基于密度峰值搜索聚类的超像素分割算法

张志龙[1),2)] 李爱华[2)] 李楚为[1)]

[1)](国防科技大学电子科学学院 长沙 410073)

[2)](蒙特克莱尔州立大学科学与数学学院 新泽西 07043 美国)

摘 要 本文提出了一种新的基于密度峰值搜索聚类的超像素分割算法.首先在图像平面内的局部区域内估计像素的局部密度.其次为每个像素寻找一个距离最近的大密度像素并计算两个属性:距离和归属.之后根据距离和归属将所有像素组织成一个归属关系树,该树反映了像素之间的归属关系.然后选择局部密度和距离较大的像素作为超像素的种子,并标记在归属关系树中.最后在归属关系树中搜索距离每个像素最近的超像素种子为其分配标记,实现超像素分割.该算法有两个优势:超像素分割过程无需迭代优化,计算速度非常快;可以精确控制超像素的数目和大小,使用灵活.与其它 9 种同类算法的对比实验表明:文中算法在边缘召回率、欠分割误差、可达分割精度、计算和存储复杂性方面表现出比较优越的性能.

关键词 超像素;分割;聚类;密度峰值;树

中图法分类号 TP391 **DOI 号** 10.11897/SP.J.1016.2020.00001

Superpixel Segmentation Based on Clustering by Finding Density Peaks

ZHANG Zhi-Long[1),2)] LI Ai-Hua[2)] LI Chu-Wei[1)]

[1)](*School of Electronic Science, National University of Defense Technology, Changsha* 410073)

[2)](*College of Science and Mathematics, Montclair State University, New Jersey* 07043, *USA*)

Abstract Superpixel segmentation has been widely used as a pre-processing procedure in many computer vision applications, such as object tracking, 3D reconstruction, visual saliency estimation, object detection, and medical image segmentation. In this paper, we present a novel superpixel segmentation algorithm, which is inspired by a new clustering method published in Science in 2014. Our algorithm produces superpixels through clustering pixels by searching and finding density peaks. This algorithm consists of five steps. Firstly, we estimate the density for each pixel in a local circular neighborhood in the image plane. Density is formulated as a weighted similarity between the central pixel and all the surrounding ones, while the radius of the neighborhood can be determined experientially by the superpixel number that we desired. Secondly, we search the nearest pixel with a larger density for each pixel and calculate the distance between them. For each pixel, the index of the nearest pixel and the distance to it are two attributes named as ascription and distance, respectively. In the third step, we construct an ascription relation tree and assemble all the pixels into the tree based on their distances and ascriptions. A leaf of the tree represents a pixel. A directed edge in the tree starts from a pixel and arrives at its ascription and is weighted by the corresponding distance. The tree reflects the ascription relationship among all the pixels in the input image. In the fourth step, we select several pixels with large densities and distances as the seeds of superpixels. Then we assign each seed a unique label in the tree. In the final step, by searching the tree, we find the closest superpixel seed for each pixel and assign the label of the

收稿日期:2019-02-13;在线出版日期:2019-09-24.本课题得到国家自然科学基金(61101185)、国防科技重点实验室基金(61425030301)资助.**张志龙**,博士,副研究员,主要研究方向为图像处理、自动目标识别.E-mail:zhangzhilong@nudt.edu.cn.**李爱华**,博士,教授,主要研究领域为应用数学、图论.**李楚为**,硕士研究生,主要研究方向为图像处理、自动目标识别.

seed to it. Our algorithm has many advantages. It is flexible because it can automatically select the seeds and accurately control the number and the size of the superpixels it produced. It is fast because no iterative optimization is involved in the algorithm. Its computational complexity does not rely on the number of superpixels. We compare our algorithm to nine state-of-the-art methods on two benchmark datasets, BSDS300 and BSDS500. The comparison methods are categorized into two main types: 5 classical methods (SLIC, QS, DB, NC, and GB) and 4 latest canonical ones (LSC, ERS, SEEDS, and FLIC). We conducted extensive experiments to confirm the performance of our algorithm. We start with qualitatively examine the superpixels of all the methods. The examination results reveal that the superpixels of our algorithm adhere to the ground-truth edges more accurately than the other methods. To confirm the superior performance of our algorithm, we further evaluate all the methods quantitatively by their scores on four widely used measures: the boundary recall rate, the under-segmentation error, the achievable segmentation accuracy, and the computational complexity. The evaluation results reveal that our algorithm significantly outperforms the 5 classical methods. It also achieves better or similar scores than the 4 latest canonical methods. The runtime of our algorithm does not rise as the number of superpixels increases.

Keywords superpixel; segmentation; clustering; density peaks; tree

1 引 言

超像素的概念最早是在 2003 年由 Ren 和 Malik 提出的[1],目的在于将灰度、颜色、纹理等低级特征相似的像素聚集到一起,从而大幅度减少处理对象的数量,提高后续处理的效率.超像素在计算机视觉领域有着广泛的应用,包括目标跟踪[2-3]、三维重建[4]、显著性检测[5-6]、物体检测[7-8]、医学影像分割[9-11]等.

目前已经有很多超像素分割算法被提出,包括规则化图割算法(Normalized Cut,NC)[12]、简单线性迭代聚类算法(Simple Linear Iterative Clustering,SLIC)[13]、均值漂移算法(Mean Shift,MS)[14]、快速漂移算法(Quick Shift,QS)[15]、分水岭算法(watershed)[16]、涡流像素算法(Turbo Pixel,TP)[17]、图论算法(Graph Based superpixel,GB)[18]、超像素网格算法(Superpixel Lattices,SL)[19]、图切割算法(Graph Cut,GC)[20]、DBSCAN聚类算法(DBSCAN clustering,DB)[21]、线性谱聚类算法(Linear Spectral Clustering,LSC)[22]、流形 SLIC 算法(Manifold-SLIC,MSLIC)[23]、熵率算法(Entropy Rate Superpixel,ERS)[24]、能量驱动采样超像素算法(Superpixels Extracted via Energy-Driven Sampling,SEEDS)[25]、快速 SLIC 算法(Fast Linear Iterative Clustering,FLIC)[26-27]等等.这些方法在超像素分割方面各有优缺点.目前,能否提出一种高质量的实时的超像素算法,在边缘附着能力、紧凑性、规则性、计算复杂性、数目和迭代次数可控性方面具有优良的性能,仍然是一个具有挑战性的问题.针对超像素分割问题,本文提出了一种基于密度峰值聚类的快速超像素分割算法.该算法首次将密度峰值聚类算法[28]用于超像素分割,在性能方面与现有算法相比也有明显的优势.

超像素分割本质上是一个像素聚类问题.新的聚类算法往往会催生新的超像素分割算法.由于图像像素在特征空间中形成的聚类通常具有较为复杂的形状,因此只有那些能够适应复杂聚类形状的聚类算法才能得到较好的聚类结果.最近,文献[28]提出了一种新的聚类思想,认为聚类中心是那些密度较大、同时到其他聚类中心距离较远的样本,由此导出了一种通过搜索密度峰值进行聚类的算法.用 $x_i, i=1,2,\cdots,I$ 表示待聚类的元素,ρ_i 表示其局部密度,δ_i 表示其距离.元素 x_i 的局部密度 ρ_i 定义为

$$\rho_i=\sum_j \chi(d_{ij}-d_c) \tag{1}$$

其中,$\chi(u)=\begin{cases}1, & u<0\\ 0, & \text{其他}\end{cases}$,$d_{ij}=\|x_i-x_j\|_2$ 是元素 x_i 和 x_j 的欧式距离,d_c 是截止距离.因此,ρ_i 是到数据 x_i 的距离小于 d_c 的所有元素的数目.

元素 x_i 的距离 δ_i 定义为该元素到其周围高密度点的距离的最小值,即

$$\delta_i = \min_{j:\rho_j>\rho_i}(d_{ij}) \tag{2}$$

文献[28]以 ρ 和 δ 为坐标轴绘制的数据散布图称为决策图. 在决策图中，聚类中心通常因为密度高且距离大而从大量样本中突显出来. 作者进一步以 $\lambda=\rho\cdot\delta$ 作为指标，选择 λ 较大的样本作为聚类中心. 当聚类中心确定后，聚类的方法就是将剩余的样本归入与其距离最近的高密度元素所属的聚类.

该算法的优点包括：能够适应复杂的聚类形状，能够自动确定聚类中心，无需迭代优化因而计算效率较高. 该算法的不足在于：需要计算所有样本两两之间的距离，因此计算量和存储量较大.

