

3D感知技术与实践

人体、物体识别与点云粗配准

内容

- 人体肢体识别
- 物体识别与点云配准

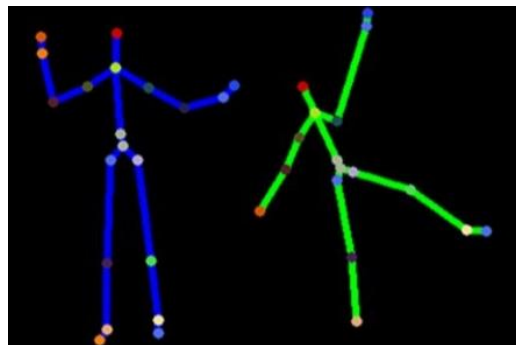
内容

- 人体肢体识别
- 物体识别与点云配准

人体动作识别

计算机视觉邻域有大量和
人体动作识别有关的应用

- 行为识别
- 运动员训练
- 人机交互
- 体感游戏/实时VR合成
- 护理/医学诊疗



人体动作识别

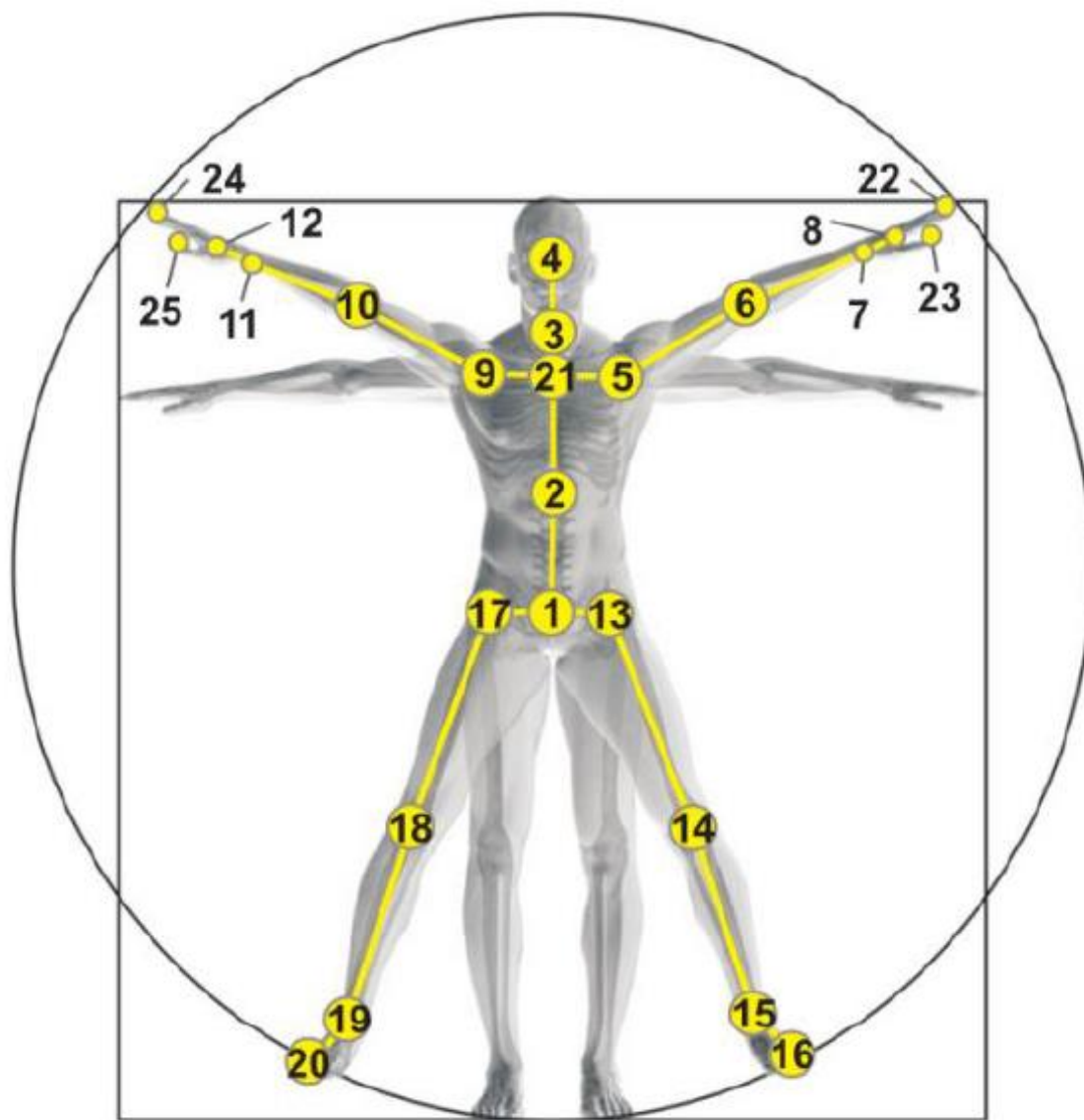
- 两种动作识别的途径

- 直接法——由点云获得动作
- 间接法——先从点云得到骨架，再通过骨架得到动作

- 后面的内容围绕骨架提取方法进行介绍

- 骨架的模型

- 不是严格医学意义上的骨架
- 25关节的简易“骨架模型”



人体骨架提取——基于RGB和神经网络的方法

- CMU的OpenPose是比较成功的基于RGB的骨架提取算法，能够同时识别多人动作，它使用深度神经网络实现
- 运算量很大，需要GPU实现，运算效率低、但RGB数据获取方便

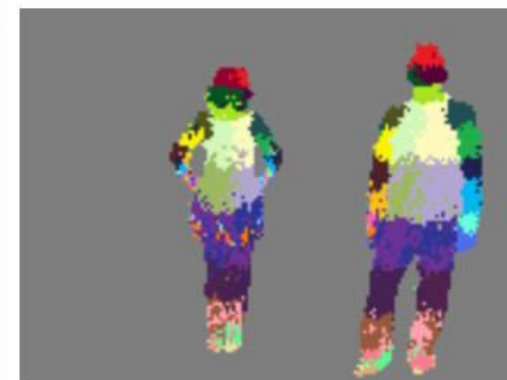


人体骨架提取——基于深度图

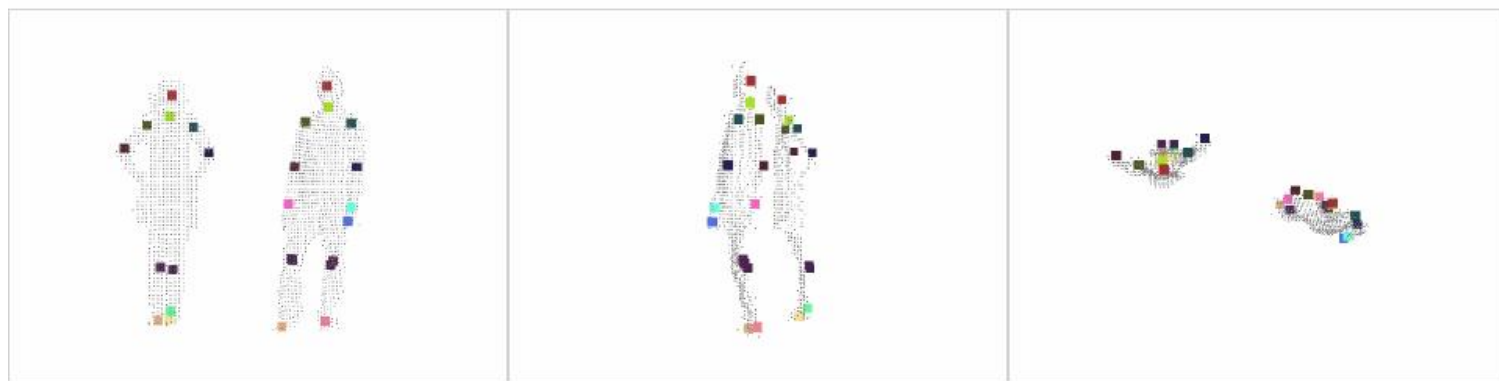
- 相比RGB人体骨架识别方法，从深度图能够更加高效的获取人体骨架数据，可以在CPU甚至嵌入式系统上实现运算

- 为什么能够高效？

- 简单快速前背景分离
- 得到真实物理尺寸，不受距离影响
- 利用人体表曲面信息



- 右图识别人体时，去除了背景，使得问题简化
- 通过3D空间点云分割，分离两个人的点云，分别进行识别



人体骨架提取

Real-Time Human Pose Recognition in Parts from Single Depth Images

- 获得骨架的主要流程
- 右边看到每个环节对应的数据可视化效果

- 这一部分看上去最困难
- 如何实现的?

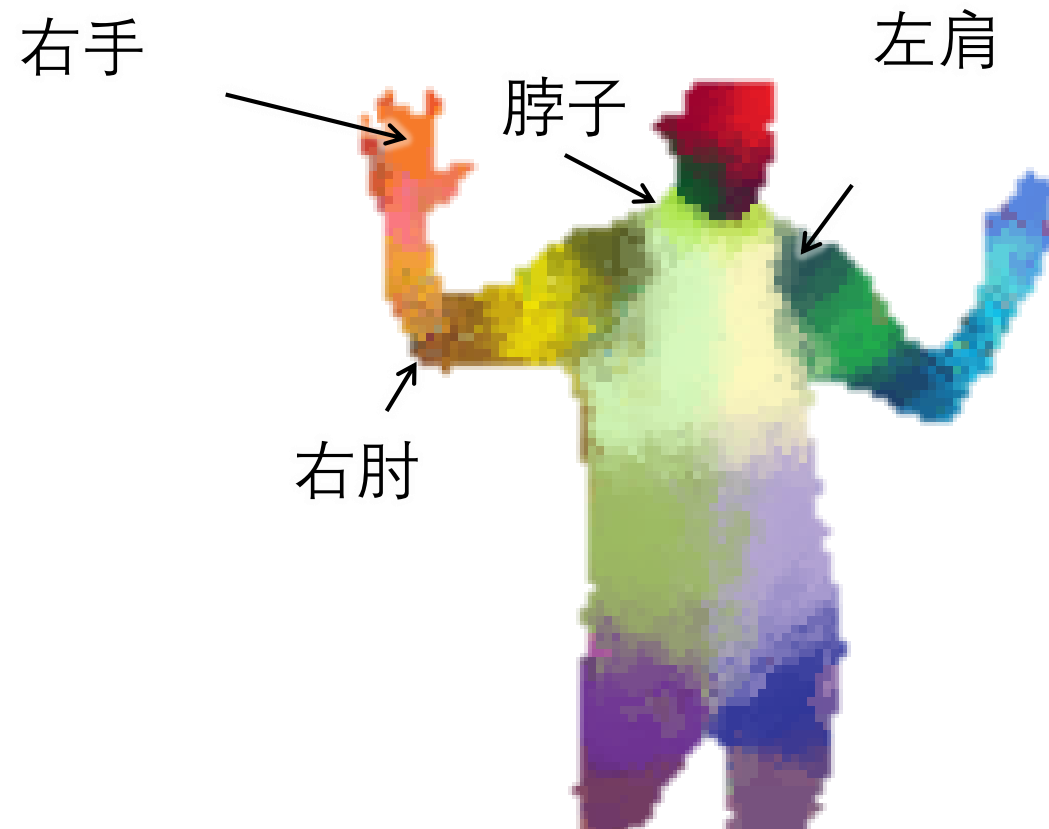


人体骨架提取

Real-Time Human Pose Recognition in Parts from Single Depth Images

获得骨架的主要流程

- 深度图上逐像素的分类

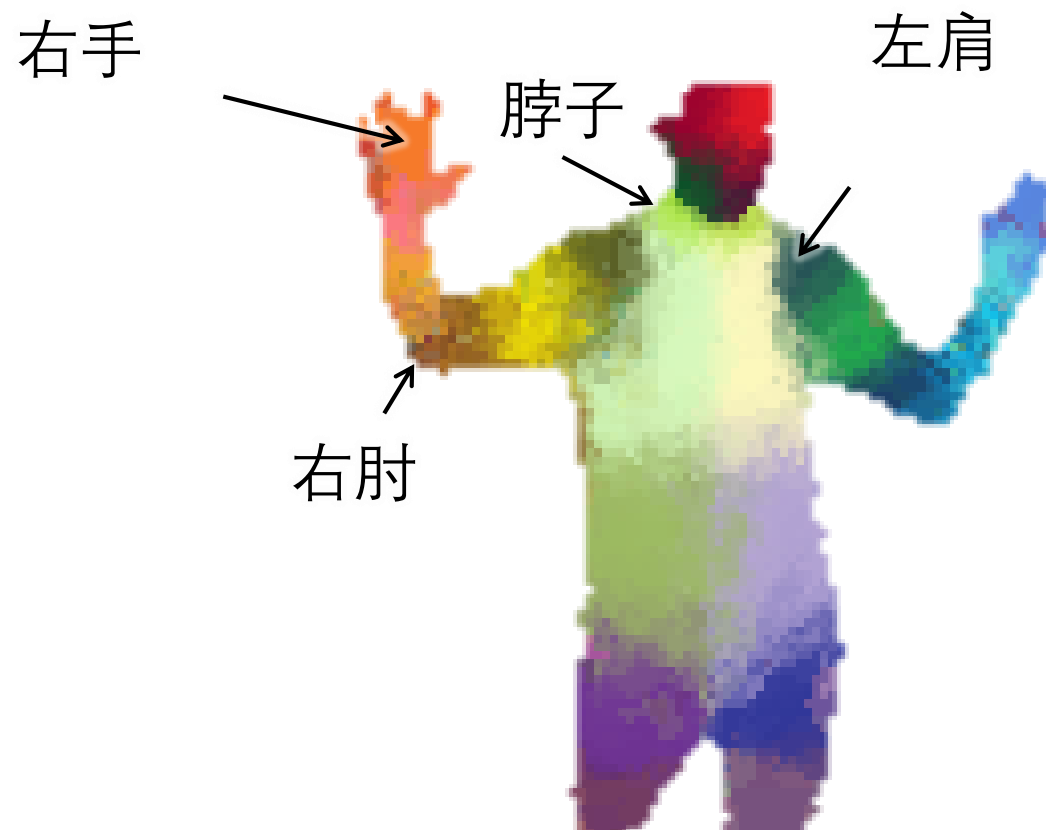


人体骨架提取

Real-Time Human Pose Recognition in Parts from Single Depth Images

深度图上逐像素的分类，基本思路：

- 对每个前景像素，估算其属于人体各个“部件”的概率（见右图）
- 通过训练分类器的方式实现上述功能
- 对于每个像素，哪些信息能够用于对他分类、求出它属于不同身体部件的概率？



人体骨架提取

Real-Time Human Pose Recognition in Parts from Single Depth Images

对于每个像素，哪些信息能够用于对他分类？

- 利用当前像素和周围特定位置像素的深度差分用线索
- 计算“点对”的深度差特征： $f(I, \mathbf{x}; \mathbf{v})$

$$f(I, \mathbf{x}; \mathbf{v}) = d_I(\mathbf{x}) - d_I(\mathbf{x} + \Delta)$$

深度图 I 中像素 \mathbf{x} 和它附近 Δ 位置像素深度差

待分类像素深度 $d_I(\mathbf{x})$

特定邻近点像素深度 $d_I(\mathbf{x} + \Delta)$

像素坐标 \mathbf{x}

邻近像素相对方向 Δ

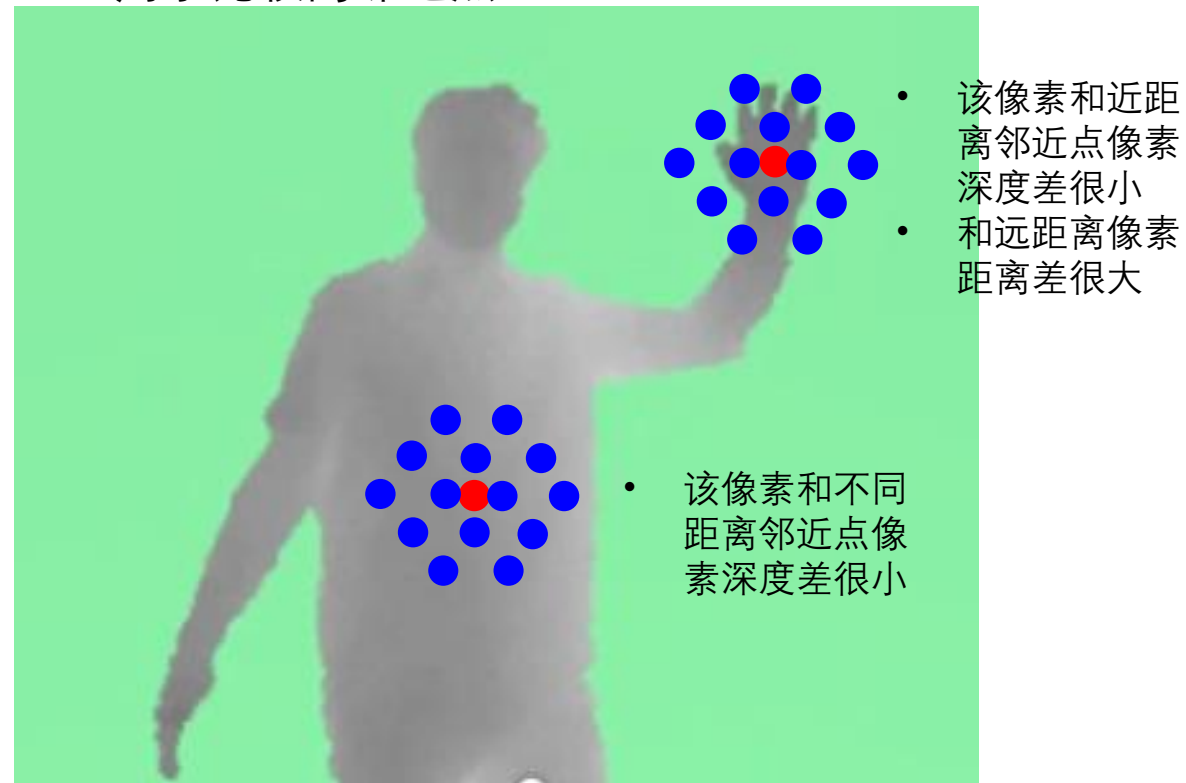
$$\Delta = \frac{\mathbf{v}}{d_I(\mathbf{x})}$$

位移距离反比于深度，用于补偿近大远小现象

注意：1) \mathbf{v} 是2元素向量，分别代表移动方向的x/y分量

2) 这里的 \mathbf{x} 和 \mathbf{v} 是指深度图上的位置和移动（不是3D空间）

以下面红色像素点分类为例，蓝色点是用于比较的邻近点



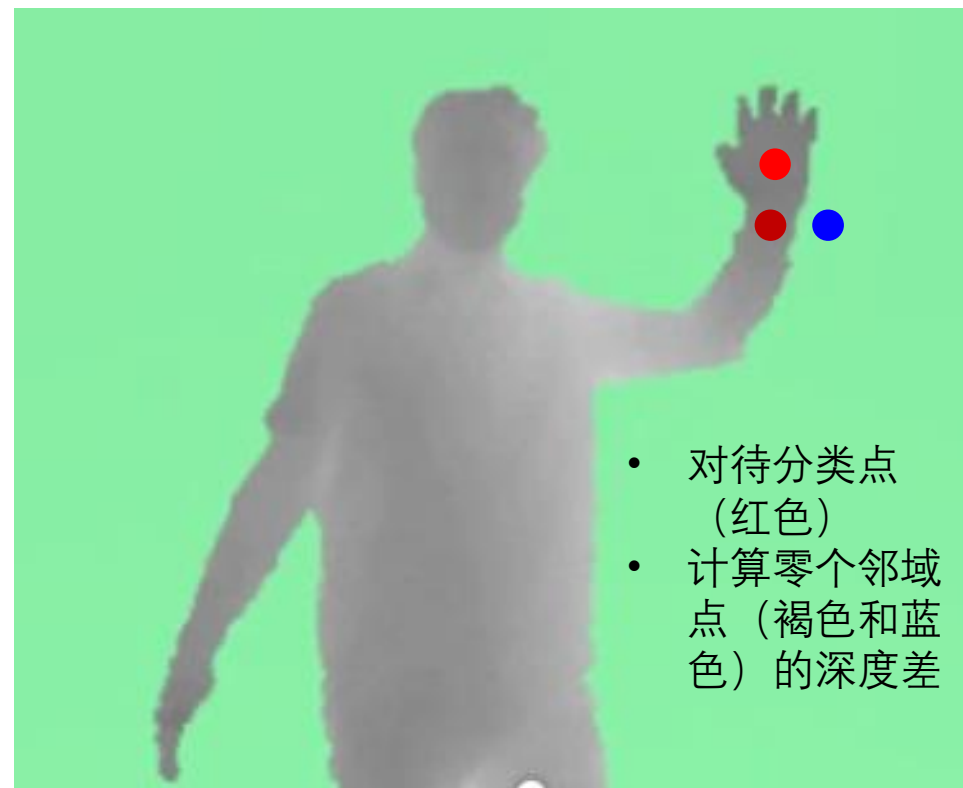
人体骨架提取

Real-Time Human Pose Recognition in Parts from Single Depth Images

- 计算“点对”深度差方法的拓展
- 计算邻域两点之间的深度差（而不是计算待分类像素和邻域的深度差）

$$d_I \left(\mathbf{x} + \frac{\mathbf{u}}{d_I(\mathbf{x})} \right) - d_I \left(\mathbf{x} + \frac{\mathbf{v}}{d_I(\mathbf{x})} \right)$$

