# 城市空间建模与仿真

## 第十三讲　城市空间三维数据特征学习与识别-超像素与点云超体素分割

任课教师：汤圣君
建筑与城市规划学院　城市空间信息工程系



其中部分图片来自互联网和同行专家

# 目录
CONTENTS

# 目录

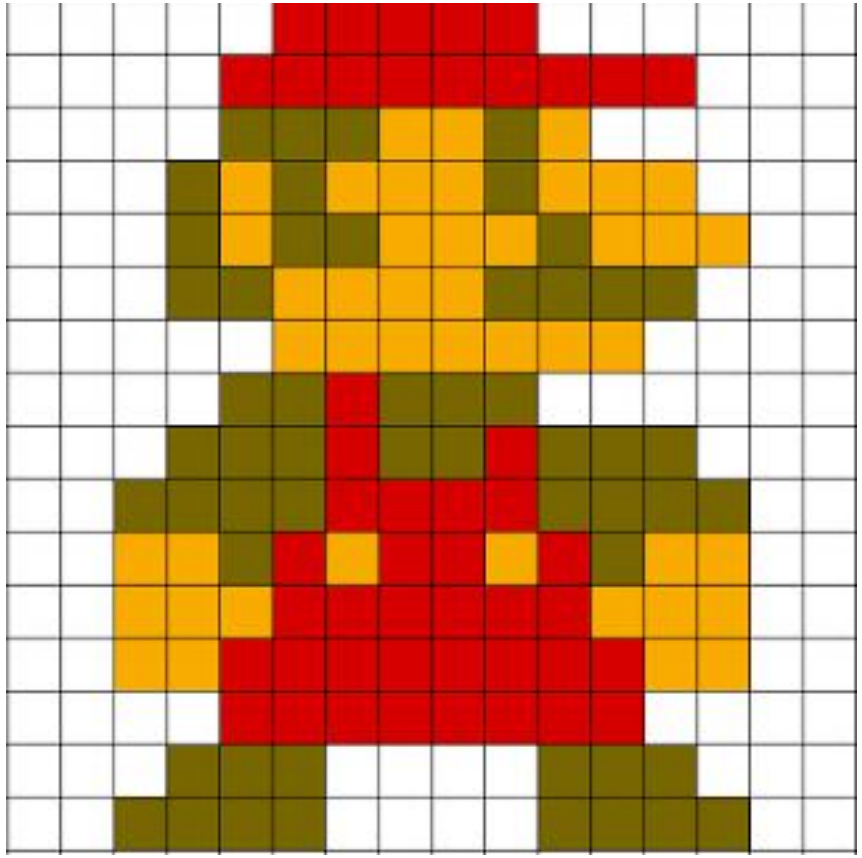
CONTENTS

# 超像素概念

- 由于图像分辨率的不断增大，导致了传统的基于像素的彩色图像分割算法的运算时间越来越长，如何**降低数据运算量**变成了彩色图像分割中的一大难题。

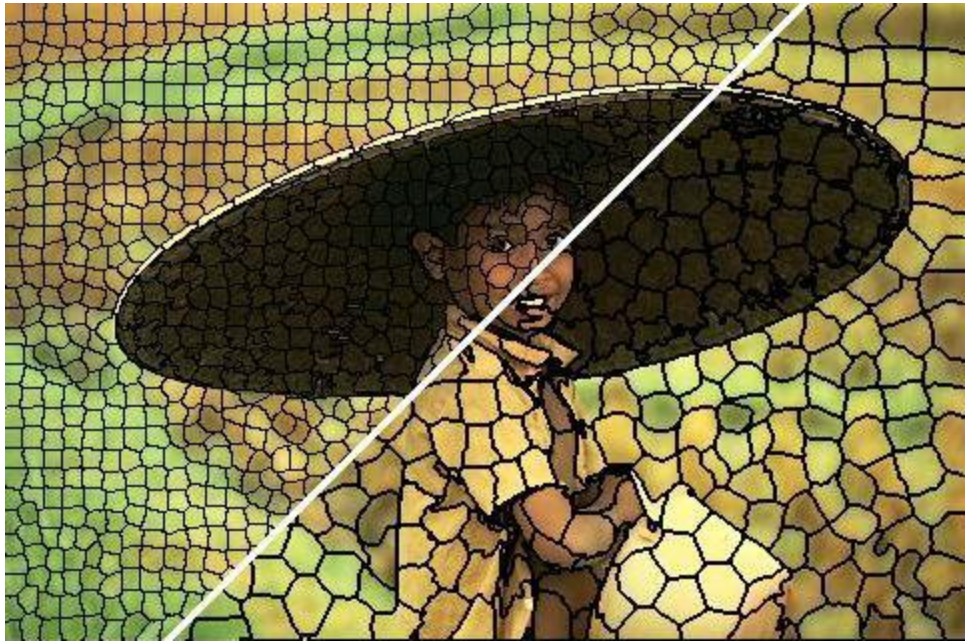
- 在图像算法中，**无监督的过分割是一种广泛的预处理步骤，将图像分割成具有相似属性的像素区域，**称之为超像素分割，该方法减少了之后后期算法计算的的成本，并且信息损失最小

- 超像素最大的功能之一，便是作为图像处理其他算法的预处理，在**不牺牲太大精确度的情况下降维！**



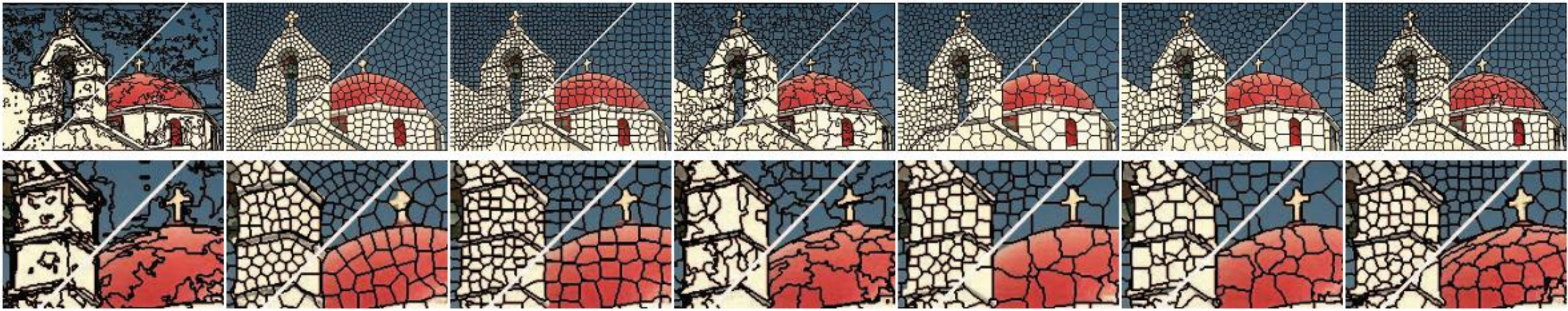
# 超像素概念

- 所谓超像素，即在图像中由一系列位置相邻且颜色、亮度、纹理等特征相似的像素点组成的小区域，这些小区域大多保留了进一步进行图像分割的有效信息，且一般不会破坏图像中物体的边界信息。
- 作用：
  - 大大降低了维度
  - 异常像素点剔除



# 超像素分割方法

- 已经广泛用于图像分割、姿势估计、目标跟踪、目标识别等计算机视觉应用。几种常见的超像素分割方法及其效果对比如下：



**Graph-based　　NCut　　Turbopixel　　Quick-shift　　Graph-cut a　　Graph-cut b　　SLIC**

- 对这些图像像素的计算成本随着节点的数目的增加而急剧的上升，这以为这每个像素对应一个节点求解图的方法变得十分困难，这就限制了他们在需要实时分割中的应用。

- 比较新的算法是，出现了一类速度明显更快的超混合方法——简单线性迭代聚类（SLIC）

# 超像素分割方法

- SLIC（simple linear iterativeclustering），即简单的线性迭代聚类。它是2010年提出的一种思想简单、实现方便的算法
  - 将彩色图像转化为CIELAB颜色空间和XY坐标下的5维特征向量
  - 对5维特征向量构造距离度量标准，对图像像素进行局部聚类的过程。
- SLIC算法能生成紧凑、近似均匀的超像素，在运算速度，物体轮廓保持、超像素形状方面具有较高的综合评价

