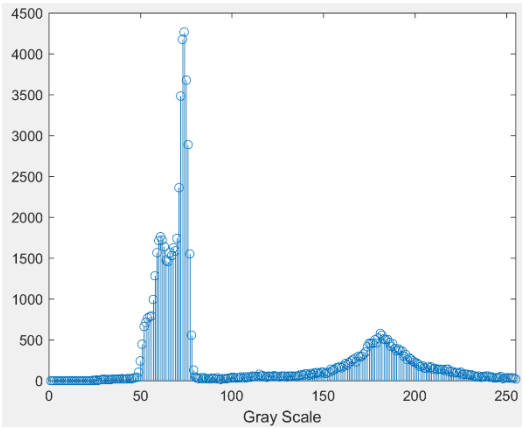
原理和分类

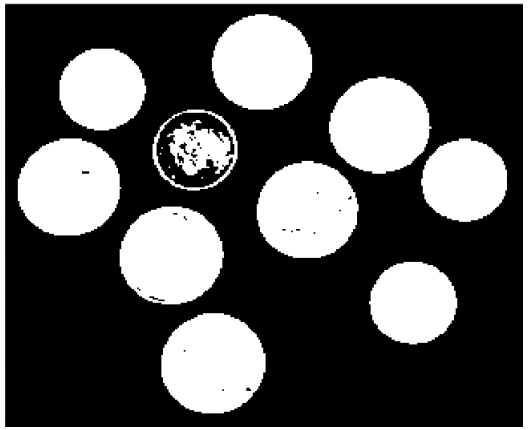
单阈值分割图像

- 对灰度图（取值在gmin和gmax之间）确定一个灰度阈值T
- （gmin < T < gmax）

$$g(x,y)=\begin{cases}1 & \text{如 } f(x,y) > T \\ 0 & \text{如 } f(x,y) \leqslant T\end{cases}$$

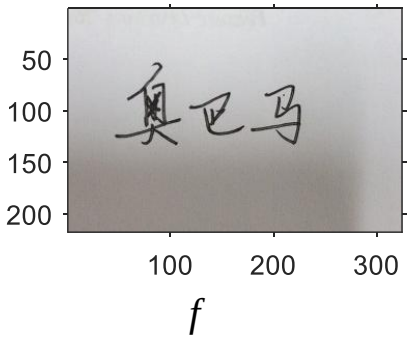


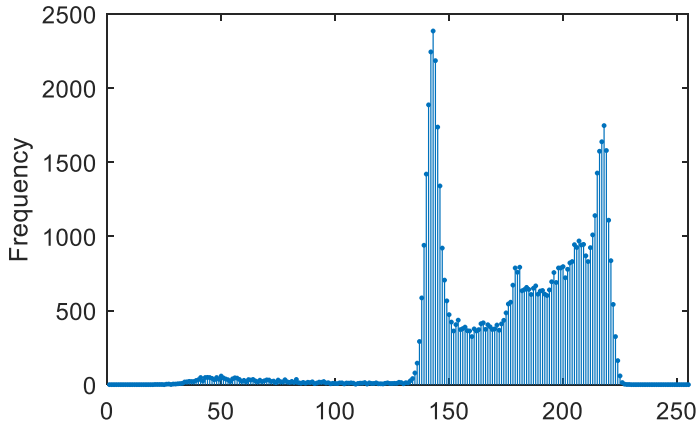


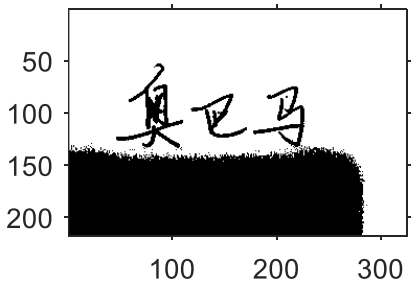


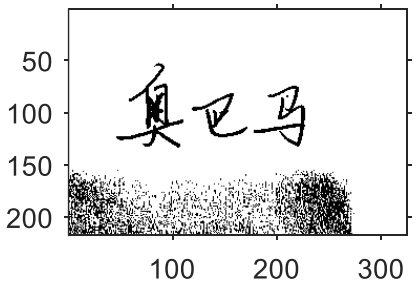
原理和分类

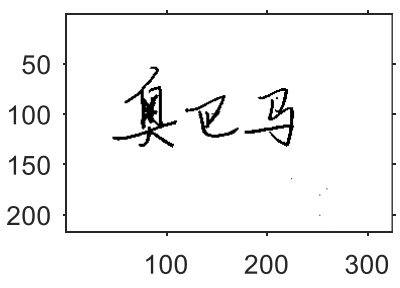
- 实例：生成电子签名
  - 课后作业：做一份自己的电子签名

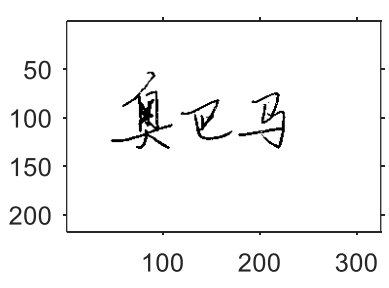








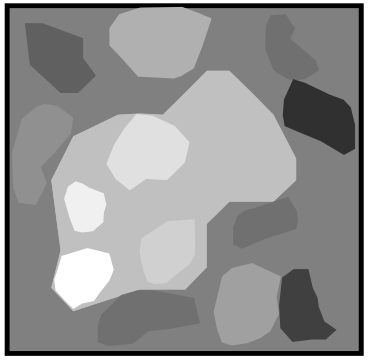




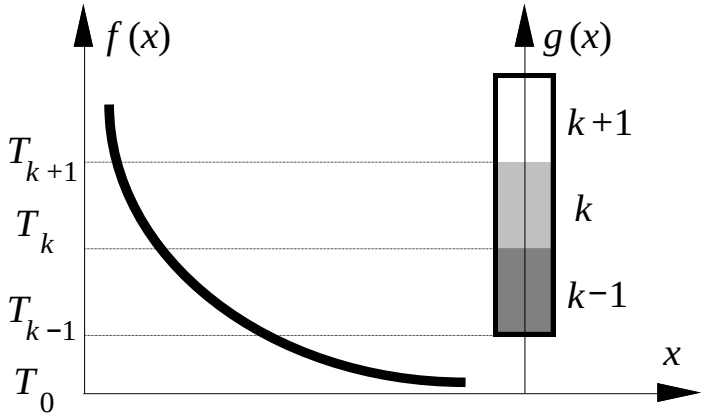
原理和分类

- 多阈值分割图像
  - 确定一系列分割阈值

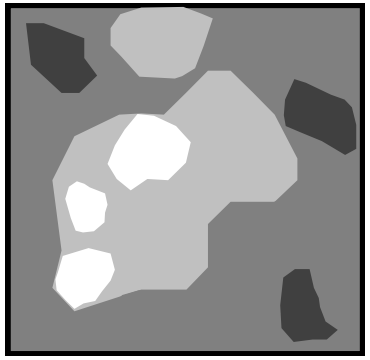
$$g(x,y)=k \qquad \text{如} \quad T_k < f(x,y) \leqslant T_{k+1} \qquad k=0,1,2,\cdots,K$$



(a)



(b)



(c)

Otsu's 方法 (1979)

- 基本思想
  - 假设图像像素可分为两类，选择分割阈值，使得类内方差最小。
- 形式化表达
  - 给定一个图像I，计算其归一化的直方图$P(i)$
    - ✓ 归一化：直方图向量中所有元素值之和为1
  - 选择阈值t，将像素分为两类，每类概率：

$$q_1(t)=\sum_{i=1}^{t}P(i) \qquad q_2(t)=\sum_{i=t+1}^{T}P(i)$$

  - 每类像素的灰度均值：

$$\mu_1(t)=\sum_{i=1}^{t}\frac{i\cdot P(i)}{q_1(t)} \qquad \mu_2(t)=\sum_{i=t+1}^{T}\frac{i\cdot P(i)}{q_2(t)}$$

- Otsu, N., "A Threshold Selection Method from Gray-Level Histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 9, No. 1, 1979, pp. 62-66.

Otsu's 方法

形式化表达

每类像素的灰度方差：

$$\sigma_1^2(t)=\sum_{i=1}^{t}[i-\mu_1(t)]^2\frac{P(i)}{q_1(t)} \qquad \sigma_2^2(t)=\sum_{i=t+1}^{T}[i-\mu_2(t)]^2\frac{P(i)}{q_2(t)}$$

- 加权类内方差：

$$\sigma_w^2(t)=q_1(t)\sigma_1^2(t)+q_2(t)\sigma_2^2(t)$$

- 最优阈值选择（穷举法）：

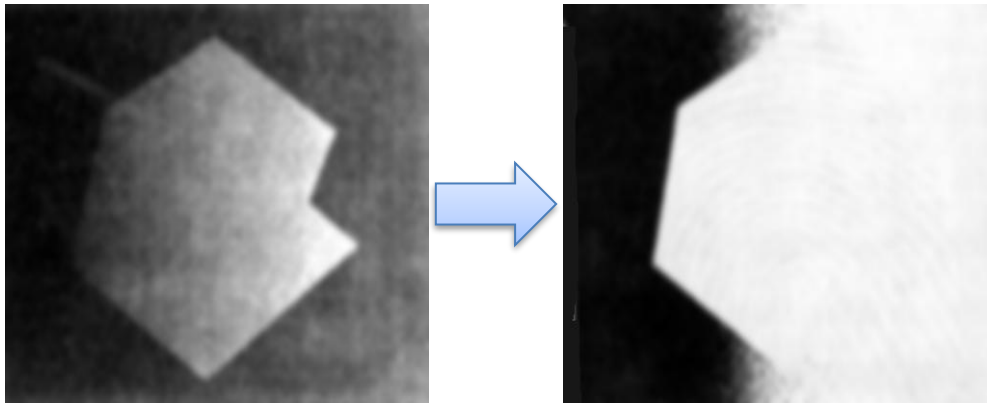
$$\tau=\arg\min_t \sigma_w^2(t)$$

- 二值化图像分割：

$$B(i,j)=\begin{cases}1 & I(i,j)\geq\tau\\ 0 & I(i,j)<\tau\end{cases}$$

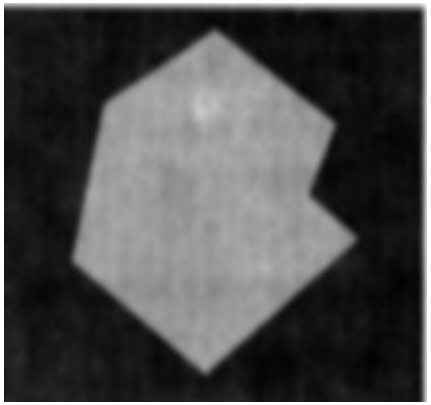
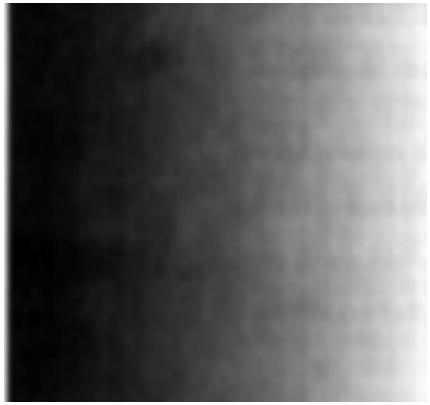
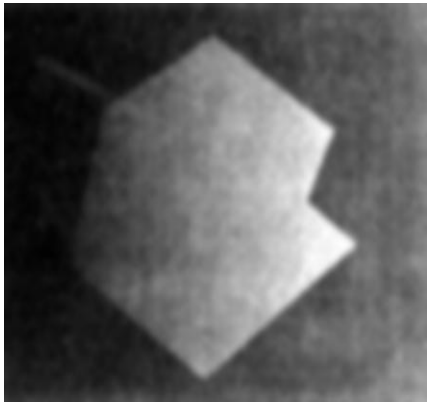
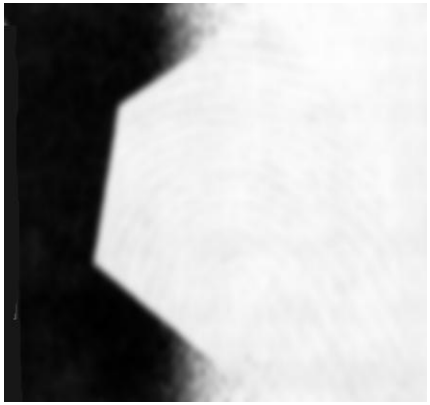
依赖坐标的阈值选取

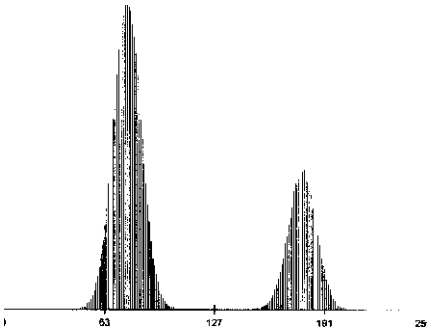
- 全局阈值不能兼顾图像各处的情况
  - 解决方案：用与坐标相关的一系列阈值来对图像分割
  - 阈值除根据f (x, y)和p(x, y)有关，还与x, y有关

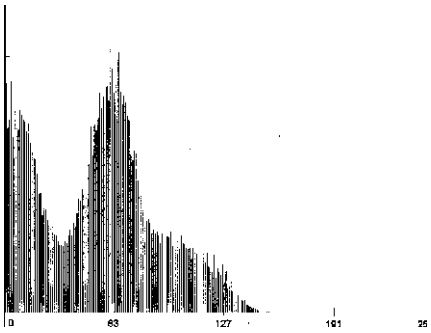


- 基本思想/思路： 为每个像素计算一个分割阈值
  - 将图像分解成一系列子图像
  - 对每个子图像计算一个阈值
  - 用这些子图像阈值，对每个像素的阈值进行插值
  - 用插值结果（阈值曲面）进行分割

光照不均匀对分割的影响

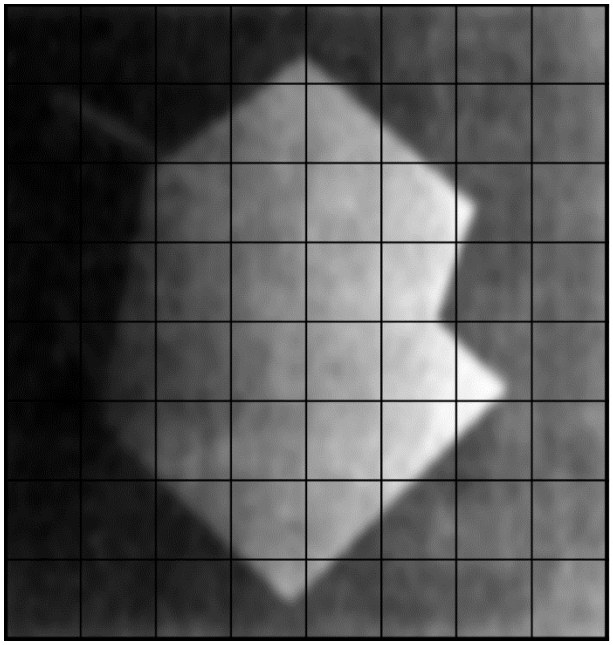







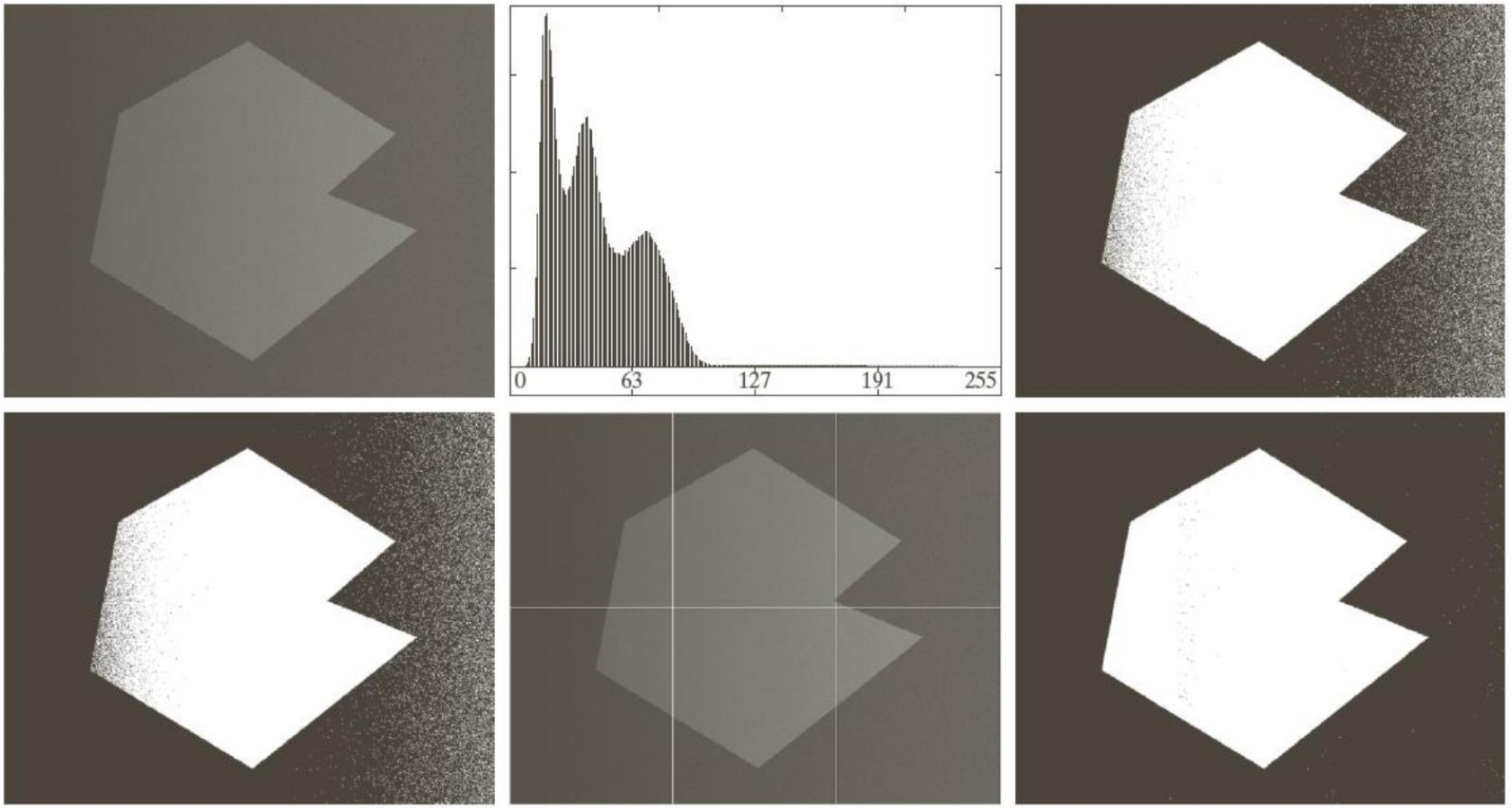


变化阈值法

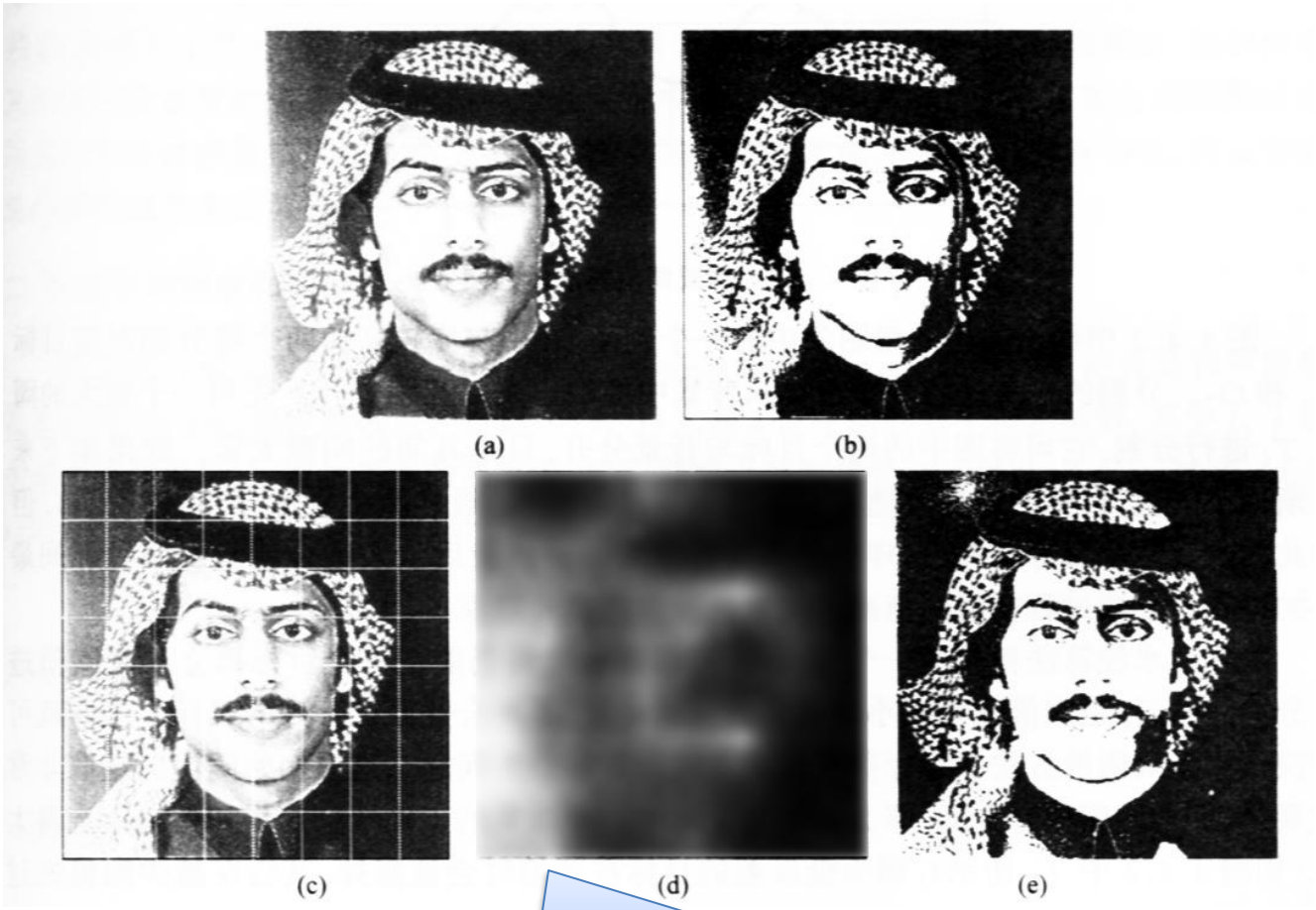
1. 将整幅图像分成一系列互相之间有50%重叠的子图像
   - 如何确定子图像的大小？
2. 做出每个子图像的直方图
3. 检测各个子图像的直方图是否为双峰的。如是，则采用最优阈值法确定一个阈值；否则，不进行处理
   - 如何判断双峰分布？方差
4. 根据对直方图为双峰的子图像得到的阈值，通过插值得到所有子图像的阈值
5. 根据各子图像的阈值，再通过插值得到所有像素的阈值，然后对图像进行分割



依赖坐标的阈值选取



分块取阈值一例



阈值图像，每个像素都有一个对应阈值

区域生长法

- 基本思想：
  - 均匀性度量准则：将相似像素组合起来构成区域
- 基本步骤：
  1. 选择区域的种子像素
  2. 确定将相邻像素包括进来的准则
  3. 制定生长停止的规则
- 讨论：
  1. 种子像素的选取
     - ✓ 聚类中心
     - ✓ 交互选取
  2. 生长准则（依赖具体应用）

区域生长法

生长示例

1. 根据直方图选取聚类中心的象素为种子
2. 根据与种子象素灰度差（<*T*）判断是否生长
3. 根据图象边缘确定生长何时终结

原始图

1	0	4	7	5
1	0	4	7	7
0	**1**	5	**5**	5
3	0	5	6	5
3	3	5	6	4

$T = 3$

1	1	5	5	5
1	1	5	5	5
1	1	5	5	5
1	1	5	5	5
1	1	5	5	5

$T = 2$

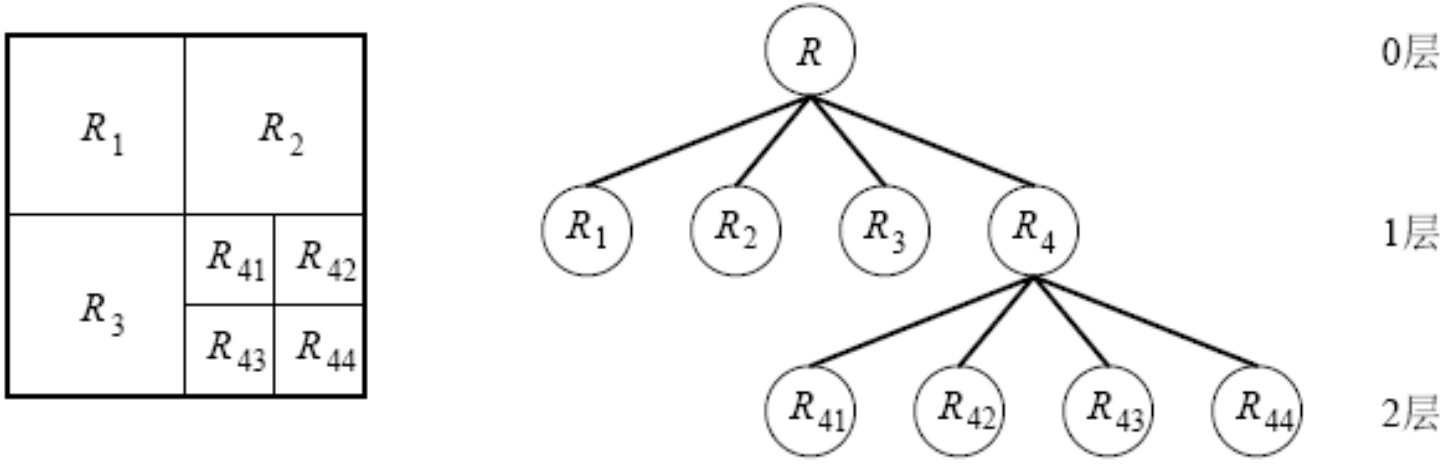
1	1	5	7	5
1	1	5	7	7
1	1	5	5	5
3	1	5	5	5
3	3	5	5	5

