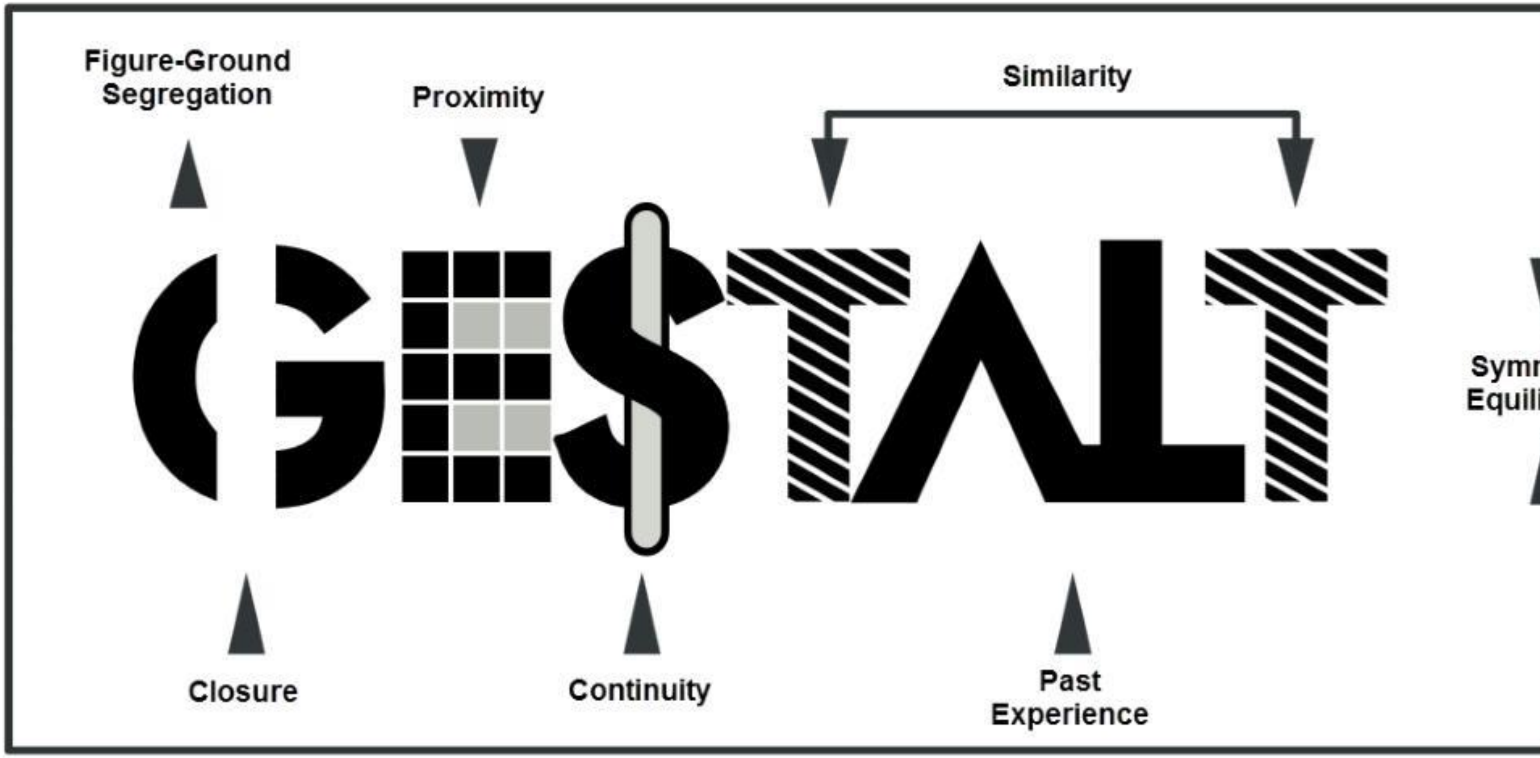
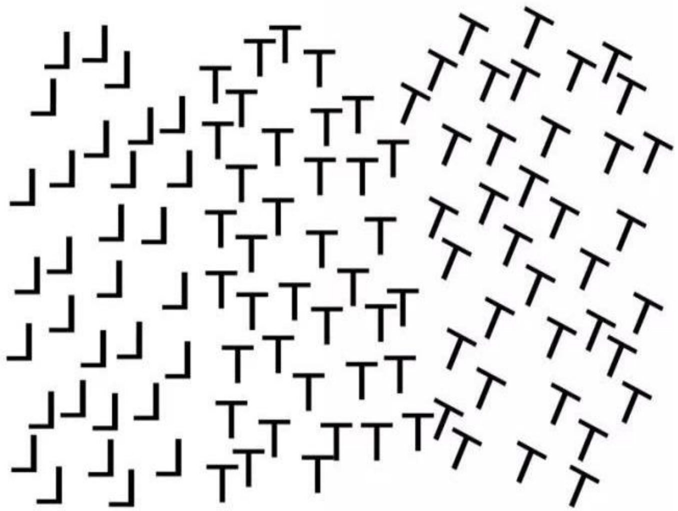
Image Reorganization



盖式塔原理是指，人类感知能够从在对象或概念的不完全表示中总结出模式，并能从这些模式中推断出整体的本质。

Image Reorganization



相似律

接近律

Image Reorganization

连续律

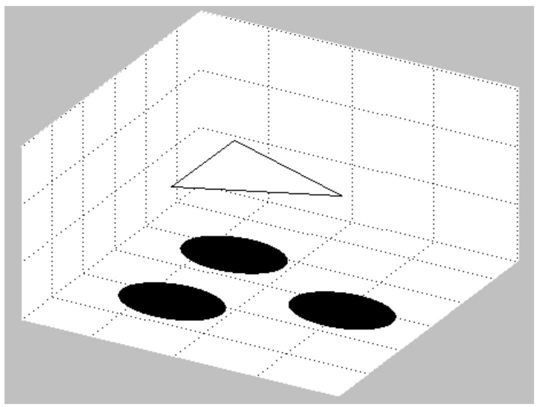
平衡律

Image Reorganization

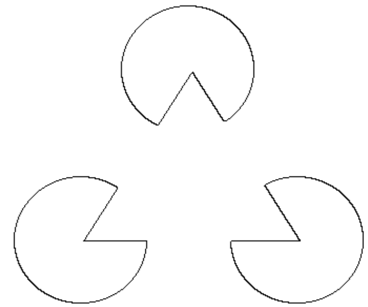
图底律

闭合律

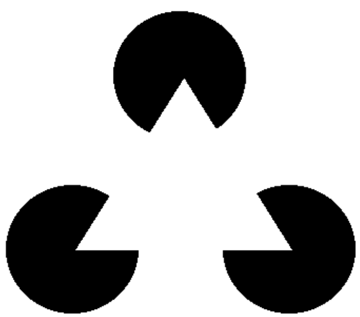
Image Reorganization



场景图



灰度轮廓



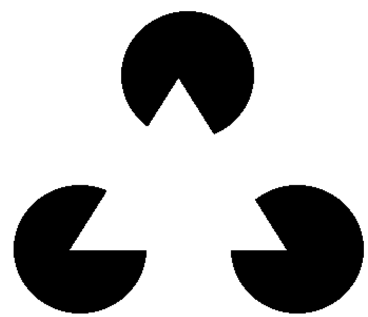
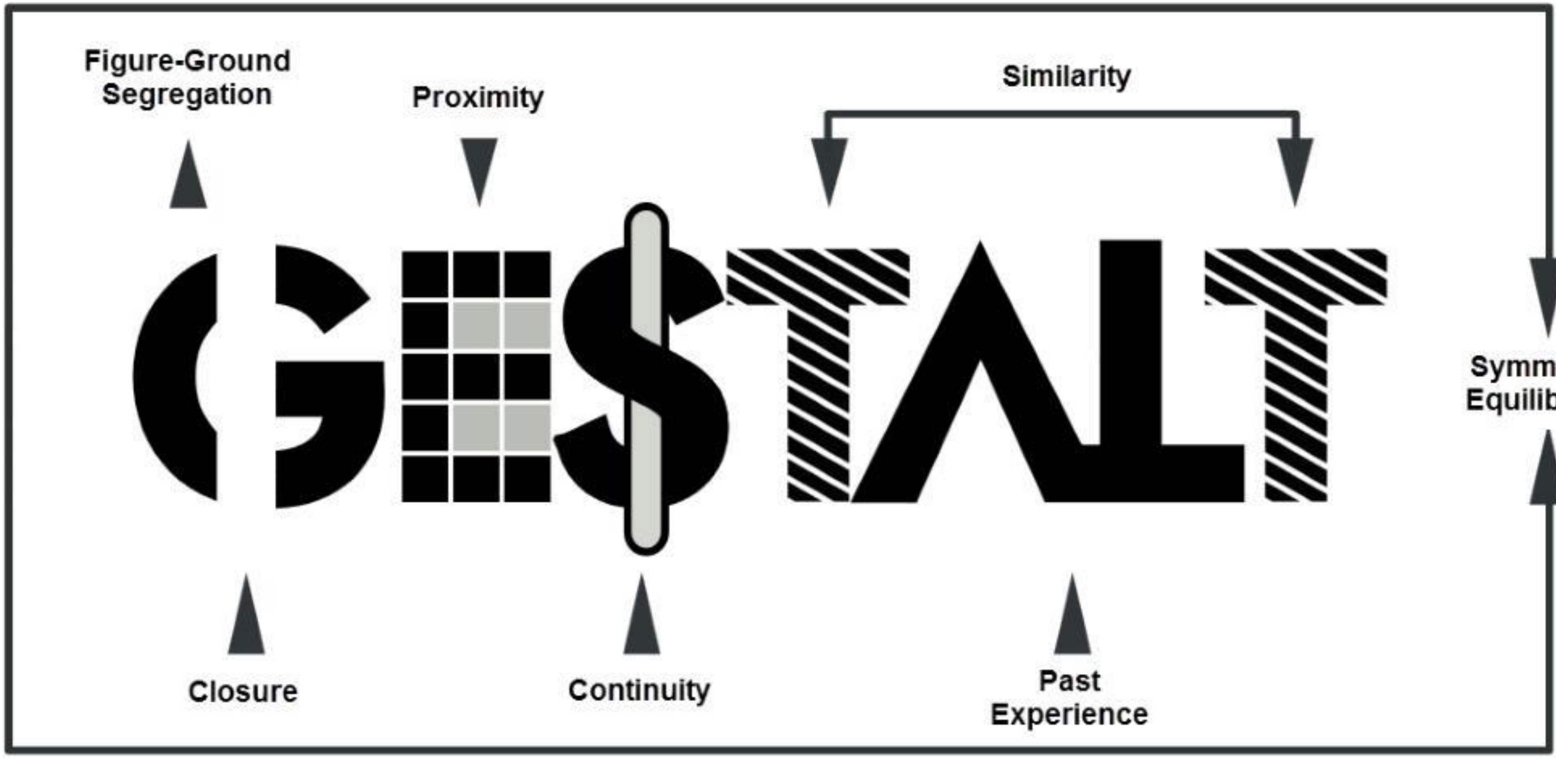
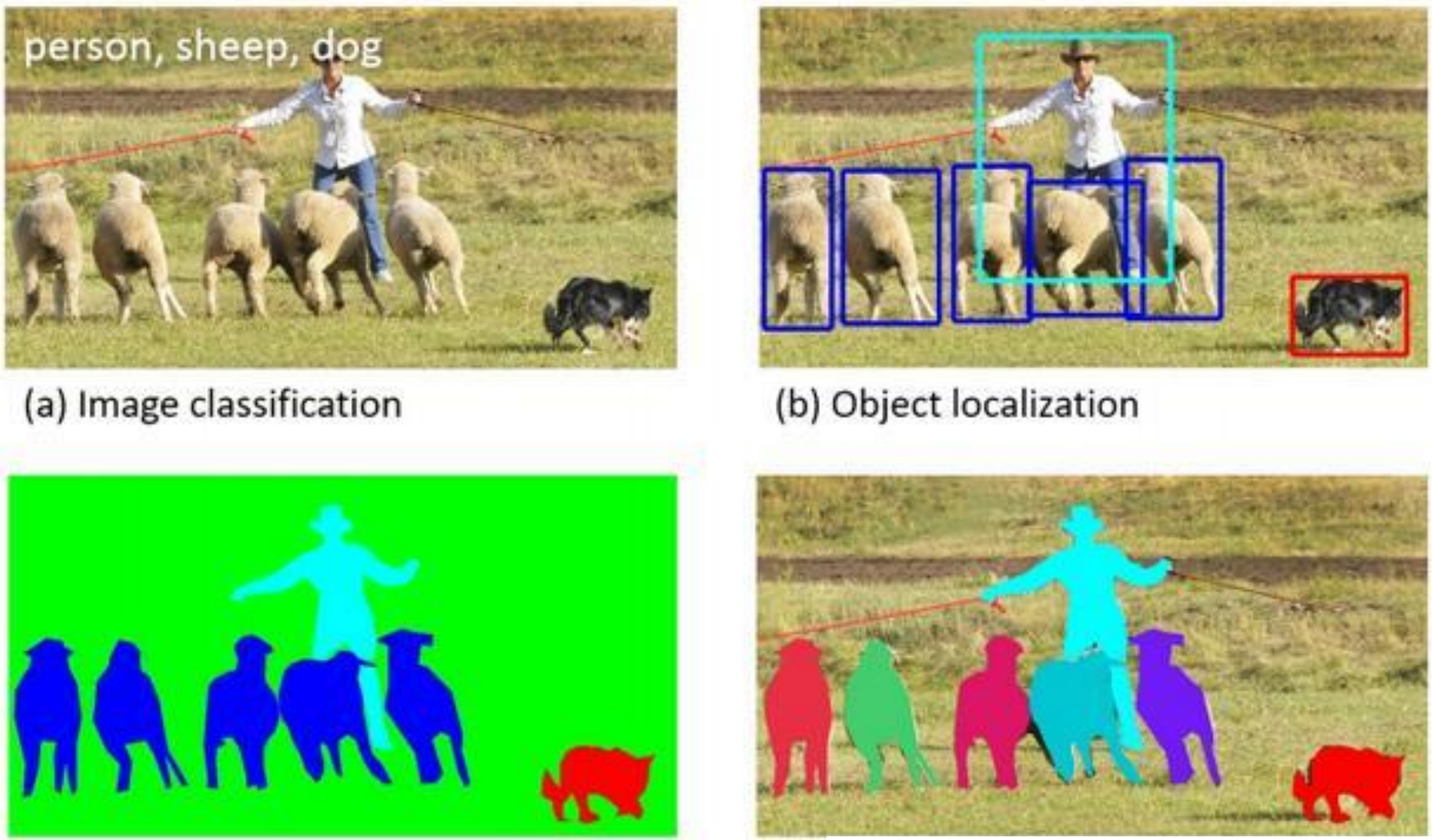


Image Reorganization



盖式塔原理是指，人类感知能够从在对象或概念的不完全表示中总结出模式，并能从这些模式中推断出整体的本质。

Image Segmentation



(a) Image classification

(b) Object localization

(c) Semantic segmentation

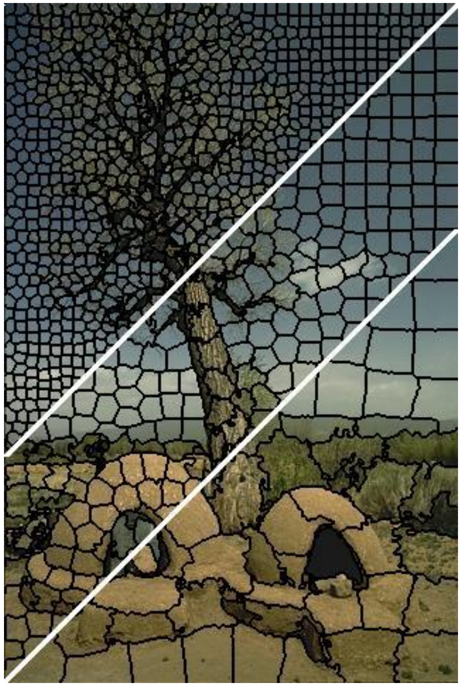
(d) Instance Segmentation

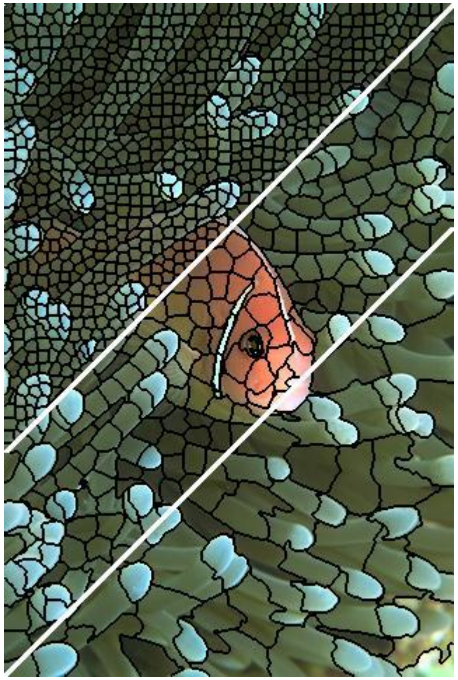
Image Reorganization

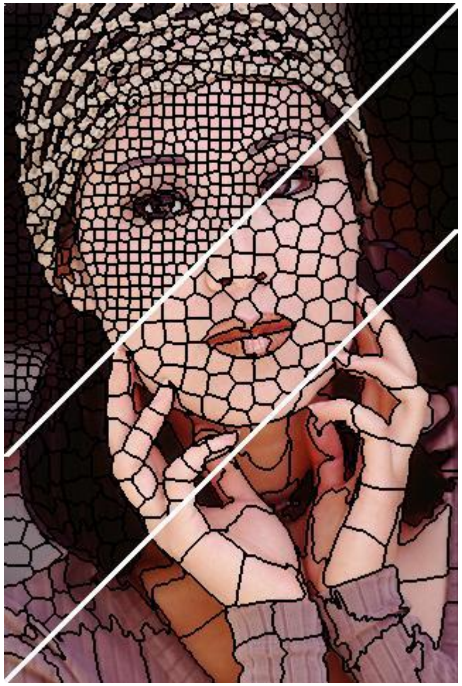
- Cluttering-based methods
  - Mean Shift
  - SLIC (Super-pixel)
- Graph-based methods
  - Graph cut
  - Ncuts
- Interaction-based methods
  - Grab Cut
  - Active Contour
- Learning-based methods

Image Segmentation

- Cluttering-based segmentation – SLIC (TPAMI'12)