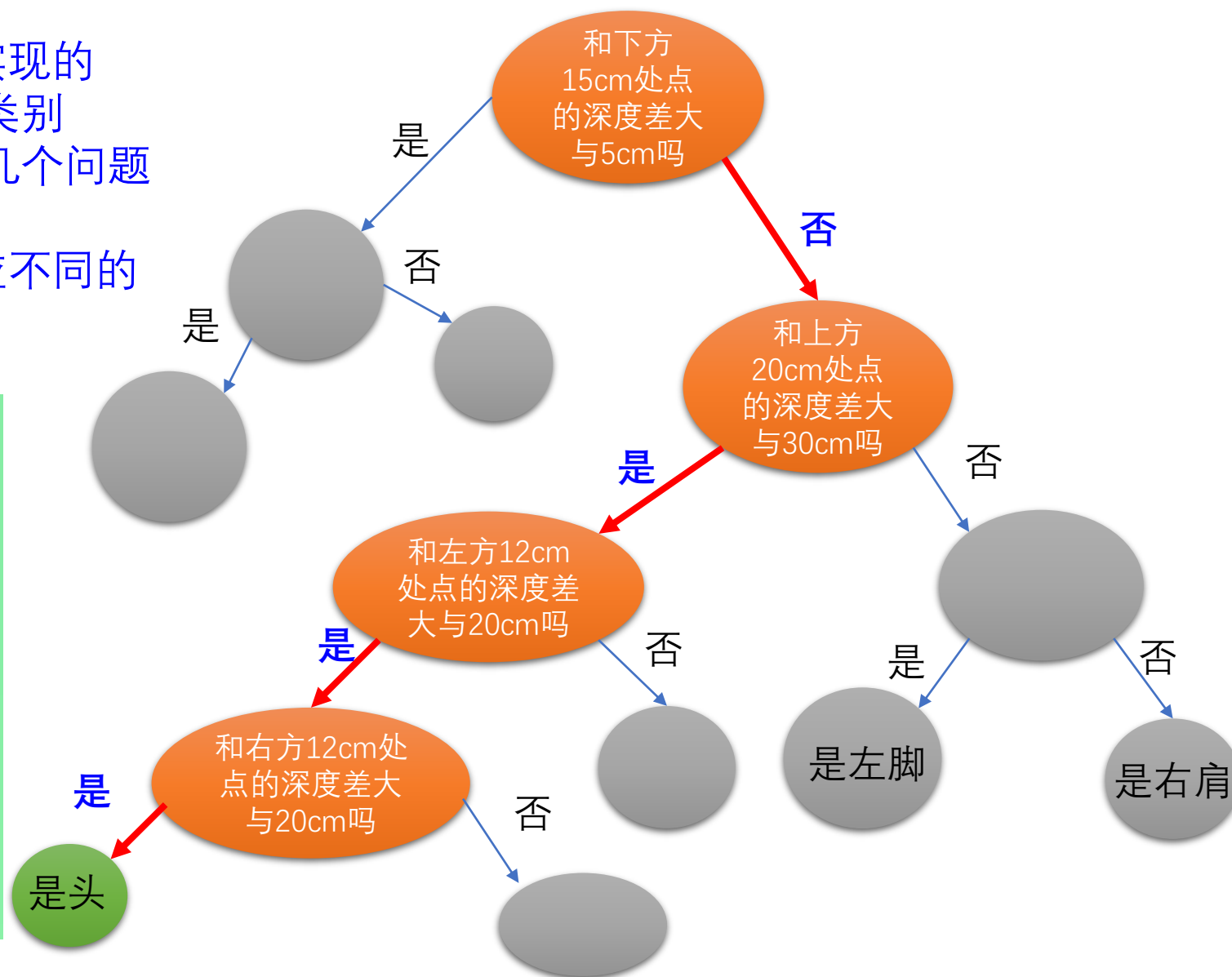
点x分别沿着u和v方向移动（指深度图上移动）



- 对待分类点（红色）
- 计算两个邻域点（褐色和蓝色）的深度差

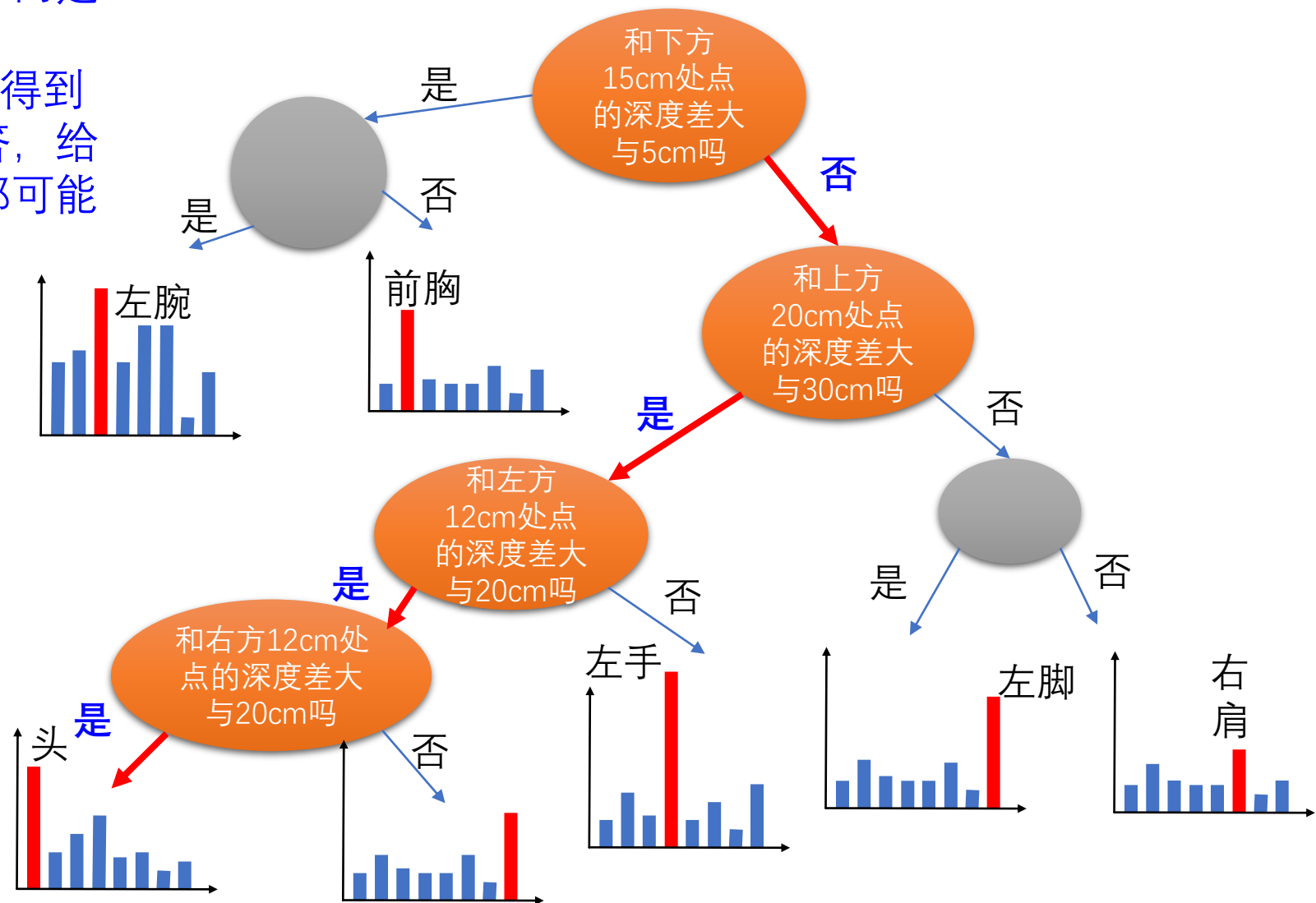
人体骨架提取——基于决策树的像素分类

- 分类过程是通过构建一颗“提问树”实现的
- 用过有限的问题知道待分类像素的类别
- 比如下图红色像素，通过右图所示几个问题来确认他是头部
- 不同的回答导向不同的“叶子”，对应不同的识别结果



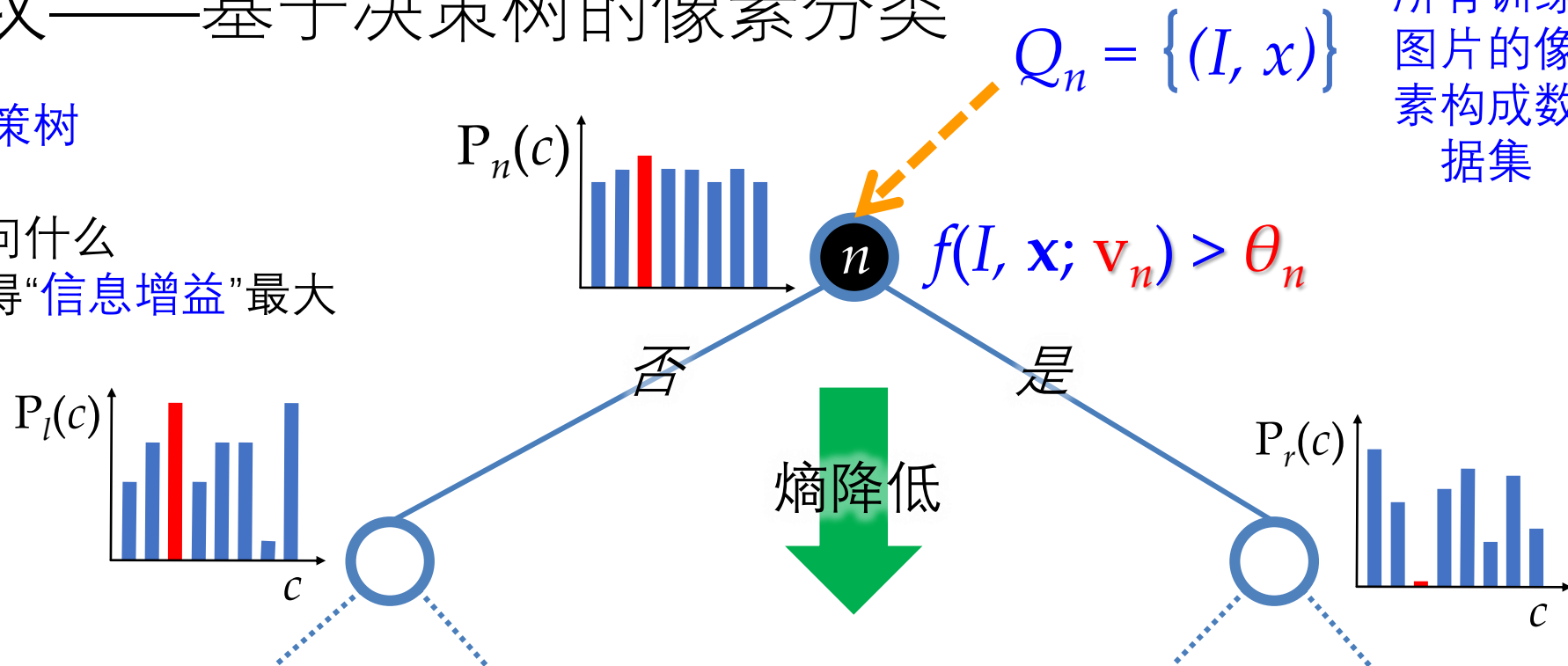
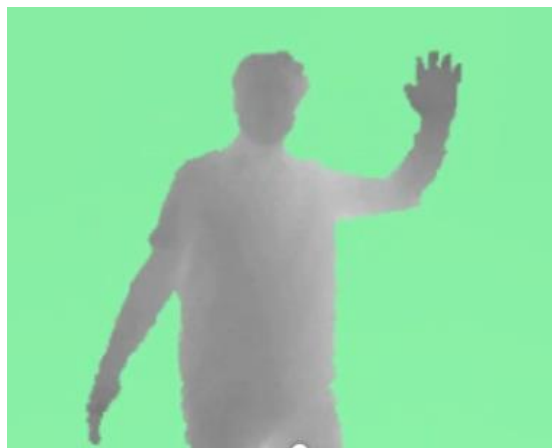
人体骨架提取——基于决策树的像素分类

- 改进：叶节点不对像素进行硬分类，而是给出“类型概率”
- 叶节点的“类型概率”可以从训练数据得到
- 看右边例子，对于一系列问题的回答，给出不同身体部位的可能性，其中头部可能最大，但不排除其他可能



人体骨架提取——基于决策树的像素分类

- 前一页的问答树——决策树
- 如何设计决策树？
 - 确定每个节点应该问什么
 - 选择问题的内容使得“信息增益”最大



- 所选的问题 (\mathbf{v}, θ) 将上一层节点的(训练)数据集 Q_n 分为两部分 Q_l 和 Q_r
- 好的问题使得 Q_l 和 Q_r 中不同身体部件的样本差异尽可能大——我们可以用熵衡量
- $E(Q)$ 是数据集 Q 的离散熵的估计算法 $= -\sum_{i=1}^n P_i * \log(P_i)$

选择 (\mathbf{v}, θ) 最大化信息增益

$$\Delta E = -\frac{|Q_l|}{|Q_n|} E(Q_l) - \frac{|Q_r|}{|Q_n|} E(Q_r)$$

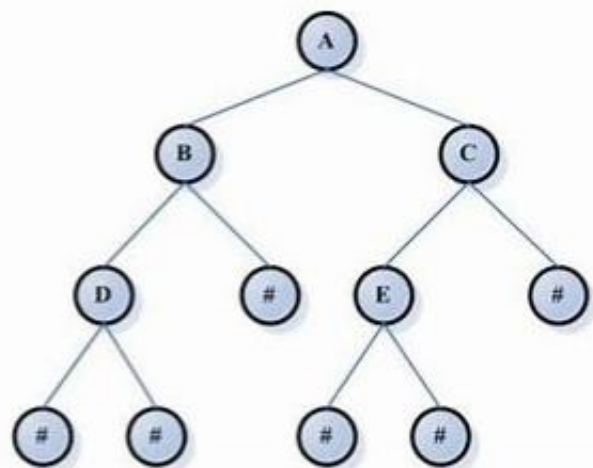
i 是训练集 Q 中, 属于身体部件 i 的数据相对 Q 的总数据量比例

所有训练
图片的像
素构成数
据集

人体骨架提取——决策树的深度

下面考虑决策树的设计的另一个问题——深度

- 问的问题层数越多，分类越准确
- 树越深越好？



输入深度图



真实答案



决策树分类结果

(叶节点
最高概率的
身体部
件类型)



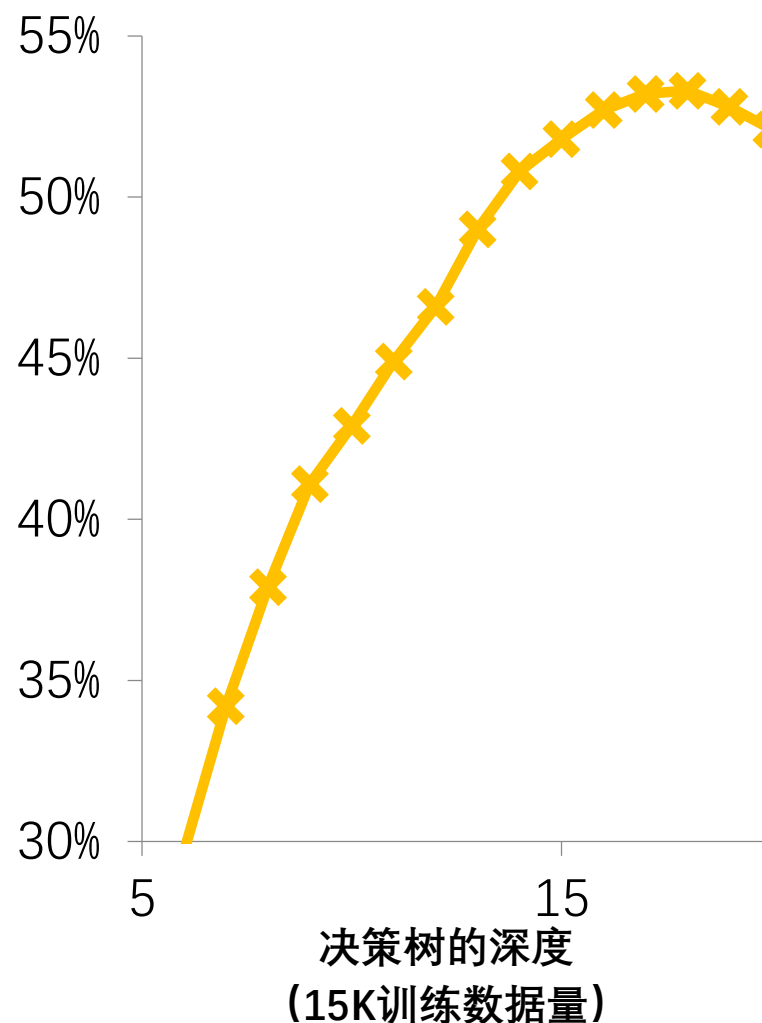
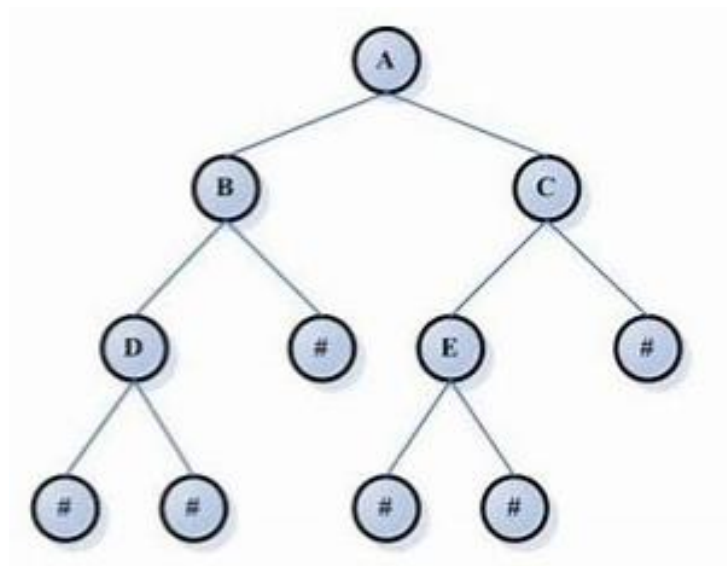
决策树深度 18