$T = 7$

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

分裂合并

- 主要步骤
  - 先把图像分成任意大小且不重叠的区域
  - 然后再（根据准则）合并或分裂这些区域
  - 迭代进行直到实现分割



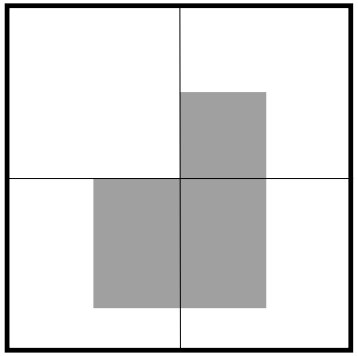
图像的四叉树表达法

分裂合并

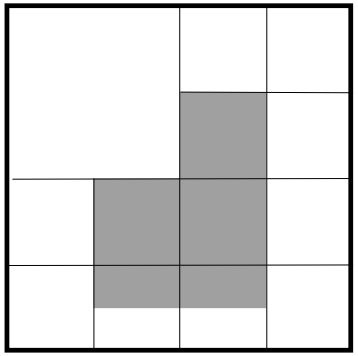
- 令R代表整个图像区域，P代表逻辑谓词
- 把R连续地分裂成越来越小的1/4的正方形子区域Ri，并且始终使 $P(R_i) = \text{TRUE}$
  1. 对任一个区域R_i，如果$P(R_i) = \text{FALSE}$，就将其分裂成不重叠的四等分
  2. 对相邻的两个区域R_i和R_j，如果$P(R_i \cup R_j) = \text{TRUE}$，就将它们合并起来
  3. 如果进一步的分裂或合并都不可能了，则结束

分裂合并

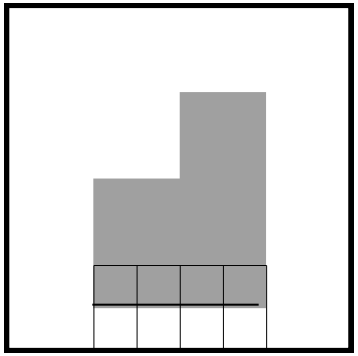
示例（四叉树）：分裂　　分裂　　合并



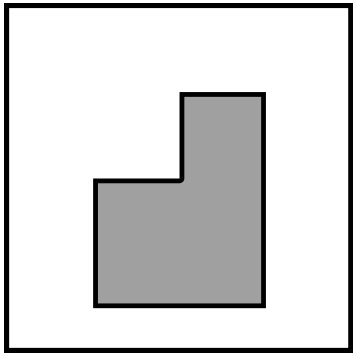
(a)



(b)



(c)



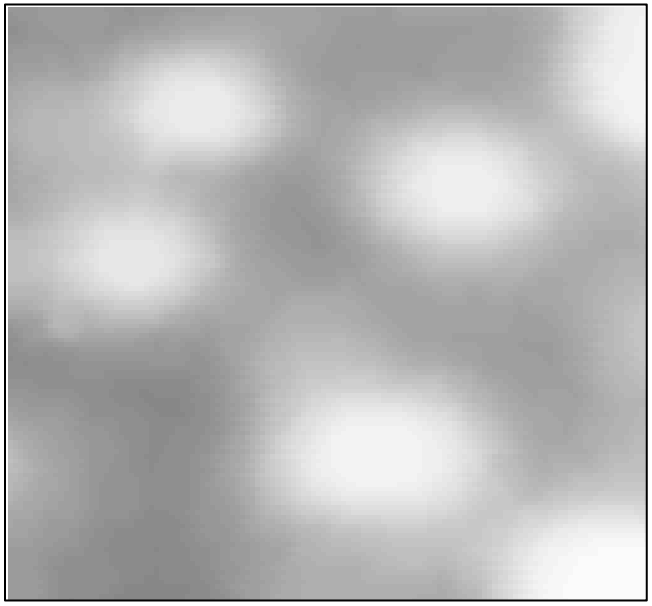
(d)

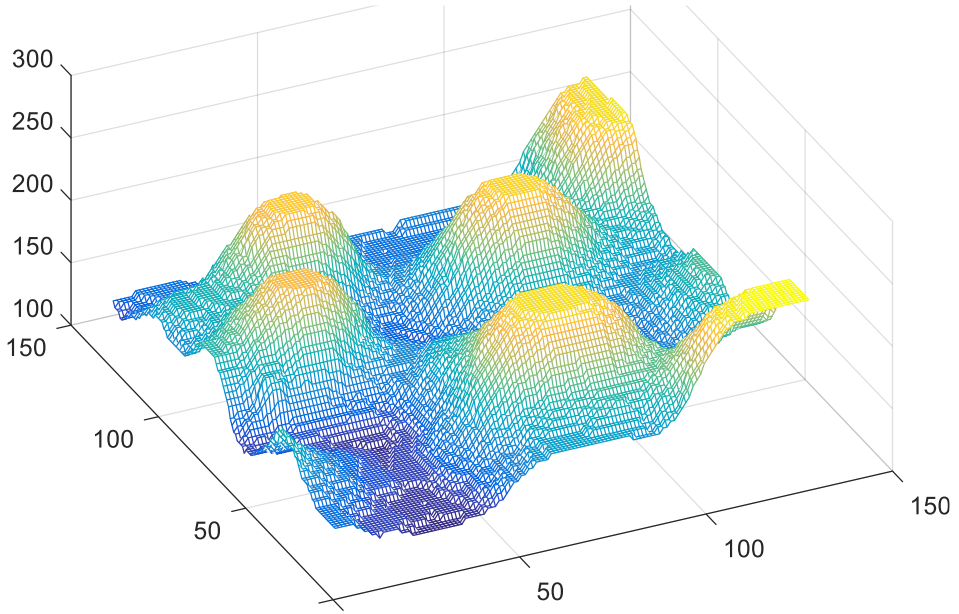
“分-合”图像分割方法

- 确定均匀性测度，构造四分树结构
- 选择初始分割层（一般为中间某一层）
- 分裂处理
  - 从中间层开始，计算各块均匀性测度。对于均匀性测度为假的那些块，一分为四，重新编码。重复进行，直到各块的均匀性测度为真。
- 合并处理
  - 从同一中间层开始，测试同属于一个父节点的四块，如果它们之并的均匀性测度为真，则合并这四块为一块。重复进行，直至不再存在可以合并的那些块。
- 组合处理
  - 使用该数据编码判断位置，对相邻的大小不一，或者虽然大小一样，但不能合并为一个父节点的区域，进行均匀性测度测试，合并均匀测度度量为真的一对区域。反复重复这一运算，直到不再存在可以合并的区域。
- 小区处理
  - 清除小区等整理工作。

分水岭分割算法

- 分水岭（watershed，也称分水线/水线）
- 把图象看成3-D地形的表示，即2-D的地基（对应图像空间）加上第3维的高度（对应图像灰度）
- 图像的**梯度图**也可以视为3D地形
  - 图像梯度图3D地形中的**峰岭**对应目标的边界

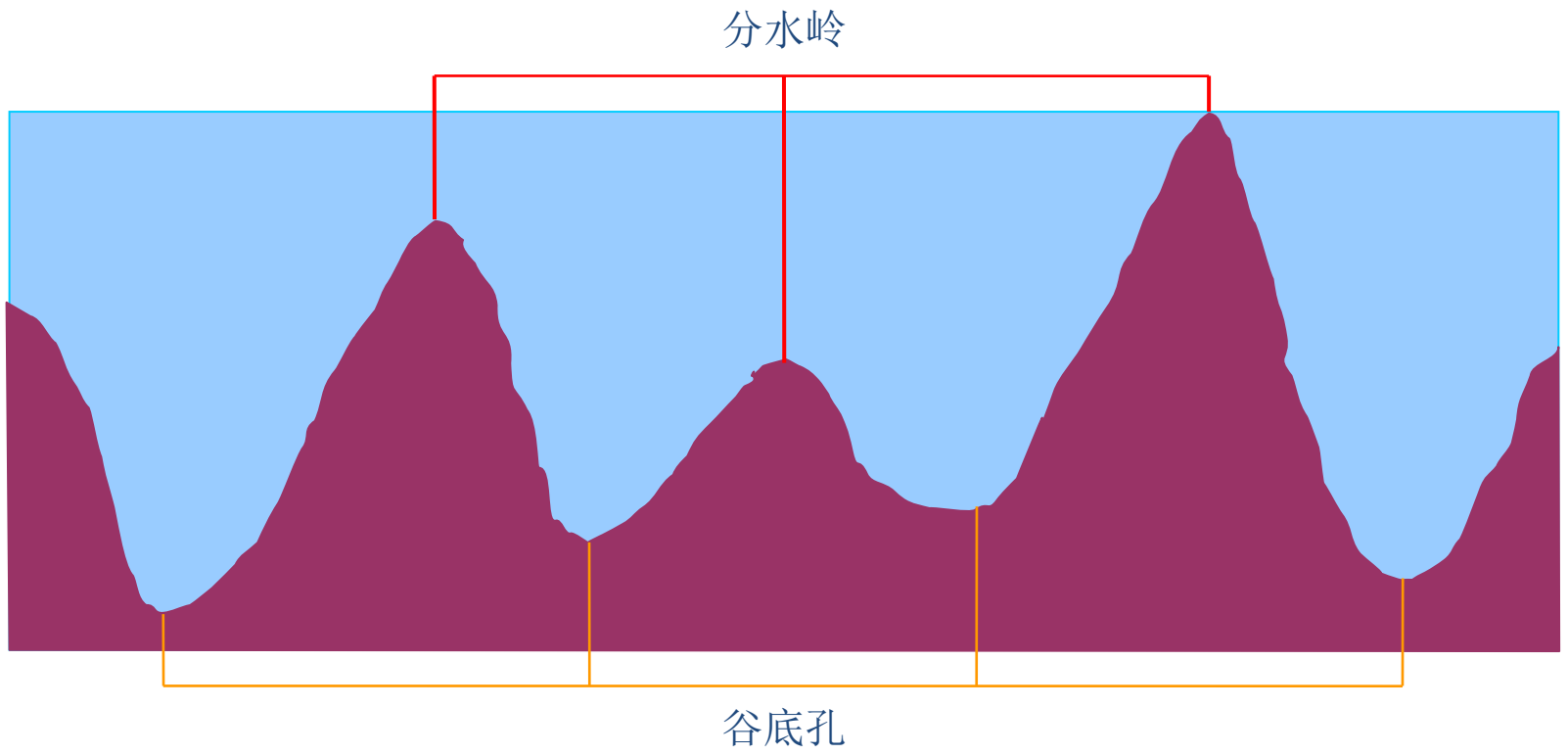




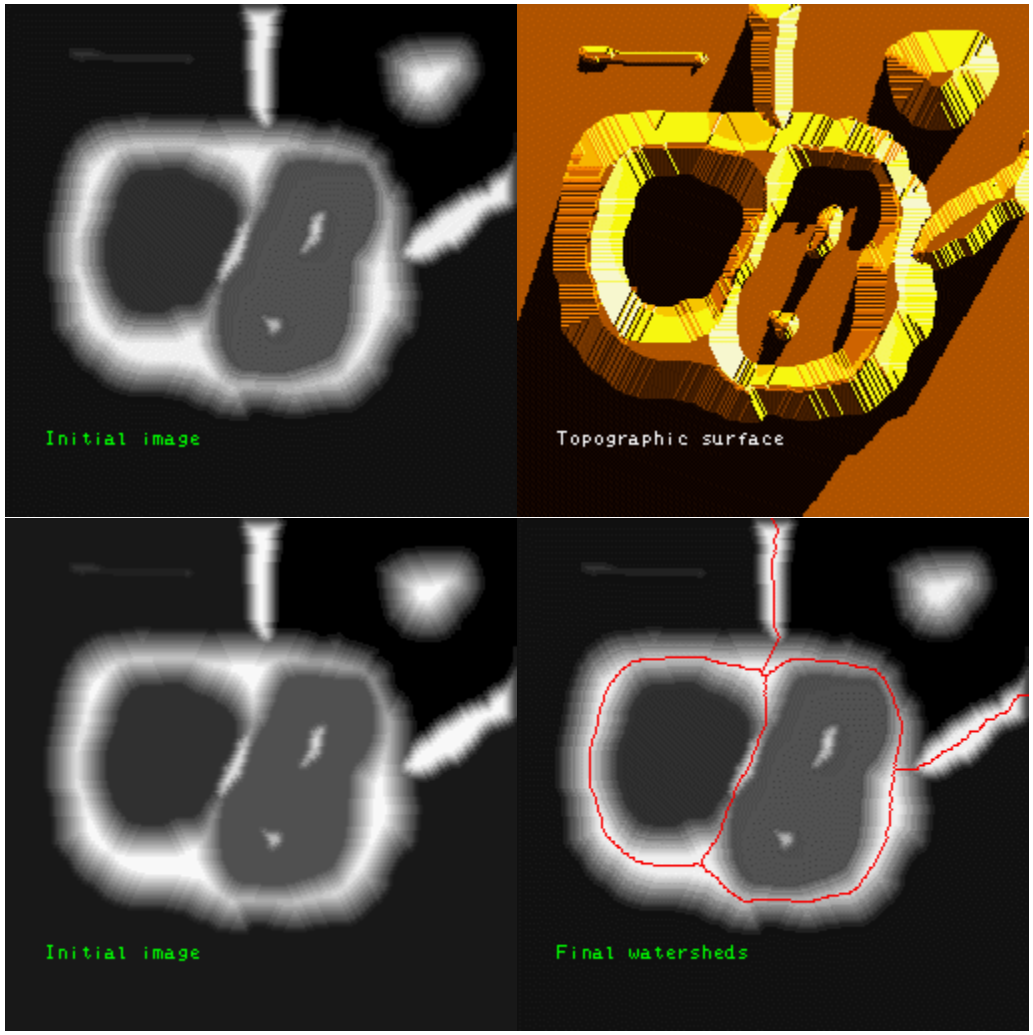
基本原理和步骤

分水岭算法原理

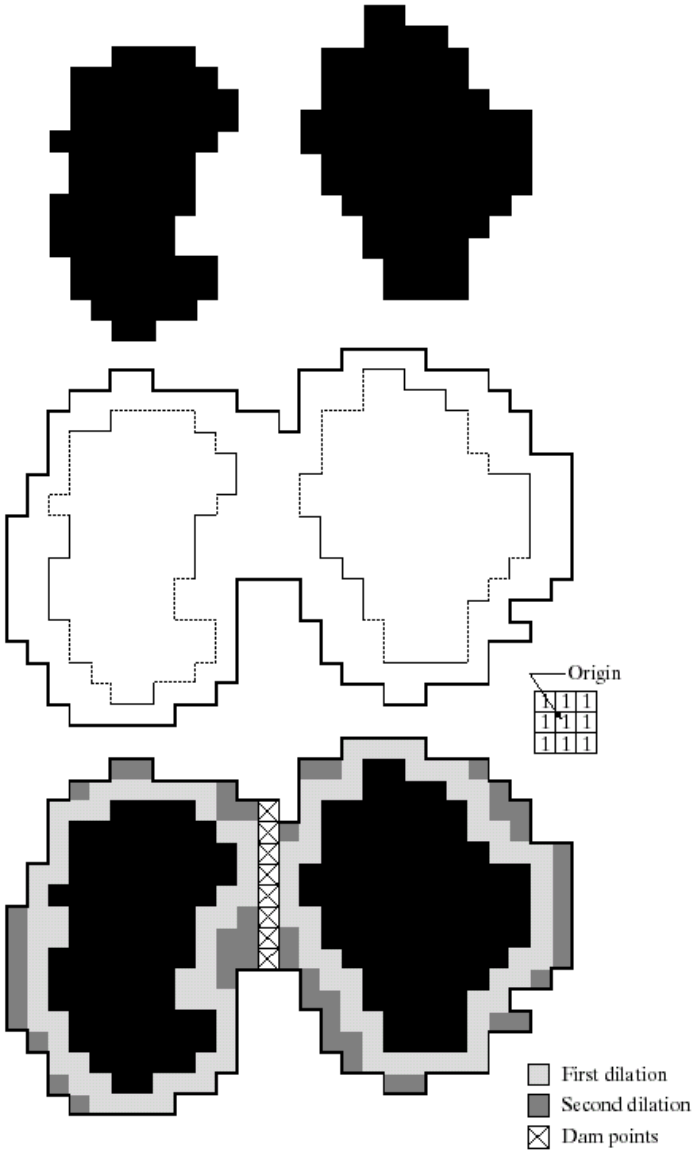
- 下图横轴表示像素坐标，纵轴表示像素灰度的梯度幅值
- 建立不同目标间的分水岭



分水岭算法过程演示


Initial image
Topographic surface
Initial image
Final watersheds

分水岭算法-构筑水坝



基本原理和步骤

分水岭计算步骤

1. 待分割图像$f(x, y)$，其梯度图像为$g(x, y)$
2. 用$M_1, M_2, \ldots, M_R$表示$g(x, y)$中各**局部极小值**的像素位置，$C(M_i)$为与M_i对应的区域中的像素坐标集合（即期望的最终分割结果）
3. 用 n 表示当前梯度阈值，$T[n]$ 代表记为 (u, v) 的像素集合，要求 $g(u, v) < n$；$T[n]$可对应多个联通体：

$$T[n] = \{(u, v) \mid g(u, v) < n\}$$

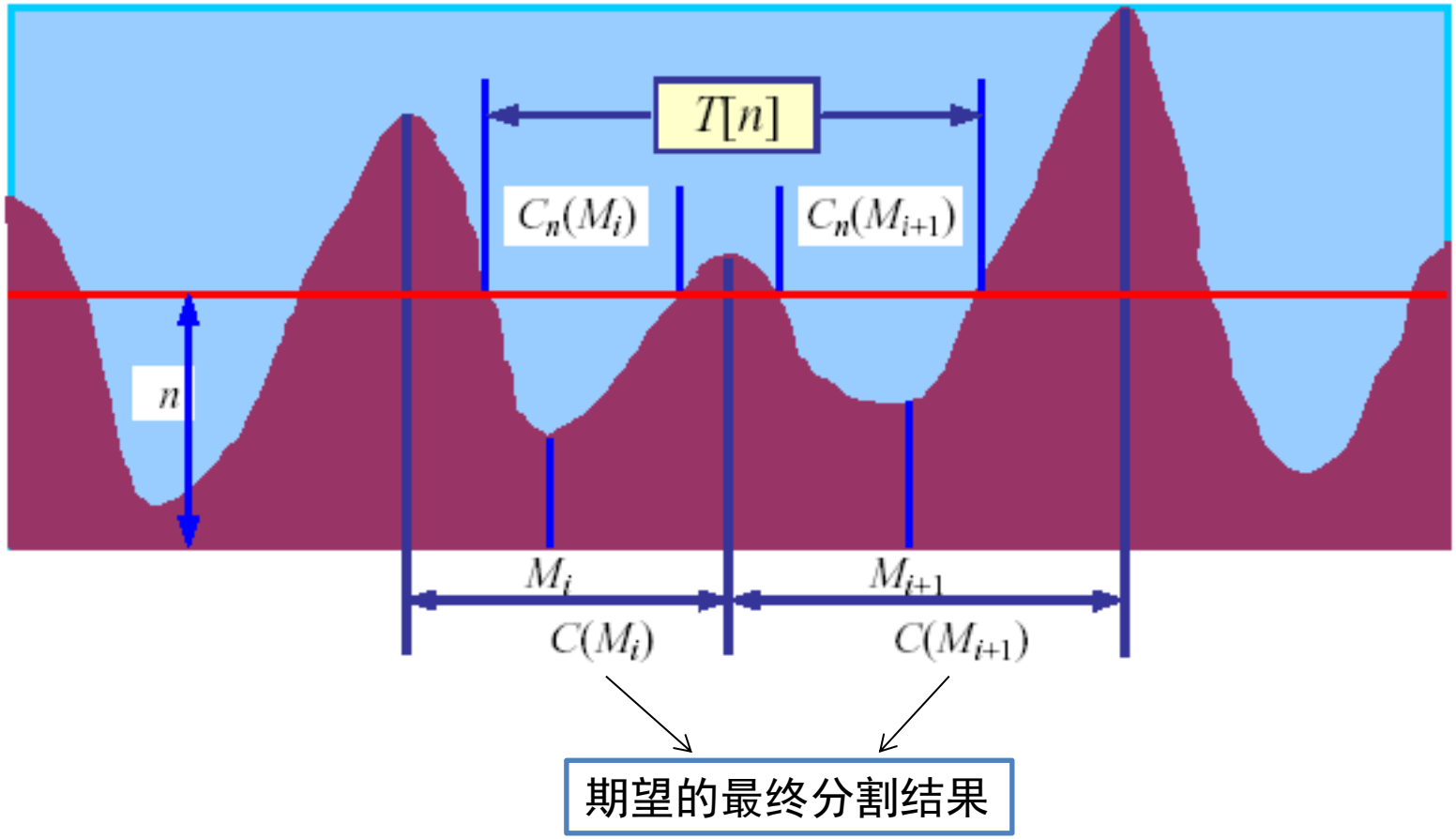
4. 对M_i所在的区域，其中满足条件的坐标集合$C_n(M_i)$可看作一幅二值图像：

$$C_n(M_i) = C(M_i) \cap T[n]$$

基本原理和步骤

分水岭计算步骤

$$C_n(M_i) = C(M_i) \cap T[n]$$



基本原理和步骤

分水岭计算步骤

- 用$C[n]$代表：在梯度阈值为n时，图象中所有满足条件的象素：

$$C[n] = \bigcup_{i=1}^{R} C_n(M_i)$$

- $C[\max + 1]$将是所有区域的并集：

$$C[max + 1] = \bigcup_{i=1}^{R} C_{max+1}(M_i)$$

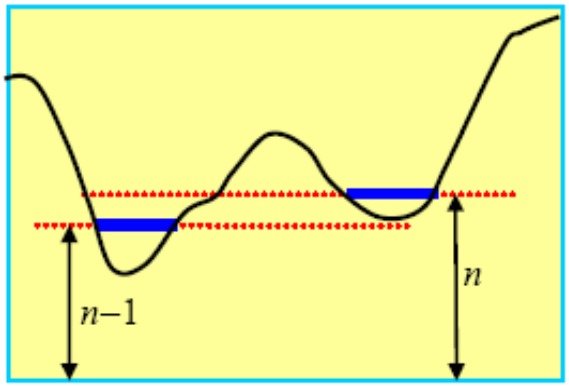
- $C[n-1]$是$C[n]$的子集， 也是$T[n]$的子集

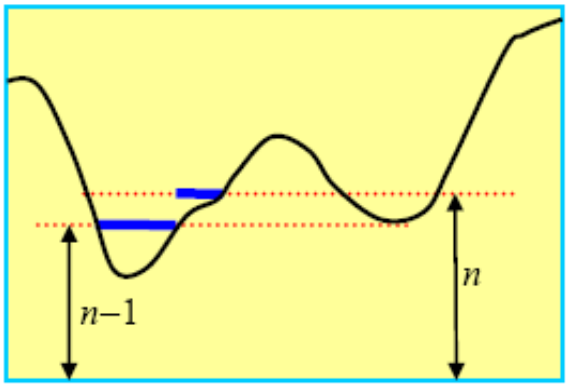
基本原理和步骤

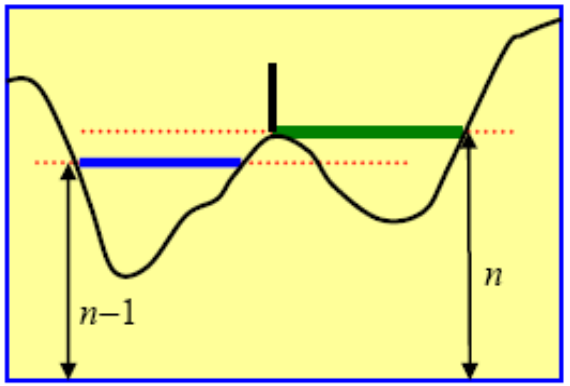
分水岭计算步骤

- 令 S 代表$T[n]$中的连通组元集合，对每个连通组元$s \in S[n]$，有3种可能性：

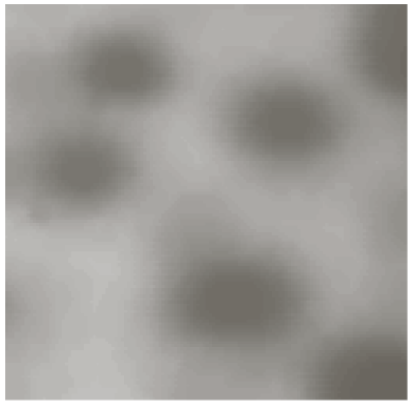
1. $s \cap C[n-1]$是1个空集
   - ✓ $C[n]$可由把连通组元 s 加到$C[n-1]$中得到
2. $s \cap C[n-1]$里包含$C[n-1]$中的一个连通组元
   - ✓ $C[n]$可由把连通组元 s 加到$C[n-1]$中得到
3. $s \cap C[n-1]$里包含$C[n-1]$中一个以上的连通组元
   - ✓ 需要在s中建分水岭