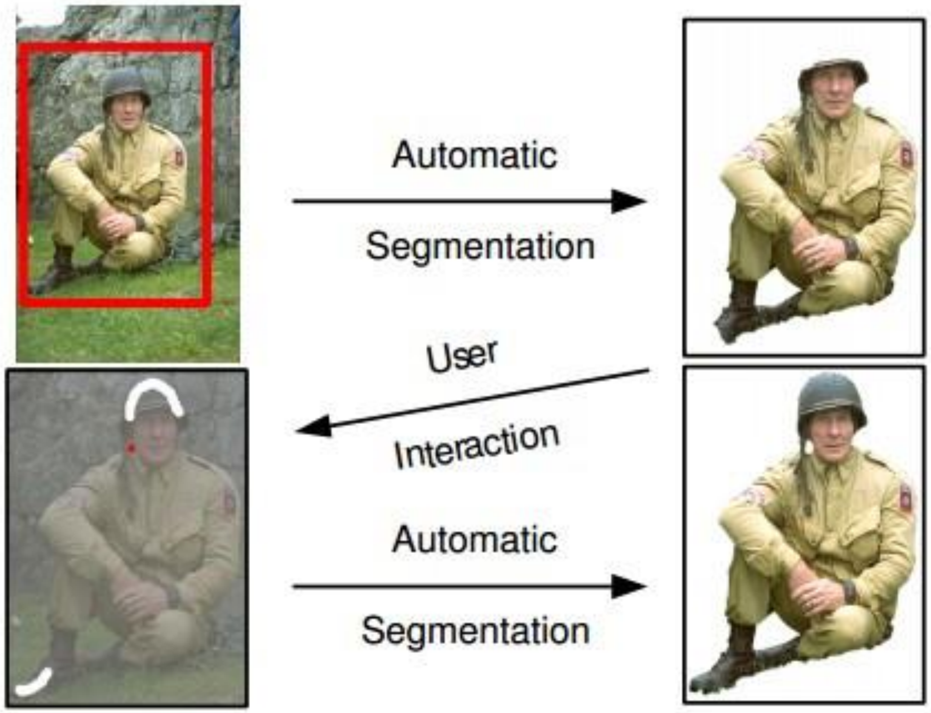
https://ivrl.epfl.ch/research-2/research-current/research-superpixels/

Image Segmentation

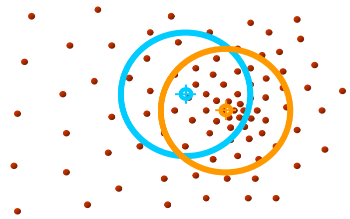
- Graph-based segmentation

Image Segmentation

- Interaction segmentation



Mean Shift Segmentation Algorithm

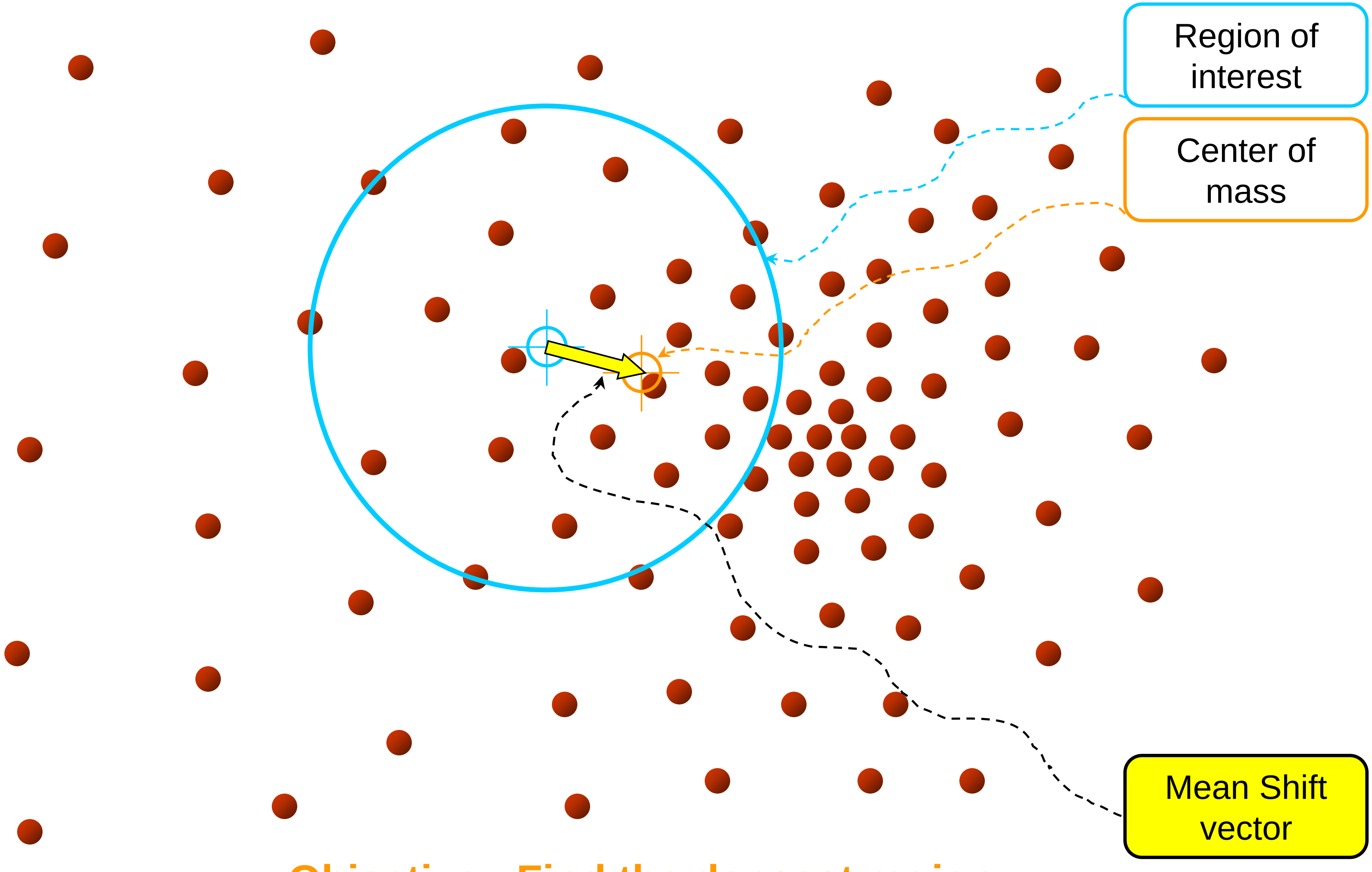


主要内容

- **Mean Shift 理论**
  - What is Mean Shift ?
  - Density Estimation Methods
  - Deriving the Mean Shift
  - Mean shift properties
- **Mean Shift 的应用**
  - Clustering
  - Segmentation