本文将这种新的聚类算法用于解决超像素分割问题，提出了一种新的超像素分割算法. 算法的输入是 CIE Lab 空间中的彩色图像. 图像的每个像素用一个 5 维特征向量 $\boldsymbol{x}=[l,a,b,x,y]^{\mathrm{T}}$ 表示. 该算法的主要创新在于：首先在图像平面中的局部区域内估计像素的局部密度 ρ；其次根据局部密度确定每个像素的距离 δ 和归属 f，建立全体像素的归属关系树 T；然后根据 ρ 和 δ 自动选出各个超像素的种子 x_c，并按顺序标记在归属关系树 T 中；最后通过搜索归属关系树 T 为每个像素分配标记，完成超像素分割. 该算法的优势体现在：超像素分割过程没有任何迭代优化，计算速度较快；可以自动确定超像素的数目，也可以人为控制超像素的大小，使用灵活. 实验结果表明：文中算法在边缘一致性、欠分割误差、可达分割精度、计算和存储效率方面均优于或接近现有主要算法.

本文的贡献主要体现在 3 个方面：(1) 提出了一个新的超像素分割算法，该算法的计算和存储效率有明显优势；(2) 提出一种根据像素的局部密度和距离属性自动选取超像素种子的方法，为灵活控制超像素的数量和大小提供了一种技术手段；(3) 构建了一个体现像素距离和归属关系的树结构，提高了超像素分割的计算效率.

本文第 2 节回顾现有的超像素分割算法；第 3 节详细介绍文中算法的主要步骤；第 4 节通过实验对比文中算法与现有算法的性能；第 5 节是讨论和总结.

2　相关工作

自从 2003 年 Ren 和 Malik 提出超像素的概念以来，已经出现了大约 28 种超像素分割算法，包括：基于密度的算法，基于图的算法，基于聚类的算法，基于 watershed 的算法，基于 path 的算法，基于小波的算法，轮廓演化算法，能量优化算法，等等. 感兴趣的读者可以参考文献[29]. 这里主要回顾与本文算法有关的三类超像素算法：基于密度的算法、基于图的算法和基于聚类的算法.

2.1　基于密度的算法

常见的基于密度的超像素算法有 MS 算法[14]和 QS 算法[15]. 这两种算法都是在一个计算好的密度图中进行模式搜索，收敛到同一个模式的所有像素构成一个超像素. 由这类算法得到的超像素通常形状不规则，大小不一致，而且超像素的数量、大小和紧凑性无法控制. MS 算法的计算复杂性是 $O(n^2)$，QS 算法的计算复杂性是 $O(d\cdot n^2)$，运行速度都比较慢，这里的 n 是像素数目，d 是一个较小的常量.

2.2　基于图的算法

基于图的超像素算法将待分割图像看成一个无向图. 图的顶点是像素，图的边是像素之间的关系，边的权值反映的是颜色相似性或差异性，通过对图的切割(简称图割)实现超像素分割. 不同的图采用了不同的图割算法. NC[12]和 GC[20]采用的是图划分算法. 其中，NC 算法以轮廓和纹理为线索，通过反复迭代的图划分使代价函数达到全局最小值. 该算法产生的超像素非常规则，但是边缘附着性能较差、运行速度较慢、计算复杂度为 $O(n^{3/2})$. GB[18]、ERS[24]和 POISE[30]采用的是自底向上、将像素融合成超像素的策略. 该算法产生的超像素边缘附着性好，但是超像素的形状和大小不规则，超像素数目和紧凑性不能通过外部控制. GB 的计算复杂度为 $O(n\log n)$，ERS 的计算复杂度为 $O(nN^2\log N)$，这里 N 是超像素的数目. 伪布尔优化算法(Pseudo-Boolean optimization，PB)[31]采用去边算法实现图划分.

2.3　基于聚类的算法

基于聚类的算法将超像素看成像素的聚类，因此可以采用很多聚类算法生成超像素. SLIC[13]采用了高效的 k 均值聚类，是很流行的超像素分割算法，该算法的主要缺点是只以像素颜色和位置作为特征，边缘附着性能一般，当超像素较大的时候尤为明显，计算复杂度为 $O(n)$. MSLIC[23]是对 SLIC 的改进，计算复杂度为 $O(n)$，而且运算速度与超像素数目无关. LSC[22]结合了规则化图切割(normalized cut)和 k 均值聚类的优点，提取的超像素质量更高，但是特征计算复杂，导致运行速度较慢，计算复杂性是 $O(n)$. TP[17]通过不断扩大规则分布的种子像素，生成网格状、大小均匀的超像素，但是其边缘附着性能差、运行速度慢，计算复杂性是 $O(n)$. DB[21]采用

DBSCAN 聚类算法进行超像素分割，运行速度较快，超像素紧凑性好，但是边缘附着性能差，计算复杂性是 $O(n)$. SEEDS 算法[25]通过逐像素调整超像素的边缘位置实现能量函数的优化，算法速度较快，但是形状不规则，像素数目难以精确控制，其计算复杂度也没有明确的理论分析.

以上各种算法的计算复杂度见表 1.

表 1 各种算法的计算复杂度

算法	类别	计算复杂度
MS	密度算法	$O(n^2)$
QS	密度算法	$O(d\cdot n^2)$
NC/GC	图算法	$O(n^{3/2})$
GB/POISE	图算法	$O(n\log n)$
ERS	图算法	$O(nN^2\log N)$
SLIC	聚类算法	$O(n)$
MSLIC	聚类算法	$O(n)$
FLIC	聚类算法	$O(n)$
LSC	聚类算法	$O(n)$
TP	聚类算法	$O(n)$
DB	聚类算法	$O(n)$

3 文中算法

文中算法的主要思想是以像素的局部密度作为线索，通过像素聚类实现超像素分割. 具体来说：首先以每个像素的局部密度作为参照，找到距离每个像素最近的大密度点，建立像素的归属关系树；然后选取密度和距离都较大的像素作为超像素的种子，并在归属关系树中将这些种子标记出来；最后通过搜索归属关系树为每个像素赋予超像素标号，实现超像素分割. 算法的流程如图 1 所示.

选择超像素种子的方法受到文献[28]的启发. 我们在这个过程能够灵活地控制超像素的数目或大小，这是文中算法的一个显著的优点. 另外，文中算法不需要迭代优化，因此计算和存储效率都比较高.

下面逐一说明文中算法的各个步骤.

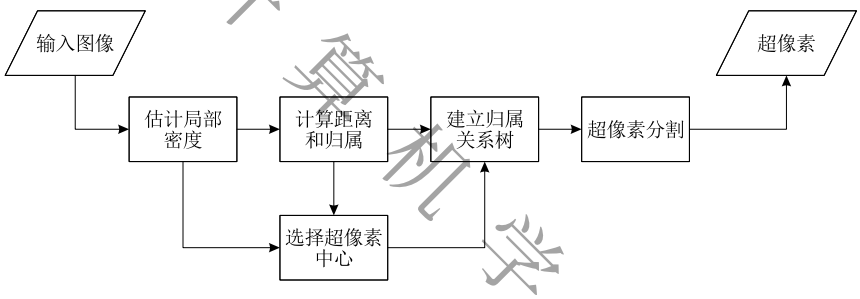


图 1 文中算法的流程图

3.1 估计局部密度

假设输入图像是 CIE Lab 色度空间中的彩色图像，记为 $[l,a,b]_{(x,y)}$，其中 l,a,b 分别代表亮度和两个色度，x,y 分别代表像素的横坐标和纵坐标. 分别用 W 和 H 代表图像的宽度和高度，用 $N=W\times H$ 代表图像中的像素数目，用一个 5 维向量 $\boldsymbol{p}_i=[l_i,a_i,b_i,x_i,y_i]^{\mathrm{T}}$ 描述图像中的像素 i. 分别用 $\|\boldsymbol{p}_i-\boldsymbol{p}_j\|_{xy}$ 和 d_{ij} 表示像素 i 和像素 j 的平面距离和特征距离，其定义如下：

$$\|\boldsymbol{p}_i-\boldsymbol{p}_j\|_{xy}=\sqrt{(x_i-x_j)^2+(y_i-y_j)^2} \tag{3}$$

$$d_{ij}=\sqrt{(l_i-l_j)^2+(a_i-a_j)^2+(b_i-b_j)^2+(x_i-x_j)^2+(y_i-y_j)^2} \tag{4}$$

在图像平面内以像素 i 为中心、半径为 ε_{xy} 的一个圆形区域内估计像素 i 的局部密度，记为 ρ_i. 为了使 ρ_i 能够反映圆形区域中的所有像素在像素 i 周围的聚集程度，定义 ρ_i 如下：

$$\rho_i=\sum_{\|\boldsymbol{p}_i-\boldsymbol{p}_j\|_{xy}\le\varepsilon_{xy}}\frac{1}{1+d_{ij}} \tag{5}$$

由于我们是在半径为 ε_{xy} 的圆形区域内估计像素的密度，因此称 ρ_i 为像素的局部密度. 半径 ε_{xy} 作为算法的主要参数，对局部密度的估计结果和超像素分割结果都有非常重要的影响. 在后续实验中将会看到，ε_{xy} 直接影响初始分割的超像素数量.

