第六章：图像识别—图像分类与检索

中国科学技术大学
电子工程与信息科学系

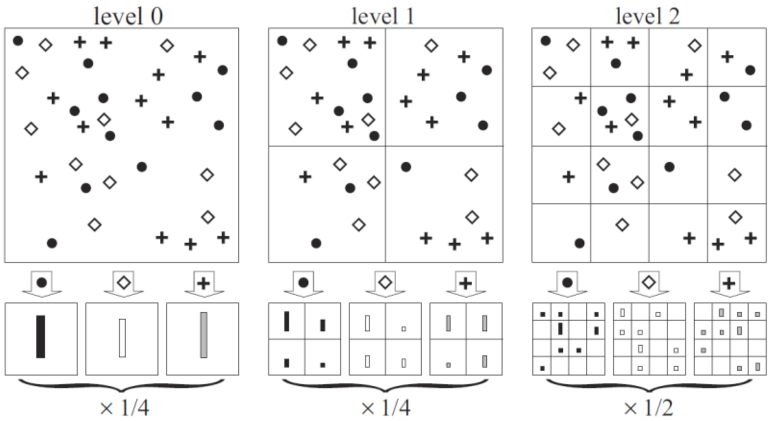
主讲教师：李厚强 (lihq@ustc.edu.cn)
周文罡 (zhwg@ustc.edu.cn)

图像分类与检索

- 图像分类
  - Spatial Pyramid Matching
  - KNN
  - SVM
- 图像检索

面向图像分类的空间金字塔匹配

- 空间金字塔匹配
  - 将图像划分为0, …, L个尺度



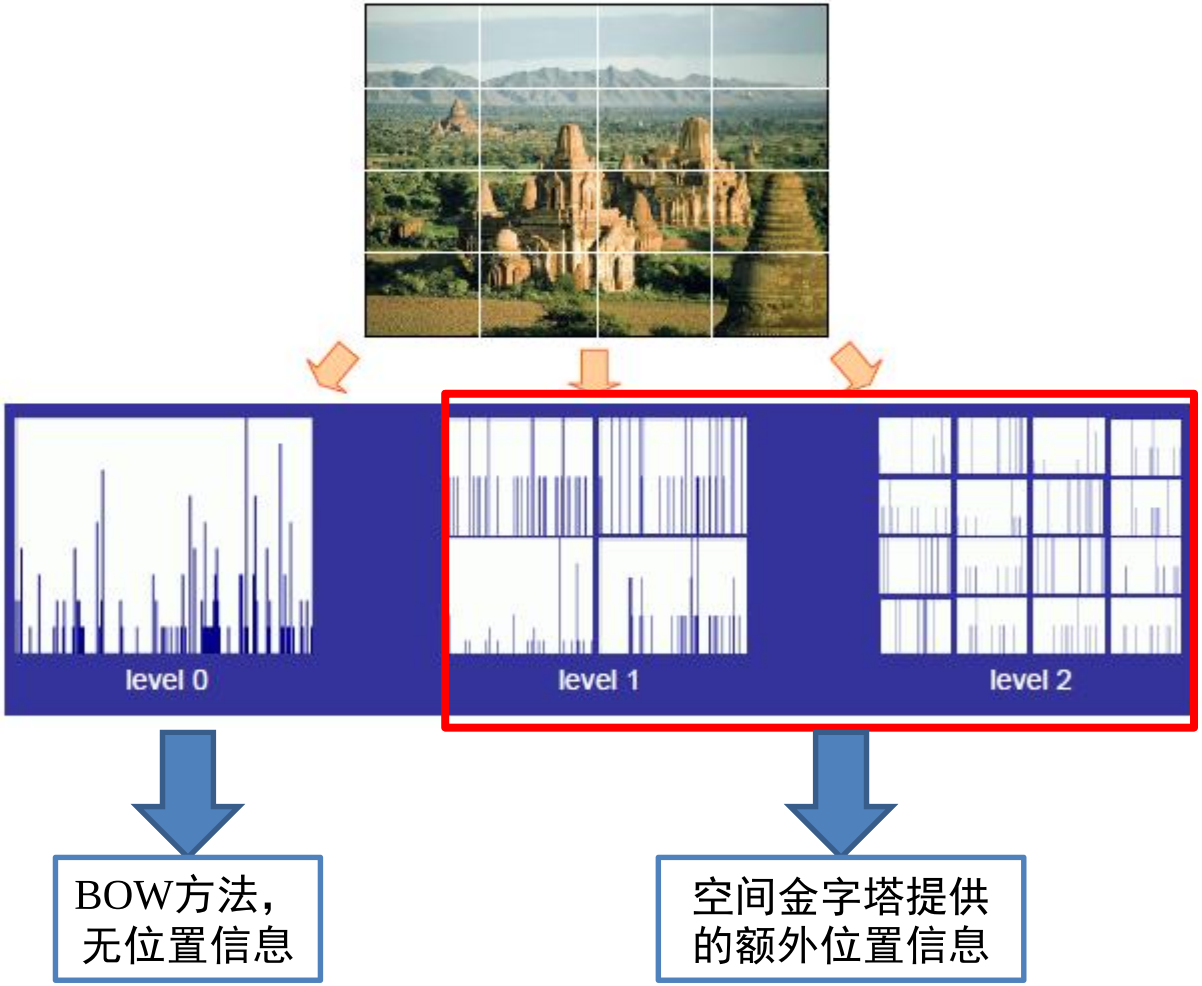
  - Pyramid match kernel

$$
\begin{aligned}
\kappa^L(X,Y) &= \mathcal{I}^L + \sum_{\ell=0}^{L-1} \frac{1}{2^{L-\ell}} \left(\mathcal{I}^\ell - \mathcal{I}^{\ell+1} \right) \\
&= \frac{1}{2^L} \mathcal{I}^0 + \sum_{\ell=1}^{L} \frac{1}{2^{L-\ell+1}} \mathcal{I}^\ell .
\end{aligned}
$$

- Lazebnik S, Schmid C, Ponce J. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. IEEE CVPR, 2006, 2: 2169-2178.

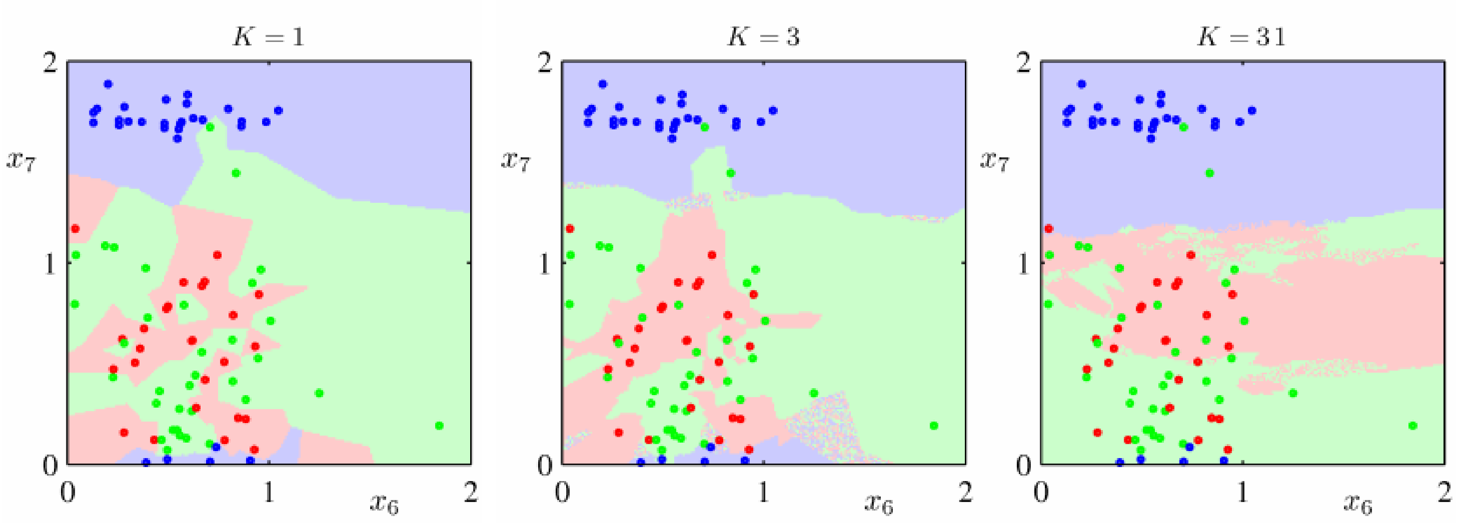
基于空间金字塔的图像分类

- 空间金字塔相比BOW的优点
  - 包含了局部特征的位置信息



图像分类模型

- K-NN分类器
  - 将类别预测转换为求解最近邻问题
  - 预测类别 → 数据库中近邻图像的类别（多数类别）
  - 参数K的影响



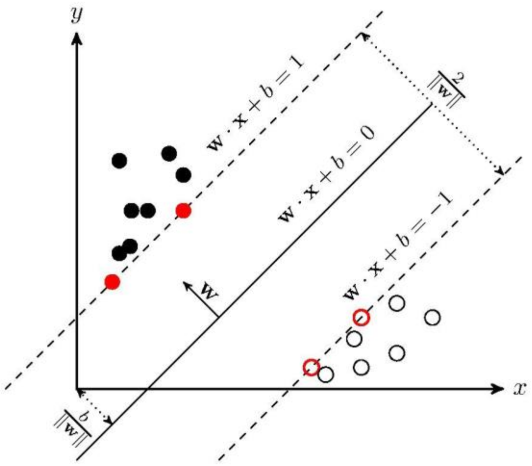
  - 优点：不需要训练分类模型，简洁
  - 缺点：依赖训练数据，预测速度慢（受ANN算法性能影响）