随着深度
增加，分
类（用色
彩标注）
约精细，
越正确

人体骨架提取——决策树的深度

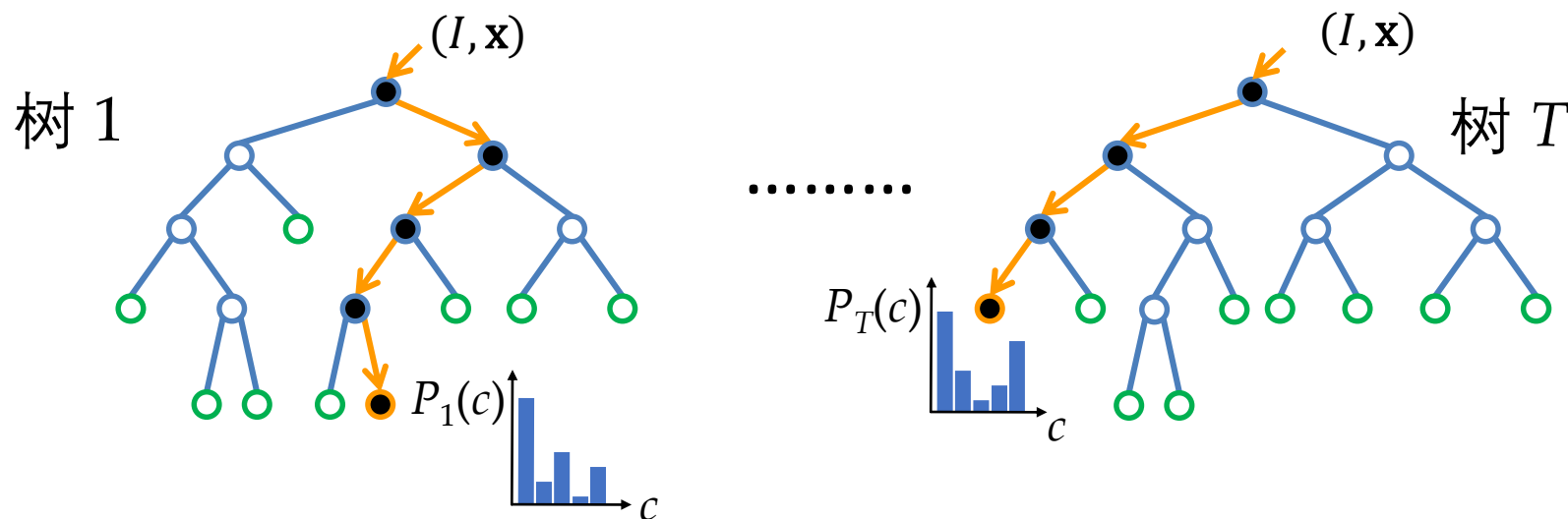
决策树的深度不能无约束地增加



- 训练数据分类正确率随着树的深度上升而提高
- 验证数据当树的深度高于特定值之后，反而下降

人体骨架提取——随机森林

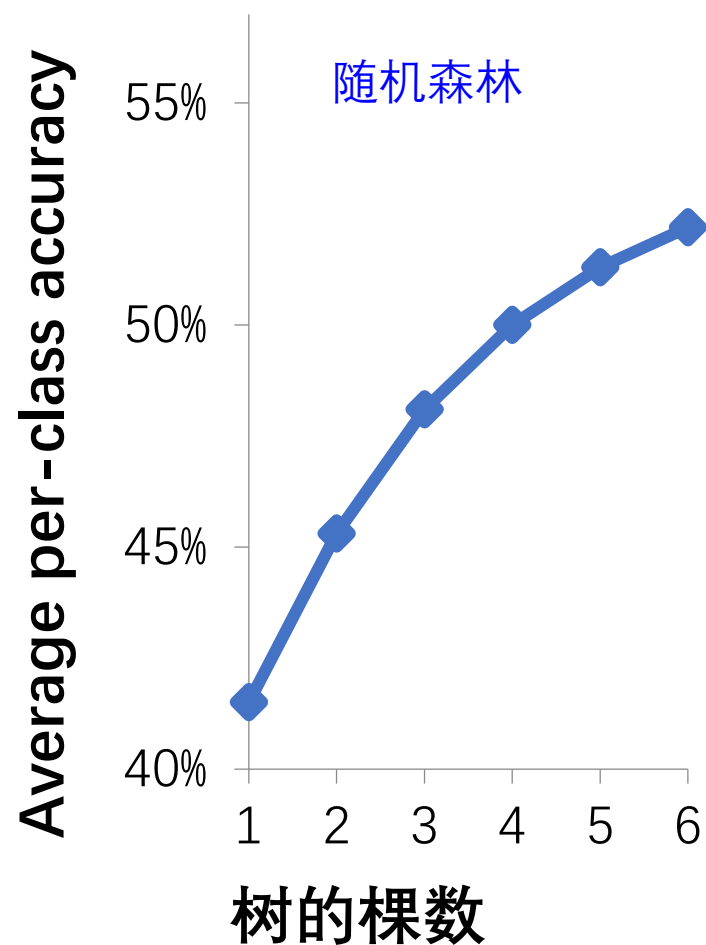
- 如何提升正确率，避免单棵决策树的“过拟合”？
- 建立多棵决策树——**随机森林**
- 每棵决策树需要在构建时确保他们的“独立性”（如何实现？如下）
 - 每棵树训练使用的样本从一个总的训练样本集合中随机抽取
 - 建树过程中，每个节点允许使用的特征集合（这里是问题集）是随机选取的一个特征子集，从这个子集中找出节点分裂的最优问题（使得信息增益最大的问题）



- 最后输出的分类结果(像素归属不同身体部件的概率)是所有决策树分类结果的平均

$$P(c|I, \mathbf{x}) = \frac{1}{T} \sum_{t=1}^T P_t(c|I, \mathbf{x})$$

人体骨架提取——随机森林



真值



不同数量随机森林的分类效果

1 棵树

3 棵树

6 棵树



(不同颜色表示
叶节点最高平
均概率对应的
身体部件类型)

人体骨架提取——从像素分类得到关节定位

再次检查一下分类效果

- 逐个像素的分类正确率能最高到60%左右
- 看上去不很高
- 反映在分类结果上呈现“类别噪声”
- 如何去除“类别噪声”？
- 这个问题会同关节定位同同时解决



人体骨架提取——从像素分类得到关节定位

exp $\left(-\left\|\frac{\hat{\mathbf{x}} - \hat{\mathbf{x}}_i}{b_c}\right\|^2\right)$

分一下几步实现关节定位

每个像素的类型“扩散到空间”

用Meanshift, 找到从空间“漫布”的类型(身体部件)的中心点

根据身体部件的中心点位置, 加上骨架比例约束, 找到关节点

对外扩散到的空间位置

深度图像素 x_i 对应空间坐标

$$f_c(\hat{\mathbf{x}}) \propto \sum_{i=1}^N w_{ic} \exp\left(-\left\|\frac{\hat{\mathbf{x}} - \hat{\mathbf{x}}_i}{b_c}\right\|^2\right)$$

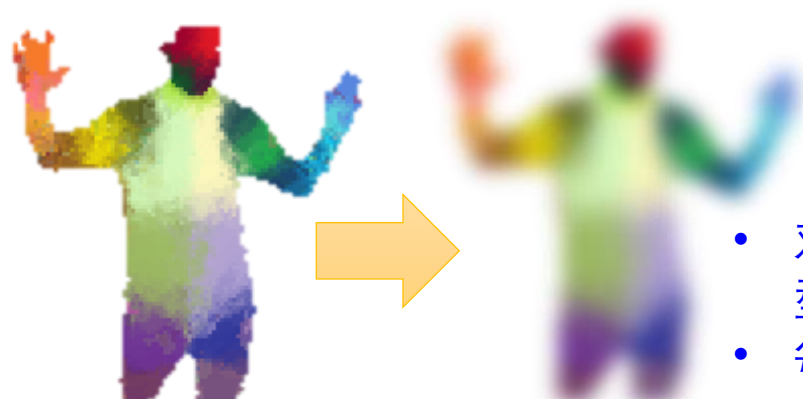
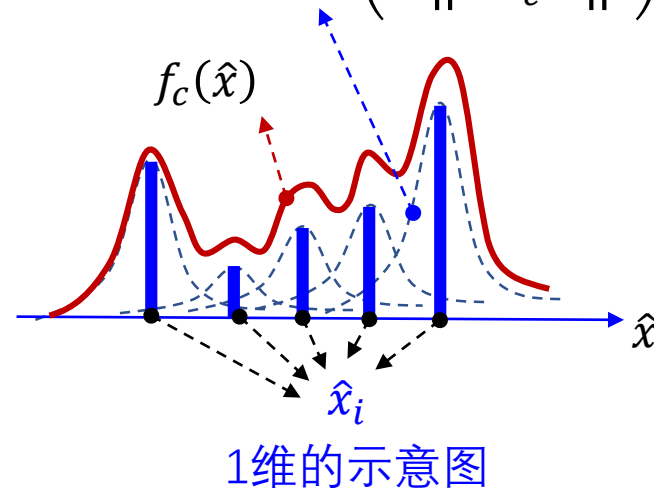
- 像素 x_i 对类别“扩散”的贡献
- 和像素 x_i 归属类别 c 的概率以及他的尺寸(面积)有关的权重

训练获得的常数, 对应扩散程度

$$w_{ic} = P(c|I, \mathbf{x}_i) \cdot d_I(\mathbf{x}_i)^2$$

像素 x_i 归属类别 c 的概率

深度平方和像素对应空间曲面面积成正比



- 对于每个类型 c , 空间任何一点 $\hat{\mathbf{x}}$ 对应的类型“权重”为 $f_c(\hat{\mathbf{x}})$
- 每个身体部件, 对应空间“一团彩色烟雾”
- 身体部件通过计算“烟雾”浓度最高的区域的到 (下一页meanshift算法)

人体骨架提取——从像素分类得到关节定位

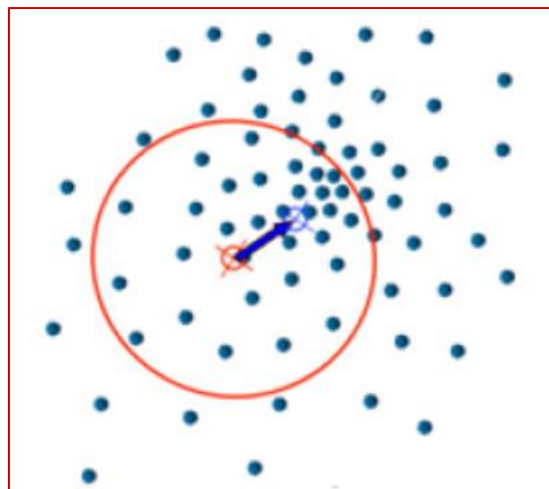
分一下几步实现关节定位

每个像素的类型“扩散到空间”

用Meanshift, 找到从空间“漫布”的类型(身体部件)的中心点

根据身体部件的中心点位置, 加上骨架比例约束, 找到关节点

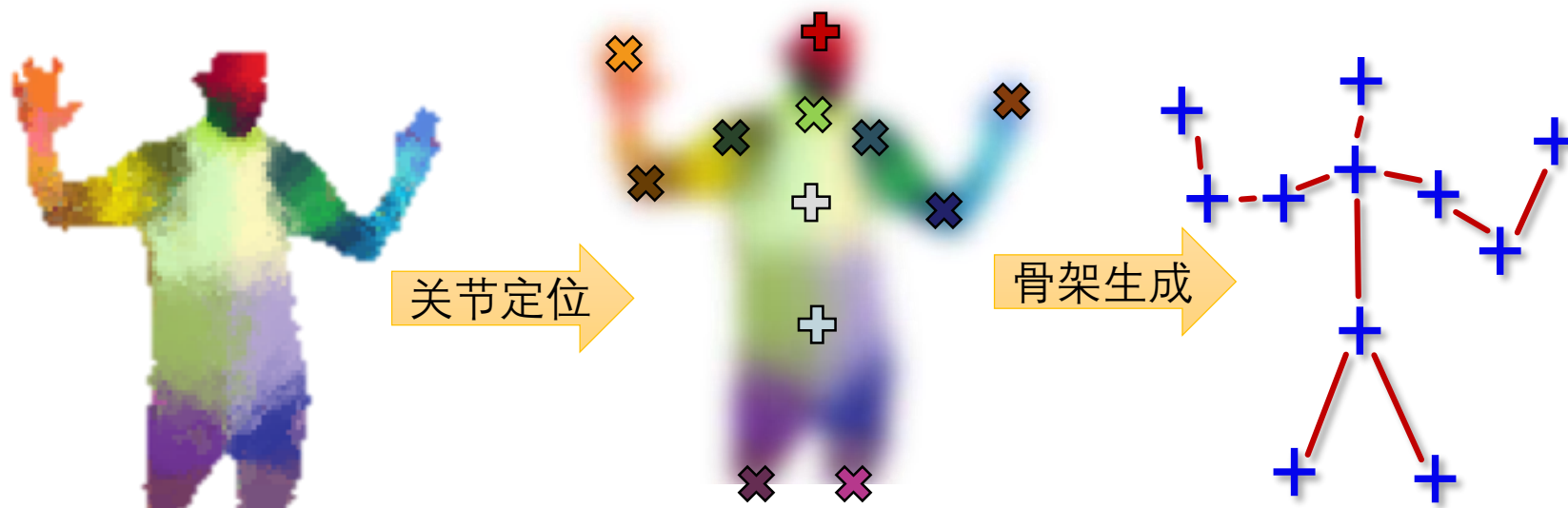
Meanshift算法示意图



- 设图中的点对应类型c的身体部件
- 我们希望估算出身体部件c的中心

1. 初选一个中心, 以它为球心, 画球
2. 计算球内点的重心
3. 移动球心到上一步计算得到的重心
4. 重复步骤2/3直到球心不再移动

- 我们使用的算法和上述meanshift还有少量差别——重心是从“漫布”在球内的类型c的权重 $f_c(\hat{x})$ 计算得到的



人体骨架提取

Real-Time Human Pose Recognition in Parts from Single Depth Images

获得骨架的主要流程



内容

- 人体肢体识别
- 物体识别与点云粗配准

物体识别与点云粗配准



为何这几个主题放在一起?

- 识别物体的不少算法同时也可以看成是物体匹配算法
 - 他们的识别过程看成是待识别物体旋转、平移然后测试它和“样本”的匹配程度（注意：也有不按这一流程的方法）
 - 因此很大一类识别算法被归类为配准算法（使用关键词：registration、retrieval等）
- 方法的大致分类
 - 基于形状模板
 - 之前手势识别可以看成是最简单的一种实现
 - 基于ICP
 - 匹配成功时，同时得到精确位姿参数
 - 运算量大，传统ICP算法往往陷入局部极小（由于噪声、点对匹配错误等）
 - 基于特征点匹配
 - 通过点云中，少数几个关键点位置及其局部特征用于匹配和配准
 - 匹配运算量小，只用到部分点云信息，匹配成功得到的位姿参数精度ICP低（一般匹配成功后再进行一轮ICP）
 - 基于概率或（广义的）Hough变换
 - 基于“统计”投票获取位姿
- 有大量的算法研究成果，但我们在有限篇幅只介绍其中的几个算法

点云识别、粗配准和查找

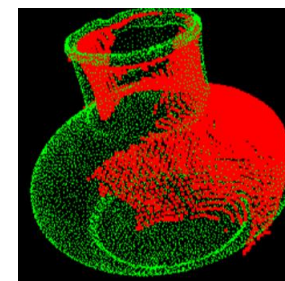
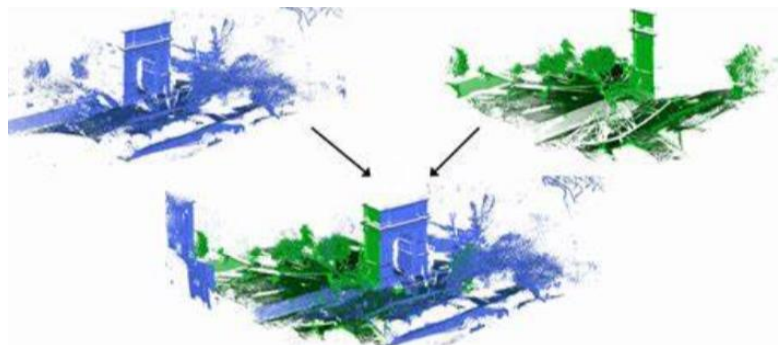
(Point Cloud Registration / Retrievals)

应用场景

- 物体识别
- 地点识别
- 点云精细融合需要的粗配准

☹️ • 和之前ICP什么差别

ICP用于精细点云配准，当点云对应较差时，难以收敛






内容

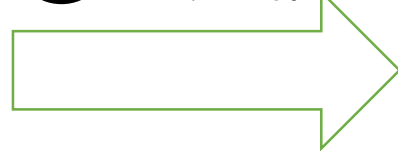
- 人体肢体识别
- 物体识别与点云配准
 - 基于关键点特征的点云识别/粗配准流程 (ISS特征方法)
 - SHOT点云特征提取算法
 - PFH点云特征提取算法
 - PPF点云识别配准算法

点云识别、粗配准和查找——基于关键点的流程

- 局部特征

- 点云局部的形状参数
-  • 如何实现配准识别?
-  • 通过不同位置的局部特征, 加上几何约束实现配准识别

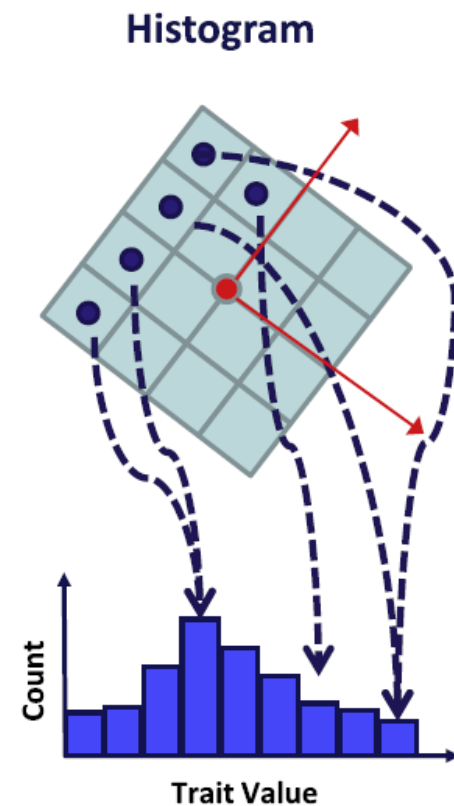
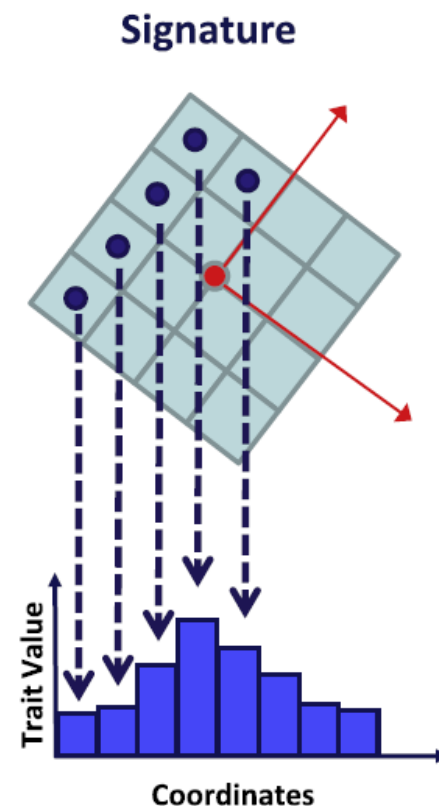
 为何这样能够工作?



