SLIC算法学习报告

超像素生成算法简介 SLIC算法流程及复现思路 SLIC算法优点及创新点总结

超像素生成算法简介

超像素算法**将像素组合成感知有意义的原子区域**,其可以用于替换像素网格的刚性结构。它们捕获图像冗余,提供计算图像特征的方便原语,并且大大降低了后续图像处理任务的复杂性。

目前存在许多产生超像素的方法,每种具有其自身的优点和缺点,可更好地适合于特定应用,大致分类为基于图或梯度上升的方法。

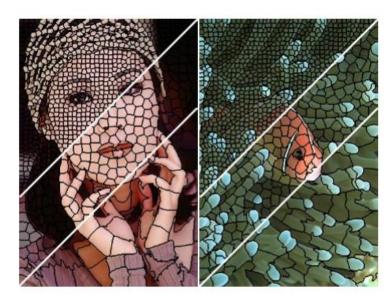
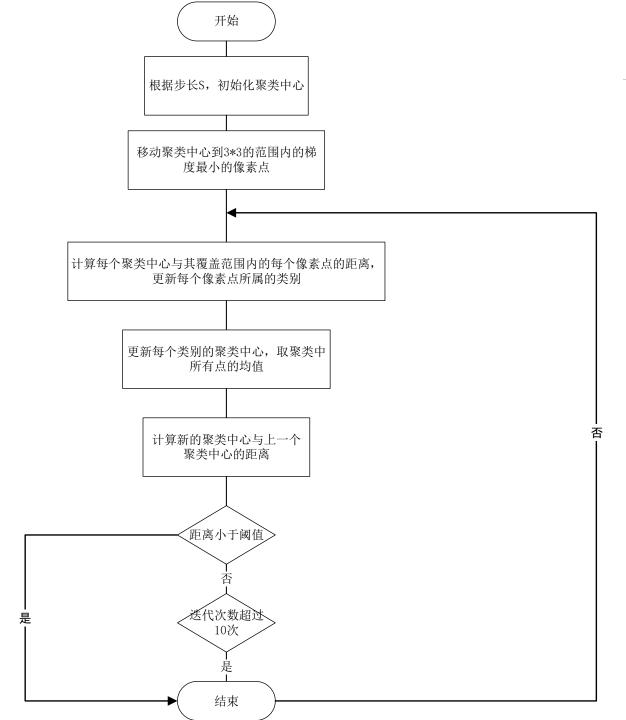


图: 使用SLIC分割成尺寸(大约)为64,256和1024的超像素结果

超像素生成算法衡量指标	良好的超像素生成算法需具备的 特点	SLIC表现
图像边界的粘附性	超像素应当良好地粘附到图像边界	对图像边界有良好的粘附性
算法速度与存储效率	计算速度快,存储器效率高且易于 使用	比现有算法快,有更高的存储效率
对分割性能的影响	当用于分割目的时,超像素应当增 加速度并提高结果质量	分割效果优于现有算法
是否能够控制超像素生成个数	是	是
是否能够控制超像素生成的复杂度	是	是
是否易于扩展到更高维	是	是

	Graph-based				Gradient-ascent-based					
	GS04 [8]	NC05 [23]	SL08 [21]	GCa10 ^b [26]	GCb10 ^b [26]	WS91 [28]	MS02 [4]	TP09 ^b [15]	QS09 [25]	SLIC
Adherence to boundaries				. ,		. ,				
Under-segmentation error (rank)	0.23	0.22	-	0.22	0.22	-	-	0.24	0.20	0.19
Boundary recall (rank)	0.84	0.68	-	0.69	0.70	-	-	0.61	0.79	0.82
Segmentation speed										
320×240 image	$1.08s^a$	178.15s	-	5.30s	4.12s	-	-	8.10s	4.66s	0.36s
2048×1536 image	$90.95 \mathrm{s}^a$	N/A^c	-	315s	235s	-	-	800s	181s	14.94s
Segmentation accuracy (using [11] on MSRC)	74.6%	75.9%	-	-	73.2%	-	-	62.0%	75.1%	76.9%
Control over amount of superpixels	No	Yes	Yes	Yes	Yes	No	No	Yes	No	Yes
Control over superpixel compactness	No	No	No	No^d	No^d	No	No	No	No	Yes
Supervoxel extension	No	No	No	Yes	Yes	Yes	No	No	No	Yes