图像分类模型：SVM分类器

- 基于SVM的二类分类
  - 学习几何间隔最大的分离超平面

$$\min_{\boldsymbol{w},b} \ \frac{1}{2}\|\boldsymbol{w}\|^2$$

$$s.t. \ \ y_i\left(\boldsymbol{w}\cdot\boldsymbol{x_i}+b\right)\geq 1, \ i=1,2,\ldots,N$$



  - 将有约束的原始目标函数转换为无约束的拉格朗日目标函数

$$L\left(\boldsymbol{w},b,\boldsymbol{\alpha}\right)=\frac{1}{2}\|\boldsymbol{w}\|^2-\sum_{i=1}^{N}\alpha_i\left(y_i\left(\boldsymbol{w}\cdot\boldsymbol{x_i}+b\right)-1\right)$$

- 基于SVM的*K*类分类
  - One-versus-all rule：训练SVM分类器将某一类与其他(*K*-1)类分开
  - 对于每一类图像，均训练一个one-versus-all SVM分类器
  - 测试时，图像被分到响应最强的分类器所对应的类别

图像分类与检索

- 图像分类
- 图像检索
  - 倒排索引
  - 空间验证
  - 二值哈希

图像数据库索引：正向索引

- 正向索引
  - 每幅图像表达为一个维度为K的矢量
  - 遍历数据库图像，一一计算与查询图像的相似性得分

query

1
0.7
0
$\vdots$
0

	I_1	I_2		I_j		I_M
D_1	0	0.5	$\cdots$	0	$\cdots$	0.5
D_2	0.8	0.2	$\cdots$	0	$\cdots$	0
D_3	0	0	$\cdots$	1	$\cdots$	0
$\vdots$	$\vdots$	$\vdots$	$\cdots$	$\vdots$	$\cdots$	$\vdots$
D_K	0	0	$\cdots$	0.9	$\cdots$	0.1

$[s_1, s_2, \cdots, s_M]$

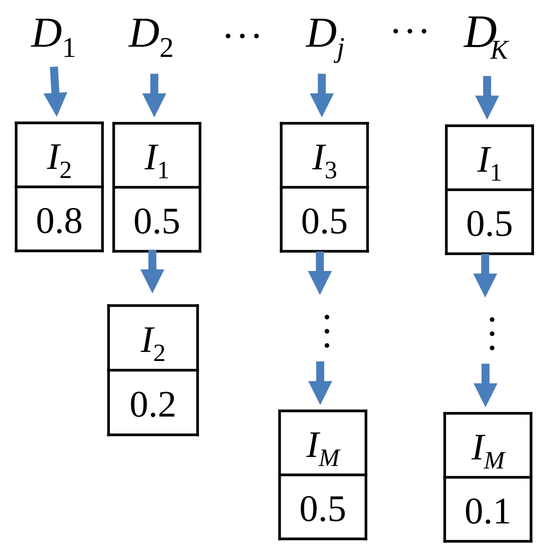
- 在大型数据集中， $M \gg K$ 。遍历M张图像逐一计算相似性，耗时太久
- 比较时，需要逐一比较两个向量的所有元素，计算冗余
- 如何改进？

图像数据库索引：倒排索引

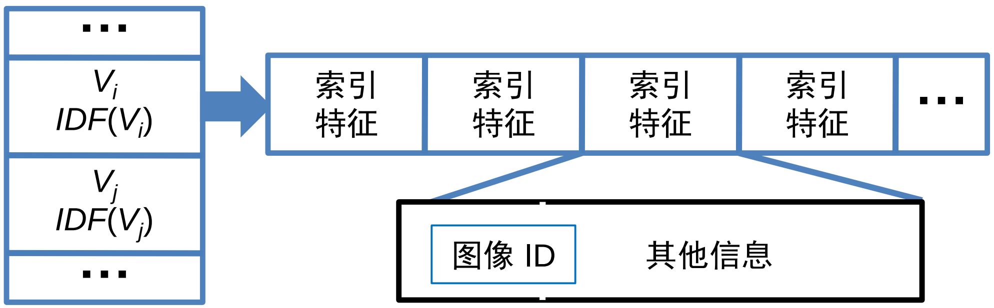
- □ 先验条件
  - ■ 图像视觉表征向量的稀疏性，非零元素比例低，比如< 1%
  - ■ 只需存储向量中的非零元素 → 视觉单词在词典中的索引
- □ 倒排索引的优势
  - ■ 高效的存储
  - ■ 高效的计算
    - ✓ 避免比较向量中的非0元素

$$
\begin{aligned}
D(I_q, I_m) &= \sum_{i=1}^{N} |q_i - m_i|^p \\
&= 2 + \sum_{i|q_i \neq 0, m_i \neq 0} (|q_i - m_i|^p - q_i^p - m_i^p)
\end{aligned}
$$

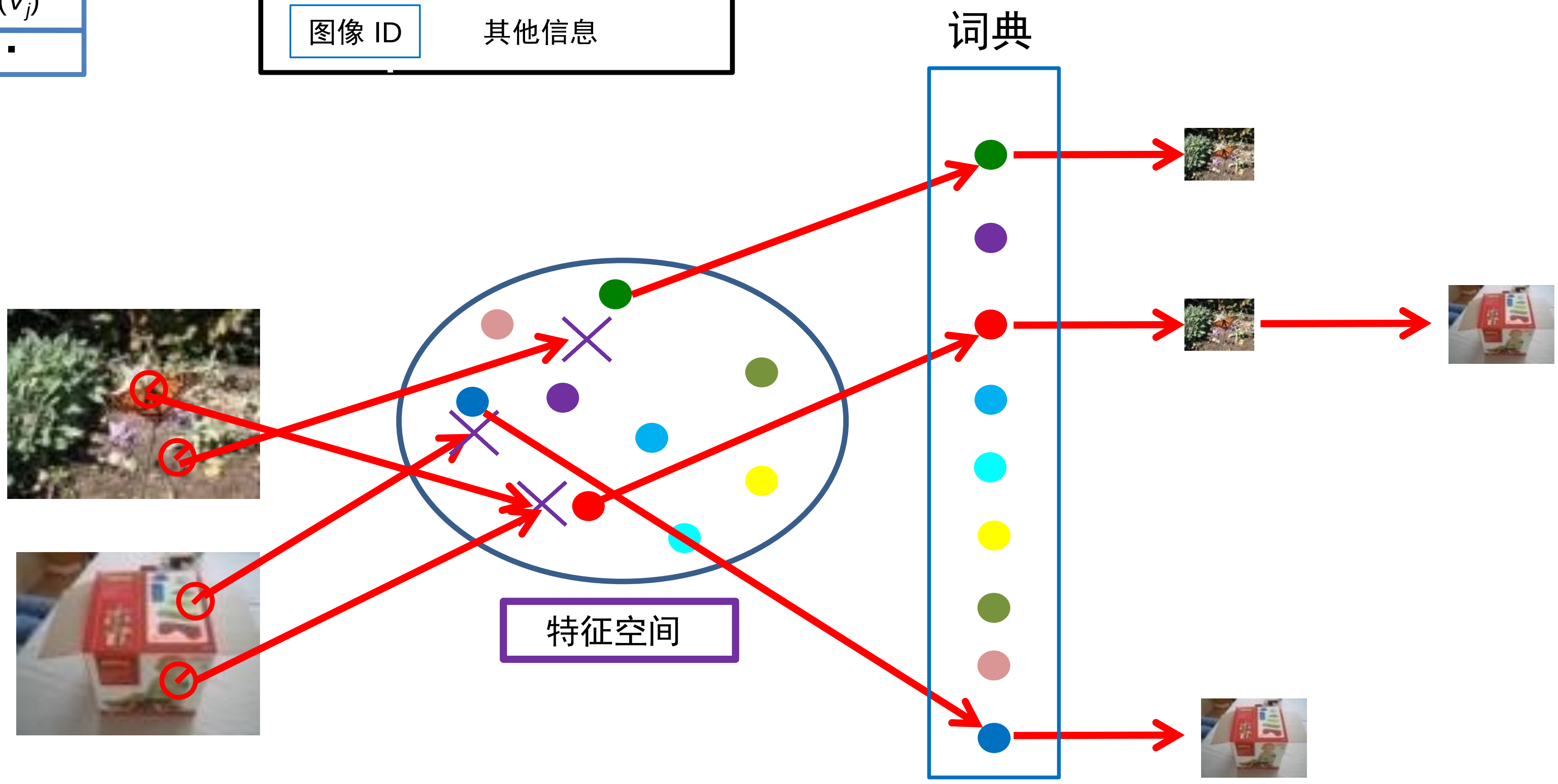
	D_1	D_2	$\cdots$	D_j	$\cdots$	D_K
I_1	0	0.5	$\cdots$	0	$\cdots$	0.5
I_2	0.8	0.2	$\cdots$	0	$\cdots$	0
I_3	0	0	$\cdots$	1	$\cdots$	0
$\vdots$	$\vdots$	$\vdots$	$\cdots$	$\vdots$	$\cdots$	$\vdots$
I_M	0	0	$\cdots$	0.9	$\cdots$	0.1