从本质上说，超像素是图像平面中一些空间相邻、属性相近的像素的集合，就像图 2 中心的深色圆

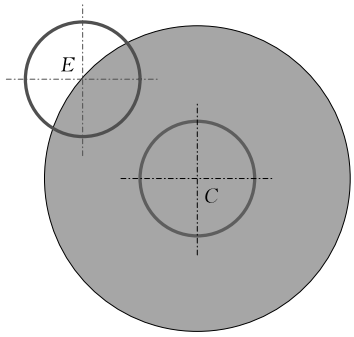
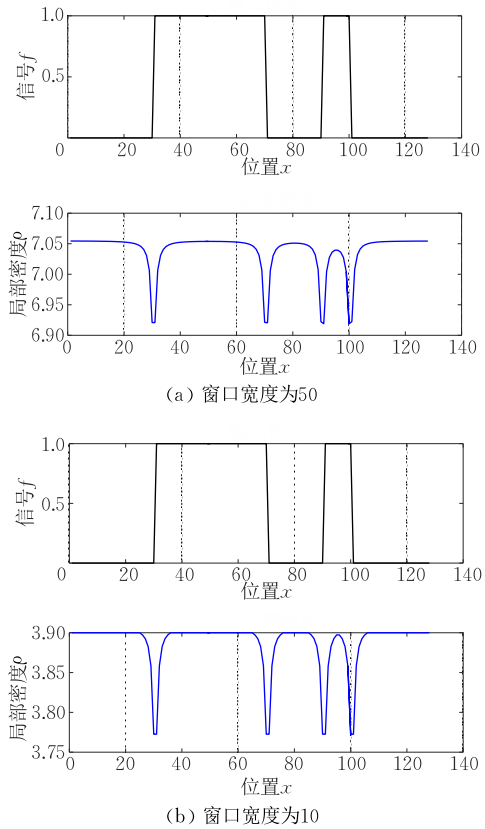


图 2 超像素的示意图

形那样. 如果我们在局部范围(例如图中以 C 和 E 为中心的两个圆形区域)内分析每个像素的密度属性,就会发现处于内部的像素 C,由于其周围聚集了更多属性相近的像素,因此密度较大;而处于外围的像素 E,由于其周围存在大量属性不同的像素,因此密度较小. 这表明:越靠近超像素边缘的位置,密度越低.

下面我们用一维信号中不同长度的区间来模拟不同大小的超像素,分析窗口大小对局部密度估计结果的影响. 图 3 中的信号 f 是一个长度为 128 的一维方波,包含 3 个取值为 0 的区间([0,30],[70,90],[100,128]),2 个取值为 1 的区间([30,70],[90,100]). 图 3(a)中的曲线 ρ 是以宽度为 50 的窗口计算出的局部密度;图 3(b)中的曲线 ρ 是以宽度为 10 的窗口计算出的局部密度. 可以看出:区间边界的密度值较小,区间内部的密度值较大,区间中心的密度值最大;从区间边界到区间中心密度不断增大. 这在区间[90,100]上表现的尤为明显. 另一方面,局部密度与窗口大小有关:当窗口较大时,大区间和小区间的最大密度值有明显的差异,见图 3(a);当窗口较小时,大区间和小区间的最大密度值会非常接近,见图 3(b).



(a) 窗口宽度为50

(b) 窗口宽度为10

图 3　窗口大小对密度估计结果的影响

3.2　计算距离和归属

像素 i 的距离和归属分别表示为 δ_i 和 f_i. 我们将像素 i 的距离 δ_i 定义为:在以像素 i 为中心、半径为 ε_{xy} 的圆形区域内,像素 i 到最近的大密度像素的距离,可以形式化表示为

$$\delta_i = \min_{\substack{j:\rho_j>\rho_i \\ \|\boldsymbol{p}_j-\boldsymbol{p}_i\|_{xy}\le\varepsilon_{xy}}} (d_{ij}) \tag{6}$$

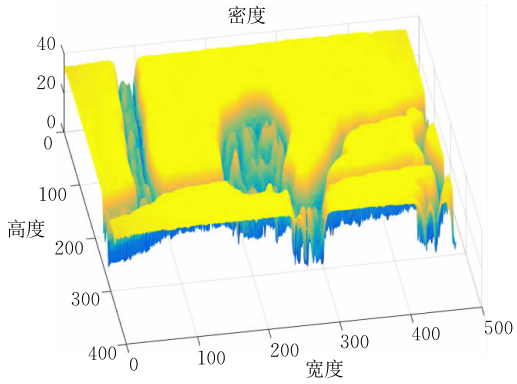
在计算距离 δ_i 的同时,我们也找到了与像素 i 距离最近的大密度像素 f_i:

$$f_i = \arg\min_{\substack{j:\rho_j>\rho_i \\ \|\boldsymbol{p}_j-\boldsymbol{p}_i\|_{xy}\le\varepsilon_{xy}}} (d_{ij}) \tag{7}$$

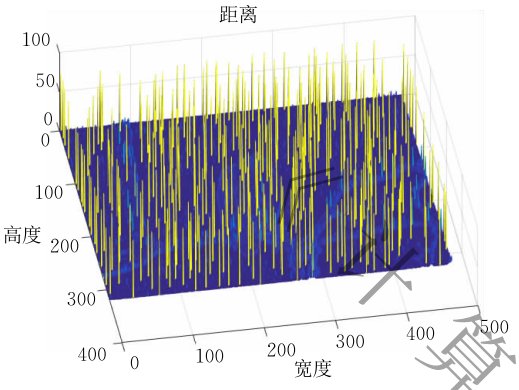
由于像素 f_i 距离像素 i 最近,密度又比像素 i 大. 从聚类的角度来看,如果像素 i 不能成为一个单独的聚类,那么将像素 i 归入像素 f_i 所在的聚类是合适的. 因此,我们称像素 f_i 是像素 i 的归属. 如果像素 i 是半径为 ε_{xy} 的圆形区域内密度最大的像素,我们就设置其距离 $\delta_i=\varepsilon_{xy}$,归属 $f_i=\varphi$. 这里的 φ 是空集符号,表示像素 i 作为局部密度最大值没有归属像素.

图 4 显示了一幅图像的局部密度和按照以上方法得到的距离,其中图 4(a)是原图像,图 4(b)是各像素的局部密度,图 4(c)是各像素的距离,图 4(d)是决策图. 在这个实验中,计算局部密度的圆形区域半径 $\varepsilon_{xy}=8$,产生了 321 个局部密度极大值像素,它们的距离都是 ε_{xy}. 这些像素在图 4(d)的决策图中聚集成右上角的一条直线. 如果不介意超像素的数目,就可以用这些局部密度极大值像素作为超像素的种子进行超像素分割;如果希望超像素的数目更少,则还要做进一步的全局处理,为每个局部密度极大值像素计算距离并寻找归属. 在全局处理之前,一个局部密度极大值像素的距离是 ε_{xy},归属是 φ. 全局处理的方法是在所有局部密度极大值像素中寻找与其距离最近的大密度像素,从而得到其新的距离和归属. 图 4(e)是处理后整个图像的距离值,图 4(f)是处理后所有像素在密度-距离空间中的分布. 显然,经过全局处理后整幅图像中只有一个像素因为密度最大而成为没有归属的像素.

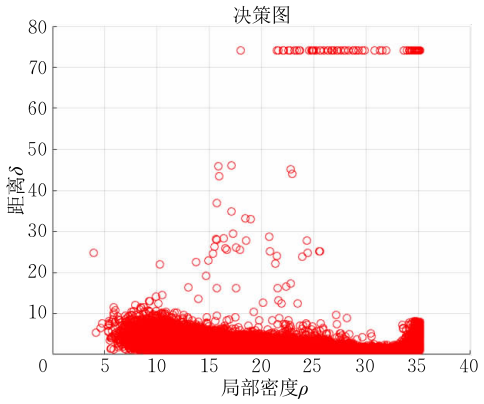
(a) 原图像



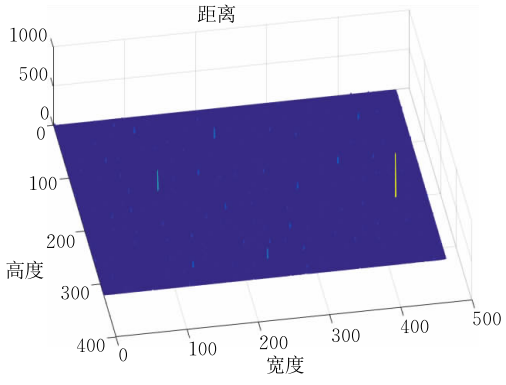
(b) 局部密度



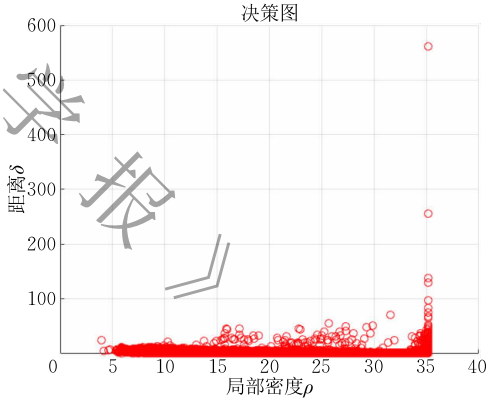
(c) 距离



(d) 决策图



(e) 修正的距离



(f) 修正后的决策图

图 4 图像的局部密度、距离和决策图

3.3 构建归属关系树

根据每个像素的局部密度 ρ_i、距离 δ_i 和归属 f_i，可以将所有像素组织成一棵归属关系树，为后续处理做好数据准备. 构建归属关系树的具体方法是：用一个长度为 δ_i 的有向线段将像素 i 连接到它的归属像素 f_i. 如果一个像素的归属为 φ，这个像素就成为树的根节点. 图 5 示范了归属关系树的构建方法，图中每个圆形代表一个像素，每条有向线段将一个像素连接到它的归属，线段长度为该像素的距离值；圆形 n_4 和 n_9 代表局部密度极大值像素；圆形 n_8 代表局部密度最大的像素，其归属为 φ，是整棵树的根.