- 看一个局部特征识别的例子
 - 从下面的描述能够知道这是什么物体?
 - 从一个物体中找到了4个角点
 - 任意两个角点的距离相同
 - 任意三个角点连接的三角形中心是平面
- 很大可能就是4面体

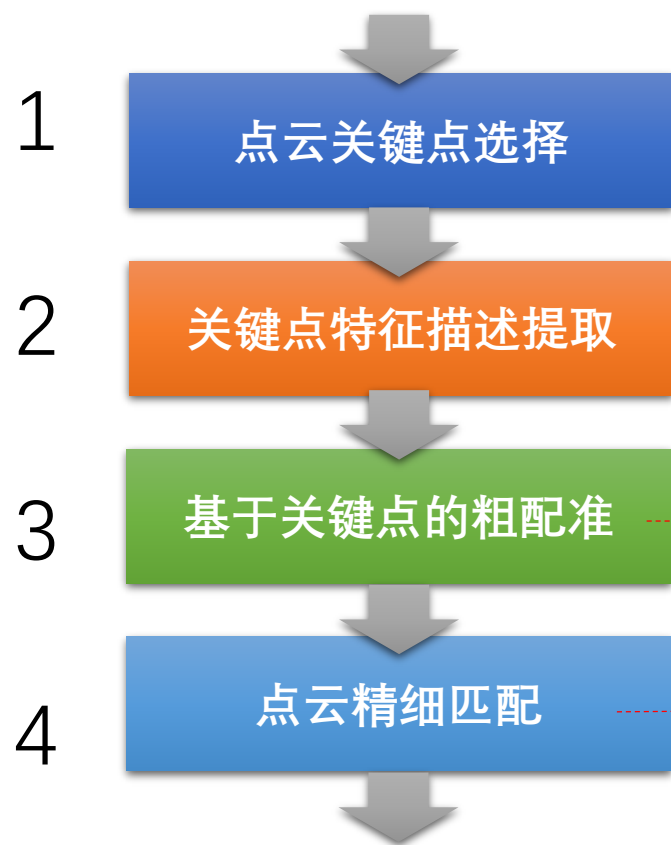
匹配使用的特征——局部特征

- 两种局部特征的表述方案
 - 基于（有序排列的）几何参数
 - 基于几何参数的统计量——更加稳定，降低噪声影响，点云是无序的，分布带有一定随机性的点云



点云识别和粗配准

基本流程pipeline



- 对于点云识别，可以只执行到第3步
- 对于点云融合，需要执行到第4步
- 后面我们会介绍1/2/3步，第4步可以参考之前章节PPT内容

- 会展开有更细的步骤
- 需要使用RANSAC尝试多次
- 粗配准可能失败
 - 比如目标识别应用中，识别待识别的点云和候选点云不一致

- 比如用ICP算法精确计算旋转平移矩阵

点云识别和粗配准——点云关键点选择

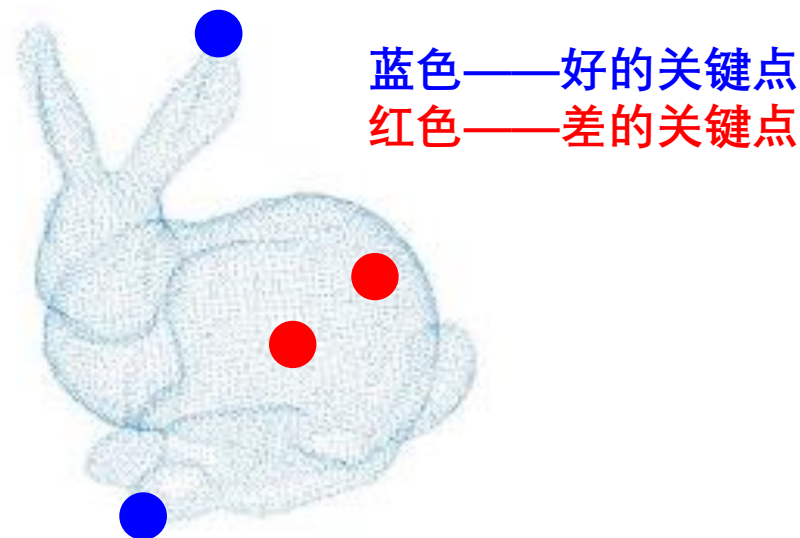
基本流程



选为关键点的点云位置有什么要求？

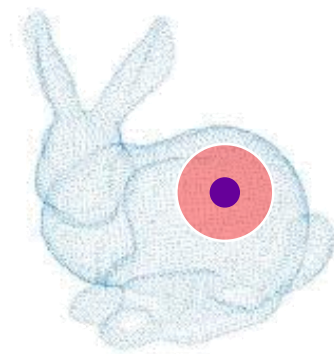
- 独特——不平凡的，在点云中少见的
- 明显——不容易被点云噪声、畸变所埋没

看下面例子



关键点选择——具体算法1（来自ISS算法流程的关键点筛选部分）

- 基于局部点云“特征值”（PCA）分析的方法识别
- 下面介绍的方法来自ISS性状描述算法
 - 和之前的要求相比，我们还需要从关键点唯一的识别出方向
 - 方向信息用于建立局部参考坐标系，计算局部特征



1. 对待选点 \mathbf{p} ，对半径 r_{frame} 的球内点 $\{\mathbf{p}_n\}$ 计算“加权协方差矩阵”

$$\mathbf{M}(\mathbf{p}) = \frac{\sum_{|\mathbf{p}_n - \mathbf{p}| < r_{frame}} w_n (\mathbf{p}_n - \mathbf{p})(\mathbf{p}_n - \mathbf{p})^T}{\sum_{|\mathbf{p}_n - \mathbf{p}| < r_{frame}} w_n}$$

加权系数 w_n 和点云的“局部密度”有关

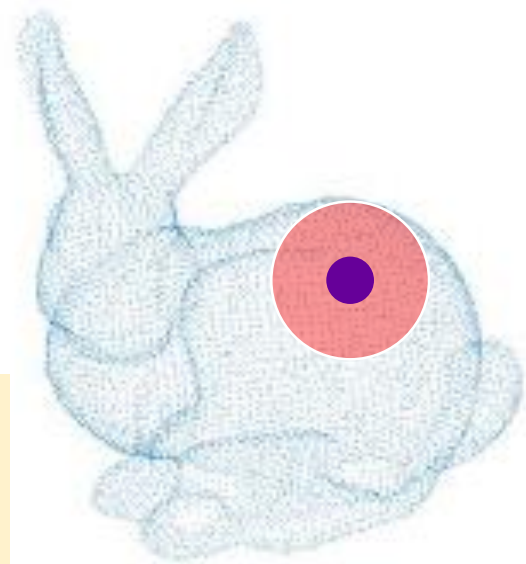
$$w_n = \frac{1}{\|\{\mathbf{p}_m : |\mathbf{p}_m - \mathbf{p}_n| < r_{density}\}\|}$$

表示围绕点 p ，半径 $r_{density}$ 的球内点的数量的倒数

备注：

- 这里 \mathbf{p} 和 \mathbf{p}_n 是3元素列向量，代表点的坐标
- 计算 $\mathbf{M}(\mathbf{p})$ 的公式和之前计算点云局部法向量类似，但没有扣除局部点云的均值。当 \mathbf{p} 和 r_{frame} 球内点云中心接近时， $\mathbf{M}(\mathbf{p})$ 和计算云局部法向量时使用的协方差矩阵是很接近的
- $\mathbf{M}(\mathbf{p})$ 的特征值分解能够代表点云形状特性（参照上一条）
- 这里的 w_n 用于补偿不同位置点云密度差异带来的影响

关键点选择——具体算法2



2. 计算“协方差阵” $\mathbf{M}(\mathbf{p})$ 的特征值分解，特征值和特征向量分别记作： $\{\lambda_1, \lambda_2, \lambda_3\}$ 和 $\{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$ ，其中特征值按次序从大到小排列，即 $\lambda_1 \geq \lambda_2 \geq \lambda_3$

3. 将满足下面条件的点选作“关键点”：

- 特征值看成椭球体的三轴半径
- 这一条件表明关键点邻域的点大致看成是一个三轴长度明显有差别的椭球体，并且该椭球体最短轴足够长

$$\begin{cases} \frac{\lambda_2}{\lambda_1} < \theta_1 \\ \frac{\lambda_3}{\lambda_2} < \theta_2 \\ \lambda_3 > \theta_3 \end{cases}$$

特征值有明显大小差别
(用于得到对噪声不敏感的“稳定”的参考坐标系)

滤除平坦区域的点，
以“角点”作为特征点

备注：这里的 θ_1 、 θ_2 、 θ_3 是需要用户调参的内容

上述要求是为了从中得到的坐标轴（见下一页）比较稳定（robust）

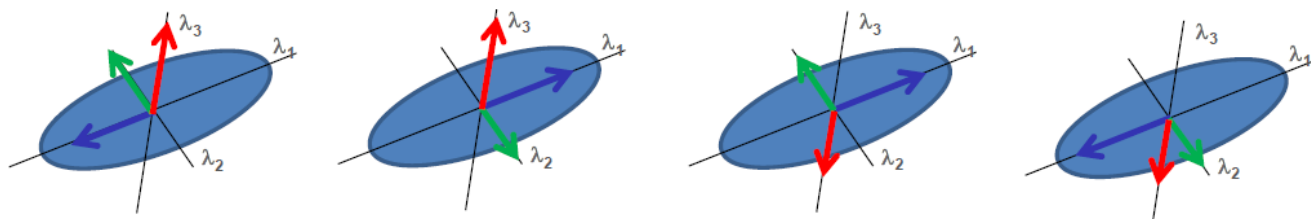
关键点选择——具体算法3

构建参考坐标系——用于特征数据获取

4. 如果 \mathbf{p} 点被选作“关键点”，则以“协方差阵” $\mathbf{M}(\mathbf{p})$ 的三个特征向量分别作为局部参考坐标系的 x, y, z 轴， \mathbf{p} 为原点
 (x 对应最大特征值、 z 对应最小特征值)

备注：

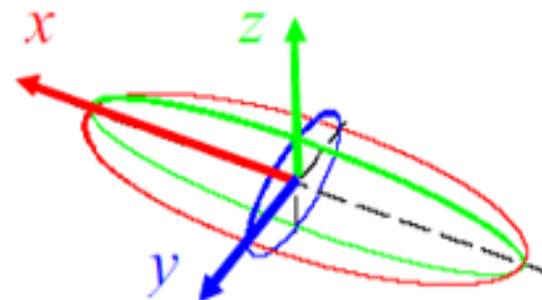
- 论文 (ISS) 中将局部参考坐标系记作Reference Frame: RF
- 这里有方向“模糊度”问题，有下面4中坐标轴的方向选择方案
- 后面会介绍一个去除模糊度的方案，但存在各种其他的方案



- 参考前面内容：“协方差阵” $\mathbf{M}(\mathbf{p})$ 的三个特征向量分别作为局部参考坐标系的 x, y, z 轴， \mathbf{p} 为原点

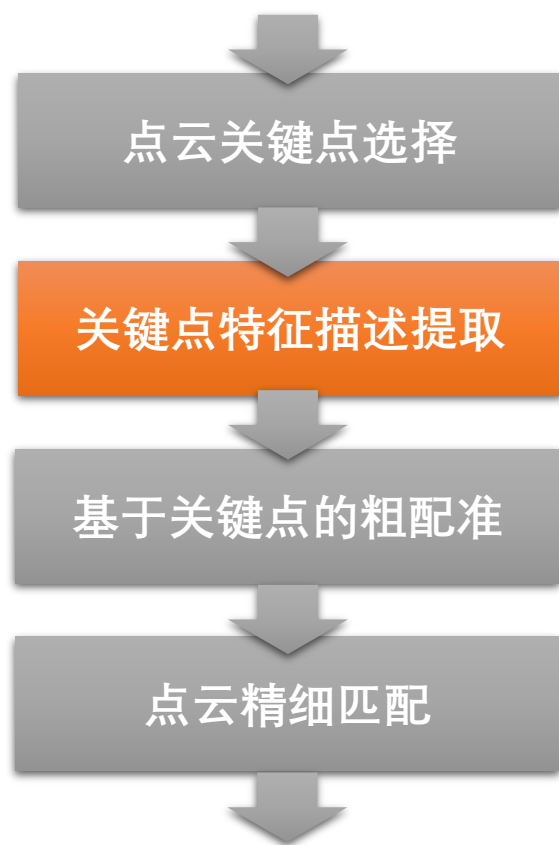
$$\text{选择特征点要求} \begin{cases} \frac{\lambda_2}{\lambda_1} < \theta_1 \\ \frac{\lambda_3}{\lambda_2} < \theta_2 \\ \lambda_3 > \theta_3 \end{cases}$$

使得坐标系的三个轴方向明确，对噪声不敏感



点云识别和粗配准——关键点特征提取

基本流程

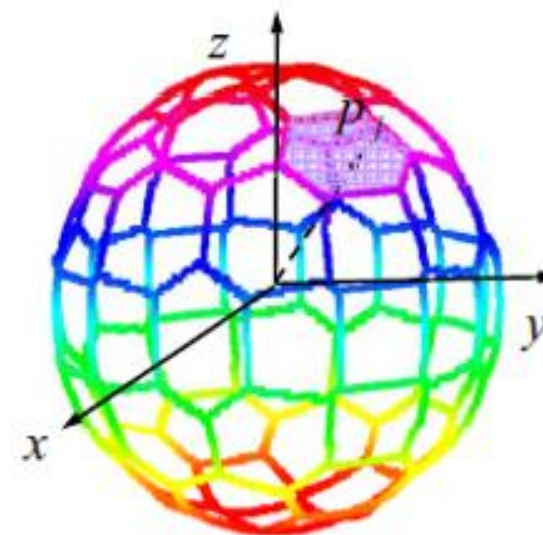


- ☹️ • 前面在计算并筛选出关键点时不就已经计算过协方差矩阵及其特征值分解了?
- 😊 • 仅仅3个特征值不足以描述点云的多样性, 因此需要进一步提取描述特征参数

点云识别和粗配准——ISS特征数据提取

计算步骤如下：

1. 构建局部坐标系
 2. 以关键点为坐标原点，构建球体
 3. 球体切分若干块
 4. 统计每块球体内的点云数量，计算时考虑点云疏密进行补偿
- 所有分块的统计数据(有个原始的向量)作为关键点的特征描述

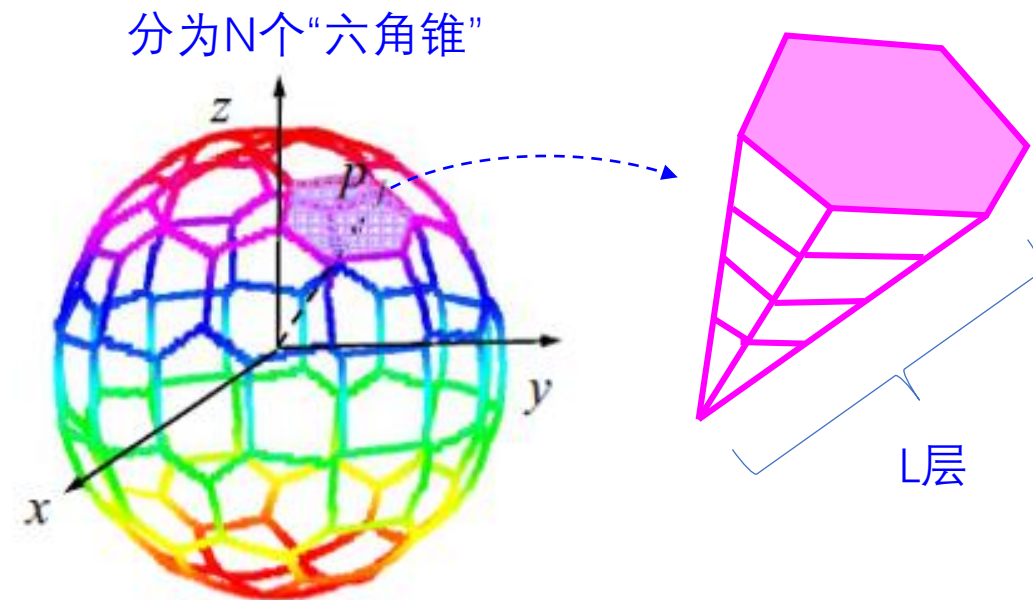


点云识别和粗配准——ISS特征数据提取

计算步骤如下：

1. 构建局部坐标系
 2. 以关键点为坐标原点，构建球体
 3. 球体切分若干块
 4. 统计每块球体内的点云数量，计算时考虑点云疏密进行补偿
- 所有分块的统计数据(有个原始的向量)作为关键点的特征描述

分块细节



- 按空间角度将球体划分成尺寸一致的 N 个“六角锥”
- 对每个六角锥根据到球心距离等分成 L 层
- 将空间分为 $K=1+(L-1)N$ 块（为何不是 LN 块？见下面）
- 最接近球心的 N 个“六角锥”的“尖顶”合并为一个块

统计这 K 个分块内的点云数据，
得到 K 维的特征描述向量

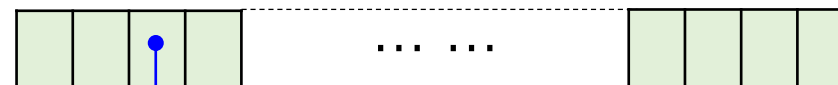


点云识别和粗配准——ISS特征数据提取

计算步骤如下：

1. 构建局部坐标系
 2. 以关键点为坐标原点，构建球体
 3. 球体切分若干块
 4. 统计每块球体内的点云数量，计算时考虑点云疏密进行补偿
- 所有分块的统计数据(有个原始的向量)作为关键点的特征描述

统计这K个空间分块内的点云数据，得到K维的特征描述向量



- 填入： $\sum_n w_n$ 😐 为何不直接填点云数目？
- n 是落入对应分块的点云中点的序号
- $w_n = \frac{1}{\|p_m: |p_m - p_n| < r_{density}\|}$ ，是点 p_k 邻域(半径 $r_{density}$ 球内)点云数量的倒数
- w_n 用于补偿点云密度带来的统计量波动 😊
- 每个元素的内容大致看成它对应空间块中点云占据的体积

注意：中文“特征向量”这个词有两个不同的含义，理解时需要注意区分

1. Eigen-vector——矩阵分解得到的特征向量
2. Feature vector——描述对象特性的参数构成的向量

点云识别和粗配准——基于关键点特征的点云匹配

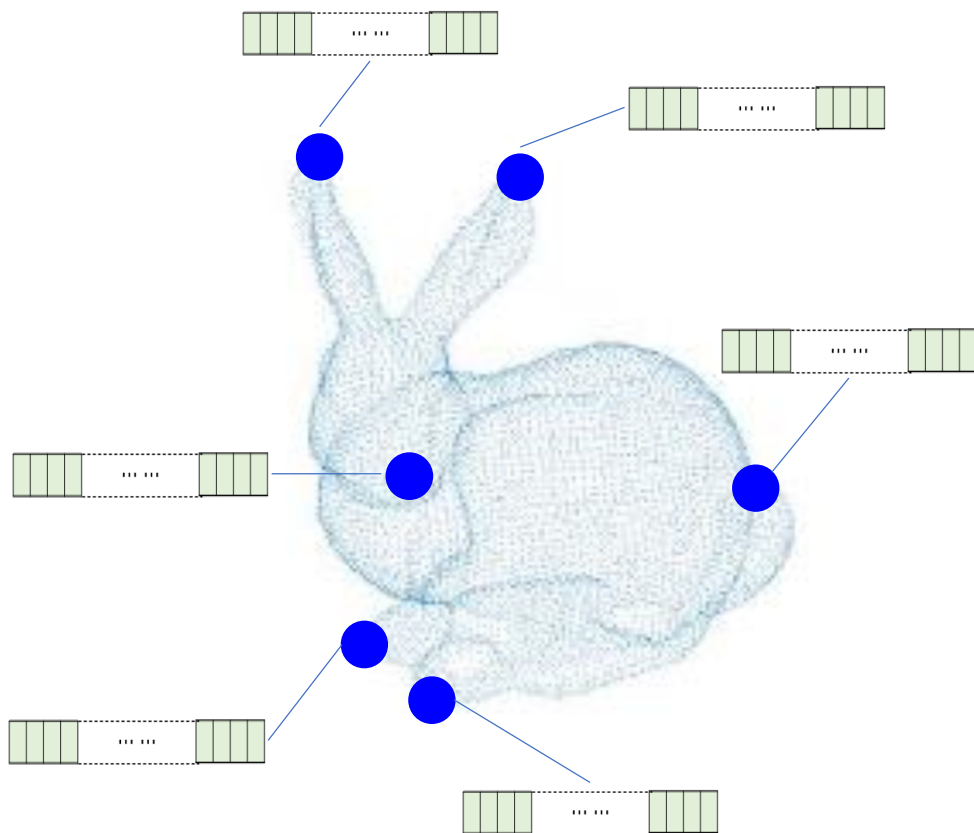
通过以上步骤，得到关键点，以及他的特征描述
每个特征点通过两个信息描述：

- 坐标： $\mathbf{p} := [x \ y \ z]^T$
- 特征描述向量： $\mathbf{v} := [v_1, v_2, \dots, v_K]^T$