图像数据库索引：倒排索引



$$
\begin{aligned}
D(I_q, I_m) &= \sum_{i=1}^{N} |q_i - m_i|^p \\
&= 2 + \sum_{i|q_i \neq 0, m_i \neq 0} (|q_i - m_i|^p - q_i^p - m_i^p)
\end{aligned}
$$



图像分类与检索

- 图像分类
- 图像检索
  - 倒排索引
  - 空间验证
  - 二值哈希

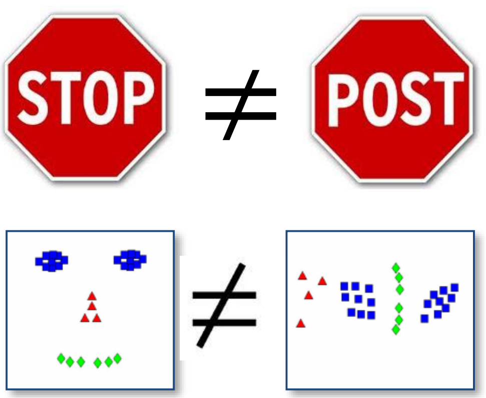
空间验证

- 动机
  - 局部特征匹配时缺少对位置信息的校验
  - 通过检验几何一致性去掉错误的匹配点对

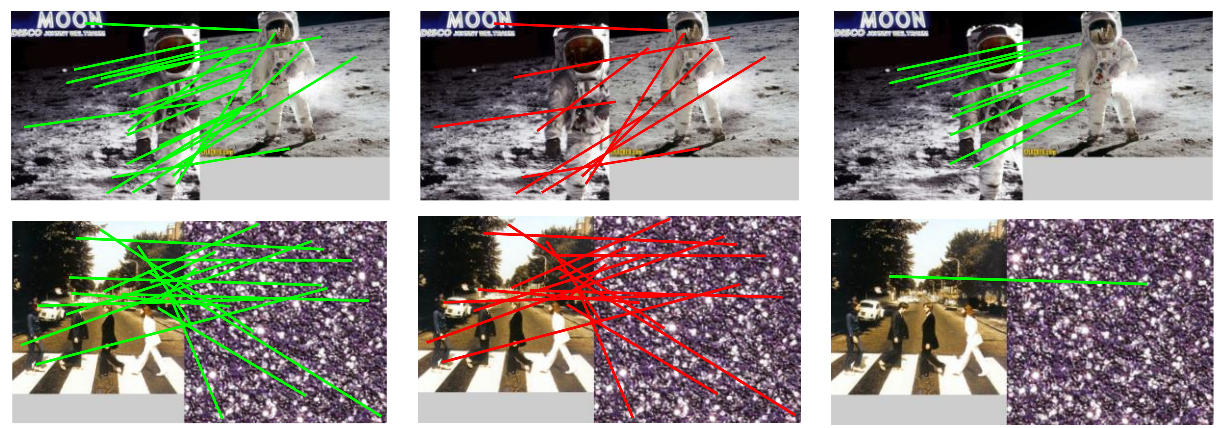
红线: 几何一致的匹配点对
蓝线: 几何不一致的匹配点对

视觉几何上下文表达

- 视觉单词以特定空间布局表达视觉语义
  - 利用几何上下文提升图像匹配质量
  - 有助于准确度量图像内容相关性



空间布局 vs. 视觉语义

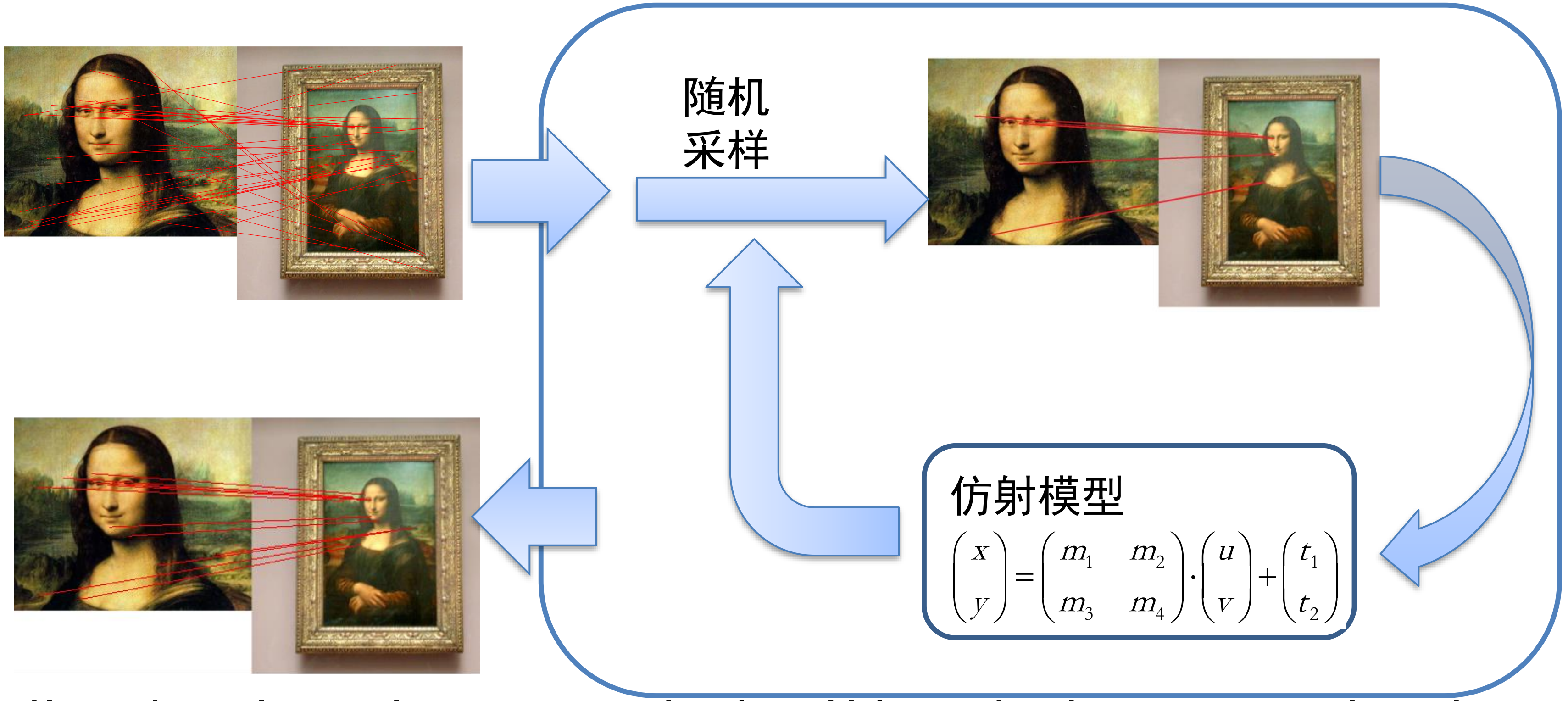


原始匹配　　噪声单词匹配　　几何一致的匹配

- 困难和挑战
  - 几何上下文结构化表达：便于图像匹配
  - 几何上下文快速匹配：保证实时检索

几何校验: RANSAC

- RANSAC算法示例
  - 通过匹配的特征点对估计图像的仿射变换



Fischler, *et al.*, **RAN**dom **SA**mple **C**onsensus: a paradigm for model fitting with applications to image analysis and automated cartography, *Comm. of the ACM*, 24:381-395, 1981