Mean Shift 理论

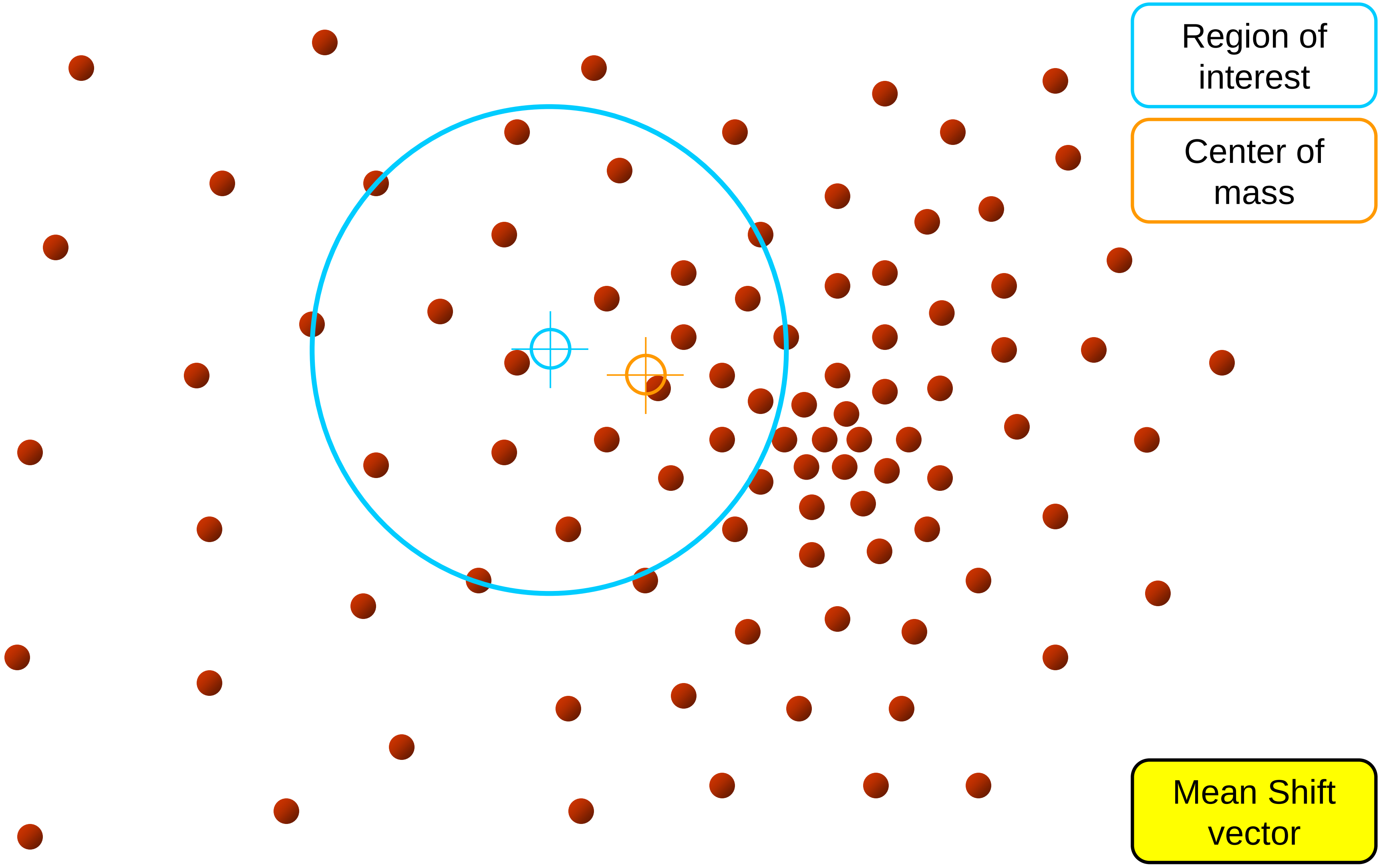
基本原理



Objective : Find the densest region

Distribution of identical billiard balls

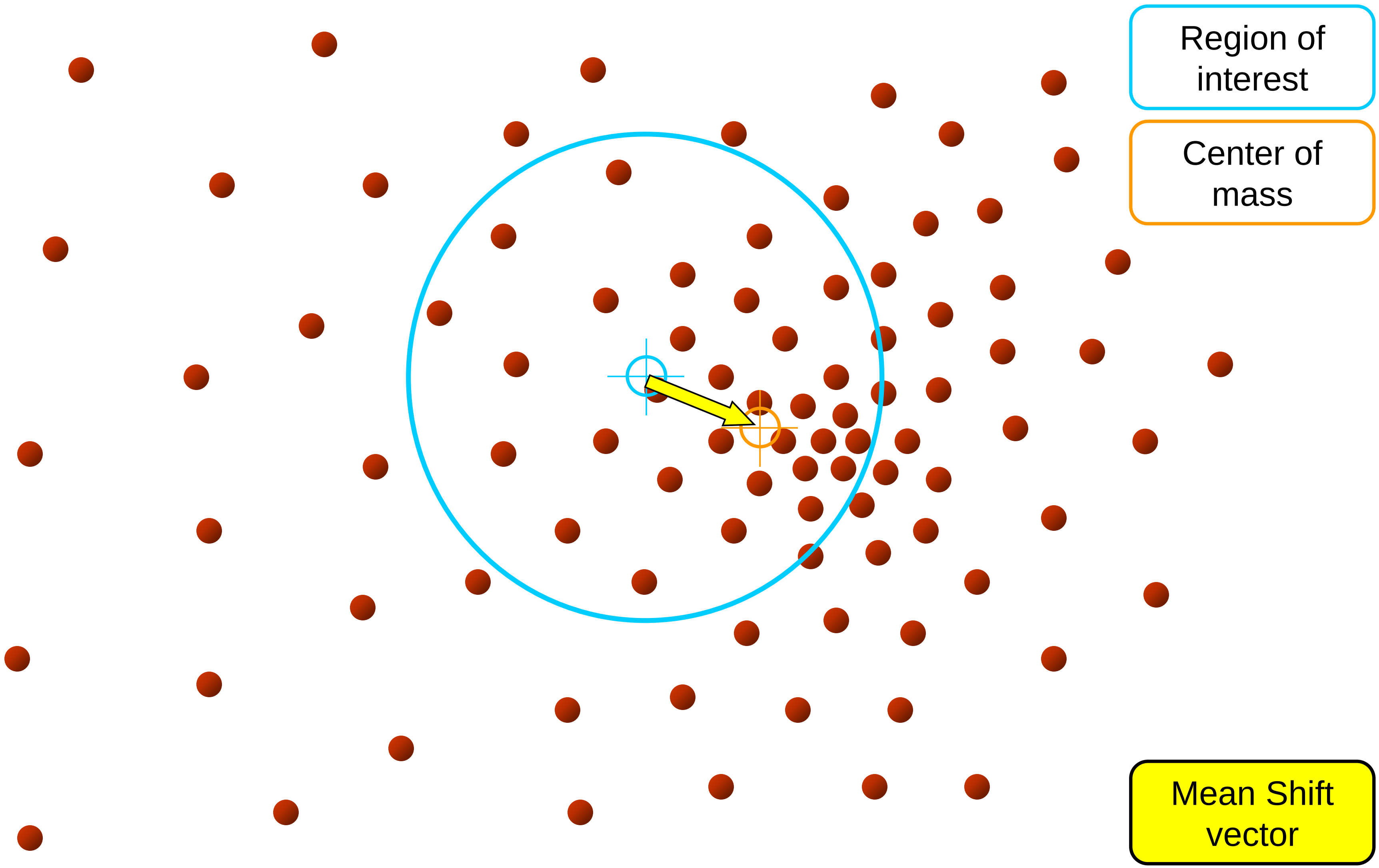
基本原理



Objective : Find the densest region

Distribution of identical billiard balls

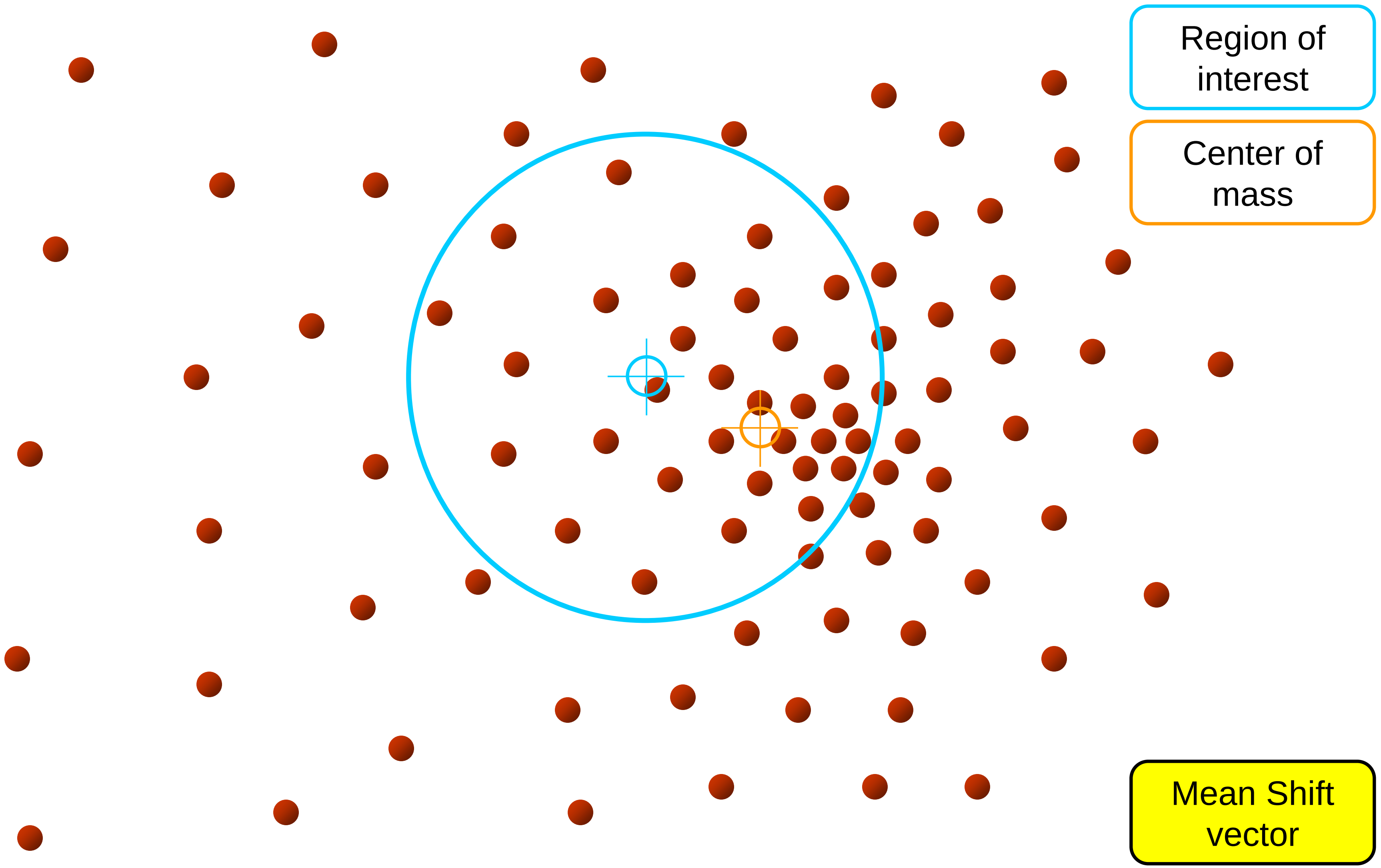
基本原理



<u>Objective</u> : Find the densest region

Distribution of identical billiard balls

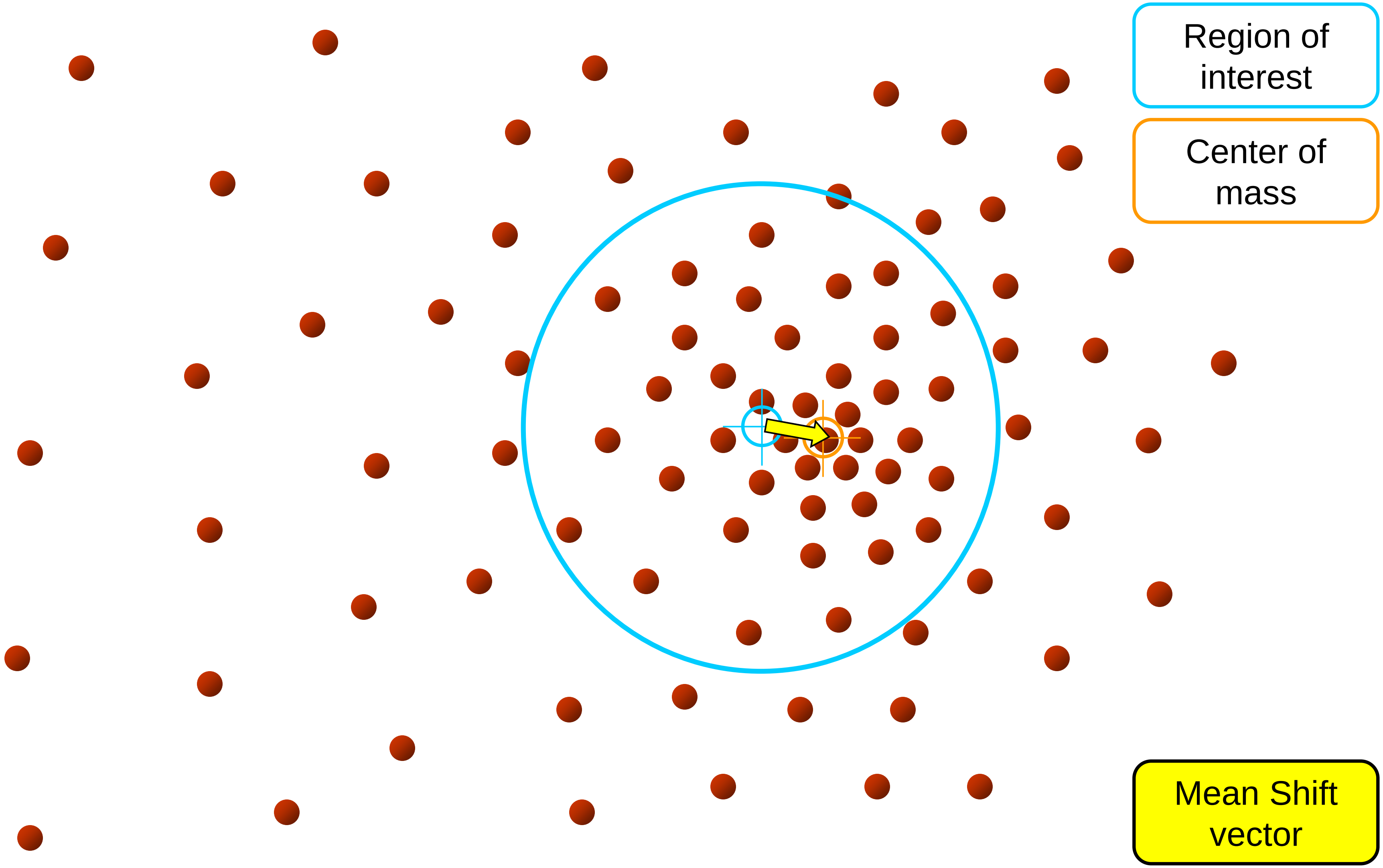
基本原理



<u>Objective</u> : Find the densest region

Distribution of identical billiard balls

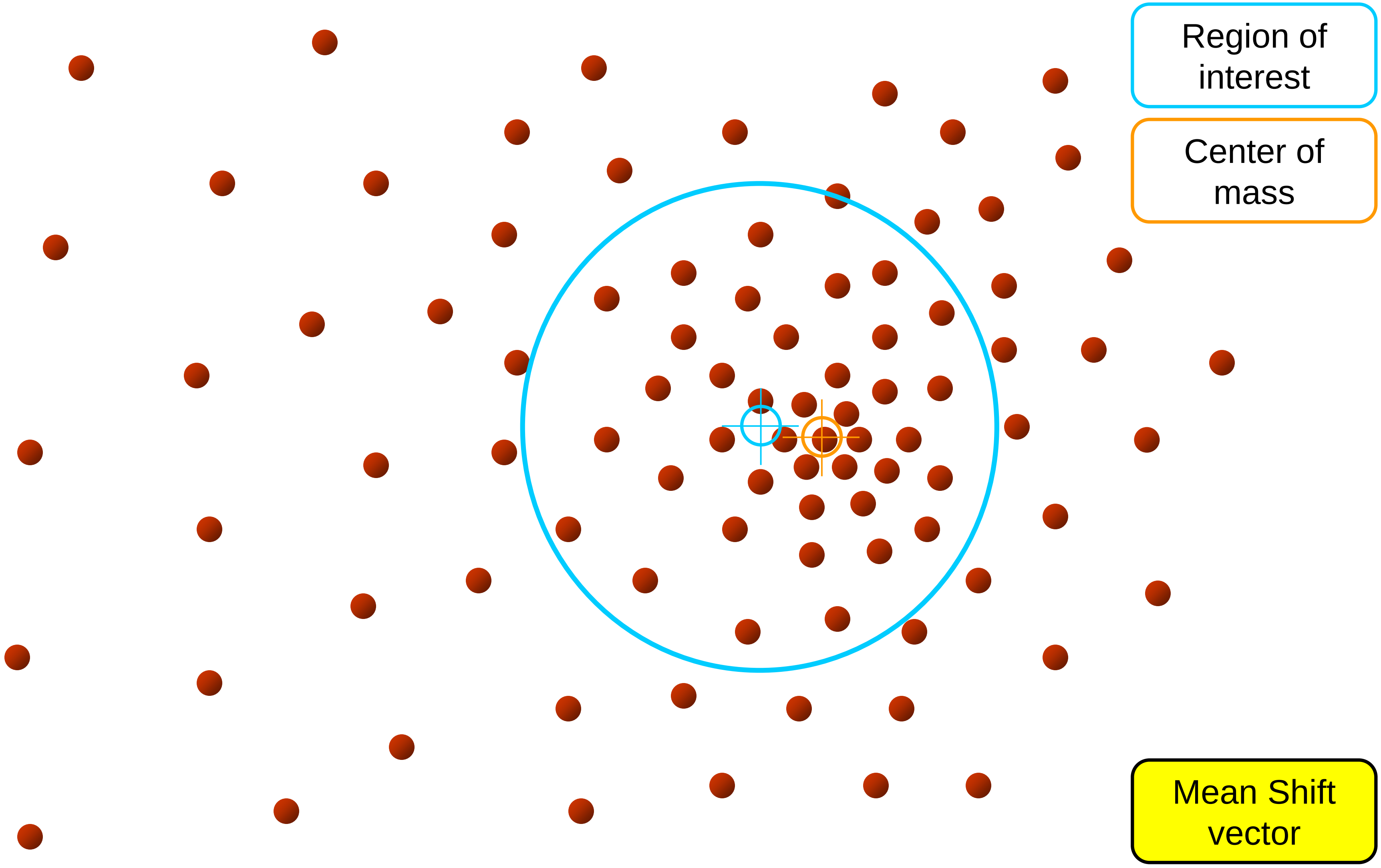
基本原理



Objective : Find the densest region

Distribution of identical billiard balls

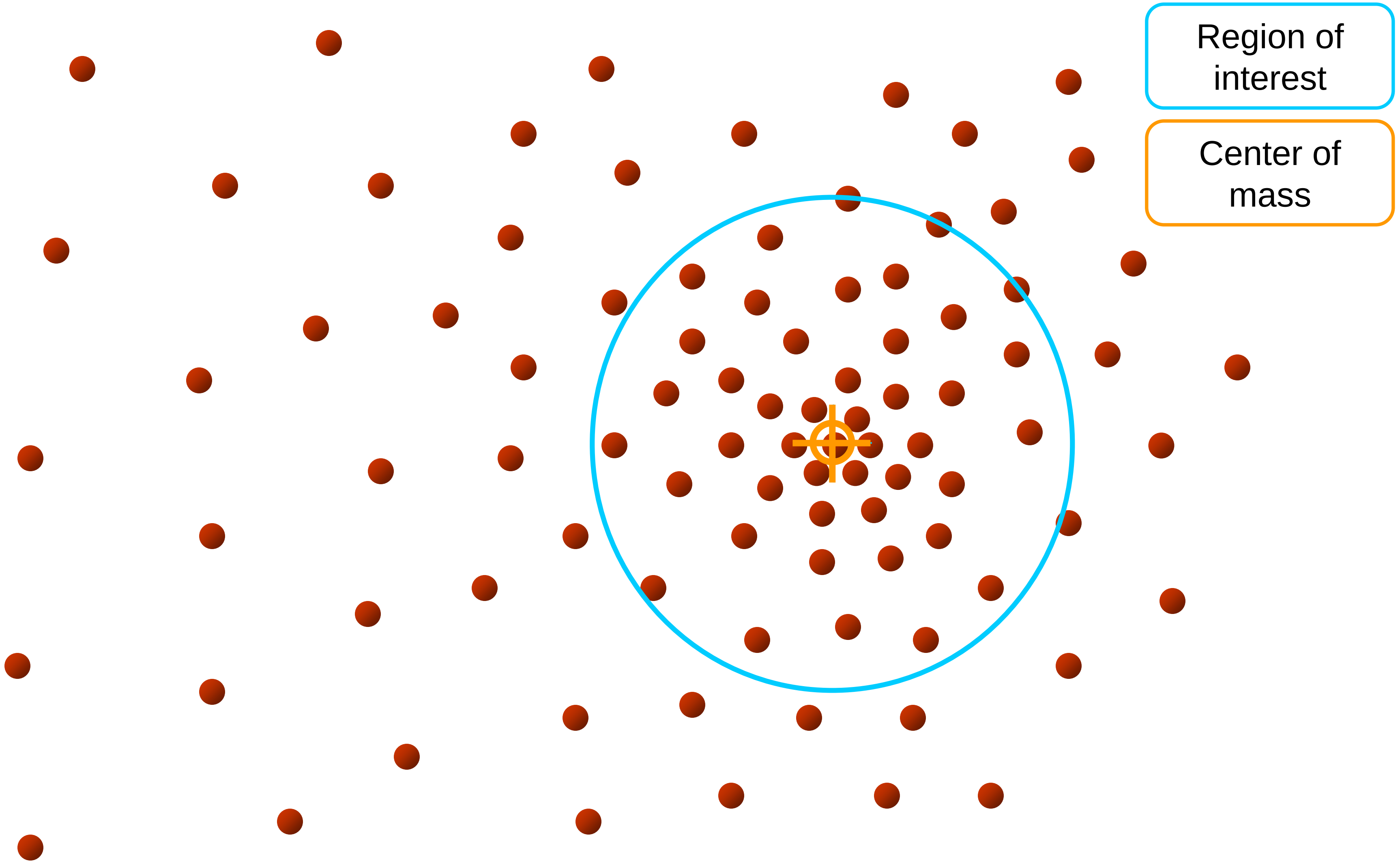
基本原理



<u>Objective</u> : Find the densest region

Distribution of identical billiard balls

基本原理



<u>Objective</u> : Find the densest region

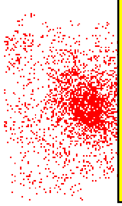
Distribution of identical billiard balls

什么是Mean Shift？

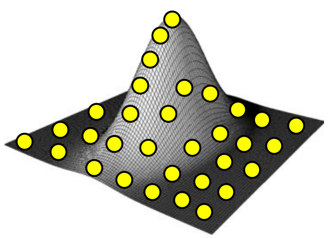
计算工具:
根据一组数据样本,找到能够反映数据在高维特征空间分布的概率密度函数(PDF)的模型集合

<u>特征空间的PDF</u>
- 颜色空间
- 尺度空间
- 任何可构建的特征空间
- …

etric
mation

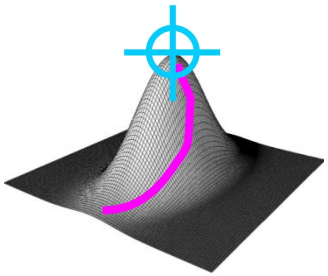


Data



Discrete PDF Representation

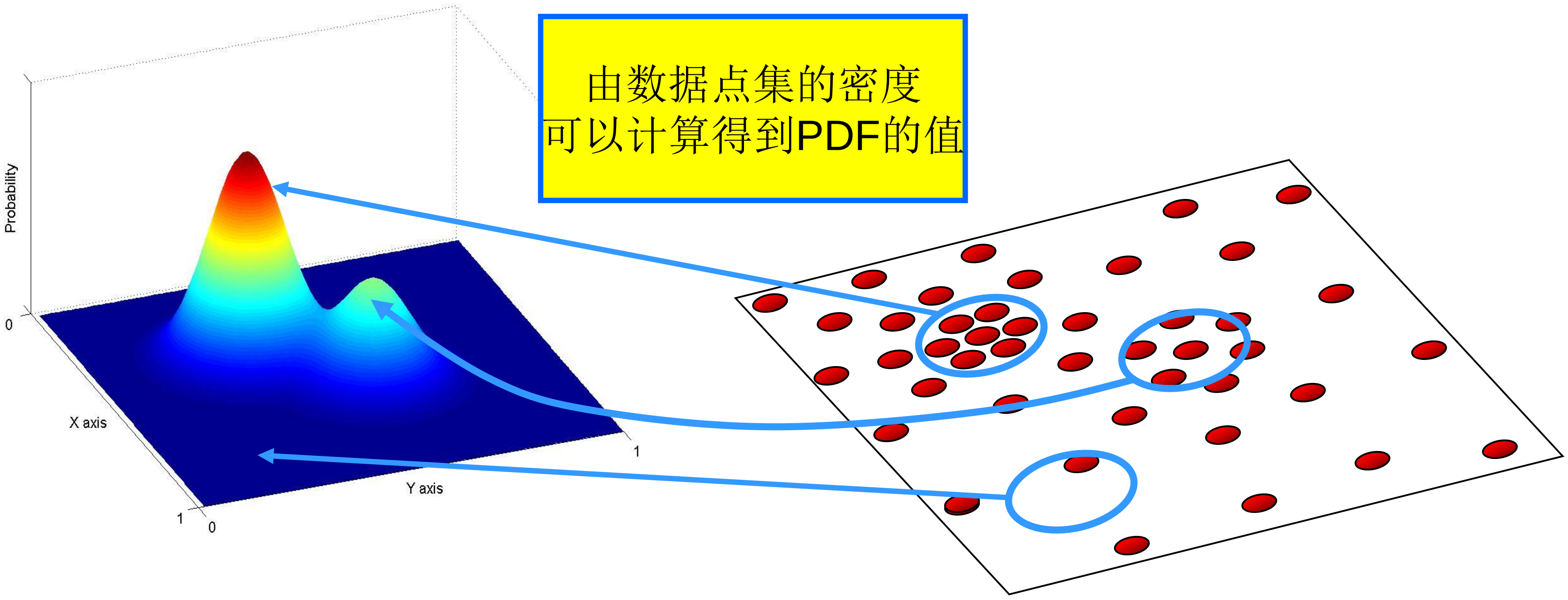
Non-parametric
Density **GRADIENT** Estimation
(Mean Shift)



PDF Analysis

非参数化的概率密度估计

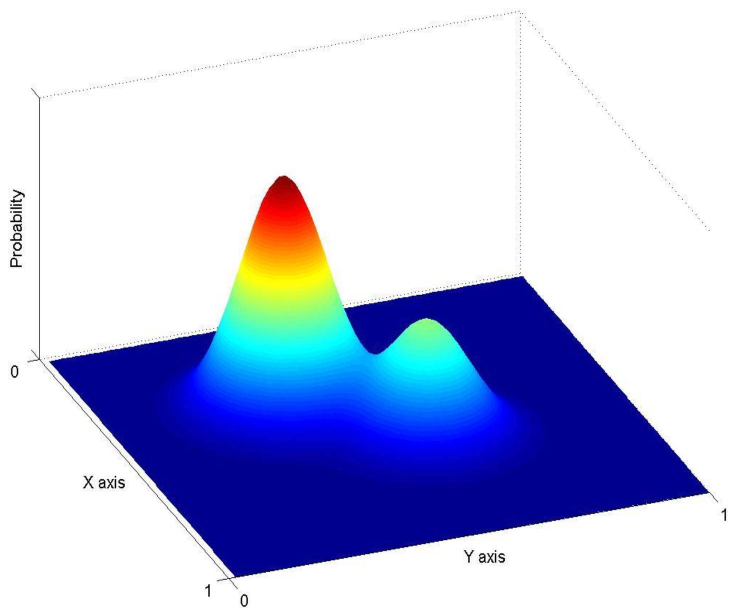
假设: 采样得到的数据点集服从一个隐含的概率密度函数(PDF)