- 点云可以用一系列特征点构成的集合来描述其特点

$$\{(\mathbf{p}_1, \mathbf{v}_1), (\mathbf{p}_2, \mathbf{v}_2), \dots, (\mathbf{p}_N, \mathbf{v}_N)\}$$

- 特征点数据量远小于点云数据总量
- 仅仅利用特征点数据能够快速实现点云的识别和粗配准

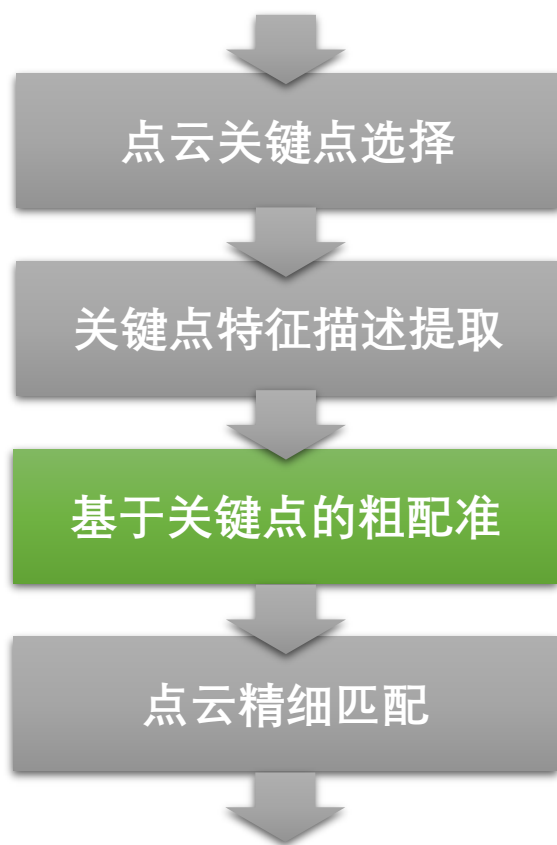


注意：中文“特征向量”这个词有两个不同的含义，理解时需要注意区分

1. Eigen-vector——矩阵分解得到的特征向量
2. Feature vector——描述对象特性的参数构成的向量

点云识别和粗配准——关键点特征提取

基本流程



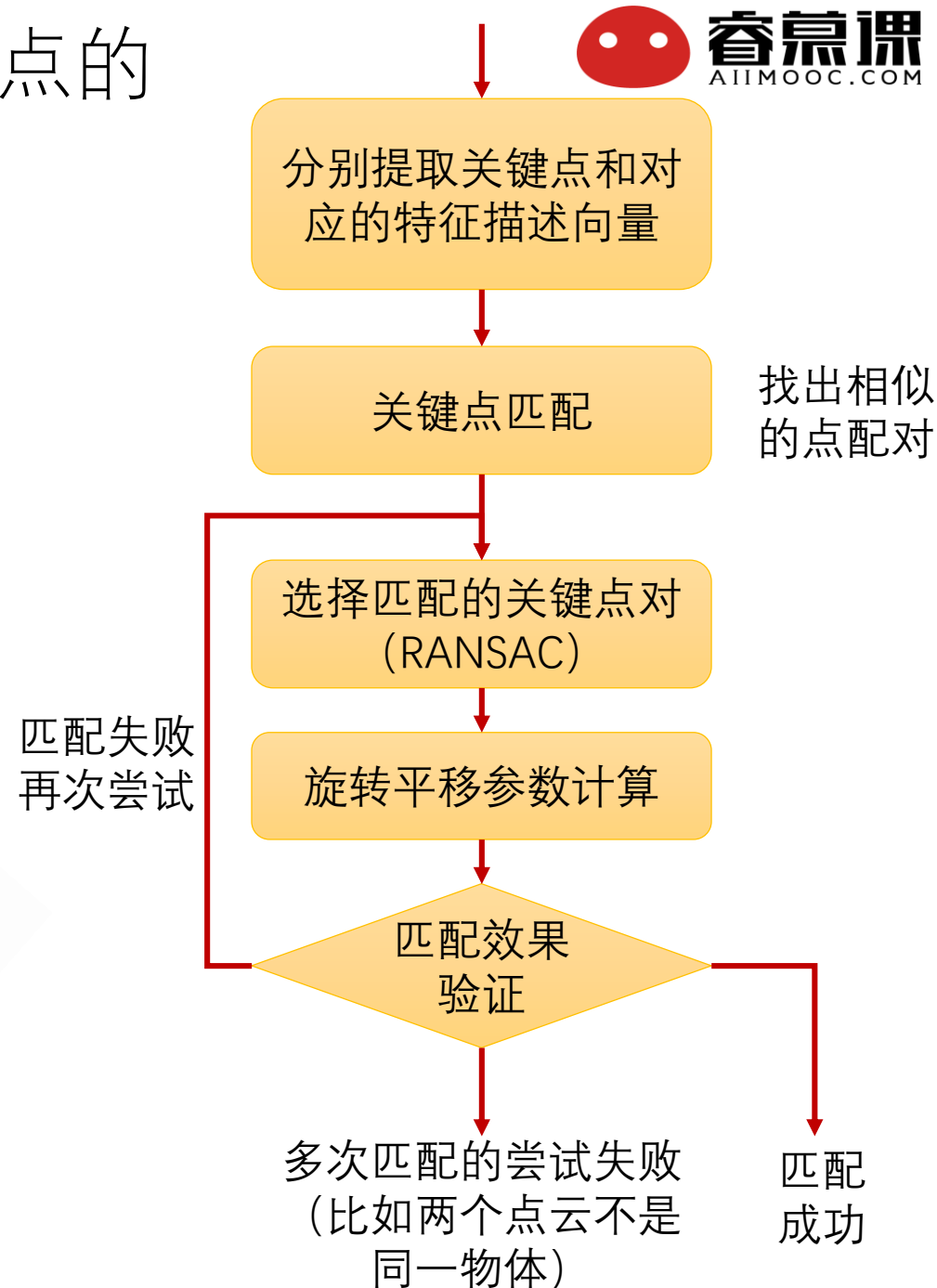
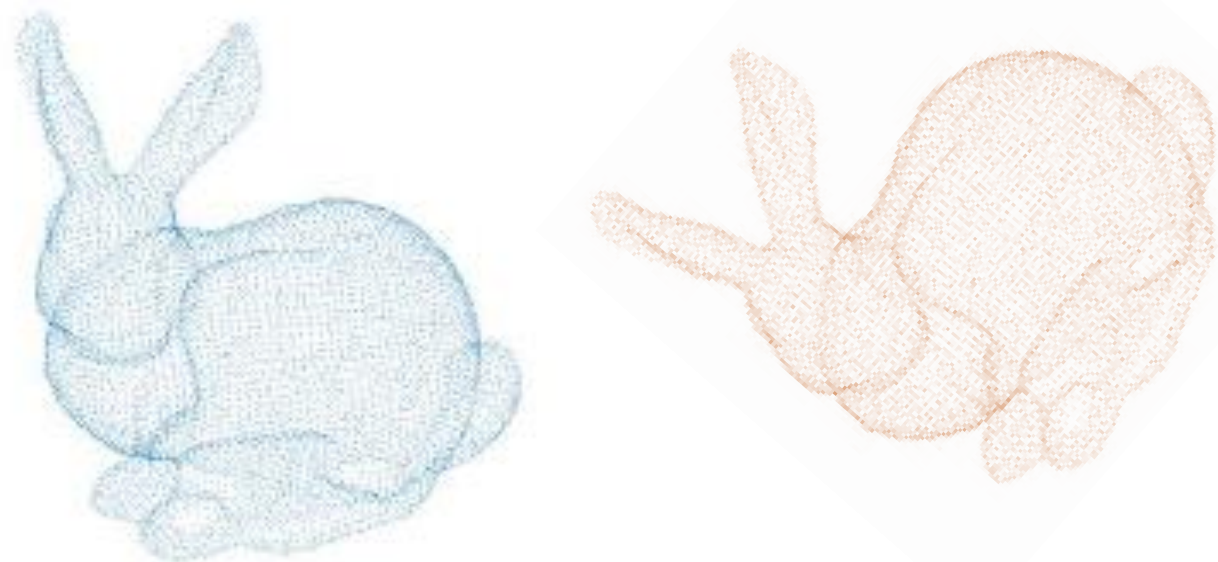
点云识别和粗配准——基于关键点的粗配准算法流程

任务：

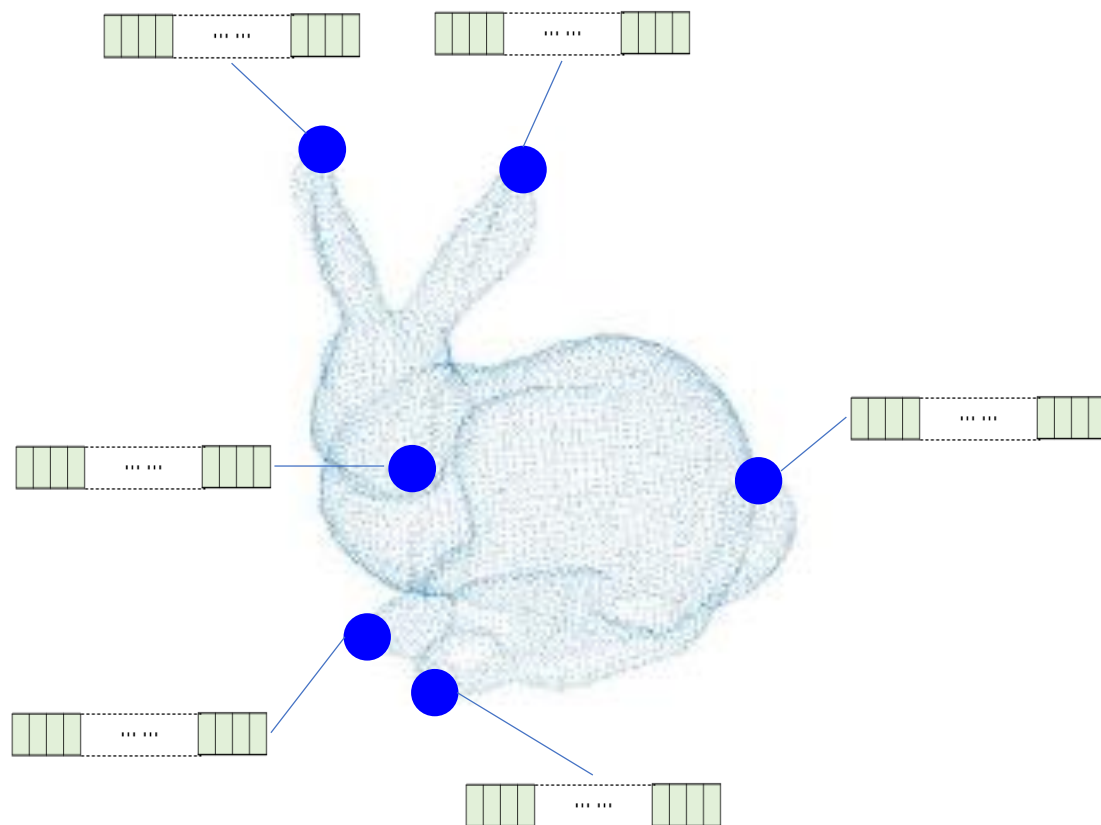
1. 确定两个点云是否对应相同的物体
2. 如果是相同物体，他们之间的旋转平移关系是多少？

难点：

1. 两个点云尺寸可能很大，传统ICP运算量极大
2. 存在坐标噪声和部分区间点云缺失等问题

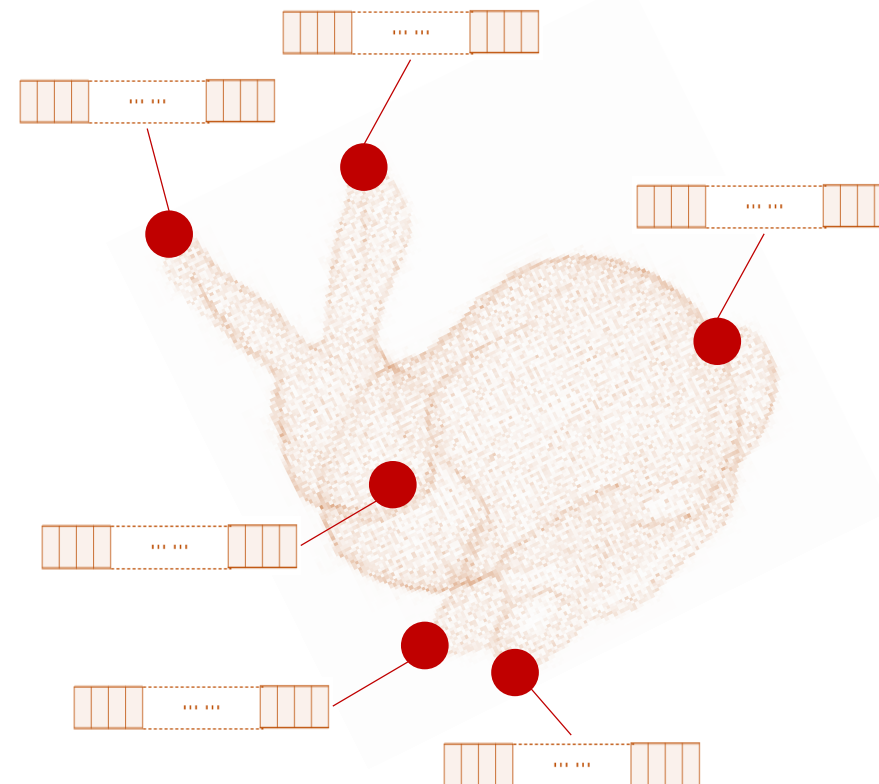


点云识别和粗配准



$$\{(\mathbf{p}_1, \mathbf{v}_1), (\mathbf{p}_2, \mathbf{v}_2), \dots, (\mathbf{p}_N, \mathbf{v}_N)\}$$

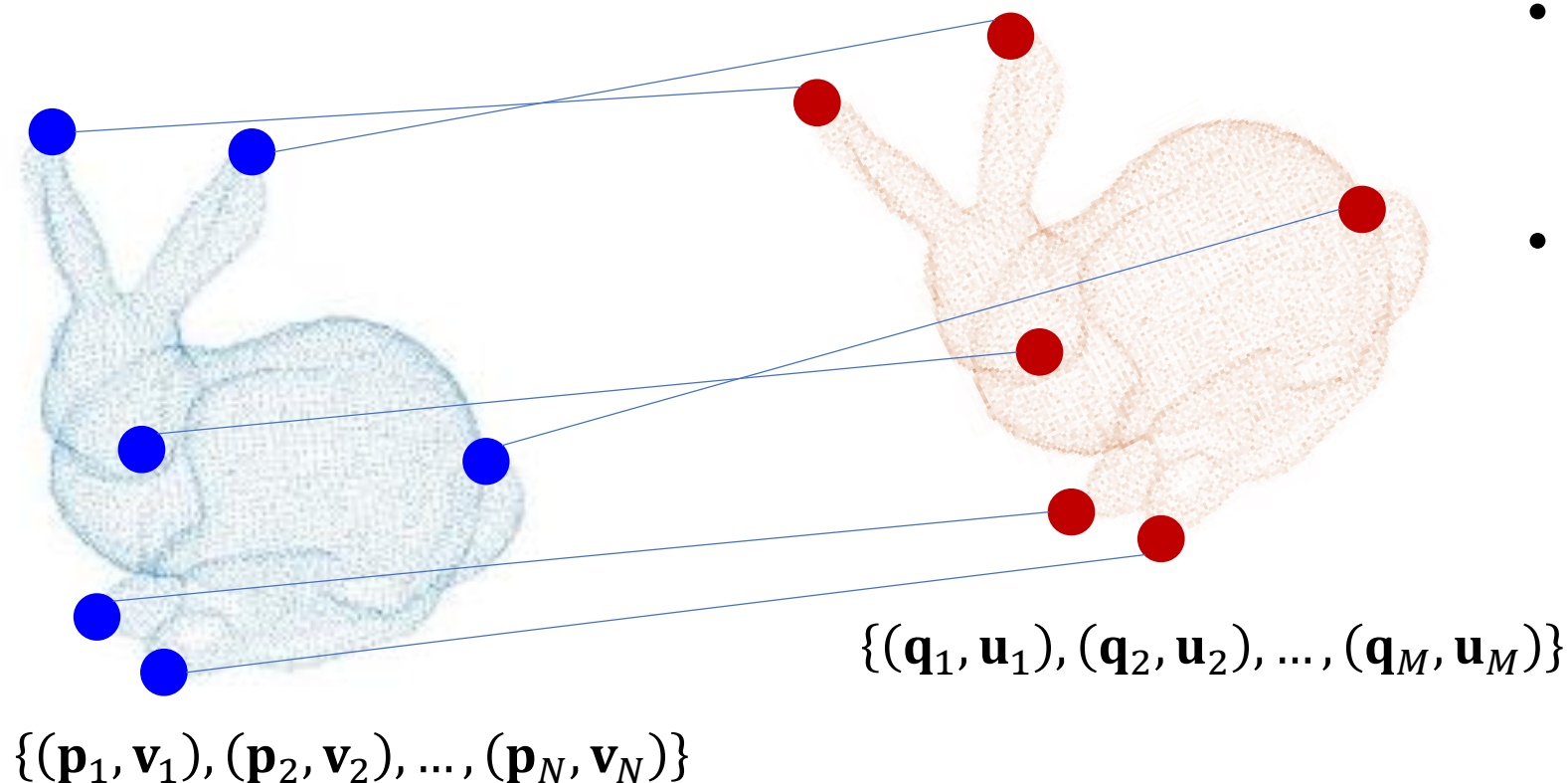
用 $(\mathbf{p}_n, \mathbf{v}_n)$ 代表左边关键点的坐标和对应特征描述向量



$$\{(\mathbf{q}_1, \mathbf{u}_1), (\mathbf{q}_2, \mathbf{u}_2), \dots, (\mathbf{q}_M, \mathbf{u}_M)\}$$

用 $(\mathbf{q}_n, \mathbf{u}_n)$ 代表右边关键点的坐标和对应特征描述向量

点云识别和粗配准——关键点匹配



- \mathbf{p}_n 和 \mathbf{q}_m 配对的准则是他们的特征描述向量 \mathbf{v}_n 和 \mathbf{u}_m 很相识
- 如何计算相似度?
 - 夹角余弦(越大越相似)

$$\frac{\langle \mathbf{v}_n, \mathbf{u}_m \rangle}{\|\mathbf{v}_n\|_2 \|\mathbf{u}_m\|_2}$$
 - 欧式距离(越小越相似)

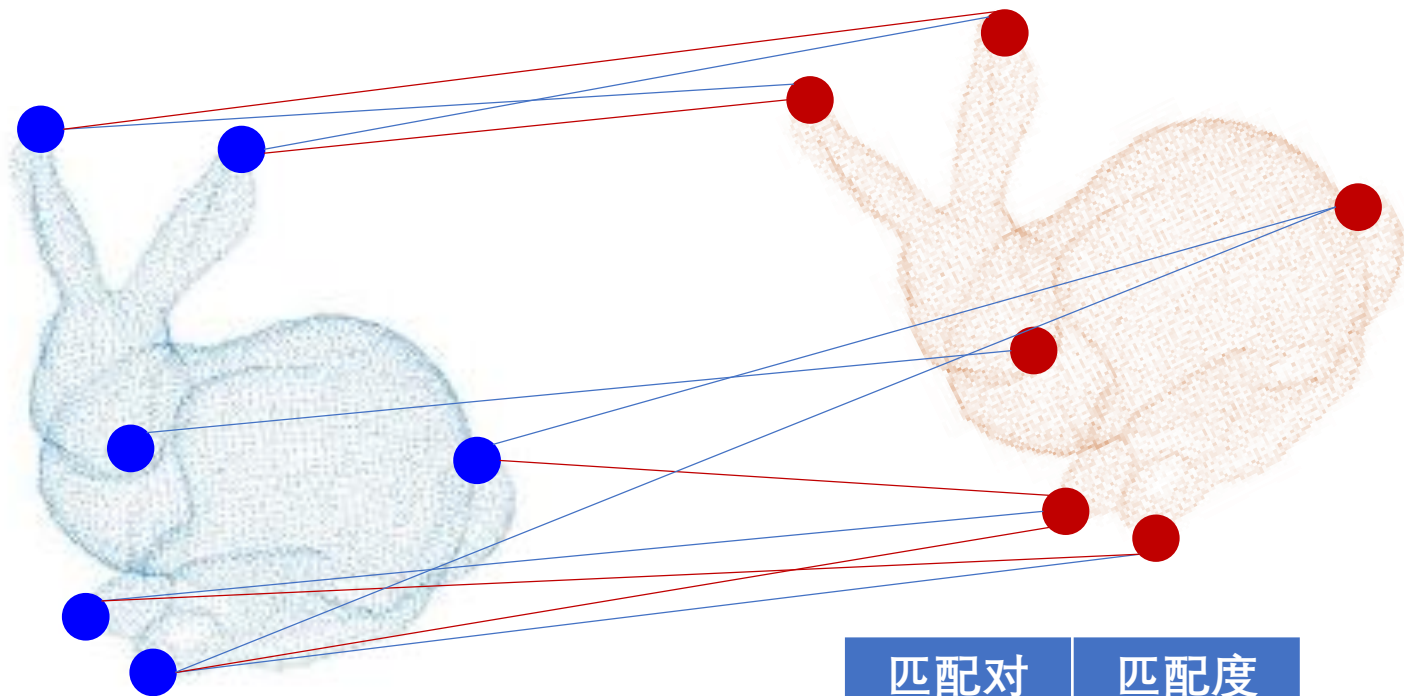
$$\|\mathbf{v}_n - \mathbf{u}_m\|_2$$
 - 其他(例子)

$$|\mathbf{v}_n - \mathbf{u}_m| \text{ 的最大元素}$$

备注:

1. 上面公式里 $\langle \mathbf{v}, \mathbf{u} \rangle$ 表示两个向量 \mathbf{v}, \mathbf{u} 逐元素相乘后求和
2. $\|\mathbf{v}\|_2$ 表示向量 \mathbf{v} 元素平方和的开根号

点云识别和粗配准——关键点匹配



匹配对	匹配度
p1,q3	0.7
p1,q4	0.8
p2,q12	0.83
p3,q4	0.77
...	...