几何校验: RANSAC

- **RANSAC:**
  - 通过正确匹配点对估计仿射模型来排除错误匹配点对
  - inliers：正确的匹配点对
  - outliers：错误的匹配点对
- **RAN**dom **SA**mple **C**onsensus (RANSAC)的先验条件
  - 原始数据由inliers和outliers组成
  - inliers的子集可以正确的估计图像间的仿射变换
- 通过RANSAC估计仿射变换

1.迭代的随机选取匹配点对当作假设的*inliers*
2.根据假设的inliers计算一个仿射模型
3.其他数据点根据上述的仿射模型判断是否是*inliers*
4.通过所有的inliers重新估计仿射模型
5.通过所有的匹配点对与模型的拟合程度计算误差

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} m_1 & m_2 \\ m_3 & m_4 \end{pmatrix} \cdot \begin{pmatrix} u \\ v \end{pmatrix} + \begin{pmatrix} t_1 \\ t_2 \end{pmatrix}$$

- 缺点：由于随机采样点对估计模型要重复多次导致计算量大，计算复杂度为$O(N^3)$

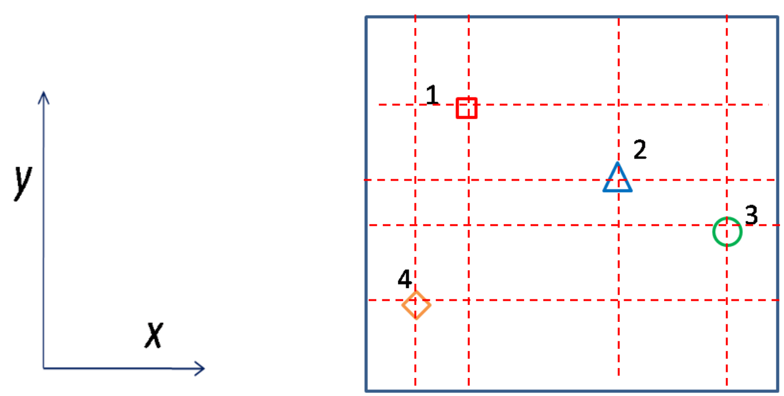
Fischler, *et al.*, **RAN**dom **SA**mple **C**onsensus: a paradigm for model fitting with applications to image analysis and automated cartography, *Comm. of the ACM*, 24:381-395, 1981

几何校验: 空间编码(Spatial Coding)

- **核心思想: 建立空间编码矩阵**
  - 匹配特征的相对空间位置关系

$$Xmap(i, j) = \begin{cases} 0 & \text{if } x_j > x_i \quad \text{right to } x_i \\ 1 & \text{if } x_j \le x_i \quad \text{left to } x_i \end{cases} \qquad Ymap(i, j) = \begin{cases} 0 & \text{if } y_j > y_i \quad \text{above } y_i \\ 1 & \text{if } y_j \le y_i \quad \text{below } y_i \end{cases}$$

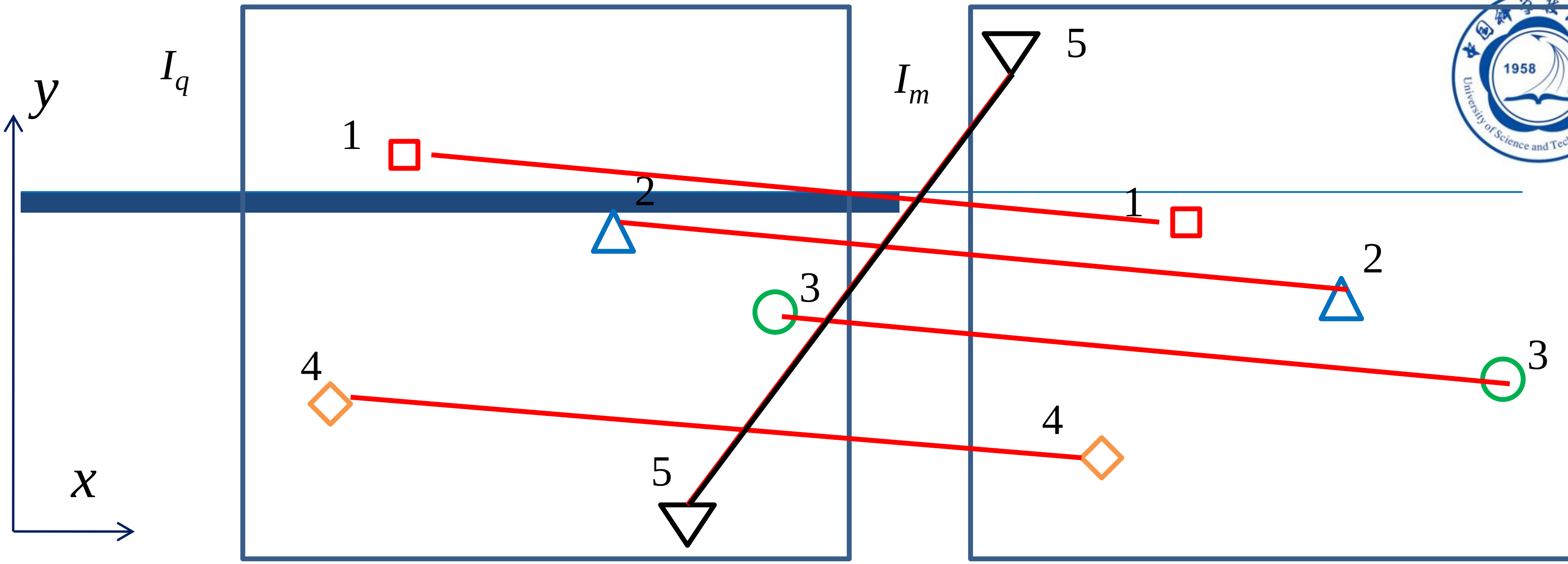
参考点: *i*



$$Xmap = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$Ymap = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Wengang Zhou, Yijuan Lu, Houqiang Li, Y. Song, and Qi Tian, "Spatial coding for large scale partial-duplicate web image search," *ACM International Conference on Multimedia* (*MM*), pp.131-140, 2010.



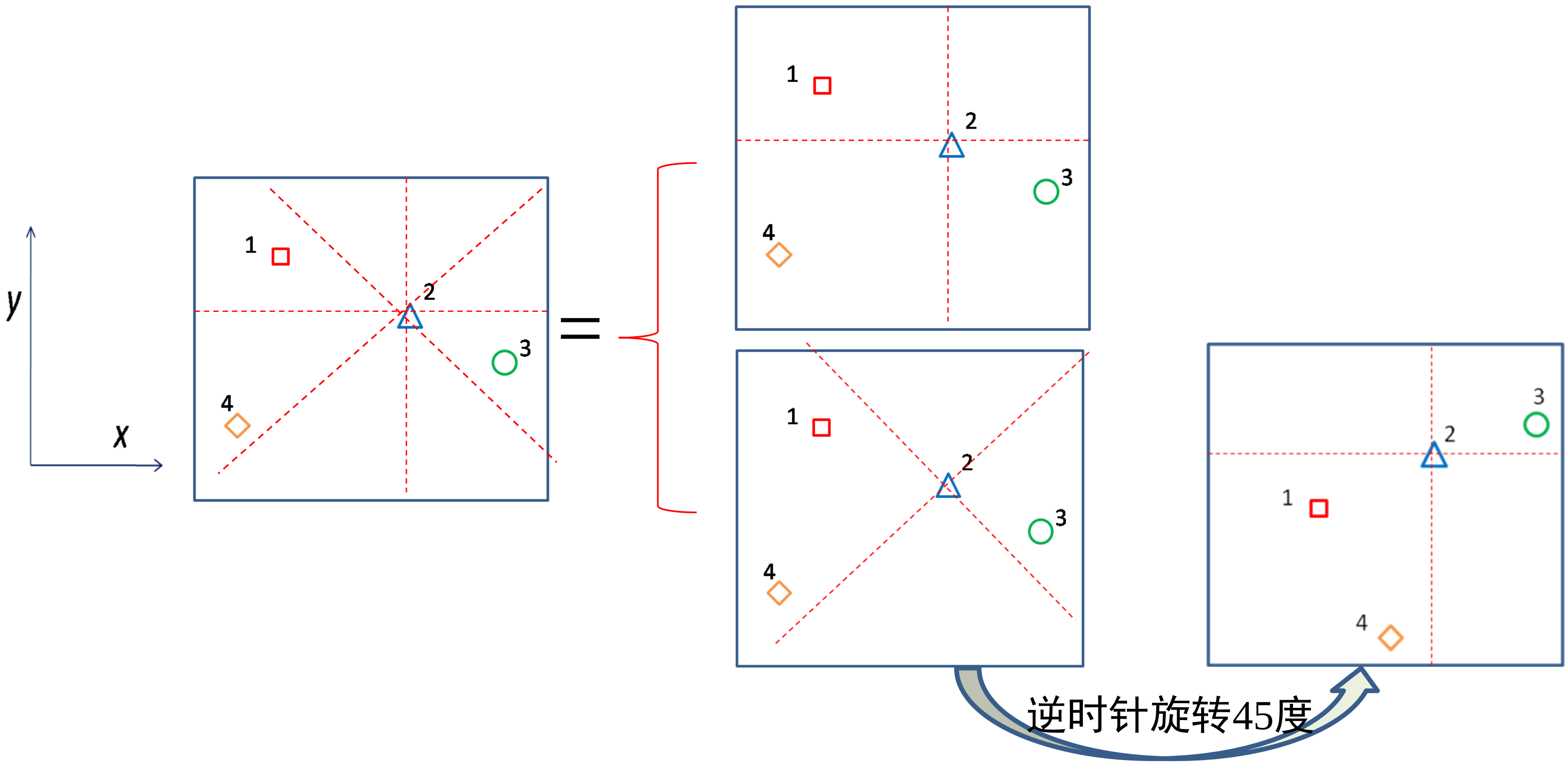
$$Vx = Xq \oplus Xm = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \end{pmatrix} \oplus \begin{pmatrix} 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{pmatrix}, \qquad Sx = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \\ \boxed{3} \end{pmatrix}$$