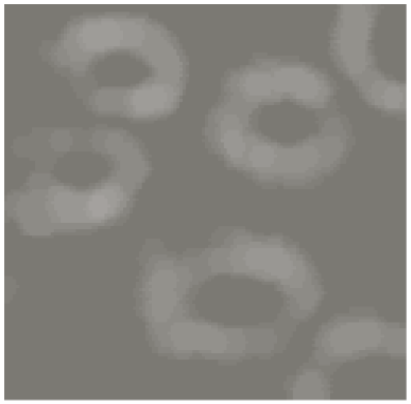




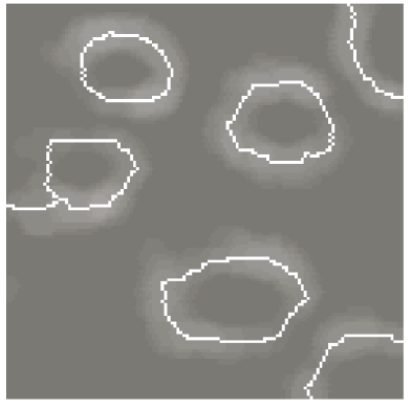
分水岭分割算法实例



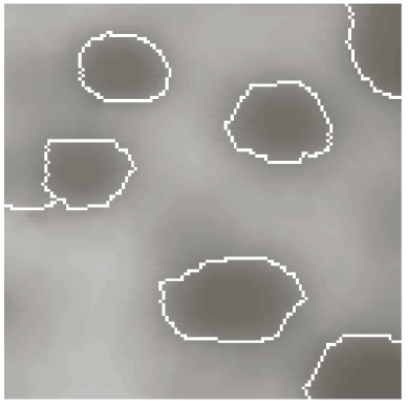
原始图



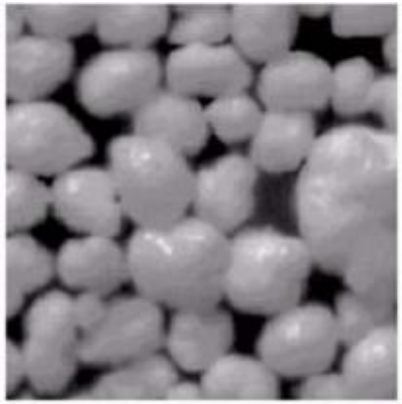
梯度图



梯度图上的分水岭



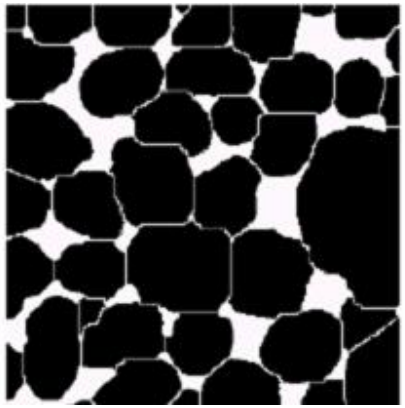
原图上的分水岭



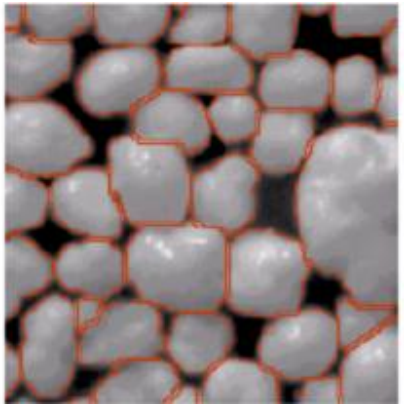
原始图



阈值分割



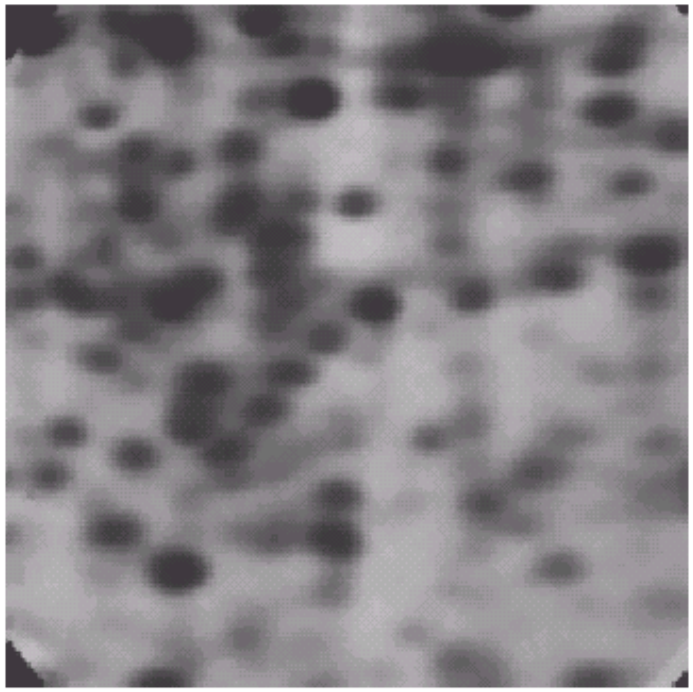
分水岭

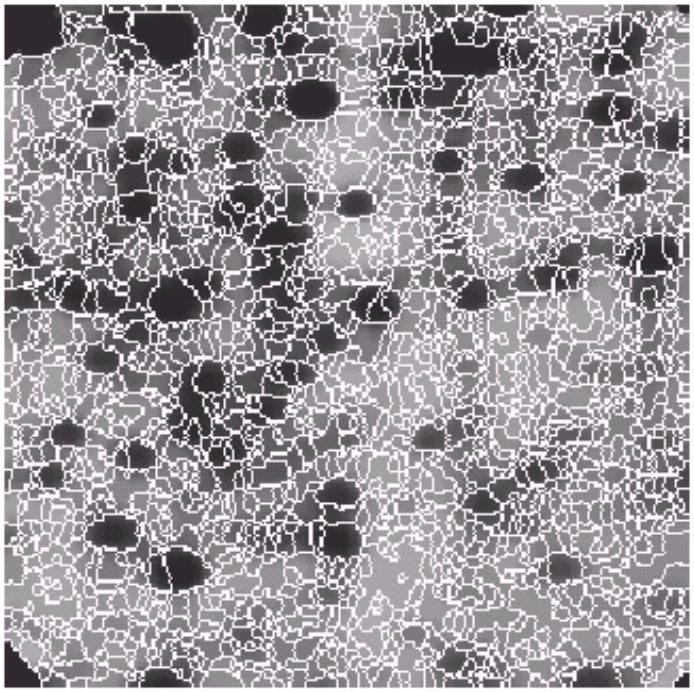


叠加轮廓

算法改进

- 过分割 (over-segmentation)
  - 分水岭算法依赖局部极小值的种子点





a b

FIGURE 10.47
(a) Electrophoresis image. (b) Result of applying the watershed segmentation algorithm to the gradient image. Oversegmentation is evident. (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)

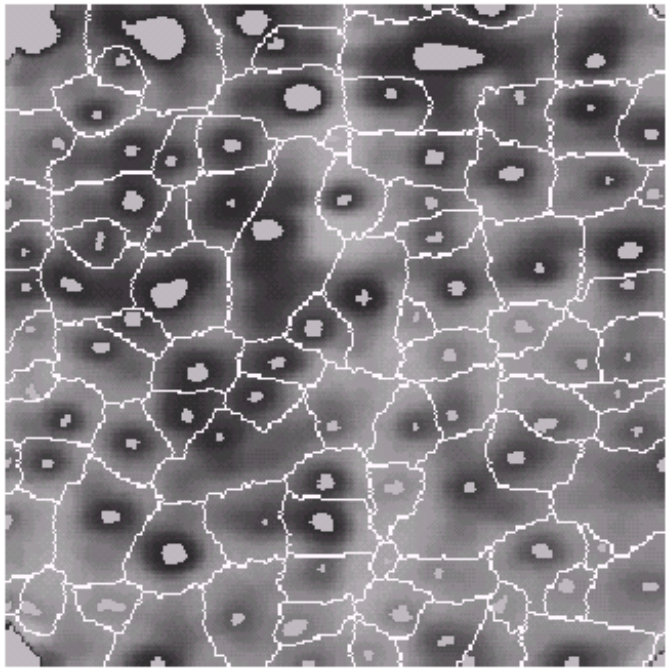
算法改进

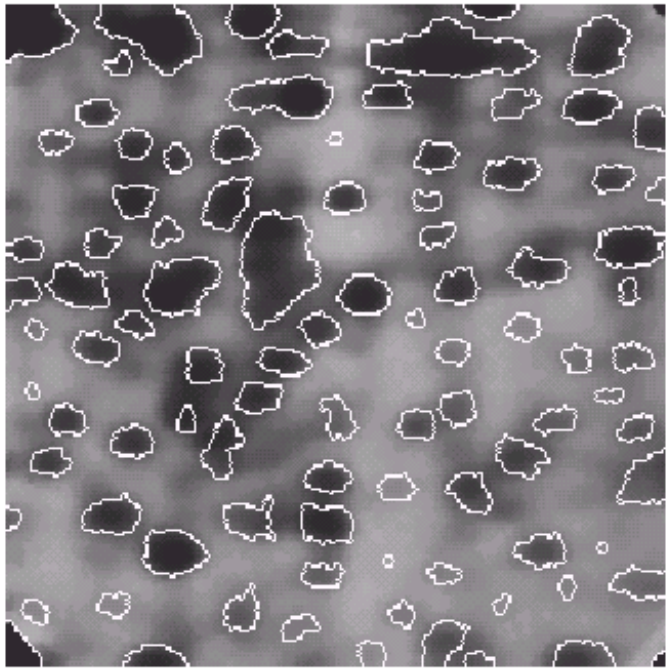
利用标记控制分割

- 过分割（over-segmentation）
  - ✓ 受图像中噪声和其它不规则结构影响
- 利用标记（marker）
  - ✓ 图象中的一个连通组元（基于灰度和连通性）
    - ➢ 内部标记：对应目标
    - ➢ 外部标记：对应背景（分水岭）

算法改进

- 利用标记控制分割
  - 限定允许的分割区域数目



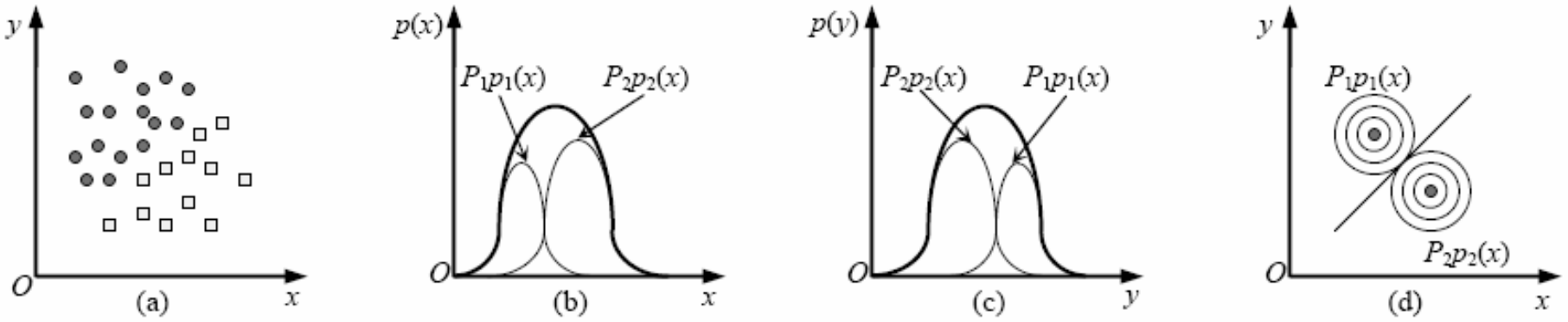


a b

FIGURE 10.48
(a) Image showing internal markers (light gray regions) and external markers (watershed lines). (b) Result of segmentation. Note the improvement over Fig. 10.47(b). (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)

聚类分割方法

- 聚类分割：
  - 利用图像特征的分布特性，通过将像素分簇聚类，实现图像分割
  - 视每个像素为一个样本，其表达信息可包括灰度、坐标位置、颜色、纹理，或其他特征
  - 灰度聚类
    - ✓ 取阈值，1-D聚类
  - 高维特征空间聚类
    - ✓ 区分能力较强



高维空间聚类的优点

K-means 聚类算法

- 基本步骤：
  1. 选择K个初始分类中心$\{\mu_1, \cdots, \mu_K\}$，每个分类中心代表一个类簇；
  2. 使用最小距离法将所有样本分类，即若$\forall j \neq i, Dist(\boldsymbol{x}, \boldsymbol{\mu}_i) < Dist(\boldsymbol{x}, \boldsymbol{\mu}_j)$，则将$\boldsymbol{x}$分为第$i$类；
  3. 根据第2步的分类结果，重新计算各类中心，并将此作为各类新的中心；
  4. 反复进行2、3步，直到各类中心趋于稳定。

问题

1. **如何确定K 值？**
   - ✓ **人为指定？自适应确定？**
2. **如何初始化聚类中心？**
   - ✓ **随机选样本作为聚类中心**
3. **如何定义距离测度？**
   - ✓ **欧式距离、汉明距离、……**

K-means 聚类形式化分析

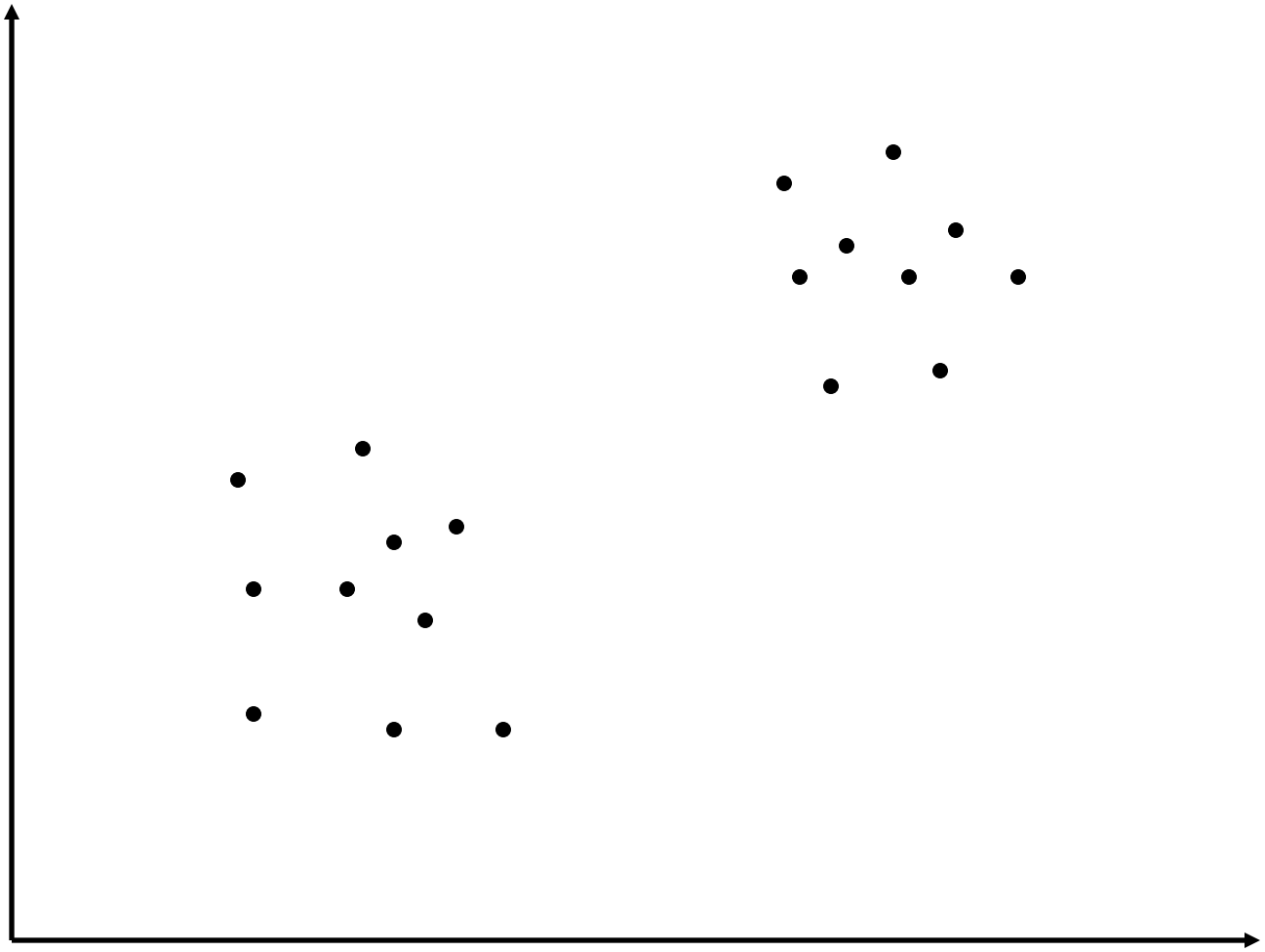
- 给定一个样本集$\{\mathrm{x}_1, \cdots, \mathrm{x}_N\}$，其中$x_i \in R^D$
- 目标：将样本集划分为K个簇，表示为$\{\mu_1, \cdots, \mu_K\}$
  - 划分关系$r_{n,k} = 1$：第 n 个样本被划分到第 k 簇(类)
- 优化目标：

$$\arg \min_{\substack{\{\mu_1, \cdots, \mu_K\} \\ \{r_{1,1}, \cdots, r_{NK}\}}} J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \left\| \mathbf{x}_n - \boldsymbol{\mu}_k \right\|^2$$

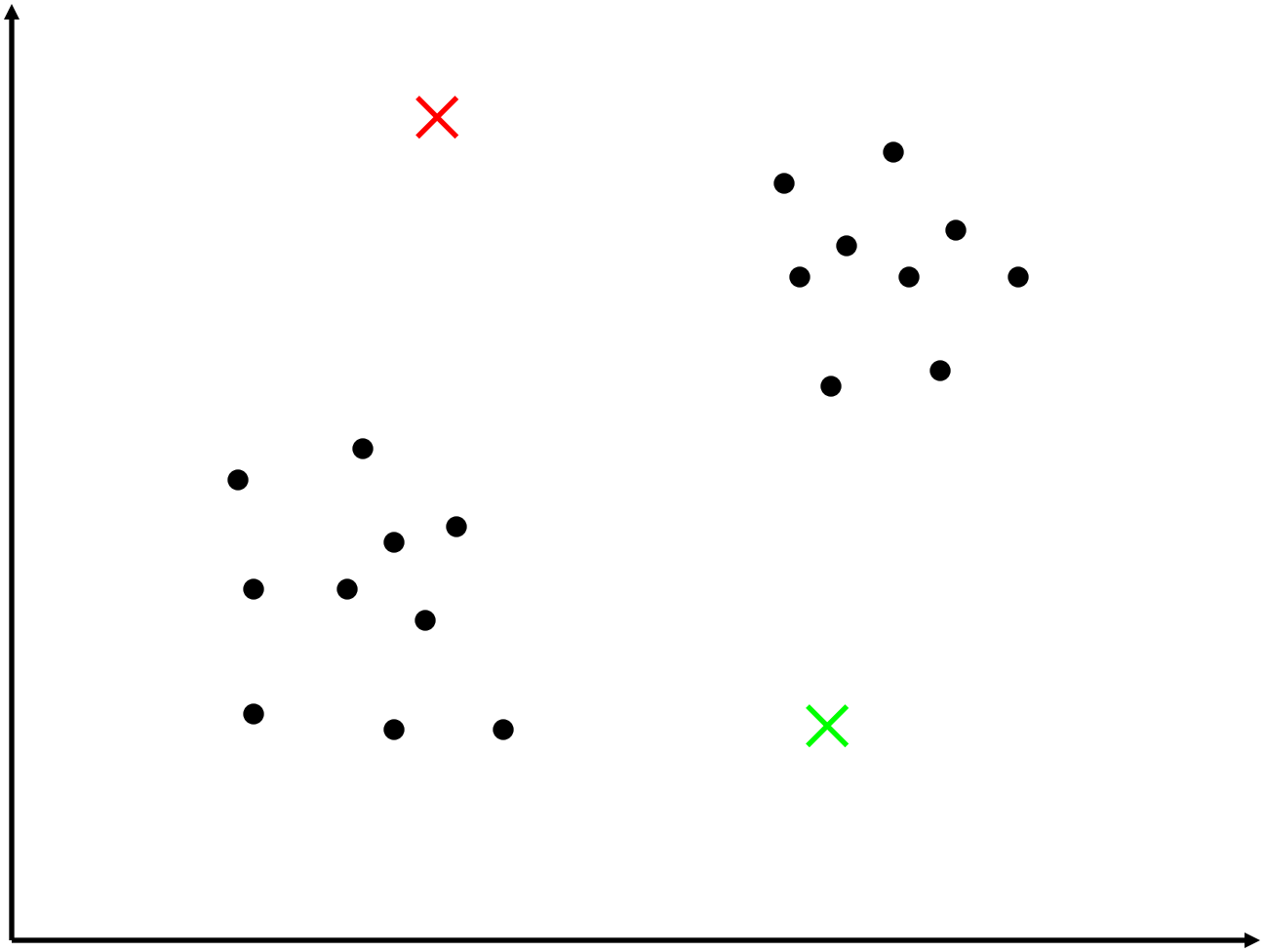
计算复杂度？

- 方法
  - 固定$\{\boldsymbol{\mu}_k\}$，解$\{r_{nk}\}$：$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_j \left\| \mathbf{x}_n - \boldsymbol{\mu}_j \right\| \\ 0 & \text{otherwise} \end{cases}$ $\sim O(NKD)$
  - 固定$\{r_{nk}\}$，解$\{\boldsymbol{\mu}_k\}$：$\boldsymbol{\mu}_k = \frac{\sum_n r_{nk} \mathbf{x}_n}{\sum_n r_{nk}}$ $\sim O(ND)$
  - 迭代进行上面两步，直至收敛