关键点匹配表(列出相似度超过
门限的匹配关键点对)

- 通过相似度进行匹配可能有一定“模糊度” 😞
- 两个点云不同部位关键点特征描述向量的确很接近
- 人为设定的判定匹配的阈值太宽松
- 噪声、点云疏密分布差异等
- 匹配结果不再是1对1,而是多对多
- RANSAC算法
 1. 从匹配的特征对集合中随机选出一个子集
 2. 用ICP算法计算旋转平移矩阵
 3. 验证匹配结果是否能够让绝大多数特征点对重合
 4. 如果3验证不通过, 则回到步骤1

点云识别和粗配准——ICP算法(回顾)

1. 预处理——关键点平移: $\begin{cases} \mathbf{p}_n \leftarrow \mathbf{p}_n - \bar{\mathbf{p}} \\ \mathbf{q}_n \leftarrow \mathbf{q}_n - \bar{\mathbf{q}} \end{cases}$, 其中 $\begin{cases} \bar{\mathbf{p}} = \frac{1}{N} \sum_n \mathbf{p}_n \\ \bar{\mathbf{q}} = \frac{1}{N} \sum_n \mathbf{q}_n \end{cases}$ 是关键点平均坐标

2. 计算代价函数的最小化解: $\mathbf{R}^* = \underset{\mathbf{R}}{\operatorname{argmin}} \sum_n \|\mathbf{p}_n - \mathbf{R} \mathbf{q}_{\text{match}(\mathbf{p}_n)}\|^2$

☺最小二乘损失函数,

☹需要额外的约束—— \mathbf{R} 是正交阵(旋转矩阵)

☺使用SVD求解

- 计算3x3矩阵: $\mathbf{H} = \sum_n \mathbf{q}_{\text{match}(\mathbf{p}_n)} \mathbf{p}_n^T$
- 进行SVD分解: $\mathbf{H} = \mathbf{U} \mathbf{\Lambda} \mathbf{V}^T$
- 计算得到: $\mathbf{R}^* = \mathbf{V} \mathbf{U}^T$

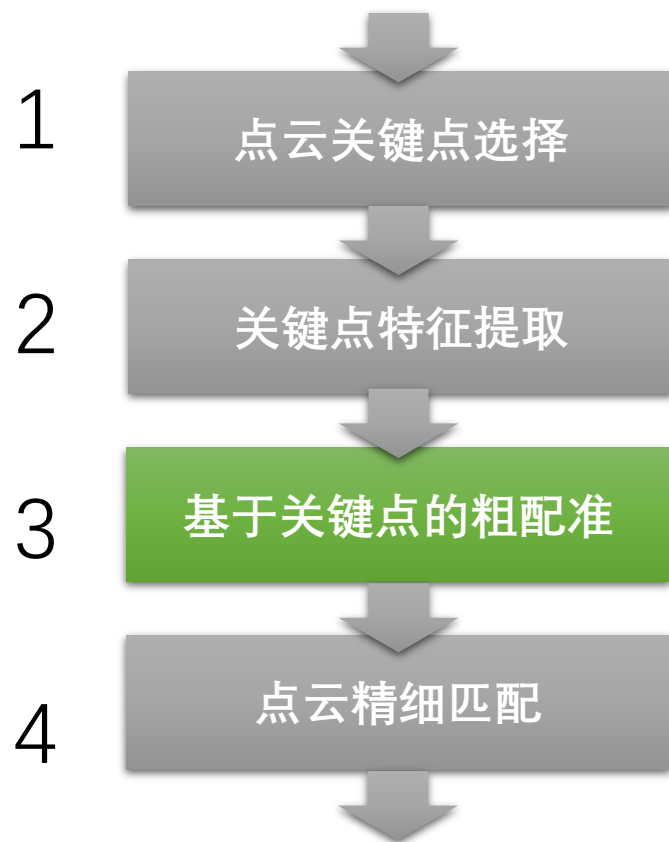
4. 修正点云位置: $\mathbf{q}_n \leftarrow \mathbf{R}^* \mathbf{q}_n$, 回到步骤2

- (注意: 一般还需检验 \mathbf{R}^* 的行列式是否=1, 确保它是旋转阵)

$\text{match}(\mathbf{p}_n)$ 是从关键点匹配表(见上一页)中抽出的下标配对关系

点云识别和粗配准

基本流程



- 获得关键点之后，有多种方式提取特征描述，包括之前的ISS特征描述算法
- 后面介绍其他几种常用的算法

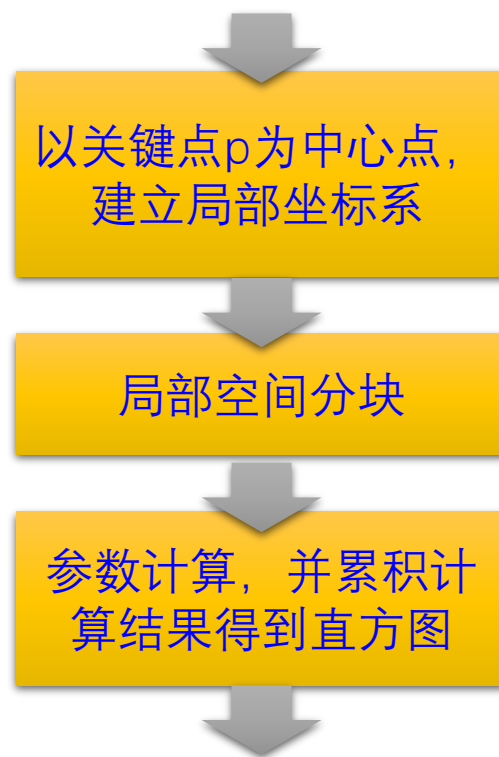
内容

- 人体肢体识别
- 物体识别与点云配准
 - 基于关键点特征的点云识别/粗配准流程（ISS特征方法）
 - SHOT点云特征提取算法
 - PFH点云特征提取算法
 - PPF点云识别配准算法

特征提取算法：

SHOT(Signature of Histograms of Orientations)

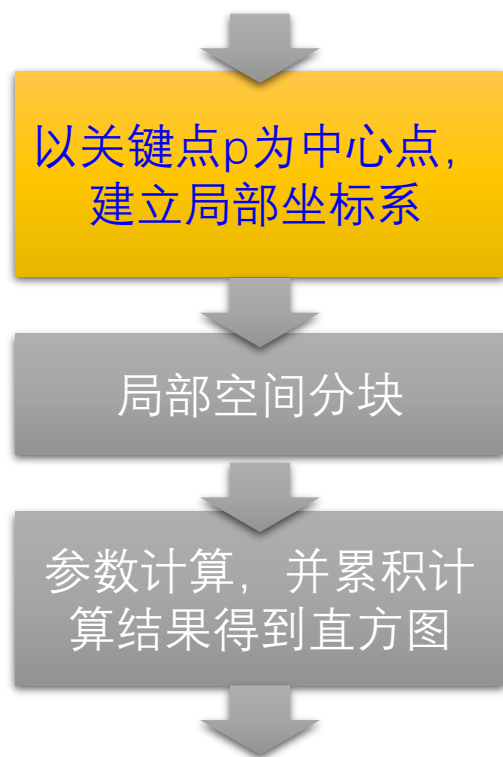
基本流程



- 和ISS方法有类似点
- 在空间分割个直方图计算方法上不同

特征提取算法：

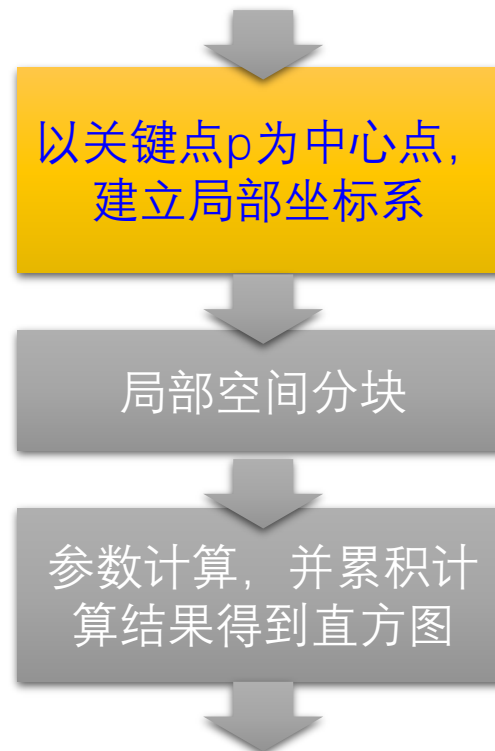
SHOT(Signature of Histograms of Orientations)



- 和ISS方法类似，围绕关键点p计算坐标协方差矩阵： \mathbf{M}
- 对上述3x3矩阵分解得到特征向量，依特征值从大到小，分别设为局部坐标系的x, y, z轴，坐标系中心在关键点p

SHOT特征提取——局部坐标系构建

- \mathbf{p} 为待分析的中心点
- 计算 \mathbf{p} 的局部法向量
 - 邻域点云对法向量的贡献按距离加权
 - 修改公式，没有通过均值计算去中心化，仅仅把 \mathbf{p} 当成中心
- X/Y/Z 通过 \mathbf{M} 的特征向量获得
- Z 轴和最小特征值对应的特征向量平行
- X 轴和最大特征值对应的特征向量平行



$$\mathbf{M} = \frac{1}{k} \sum_{i=0}^k (\mathbf{p}_i - \hat{\mathbf{p}})(\mathbf{p}_i - \hat{\mathbf{p}})^T, \quad \hat{\mathbf{p}} = \frac{1}{k} \sum_{i=1}^k \mathbf{p}_i$$

传统的法向量求取，SHOT对其进行了改造，没有使用上面的计算方式

$$\mathbf{M} = \frac{1}{\sum_{i: d_i \leq R} (R - d_i)} \sum_{i: d_i \leq R} (R - d_i) (\mathbf{p}_i - \mathbf{p})(\mathbf{p}_i - \mathbf{p})^T$$

3x3矩阵

定义邻域的球体半径

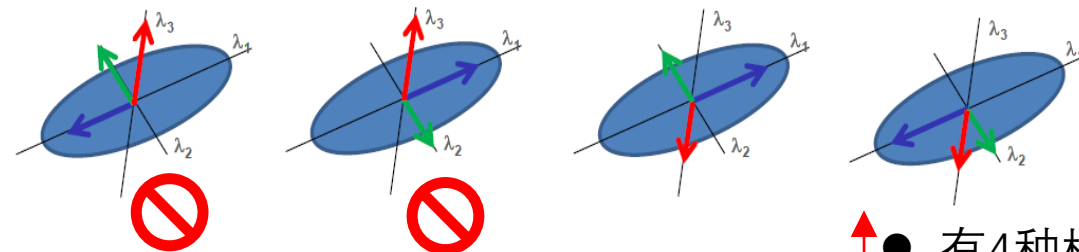
$$d_i = \|\mathbf{p}_i - \mathbf{p}\|_2$$

邻域点 中心点

权重，和ISS算法相比，这里计算权重的方式有所不同

SHOT特征提取——局部坐标系

$$\mathbf{M} = \frac{1}{\sum_{i: d_i \leq R} (R - d_i)} \sum_{i: d_i \leq R} (R - d_i) (\mathbf{p}_i - \mathbf{p})(\mathbf{p}_i - \mathbf{p})^T$$



- 得到了3个局部坐标轴之后呢？
- 旋转邻域点云，得到局部坐标系下的坐标
- 旋转矩阵是什么？老坐标系下，新坐标系3轴向量（就是M的三个特征向量）构成的矩阵的逆
- 方向模糊度如何解决——x坐标轴指向尽量和邻域内点的指向一致

$S_x^+ \doteq \{i : d_i \leq R \wedge (\mathbf{p}_i - \mathbf{p}) \cdot \mathbf{x}^+ \geq 0\}$ —— 和暂定的x轴方向一致的点集

$S_x^- \doteq \{i : d_i \leq R \wedge (\mathbf{p}_i - \mathbf{p}) \cdot \mathbf{x}^- > 0\}$ —— 和暂定的x轴方向相反的点集

$\tilde{S}_x^+ \doteq \{i : i \in M(k) \wedge (\mathbf{p}_i - \mathbf{p}) \cdot \mathbf{x}^+ \geq 0\}$ —— 缩小了的领域内，和暂定的x轴方向一致的点集

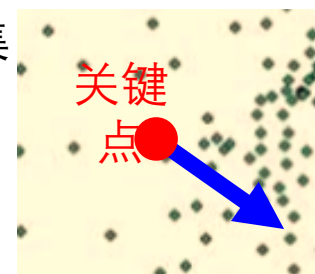
$\tilde{S}_x^- \doteq \{i : i \in M(k) \wedge (\mathbf{p}_i - \mathbf{p}) \cdot \mathbf{x}^- > 0\}$ —— 缩小了的领域内，和暂定的x轴方向相反的点集

$$\mathbf{x} = \begin{cases} \mathbf{x}^+, & |S_x^+| > |S_x^-| \\ \mathbf{x}^-, & |S_x^+| < |S_x^-| \\ \mathbf{x}^+, & |S_x^+| = |S_x^-| \wedge |\tilde{S}_x^+| > |\tilde{S}_x^-| \\ \mathbf{x}^-, & |S_x^+| = |S_x^-| \wedge |\tilde{S}_x^+| < |\tilde{S}_x^-| \end{cases}$$

比较点集尺寸决定
x轴的最终方向

$M(k) \doteq \{i : |m - i| \leq k, m = \arg \operatorname{median}_j d_j\}$
将邻域半径缩小到当前邻域点距离的中值

- 有4种模糊度，
- 删除不符合右手法则的，还剩下两个模糊度

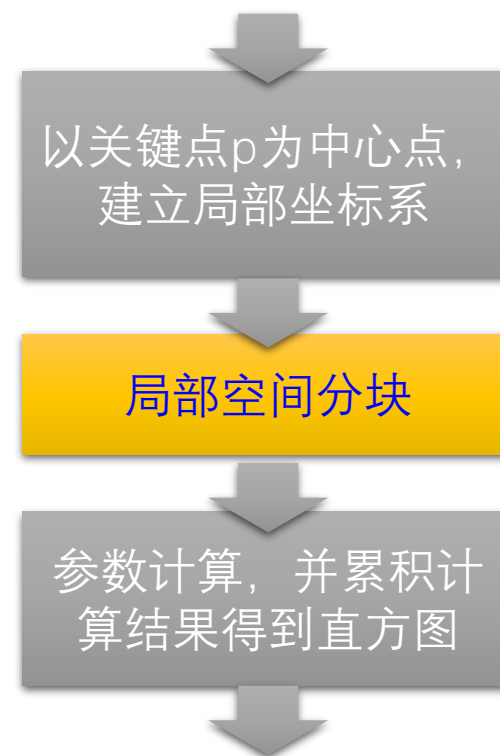
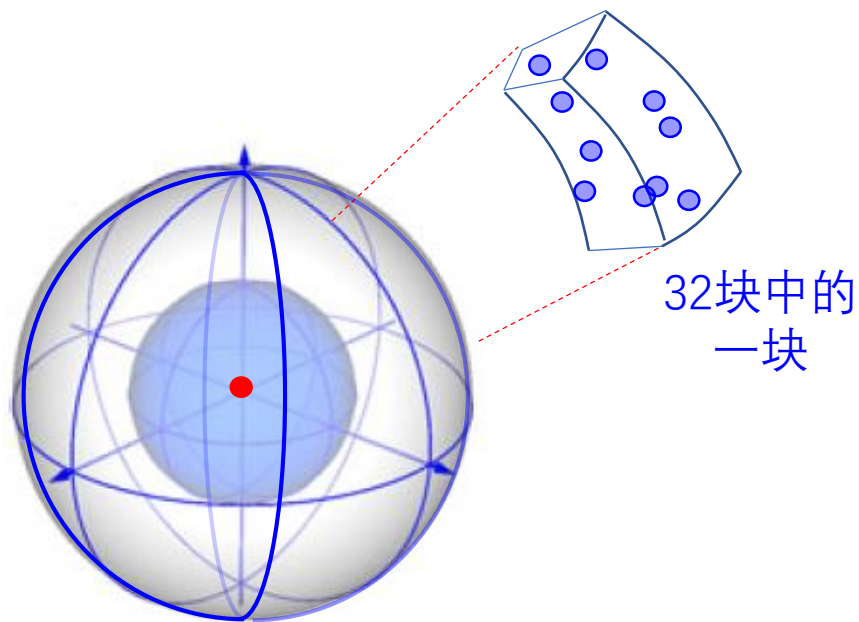


X轴指向

SHOT特征提取算法

下面介绍如何构建特征描述向量

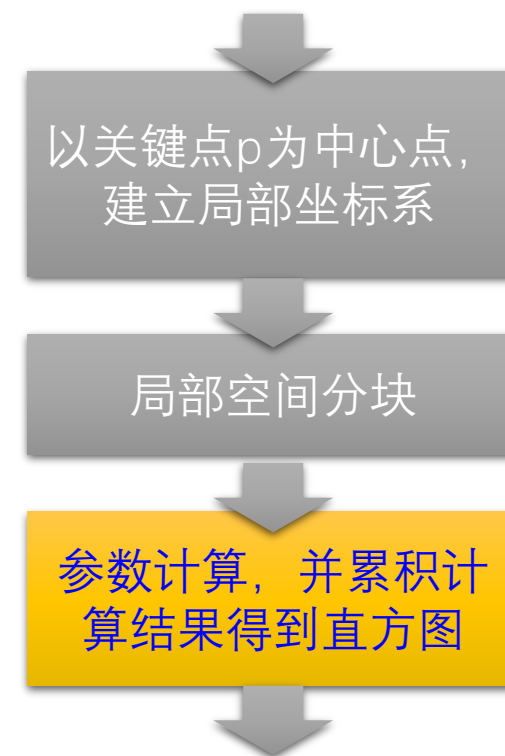
1. 构造半径为 r 的球形区域
2. 球体分割成32块，划分方法是：
 - 径向2个“球层”
 - 经度方向8分
 - 纬度方向2分



