光电技术及应用

DOI：10.16818/j.issn1001－5868.2022011305

基于Kmeans++的自适应超像素投点方法

杨志立，张　东*

（武汉大学 物理科学与技术学院 微电子系，武汉 430072）

摘　要： 超像素分割作为目标分割的预处理环节，能够极大地减少后续处理的数据量，对图像分割起着至关重要的作用。在大部分超像素生成算法中，初始种子点的选取都是以规则网格或随机确定，这容易导致欠分割。为了得到良好的初始种子点分布，减少种子点选取引起的欠分割，提出了一种基于Kmeans++的自适应确定超像素种子点方法，并由此改进了简单非迭代聚类算法（Simple Non-Iterative Clustering，SNIC）。实验结果表明，在不耗费大量计算成本的前提下，改进的SNIC算法相比传统算法能够得到更高的边界召回率和更低的欠分割错误率。

关键词： 图像分割；超像素；SLIC；SNIC；Kmeans++

中图分类号： TP391　**文章编号：** 1001－5868(2022)03－0585－07

An Adaptive Initializing Superpixel Seed Points Method Based on Kmeans++

YANG Zhili，ZHANG Dong

（Department of Microelectronics，School of Physics and Technology，Wuhan University，Wuhan 430072，CHN）

Abstract： As a pre-processing step of target segmentation，superpixel can greatly reduce the amount of subsequent data processing，and plays a vital role in image segmentation. In most superpixel algorithms，seed points are sampled on a regular grid or initialized randomly，which easily leads to under-segmentation. In order to obtain a good distribution of seed point and avoid under-segmentation，an adaptively initializing superpixel seeds method based on Kmeans++ is proposed and used to improve the algorithms of SNIC. The experimental results show that the improved SNIC algorithm can get higher boundary recall rate and lower under-segmentation error rate than that of the traditional algorithm without a lot of computational cost.

Key words： image segmentation；superpixel；SLIC；SNIC；Kmeans++

0　引言

超像素分割作为图像分割的预处理步骤，其目的在于将图像中的所有像素点先划分为局部、同质化的、具有一定相同结构的小区域。这些小区域也称作超像素。以超像素为单位处理图像相比以像素为单位处理图像能够大大减少计算量，同时以超像素为单位能够很好地提取区域特征，有利于后续图像处理。超像素的概念一经提出就得到研究人员的关注。

目前超像素算法根据实现过程大致可以分为两大类，基于图论的方法和基于梯度的方法。基于图论的方法是将像素点作为图中的顶点，像素点之间的相似度作为图中边的权值，然后通过求解自定义的函数最小值来实现分割，其方法包括：Ncut[1]，Graph-Based[2]，Superpixel lattice[3]等。基于梯度的方法主要是从粗略的像素初始聚类开始，使用梯度上升方法迭代聚类，直到满足某些收敛条件，最终得到超像素，其方法包括：Mean-Shift[4]，Quick-Shift[5]，Watershed[6]，Turbopixel[7]，SLIC[8]，SNIC[9]等。目前用的最多的超像素算法是由Achanta等提出的SLIC算法，该算法能够得到较为紧凑和均匀的超像素，并且分割结果较为准确。大

收稿日期：2022－01－13.

基金项目：国家重点研发计划项目(2011CB707900).

* 通信作者：张　东　　E-mail：dz-whu@163.com

部分研究也围绕改进 SLIC 算法展开。

关于改进超像素算法的研究有很多，例如：结合均值漂移算法和融合纹理信息的 SLIC 算法[10]更加适合遥感图像的分割；对 SLIC 增加亮度相似性限制条件等改进使其对红外测温图像分割更有优势[11]；利用 SLIC 算法对传统 GrabCut 进行改进从而加速分割[12]；融合阈值分割与 SNIC 对遥感图像进行分割[13]，通过拉普拉斯锐化预处理，对 SLIC 生成的超像素建立无向图等对区域生长算法进行改进[14]，使其受种子点选取影响较小，以解决分割空洞等问题。当前对于超像素算法改进取得了很多成果，但是对于聚类与区域生长的超像素算法初始种子点选取关注则较少。种子点的选取对于超像素具有重要意义，文献[14]描述了区域生长算法依赖于初始种子点选取。而大部分的超像素算法基本都是按照规定网格初始化种子点，或是采用随机投点的方式，由于图像中目标的大小不一致，在设置超像素数目较少时，规定网格化投点与随机投点必然容易产生欠分割现象。所以良好的初始种子点分布对于聚类产生超像素非常有意义。