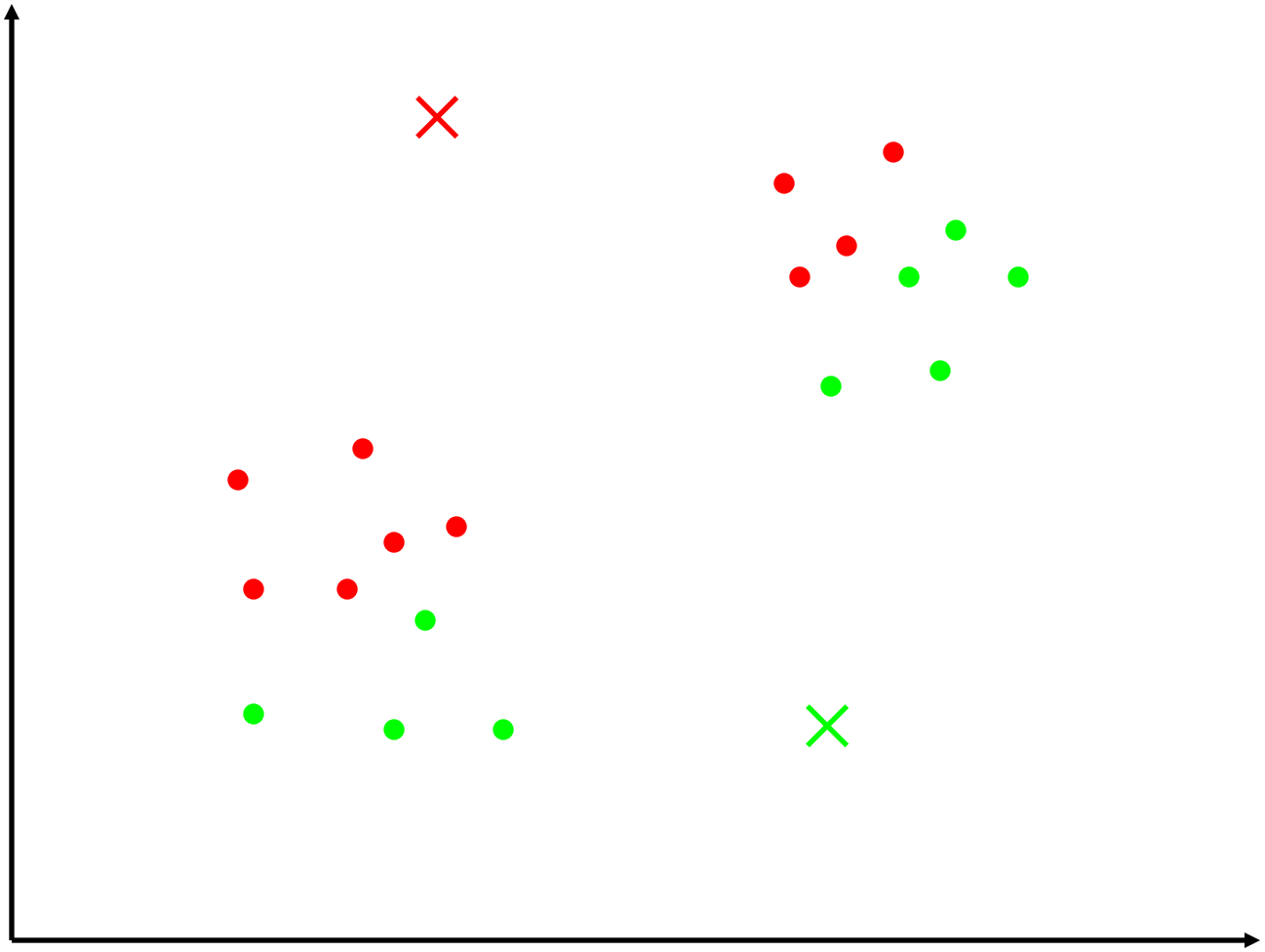
K-means 聚类过程演示 I



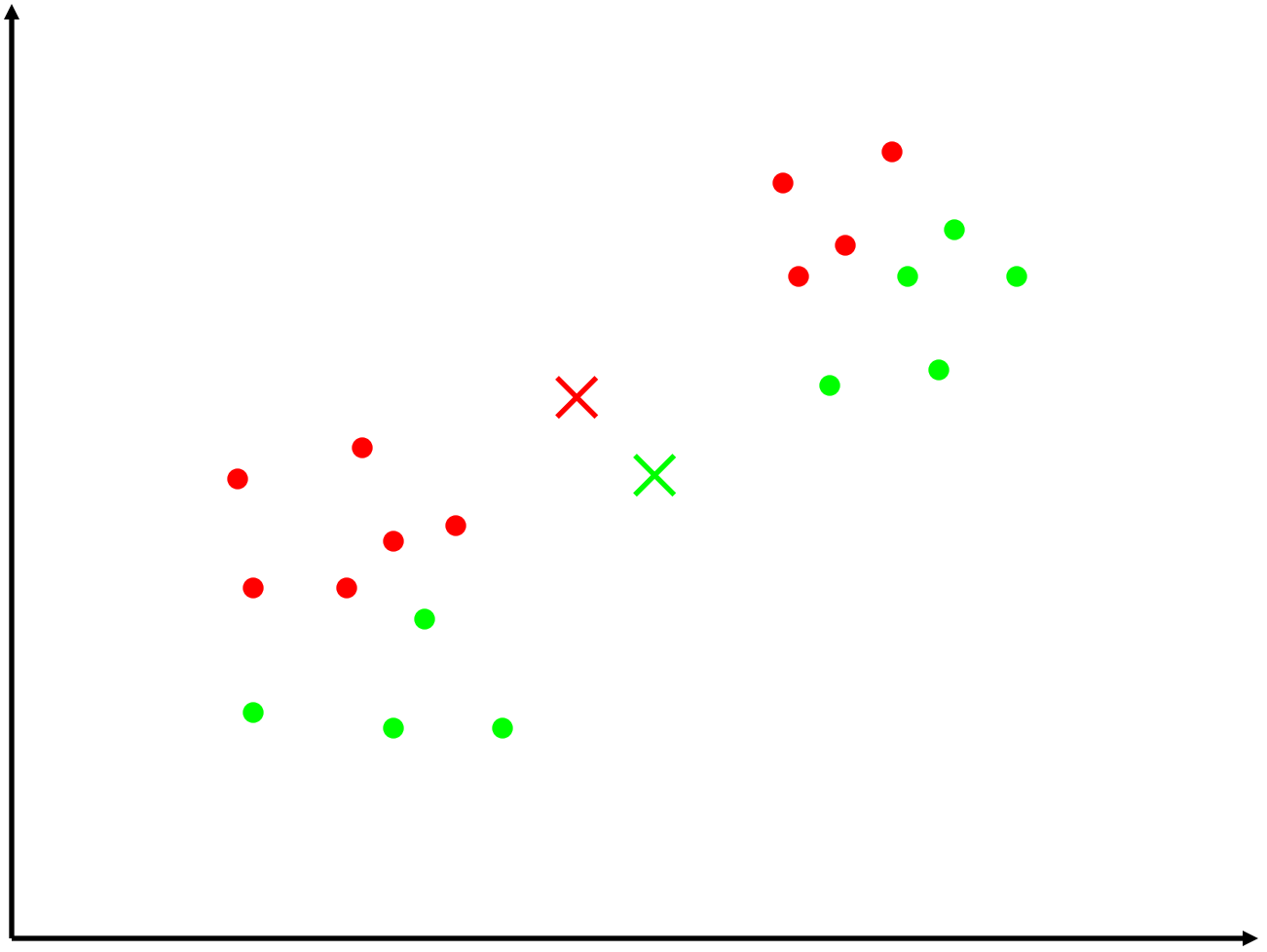
K-means 聚类过程演示 I



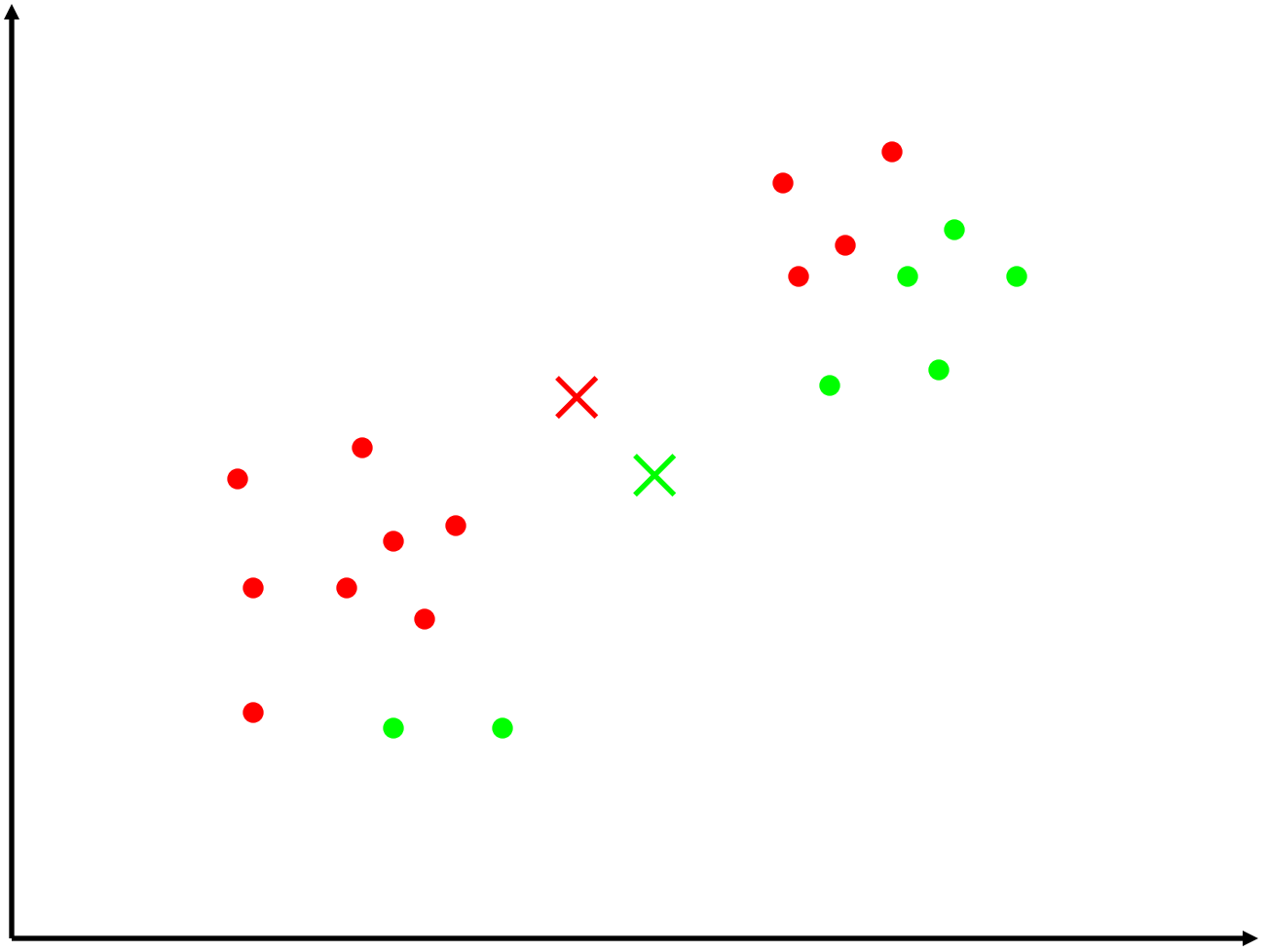
K-means 聚类过程演示 I



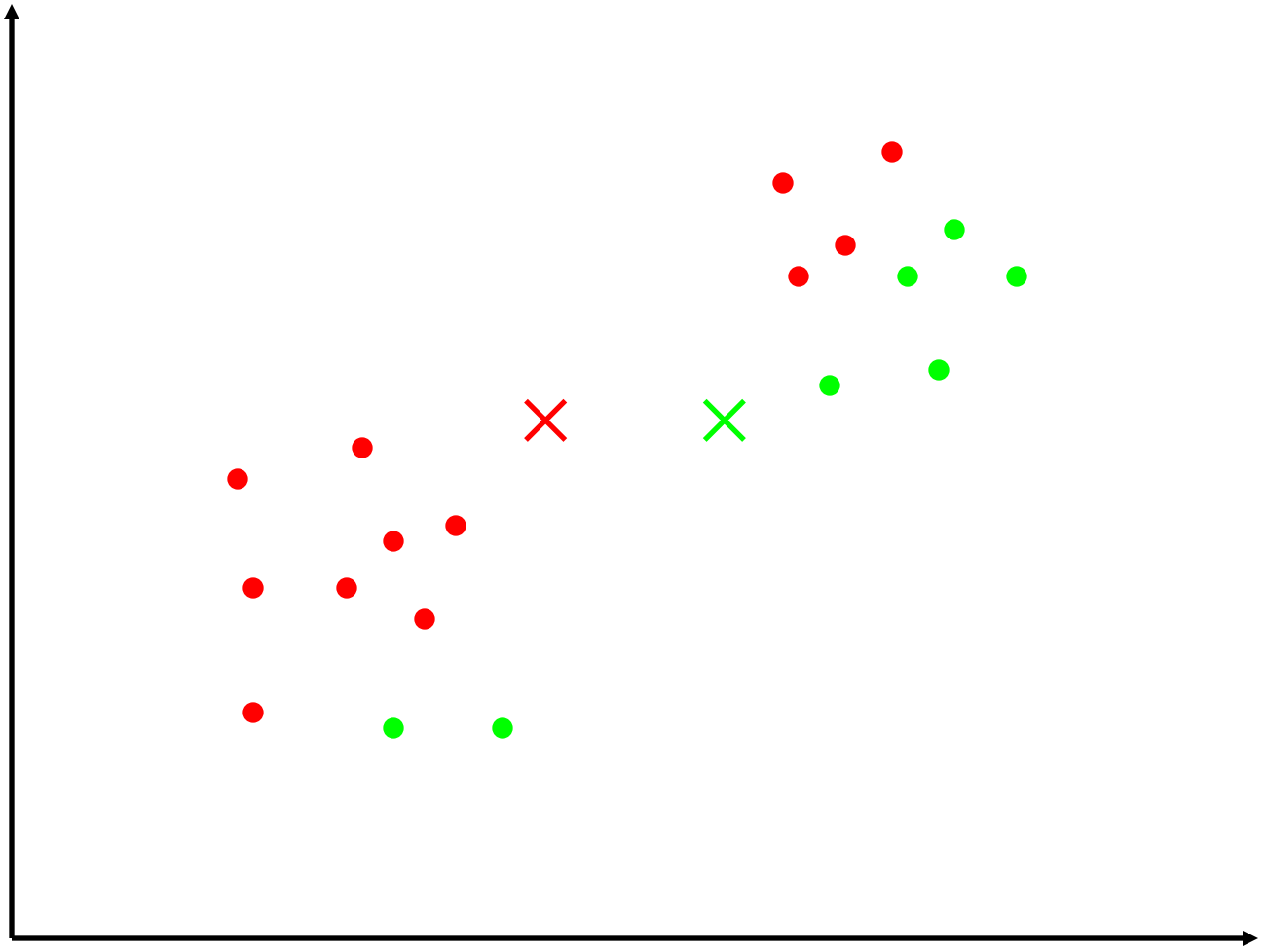
K-means 聚类过程演示 I



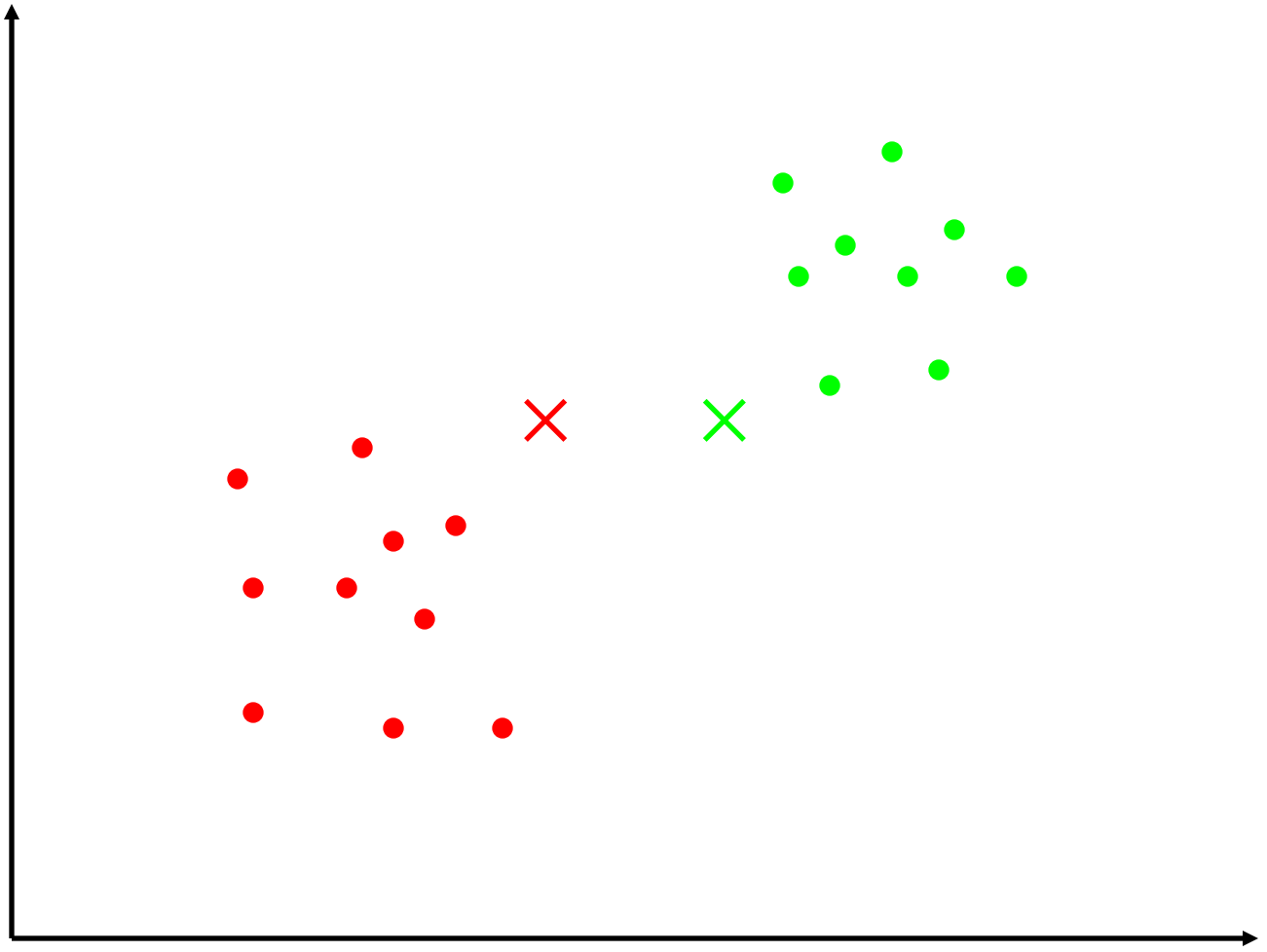
K-means 聚类过程演示 I



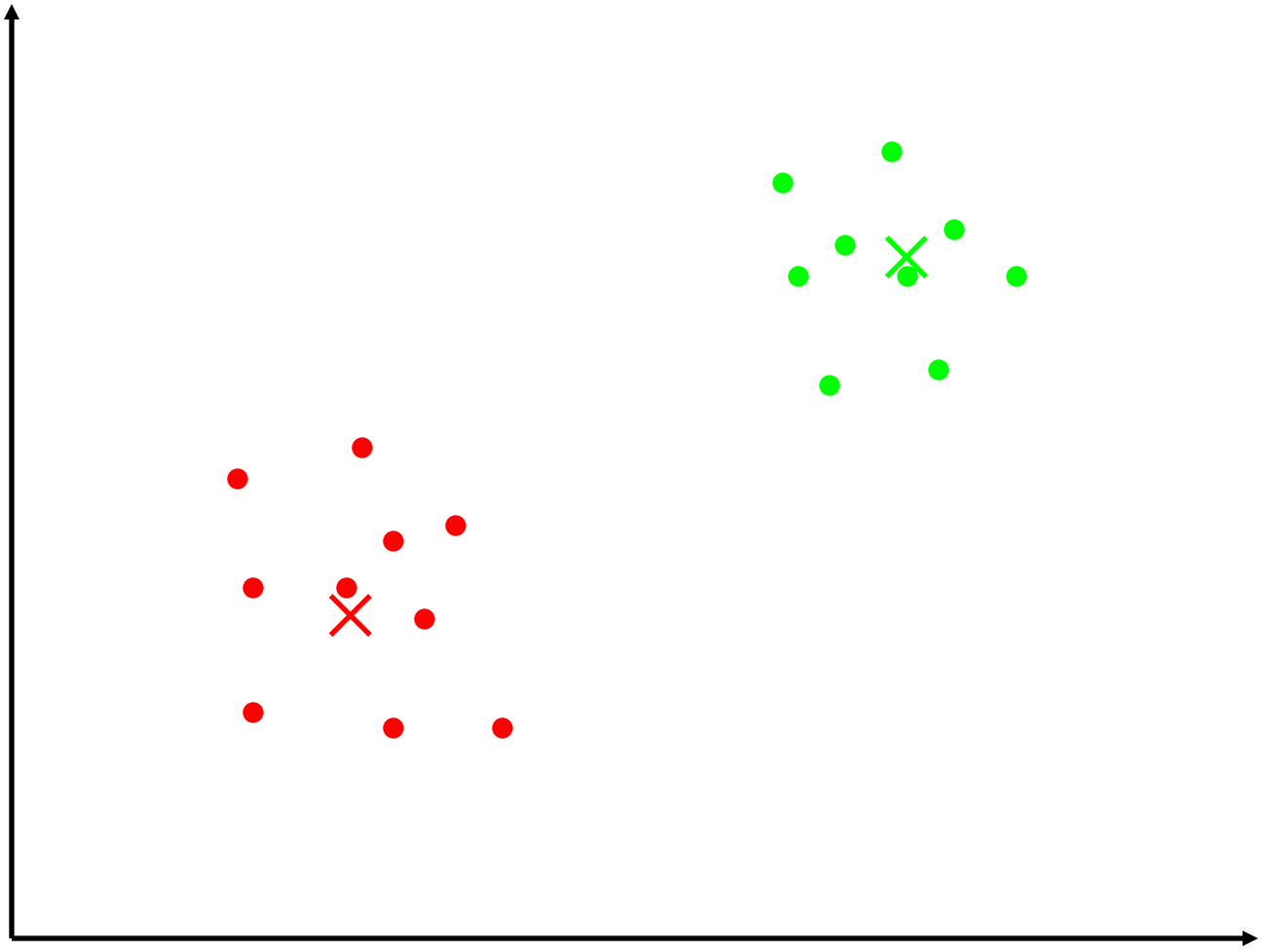
K-means 聚类过程演示 I



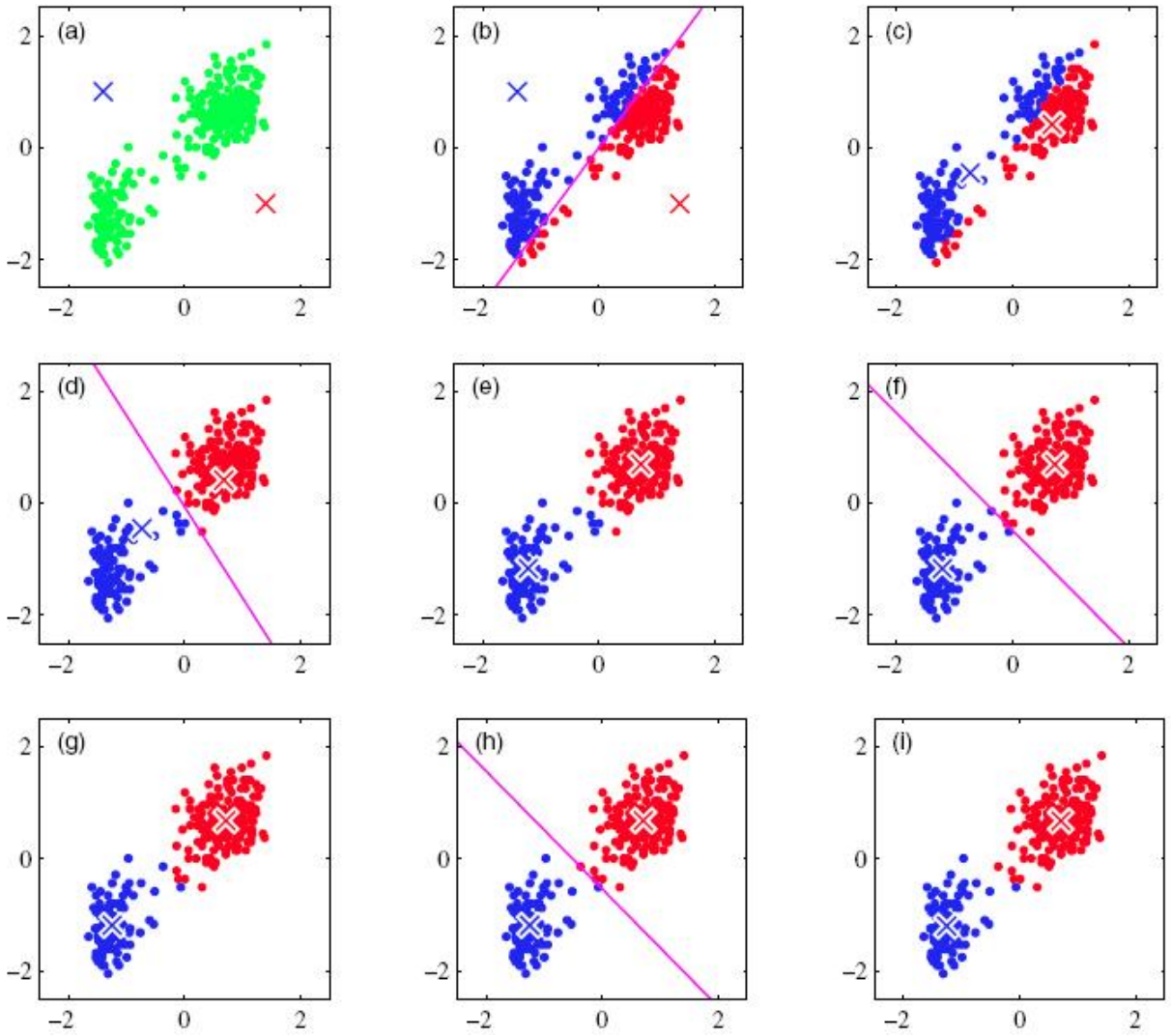
K-means 聚类过程演示 I

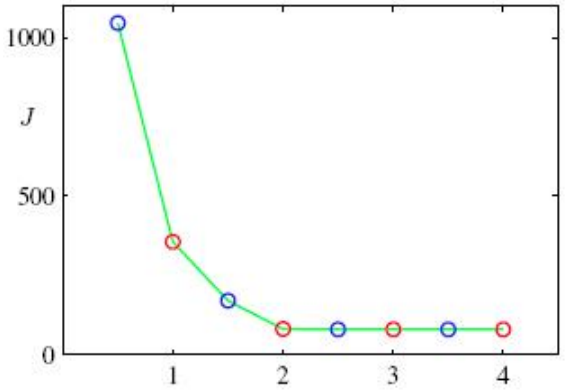


K-means 聚类过程演示 I



K-means algorithm 聚类过程演示





代价的值变化图。经过三次迭代，算法逐渐收敛。

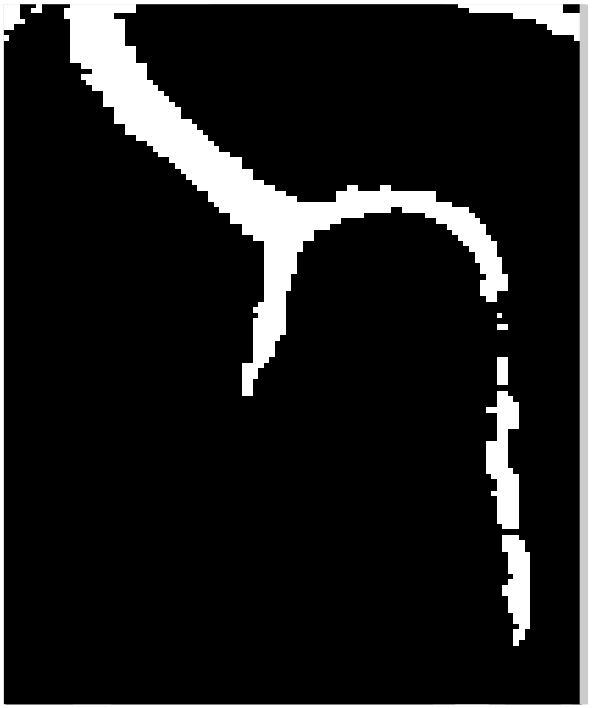
主动轮廓模型

- 传统的图像分割方法存在的问题
  - 基于阈值的分割
    - ✓ 难以选择合适的阈值
  - 边缘检测
    - ✓ 边缘没有闭合，难以得到完整目标区域

血管图像



二值化



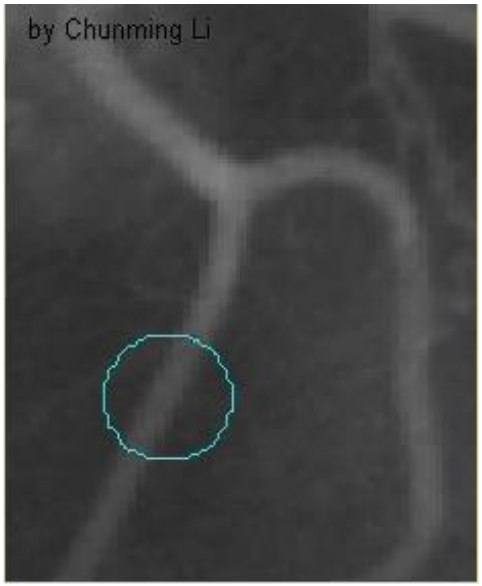
边缘检测

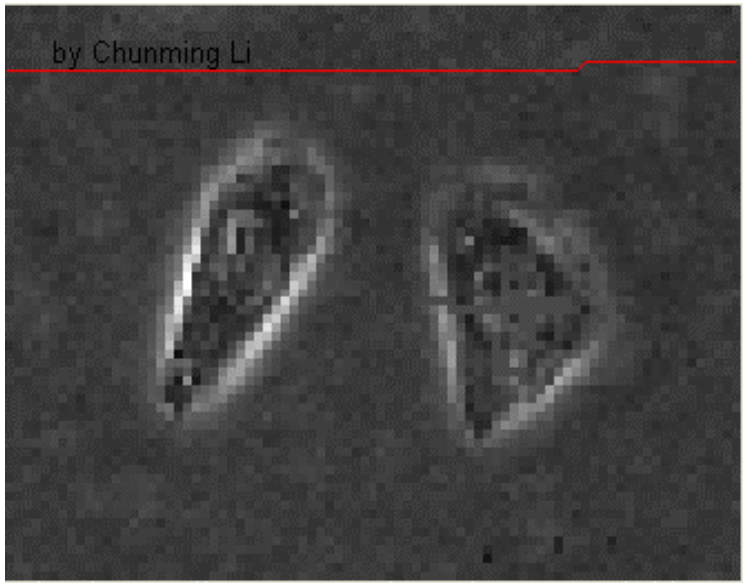


基于主动轮廓模型的图像分割

ACM: Active Contour Model

- 基本思想：通过在图像中演化一组动态曲线，自适应地控制曲线的演化速度，使得其停驻在物体边缘处，从而检测出图像中感兴趣的物体
- 两种类型
  - ✓ 参数化主动轮廓 ： snake
  - ✓ 几何主动轮廓：水平集方法 (level set)





主动轮廓模型的优势

- 易于对目标进行描述
- 亚像素的精度
- 易于融入各种信息
  - 如形状先验信息、运动信息
- 成熟的数学工具
  - 变分法、偏微分方程(PDE)，微分几何等

基于主动轮廓模型的图像分割

- 参数化主动轮廓 ： snakes
  - 经典工作：”Snakes: Active Contour Models”

> Snakes: Active contour models
> M Kass, A Witkin, D Terzopoulos - International journal of computer vision, 1988 - Springer
> A snake is an energy-minimizing spline guided by external constraint forces and influenced by image forces that pull it toward features such as lines and edges. Snakes are active contour models: they lock onto nearby edges, localizing them accurately. Scale-space …
> ☆ 99 被引用次数：23964 相关文章 所有 47 个版本 »