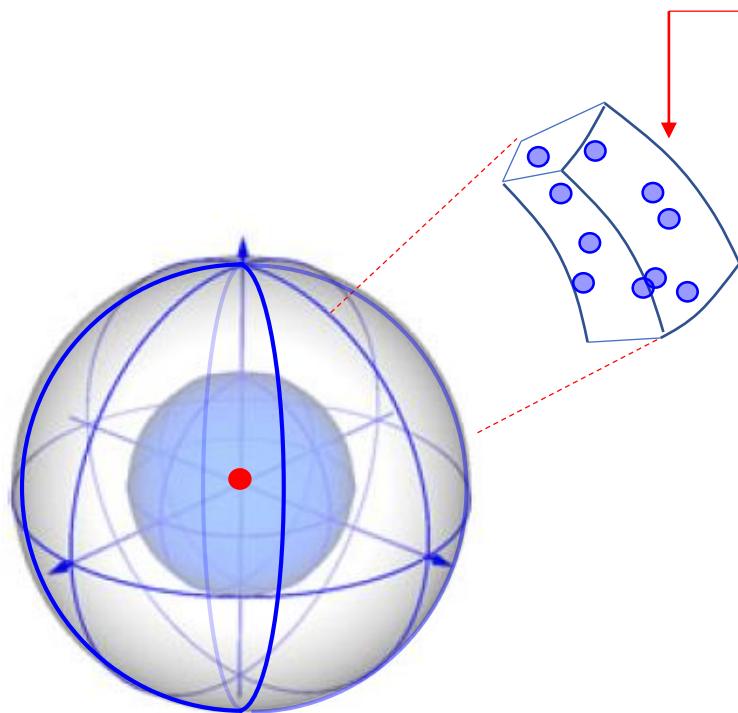
特征提取算法:

SHOT(Signature of Histograms of Orientations)

3. 在球体分块计算计算分块内所有点和中心点的法向量夹角余弦
 4. 计算每个分块的夹角余弦值的直方图, 分成11个量化区间
- 一共得到352 ($=11 \times 32$)维特征



SHOT特征提取——夹角余弦直方图



- 对球体上的一个分块（共32块中的一块）
- 计算其中每点 q 和中心点 p 的法向量夹角余弦

$$\cos \theta_q = \langle \mathbf{z}, \mathbf{n}_q \rangle$$

$\cos \theta_q$ 夹角
 \mathbf{z} 关键点的法向量（就是新坐标系里的Z轴）
 \mathbf{n}_q 点 p 的法向量

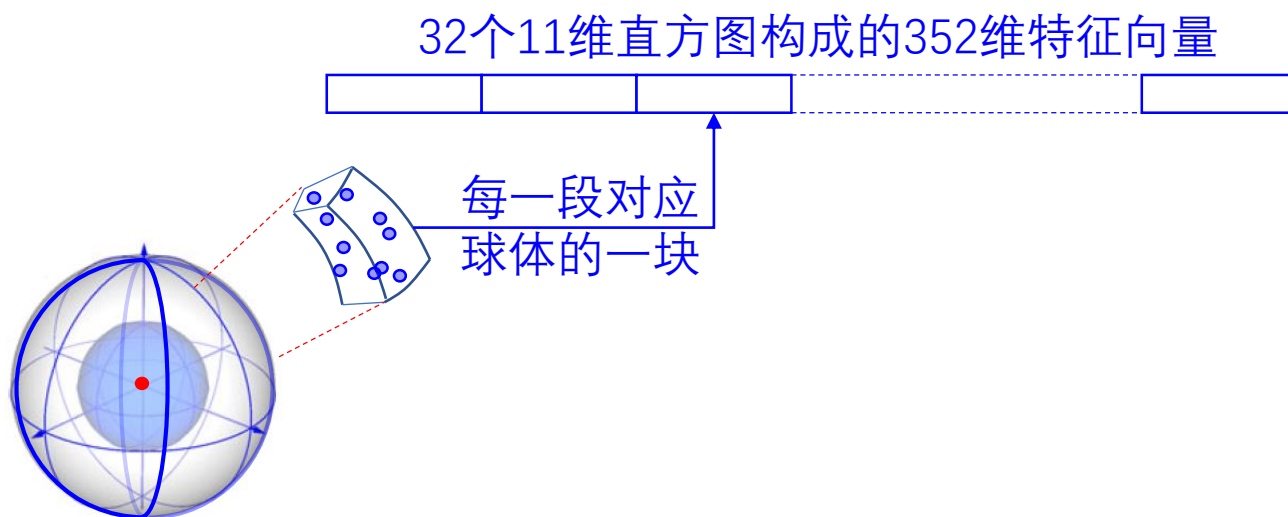
- 将角度余弦值区间分为11等分，计算落入每个区间的角度余弦值数量，构成直方图（这一过程实现时，进行了改进，不是直接计数，见下一页）

备注：

1. 这里计算夹角余弦时假设 \mathbf{z} 和 \mathbf{n} 是长度=1的单位向量（向量元素平方和=1），如果不是，则改成： $\cos \theta_q = \frac{\langle \mathbf{z}, \mathbf{n}_q \rangle}{\|\mathbf{z}\| \|\mathbf{n}_q\|}$
2. 公式里 $\langle \mathbf{z}, \mathbf{n}_q \rangle$ 表示两个向量 \mathbf{z}, \mathbf{n}_q 逐元素相乘后求和， $\|\mathbf{z}\|_2$ 表示向量 \mathbf{z} 元素平方和的开根号

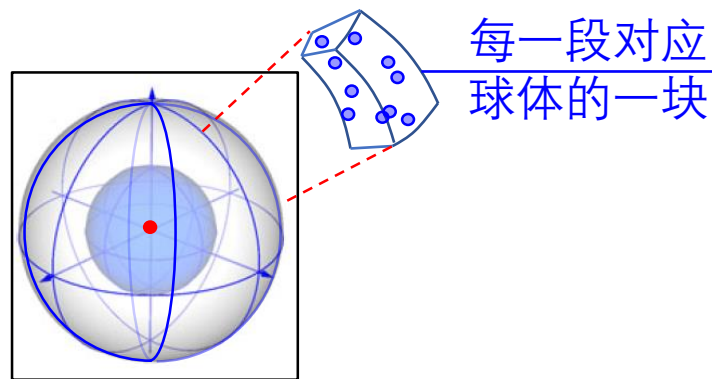
SHOT特征提取——直方图数据生成

- 考虑点的位置，每个点会影响352维特征描述向量的不止一个元素值
 - 比如考虑点 q ，它会影响哪些特征向量元素呢？
 - 对每个点 q ，对所影响到的特征描述向量的元素（若干个直方图的若干个bin，不止一个）不是简单加1，而是加上一个0-1的浮点数： $(1-d)$
 - 加多少？
 - 每个点 q 和分块之间的相对位置，以及夹角 \cos 值，计算它对32个11元素的直方图的“影响力”。



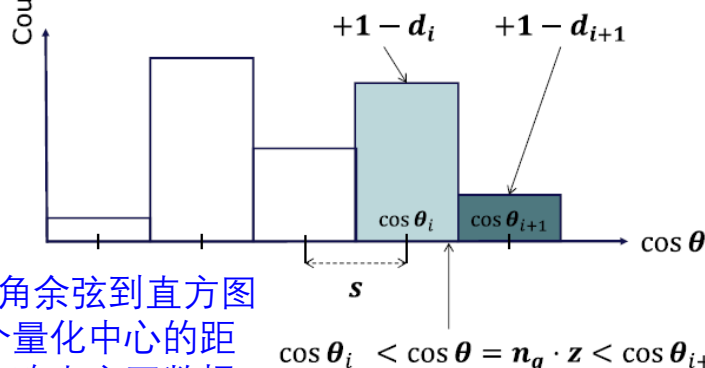
SHOT特征提取——直方图数据生成(细节)

32个11维直方图构成的352维特征向量



$$d_i = |n_q \cdot z - \cos \theta_i|/s$$

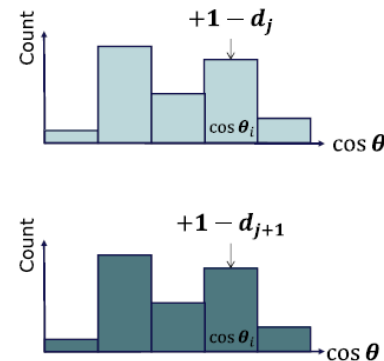
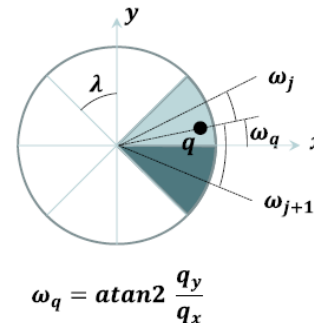
$$d_{i+1} = |n_q \cdot z - \cos \theta_{i+1}|/s$$



点q的夹角余弦到直方图最近两个量化中心的距离用于修改直方图数据

(a) Interpolation on normal cosines

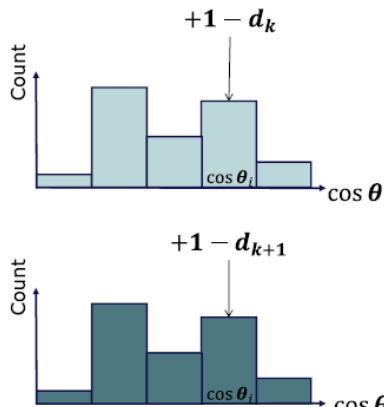
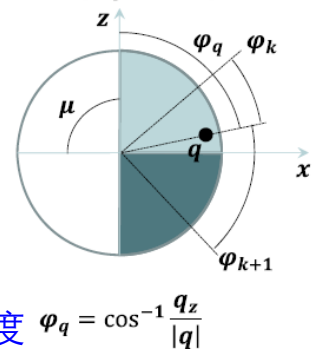
$$d_j = |\omega_q - \omega_j|/\lambda$$
$$d_{j+1} = |\omega_q - \omega_{j+1}|/\lambda$$



点q位置相邻经度球块对应左右两块直方图内容增量

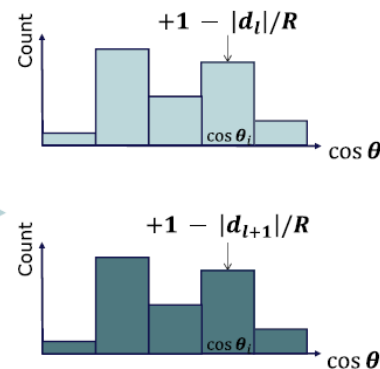
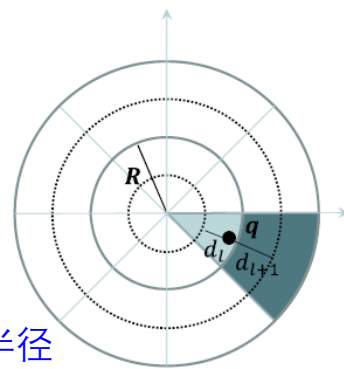
(b) Interpolation on azimuth

$$d_k = |\varphi_q - \varphi_k|/\mu$$
$$d_{k+1} = |\varphi_q - \varphi_{k+1}|/\mu$$



点q位置相邻纬度球块对应的上下两块直方图内容增量

(c) Interpolation on elevation



点q位置相邻半径球块对应的里外两块直方图内容增量

(d) Interpolation on distance

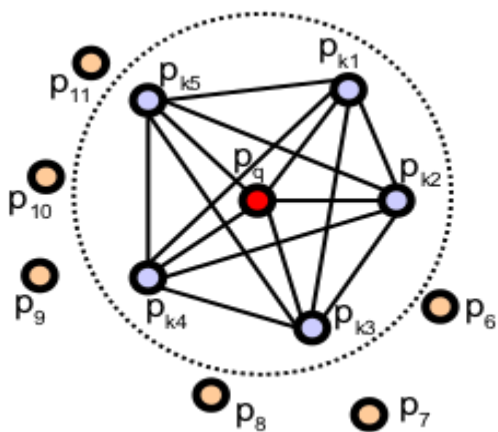
- 对于每个点q, 分4轮查找影响的特征向量的元素
- 受影响元素加上1-d

内容

- 人体肢体识别
- 物体识别与点云配准
 - 基于关键点特征的点云识别/粗配准流程（ISS特征方法）
 - SHOT点云特征提取算法
 - PFH点云特征提取算法
 - PPF点云识别配准算法

点特征直方图(Point Feature Histograms)

- 它没有像前面ISS或者SHOT那样精细统计邻域空间分块内的直方图，而是无差别地计算邻域点集总体的参数直方图
- 对关键点，获取它的近邻，并计算这些近邻点的所有成对组合关系的直方图
- 用到近邻点的局部法向量信息



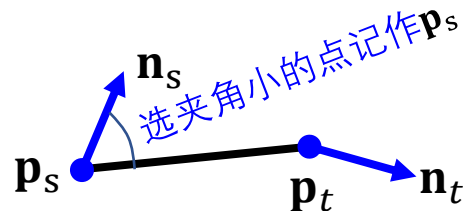
- 红色是关键点
- 蓝色是关键点的半径 r 球体内的近邻点
- 图中连线显示了这些近邻点（包括关键点）的所有可能的配对（完全连接）

注意：后面的内容来自论文，和PCL实现的PFH有所不同

点特征直方图(Point Feature Histograms, 步骤1、2、3)

步骤1、2、3

1. 对于点对 \mathbf{p}_s 和 \mathbf{p}_t , 首先确保 \mathbf{p}_s 、 \mathbf{p}_t 的连线和 \mathbf{p}_s 法向量夹角小于他们的连线和 \mathbf{p}_t 法向量夹角
(如果不是, 则交换 \mathbf{p}_s 和 \mathbf{p}_t 的命名)



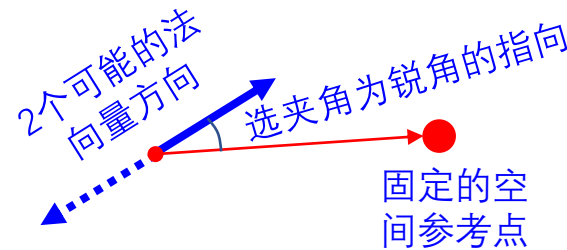
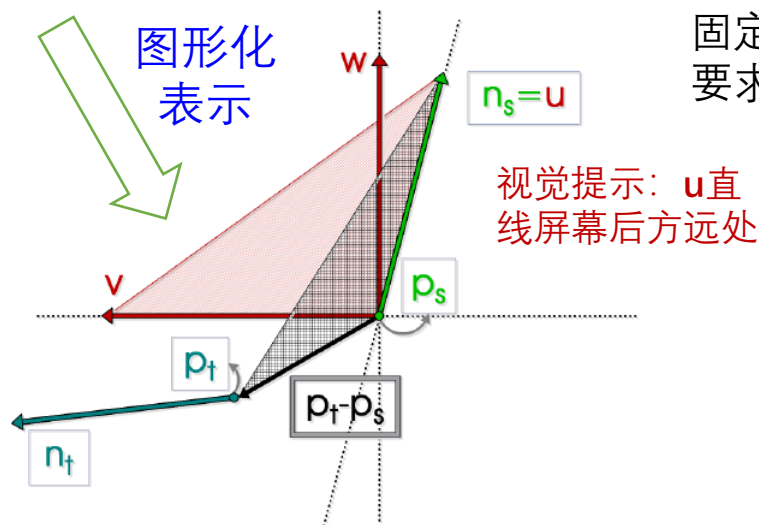
点怎么定义法向量? 是指点所在邻域点云通过PCA计算得到的法向量

2. 针对点对 $(\mathbf{p}_s, \mathbf{q}_t)$ 建立三个正交坐标轴 $(\mathbf{u}, \mathbf{v}, \mathbf{w})$ (即: 3个归一化的正交向量)

$$3. \begin{cases} \mathbf{u} = \mathbf{n}_s \\ \mathbf{v} = \frac{(\mathbf{p}_t - \mathbf{p}_s)}{\|\mathbf{p}_t - \mathbf{p}_s\|} \times \mathbf{u} \\ \mathbf{w} = \mathbf{u} \times \mathbf{v} \end{cases}$$

注意: \mathbf{v} 不同于原始论文

- \mathbf{p}_s 的局部法向量 (归一化了)
- 通过PCA分析得到 (最小特征值对应的特征向量)
- 注意: 法向量有正反方向模糊度, 需要用固定的准则确定使用其中一个方向, 比如要求指向特定的空间位置



点特征直方图(Point Feature Histograms, 步骤4、5、6)

步骤4、5、6

4. 计算4个特征参数 $\{f_0, f_1, f_2, f_3\}$

$$\begin{cases} f_0 = \langle \mathbf{v}, \mathbf{n}_t \rangle & \text{夹角} \\ f_1 = \|\mathbf{p}_t - \mathbf{p}_s\| & \text{距离} \\ f_2 = \langle \mathbf{u}, \mathbf{p}_t - \mathbf{p}_s \rangle / f_1 & \text{夹角} \\ f_3 = \text{atan}(\langle \mathbf{w}, \mathbf{n}_t \rangle, \langle \mathbf{u}, \mathbf{n}_t \rangle) & \text{角度 函数atan2} \end{cases}$$

5. 4个特征分别和门限 $\{s_0, s_1, s_2, s_3\}$ （用户指定的参数）比较，比较结果以4-bit二进制形式表示，得到取值范围为0-15的整数idx

4-bit二进制表示的整数

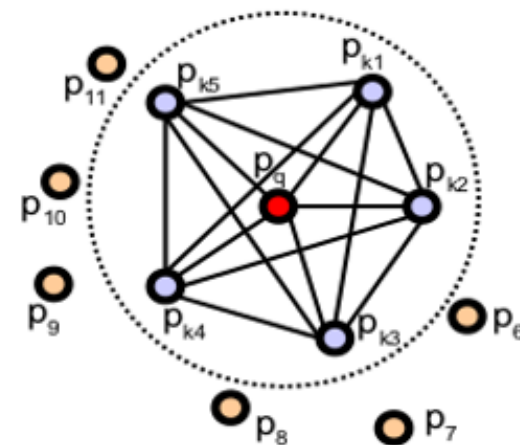
b_3	b_2	b_1	b_0
-------	-------	-------	-------

 对于第n比特，当 $f_n > s_n$ 时 $b_n=1$ ，否则 $b_n=0$

6. 对于点p邻域的K个点，共考虑 $\frac{K(K-1)}{2}$ 种配对可能，得到 $\frac{K(K-1)}{2}$ 个idx，

我们计算 $\frac{K(K-1)}{2}$ 个idx值计算**直方图**，由于idx取值为0-15，直方图就直接分成16个离散区间，对应idx的每种取值

这里16元素的直方图，就是围绕给定关键点的特征描述向量



内容

- 人体肢体识别
- 物体识别与点云配准
 - 基于关键点特征的点云识别/粗配准流程（ISS特征方法）
 - SHOT点云特征提取算法
 - PFH点云特征提取算法
 - PPF点云识别配准算法

Point Pair Features (PPF)

- 这一方法看成是hough算法的一种
- 它不是基于关键点特征的，而是统计全体点的特征
- 通过参数投票机制得到配准的点以及对应的位姿信息

- 应用场景

- 识别场景中的物件
- 被识别对象的完整3D模型事先获取
- 为加快速度，数据库内的3D模型进行预处理建立查询表



Point Pair Features——算法细节1

- 识别匹配前的准备工作——为待识别物体的完整3D模型样本建立查询表
- 物体模型样本以表面点云的形式给出(不包括物体内部)
- 查询表内存放点云中所有两两配对的点的信息
 - 比如对于N个点的模型点云，有 $N(N-1)$ 种配对方式
 - 由于PPF特征是和配对点的次序有关，对每个配对的点，交换点的次序得到的是不同的特征，因此实际上需要保存 N^2 对信息