## SLIC算法步骤

- 撒种子。将K个超像素中心分布到图像的像素点上。
- 微调种子的位置。以K为中心的3×3范围内，移动超像素中心到这9个点中梯度最小的点上。这样是为了避免超像素点落到噪点或者边界上。
- 初始化数据。取一个数组label保存每一个像素点属于哪个超像素。dis数组保存像素点到它属于的那个超像素中心的距离。
- 对每一个超像素中心x，它2S范围内的点：如果点到超像素中心x的距离（5维）小于这个点到它原来属于的超像素中心的距离，那么说明这个点属于超像素x。更新dis，更新label。
- 对每一个超像素中心，重新计算它的位置。
- 重复4 5 两步。

# 超像素分割方法

- 距离度量。包括颜色距离和空间距离。对于每个搜索到的像素点，分别计算它和该种子点的距离

$$d_c = \sqrt{(l_j - l_i)^2 + (a_j - a_i)^2 + (b_j - b_i)^2},$$

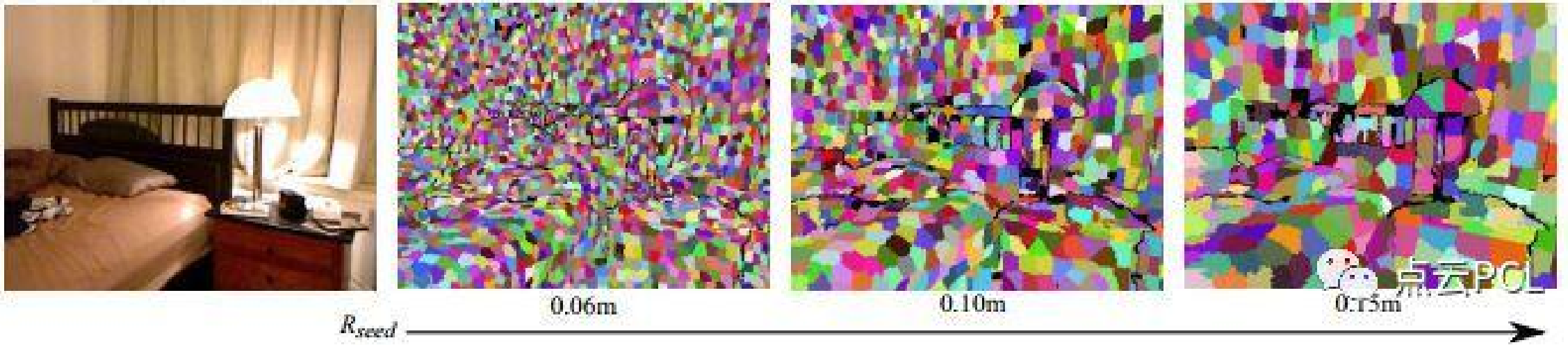
$$d_s = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2},$$

$$D' = \sqrt{\left(\frac{d_c}{N_c}\right)^2 + \left(\frac{d_s}{N_s}\right)^2}.$$

- 由于每个像素点都会被多个种子点搜索到，所以每个像素点都会有一个与周围种子点的距离，取最小值对应的种子点作为该像素点的聚类中心。

# 超像素分割方法-Superpixels

- 深度自适应Superpixels[2]最近将这种思想扩展到使用深度图像，通过增加深度和点云的法向角的维度来扩展聚类空间。虽然DASP是有效的，并且给出了有希望的结果，但它没有充分利用RGB+D数据，**仍然停留在2.5D方法的类中，因为它没有明确考虑三维连通性或几何性质。**