- 几何主动轮廓：水平集方法 (level set)
  - From Snake to Level Set
  - Classical Level Set Model

参数化主动轮廓 (Kass et al, 1988)

对于一个轮廓曲线$\mathrm{v}(s)=[x(s),y(s)]$,定义能量函数：

$$E^*_{\text{snake}} = \int_0^1 E_{\text{snake}}\big(\mathbf{v}(s)\big)\,\mathrm{d}s$$

$$= \int_0^1 \Big(E_{\text{int}}\big(\mathbf{v}(s)\big) + E_{\text{image}}\big(\mathbf{v}(s)\big) + E_{\text{con}}\big(\mathbf{v}(s)\big)\Big)\,\mathrm{d}s$$

曲线内在的弯曲能

图像相关的能量

外在的约束力

$$E_{\text{int}} = \alpha(s)\left|\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}s}\right|^2 + \beta(s)\left|\frac{\mathrm{d}^2\mathbf{v}}{\mathrm{d}s^2}\right|^2$$

$$E_{\text{image}} = w_{\text{line}}E_{\text{line}} + w_{\text{edge}}E_{\text{edge}} + w_{\text{term}}E_{\text{term}}$$

主动轮廓的演化

- Snakes能量函数

$$E_{\text{int}} = \alpha(s)\left|\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}s}\right|^2 + \beta(s)\left|\frac{\mathrm{d}^2\mathbf{v}}{\mathrm{d}s^2}\right|^2$$

$$E_{\text{image}} = w_{\text{line}}E_{\text{line}} + w_{\text{edge}}E_{\text{edge}} + w_{\text{term}}E_{\text{term}}$$

- 最陡梯度下降算法(变分法):

欧拉-拉格朗日方程

$$\begin{cases} \frac{\partial}{\partial t}C(s,t) = \alpha C''(s,t) - \beta C''''(s,t) - \nabla E_{\text{ext}} \\ C(s,0) = C_0(s) \end{cases}$$

$$E_{\text{ext}} = E_{\text{image}} + E_{\text{con}}$$

内在力: 规则化（Regularize）轮廓曲线，平滑性约束

外　力: 推动轮廓向感兴趣的物体边界运动

Demo for Snakes



曲线微分几何

- 平面曲线可以表达为

$$\mathcal{C}:[0,1]\rightarrow\Re^2 \qquad \mathcal{C}(p)=(x(p),y(p))$$

- 其中
  - 切向量：

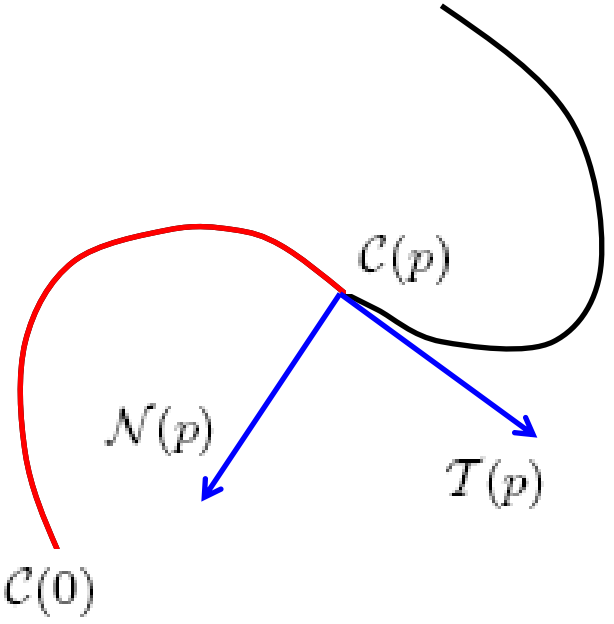
$$\mathcal{T}(p)=\mathcal{C}'(p)=(x'(p),y'(p))$$

  - 法向量

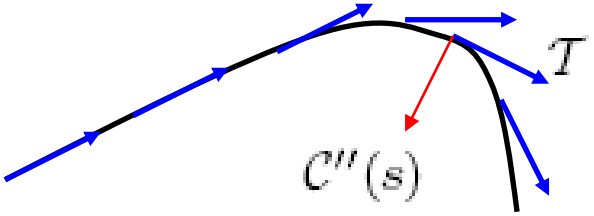
$$\mathcal{N}(p)=(\frac{-y'(p)}{\sqrt{x'(p)^2+y'(p)^2}},\frac{x'(p)}{\sqrt{x'(p)^2+y'(p)^2}})$$

  - 弧线长度: 从 $\mathcal{C}(0)$ 到 $\mathcal{C}(p)$

$$s(p)=\int_0^p|\mathcal{C}'(s)|ds$$



曲率 (curvature)



- 采用弧长 s 作为参数
  - $ds = |C'(p)|dp = \sqrt{(x'(p))^2+(y'(p))^2}dp$
  - 切向矢量 $T(s) = C'(s)$ 是一个单位向量

$$T(s) = C'(s) = \left(\frac{dx}{ds}, \frac{dy}{ds}\right) = \left(\frac{dx/dp}{ds/dp}, \frac{dy/dp}{ds/dp}\right) = \left(\frac{x'(p)}{\sqrt{(x'(p))^2+(y'(p))^2}}, \frac{y'(p)}{\sqrt{(x'(p))^2+(y'(p))^2}}\right)$$

单位切向量 T 定义了曲线的方向

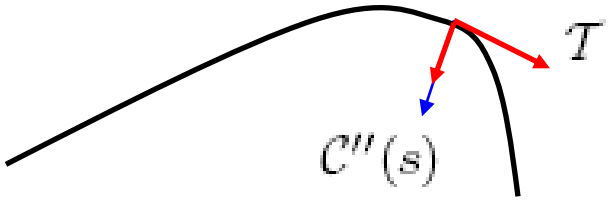
- 曲率描述了曲线方向改变的速度
- 再次求导: $C''(s) = T'(s)$
- 注意 $C'(s) \cdot C'(s) \equiv 1 \Longrightarrow C''(s) \cdot C'(s) = 0$

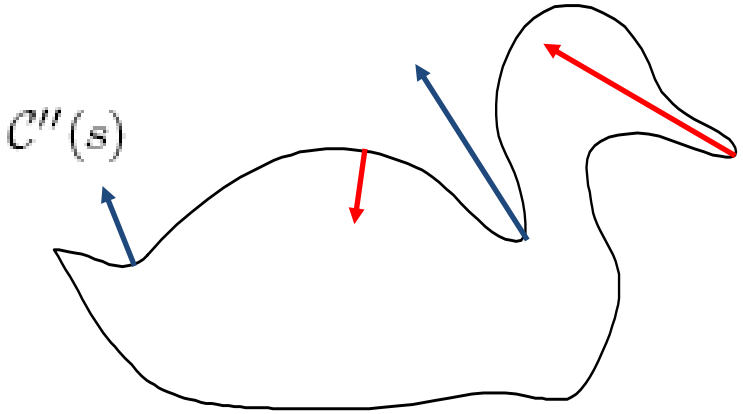
曲率 (curvature)

- $\mathcal{C}''(s)$和 $\mathcal{N}$均正交于 $\mathcal{T}$

  ⇒ $\mathcal{C}''(s)$ 平行于 $\mathcal{N}$

  ⇒ $\mathcal{C}''(s) = \kappa(s)\mathcal{N}(s)$





- 曲线的曲率：曲线上某个点的切线方向角对弧长的转动率
  - 对于一般的参数化形式 $\mathcal{C}(p) = (x(p), y(p))$ 曲率可以表示为：

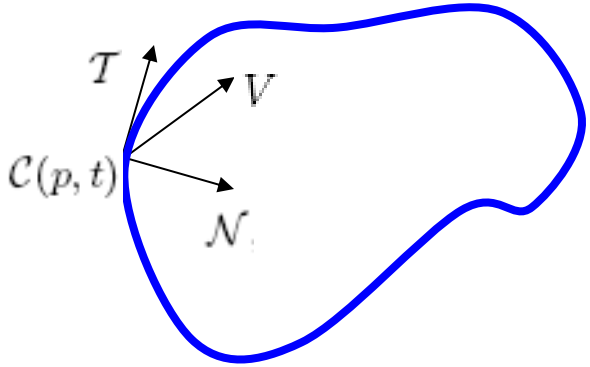
$$\kappa(p) = \frac{x'(p)y''(p) - y'(p)x''(p)}{(x'(p)^2 + y'(p)^2)^{3/2}}$$

动态曲线

- 动态曲线随时间变化： $\mathcal{C}(p,t) = (x(p,t), y(p,t))$

- 曲线运动由曲线演化方程确定:

$$\frac{\partial \mathcal{C}(p,t)}{\partial t} = V(p,t)$$



- 切向量 $\mathcal{T}$ 和法向量 $\mathcal{N}$ 构成 $\Re^2$ 空间的正交基

$$\Rightarrow \quad \frac{\partial \mathcal{C}(p,t)}{\partial t} = \alpha \mathcal{T} + \beta \mathcal{N}$$

问题：当$\alpha \neq 0, \beta = 0$时，动态曲线随时间是否有变化？

几何曲线演化

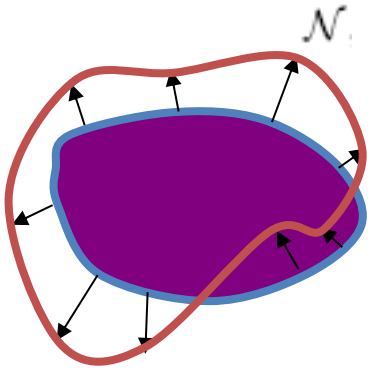
- 定理: 设β与参数表示方式无关. 若$\mathcal{C}(p,t)$ 按照以下方程进行演化：

$$\frac{\partial \mathcal{C}(p,t)}{\partial t} = \alpha \mathcal{T} + \beta \mathcal{N}$$

则存在$\mathcal{C}(p,t)$的另一个参数化形式$\bar{\mathcal{C}}(p,t)$，使得$\bar{\mathcal{C}}(p,t)$为以下方程的解：

$$\frac{\partial \bar{\mathcal{C}}(q,t)}{\partial t} = \bar{\beta} \mathcal{N}$$

其中$\bar{\beta} = \beta$对于同一个点：$\bar{\mathcal{C}}(q) = \mathcal{C}(p)$



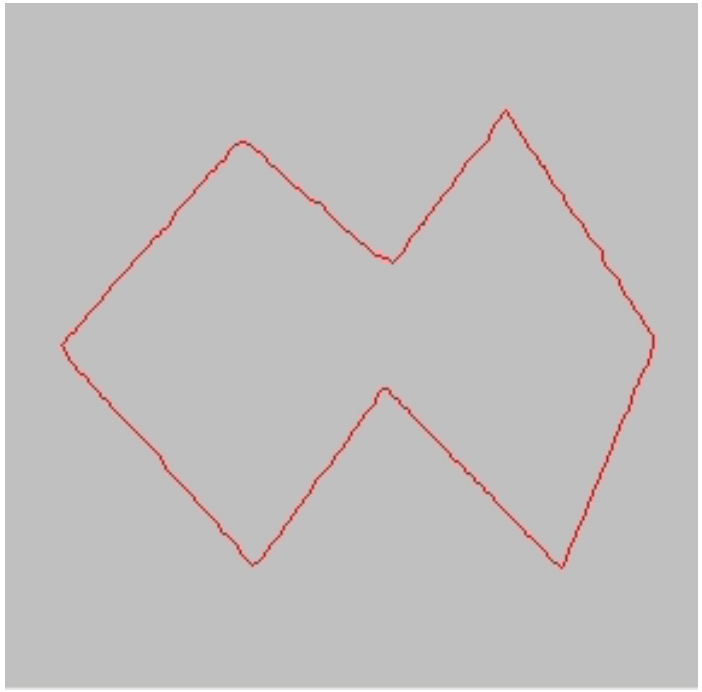
- 参数演化的一般方程：

$$\frac{\partial \mathcal{C}}{\partial t} = F \mathcal{N}$$

常速运动和平均曲率运动

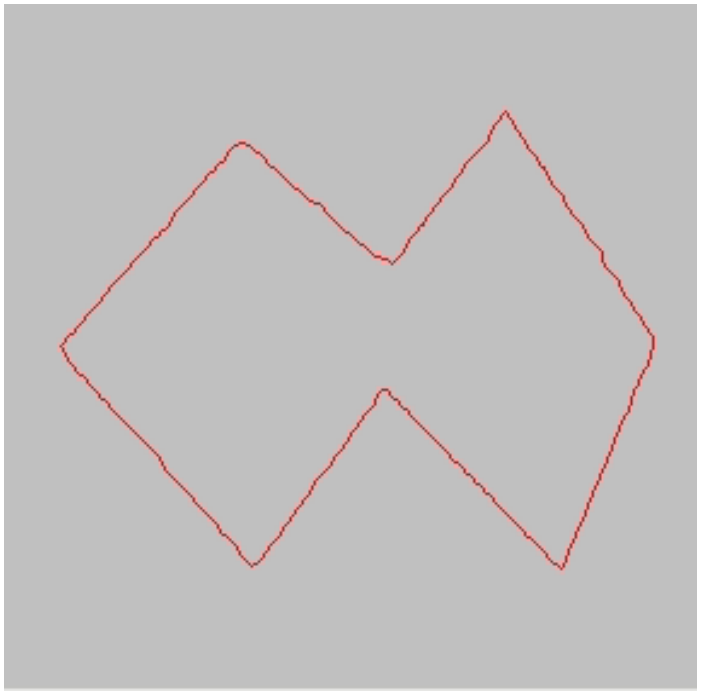
常速运动
(Area decreasing/increasing)

$$\frac{\partial \mathcal{C}}{\partial t} = c\mathcal{N}$$



平均曲率运动
(Length shortening flow)

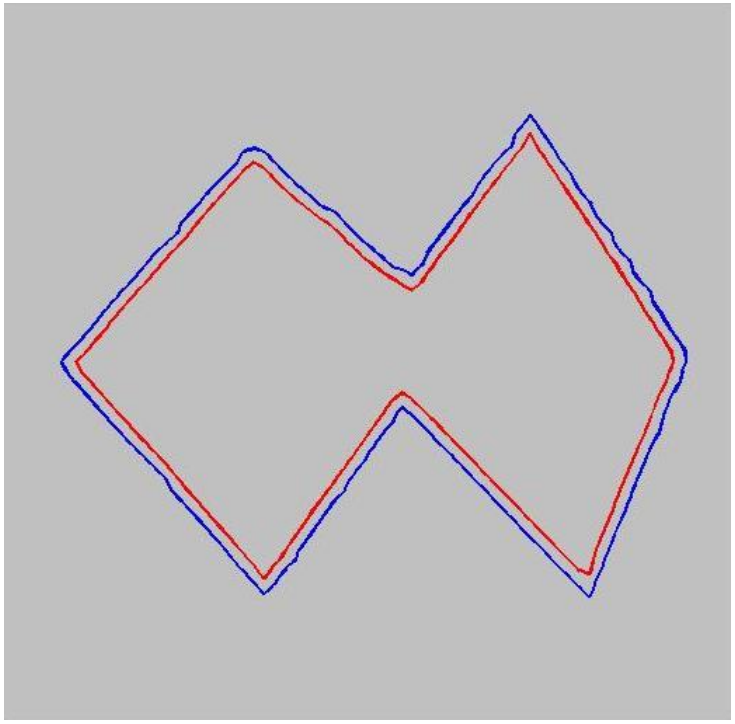
$$\frac{\partial \mathcal{C}}{\partial t} = \kappa\mathcal{N}$$



演化中间结果

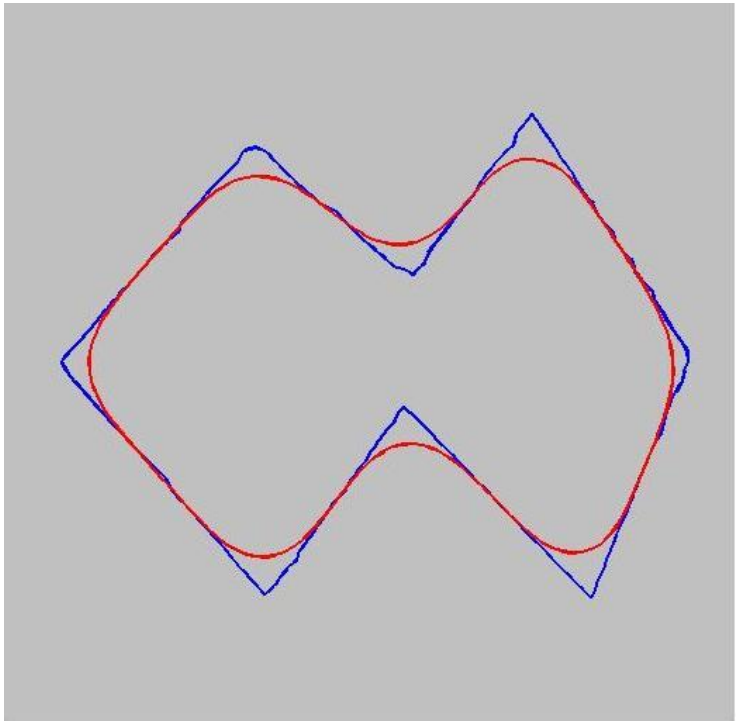
常速运动
(Area decreasing/increasing)

$$\frac{\partial \mathcal{C}}{\partial t} = c\mathcal{N}$$



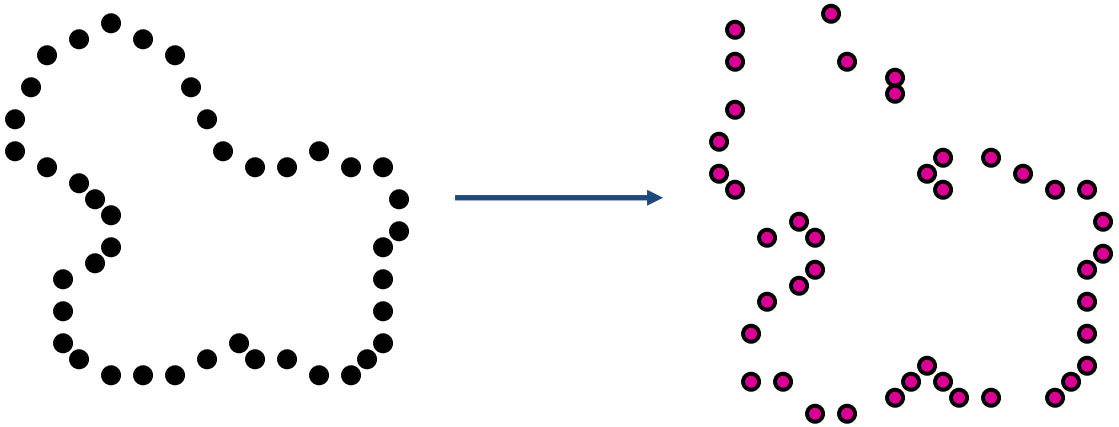
平均曲率运动
(Length shortening flow)

$$\frac{\partial \mathcal{C}}{\partial t} = \kappa\mathcal{N}$$

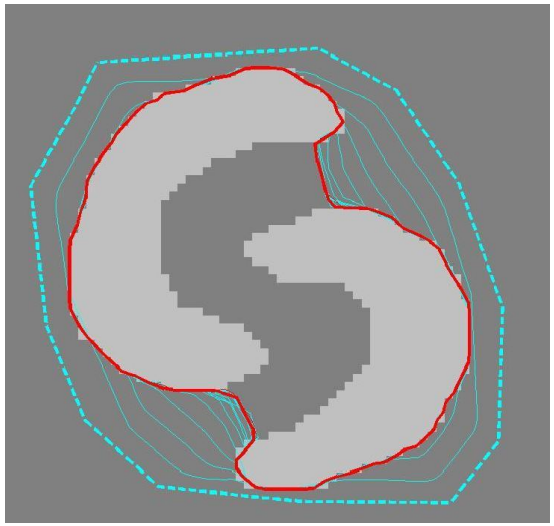


参数化主动轮廓的缺陷

- 演化过程中重新参数化: 对于3D 曲面尤为困难



- 不能处理拓扑变化



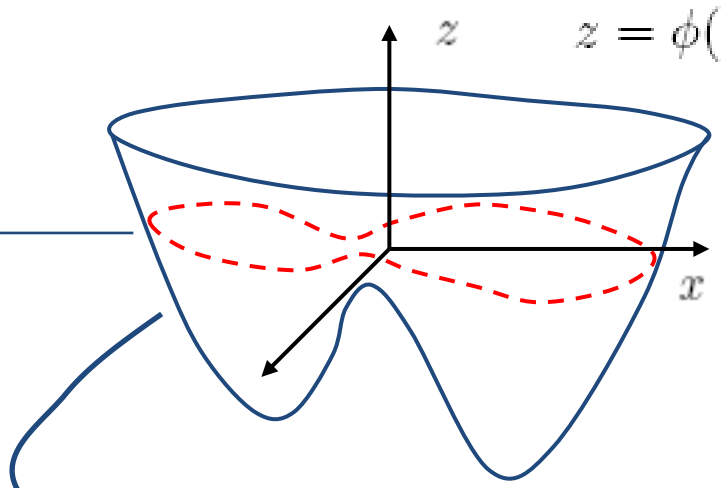
解决途径

- 基于参数化主动轮廓框架下的方法:
  - T-Snakes: topologically adaptable snakes
    - ✓ McInerney and Terzopoulos, 1995
    - ✓ Ray and Acton, 2003
    - ✓ Li, Liu, and Fox, 2005

- 更好的选择:
  - 水平集方法 (Level set methods)

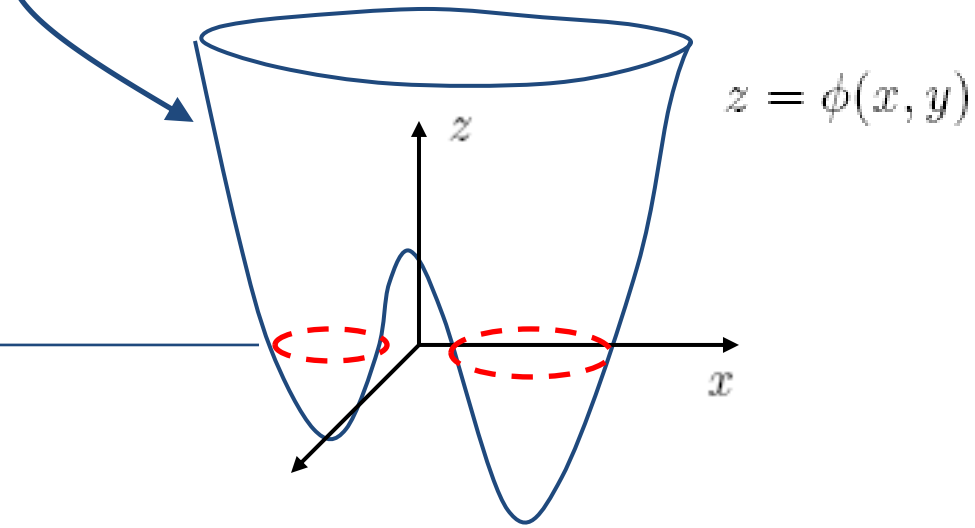
曲线的水平集描述

- **水平集**：曲面上，同一等高线上的点的集合
- 升维：2D → 3D



问题：如何选择水平集函数$\phi(x, y)$?