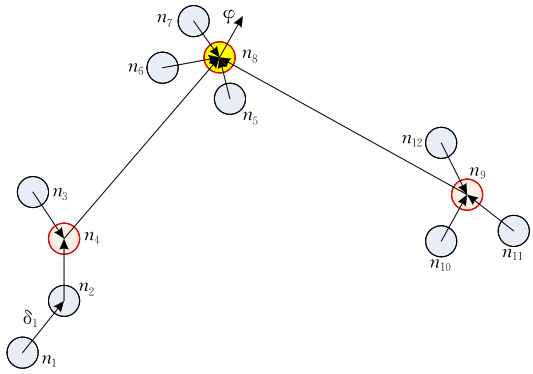


图 5 归属关系树

基于归属关系树可以完成 3 个处理过程：

(1) 可以计算每个节点的子节点数量，记为 son_i，它可以反映超像素的大小. 例如，节点 n_4 的子节点数量 $son_i=3$；

(2) 可以通过将某个节点的归属设置为 φ，得到一棵以该节点为根的子树. 例如，将图 5 中节点 n_4 和 n_9 的归属设置为 φ，可以得到图 6 所示的多棵子树，每棵子树对应一个超像素；

(3) 可以从任意节点开始上溯，找到它所在的子树的根，并从根节点那里获得超像素标号.

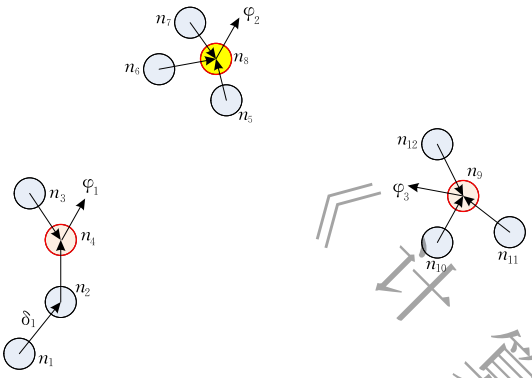


图 6　将归属关系树分割成多棵子树

3.4　选择超像素的种子

如果把超像素看作像素的聚类，那么超像素的种子就是这个聚类的中心，也是超像素中密度最大的像素. 文献[28]认为：聚类中心应该具有较大的密度，同时与其它聚类中心相距较远. 因此，可以根据像素的密度和距离选择超像素的种子. 具体方法是：为每个像素定义索引 λ_i，它是密度 ρ_i 和距离 δ_i 的乘积，即 $\lambda_i=\rho_i\delta_i$，然后根据需要选取索引最大的前 K 个像素作为超像素的种子，记为 $c_1,\cdots,c_K$.

选出超像素的种子之后，在归属关系树中将所有种子像素 c_i 的归属 f_{c_i} 设置为空标志 φ_i，即令 $f_{c_i}=\varphi_i, i=1,\cdots,K$. 这里的 φ_i 是种子像素 c_i 的标号. 这样以来，归属关系树就被分割成 K 个子树，每个子树对应一个超像素，每个种子像素就成为这棵子树的根.

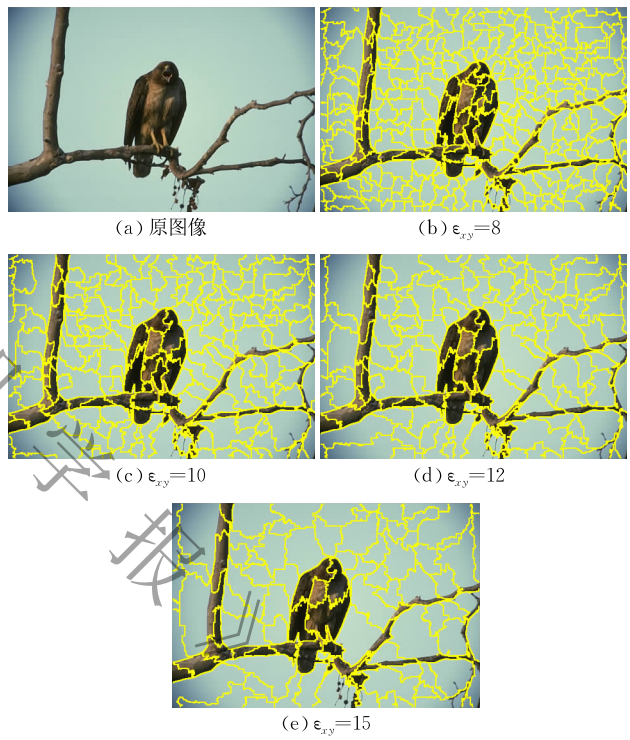
3.5　超像素分割

超像素分割是给同一个超像素中的所有像素赋予相同的标号. 基于标记了的超像素种子的归属关系树，获得超像素标号的方法变得比较简单，即从某一像素出发，沿着归属关系不断上溯到子树的根节点，并以根节点的标号作为该像素的标号. 以图 6 中的像素 n_1 为例，沿着边 $\overrightarrow{n_1n_2}$ 和 $\overrightarrow{n_2n_4}$ 可以到达根节点 n_4，因此 n_1 的标号 $l_1=\varphi_1$.

3.6　窗口大小对超像素分割结果的影响

像素的局部密度是在半径为 ε_{xy} 的圆形区域内计算的. 由于密度直接决定像素的归属，因此窗口大小也会对超像素分割结果产生影响. 我们在第 3.1 节以 1 维信号为例，分析了窗口大小对密度估计结果的影响. 这一节通过实验分析窗口大小对超像素分割结果的影响.

我们分别在 3 幅图像上进行实验，窗口半径 ε_{xy} 依次取为 8、10、12、15 像素，对应的局部区域面积分别为 220、316、440、708 像素. 图 7 给出了对其中一幅图像采用不同窗口大小的超像素分割结果. 图 8 给出了超像素平均大小与窗口大小的关系. 由图 7



(a) 原图像　(b) $\varepsilon_{xy}=8$　(c) $\varepsilon_{xy}=10$　(d) $\varepsilon_{xy}=12$　(e) $\varepsilon_{xy}=15$

图 7　窗口大小对超像素分割结果的影响

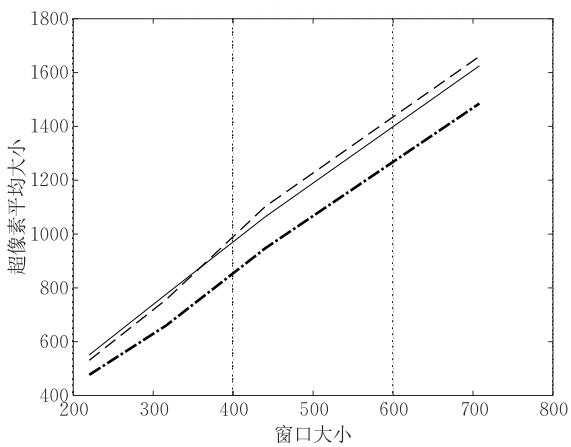


图 8　超像素平均大小与窗口大小的关系

可以看出:随着窗口大小的增加,超像素数目越来越少,超像素的面积越来越大.图 8 则表明:超像素平均大小与窗口大小近似满足线性关系;窗口越大,超像素的面积越大、数量越少.

3.7 超像素数目的选择

在第 3.2 节我们提到,首先在一个半径为 ε_{xy} 的圆形区域内计算各个像素的距离,这样会得到若干密度极大值像素.如果不介意超像素的数目,就可以直接以这些像素作为超像素的种子.如果希望得到的超像素少一些,就需要进一步计算密度极大值像素的距离和归属;然后根据每个像素的索引 $\lambda_i=\rho_i\delta_i$ 由大到小选择一定数目的像素作为超像素的种子.