# 目录

CONTENTS

02 3D点云超体素分割

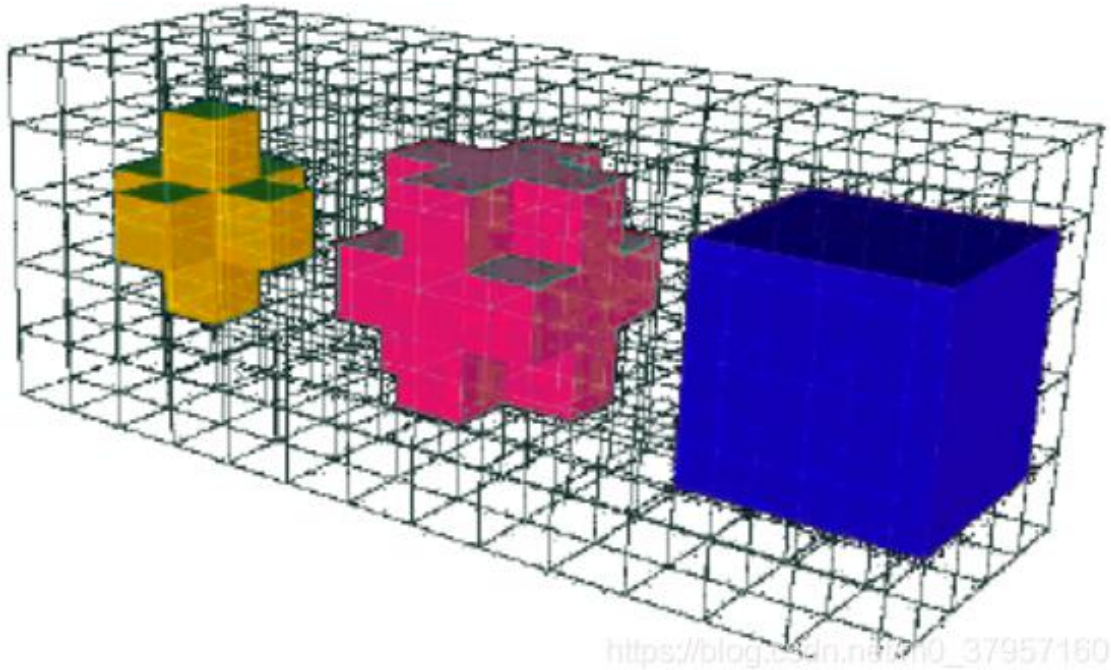
# 超体素分割

(Voxel Cloud Connectivity Segmentation VCC)

- 点云体素连接性分割（VCCS）是一种从三维点云数据生成超像素和超体素的新方法。VCCS产生的超体素比最新的方法更符合物体边界，同时该方法实时性更好。
- 优势
  - 超体素簇的种子是通过分割三维空间而不是投影到图像层面来实现的。这可以确保超体素是根据场景的几何属性均匀分布。
  - 迭代聚类算法在考虑聚类点时，对被占用的体素进行严格的空间连通性。这意味着超体素不能在三维空间中连接不相交的边界，即使它们在投影平面上是相连的。