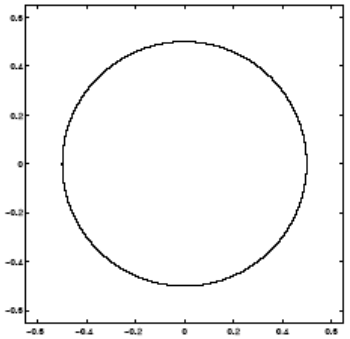
- 降维：3D → 2D



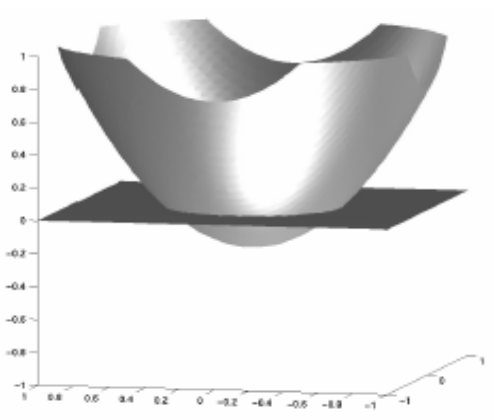
水平集函数 (level set function)

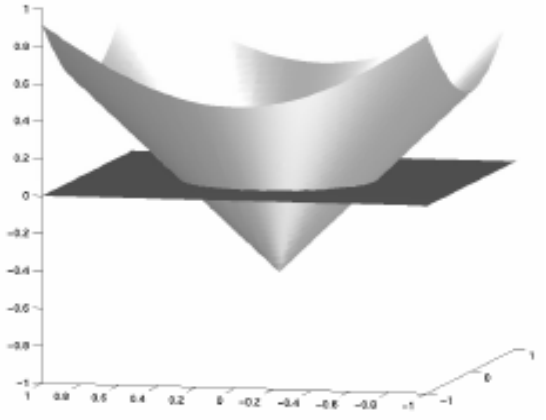
- 如何选择水平集函数$\phi(x,y)$?
  - 曲面梯度方向与水平面夹角为45度

轮廓线



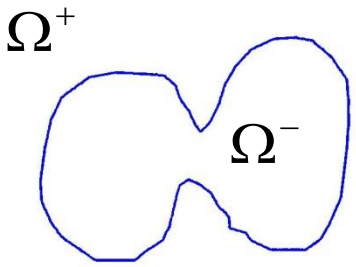
零水平集相同的水平集函数



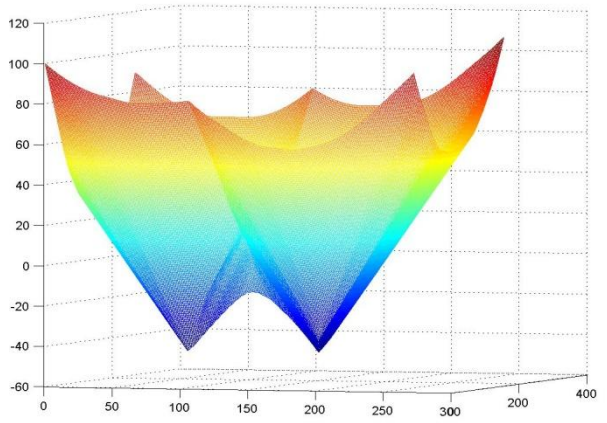


符号距离函数 (Signed distance function)

轮廓 C



符号距离函数



符号距离函数定义：$\phi(\mathbf{x}) = \begin{cases} dist(\mathbf{x}, C) & \text{if } \mathbf{x} \text{ is outside } C \\ 0 & \mathbf{x} \in C \\ -dist(\mathbf{x}, C) & \text{if } \mathbf{x} \text{ is inside } C \end{cases}$

可以证明：$|\nabla \phi(\mathbf{x})| = 1$

从曲线演化到水平集演化

- 曲线演化方程：

$$\frac{\partial \mathcal{C}}{\partial t} = F\mathcal{N}$$

其中 F 为速度函数，N 为曲线 C 的法向量

- 把动态**闭合**曲线$C(\mathbf{p},t)$作为零水平集，嵌入到一个随时间变化的函数$\phi(\mathbf{x},t)$ 中，使得$\phi(C(\mathbf{p},t),t)=0$

- 对$\phi(C(\mathbf{p},t),t)$关于时间 t 求微分 $\Rightarrow$ $\frac{\partial \phi}{\partial t} + \nabla\phi \cdot \frac{\partial \mathcal{C}}{\partial t} = 0$

$$\frac{\partial \phi}{\partial t} = -\nabla\phi \cdot \frac{\partial \mathcal{C}}{\partial t} = -\nabla\phi \cdot F\mathcal{N}$$

$$\mathcal{N} = -\frac{\nabla\phi}{|\nabla\phi|}$$

- 水平集演化方程:

$$\frac{\partial \phi}{\partial t} = F|\nabla\phi|$$

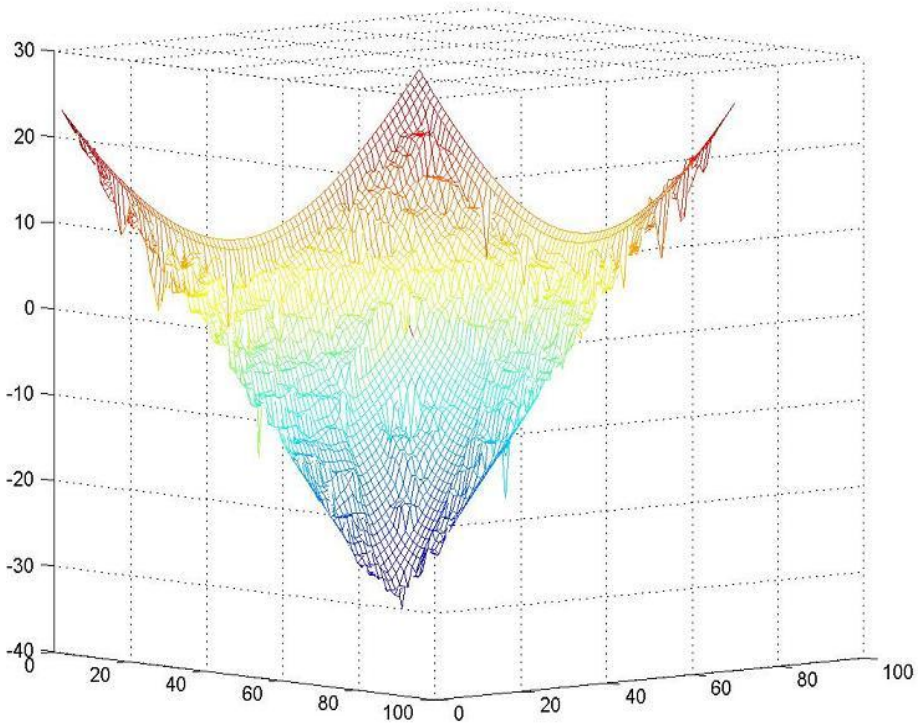
一些特殊的演化方程

- 平均曲率运动 $\dfrac{\partial \mathcal{C}}{\partial t} = \kappa \mathcal{N}$ ⇒ $\dfrac{\partial \phi}{\partial t} = \operatorname{div}\left(\dfrac{\nabla \phi}{|\nabla \phi|}\right)|\nabla \phi|$
- 常速率运动 $\dfrac{\partial \mathcal{C}}{\partial t} = c\mathcal{N}$ ⇒ $\dfrac{\partial \phi}{\partial t} = c|\nabla \phi|$
- 对流运动 $\dfrac{\partial C}{\partial t} = \langle \vec{V}, N \rangle N$ ⇒ $\dfrac{\partial \phi}{\partial t} = -\langle \vec{V}, \nabla \phi \rangle$
- 测地线主动轮廓 $\dfrac{\partial \mathcal{C}}{\partial t} = (c+g)\kappa \mathcal{N} - \langle \nabla g, \mathcal{N} \rangle \mathcal{N}$

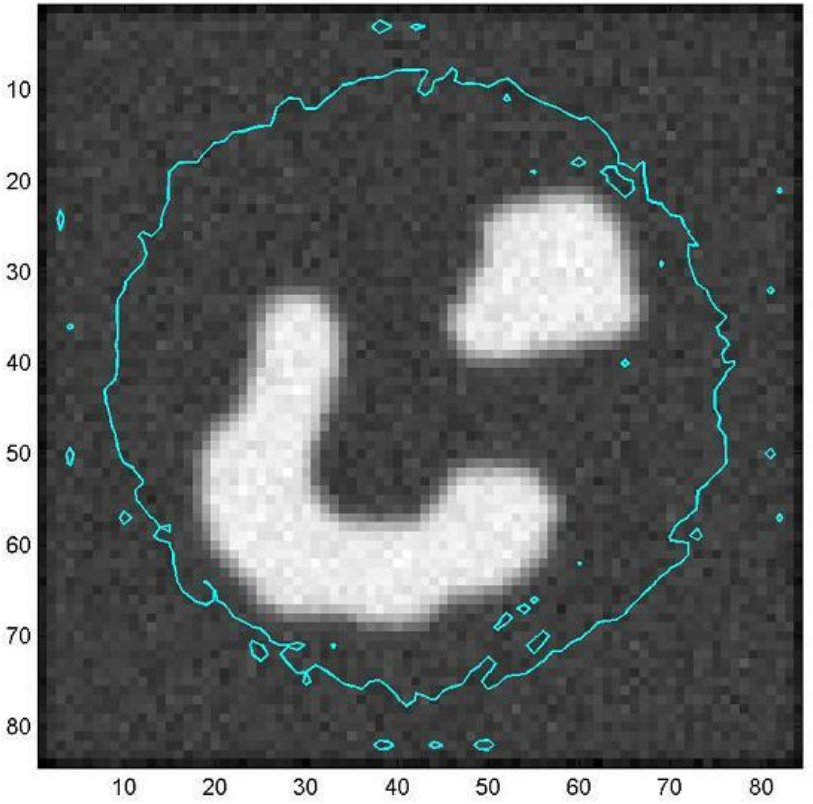
$$\frac{\partial \phi}{\partial t} = \left(c + \operatorname{div}\left(\frac{\nabla \phi}{|\nabla \phi|}\right)\right)|\nabla \phi| + \langle \nabla g, \nabla \phi \rangle$$

演化不稳定　重新初始化

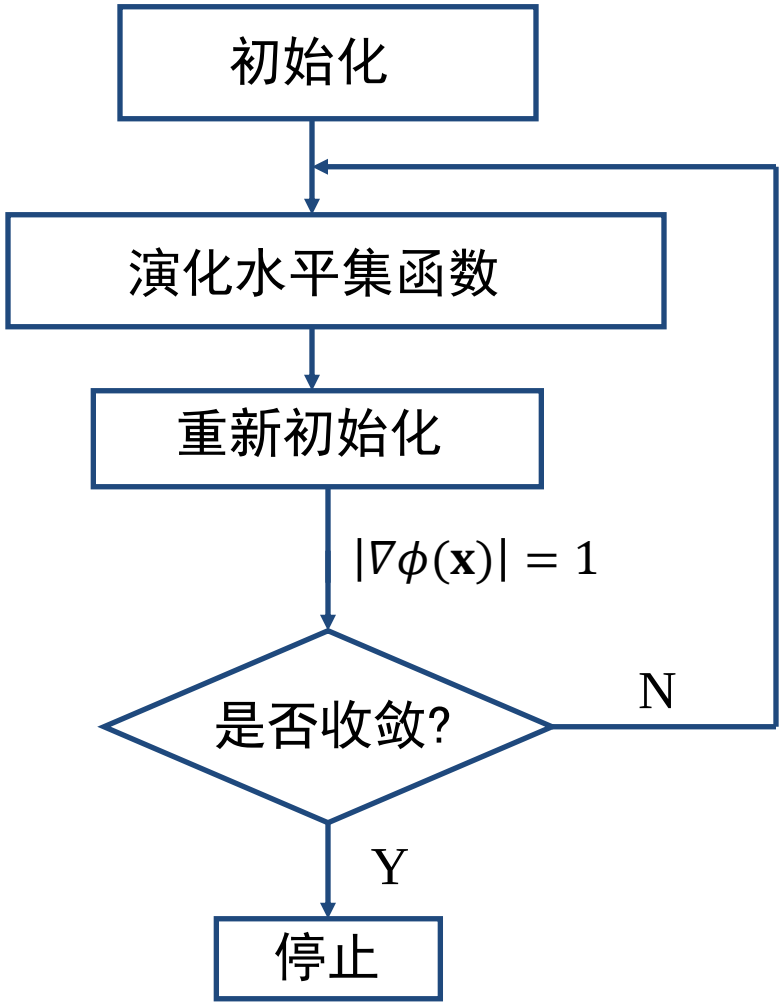
降质(Degraded)的水平集函数, 50步迭代后, 时间步长为0.1



降质水平集函数的零水平集



水平集方法的一般步骤



Level Set without Re-initialization

- 将re-initialization形式化为能量泛函: $\mathcal{P}(\phi) = \int_\Omega \frac{1}{2}(|\nabla\phi| - 1)^2 dxdy$
- 定义变分能量函数 $\mathcal{E}(\phi) = \mu\mathcal{P}(\phi) + \mathcal{E}_{g,\lambda,\nu}(\phi)$

$$= \mu\mathcal{P}(\phi) + \lambda\mathcal{L}_g(\phi) + \nu\mathcal{A}_g(\phi)$$

其中 $\mathcal{L}_g(\phi) = \int_\Omega g\delta(\phi)|\nabla\phi|dxdy$ $\mathcal{A}_g(\phi) = \int_\Omega gH(-\phi)dxdy$ $g = \frac{1}{1+|\nabla G_\sigma * I|^2}$

- 对能量函数求梯度

$$\frac{\partial\mathcal{E}}{\partial\phi} = -\mu[\Delta\phi - \mathrm{div}(\frac{\nabla\phi}{|\nabla\phi|})] - \lambda\delta(\phi)\mathrm{div}(g\frac{\nabla\phi}{|\nabla\phi|}) - \nu g\delta(\phi)$$

- 基于梯度下降算法，得到演化方程：

$$\frac{\partial\phi}{\partial t} = \mu[\Delta\phi - \mathrm{div}(\frac{\nabla\phi}{|\nabla\phi|})] + \lambda\delta(\phi)\mathrm{div}(g\frac{\nabla\phi}{|\nabla\phi|}) + \nu g\delta(\phi)$$

- Li, Chunming, et al. "Level set evolution without re-initialization: a new variational formulation." *IEEE CVPR*, 2005.

变分法推导示例

An instance for calculus of variations:

Define a functional :$E(\phi) = \int_\Omega \frac{1}{2}\left(|\nabla\phi| - 1\right)^2 dxdy$

$$= \int_\Omega \frac{1}{2}\left(|\nabla\phi|^2 - 2|\nabla\phi| + 1\right)dxdy$$

Let $F(\phi) = \frac{1}{2}\left(|\nabla\phi|^2 - 2|\nabla\phi| + 1\right)$

$$= \frac{1}{2}\left((\phi_x^{\,2} + \phi_y^{\,2}) - 2\sqrt{\phi_x^{\,2} + \phi_y^{\,2}} + 1\right)$$

in terms of a small variable δ and an arbitrary function h which satisfies : $h\big|_{\partial\Omega} = 0$, we can get:

$$F(\phi + \delta h) = \frac{1}{2}\left[\ ((\phi + \delta h)_x^{\,2} + (\phi + \delta h)_y^{\,2}) - 2\sqrt{(\phi + \delta h)_x^{\,2} + (\phi + \delta h)_y^{\,2}} + 1\right]$$

变分法推导示例

then:

$$\frac{\partial \mathrm{F}(\phi+\delta h)}{\partial \delta}=h_x(\phi+\delta h)_x+h_y(\phi+\delta h)_y-\frac{h_x(\phi+\delta h)_x+h_y(\phi+\delta h)_y}{\sqrt{(\phi+\delta h)_x^{\,2}+(\phi+\delta h)_y^{\,2}}}$$

$$=\nabla h\bullet\nabla(\phi+\delta h)-\frac{\nabla h\bullet\nabla(\phi+\delta h)}{\sqrt{(\phi_x^{\,2}+\phi_y^{\,2})+\delta^2(h_x^{\,2}+h_y^{\,2})+2\delta\nabla h\bullet\nabla\phi}}$$

$$\frac{\partial \mathrm{F}(\phi+\delta h)}{\partial \delta}\Big|_{\delta\to 0}=\nabla h\bullet\nabla\phi-\frac{\nabla h\bullet\nabla\phi}{\sqrt{(\phi_x^{\,2}+\phi_y^{\,2})}}$$

$$Then\quad \frac{\partial E(\phi+\delta h)}{\partial \delta}\Big|_{\delta\to 0}=\int_\Omega\left(\nabla h\bullet\nabla\phi-\frac{\nabla h\bullet\nabla\phi}{\sqrt{(\phi_x^{\,2}+\phi_y^{\,2})}}\right)dxdy$$

$$=\int_\Omega(h_x\phi_x+h_y\phi_y)dxdy-\int_\Omega\frac{(h_x\phi_x+h_y\phi_y)}{\sqrt{(\phi_x^{\,2}+\phi_y^{\,2})}}dxdy$$

变分法推导示例

$$\because \frac{\partial}{\partial x}\left[\phi_x h\right] = h_x\phi_x + h\phi_{xx},$$

$$\frac{\partial}{\partial y}\left[\phi_y h\right] = h_y\phi_y + h\phi_{yy},$$

$$\therefore \left.\frac{\partial E(\phi+\delta h)}{\partial \delta}\right|_{\delta\to 0} = \int_\Omega \frac{\partial}{\partial x}\left[\phi_x h\right] + \frac{\partial}{\partial y}\left[\phi_y h\right] dxdy - \int_\Omega h\phi_{xx} + h\phi_{yy} dxdy$$

$$- \int_\Omega \frac{\partial}{\partial x}\left[\frac{\phi_x}{|\nabla\phi|}h\right] + \frac{\partial}{\partial y}\left[\frac{\phi_y}{|\nabla\phi|}h\right] dxdy + \int_\Omega \frac{\partial}{\partial x}\left[\frac{\phi_x}{|\nabla\phi|}\right] h + h\frac{\partial}{\partial y}\left[\frac{\phi_y}{|\nabla\phi|}\right] dxdy$$

According to Green Equation: $\iint_D \left(\frac{\partial Q}{\partial x} - \frac{\partial P}{\partial y}\right) dxdy = \oint_L Pdx + Qdy$, and $h\big|_{\partial\Omega} = 0,$

we can get

$$\int_\Omega \frac{\partial}{\partial x}\left[\phi_x h\right] + \frac{\partial}{\partial y}\left[\phi_y h\right] dxdy = \oint_{\partial\Omega} h\left(\phi_x dy - \phi_y dx\right) = 0$$

变分法推导示例

$$\int_\Omega \frac{\partial}{\partial x}\left[\frac{\phi_x}{|\nabla\phi|}h\right]+\frac{\partial}{\partial y}\left[\frac{\phi_y}{|\nabla\phi|}h\right]dxdy=\oint_{\partial\Omega} h\left(\frac{\phi_x}{|\phi|}dy-\frac{\phi_y}{|\phi|}dx\right)=0$$

Then

$$\frac{\partial E(\phi+\delta h)}{\partial\delta}\Big|_{\delta\to 0}=-\int_\Omega h\phi_{xx}+h\phi_{yy}dxdy+\int_\Omega\left(\frac{\partial}{\partial x}\left[\frac{\phi_x}{|\nabla\phi|}\right]h+h\frac{\partial}{\partial y}\left[\frac{\phi_y}{|\nabla\phi|}\right]\right)dxdy$$

$$=-\int_\Omega h\triangle\phi dxdy+\int_\Omega h\nabla\bullet\left[\frac{\nabla\phi}{|\nabla\phi|}\right]dxdy$$

When $E(\phi)$ reach the minimal,

$$\frac{\partial E(\phi+\delta h)}{\partial\delta}\Big|_{\delta\to 0}=-\int_\Omega h\left(\triangle\phi-\nabla\bullet\left[\frac{\nabla\phi}{|\nabla\phi|}\right]\right)dxdy=0,$$

变分法推导示例

Since function h is arbitrary, we obtain:

$$\triangle\phi - \nabla\bullet\left[\frac{\nabla\phi}{|\nabla\phi|}\right] = 0$$

The above equation is the Euler-Lagrange Equation.

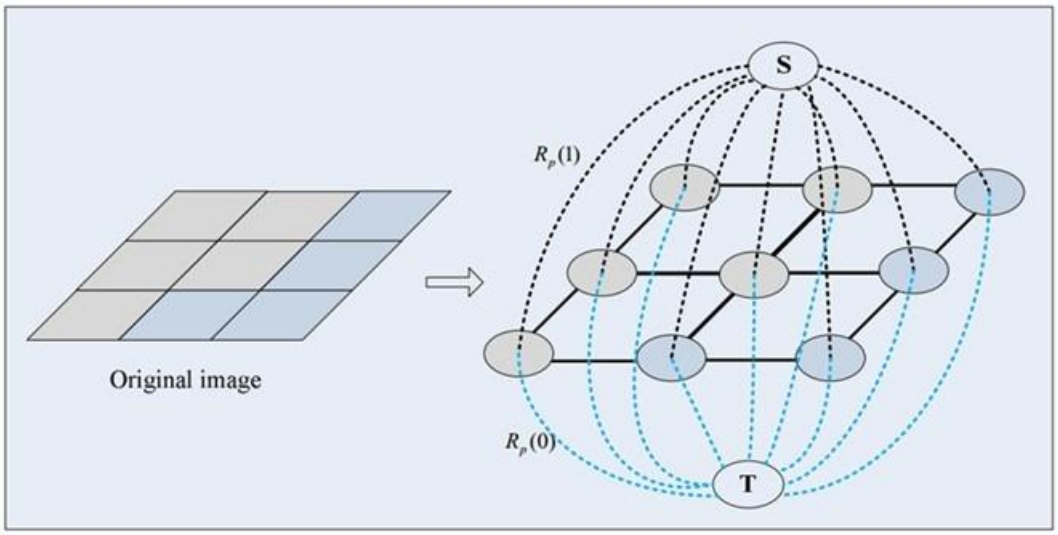
Generally, the gradient of functional $E(\phi)$ is denoted as $-\left(\triangle\phi - \nabla\bullet\left[\frac{\nabla\phi}{|\nabla\phi|}\right]\right)$,

algorithm, we get the LEVLE SET evolution equation:

$$\phi_t = \triangle\phi - \nabla\bullet\left[\frac{\nabla\phi}{|\nabla\phi|}\right]$$

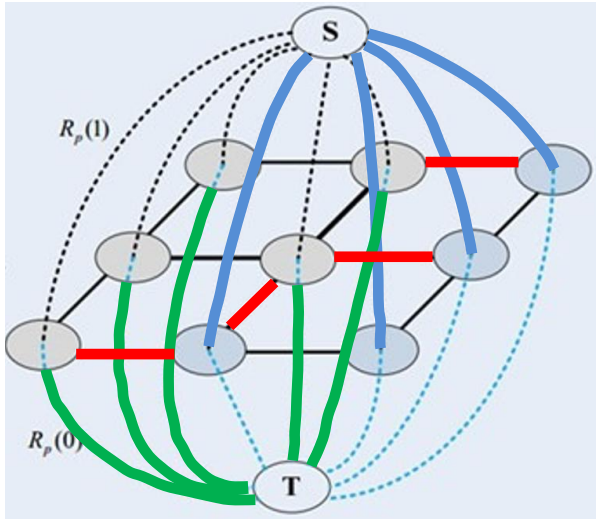
Graph Cut 图割法

- Graph cut：一种能量函数的优化方法
  - 基于图像定义graph（包括节/顶点和边）
  - 第一种顶点和边
    - ✓ 普通顶点，对应于图像中的每个像素。每两个四邻域顶点的连接就是一条边（图中实线），这种边也叫n-links。
  - 第二种顶点和边
    - ✓ 两个终端顶点，分别称作S（source：源点）和T（sink：汇点）。
    - ✓ 每个普通顶点和这2个终端顶点之间都有连接，组成第二种边（图中虚线），这种边也叫t-links。



Graph Cut

- Cut：一个cut（割）就是图中边集合E的一个子集C
  - 在图中去掉割(cut)后源点和汇点之间再没有通路，而加入割中的任一条边后都会产生一条通路。
- 图中每条边都有一个非负的权值w_e，也可以理解为cost（代价或者费用）
  - 割的cost（表示为$|C|$）就是边子集C的所有边的权值的总和。
- 最小割（min cut）：权值之和最小的割
  - 这个最小割把图的顶点划分为两个不相交的子集S和T，分别对应于图像的前景像素集和背景像素集。



图像分割问题转化为找最小割问题

图中边的权值就决定了最后的分割结果。那么给定一张图像，图中边的权值怎么确定呢？

能量函数

- 假设图像的分割为x时，图像的能量可以表示为：

$$E_x = E_{smooth}(x) + E_{data}(x)$$

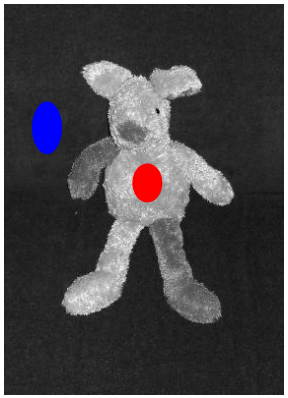
  - E_x：表示关于x的损失函数，也叫能量函数。图割的目标就是优化能量函数使其值达到最小。
  - Data项：普通顶点与终端顶点之间的边的权值
  - Smooth项：普通顶点之间的边的权值

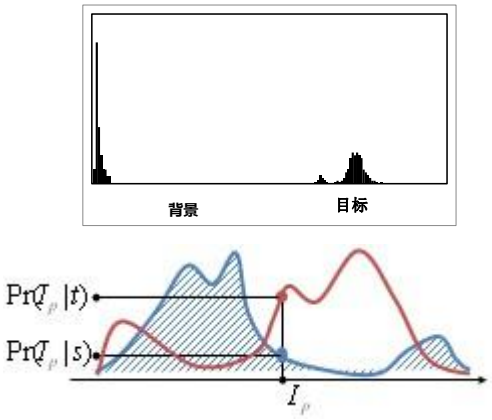
构造能量函数

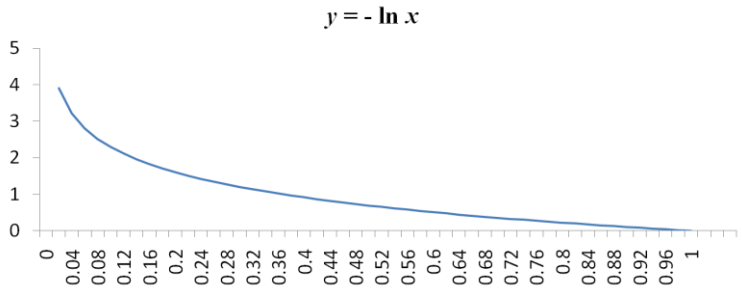
Data项

$$E^1(x_i) \propto -\ln \Pr(I_i|A_i)$$

$A_i \in \{s,t\}$ 表示像素 x_i 为前景或者背景

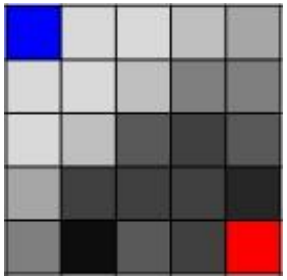






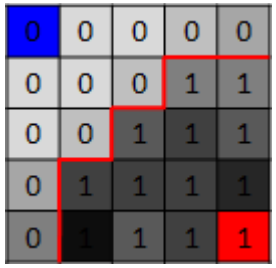
条件概率越高，能量越低

Smooth项：针对相邻像素



$$E^2(x_i,x_i) = \exp\left(-\frac{(I_i - I_j)^2}{2\sigma^2}\right) \cdot \delta(A_i, A_j)$$

$$\delta(A_i, A_j) = \begin{cases} 1 & \text{if } A_i \neq A_j \\ 0 & \text{if } A_i = A_j \end{cases}$$



x_i 与x_j相邻

优化（最小化）能量函数方法：穷举、模拟退火、graph cut……

构造能量函数

- Data项
  - 仅有data项，分割结果相当于用颜色的先验概率对像素进行分类
  - 分类的结果没有考虑相邻像素之间的空间关系，造成分割不连续

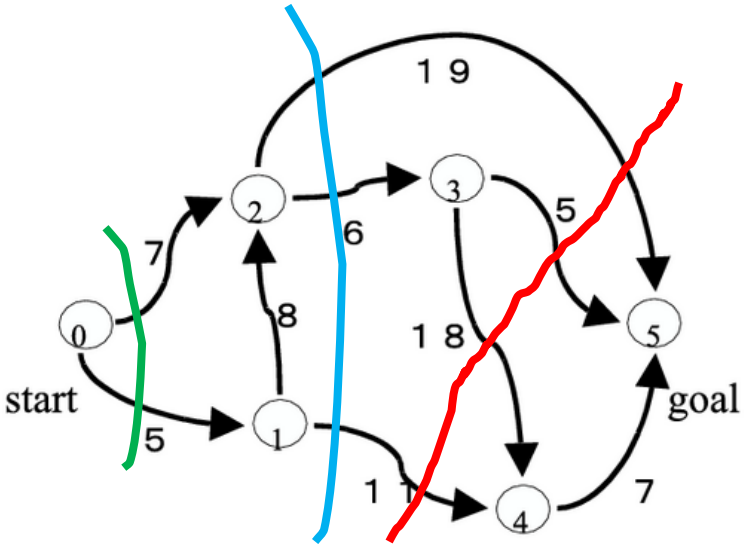
- Smooth项
  - 在data项提供先验信息后，再加入相邻像素间的约束关系，得到平滑的分割结果

$$E_{\boldsymbol{x}} = E_{data}(\boldsymbol{x}) + \lambda E_{smooth}(\boldsymbol{x})$$