如果选择的超像素数目少于密度极大值的数目,就意味着有些索引小的超像素要被其它超像素合并.我们通过图 9 的实验来验证超像素合并的合理性.在这个实验中,图 9(a)是原图像,窗口半径为 8 像素,产生了 321 个密度极大值像素,以这些像素作为种子得到的超像素分割结果如图 9(b)所示,图 9(c)至图 9(f)分别是选择索引最大的前 250、200、150、100 个种子得到的超像素.对比这组结果不难发现:随着超像素数目的减少,一些面积小的超像素与附近面积大、特征相似的超像素合并到了一起;合并过程中超像素边缘与物体边缘保持了较高的一致性.这表明:根据距离和密度选择超像素的方法是合理的.

(a) 原图像 (b) 321个超像素

(c) 250个超像素 (d) 200个超像素

(e) 150个超像素 (f) 100个超像素

图 9 超像素数目选择实验

我们进一步结合图 3(a)进行分析,其中区间[90,100]模拟了一个长度为 10、取值为 1 的较小的超像素,它左侧是长度为 20、取值为 0 的超像素,右侧是长度为 28、取值为 0 的超像素.由于左侧的超像素是离它最近、密度又比它大的超像素,因此它首先会被这个超像素合并.这意味着:由数目选择引起的超像素合并倾向于发生在相邻而且面积较小的超像素之间.

3.8 计算和存储复杂性

本文算法进行超像素分割时不需要任何迭代和优化处理.算法中计算量最大的部分是估计每个像素的局部密度,其次是计算每个像素的距离.由于是在半径为 ε_{xy} 的一个圆形区域内估计每个像素的密度和距离,而且距离的计算只针对大密度像素,因此总的计算量为 $\pi\varepsilon_{xy}^2 n$.此后,无论是构建归属关系树、选择超像素种子,还是完成超像素分割,都不需要计算像素之间的距离.因此,本文算法的计算复杂度为 $O(n)$.

至于存储复杂性,本文算法只需要记录每个像素的 3 个属性,即密度、距离、归属,存储量是 $3n$.因此,算法的存储复杂性也是 $O(n)$.

4 与现有算法的对比实验

这一节我们将本文算法与 9 种有代表性的超像素算法进行定性和定量的比较,分别是:基于密度的 QS 算法[15];基于聚类的 SLIC 算法[13]、DB 算法[21]、LSC 算法[22]、SEEDS 算法[25]、FLIC 算法[26-27];基于图的 NC 算法[12]、GB 算法[18]、ERS[24].选取这些算法参与比较的原因是:SLIC 是广泛使用的超像素分割算法;QS 采用了基于密度的模式搜索策略;DB 采用了新的 DBSCAN 聚类算法实现超像素分割;NC 和 GB 则是代表性的图方法;LSC、ERS、SEEDS、FLIC 是最新的权威算法.这些算法要么比较流行,要么性能比较优异,实现思路上也与本文算法存在着一定的联系.

我们选用 Berkeley 数据集 BSDS300 和 BSDS-500[32] 进行性能评测实验.BSDS300 数据集包含 300 幅 321×481 的图像,每幅图像带有由 5 至 10 人手工标注的正确分割结果.BSDS500 在 BSDS300 的基础上扩充了 200 幅新图像.

我们首先通过定性实验,将本文算法与这些算法进行对比;然后从边缘附着性能、计算和存储复杂度两方面进行定量分析.

4.1　超像素分割结果的定性分析

统一将各个算法的超像素数目设置为 300 进行实验. 图 10 给出了各种算法在 BSDS 数据集的若干图像上得到的超像素分割结果. 对比这些结果中

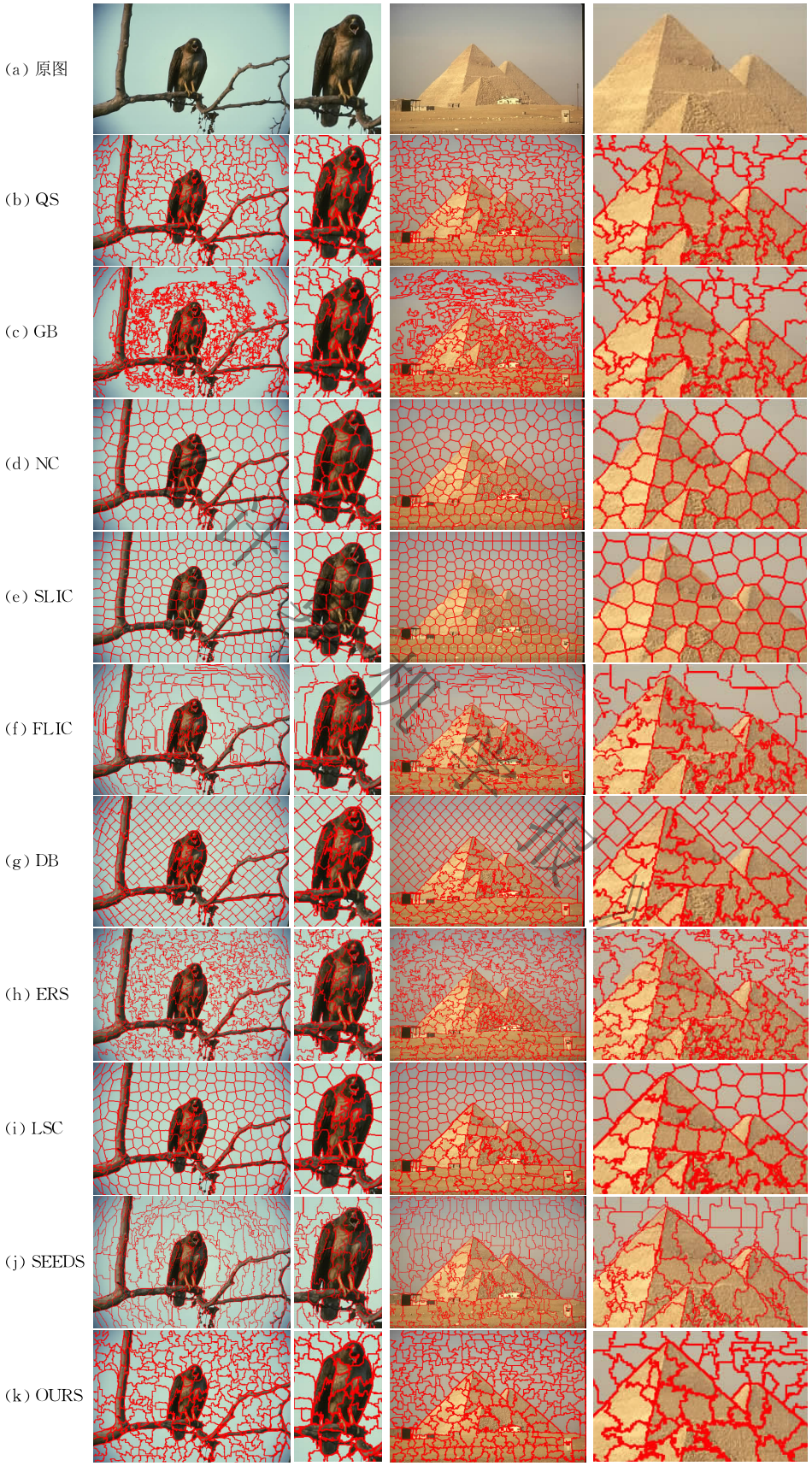


图 10　各种算法的超像素分割结果

超像素的形状以及超像素边缘与物体边缘的一致性，不难看出：本文算法得到的超像素形状不规则但大小比较均匀，而且边缘附着性能较好；ERS、SEEDS、FLIC、QS、GB 算法与本文算法的相似之处在于得到的超像素形状也不规则，其中 ERS、FLIC 和 QS 算法的分割效果与本文算法比较接近，SEEDS 算法得到的超像素大多是不规则的长条形状，而 GB 算法在均匀区域的分割性能不佳；NC、SLIC、DB、LSC 算法得到的超像素具有比较规则的形状，其中 NC、SLIC 算法的边缘附着能力一般（这一点在金字塔边缘处表现的尤为明显），LSC 算法在均匀区域得到的超像素接近规则的圆形，在非均匀区域得到的超像素形状不规则，DB 算法在均匀区域得到的超像素接近规则的菱形，但在非均匀区域得到的超像素很不规则，LSC 和 DB 算法的边缘附着性能较 SLIC 有所提高，但规则性有所降低.

图 11 显示了将各种算法的超像素数目 N 分别设置为 100、200、300 时的分割结果. 对比超像素的边缘附着性能随着超像素数目的变化，可以发现：由于本文算法是通过合并密度较小的超像素以减少超像素数目的，因此超像素的边缘附着能力并没有随着超像素数目的减少而受影响；但是对于超像素形状比较规则的算法（例如，NC、SLIC、DB、LSC）来说，其边缘附着能力则会随着种子点位置的改变和超像素数目的减少而降低.