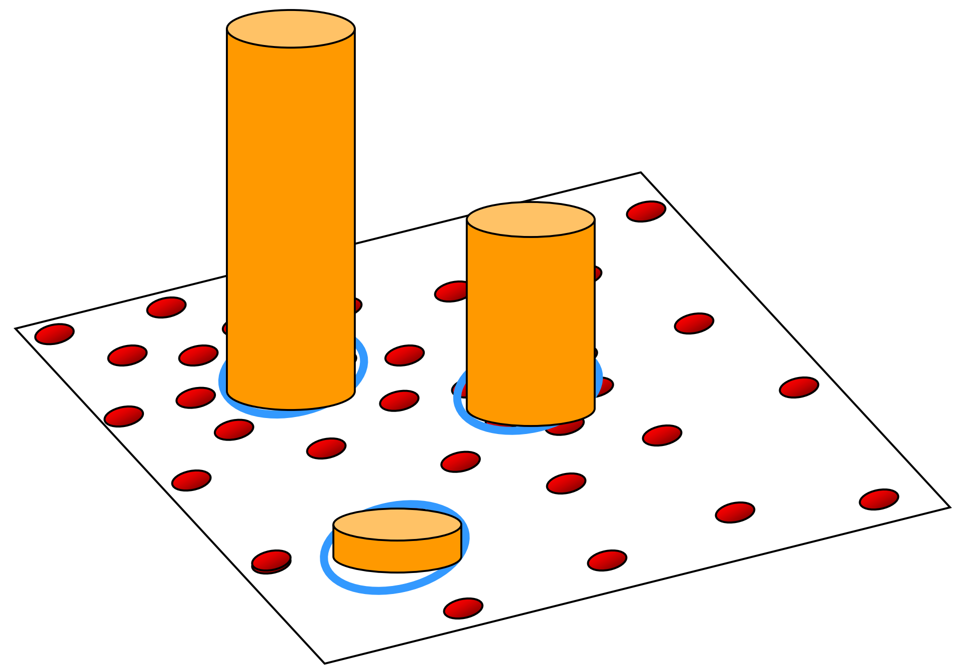
假设隐含的 PDF

真实数据点集

非参数化的概率密度估计

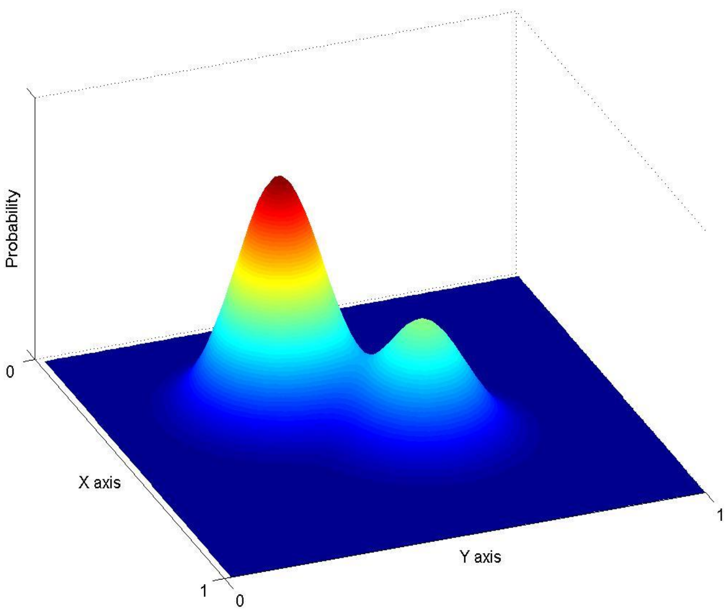


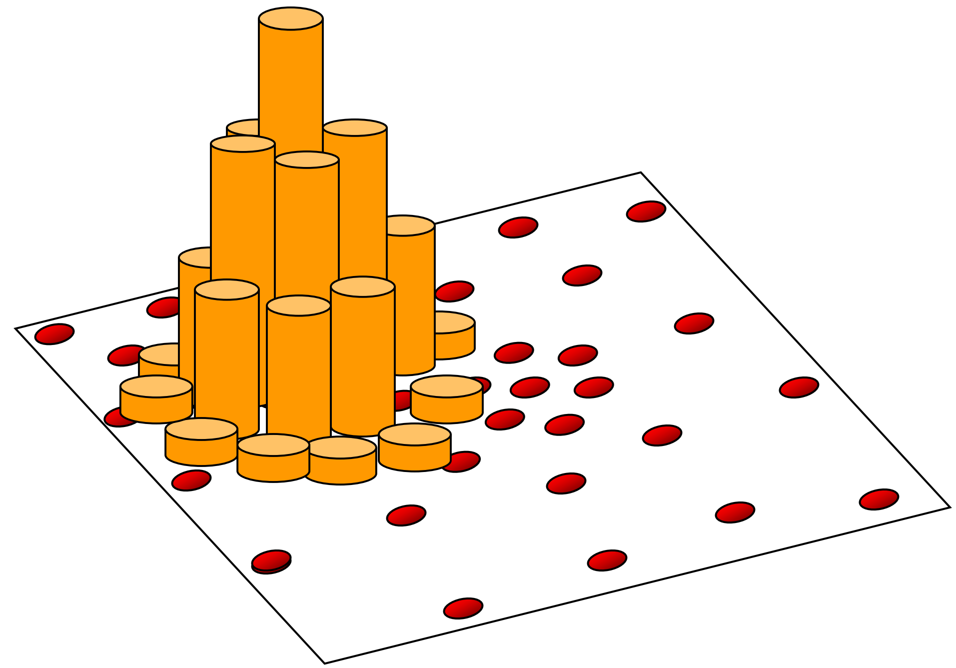
假设隐含的 **PDF**



真实数据点集

非参?数化的概率密度估计





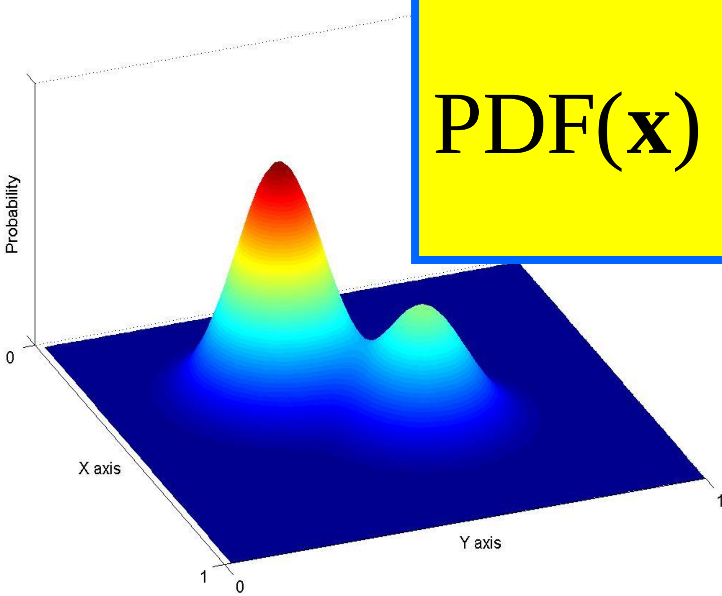
假设隐含的 PDF

真实数据点集

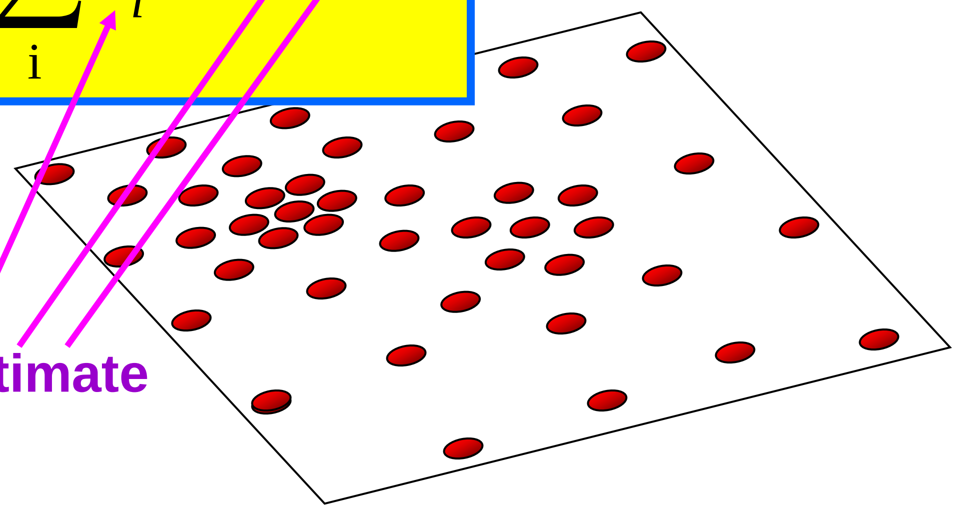
参数化的概率密度估计

假设: 采样得到的数据点集服从一个隐含的概率密度函数(PDF)

$$\mathrm{PDF}(\mathbf{x}) = \sum_{\mathrm{i}} c_i \cdot e^{-\frac{(\mathbf{x}\text{-}\mathbf{u}_i)^2}{2\sigma_i^{\,2}}}$$



Estimate



假设隐含的 PDF

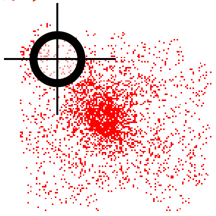
真实数据点集

核函数密度估计(Kernel Density Estimation)

Parzen Windows – 整体框架

$$P(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} K(\mathbf{x}-\mathbf{x}_i)$$

用于描述有限数量数据点集 $x_1 \ldots x_n$的函数



核函数性质:

•归一化的

$$\int_{R^d} K(\mathbf{x})d\mathbf{x} = 1$$

• 对称的

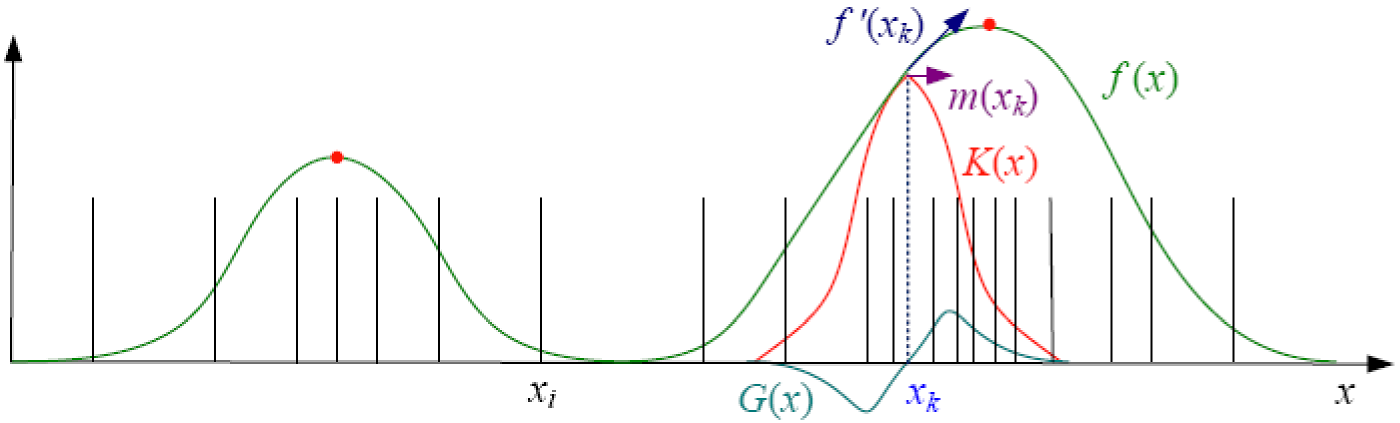
$$\int_{R^d} \mathbf{x}K(\mathbf{x})d\mathbf{x} = 0$$

•指数衰减

$$\lim_{\|\mathbf{x}\|\to\infty} \|\mathbf{x}\|^d K(\mathbf{x}) = 0$$

• ???

$$\int_{R^d} \mathbf{x}\mathbf{x}^T K(\mathbf{x})d\mathbf{x} = c\mathbf{I}$$



$f'(x_k)$
$f(x)$
$m(x_k)$
$K(x)$
x_i
$G(x)$
x_k
x

Example: Given a set of five data points x1 = 2, x2 = 2.5, x3 = 3, x4 = 1 and x5 = 6, find Parzen probability density function (pdf) estimates at x = 3, using the Gaussian function with σ = 1 as window function.

Solution:

$$\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{(x_1 - x)^2}{2}\right)$$

$$= \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{(2 - 3)^2}{2}\right) = 0.2420$$

$$\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{(x_2 - x)^2}{2}\right)$$

$$= \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{(2.5 - 3)^2}{2}\right) = 0.3521$$

$$\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{(x_3 - x)^2}{2}\right) = 0.3989$$

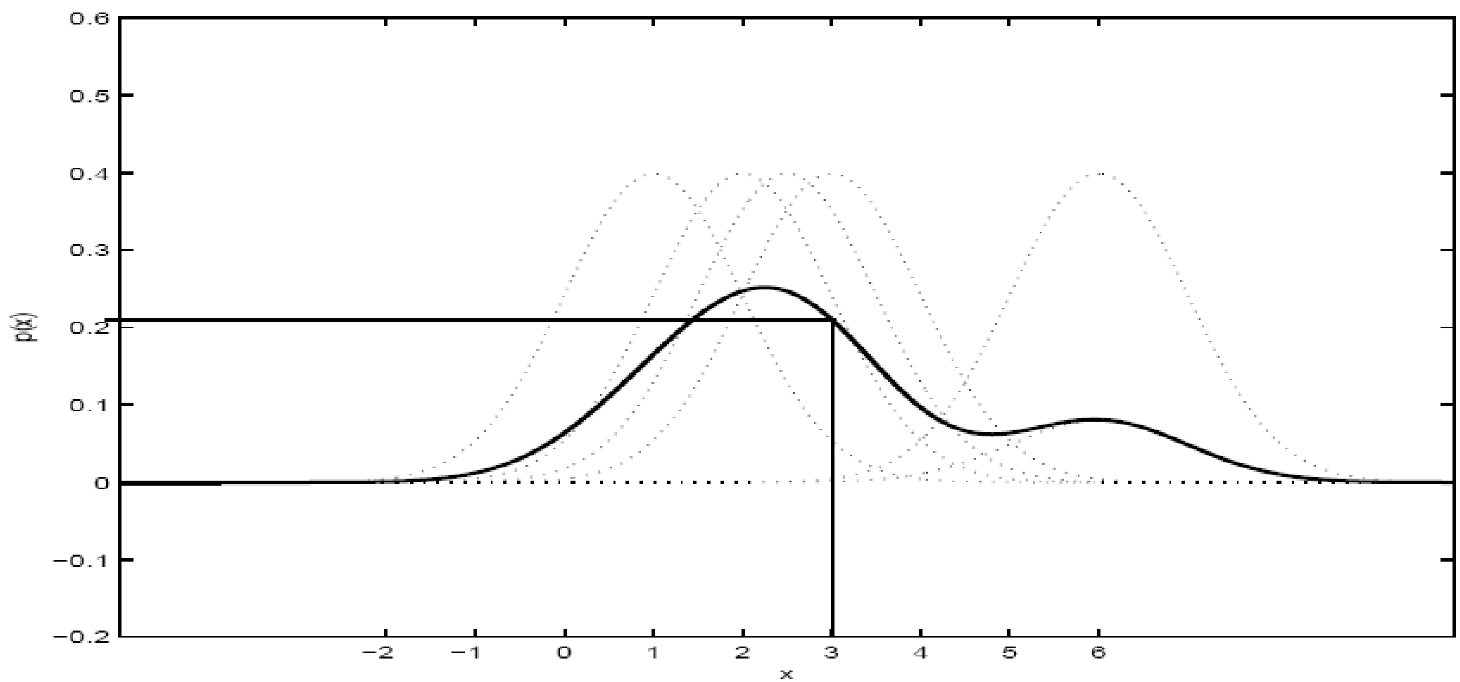
$$\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{(x_4 - x)^2}{2}\right) = 0.0540$$

$$\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{(x_5 - x)^2}{2}\right) = 0.0044$$

so

$$p(x = 3) = (0.2420 + 0.3521 + 0.3989$$

$$+0.0540 + 0.0044)/5 \quad = 0.2103$$



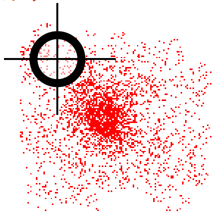
0.6
0.5
0.4
0.3
0.2
0.1
0
−0.1
−0.2
p(x)
−2
−1
0
1
2
3
4
5
6
x

核函数密度估计(Kernel Density Estimation)

Parzen Windows – 函数形式

$$P(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} K(\mathbf{x} - \mathbf{x}_i)$$

用于描述有限数量数据点集 $x_1 \ldots x_n$ 的函数



在实际应用时经常使用的函数形式为:

$$K(\mathbf{x}) = c\prod_{i=1}^{d} k(x_i)$$ or $$K(\mathbf{x}) = ck\left(\|\mathbf{x}\|\right)$$

在各个数据纬度上使用相同的核函数

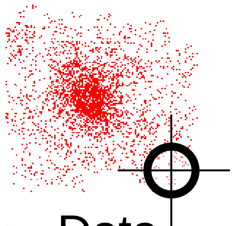
建立关于向量长度的核函数

核函数密度估计(Kernel Density Estimation)

各种类型的核函数

$$P(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} K(\mathbf{x} - \mathbf{x}_i)$$

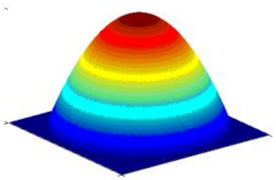
用于描述有限数量数据点集 $x_1 \ldots x_n$ 的函数



<u>核函数举例:</u>

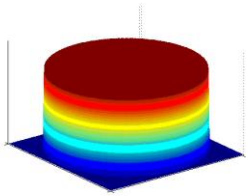
- **Epanechnikov Kernel**

$$K_E(\mathbf{x}) = \begin{cases} c\left(1 - \|\mathbf{x}\|^2\right) & \|\mathbf{x}\| \le 1 \\ 0 & \text{otherwise} \end{cases}$$



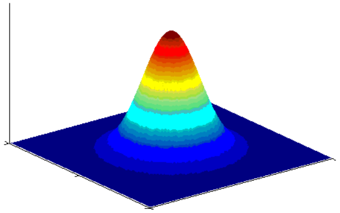
- **Uniform Kernel**

$$K_U(\mathbf{x}) = \begin{cases} c & \|\mathbf{x}\| \le 1 \\ 0 & \text{otherwise} \end{cases}$$



- **Normal Kernel**

$$K_N(\mathbf{x}) = c \cdot \exp\left(-\frac{1}{2}\|\mathbf{x}\|^2\right)$$



核函数密度估计(Kernel Density Estimation)

梯度 Gradient

$$\nabla P(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} \nabla K(\mathbf{x}-\mathbf{x}_i)$$

不估计 **PDF !**
只估计概率密度的梯度

假设使用的核函数形式为:

$$K(\mathbf{x}-\mathbf{x}_i) = ck\left(\left\|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right\|^2\right)$$

Size of window

可以得到:

$$\nabla P(\mathbf{x}) = \frac{c}{n}\sum_{i=1}^{n} \nabla k_i = \frac{c}{n}\left[\sum_{i=1}^{n} g_i\right] \bullet \left[\frac{\sum_{i=1}^{n} \mathbf{x}_i g_i}{\sum_{i=1}^{n} g_i} - \mathbf{x}\right]$$

$$g(\mathbf{x}) = -k'(\mathbf{x})$$

Kernel Density Estimation

Computing The Mean Shift

Gradient

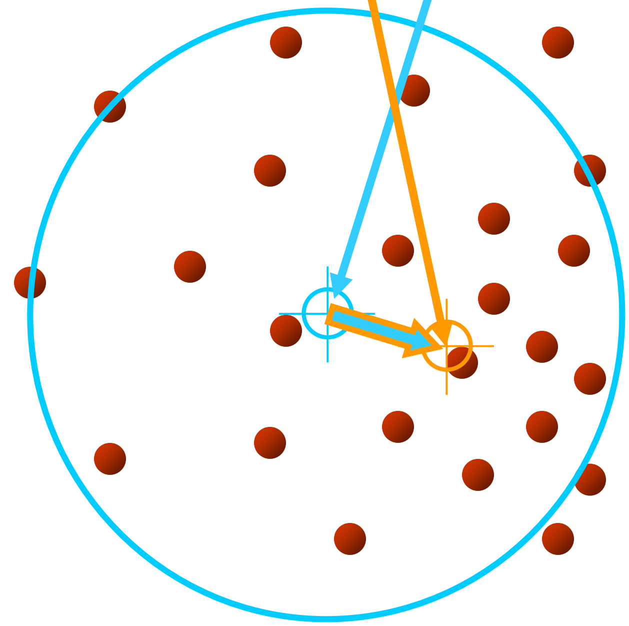
$$\nabla P(\mathbf{x}) = \frac{c}{n}\sum_{i=1}^{n}\nabla k_i = \frac{c}{n}\left[\sum_{i=1}^{n} g_i\right] \bullet \left[\frac{\sum_{i=1}^{n}\mathbf{x}_i g_i}{\sum_{i=1}^{n} g_i} - \mathbf{x}\right]$$

$$g(\mathbf{x}) = -k'(\mathbf{x})$$

Computing The Mean Shift

$$\nabla P(\mathbf{x}) = \frac{c}{n}\sum_{i=1}^{n}\nabla k_i = \frac{c}{n}\left[\sum_{i=1}^{n} g_i\right]\bullet\left[\frac{\sum_{i=1}^{n}\mathbf{x}_i g_i}{\sum_{i=1}^{n} g_i} - \mathbf{x}\right]$$

对核函数密度的梯度进行估计！



Mean Shift 简易流程:

- 计算mean shift 向量

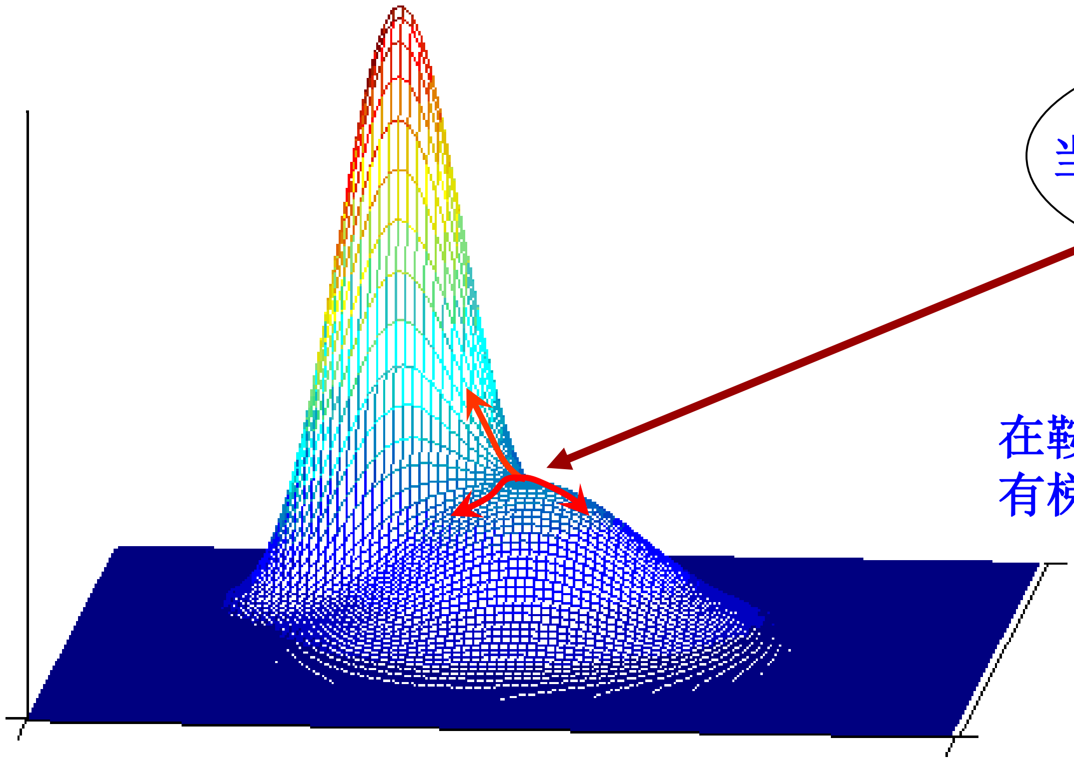
$$\mathbf{m}(\mathbf{x}) = \left[\frac{\sum_{i=1}^{n}\mathbf{x}_i g\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{h}\right)}{\sum_{i=1}^{n} g\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{h}\right)} - \mathbf{x}\right]$$

- 根据m(x)移动核函数窗口

$$g(\mathbf{x}) = -k'(\mathbf{x})$$

Mean Shift 方向检测

当遇到鞍点该如何处理?

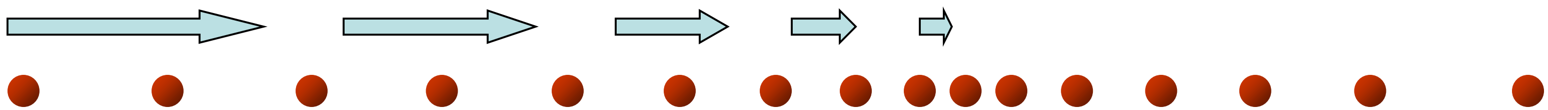


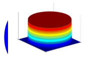
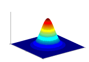
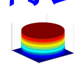
在鞍点周围四处移动,检测是否有梯度增加的方向.

<u>**Mean Shift 更新程序:**</u>

- **找到所有的mean shift的方向**
- **四处移动，找到鞍点和高地**
- **取梯度改变最大的方向**

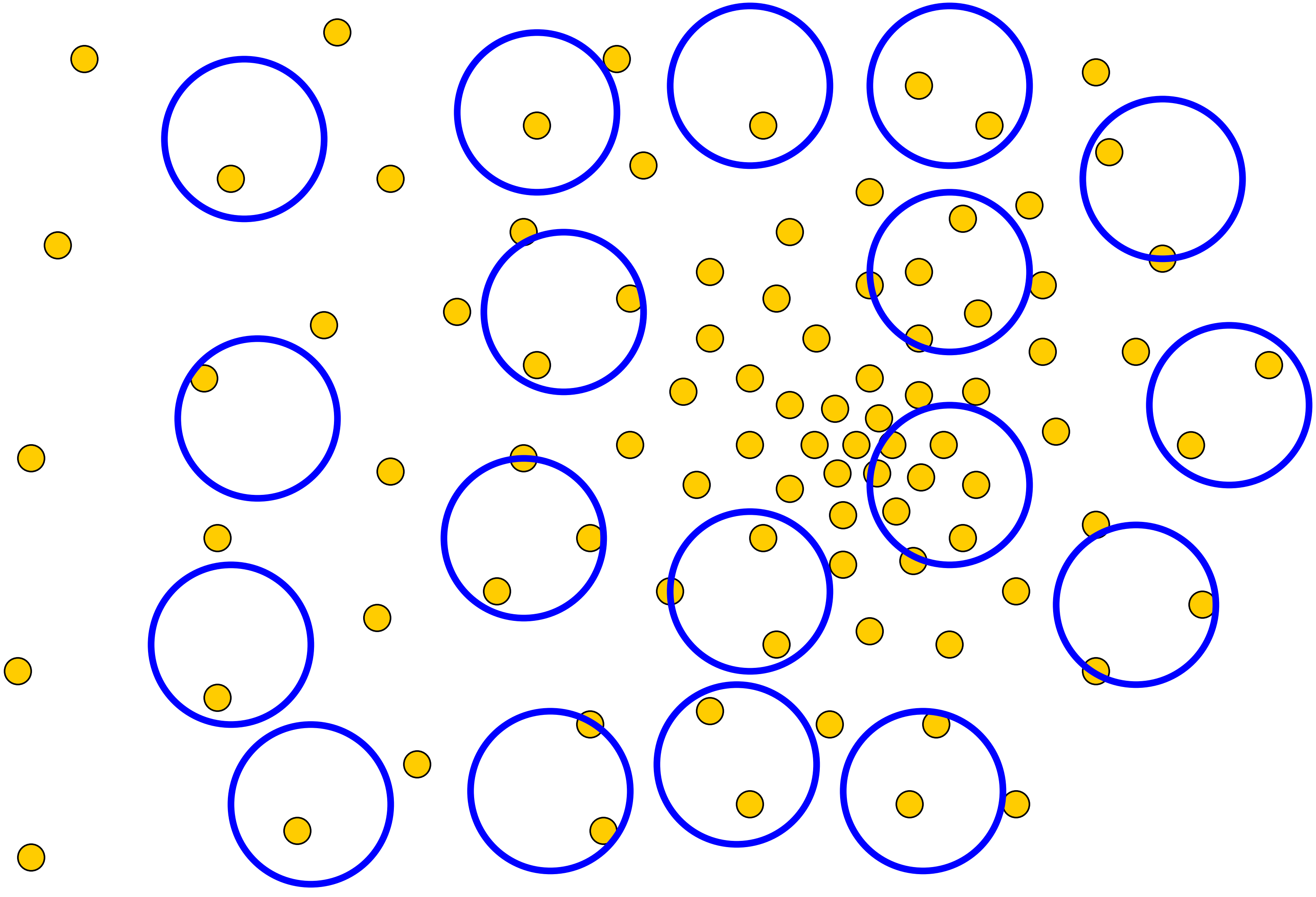
Mean Shift 特性



- 自动收敛的速度–**mean shift** 向量的大小取决于梯度本身
- 在最大值附近，步长很小而且非定值
- 理论上需要无限步数,因此需要设定一个下界。
- 对于**Uniform Kernel (****),** 可以在有限步数内收敛
- **Normal Kernel (****)** 可以得到平滑轨迹**,** 但是收敛速度慢于**Uniform Kernel (****).**

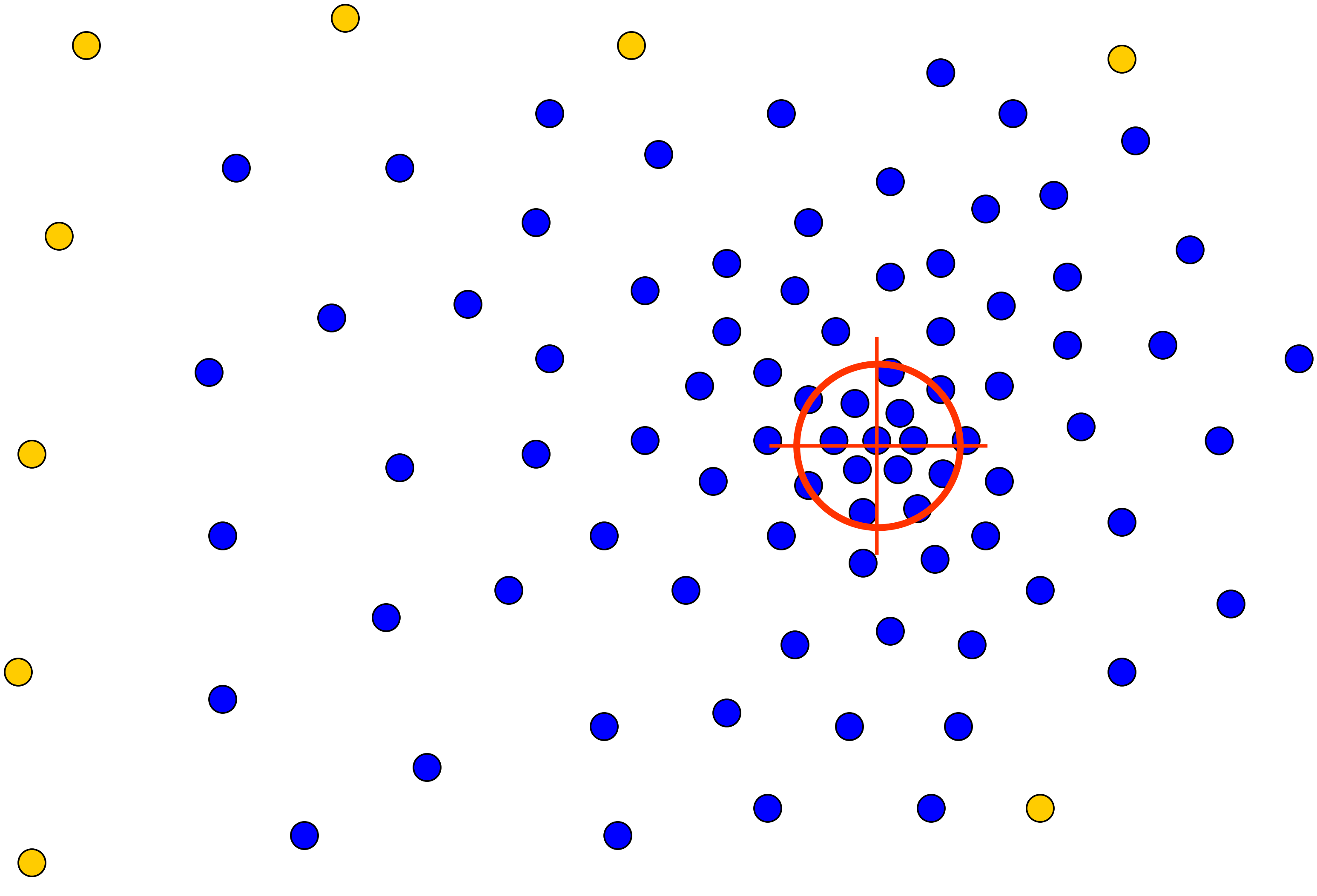
Adaptive Gradient Ascent

真实模态分析



将整个数据空间窗口化　　　　并行运行程序

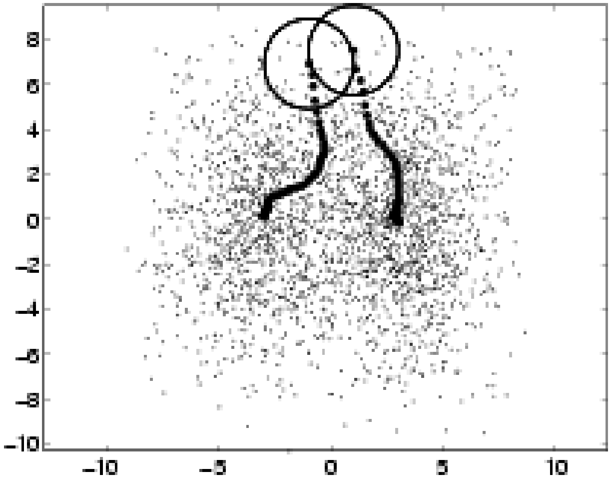
真实模态分析



蓝色的数据点是被窗口移动时经过的

真实模态分析

An example



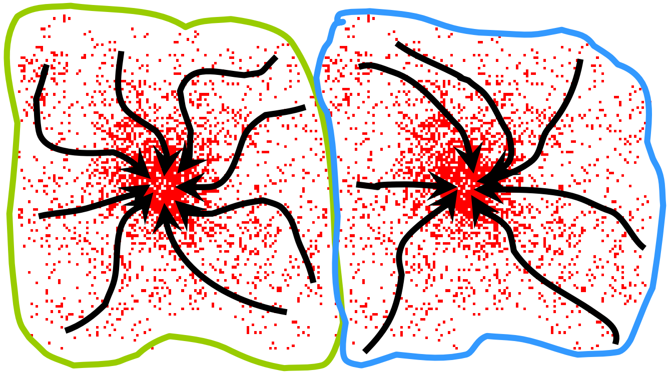
窗口是向着梯度变化最剧烈的方向移动的

Mean Shift 应用

聚类

Cluster : All data points in the ***attraction basin*** of a mode

Attraction basin : the region for which all trajectories lead to the same mode



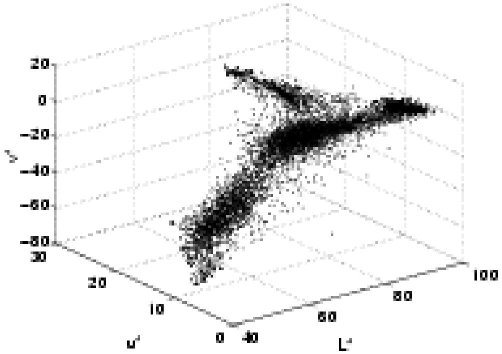
Mean Shift : A robust Approach Toward Feature Space Analysis, by Comaniciu, Meer

聚类

实例

特征空间:
L*u*v r颜色空间

输入初始窗口



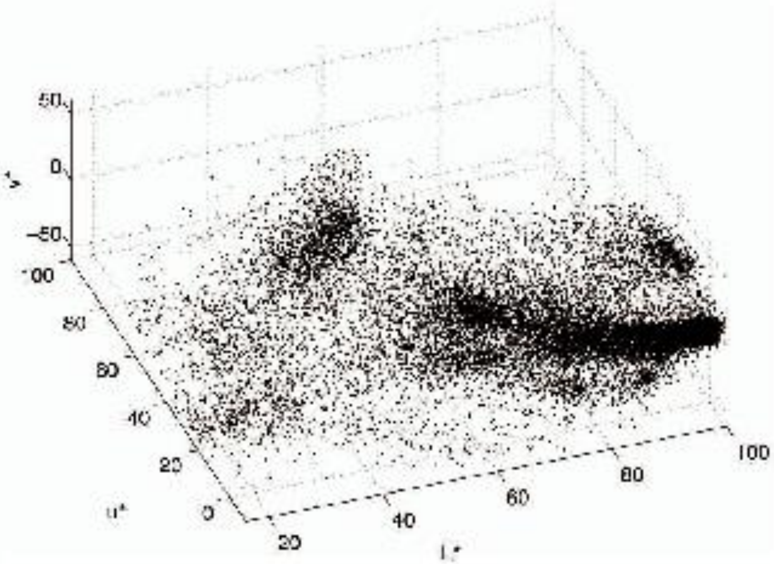
(a)