Graph cut过程

1. 给定一个图像，我们需要构建一个图，图有两类顶点，两类边和两类权值。
2. 普通顶点由图像每个像素组成，然后每两个邻域像素之间存在一条边，它的权值由Smooth项来决定。
3. 两个终端顶点s（目标）和t（背景），每个普通顶点和终端顶点都存在连接，这种边的权值由Data项来决定。
4. 这样可以确定所有边的权值，即完成图的构建。
5. 找到最小的割，这个min cut就是**权值和**最小的边的集合，这些边的断开恰好可以使目标和背景被分割开。

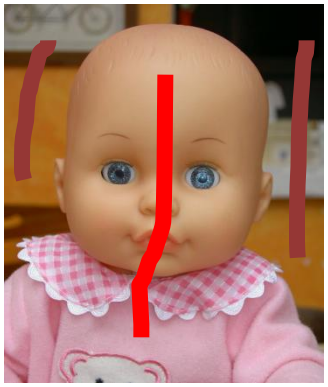
求最小割

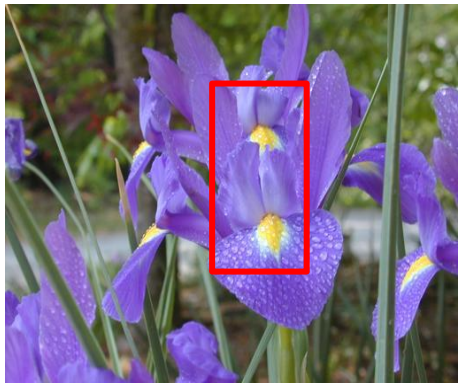


最大流最小割定理：在一个网络流中，能够从源点到达汇点的最大流量等于如果从网络中移除就能够导致网络流中断的边的集合的最小容量和。

- 最小割(min cut) -> 最大流(max flow)
  - 最大流最小割定理是网络流理论的重要定理
- 对于一个有向图（graph），权值当做弧的容量（最大流量）
- 这时从一点(start)到另一点(goal)存在最大流（max flow）
- 当流量达到最大，所有满流弧构成一个最小割（min cut）
- **通过不断增加网络的流量，即可达到最大流，从而找到最小割**
- 最小代价的cut可以以多项式时间的复杂度计算得到

采用Graph cut分割的结果

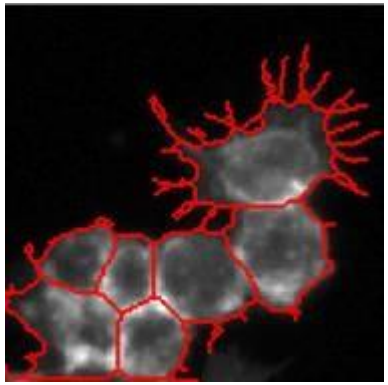




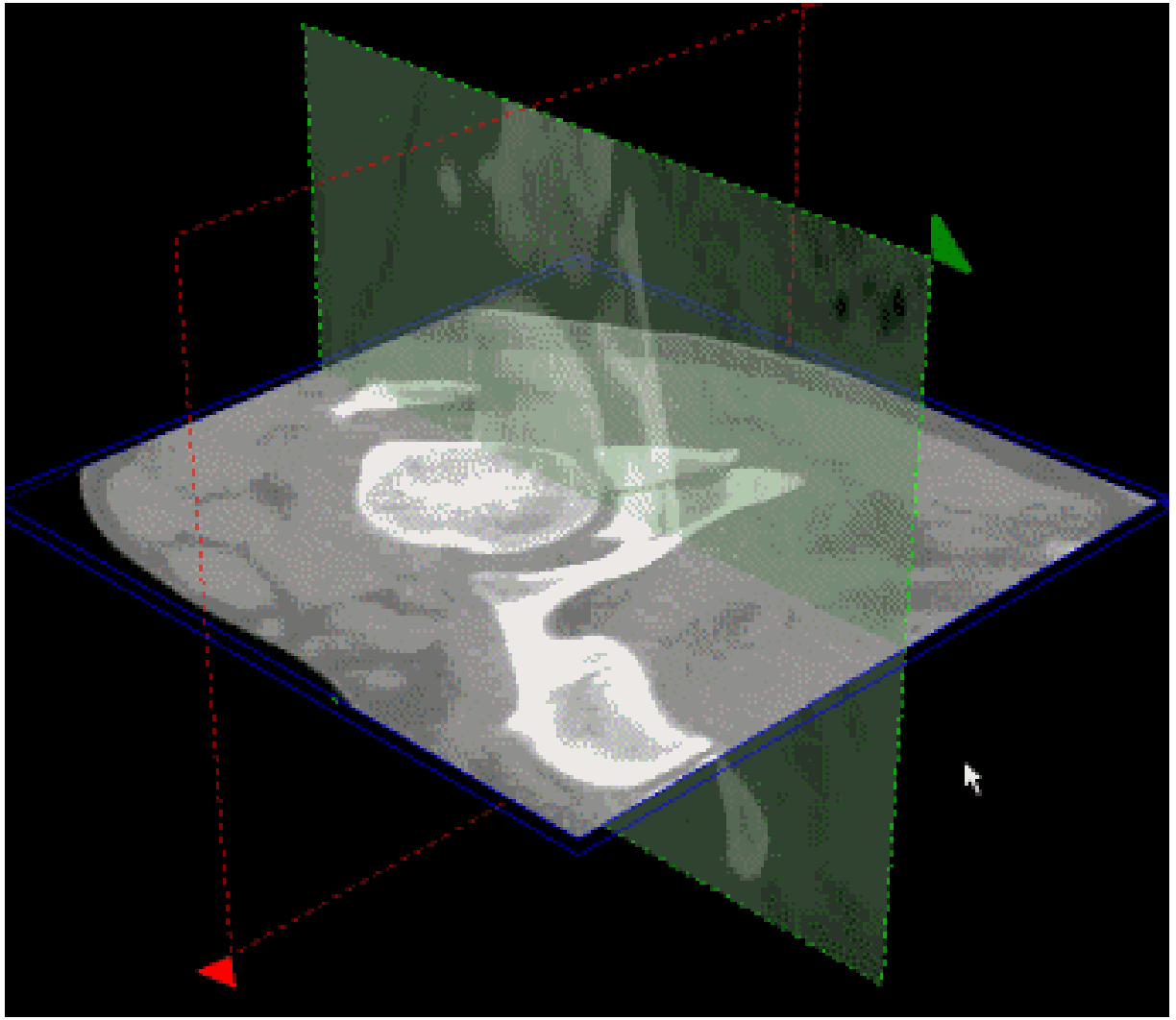








基于 Graph cut的3D 医学图像分割：bone

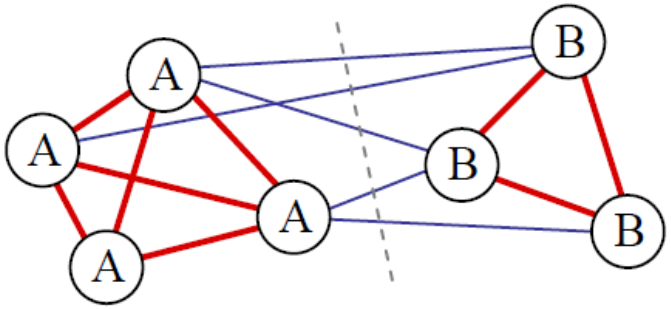


Normalized Cuts

- 将图像像素视为节点，相邻像素由边连接，如下图所示
  - 两个节点（像素）之间的边的权值由像素间相似性决定
- Normalized cuts：如下图虚线所示，将图中节点分为两组
  - 分割评价准则：

$$Ncut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)}$$

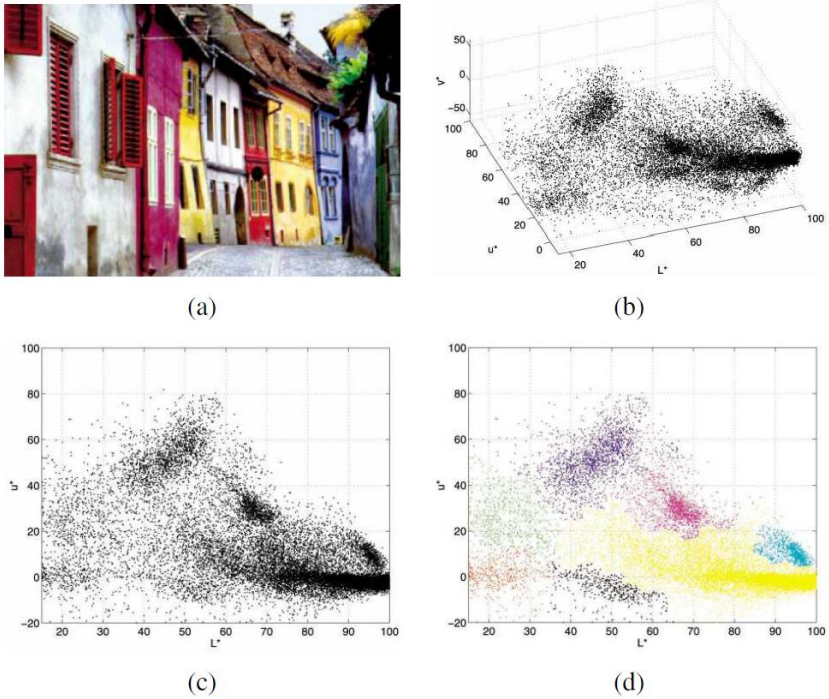
$$assoc(A,A) = \sum_{i\in A, j\in A} w_{ij} \qquad assoc(A,V) = assoc(A,A) + cut(A,B)$$

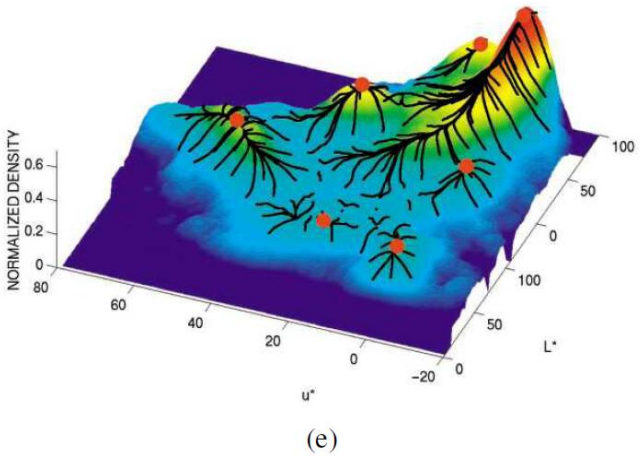


	A	B	sum
A	$assoc(A,A)$	$cut(A,B)$	$assoc(A,V)$
B	$cut(B,A)$	$assoc(B,B)$	$assoc(B,V)$
sum	$assoc(A,V)$	$assoc(B,V)$	

Mean Shift

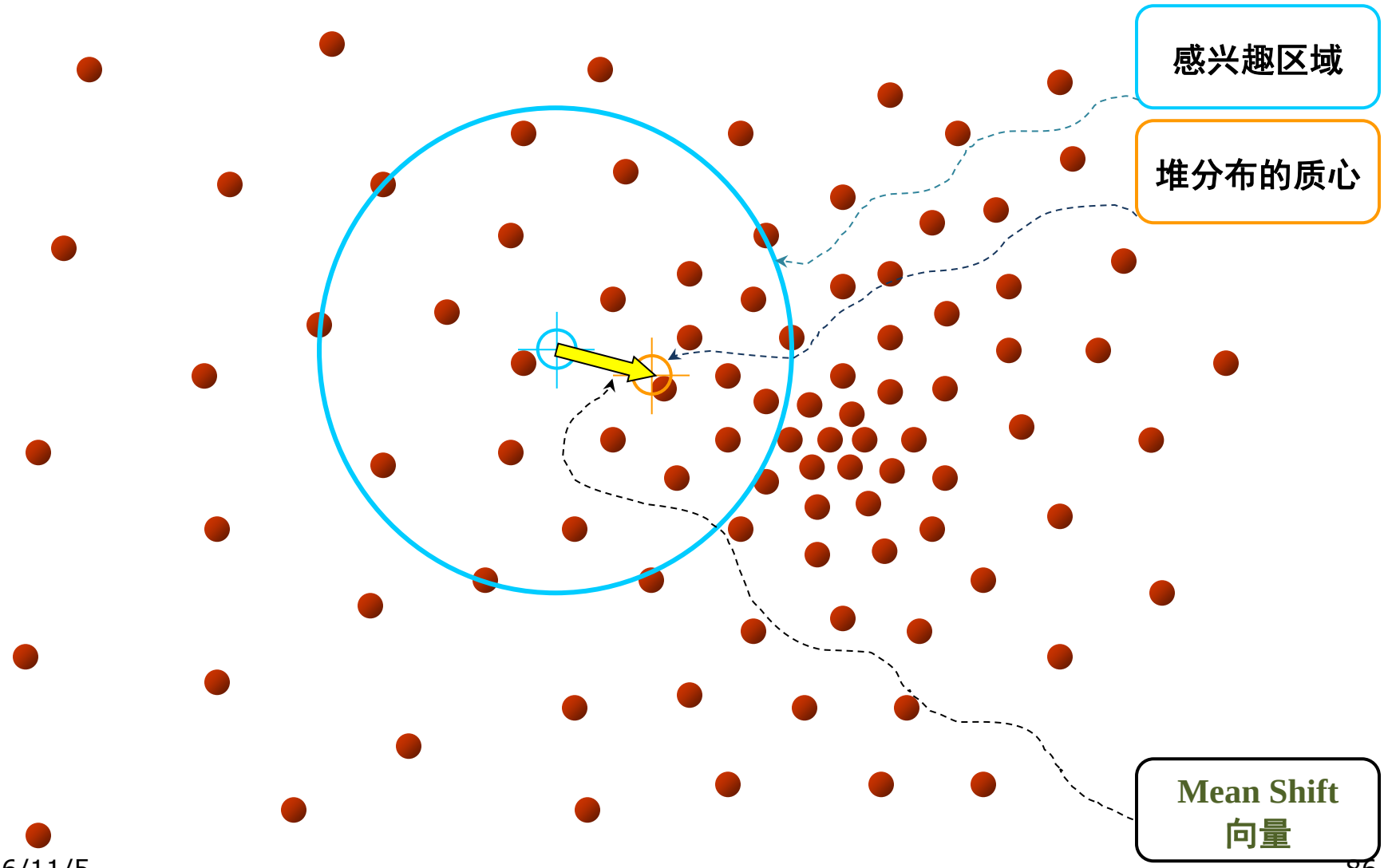
- 基本思想：在高维的数据分布中寻找模值点(peak)，通过梯度收敛于同一个模值点的像素被视为属于同一分割区域
  - 由于高维分布空间非常稀疏，常通过Parzen窗来平滑数据，从而估计数据密度