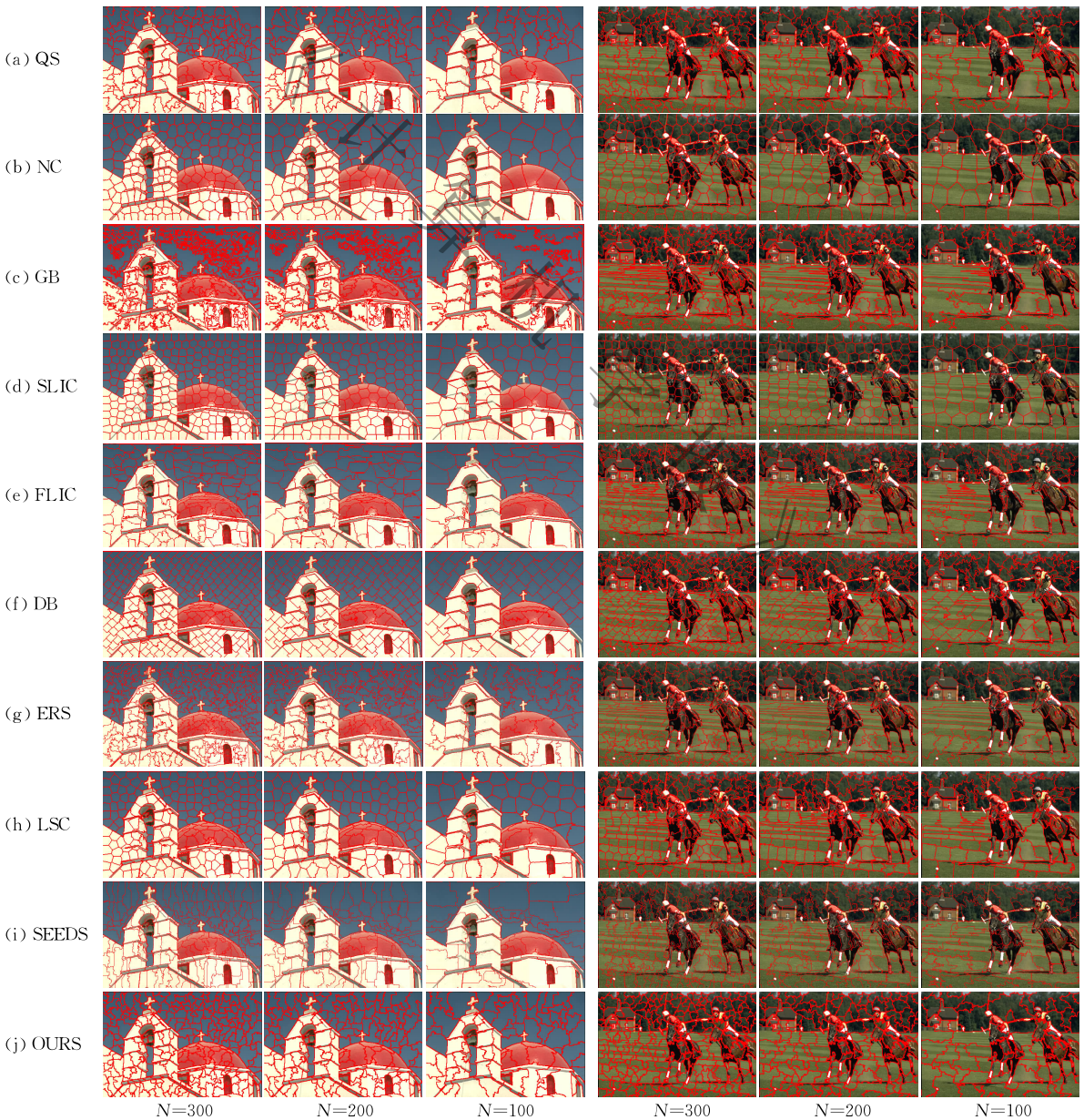


图 11 改变超像素数目得到的分割结果

4.2　边缘附着能力

边缘附着能力反映了超像素边缘与真实物体边缘的一致性，它是评价超像素分割算法好坏的重要标准. 边缘召回率 BR(Boundary Recall)、欠分割误差 UE(Under-segmentation Error)和可达分割精度 ASA(Achievable Segmentation Accuracy)是反映边缘附着能力常用的三项指标[13,29].

边缘召回率 BR 定义为与超像素边缘的距离小于 2 像素的真实边缘在所有真实边缘中所占的比例. 边缘召回率越高，意味着真实边缘丢失的越少.

欠分割误差 UE 是另一项反映超像素边缘附着能力的指标，它反映的是超像素分割 $\{s_j\}$ 中有多少像素“泄露”到区域真值 $\{g_i\}$ 之外. 对于任一超像素 s_j，与它重叠面积最大的手工分割区域记为 $\arg\max_{g_i}|s_j\cap g_i|$，则欠分割误差 UE 定义为

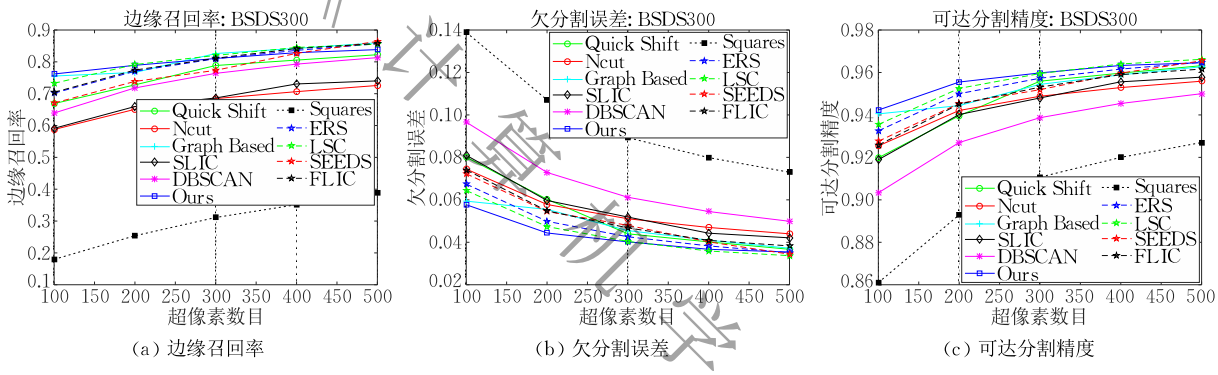
$$UE=\frac{1}{n}\sum_{s_j}[\,|s_j|-\max_{g_i}|s_j\cap g_i|\,] \tag{8}$$

其中 n 表示图像的面积.

可达分割精度 ASA 与欠分割误差 UE 的定义正好相反，反映的是超像素分割 $\{s_j\}$ 中有多少像素在区域真值 $\{g_i\}$ 之内，定义为

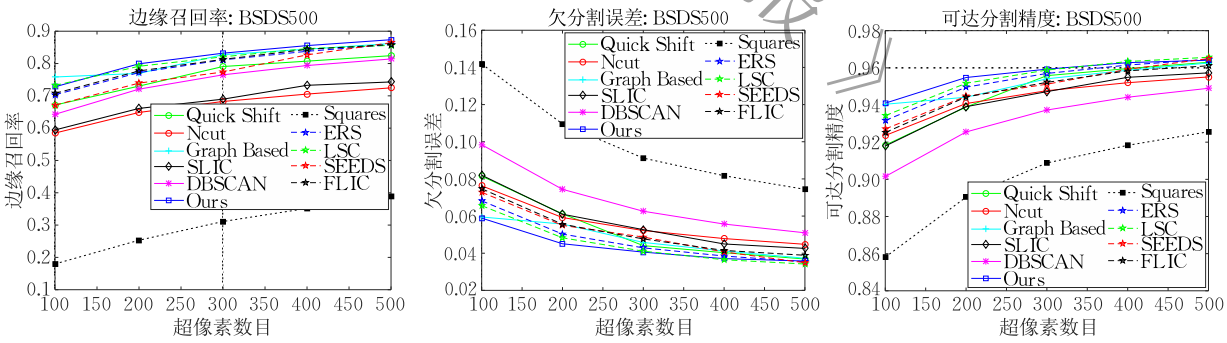
$$ASA=\frac{1}{n}\sum_{s_j}\max_{g_i}|s_j\cap g_i| \tag{9}$$

图 12 和图 13 分别显示了 10 种超像素分割算法在 BSDS300 和 BSDS500 数据集上的性能曲线. 其中图(a)、图(b)、图(c)分别是边缘召回率 BR、欠分割误差 UE、可达分割精度 ASA 与超像素数目的关系. 图中的带方框的曲线“Squares”是基准线，反映的是直接把图像分成大小相同的方形超像素的性能指标.



(a) 边缘召回率　(b) 欠分割误差　(c) 可达分割精度

图 12　各种算法在 BSDS300 上的性能曲线



(a) 边缘召回率　(b) 欠分割误差　(c) 可达分割精度

图 13　各种算法在 BSDS500 上的性能曲线

从各项指标来分析参与比较的 9 种算法，不难发现：性能较好且稳定的算法依次是 LSC、SEEDS、ERS 和 FLIC；QS 和 GB 算法的性能一般且不稳定；SLIC 和 NC 算法的性能一般但比较稳定；DB 算法的边缘召回率性能仅比 SLIC 和 NC 算法，欠分割误差和可达分割精度最差.