M

pruning

聚类

实例



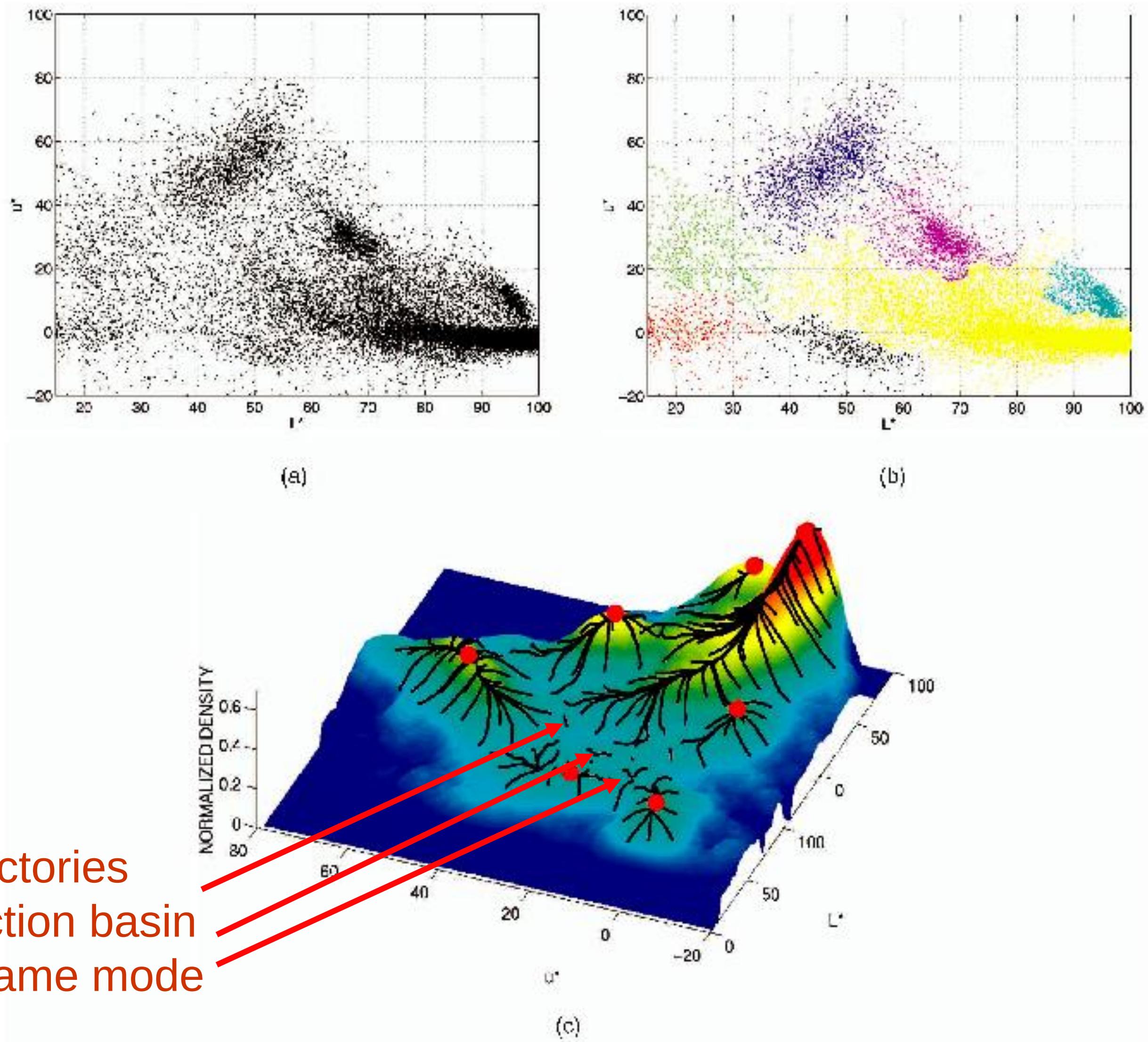
L*u*v space representation

聚类

实例

2D (L*u)
space
representation

Final clusters



Not all trajectories
in the attraction basin
reach the same mode

Segmentation

Example

…when feature space is only gray levels…

Segmentation

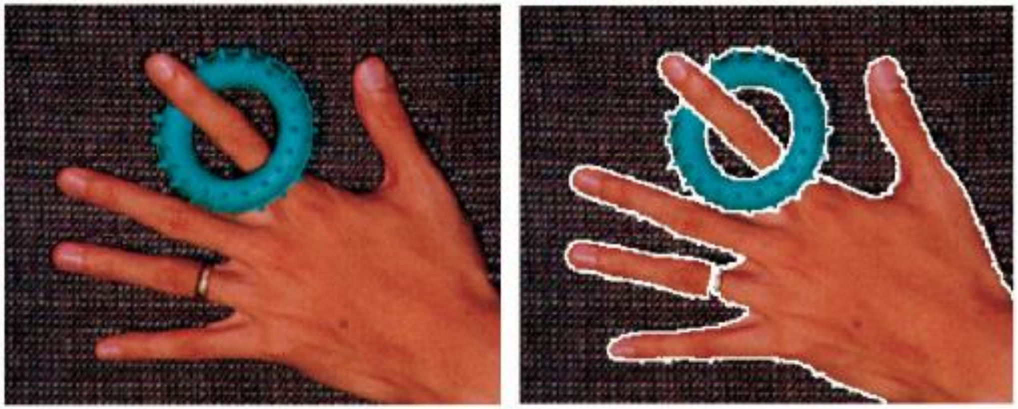
Example

Segmentation

Example

Segmentation

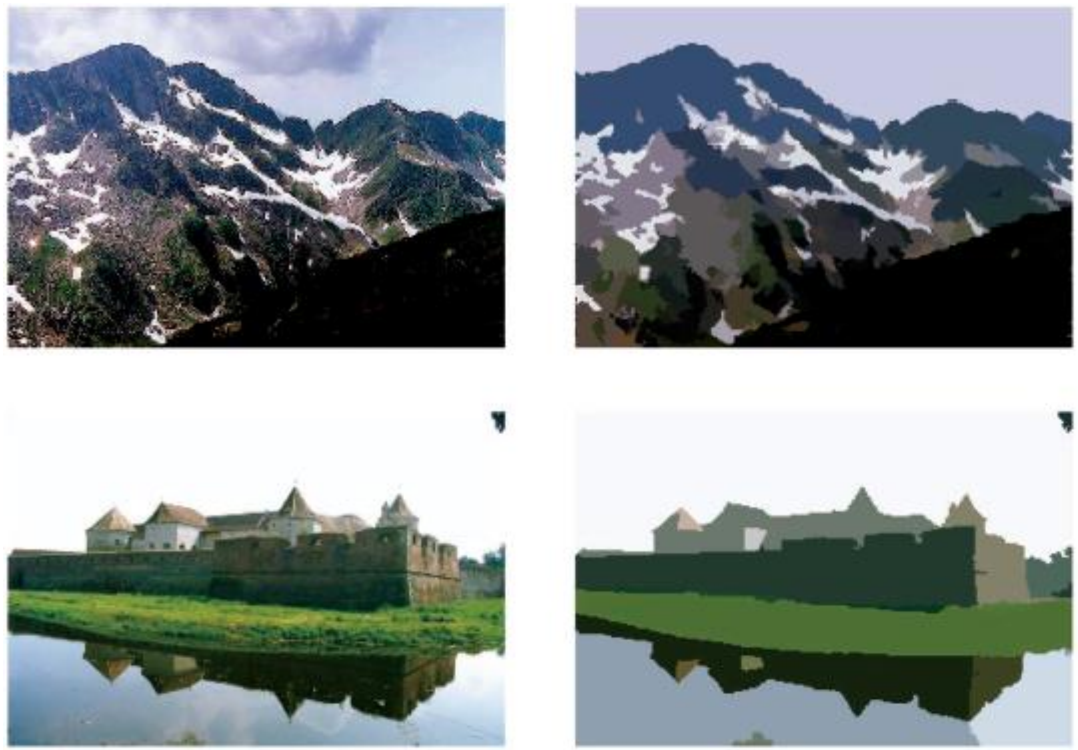
Example



Segmentation
Example

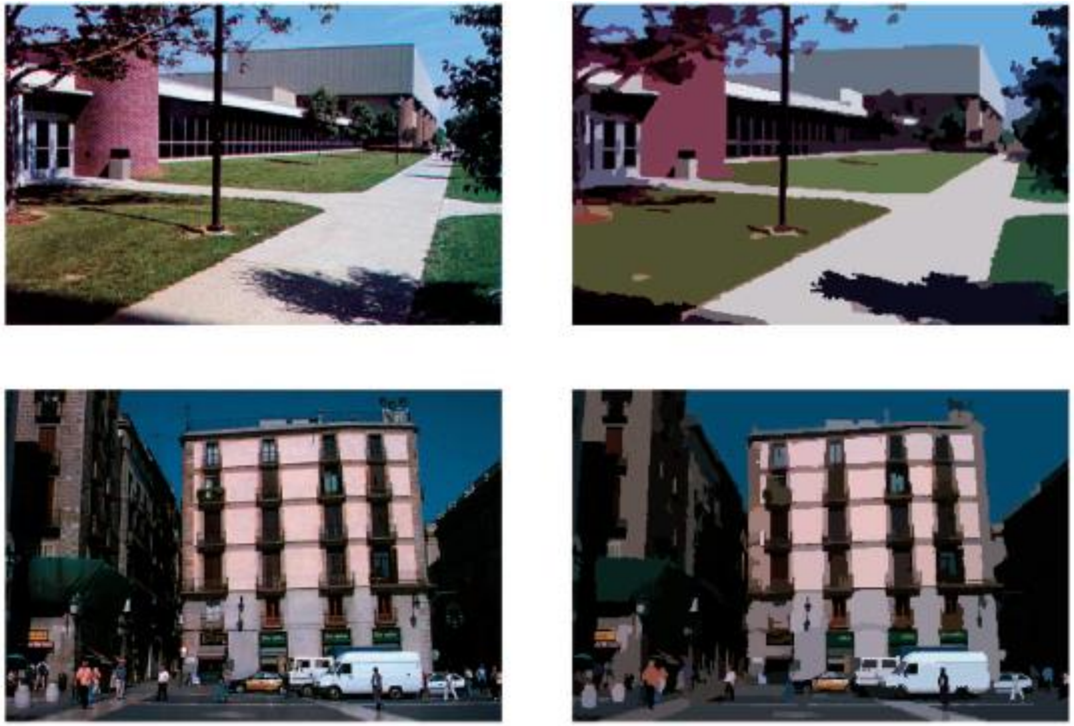
Segmentation

Example



Segmentation

Example



Normalized Cuts and Image Segmentation

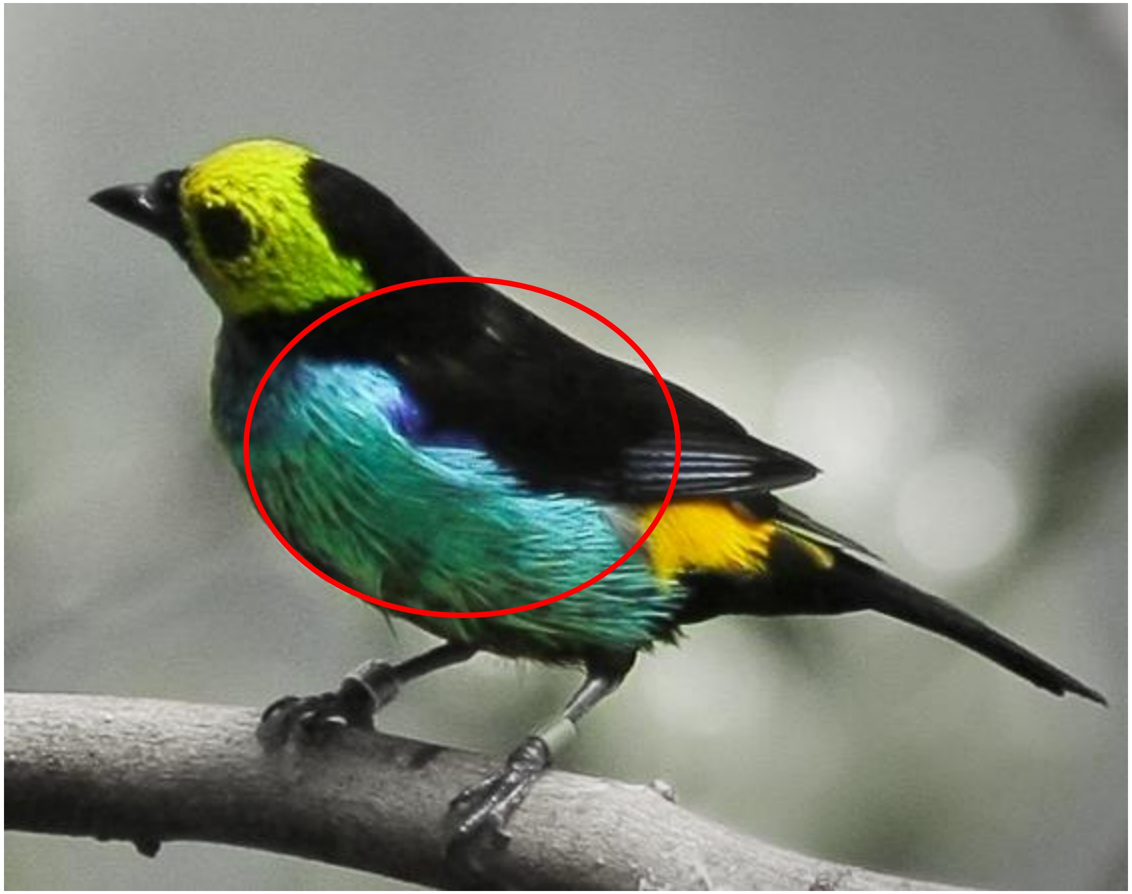
Jianbo Shi and Jitendra Malik, *Member*, *IEEE*

PAMI, 2000

基于图的图像分割

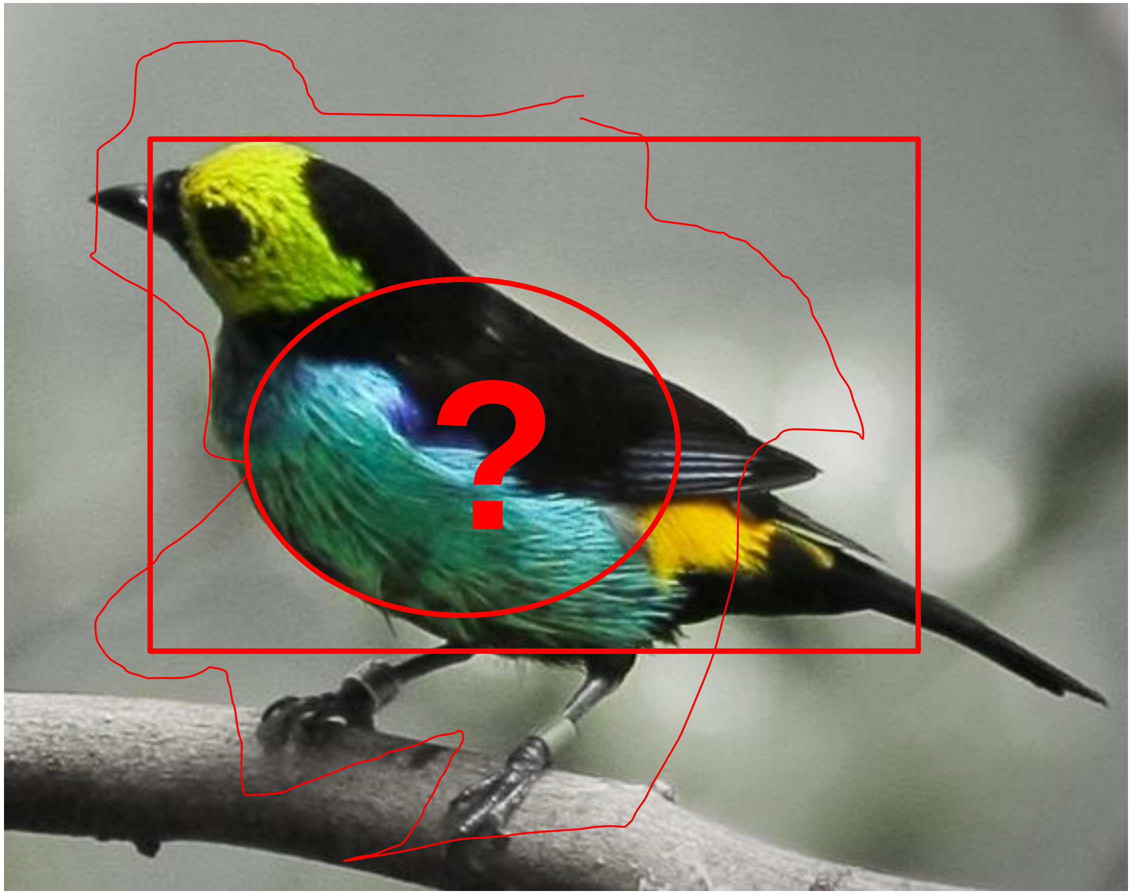
基于图的图像分割

- 首先: 选择感兴趣的区域



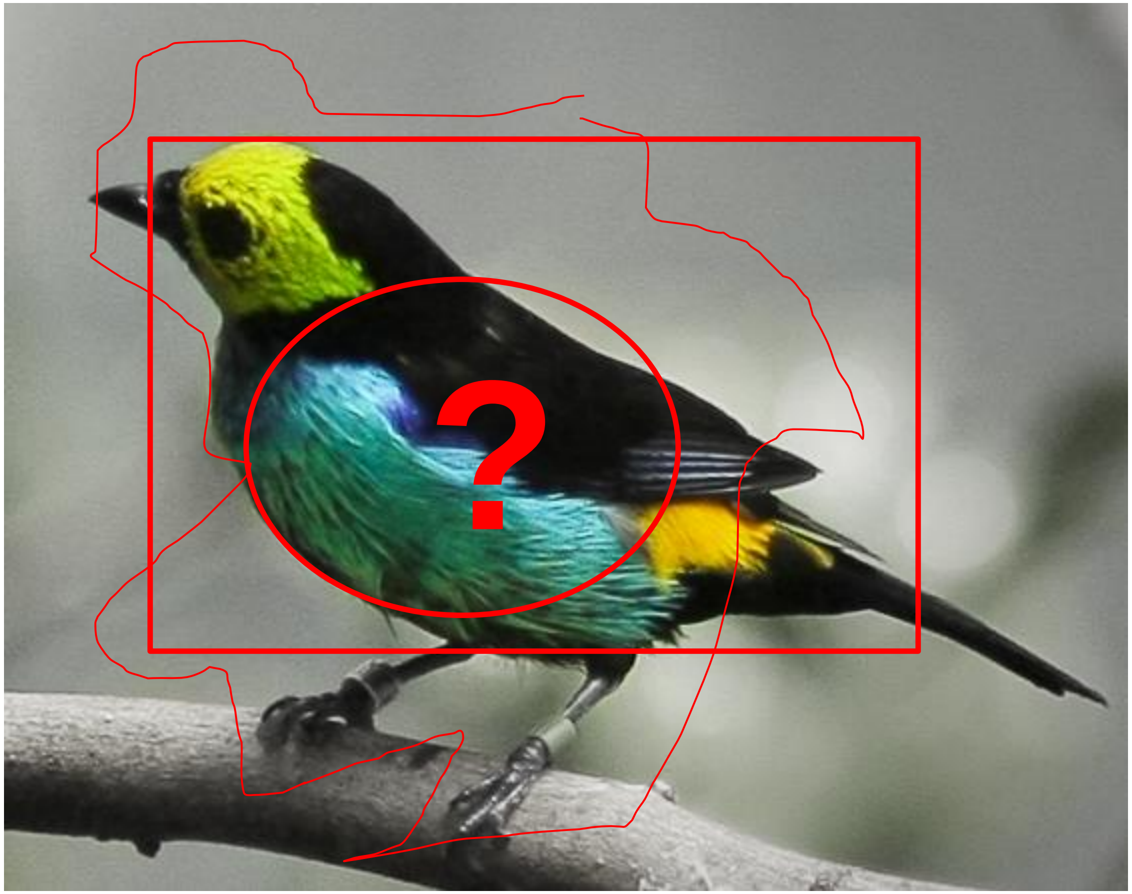
基于图的图像分割

- 怎样自动选取目标?



基于图的图像分割

- 两个要素:图(graph)和割(cut)



基于图的图像分割

- 什么是图(graph)?