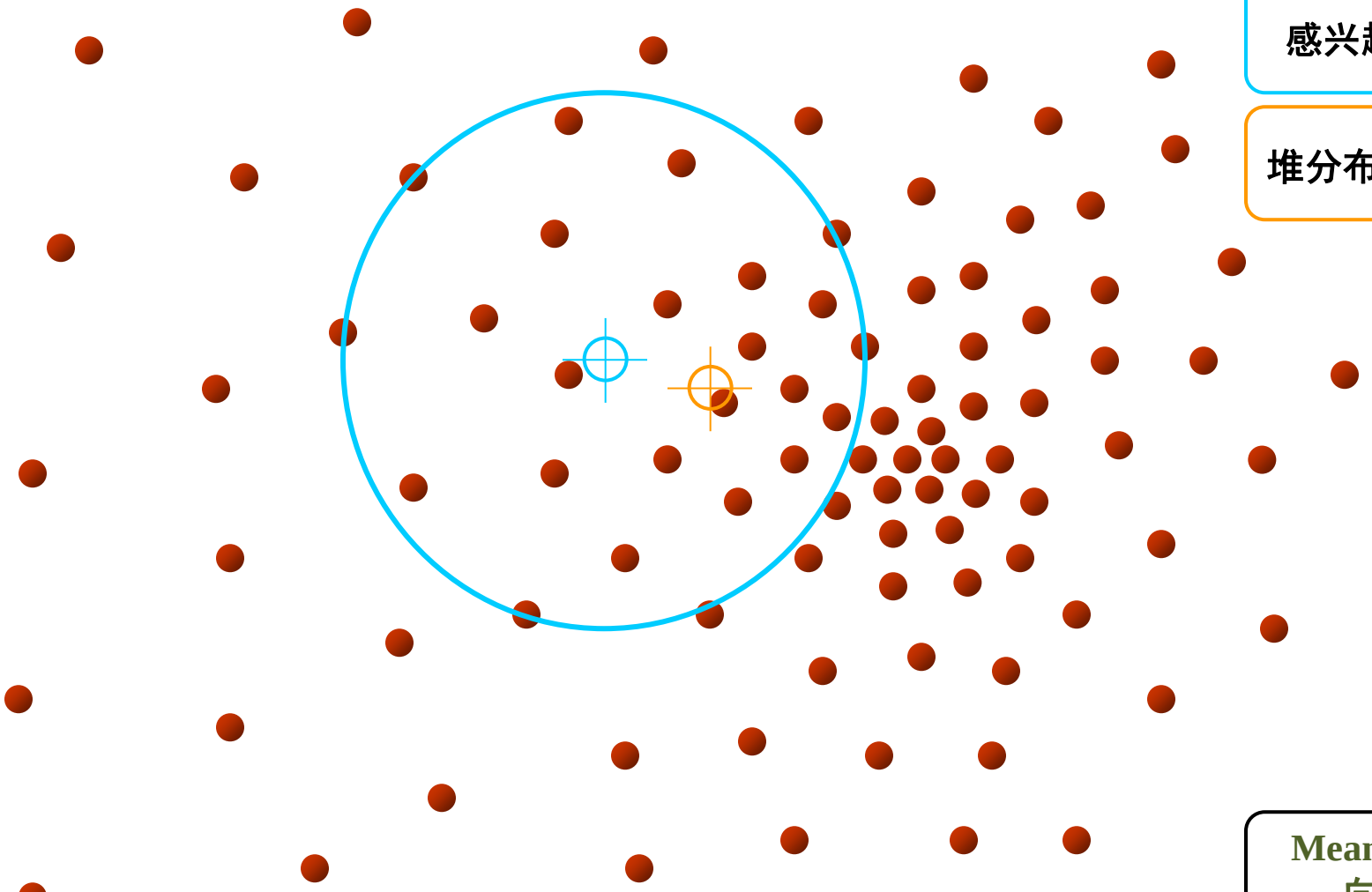
Mean Shift：直观展示

- 目标：找到密度最大的区域



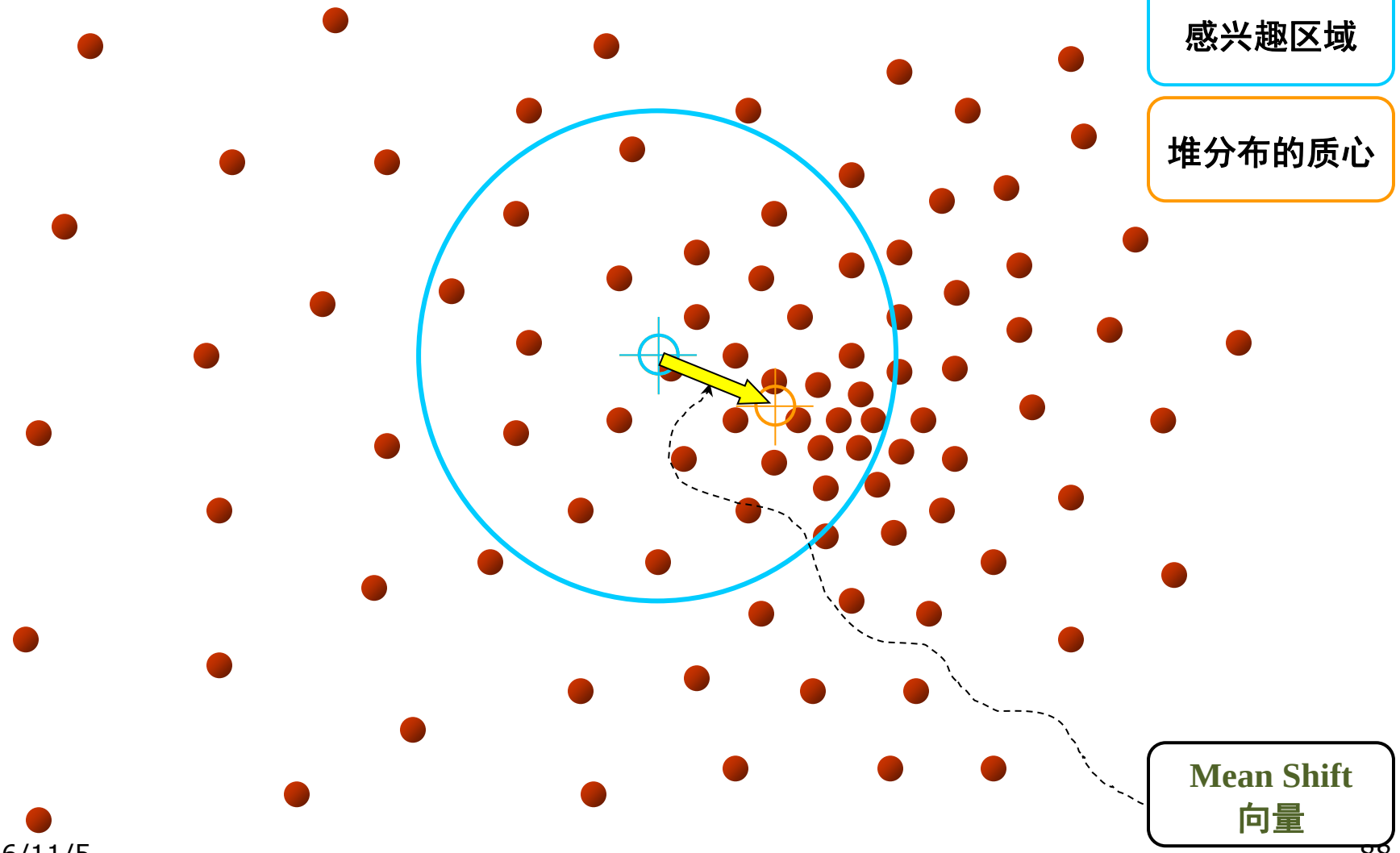
Mean Shift：直观展示

- 目标：找到密度最大的区域



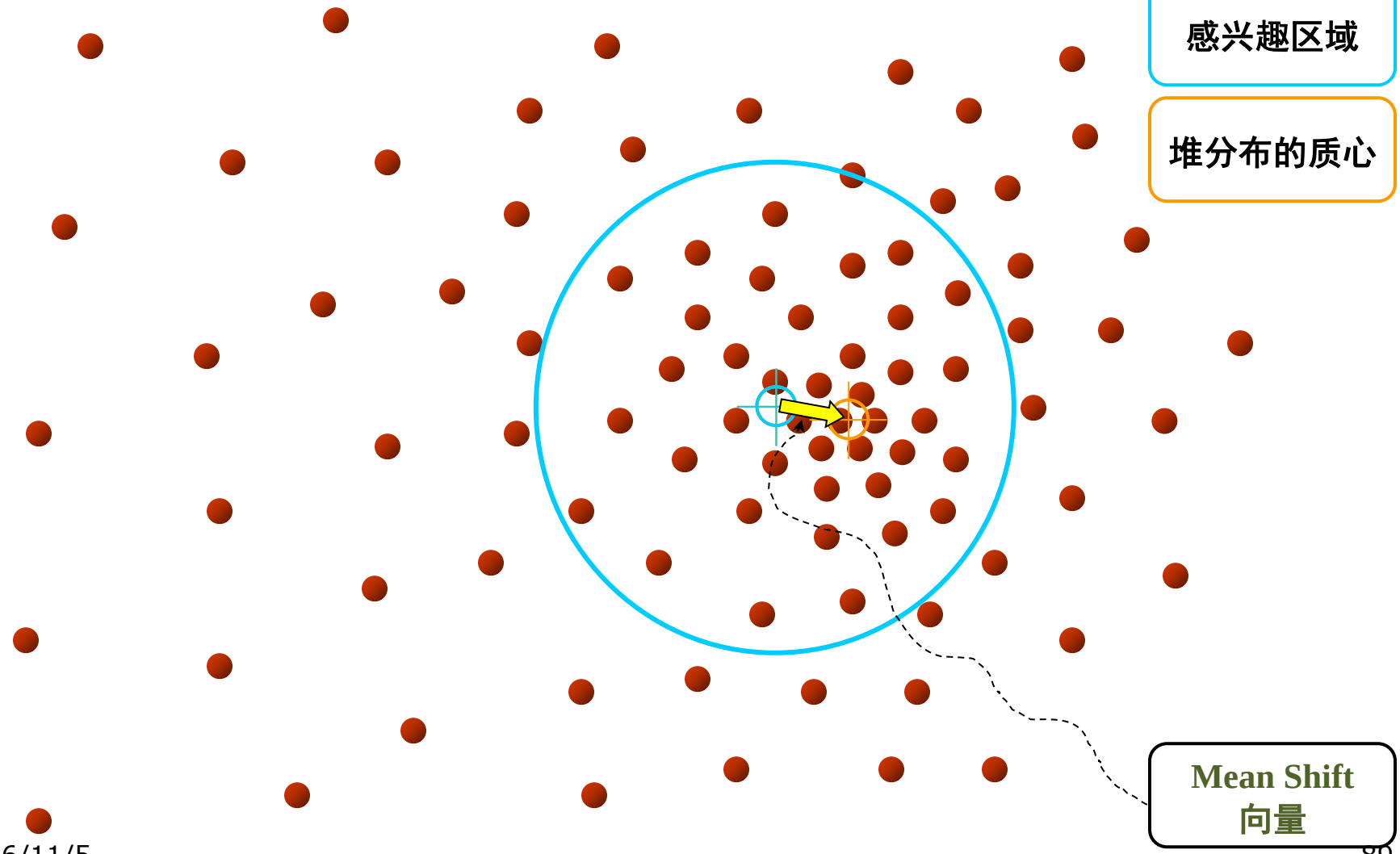
Mean Shift：直观展示

- 目标：找到密度最大的区域



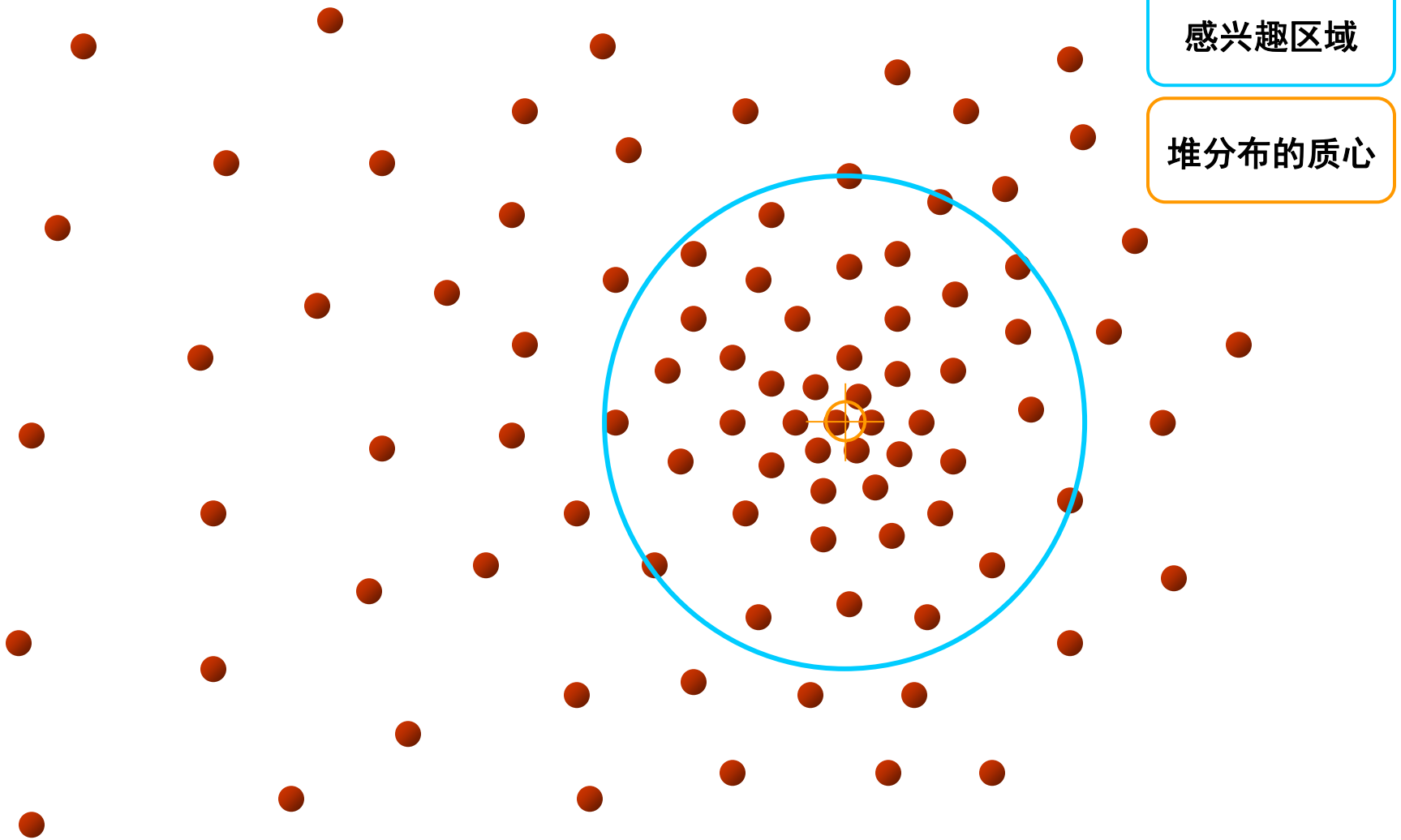
Mean Shift：直观展示

- 目标：找到密度最大的区域



Mean Shift：直观展示

- 目标：找到密度最大的区域

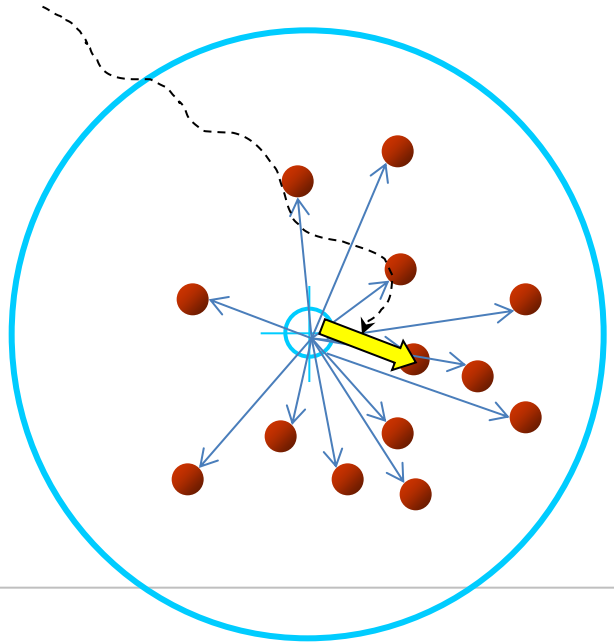


Mean Shift

- 直观图示: Mean Shift向量的方向指向概率密度函数高的方向
- 定义: D维空间中的n个样本点$\{\boldsymbol{x}_i, i = 1,2,\cdots,n\}$,在点$\boldsymbol{x}$处的Mean Shift向量的基本定义:

$$M(\boldsymbol{x}) \equiv \frac{1}{k} \sum_{x_i \in S_k} (\boldsymbol{x}_i - \boldsymbol{x})$$

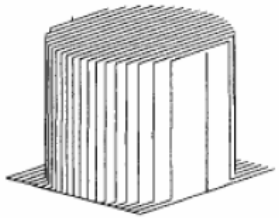
  - S_k是感兴趣区域
  - k是感兴趣区域内的样本的数目



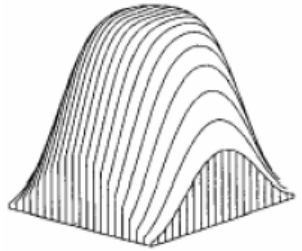
Extended Mean Shift

- 引入两个参数
  - 核函数
  - 权值

- 核函数
  - 定义$\boldsymbol{x} \in \mathcal{R}^D, \|\boldsymbol{x}\|^2 = \boldsymbol{x}^T\boldsymbol{x}$。若函数$K(\boldsymbol{x})$ 存在一个剖面函数$k(\cdot): [0, +\infty) \to \mathcal{R}$，即$K(\boldsymbol{x}) = k(\|\boldsymbol{x}\|^2)$，并且$k(r)$满足
    - ✓ 非负的
    - ✓ 非增的
    - ✓ 分段连续的，并且$\int_0^{+\infty} k(r)dr < +\infty$
  - 常用的核函数
    - ✓ 单位均匀核函数
    - ✓ 单位高斯核函数



$$F(x) = \begin{cases} 1, if \ \|x\| < 1 \\ 0, if \ \|x\| \geq 1 \end{cases}$$



$$F(x) = e^{-\|x\|^2}$$

Extended Mean Shift

- Mean Shift扩展形式：

$$m(x)=\frac{\sum_{i=1}^{n} g\left(\left\|\frac{x-x_i}{h}\right\|^2\right)\omega(x_i)(x_i-x)}{\sum_{i=1}^{n} g\left(\left\|\frac{x-x_i}{h}\right\|^2\right)\omega(x_i)}$$

- 简化运算：

$$m(x)=\frac{\sum_{i=1}^{n} x_i g\left(\left\|\frac{x-x_i}{h}\right\|^2\right)\omega(x_i)}{\sum_{i=1}^{n} g\left(\left\|\frac{x-x_i}{h}\right\|^2\right)\omega(x_i)}-x$$

Mean Shift向量的方向和样本点的概率密度函数的梯度方向是一致的

核密度梯度估计

- 给定 D 维空间中 n 个样本点$\boldsymbol{x}_i$，$f(x)$的核密度估计(也称parzen窗估计)为

$$\hat{f}(x)=\frac{1}{n}\sum_{i=1}^{n}K_H(x_i-x)$$

- 假设所有的样本点从同一个密度函数采样，对核函数进行简化和引入权重：

$$\hat{f}(x)=\frac{\sum_{i=1}^{n}k\left(\left\|\frac{x_i-x}{h}\right\|^2\right)w(x_i)}{h^d\sum_{i=1}^{n}w(x_i)}$$

- 概率密度函数$f(x)$的梯度估计为：

$$\nabla f(x)\approx\nabla\hat{f}(x)=\frac{2\sum_{i=1}^{n}(x-x_i)k'\left(\left\|\frac{x_i-x}{h}\right\|^2\right)w(x_i)}{h^{2+d}\sum_{i=1}^{n}w(x_i)}$$

核密度梯度估计

- 概率密度函数f(x)的梯度估计

$$\hat{f}(x)=\frac{\sum_{i=1}^{n}k\left(\left\|\frac{x_i-x}{h}\right\|^2\right)w(x_i)}{h^d\sum_{i=1}^{n}w(x_i)}$$

$$\nabla f(x)\approx\nabla\hat{f}(x)=\frac{2\sum_{i=1}^{n}(x-x_i)k'\left(\left\|\frac{x_i-x}{h}\right\|^2\right)w(x_i)}{h^{2+d}\sum_{i=1}^{n}w(x_i)}$$

$$g(x)=-k'(x)$$

$$G(x)=g(\|x\|^2)$$

$$=\frac{2}{h^2}\left[\frac{\sum_{i=1}^{n}g\left(\left\|\frac{x_i-x}{h}\right\|^2\right)w(x_i)}{h^d\sum_{i=1}^{n}w(x_i)}\right]\left[\frac{\sum_{i=1}^{n}g\left(\left\|\frac{x_i-x}{h}\right\|^2\right)w(x_i)x_i}{\sum_{i=1}^{n}g\left(\left\|\frac{x_i-x}{h}\right\|^2\right)w(x_i)}-x\right]$$

Mean Shift算法

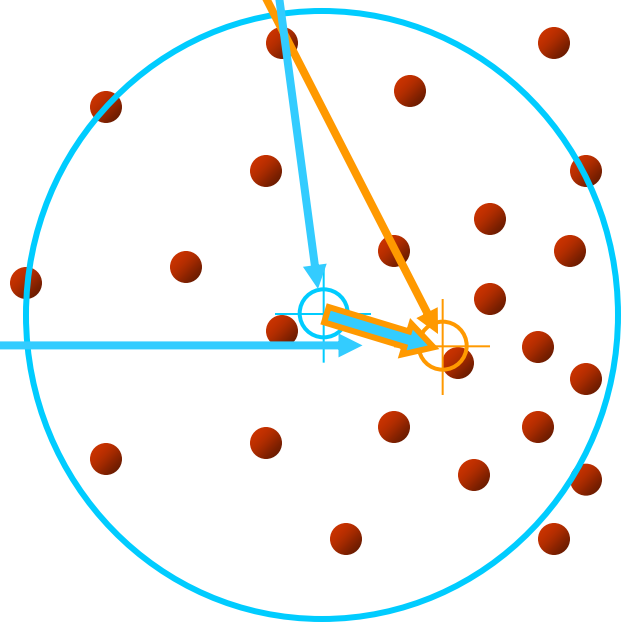
$$\hat{\nabla} f(x) = \frac{2}{h^2}\left[\frac{\sum_{i=1}^{n} g\left(\left\|\frac{x_i - x}{h}\right\|^2\right) w(x_i)}{h^d \sum_{i=1}^{n} w(x_i)}\right]\left[\frac{\sum_{i=1}^{n} g\left(\left\|\frac{x_i - x}{h}\right\|^2\right) w(x_i) x_i}{\sum_{i=1}^{n} g\left(\left\|\frac{x_i - x}{h}\right\|^2\right) w(x_i)} - x\right] = \frac{2}{h^2}\hat{f}_{h,G}(x) m_{h,G}(x)$$

$$m_{h,G}(x) = \frac{h^2}{2}\frac{\hat{\nabla} f_{h,K}(x)}{\hat{f}_{h,G}(x)}$$

Mean Shift向量方向和样本点的概率密度函数梯度方向是一致的

简单的 Mean Shift 迭代算法:

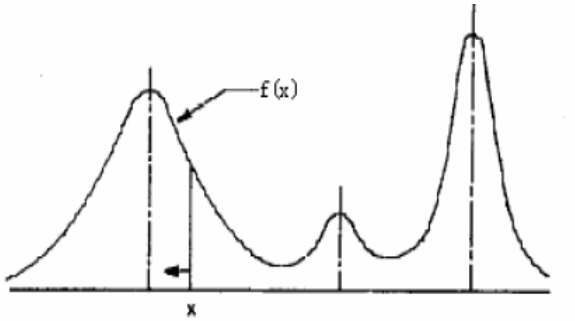
1. **计算 Mean Shift 矢量**
2. **把当前窗口移动 m(x)**
3. **返回1直至满足条件时收敛**



Mean Shift：峰值检测

- Mean Shift 总是指向密度增大最快的方向；当到达密度最大时，Mean Shift为0
  - 给定一组样本，这些样本服从的分布可以估计出来，假设如图所示
  - 给定一个初始点，那么该点就会一步步的移动直至寻到概率极大值点

$$m_{h,G}(x) = \frac{h^2}{2} \frac{\hat{\nabla} f_{h,K}(x)}{\hat{f}_{h,G}(x)}$$



- Mean Shift :
  - 本质是一个自适应的梯度上升搜索峰值的方法，能处理多模态分布
  - 用途:
    - ✓ 寻找模值点：任意初始点，运用mean shift可以找到极值点
    - ✓ 聚类：对于收敛到同一点的，聚为一类即可
    - ✓ 最优化：最优化目标转化为mean shift隐含估计的概率密度函数

Mean Shift：特征选择

特征选择

- 特征：空间位置、颜色、纹理、形状等。
- Mean Shift做平滑和分割中用到的特征：
  - ✓ 空间
  - ✓ 颜色
- 特征空间：联合特征 = 空间位置 + 颜色

$$K(x) = k(\|x\|^2) \xrightarrow{\text{联合特征}} K(\mathbf{x}) = C \cdot k_s\left(\left\|\frac{\mathbf{x}^s}{h_s}\right\|\right) \cdot k_r\left(\left\|\frac{\mathbf{x}^r}{h_r}\right\|\right)$$

物理意义：把图像看做的空间和颜色域的像素点集合

Mean Shift：平滑

- 示例

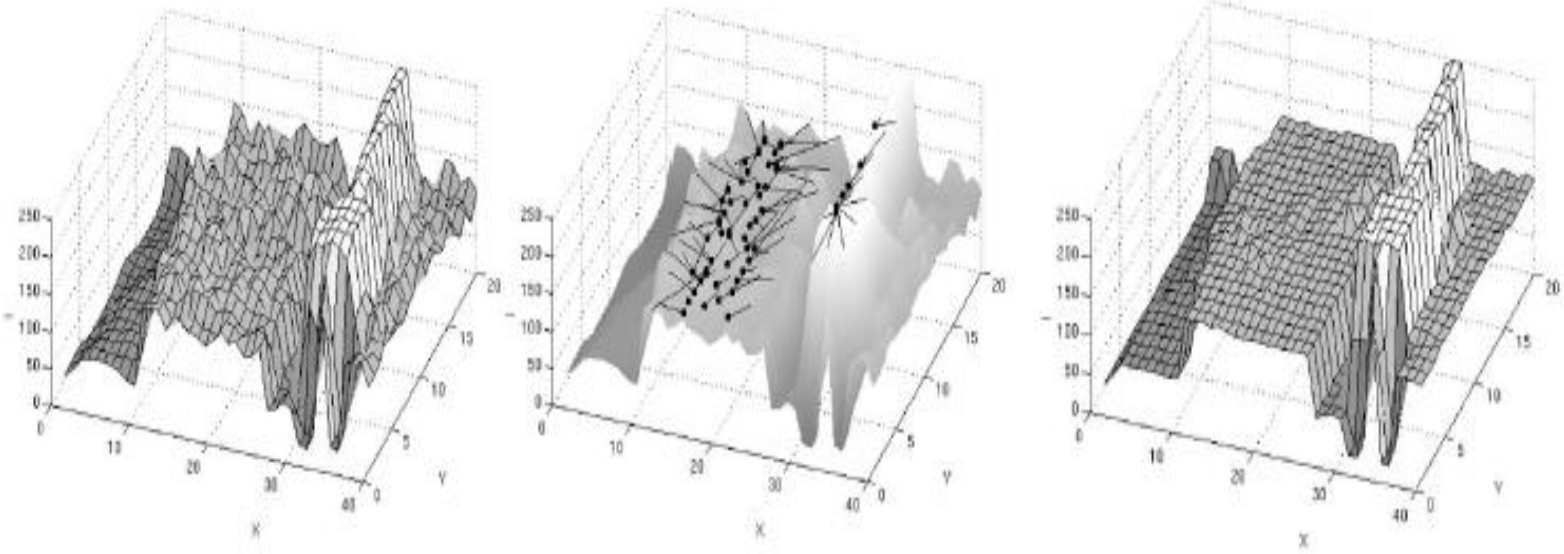
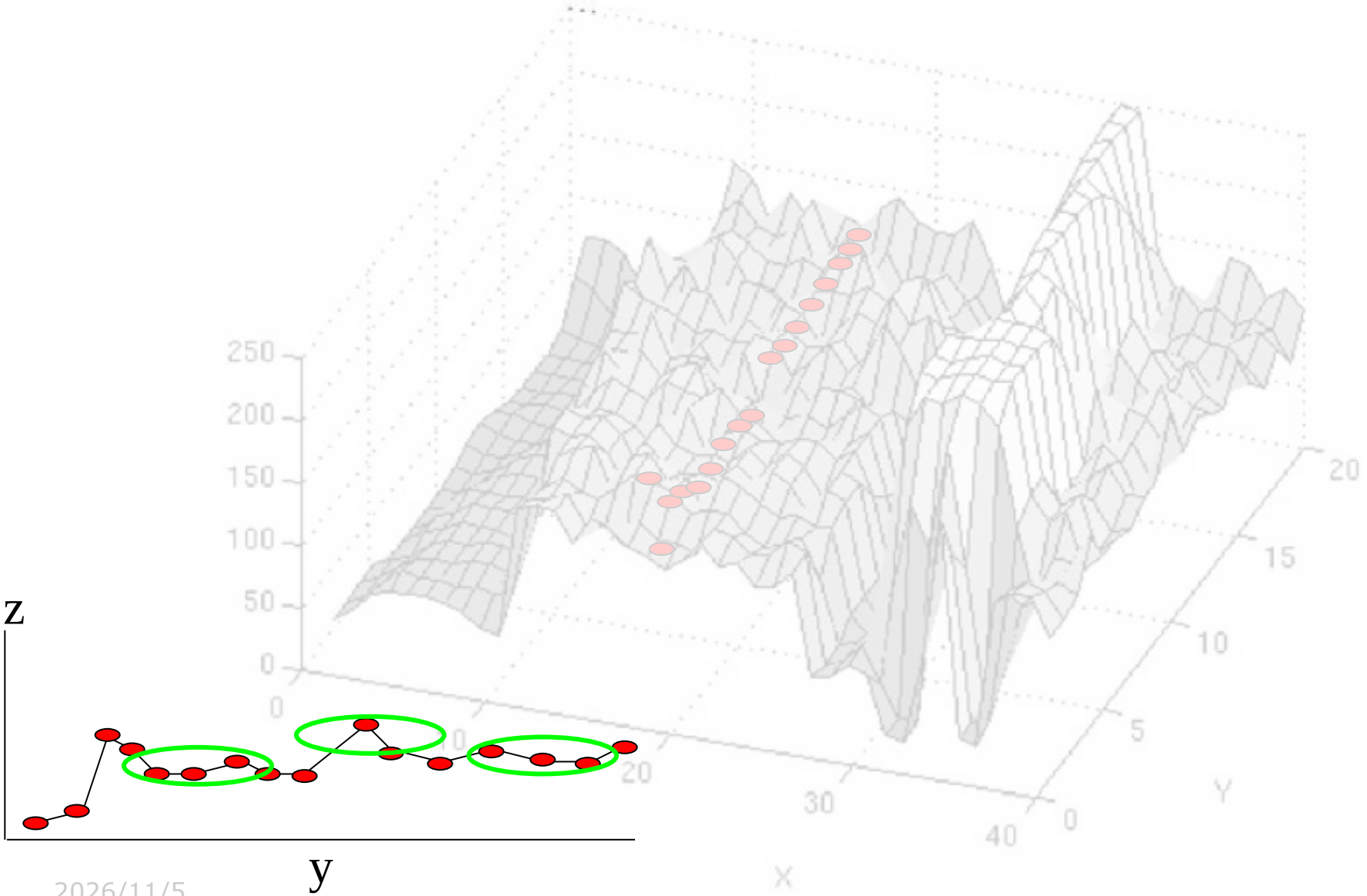


Image Data (slice)

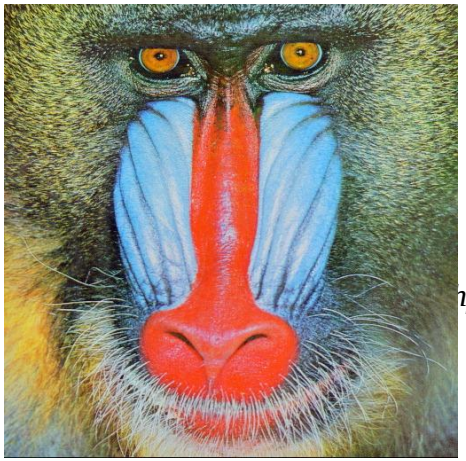
Mean Shift vectors

Smoothing result

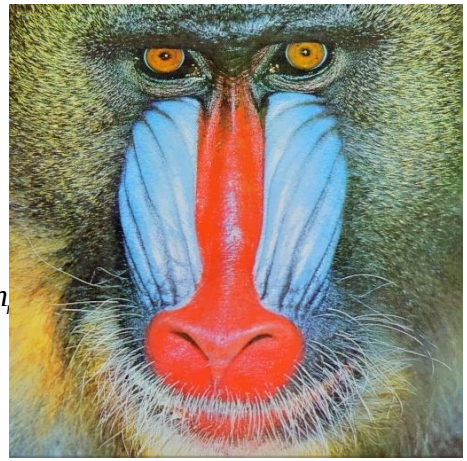
Mean Shift：平滑



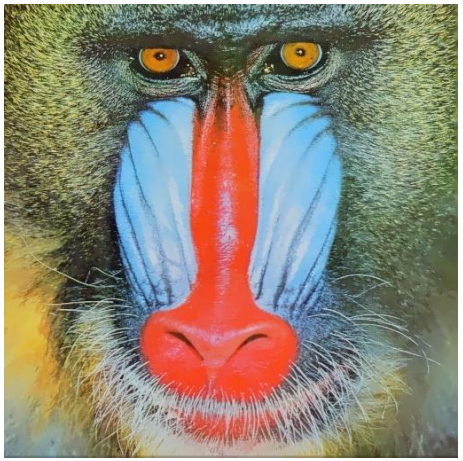
Mean Shift：平滑结果



Original baboon



$(h_s, h_r) = (8,16)$



$(h_s, h_r) = (8,40)$

$$K(\mathbf{x}) = C \cdot k_s\left(\left\|\frac{\mathbf{x}^s}{h_s}\right\|\right) \cdot k_r\left(\left\|\frac{\mathbf{x}^r}{h_r}\right\|\right)$$



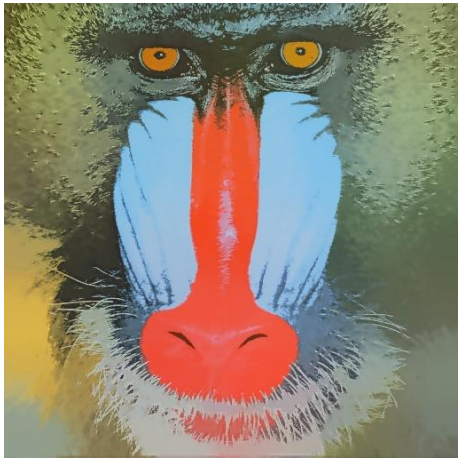
$(h_s, h_r) = (16,16)$



$(h_s, h_r) = (16,40)$



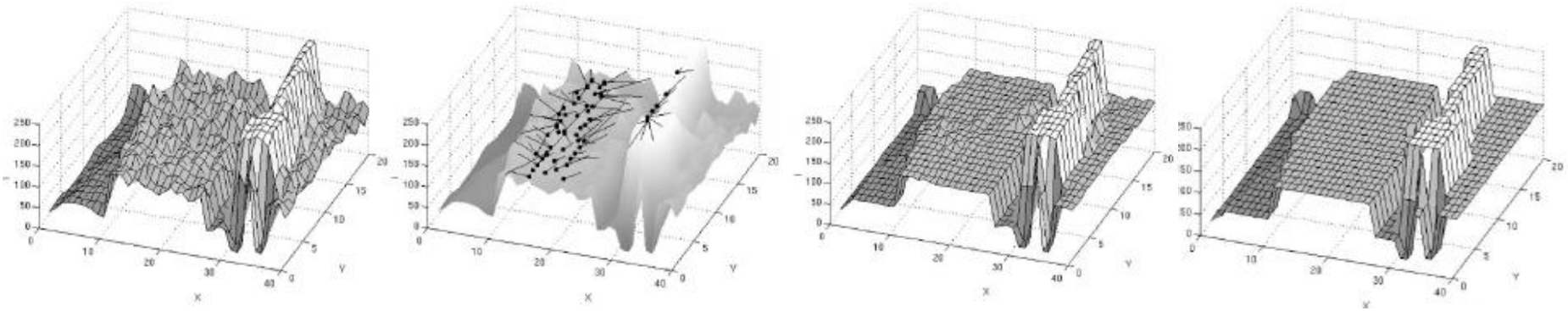
$(h_s, h_r) = (32,16)$



$(h_s, h_r) = (32,40)$

Mean Shift：图像分割

- 在平滑的基础上，分割算法的后续步骤：
  - 把这样的一些模值点聚集起来，如果它之间的距离小于空间带宽半径，并且颜色差值小于颜色带宽半径
  - 除去这样的一些区域，包含的像素数目少于M个



Mean Shift：图像分割

- 分割结果
  - 当特征空间仅仅是颜色信息时：





Mean Shift：图像分割

- 分割结果