与这 9 种算法相比，本文算法的各项性能优于或接近 LSC、SEEDS、ERS、FLIC 等权威算法或新算法，明显优于 SLIC、NC、QS、GB、DB 等传统算法，具有较为明显的性能优势. 我们也注意到，在图 12(b)和图 13(b)中，当超像素数目达到 500 时，本文算法的性能被 LSC 和 SEEDS 略微赶超，但总体上仍然非常接近.

4.3　计算和存储效率

超像素通常用于替代像素以提高后续处理的效率. 因此，超像素分割过程本身应当非常高效.

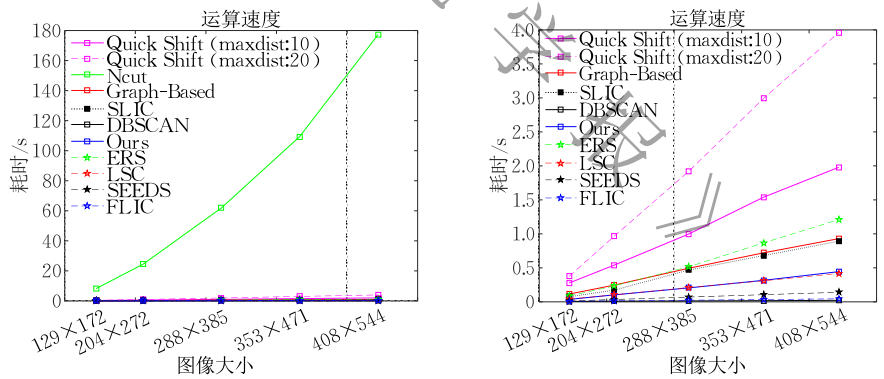
对于各种算法的计算复杂度已有不少理论分析.例如,根据文献[13]的分析:MS 的计算复杂度是 $O(d \cdot T \cdot n^2)$,QS 的计算复杂度是 $O(d \cdot n^2)$,NC 的计算复杂度是 $O(n^{3/2})$,GB 的计算复杂度是 $O(n\log n)$,SLIC 的计算复杂度是 $O(n)$.根据文献[26-27]的分析,FLIC 算法的计算复杂度为 $O(n)$.根据文献[29,33]的分析,DB、LSC 的计算复杂度为 $O(n)$,ERS 的计算复杂度为 $O(nN^2\log N)$.这里的 T 是迭代次数,n 是像素数目,N 是超像素数目.SEEDS 的计算速度较快,但其复杂度尚无严格的理论分析.我们在 3.8 节中分析了本文算法的计算复杂度也是 $O(n)$.因此,本文算法也属于计算复杂度较小的算法之一.

为了分析不同算法的实际运算速度,我们在相同的计算机平台上,运行各种算法分割相同大小的图像,并记录这些算法所耗费的时间.这个实验在一台具有 Intel i7-7700HQ 2.80 GHz 处理器和 8 GB RAM 内存的计算机上进行.我们从作者的网站上下载各种算法的源代码,并尽量在不改变算法默认参数的条件下进行实验.本文算法是用 C++编写的.源代码以 C++实现的算法还有 NC、QS、GB、SLIC、FLIC 算法.而 LSC、SEEDS、ERS、DB 算法只有以 Matlab 实现的源代码.

算法的实现方式对其精度和速度都有重要影响.但是,在这个实验中,我们暂时不考虑算法的实现方式对其计算速度的影响,而主要考察各种算法基于现有实现方式所能达到的计算速度.尽管这种比较方法带有一定的局限性,但是仍然能客观反映各种算法目前的技术状态,可以为读者选用算法提供参考.

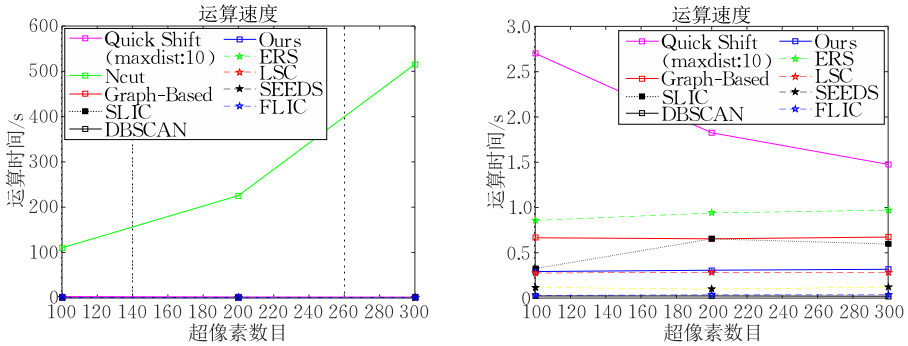
图 14 显示了各种算法的耗时(单位:s)与图像大小的关系.可以看出,各种算法的运算速度由快至慢依次为:DB、FLIC、SEEDS、LSC、本文算法、SLIC、GB、ERS、QS、NC.从图 14(a)可以看出:NC 算法的运算速度最慢,远远落后于其他算法,而且随着图像大小的增加运算速度显著降低.从图 14(b)可以看出:QS 算法的运算速度较慢,而且受算法参数"maxdist"的影响非常明显;本文算法的运算速度比 NC、QS、ERS、GB、SLIC 算法快,与 LSC 的运算速度相当,但是比 SEEDS、FLIC 和 DB 算法慢;DB 算法是运算速度最快的算法.

图 15 显示了各种算法的运算速度与超像素数目的关系.从图 15(a)可以看出:NC 算法的运算速间随着超像素数目的增加而显著增加.从图 15(b)



(a) 所有10种算法 (b) 除NC以外的9种算法

图 14 各种算法的运算速度与图像大小的关系



(a) 所有10种算法 (b) 除NC以外的9种算法

图 15 各种算法的运算速度与超像素数目的关系

度可以看出：QS 算法的运算速度随着超像素数目的增加而降低；SLIC、ERS 算法的运算速度随着超像素数目的增加而略有增加；本文算法、GB、DB、SEEDS、LSC、FLIC 算法的运算速度与超像素数目无关. 从图 15(b)还可以看出：本文算法和 LSC、SEEDS、FLIC、DB 算法都是运算时间小于 0.3 s 的算法.

超像素分割算法的存储效率同样非常重要. 这里用 n 代表图像中的所有像素，那么 SLIC 算法只需要 n 个浮点数存储每个像素到最近的聚类中心的距离就可以了，是存储效率最高的算法；GB 算法需要 $5n$ 个浮点数存储 4 连通形式的边权值和阈值，或 $9n$ 个浮点数存储 8 连通形式的边权值和阈值；本文算法需要 $3n$ 个浮点数存储像素的属性；其他算法的存储复杂性则要高的多.

4.4　实验结果分析

综合上述各种实验结果，可以将本文算法的优点归纳如下：

(1)不需要迭代优化，能够自动选取超像素的种子、精确控制超像素的数目，得到的超像素虽然形状不规则，但大小比较均匀，边缘附着性能较好；

(2)反映边缘附着性能的边缘召回率、欠分割误差、可达分割精度等性能指标优于或接近 LSC、SEEDS、ERS、FLIC 等权威算法或新算法，优于 QS、GB、DB、SLIC、NC 等经典算法；

(3)运算速度比 NC、QS、ERS、GB、SLIC 算法快，与 LSC 算法相当，但是比 SEEDS、FLIC 和 DB 算法慢；

(4)运算速度与超像素数目无关.

实验结果也反映出少数算法在某个单项指标上优于本文算法，这里也做一些分析. 首先，DB 算法的运算速度比本文算法快，但其性能远远不及本文算法. 其次，在超像素数目为 500 时，LSC 和 SEEDS 算法与本文算法的性能非常接近(其中，LSC 的运算速度与本文算法相当，而 SEEDS 的运算速度要比本文算法快). 最后，由图 10 的结果不难发现，SEEDS 算法得到的超像素边缘在物体边界处非常扭曲，LSC 算法也存在类似的不足，而本文算法则没有这种问题.

5　结　论

本文提出了一种基于密度峰值搜索聚类的超像素分割算法. 该算法首先在图像平面中的局部区域内估计像素的局部密度；其次根据局部密度确定每个像素的距离和归属，建立包含全体像素的归属关系树；然后根据局部密度和距离自动选择一定数目的像素作为超像素的种子，并标记在归属关系树中；最后通过搜索归属关系树为每个像素分配标记，实现超像素分割.

该算法的优势在于：超像素分割过程无需迭代优化，计算速度较快；可以自动选择超像素的种子、精确控制超像素的数目和大小，使用灵活. 我们从边缘召回率、欠分割误差、可达分割精度、计算和存储效率等方面将本文算法与 9 种有代表性的超像素分割算法进行对比实验和分析，实验结果表明：文中算法在边缘召回率、欠分割误差、可达分割精度方面优于其他 9 种算法，在运算速度方面优于或接近其他 6 种算法，在存储复杂性方面也有一定的优势，而且超像素数目的增加不影响文中算法的运算速度.