Row summation

$$Vy = Yq \oplus Ym = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \oplus \begin{pmatrix} 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{pmatrix}, \qquad Sy = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ \boxed{4} \end{pmatrix}$$

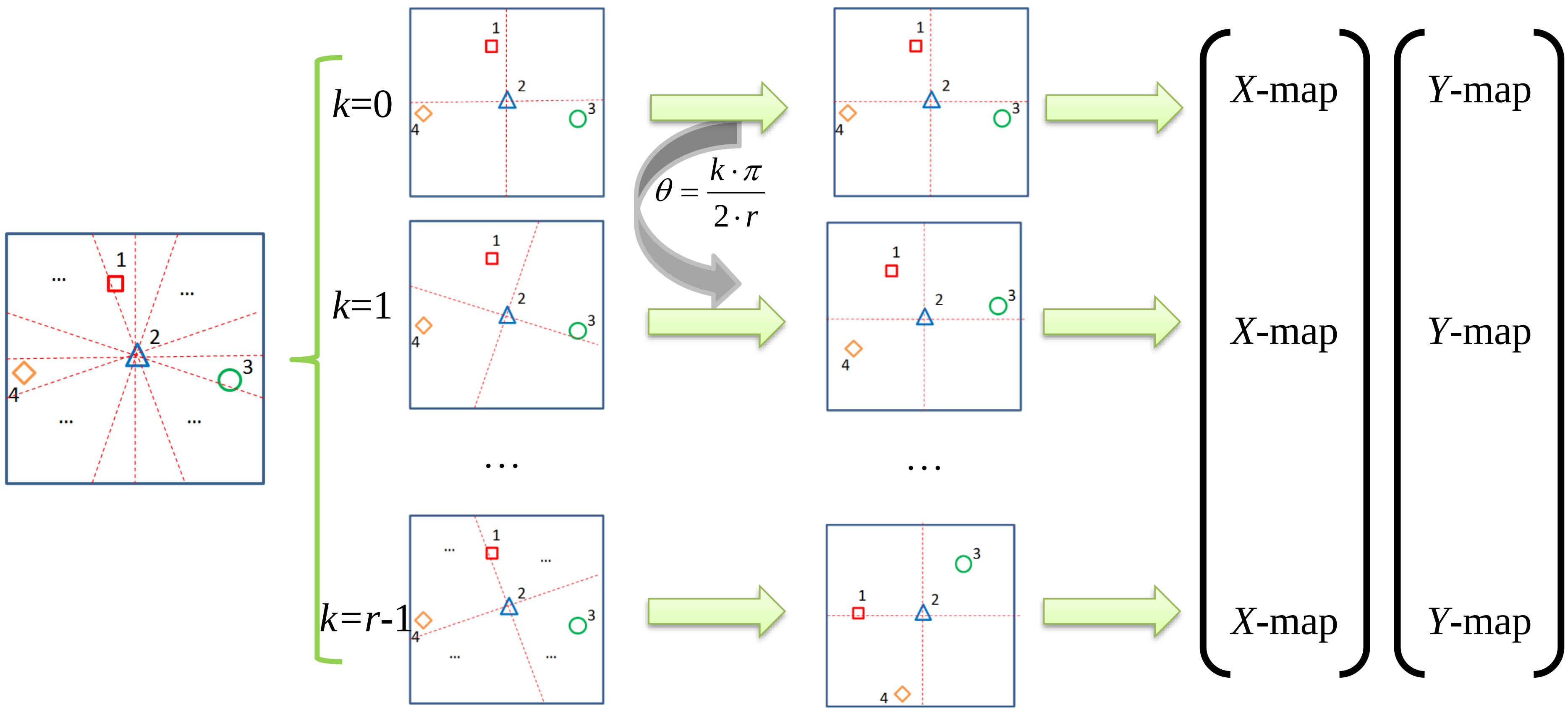
空间编码矩阵生成

- 在前面的例子中，每个象限只有一个部分
  - 现在将每个象限均匀的分成两个部分



空间编码矩阵生成

- 生成空间矩阵GX和GY
  - 每个象限均匀的分成r个部分



- Wengang Zhou, Yijuan Lu, Houqiang Li, Y. Song, and Qi Tian, "Spatial coding for large scale partial-duplicate web image search," *ACM International Conference on Multimedia* (*MM*), pp.131-140, 2010.

局部特征匹配的空间校验

- 基于空间矩阵GX和GY的验证
  - 将匹配特征对的空间矩阵进行对比

$$V_x(i,j,k) = GX_q(i,j,k) \oplus GX_m(i,j,k)$$ V_x: 空间矩阵X中不一致的程度

$$V_y(i,j,k) = GY_q(i,j,k) \oplus GY_m(i,j,k)$$ V_y: 空间矩阵Y中不一致的程度

$k=0, \ldots, r-1$; $i, j=1, \ldots, N$; N: 匹配特征对的数量

  - 迭代地查找和删除最不一致的匹配对

$$S_x(i) = \sum_{k=0}^{r-1}\sum_{j=1}^{N} V_x(i,j,k)$$

$$S_y(i) = \sum_{k=0}^{r-1}\sum_{j=1}^{N} V_y(i,j,k)$$

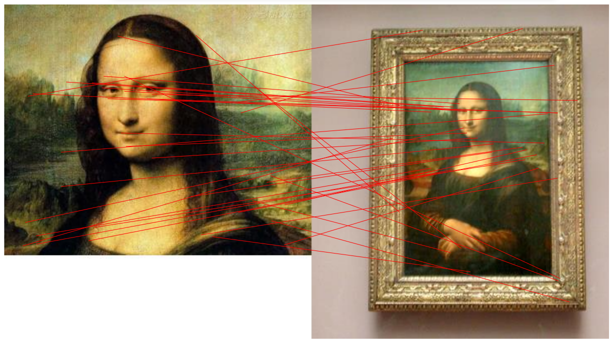
$$i^* = \arg\max_i S_x(i)$$

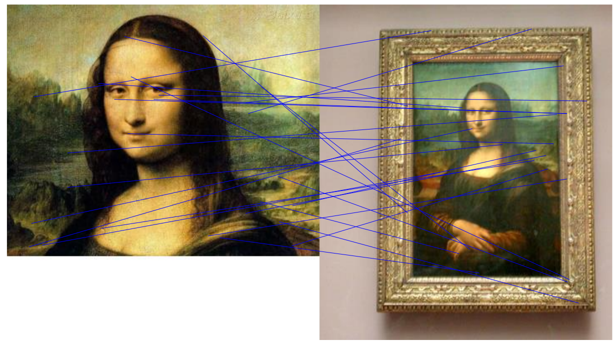
$$j^* = \arg\max_j S_y(j)$$

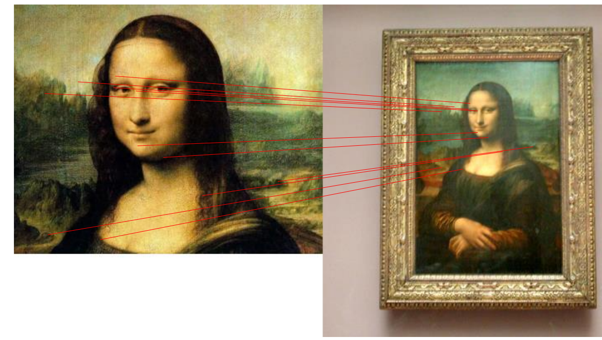
Identify i^*/ j^* and remove the corresponding row and column from both V_y and V_x