# 邻接图

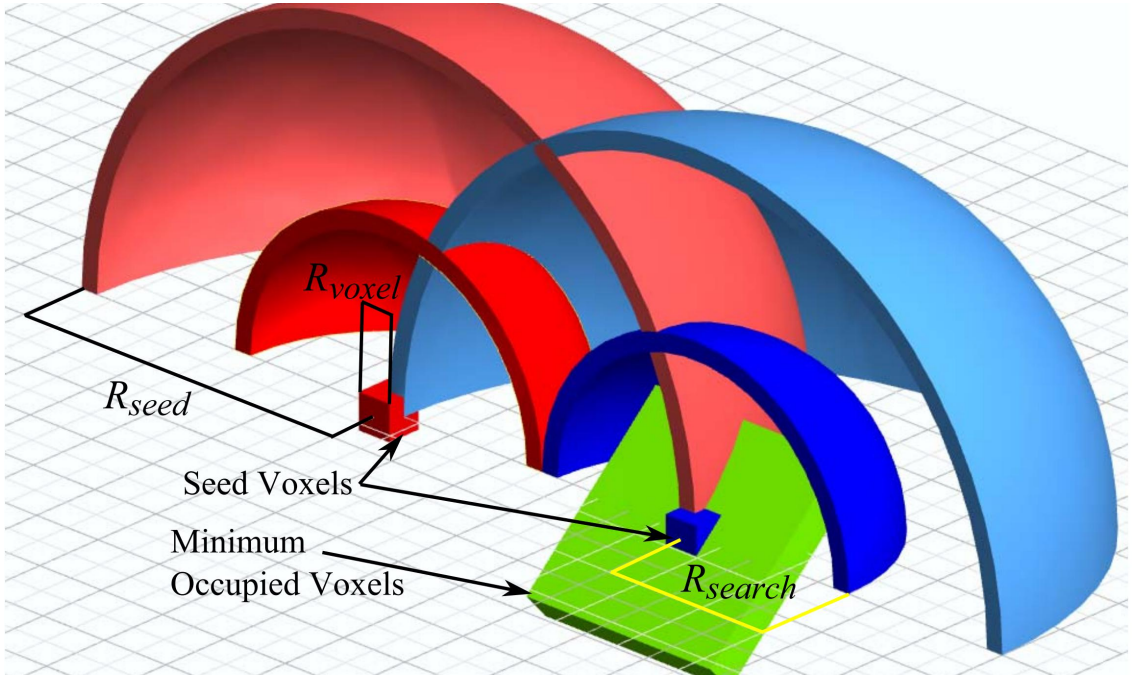
- 邻接性是该方法的一个关键步骤，这一步**能够确保各个超体素不会在空间中不相连的边界上有交集**
- 点云数据是杂乱、无组织结构的，但是通过体素化处理后，在体素空间内存在三种拓扑结构：**6 邻接、18 邻接以及 26 邻接**。其中 6 邻接的两个体素具有 6 个公共面，18 邻接的体素具有 12 条公共边和 6 个公共面，26 邻接体素在此基础上还具有 8 个公共点。
- 方法中仅仅采用26邻接体素构建体素点云的邻接图，一般是通过KD树进行实现。