SLIC算法流程

```
Algorithm 1 SLIC superpixel segmentation 

/* Initialization */
Initialize cluster centers C_k = [l_k, a_k, b_k, x_k, y_k]^T by sampling pixels at regular grid steps S.

Move cluster centers to the lowest gradient position in a
```

Move cluster centers to the lowest gradient position in 3×3 neighborhood.

```
Set label l(i) = -1 for each pixel i.
Set distance d(i) = \infty for each pixel i.
```

repeat

until $E \leq \text{threshold}$

```
/* Assignment */

for each cluster center C_k do

for each pixel i in a 2S \times 2S region around C_k do

Compute the distance D between C_k and i.

if D < d(i) then

set d(i) = D

set l(i) = k

end if

end for

end for

/* Update */

Compute new cluster centers.

Compute residual error E.
```

Class SLIC	
img	图像数据,由rgb读入,转为lab
height	图像高度
width	图像宽度
N	图像像素个数,height*width
S	初始化聚类中心采样间隔及限制聚
	类中心搜索空间的参数, $\sqrt{\frac{N}{K}}$
K	聚类个数
M	权衡颜色相似性和空间邻近度之间 相对重要性的参数,用于计算像素 间的距离
cluster	聚类中心,数据类型为list,每个 元素为自定义的Cluster类
label	每个像素点所属的聚类中心,数据 类型为dict
dis	各个像素点与其最近聚类中心的距离, numpy创建的二维数组, 初始化为np. inf

为实现SLIC算法, 定义的两个Class

Class Cluster	
num	聚类中心的编号
x, y	聚类中心的位置
1, a, b	聚类中心的图像信 息
point	该类包含的像素点, 数据类型为list

```
slic = SLIC("lenna.jpg", 500, 30)
    slic = initCluster(slic)
    slic = moveCluster(slic)
    slicProcessing(slic)
def slicProcessing(slic):
   for i in range(10):
       slic_old = copy.deepcopy(slic)
       slic = labelAssign(slic)
       slic = updateCluster(slic)
       e = getE(slic_old, slic)
       print("iter " + str(i) + " error:", e)
       saveImage(slic, 'image' + str(i) + ".png")
       if e <= 1:
          break
```

复现代码主要流程

SLIC的优点

1. 限制减少迭代时的计算量。

传统的k-means在更新时需要比较每个数据点与k个聚类中心的距离,进行所属类别的划分,时间复杂度为0(KN), SLIC限制了聚类的范围,只需要搜索聚类中心一定范围内的数据点与该中心的距离,进行所属类别更新,时间复杂度变为0(N),显着地减少了距离计算的数量。

2. 加权距离度量组合颜色和空间接近度,同时提供对超像素的尺寸和紧凑性的控制。

$$egin{array}{ll} d_c &= \sqrt{(l_j - l_i)^2 + (a_j - a_i)^2 + (b_j - b_i)^2} \ d_s &= \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2} \ D' &= \sqrt{\left(rac{d_c}{N_c}\right)^2 + \left(rac{d_s}{N_s}\right)^2}. \end{array}$$

如果将两个点之间的距离简单定义为五维欧式空间,则在不同大小的超像素中,距离差距的影响不同,导致聚类的过程中行为不一致,所以需要对距离和颜色差异做标准化,分别计算差距后,除以各自的最大的值,距离标准化的参数 $N_{\rm s}$ 即为采样间隔S,颜色的标准化参数 $N_{\rm c}$ 定义为常数m,实际试验中取[1,40]范围内的值。当m较大时,空间邻近性更为重要,所得到的超像素更加紧凑(超像素面积小,面积周长比大),m较小时,颜色距离权重更大,所得的超像素更加紧密地粘附到图像边界。

加权距离度量方式,使得该方法可以控制结果中对图像边界的粘附性,超像素生成的复杂度等。





M=30 M=5

通过结果可以看出,M值的变化对于超像素生成结果有较大影响,M较大时,超像素面积小,更加紧凑,M 较小时,所得结果更加粘附到图像边界。

SLIC的优点

- 3. 良好的扩展性, 易于处理灰度图像和3D超体素。
- 4. 参数只有K和M, 算法流程清晰, 易实现。