节点-Nodes
- **通常是像素组合**
- **有时候是样本**

边-Edges
- **连接权重(W(i,j))**
- **例如颜色差异**

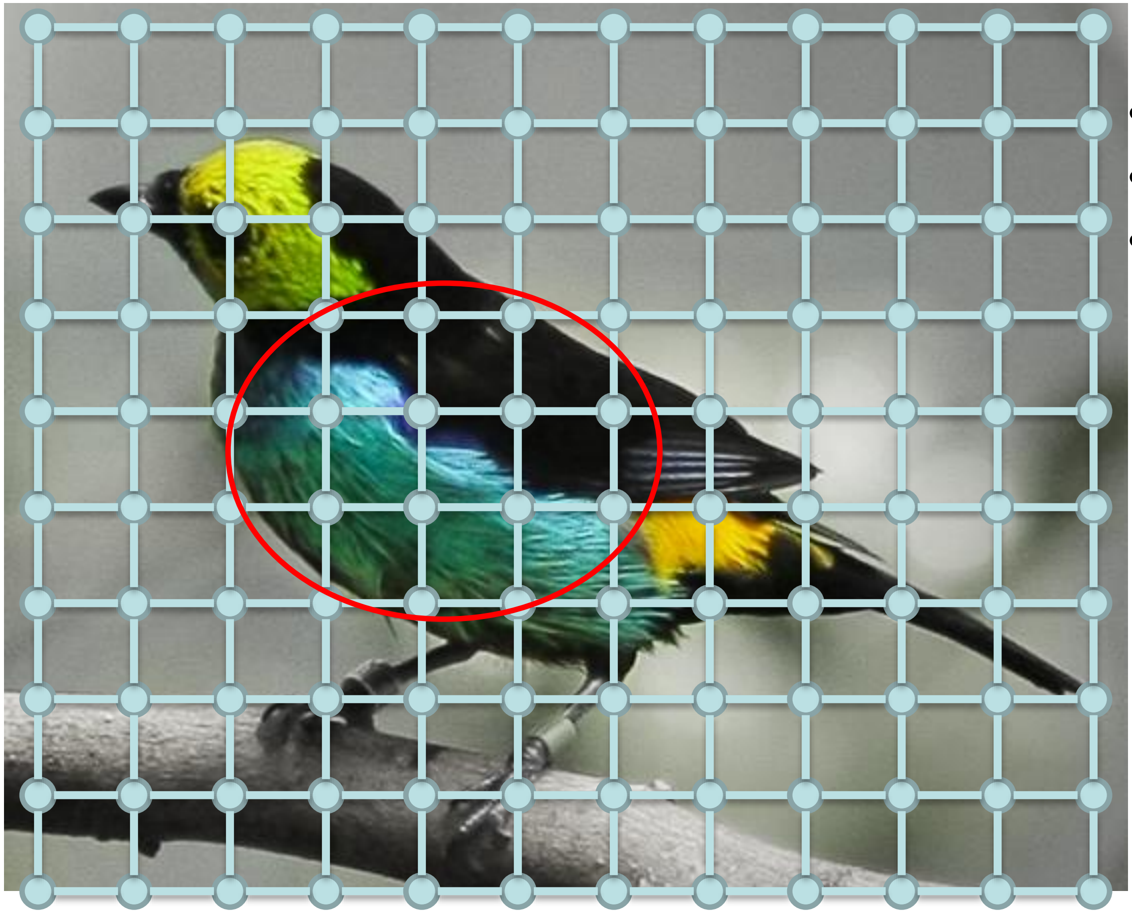
基于图的图像分割

- 什么是割(cuts)?

- **每个 “cut” -> 一些点, W(I,j)**
- **最优化问题**
- **W(i,j) = |LUV(i) – LUV(j)|**

基于图的图像分割

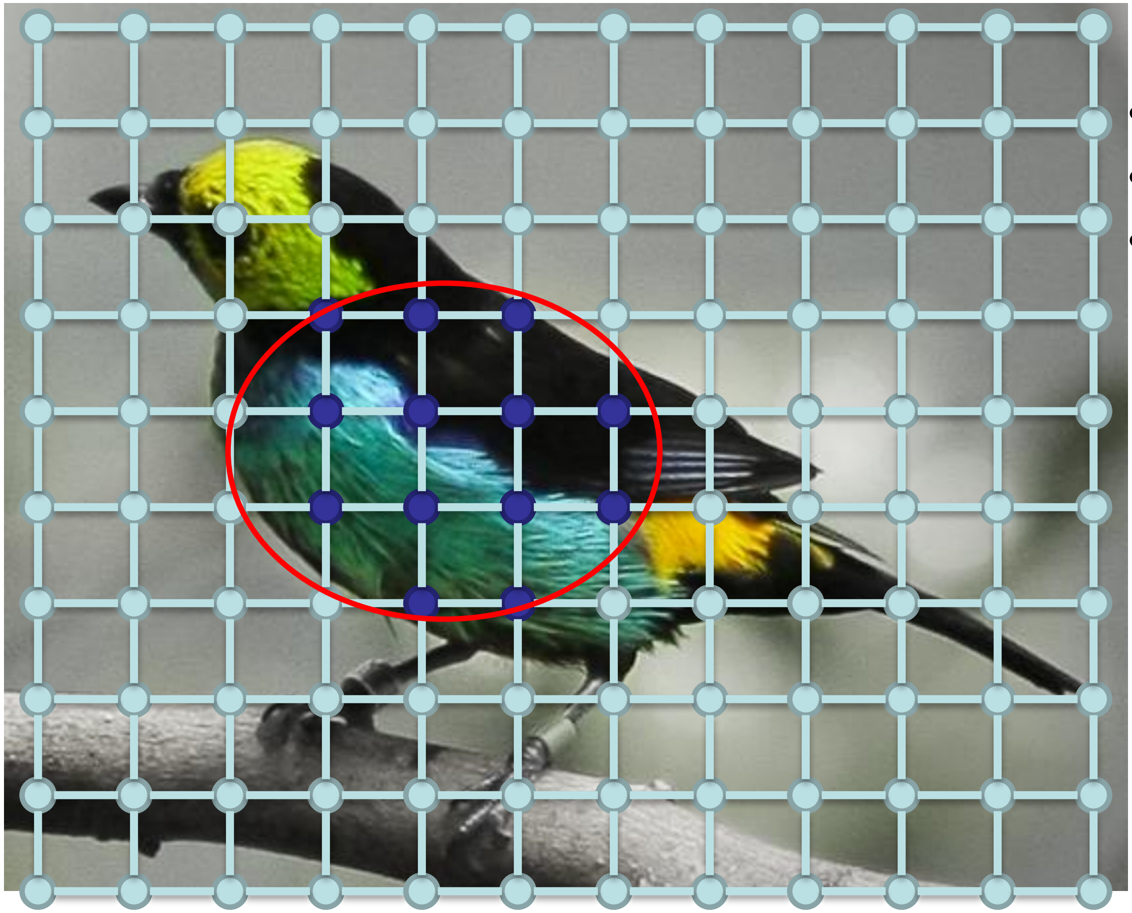
- **回到已选择的区域**



- **每个 “cut” -> 一些点, W(I,j)**
- **最优化问题**
- **W(i,j) = |LUV(i) – LUV(j)|**

基于图的图像分割

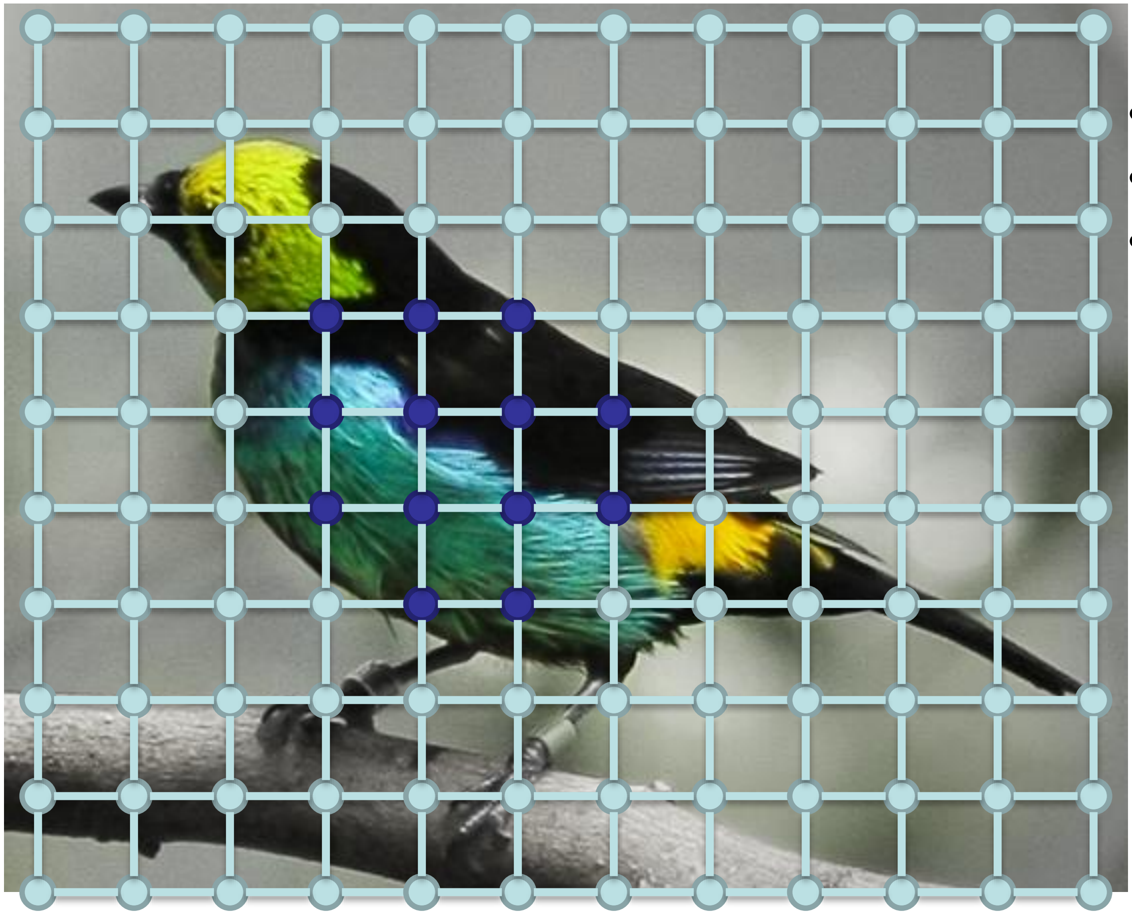
- 回到已选择的区域



- 每个 “cut” -> 一些点, W(I,j)
- 最优化问题
- W(i,j) = |LUV(i) – LUV(j)|

基于图的图像分割

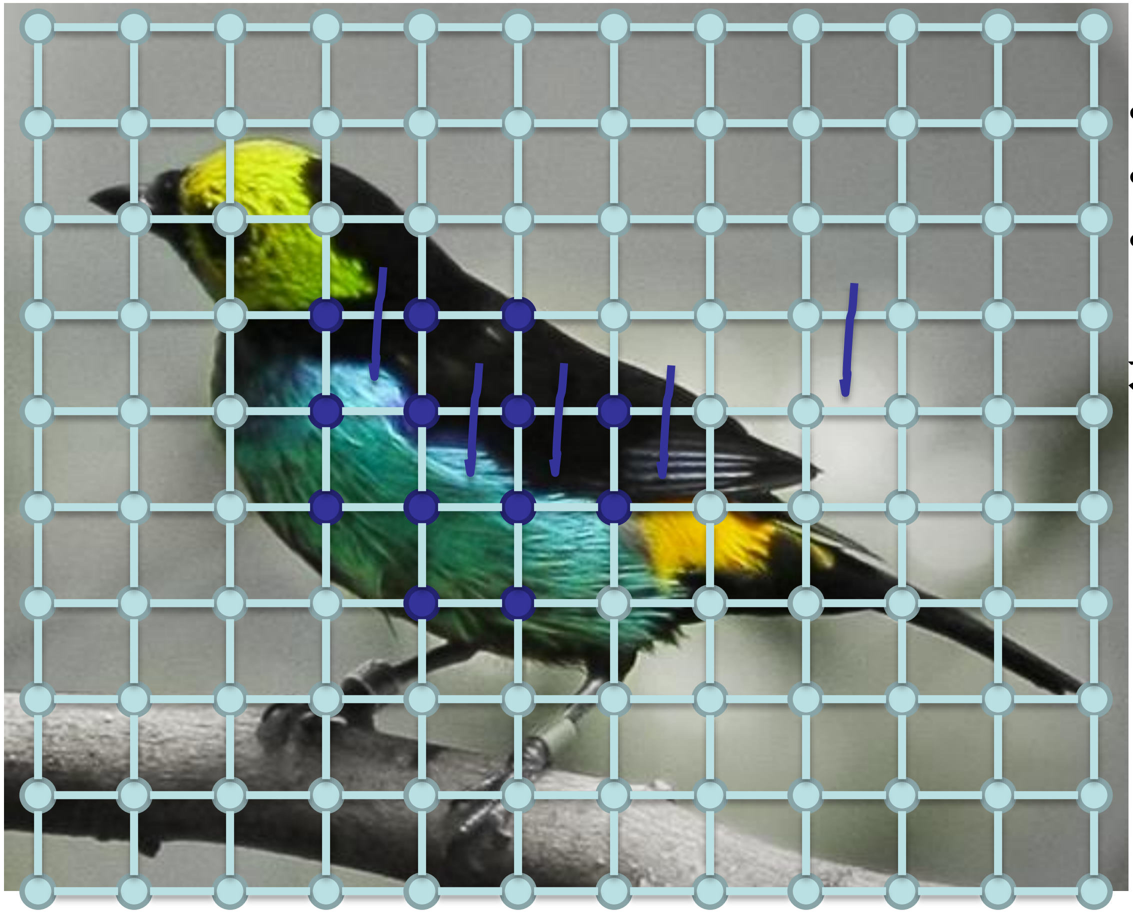
- 希望得到权重和的最大



- 每个 “cut” -> 一些点, W(I,j)
- 最优化问题
- W(i,j) = |LUV(i) – LUV(j)|

基于图的图像分割

- 希望得到权重和的最大

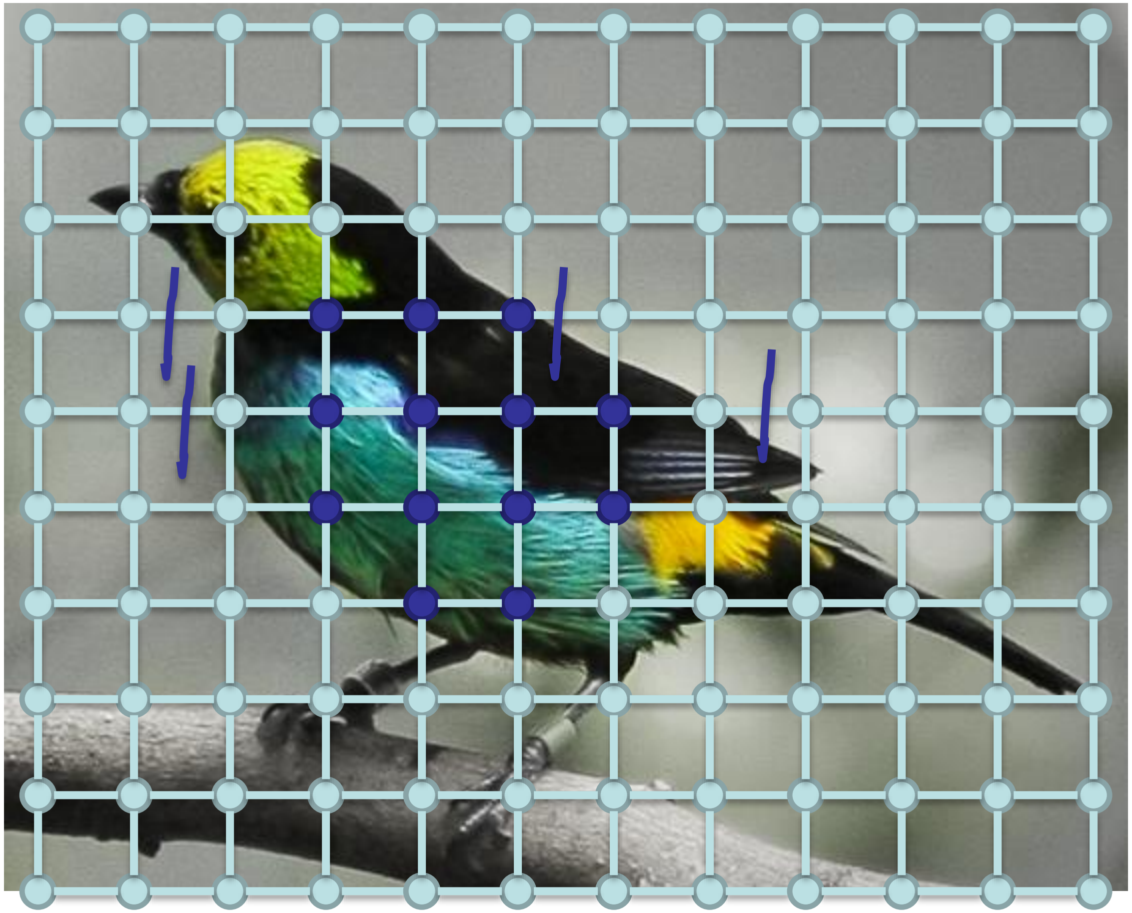


- 每个 “cut” -> 一些点, W(I,j)
- 最优化问题
- W(i,j) = |LUV(i) – LUV(j)|

这些Cuts作用在权重大的位置

基于图的图像分割

- 希望得到权重和的最大

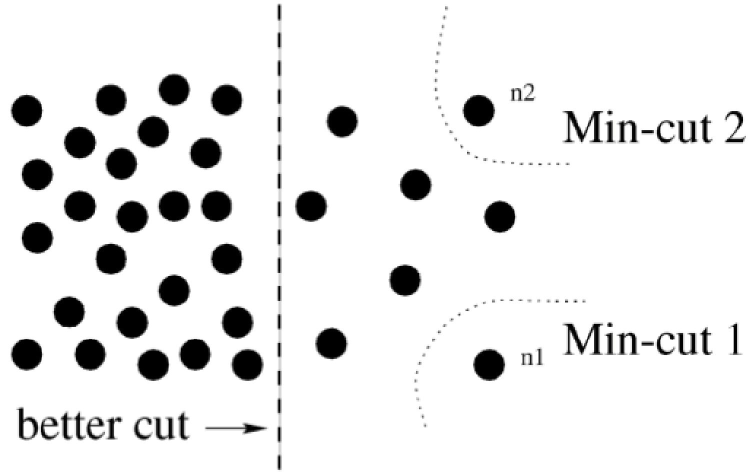


- 每个 “cut” -> 一些点, W(I,j)
- 最优化问题
- W(i,j) = |LUV(i) – LUV(j)|

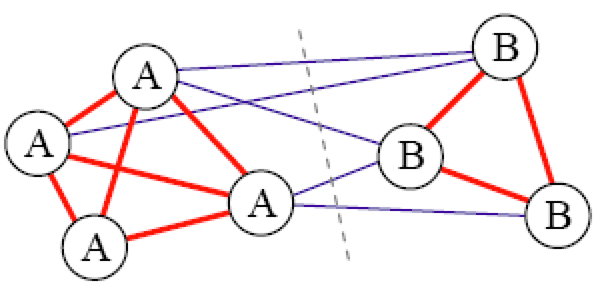
这些Cuts作用在权重小的位置

Normalized Cuts-Ncuts

- 为什么要归一化? – 避免噪声的影响!