局部匹配的空间校验实例

相关图像







不相关图像

空间验证前

识别的错误匹配对

空间验证后

图像分类与检索

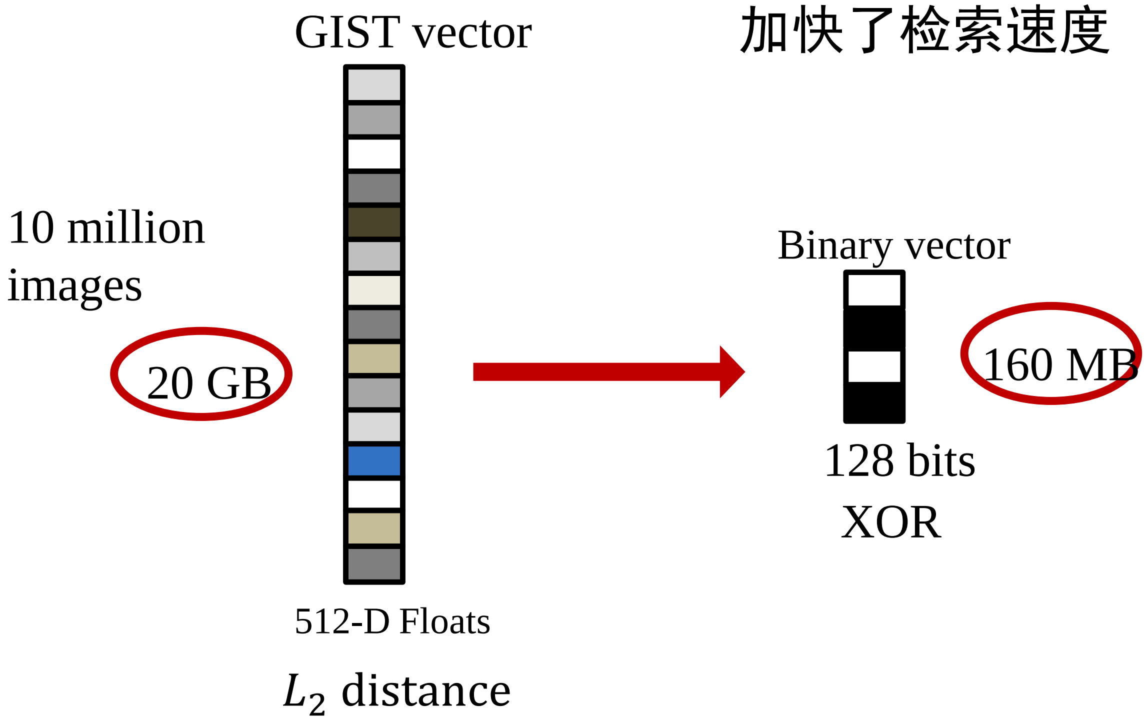
- 图像分类
- 图像检索
  - 倒排索引
  - 空间验证
  - 二值哈希