所有的26相邻体素的中心都一定要在根号3 * Rvoxle中，其中Rvoxel是指用于分割的体素分辨率

# 种子点生成

算法首先将空间点云划分为一个具有选定分辨率Rseed的体素化网格，该Rseed的大小是明显高于Rvoxel，其中种子分辨率与体素分辨率的关系如图：




Rseed确定超级体素之间的距离，而Rvoxel确定点云量化的分辨率。
Rsearch用于确定是否有足够数量的种子占用体素。

# 特征初始化

过滤后，我们将剩余的种子归类为搜索体积中梯度最小的连接体素，其中的梯度计算公式：

$$G(i) = \sum_{k \in V_{adj}} \frac{\| V(i) - V(k) \|_{CIELab}}{N_{adj}};$$

通过在特征空间中找到种子体素的中心和两个体素内的连接邻域来初始化超体素特征向量。超体的特征和距离测度进行聚类。

$$\mathbf{F} = [x, y, z, L, a, b, \mathrm{FPFH}_{1..33}]$$

其中前三者是空间坐标，L,a，b为颜色，FPFH是局部几何特征点特征直方图，FPFH是具有空间不变性的描述局部表面模型的特征。

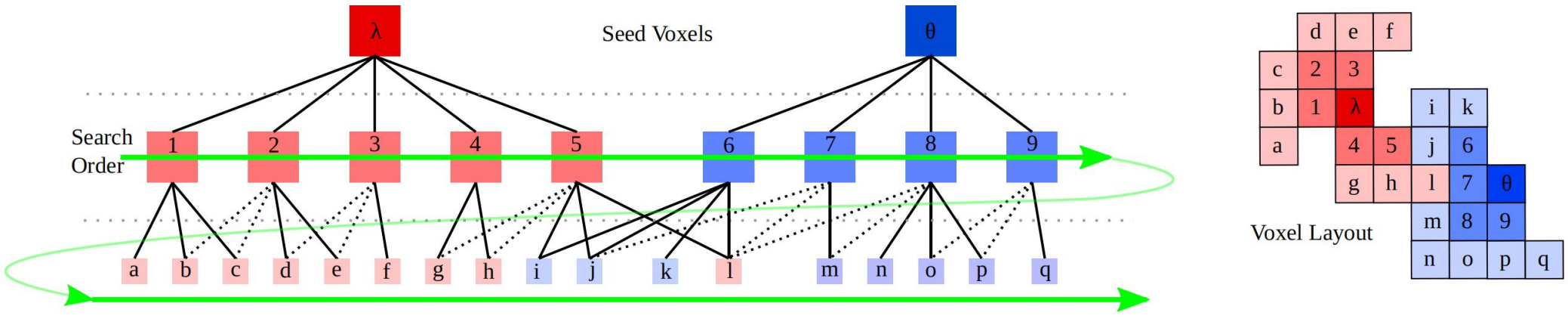
# 特征归一化

因为原始的距离信息可能会随Rseed的大小变化其相关性会发生变化，因此计算空间距离特征首先需要进行空间特征归一化操作。因此对于空间特征，色彩特征以及点云直方图特征可分别进行归一化操作。如下公式：

$$D = \sqrt{\frac{\lambda D_c^2}{m^2} + \frac{\mu D_s^2}{3R_{seed}^2} + \epsilon D_{HiK}^2},$$

- $D_s$ using the maximallydistant point considered for clustering, which will lie at a distance of $\sqrt{3}\,R_{seed}$
- Color distance $D_c$, is the euclidean distance in CIELab space, normalized by a constant m
- $D_f$ , is calculated using the HistogramIntersection Kernel

# 流约束迭代聚类




- 从距离点云簇中心最近的体素开始，我们向外流动到相邻的体素，并使用特征距离公式计算每个体素到超体素中心的距离。**如果距离是该体素所看到的最小距离，则设置其标签**，并使用邻接图将其距离中心更远的邻居添加到该标签的搜索队列中。
- 然后迭代下一个超级体素，**这样从中心向外的每一层都会同时考虑所有的超级体素**。我们反复向外搜索，直到找到每个超级体素的搜索体积的边缘（或者没有更多的邻居可以检查）
- 当我们**到达一个超级体素的邻接图的所有叶节点或者在当前级别中搜索的节点都没有设置为其标签时**，搜索就结束了。