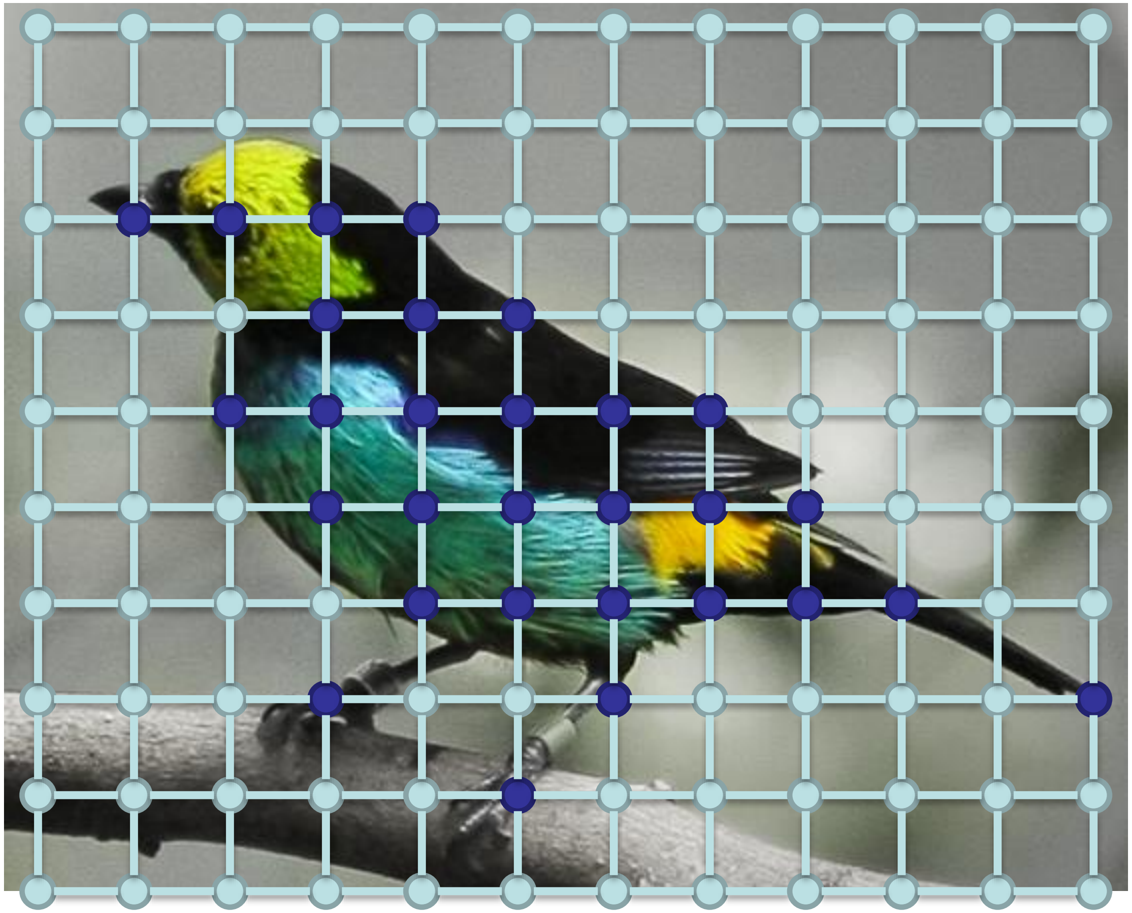
$$Ncut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)}$$



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

基于图的图像分割

- **最优化求解**



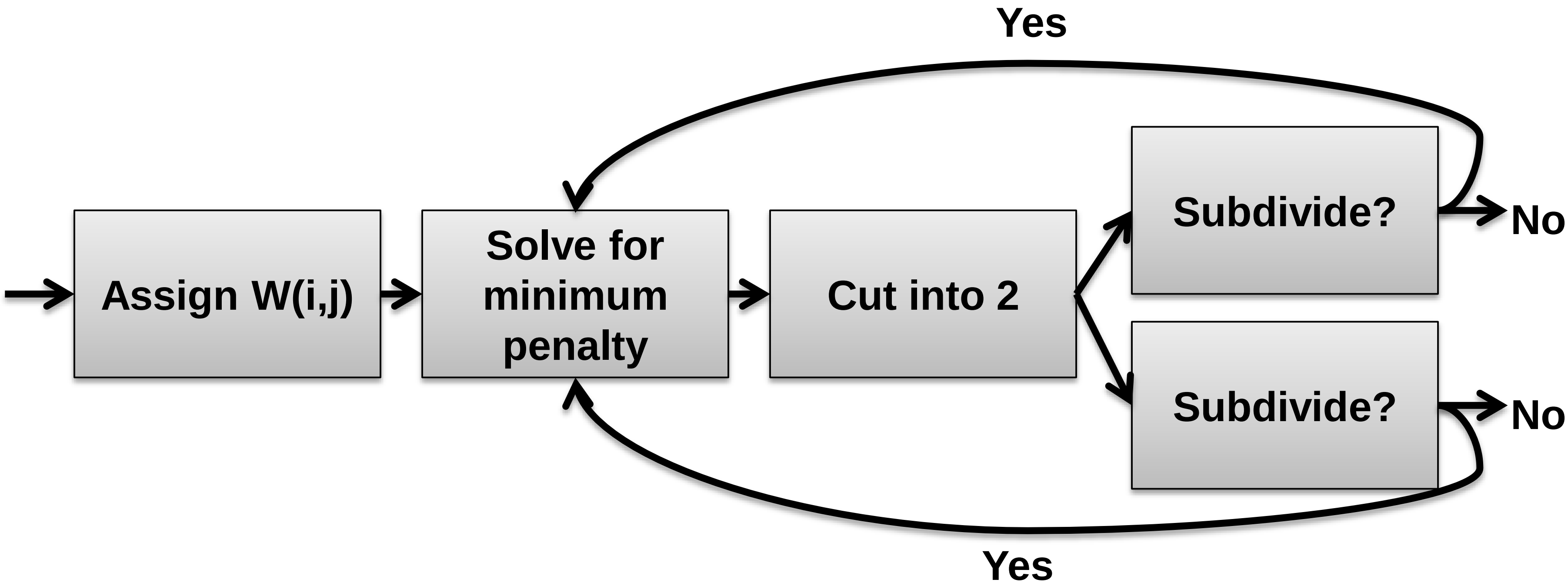
递归求解:
1.区域增长
2.If W(i,j) low
  1. **Stop**
  2. **Continue**

基于图的图像分割

- 只能得到孤立的解

算法流程

- 输入:图像
- 输出:分割块
- 每次迭代分割成两块



权重函数W(i,j)

- 颜色可能会包含比较多的噪声!
- 可以使用亮度和位置

$$w_{ij} = \boxed{e^{\frac{-\|\boldsymbol{F}_{(i)}-\boldsymbol{F}_{(j)}\|_2^2}{\sigma_I^2}}} * \boxed{\begin{cases} e^{\frac{-\|\boldsymbol{X}_{(i)}-\boldsymbol{X}_{(j)}\|_2^2}{\sigma_X^2}} & \text{if } \|X(i)-X(j)\|_2 < r \\ 0 & \text{otherwise.} \end{cases}}$$

亮度特征项

位置项

目标函数的最小化

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

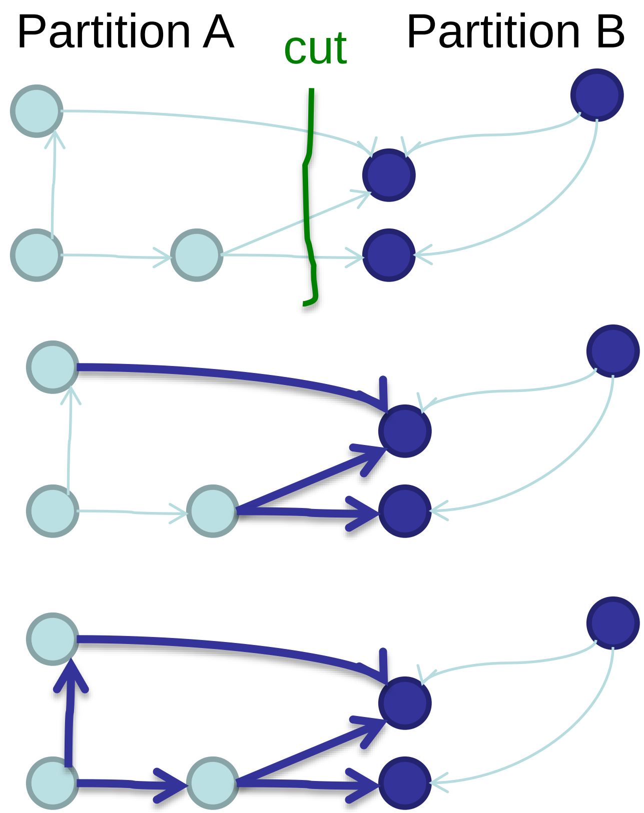
$cut(A,B)$ 与割连接的所有边的权重和

$assoc(A,V)$ 与子集**A**连接的所有边的权重和

通过对目标函数的最小化进行求解

$$Ncut(A,B)$$

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



通过对目标函数的最小化进行求解

- **求解归一化的拉普拉斯特征方程**

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

$$\mathbf{D}^{-\frac{1}{2}}(\mathbf{D}-\mathbf{W})\mathbf{D}^{-\frac{1}{2}}z = \lambda z$$

W(N x N) : weights associated with edges
D (N x N) : diagonal matrix with summation of all edge weights for the i-th pixel
N : number of pixels in the image

λ (N) : eigenvalues
z (N x N) : eigenvectors are real-valued partition indicator

O(N^3) complexity in general

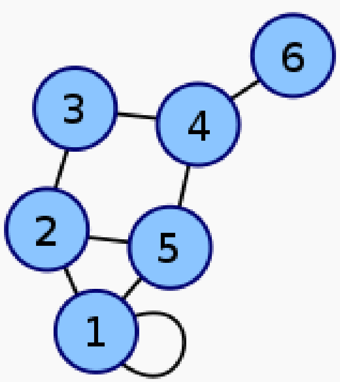
O(N^(3/2)) complexity in practice
a) Sparse local weights, b) Only need first few eigenvectors, c) Low precision

Laplacian Matrix

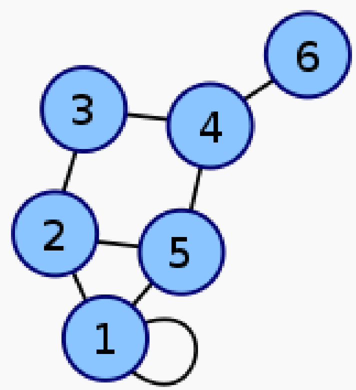
Laplacian matrix is the difference of the degree matrix and the adjacency matrix of the graph.

- Definition

degree matrix is a diagonal matrix which contains information about the degree of each vertex.

Vertex labeled graph	Degree matrix
	$\begin{pmatrix} 4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$

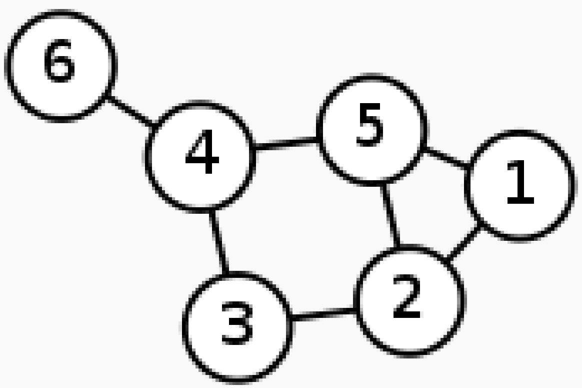
adjacency matrix is a means of representing which vertices of a graph are adjacent to which other vertices

Labeled graph	Adjacency matrix
	$\begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$

Laplacian Matrix

Laplacian matrix is the difference of the degree matrix and the adjacency matrix of the graph.

- Definition

Labeled graph	Laplacian matrix
	$\begin{pmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & -1 & 0 & -1 & 3 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{pmatrix}$

Laplacian matrix is a matrix representation of a graph

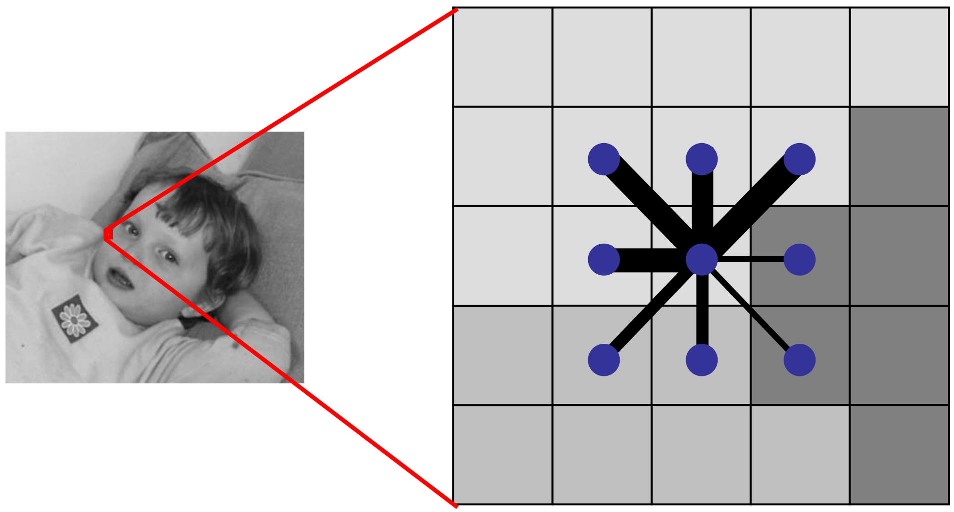
spectral graph theory

Gaussian Laplacian Matrix

Compare with the traditional weighted function:

$$L = D - W$$

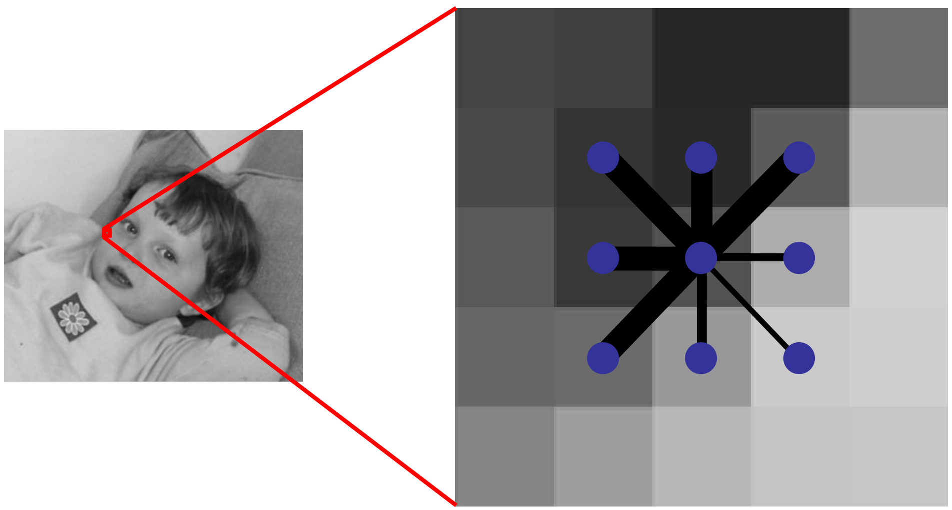
$$W_{Global}(i,j) = e^{-\|C_i - C_j\|^2 / \sigma^2}$$



Matting Laplacian Matrix

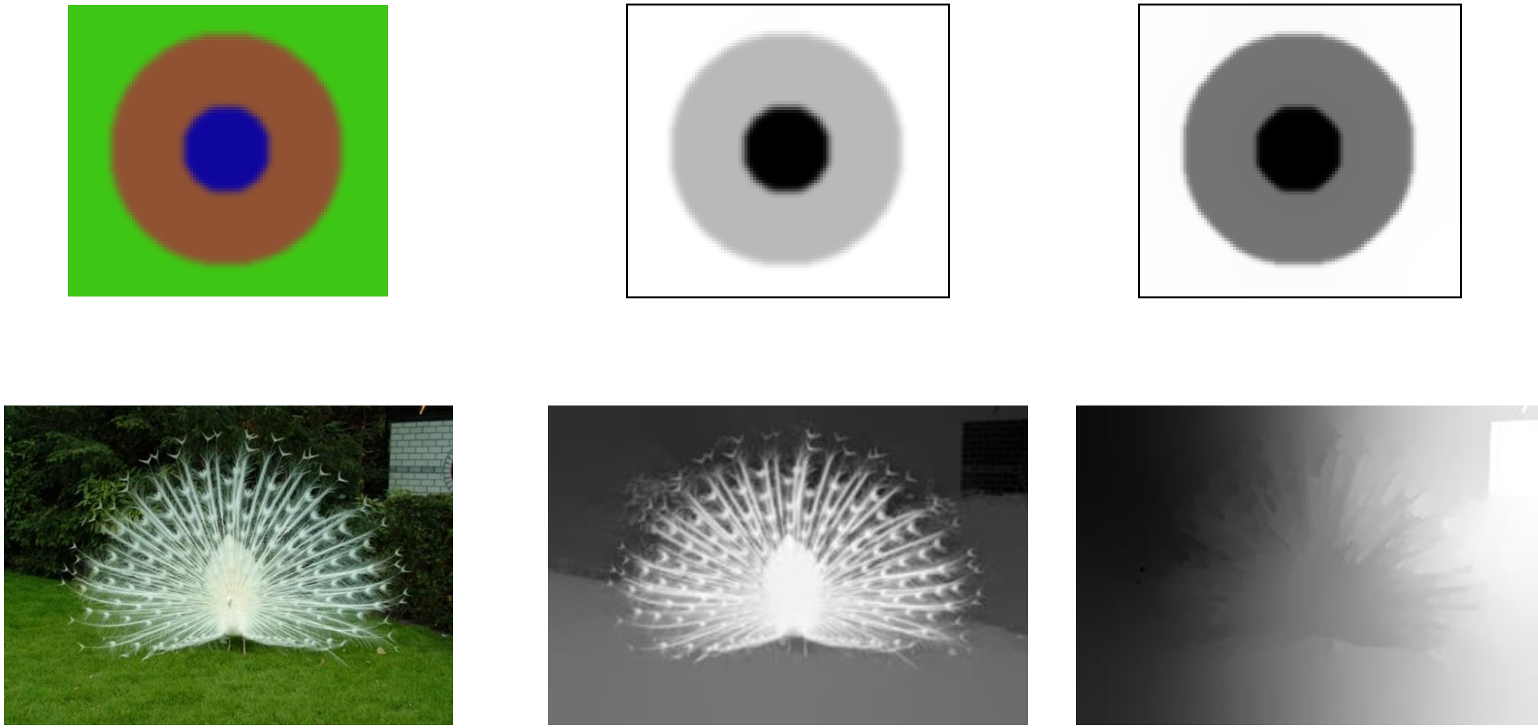
Compare with the traditional weighted function:

$$L = D - W \qquad W_{Matting}(i,j) \propto \sum_{k|(i,j)\in w_k} \left(1 + (C_i - \mu_k)^T (\Sigma_k + \varepsilon I_3)^{-1} (C_j - \mu_k)\right)$$



Using Mahalanobis Distance instead of the Euclidean Distance

Comparing eigenvectors



Input image

Matting Eigenvectors

Global-σ Eigenvectors

细分?

- 可以根据第二大的特征向量对图像进一步的分割
- $Ncut(A, B)$ < Threshold ?
  - Yes – stop here
  - No – continue to subdivide