哈希算法

- 大规模数据集图像检索任务的要求
  - 存储开销
  - 检索速度
- 二值哈希算法

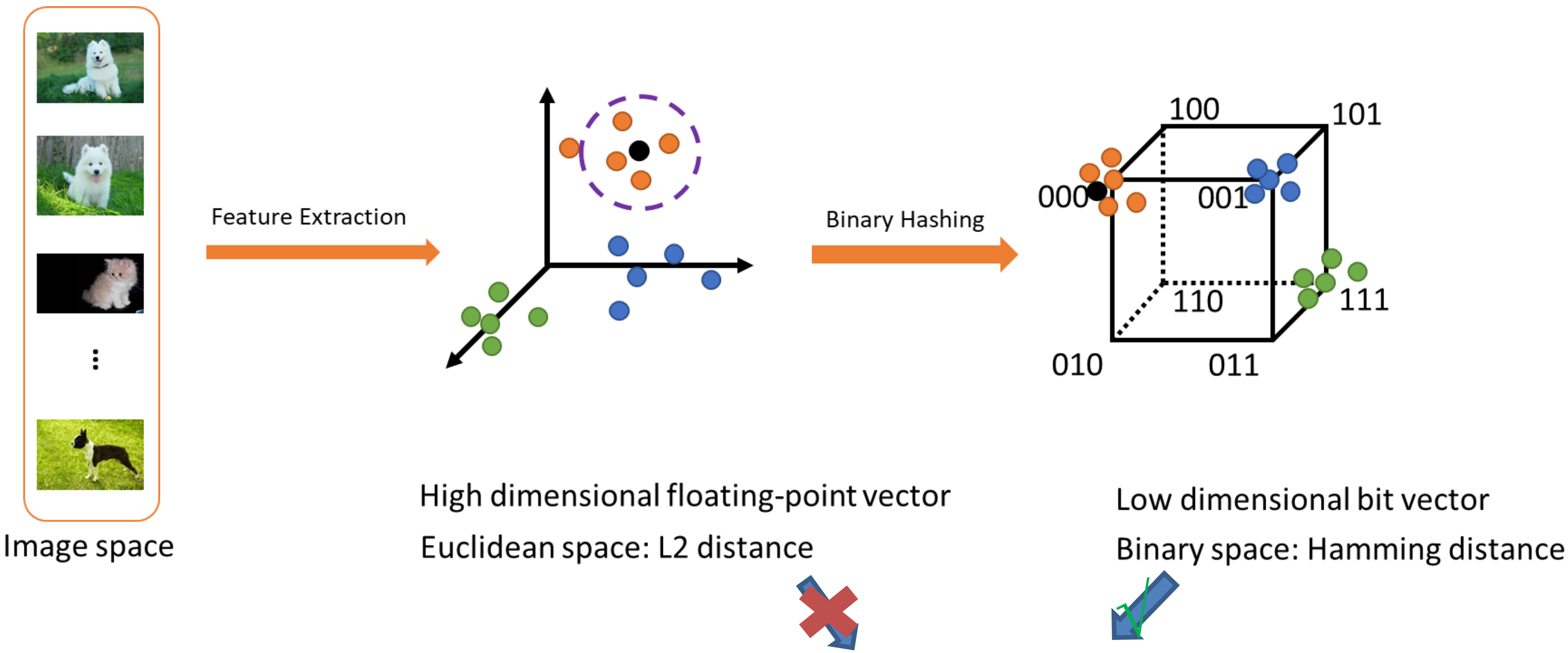
减小了存储开销

加快了检索速度



哈希算法

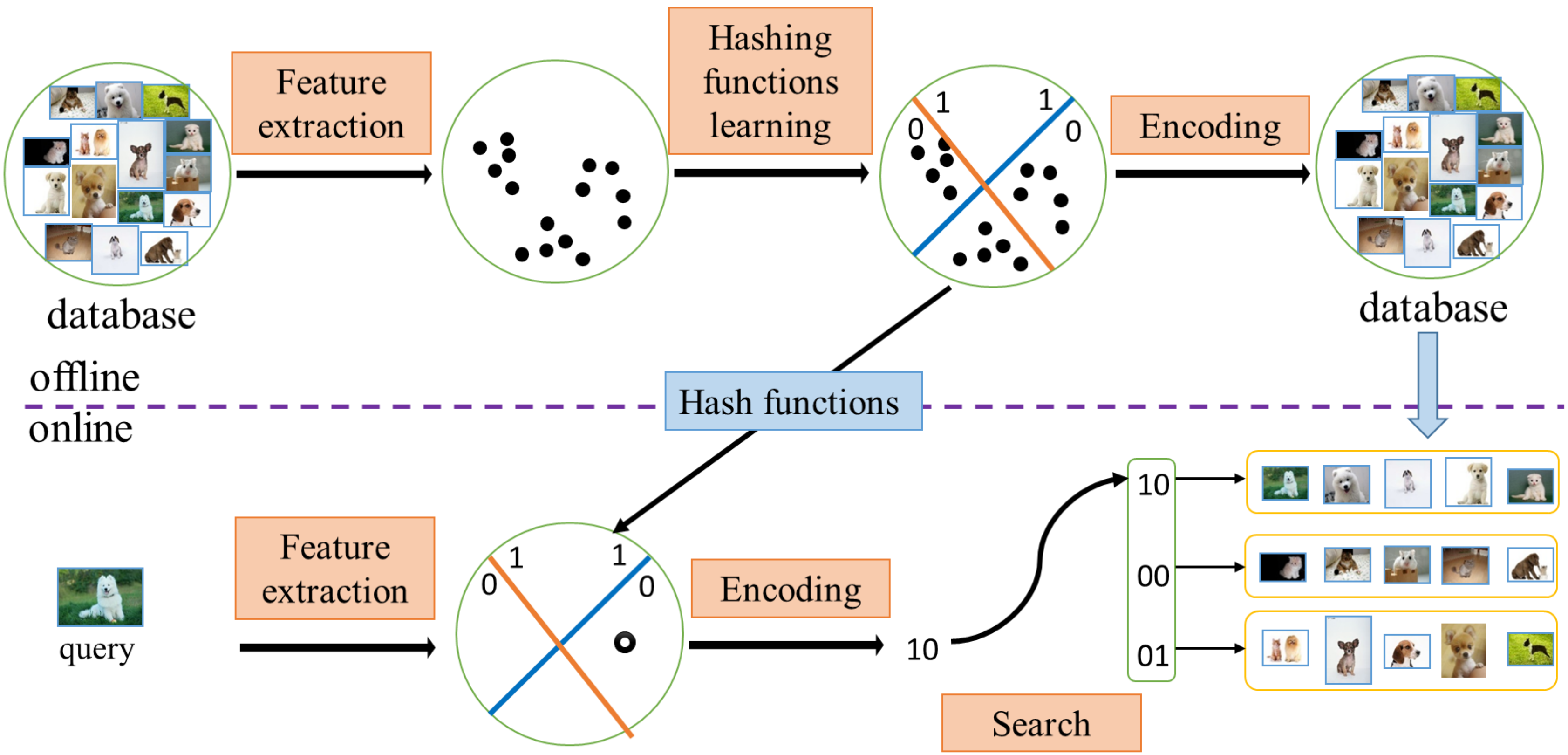
□ 哈希算法原理示意



欧式距离

哈希算法

- 哈希算法过程图例



哈希算法(1): 局部敏感哈希

- 局部敏感哈希定义
  - 高维空间的两点若距离很近，则这两点映射后的哈希值相同概率较大。
  - 若两点之间的距离较远，则他们哈希值相同概率较小。
- 正整数向量投影到汉明空间

n（2）维向量的数据集： A=(1,1) B=(2,1) C=(1,2) D=(2,2) E=(4,2) F=(4,3) ⇒ **坐标最大值：** C（4）

每个向量转换为n*C维哈希码：

值为k的坐标转换为长度为C的哈希码，前k位为1，后续位为0

A=(1,1) ⇒ (1000，1000) ⇒ 10001000

B=(2,1) ⇒ (1100，1000) ⇒ 11001000

....

F=(4,3) ⇒ (1111， 1110) ⇒ 11111110

哈希算法(1): 局部敏感哈希

- 一族哈希函数定义

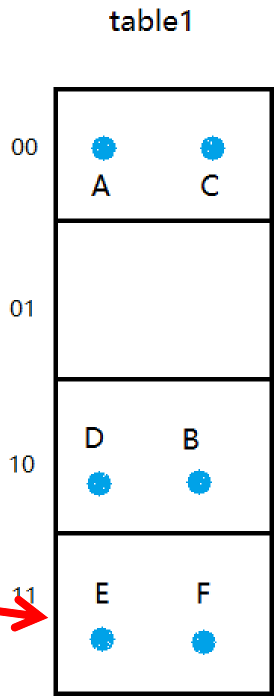
$$h_r(p)=\begin{cases} 0, 若p的第r位为0 \\ 1, 若p的第r位为1 \end{cases}$$

- 选择k个哈希函数组成构成哈希表g

A=10001000
B=11001000
C=10001100
D=11001100
E=11111100
F=11111110

g= $h_2(p), h_4(p)$ →

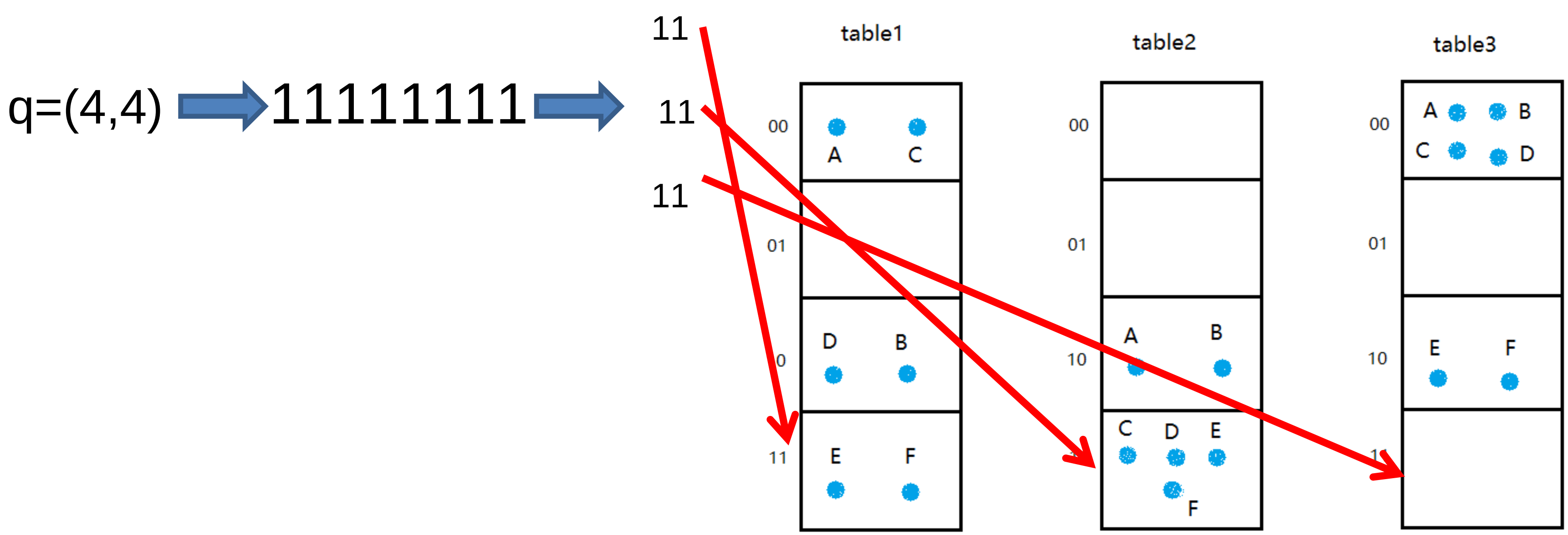
A=00
B=10
C=00
D=10
E=11
F=11



- 查询时查找被哈希表映射在同一个桶内的点
  - query q=(4,4) → 11111111 $\xrightarrow{h_2(p), h_4(p)}$ 11
  - 再将q与E,F比较，得到F是q的最近邻

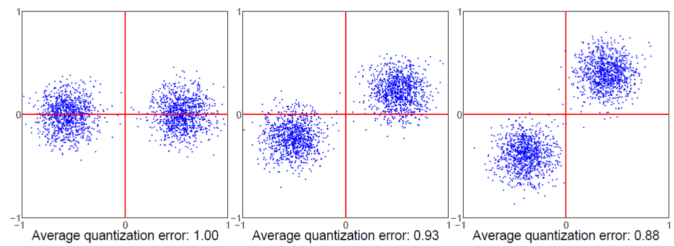
哈希算法(1): 局部敏感哈希

- 选择多组k个哈希函数组成构成多个哈希表g
  - 假设有如下结果。
    - ✓ g1分别抽取第2，4位。
    - ✓ g2分别抽取第1，6位。
    - ✓ g3分别抽取第3，8位

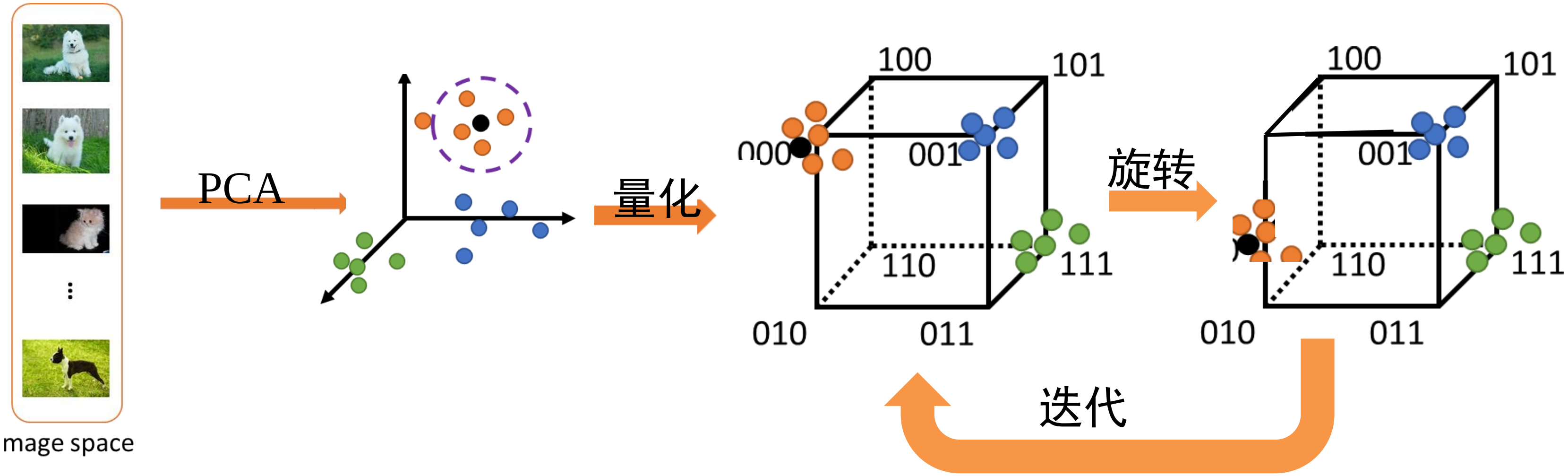


哈希算法(2): 迭代量化（ITQ）

- ITQ算法动机
  - 将原始数据映射到超立方体的顶点，求解量化误差最小的映射
  - 将超立方体在空间中旋转，求解旋转矩阵即能得到最好的映射
  - 迭代这两个步骤



(a) PCA aligned. (b) Random Rotation. (c) Optimized Rotation.