# 流约束迭代聚类

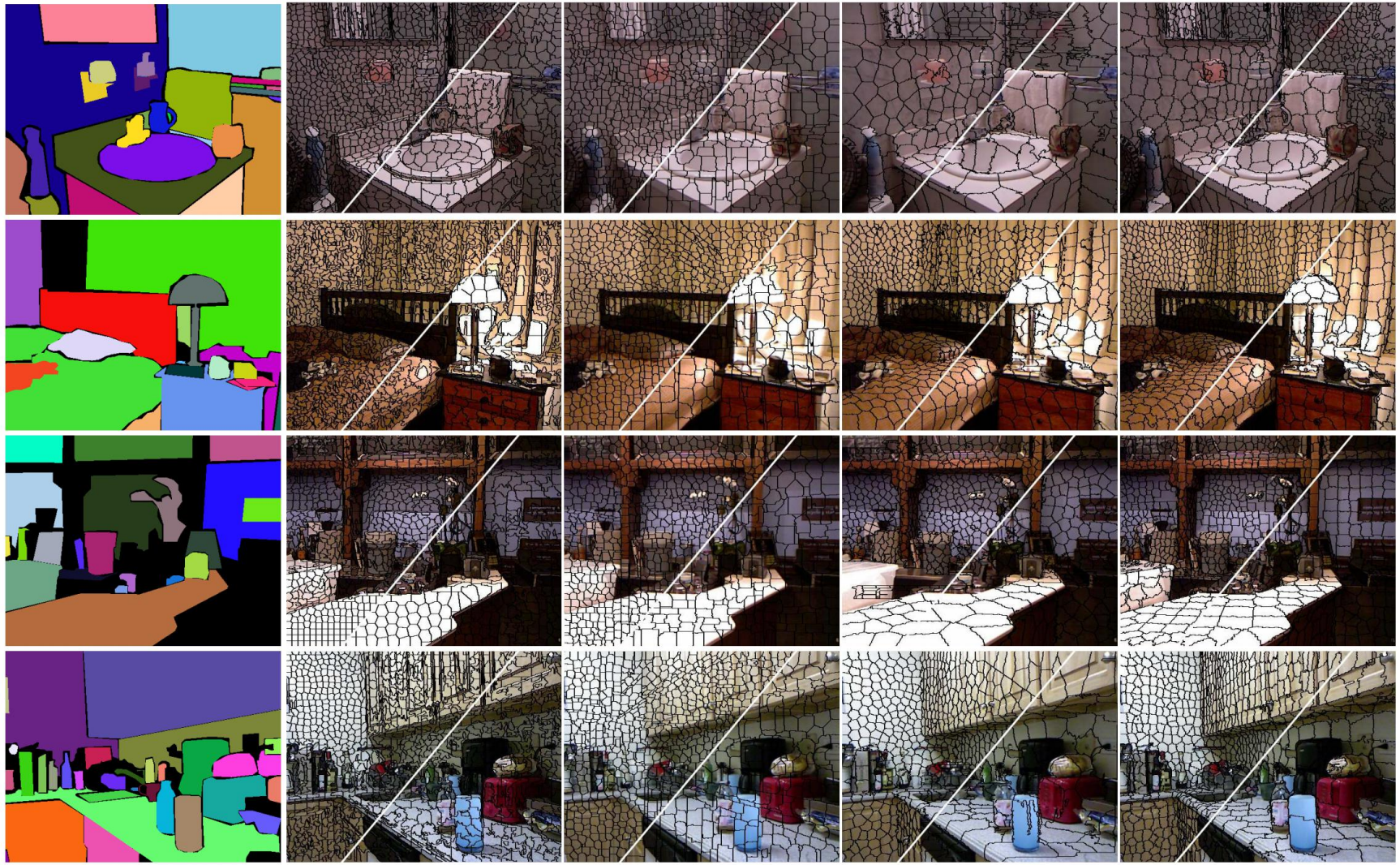
**流约束聚类算法的搜索的优势：**

1. 由于算法只考虑相邻的体素，因此超体素标签不能跨越在三维空间中实际不接触的对象边界
2. 超级体素标签在三维空间中往往是连续的，因为标签从每个超级体素的中心向外流动，在空间中以相同的速率扩展。

**总结：**

1. **超体聚类的目的并不是分割出某种特定物体，**其对点云实施过分割(over segmentation)，将场景点云化成很多小块，并研究每个小块之间的关系。
2. **本质上这种方法是对局部的一种总结，纹理，材质，颜色类似的部分会被自动的分割成一块，有利于后续识别工作。**比如对人的识别，如果能将头发，面部，四肢，躯干分开，则能更好的对各种姿态的人进行识别。