实验中我们也发现：首先，本文算法尽管运算速度较快，但并不是最快的，运算速度最快的算法分别是 DB、FLIC 和 SEEDS 算法；其次，本文算法根据密度决定像素的归属关系并确定超像素分割结果，因此得到的超像素并不像 SLIC 等算法得到的超像素那样规则；再者，由于本文算法仅仅根据像素的密度决定相邻超像素是否应该合并，这可能导致属性不同的超像素合并到一起. 这种现象提示：在合并超像素时应该考虑更多的图像属性. 这些都是进一步完善本文算法的方向.

致　谢　感谢科罗拉多大学斯普林斯分校计算机系助理教授 Jonathan Venture 与我就密度峰值搜索聚类算法在超像素分割方面的应用所进行的有意义的学术讨论！

参　考　文　献

[1] Ren X, Malik J. Learning a classification model for segmentation //Proceedings of the 9th IEEE International Conference on Computer Vision. Washington, USA, 2003: 10-17

[2] Wang S, Lu H, Yang F, Yang M-H. Superpixel tracking// Proceedings of the 13th International Conference on Computer Vision. Barcelona, Spain, 2011: 1323-1330

[3] Yang Fan, Lu Huchuan, Yang Ming-Hsuan. Robust superpixel tracking. IEEE Transactions on Image Processing, 2014, 23(4): 1639-1651

[4] Bodis-Szomoru A, Riemenschneider H, van Gool L. Superpixel meshes for fast edge-preserving surface reconstruction// Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Boston, USA, 2015: 2011-2020

[5] Perazzi F, Krahenbuhl P, Pritch Y, Hornung A. Saliency filters: Contrast based filtering for salient region detection//

Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Rhode Island, USA, 2012: 733-740

[6] He S, Lau R W H, Liu W, et al. SuperCNN: A superpixelwise convolutional neural network for salient object detection. International Journal of Computer Vision, 2015, 115(3): 330-344

[7] Shu G, Dehghan A, Shah M. Improving an object detector and extracting regions using superpixels//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Portland, USA, 2013: 3721-3727

[8] Yan J, Yu Y, Zhu X, et al. Object detection by labeling superpixels//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Boston, USA, 2015: 5107-5116

[9] Andres B, Kothe U, Helmstaedter M, et al. Segmentation of SBFSEM volume data of neural tissue by hierarchical classification//Proceedings of the Annual Symposium of the German Association for Pattern Recognition (DAGM008). Munich, Germany, 2008: 142-152

[10] Lucchi A, Smith K, Achanta R, et al. A fully automated approach to segmentation of irregularly shaped cellular structures in EM images//Proceedings of the International Conference on Medical Image Computing and Computer Assisted Interventions. Beijing, China, 2010: 463-471

[11] Lucchi A, Smith K, Achanta R, et al. Supervoxel-based segmentation of mitochondria in EM image stacks with learned shape features. IEEE Transactions on Medical Imaging, 2012, 31(2): 474-486

[12] Shi J, Malik J. Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2000, 22(8): 888-905

[13] Achanta R, Shaji A, Smith K, et al. SLIC superpixels compared to state-of-the-art superpixel methods. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2012, 34(11): 2274-2282

[14] Comaniciu D, Meer P. Mean Shift: A robust approach toward feature space analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002, 24(5): 603-619

[15] Vedaldi A, Soatto S. Quick Shift and kernel methods for mode seeking//Proceedings of the 10th European Conference on Computer Vision. Marseille, France, 2008: 705-718

[16] Vincent L, Soille P. Watersheds in digital spaces: An efficient algorithm based on immersion simulations. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1991, 13(6): 583-598

[17] Levinshtein A, Stere A, Kutulakos K, et al. TurboPixels: Fast superpixels using geometric flows. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2009, 31(12): 2290-2297

[18] Felzenszwalb P F, Huttenlocher D P. Efficient graph-based image segmentation. International Journal of Computer Vision, 2004, 59(2): 167-181

[19] Moore A P, Prince S J D, Warrell J, et al. Superpixel lattices//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Alaska, USA, 2008: 1-8

[20] Veksler O, Boykov Y, Mehrani P. Superpixels and supervoxels in an energy optimization framework//Proceedings of the European Conference on Computer Vision. Crete, Greece, 2010: 211-224

[21] Shen Jianbing, Hao Xiaopeng, Liang Zhiyuan, et al. Real-time superpixel segmentation by DBSCAN clustering algorithm. IEEE Transactions on Image Processing, 2016, 25(12): 5933-5942

[22] Li Zhengqin, Chen Jiansheng. Superpixel segmentation using linear spectral clustering//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Boston, USA, 2015: 1356-1363

[23] Liu Yong-Jin, Yu Cheng-Chi, Yu Min-Jing, He Ying. Manifold SLIC: A fast method to compute content-sensitive superpixels//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas, USA, 2016: 651-659

[24] Liu M Y, Tuzel O, Ramalingam S, Chellappa R. Entropy rate superpixel segmentation//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Colorado Springs, USA, 2011: 2097-2104

[25] Van den Bergh M, Boix X, Roig G, et al. SEEDS: Superpixels extracted via energy-driven sampling//Proceedings of the European Conference on Computer Vision. Florence, Italy, 2012: 13-26

[26] Zhao Jiaxing, Hou Qibin, Ren Bo, et al. FLIC: Fast linear iterative clustering with active search//Proceedings of the 32nd AAAI Conference on Artificial Intelligence. New Orleans, USA, 2018: 1-8

[27] Zhao Jiaxing, Ren Bo, Hou Qibin, et al. FLIC: Fast linear iterative clustering with active search. Computational Visual Media, 2018, 4(4): 333-348

[28] Rodriguez A, Laio A. Clustering by fast search and find of density peaks. Science, 2014, 344(6191): 1492-1496

[29] Stutz D, Hermans A, Leibe B. Superpixels: An evaluation of the state-of-the-art. Computer Vision and Image Understanding, 2018, 166(1): 1-27

[30] Humayun A, Li F, Rehg J M. The middle child problem: Revisiting parametric min-cut and seeds for object proposals //Proceedings of the International Conference on Computer Vision. Las Condes, Chile, 2015: 1600-1608

[31] Zhang Y, Hartley R, John Mashford, Stewart Burn. Superpixels via Pseudo-Boolean optimization//Proceedings of the International Conference on Computer Vision, Barcelona, Spain, 2011: 1387-1394

[32] Martin D, Fowlkes C, Tal D, Malik J. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics// Proceedings of the IEEE International Conference on Computer Vision. Vancouver, Canada, 2001: 416-423

[33] Wang Murong, Liu Xiabi, Gao Yixuan, et al. Superpixel segmentation: A benchmark. Signal Processing: Image Communication, 2017, (56): 28-39

ZHANG Zhi-Long, Ph. D., associate professor. His research interests include image processing, object recognition, computer vision, and artificial intelligence.

LI Ai-Hua, Ph. D., professor. Her research interests include commutative algebra, number theory, combinatorics, and graph theory.

LI Chu-Wei, master student. His research interests include image processing, object recognition.

Background

Superpixel segmentation is a fundamental problem of image analysis and computer vision. Since the introduction of the first superpixel algorithms around 2009, they have been applied to many significant issues in computer vision, such as tracking, 3D-reconstruction, saliency, object detection and so on. By today, there have been more than 28 superpixel algorithms published. Each algorithm has its advantage and disadvantage for superpixel segmentation, however. It is still very challenging to develop high quality and real-time superpixel algorithm that exhibits the properties including good boundary adherence and low computational complexity.

In this paper, we present a new superpixel algorithm and evaluate it against five state-of-the-art algorithms. The advantages of the algorithm are in two-fold: first, it is fast because no iterative optimization is in the algorithm; second, it is flexible because a user can accurately control the number and size of superpixels it produces. Experiment results review that the algorithm is superior to the five state-of-the-art algorithms in Boundary Recall Rate, Under Segmentation Error, and Complexity of Computation and Memory.

The research in this paper is a part of the National Natural Science Foundation of China (No. 61101185) and the National Key Laboratory Foundation (No. 61425030301). These two foundations are trying to find new methods for detecting tiny objects from real-world images with complex scene background, which is a crucial problem for reconnaissance and surveillance with OE sensors. Moreover, the superpixel algorithm is a significant technology in saliency calculation and object detection. In this unique area, we hold more than 3 Chinese patents, have been awarded two State Scientific and Technological Progress Awards (the First Class Award), and published more than 15 related papers.

The full name of No. 61101185 is "Registration of Infrared and Visual Images Based on Phase Grouping and Mutual Information of Orientation Information."

The full name of No. 61425030301 is "Movable Target Recognition in the Infrared Image Based on Visual Attention Model and Manifold Learning."