Yunchao Gong and Svetlana Lazebnik, "Iterative Quantization: A Procrustean Approach to Learning Binary Codes," *in CVPR 2011*.

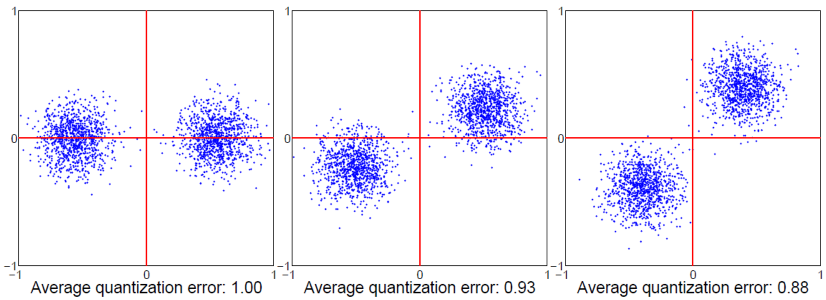
哈希算法(2): 迭代量化（ITQ）

- ITQ(Iterative Quantization)算法步骤
  - 对原始数据进行PCA降维

    $$V = XW$$

  - 最小化量化误差函数

    $$Q(B,R) = \|B - VR\|_F^2$$



(a) PCA aligned. (b) Random Rotation. (c) Optimized Rotation.

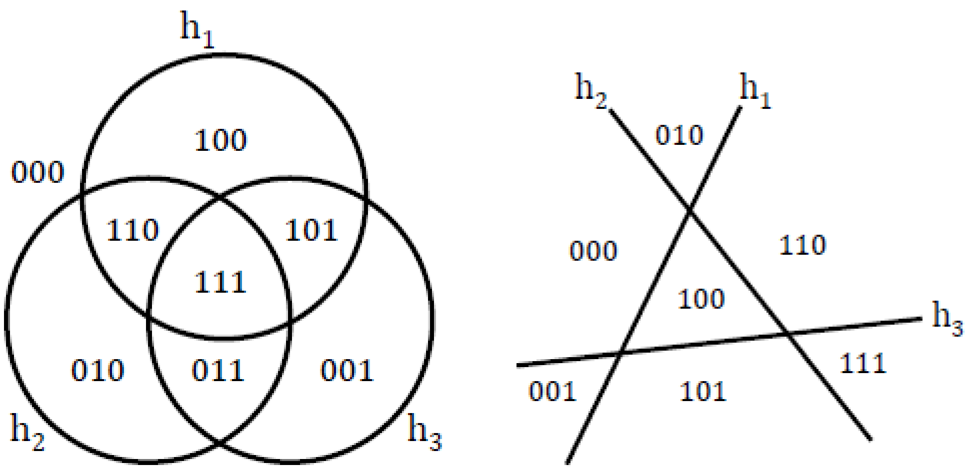
    - ✓ 固定R更新B: $B = \operatorname{sgn}(VR)$
    - ✓ 固定B更新R
      - ➢ 计算CxC矩阵 $B^T V$ 的SVD分解$S\Omega\hat{S}^T$ 然后令$R = \hat{S}S^T$
    - ✓ 迭代上述步骤，文中为五十次

- 优点
  - 没有显式的对量化过程作正交限制
  - 通过学习旋转矩阵代替了对汉明空间的操作

• Yunchao Gong and Svetlana Lazebnik, “Iterative Quantization: A Procrustean Approach to Learning Binary Codes,” *in CVPR 2011*.

哈希算法(3): 球面哈希

- 动机：用超球面而非超平面来分割空间
  - 特征空间更紧凑
  - 分割D维特征空间需要一个超球面，D+1个超平面
  - 局部敏感性较超平面更佳

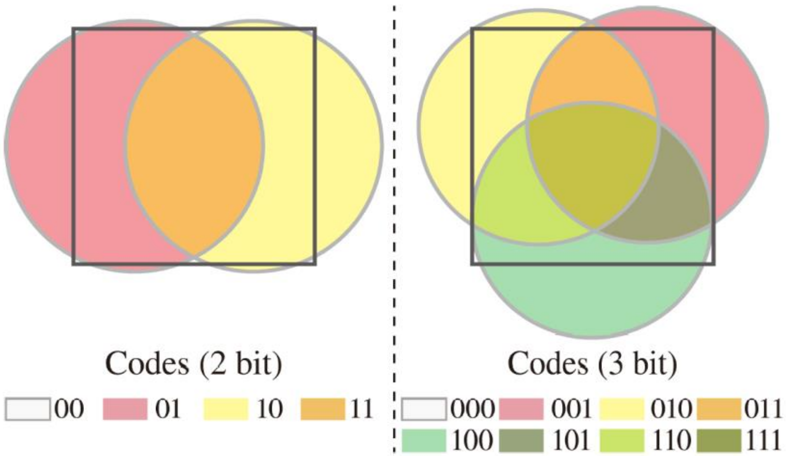


- 选择哈希函数即构建超球面：
  - 确定球心和半径

• Heo J P, Lee Y, He J, et al. Spherical Hashing: Binary Code Embedding with Hyperspheres, IEEE TPAMI, 2015.

哈希算法(3): 球面哈希

□ 球哈希示意



$$h_k(x) = \begin{cases} -1 & \text{when } d(p_k, x) > t_k \\ +1 & \text{when } d(p_k, x) \le t_k \end{cases}$$

□ 理想的超球面性质

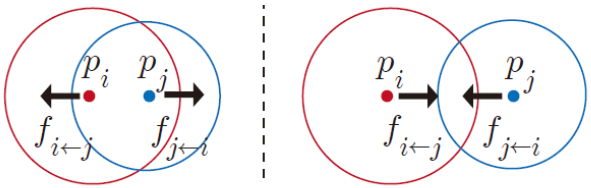
- 每个球把样本空间均分，即球内球外各占一半
- 每个球的交叉部分尽量少，即每个哈希函数相对独立

$$\begin{aligned} o_i &= \left| \{ s_k | h_i(s_k) = +1, 1 \le k \le m \} \right|, \\ o_{i,j} &= \left| \{ s_k | h_i(s_k) = +1, h_j(s_k) = +1, 1 \le k \le m \} \right|, \end{aligned}$$

- Heo J P, Lee Y, He J, et al. Spherical Hashing: Binary Code Embedding with Hyperspheres, IEEE TPAMI, 2015.

哈希算法(3): 球面哈希

- 对于训练样本点集$S = \{s_1, s_2, \cdots, s_n\}$，迭代确定超球面
  - 1. 初始化：从训练样本中随机选l个点作为初始球心$p_1, p_2, \cdots, p_l$;
  - 2. 对各个球心，确定半径$t_1, t_2, \cdots, t_l$，使得$o_l = \frac{n}{2}$;
  - 3. 对每一对哈希函数，计算$o_{i,j}$;
  - 4. $\forall i, j$, 计算$f_{i \leftarrow j} = \frac{1}{2} \frac{o_{i,j} - n/4}{\frac{n}{4}} (p_i - p_j)$



  - 5. $\forall i$,计算$f_i = \frac{1}{l} \sum_{j=1}^{l} f_{i \leftarrow j}$， $p_i = p_i + f_i$
  - 6. 重复步骤2~5，直至收敛，即满足下述条件

$$avg\left(\left|o_{i,j} - n/4\right|\right) < \varepsilon_m \frac{m}{4} \text{ 且 } std - dev(o_{i,j}) < \varepsilon_s \frac{m}{4}$$

- 基于球哈希定义的汉明距离计算：

$$d_{shd}(b_i, b_j) = \frac{|b_i \oplus b_j|}{|b_i \wedge b_j|}$$

其中 $\oplus$: 异或；$\wedge$：逻辑与

- Heo J P, Lee Y, He J, et al. Spherical Hashing: Binary Code Embedding with Hyperspheres, IEEE TPAMI, 2015.