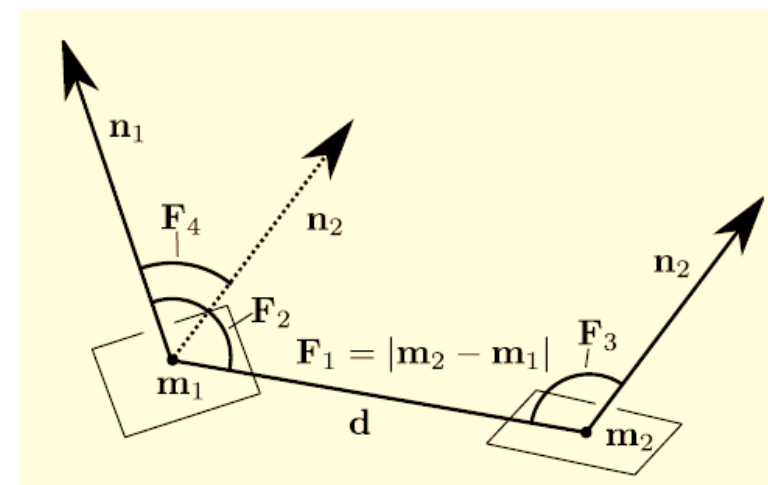
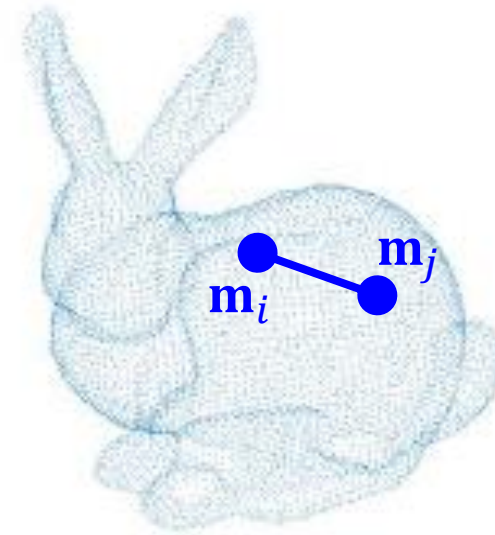
Point Pair Features——算法细节2

考虑点对 $(\mathbf{m}_i, \mathbf{m}_j)$ ，下面是需要计算的特征

1. 分别计算点 \mathbf{m}_i 和 \mathbf{m}_j 的法向量 \mathbf{n}_i 和 \mathbf{n}_j
2. 进一步计算下面4元素特征描述向量，作为点对 $(\mathbf{m}_1, \mathbf{m}_2)$ 的特征描述

$$\mathbf{F}(\mathbf{m}_i, \mathbf{m}_j) = (\|\mathbf{m}_i - \mathbf{m}_j\|_2, \angle(\mathbf{n}_i, \mathbf{d}), \angle(\mathbf{n}_j, \mathbf{d}), \angle(\mathbf{n}_i, \mathbf{n}_j))$$

- 上面 $\mathbf{F}(\mathbf{m}_i, \mathbf{m}_j)$ 对应的4个元素分别记作 (F_1, F_2, F_3, F_4) 他们的几何含义如左图和下面所示：
 - 分别对应一个距离和3个角度
 - 注意： $\mathbf{F}(\mathbf{m}_i, \mathbf{m}_j) \neq \mathbf{F}(\mathbf{m}_j, \mathbf{m}_i)$
- 我们要对模型内所有可能的两两点对计算上述4元素特征描述向量
- 我们还需要保存所有计算得到的4元素特征描述向量



这个有麻烦，这么多数据
存放困难，检索更困难



下面给出好的解决办法

Point Pair Features——算法细节3

对计算得到的所有两两点对的4元素特征描述向量 $\mathbf{F}(\mathbf{m}_i, \mathbf{m}_j)$ ，我们先对它进行量化，再用哈希表存放

$$\mathbf{F}(\mathbf{m}_i, \mathbf{m}_j) = \left(\|\mathbf{m}_i - \mathbf{m}_j\|_2, \angle(\mathbf{n}_i, \mathbf{d}), \angle(\mathbf{n}_j, \mathbf{d}), \angle(\mathbf{n}_i, \mathbf{n}_j) \right)$$

3. 对 $\mathbf{F}(\mathbf{m}_i, \mathbf{m}_j)$ 每个元素量化，

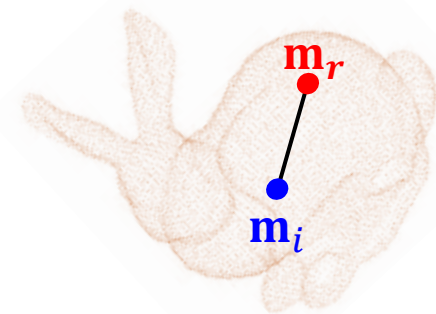
- 距离用量化阶 d_{dist} 的整数倍表示
- 角度用量化阶 $d_{angle} = \frac{2\pi}{N_{angle}}$ 的整数倍表示
- 经过上述处理后， $\mathbf{F}(\mathbf{m}_i, \mathbf{m}_j)$ 就变成了4个元素是 **整数的向量**了

4. 对量化的表示为整数的 $\mathbf{F}(\mathbf{m}_i, \mathbf{m}_j)$ 进行Hash计算

每对特征的Hash值	所有具有相同hash值的点对
Hash[F(m _i ,m _j)]	(m _i ,m _j)
10110101	(m ₁ ,m ₂), (m ₂₃ ,m ₆), (m ₂ ,m ₃)
11101011	(m ₇ ,m ₆)
11010110	(m ₂ ,m ₅), (m ₆ ,m ₂₅)
00100111	(m ₄₅ ,m ₁₂), (m ₃₁ ,m ₂₂), (m ₅ ,m ₇₂)
...	...

Hash表的示意图

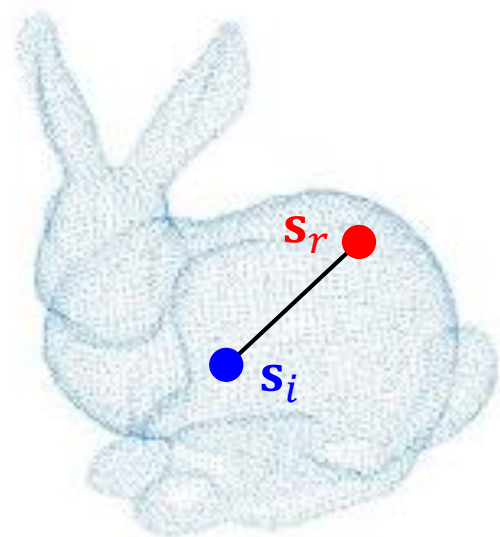
预先已知的
模型“样本”



预先计算得
到Hash表格

Point Pair Features——算法细节4

之前为模型建立了PPF查询表，下面介绍使用时，如何从这张查询表得到点云的识别或者匹配结果



待识别的
未知点云

1. 任意取一点 s_r (红色点)，作为匹配的基准点（固定下来），计算他的法向量
2. 计算它和点云中其他点(记作 s_i ，蓝色点)构成的点对的PPF向量，即： $F(s_r, s_i)$
3. 对上述特征描述向量量化，并求hash值 $idx = Hash(F(s_r, s_i))$
4. 对上述 idx ，查询实现计算的Hash表，得到对应表项内所有可能的匹配点对： $\{(m_a, m_b)\}$ （集合，可以不止一对）

Hash[F(m_i, m_j)]	(m_i, m_j)
10110101	(m_1, m_2), (m_{23}, m_6), (m_2, m_3)
11101011	(m_7, m_6)
11010110	(m_2, m_5), (m_6, m_{25})
00100111	(m_{45}, m_{12}), (m_{31}, m_{22}), (m_5, m_{72})
...	...

每对特征的
Hash值

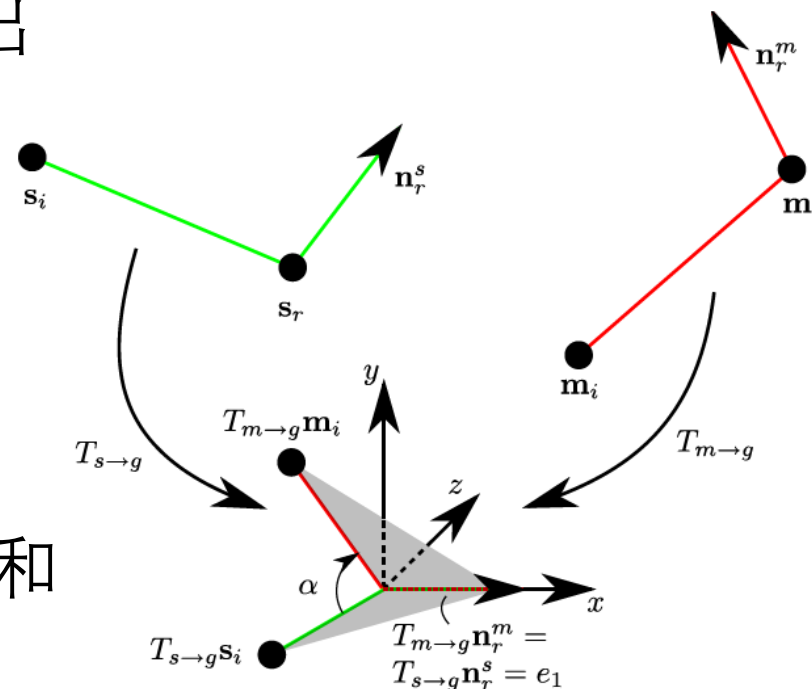
所有具有相同
hash值的点对

Point Pair Features——算法细节4

上一页根据 $(\mathbf{s}_r, \mathbf{s}_i)$ 的PPF特征，在Hash表内找到与之匹配的多个点对 $\{(\mathbf{m}_a, \mathbf{m}_b)\}$ （集合，可以不止一对），我们根据下面内容计算一个角度 α

- 考虑 $(\mathbf{s}_r, \mathbf{s}_i)$ 和匹配上的一个点对 $(\mathbf{m}_r, \mathbf{m}_i)$ ，画出对应的连线和法向量，构成两个“勾形”支架
- 我们对他们进行下面计算：
 - 平移旋转两个“勾形”支架，使得：
 - \mathbf{s}_r 和 \mathbf{m}_r 处于全局坐标系原点
 - 两个法向量 \mathbf{n}_r^s 和 \mathbf{n}_r^m 和全局坐标系的x轴重合
 - 计算对齐了的两个“勾形”支架中 $(\mathbf{s}_r, \mathbf{s}_i)$ 连线和 $(\mathbf{m}_r, \mathbf{m}_i)$ 连线之间的角度 α ，写成公式为：

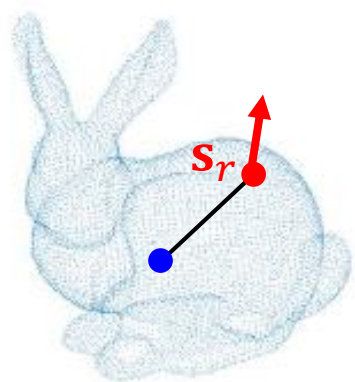
$$\mathbf{s}_i = T_{s \rightarrow g}^{-1} R_x(\alpha) T_{m \rightarrow g} \mathbf{m}_i$$



对于待识别点云上给定了的 \mathbf{s}_r ，如果数据库内点云上的 \mathbf{m}_r 的确和 \mathbf{s}_r 是相同物件相同位置的点，并且 \mathbf{s}_i 和 \mathbf{m}_i 也一样正确匹配，那么计算出来的角度 α 是一个固定值，由 \mathbf{s}_r 的位置，以及待检测点云和数据内“样本”点云的相对旋转量决定

Point Pair Features——算法细节4

1. \mathbf{s}_r 固定，对不同的 \mathbf{s}_i ，按上述方法，找到Hash表里面所有匹配的点对集合 $\{(\mathbf{m}_r, \mathbf{m}_i)\}$
2. 我们对上述集合中每对点 $(\mathbf{m}_r, \mathbf{m}_i)$ 计算 $F(\mathbf{m}_r, \mathbf{m}_i)$ ，和 $F(\mathbf{s}_r, \mathbf{s}_i)$ 比对，删除差别超出门限的点对
3. 按前一页的算法，对第2步筛选过的点对集合里每个点对 $(\mathbf{m}_r, \mathbf{m}_i)$ ，计算 α ，对它量化得到 $\hat{\alpha}$ ，在下面的2D表格中投票——对应格子 $(\mathbf{m}_r, \hat{\alpha})$ 内容+1
4. 如果待识别点云和数据库内建立Hash表格使用了相同物体，那么遍历所有 \mathbf{s}_i ，用上面计算出来的各种 α 在下面表格里的投票结果中，最大值格子对应的 $(\mathbf{m}, \hat{\alpha})$ 就是数据库里“样本”点云在和 \mathbf{s}_r 相同位置点 \mathbf{m} ，并且将 \mathbf{m} 处的法向量 \mathbf{n}^m 和待识别点云 \mathbf{s}_r 位置法向量 \mathbf{n}_r^s 对齐后两者相差的一个旋转角度 $\approx \hat{\alpha}$

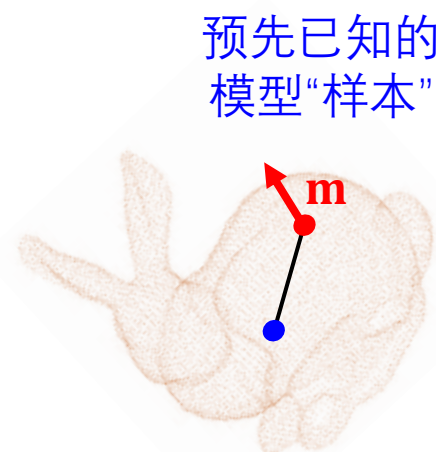


待识别的
未知点云

\mathbf{m}_1					
\mathbf{m}_2					
\mathbf{m}_3					
\vdots					
\mathbf{m}_N					

$\hat{\alpha} = 0^\circ \ 1^\circ \ 2^\circ \ \cdots \ 359^\circ$

每一行的各
列对应量化的
 α 值区间



预先已知的
模型“样本”

END

附录

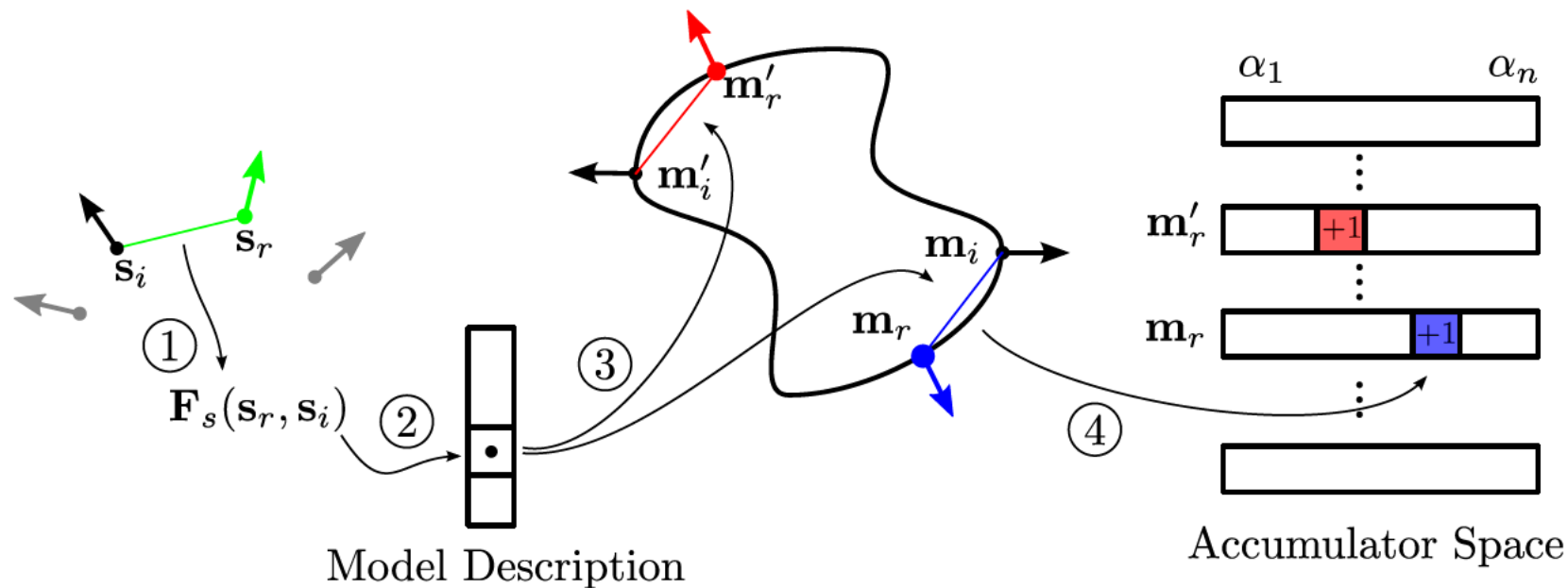


Figure 4. Visualisation of different steps in the voting scheme: (1) Reference point s_r is paired with every other point s_i , and their point pair feature \mathbf{F} is calculated. (2) Feature \mathbf{F} is matched to the global model description, which returns a set of point pairs on the model that have similar distance and orientation (3). For each point pair on the model matched to the point pair in the scene, the local coordinate α is calculated by solving $s_i = T_{s \rightarrow g}^{-1} R_x(\alpha) T_{s \rightarrow g} m_i$. (4) After α is calculated, a vote is cast for the local coordinate (m_i, α) .

附录

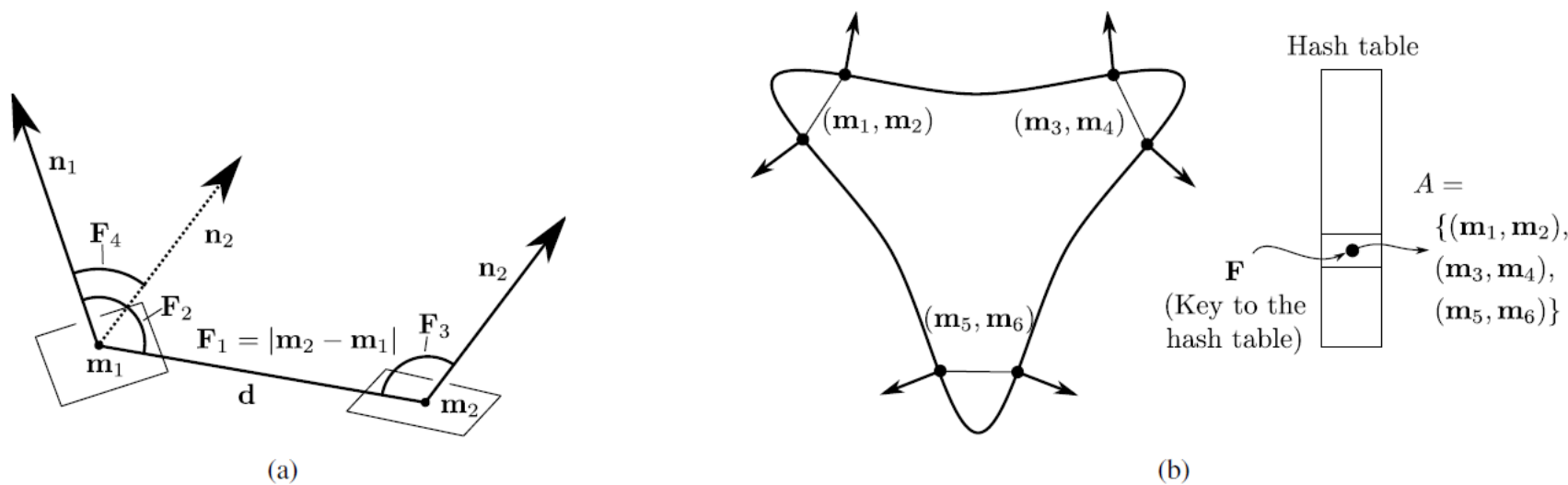


Figure 2. (a) Point pair feature \mathbf{F} of two oriented points. The component F_1 is set to the distance of the points, F_2 and F_3 to the angle between the normals and the vector defined by the two points, and F_4 to the angle between the two normals. (b) The global model description. Left: Point pairs on the model surface with similar feature vector \mathbf{F} . Right: Those point pairs are stored in the same slot in the hash table.