# 超体素分割示例



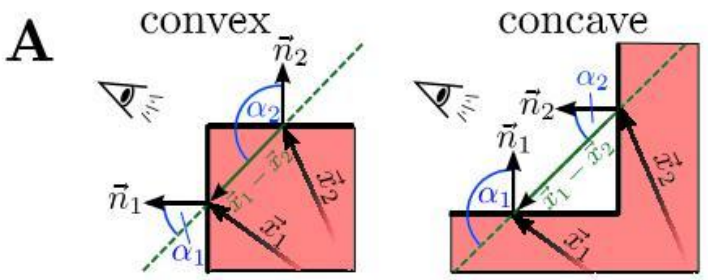
SLIC GCb10 DASP VCCS

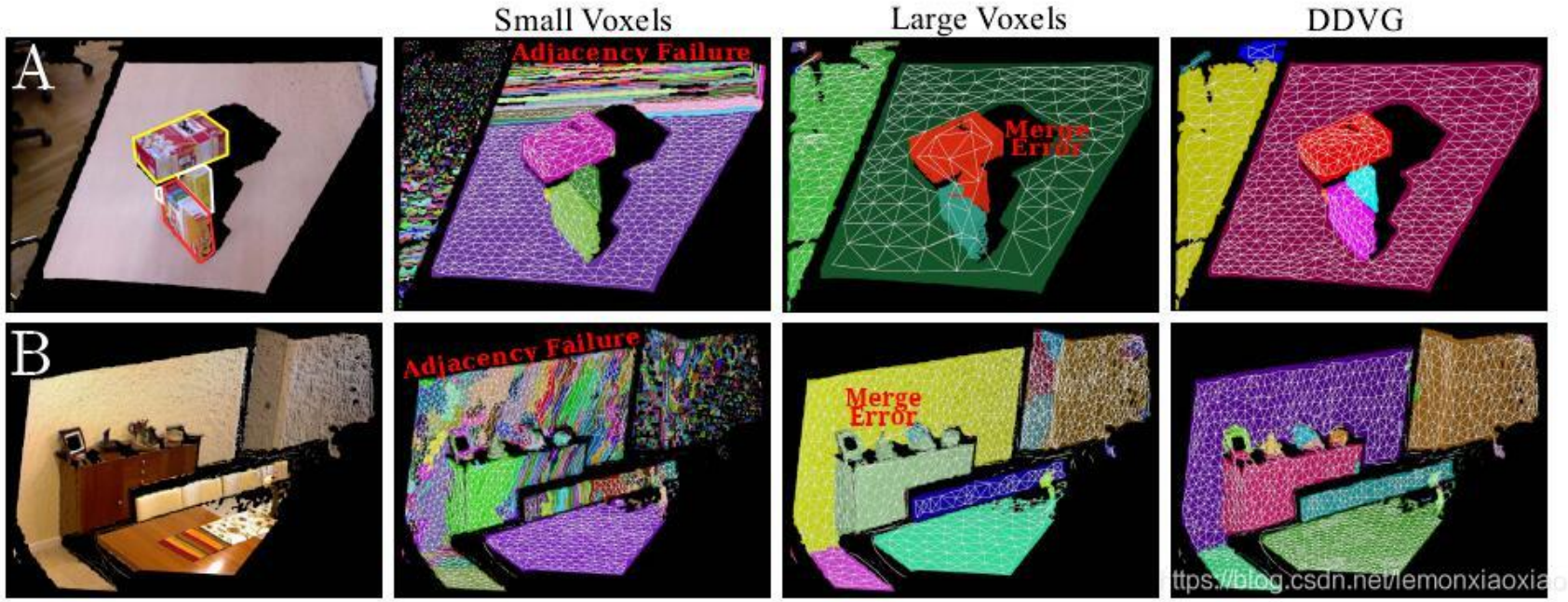
# 超体素聚类-凹凸算法

《Object Partitioning using Local Convexity》

LCCP是 Locally Convex Connected Patches 的缩写，翻译成中文叫做 ” 局部凸连接打包一波带走
“… …算法大致可以分成两个部分：1.基于超体聚类的过分割；2.在超体聚类的基础上再聚类。

原始点云 → 超体素计算 → 超体素凹凸关系计算 → 跨越凸边增长 → 分割结果



# 目录

CONTENTS

# 课堂练习：点云超体素分割

https://blog.csdn.net/lemonxiaoxiao/article/details/106262879?utm_medium=distribute.pc_relevant.none-task-blog-title-10&spm=1001.2101.3001.4242

- 要求：
  - 对任意给定点云进行不同分辨率的超体素分割，并超体素分割后的点云
  - 进行凹凸分割，输出超体素聚类后的点云


# 谢谢