Kmeans＋＋[15]是 Kmeans 聚类方法的改进，其主要不同在于初始聚类中心选取，以获得更好的初始聚类中心分布。本文基于 Kmeans＋＋聚类方法，针对超像素分割特点，提出了一种自适应确定超像素初始种子点的方法，并用此方法改进了 SNIC 算法。通过对 BSDS500 的数据集进行超像素分割实验，可以发现改进后的 SNIC 算法相比原有方法分割效果更好。

1 SLIC 算法和 SNIC 算法

1.1 SLIC 算法

SLIC 算法的核心与 Kmeans 聚类算法十分相似。首先在图像上均匀网格初始 K 个种子点（也可称作超像素聚类中心），将种子点附近范围的点赋予距离最近的种子点标签，然后不断迭代更新种子点，直至收敛（种子点更新前后位置之差小于某一个阈值）或达到最大迭代次数后停止。SLIC 算法只需要给出超像素数目 K 以及计算距离时颜色距离与位置距离的比重参数 m 即可。其距离计算公式如下：

$$d_c=\sqrt{(l_j-l_i)^2+(a_j-a_i)^2+(b_j-b_i)^2} \quad (1)$$

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

$$D=\sqrt{d_c^2+\left(\frac{d_s}{S}\right)^2 m^2} \quad (3)$$

其中，i 和 j 分别表示一副图像上不同的像素点；l，a，b 则是图像在 CIELAB 颜色空间的对应数据；x，y 分别是图像像素点对应的横纵坐标位置。d_c 与 d_s 分别表示颜色距离和空间距离，m 表示颜色距离与空间距离的比重，D 则为像素点到种子点的距离。值得注意的是，每个种子点的搜索范围是 $2S\times 2S$（$S=\sqrt{N/K}$，N 为图像大小，即像素点个数），而不是 Kmeans 聚类算法中全体的搜索范围，这极大地提高了算法效率。除此之外，SLIC 算法在初始化种子点后会进行一定的扰动，避免投点到噪声点。

1.2 SNIC 算法

为了得到更高的分割效率和较好的分割结果，SNIC 算法对 SLIC 算法进行了改进。首先仍然是在图像上用均匀网格初始 K 个种子点，然后以种子点周围的点（4 邻域或 8 邻域）到种子点的距离，根据距离大小放入优先队列（距离小的放前面），像素点出队时即分配对应的超像素标签，然后再搜索该出队的像素点周围未分配标签的像素点入队，如此反复直到所有的点分配标签。该方法更加类似于区域生长型算法。其区别于 SLIC 的主要特点是（1）仅迭代一次；（2）算法不需要记录每个像素点到最近聚类中心的距离，减少空间存储；（3）在开始的时候就保持强制联通，类似于区域生长算法；（4）种子点实时更新。

值得注意的是，SLIC 算法更新种子点是对像素点分配完标签后，对具有相同标签的像素点求平均从而更新超像素中心，需要多次迭代。SNIC 算法仅迭代一次也是由于种子点实时更新，即在每一个像素点分配标签后就更新种子点位置。在超像素不断生长的过程中，种子点也在不断发生变化。

SNIC 算法将像素记为 $e_i=\{x_i,c_i,k,d_{i,k}\}$，其中 x_i 为像素点 i 的位置，c_i 为像素点 i 的颜色，k 对应入队时附近像素点的标签，$d_{i,k}$ 则是像素点 i 到超像素 k 中心的距离，其伪代码如下：

Initialize K centroids $C[k]=\{x_k,c_k\}$ sampled on a regular grid
Set label $L[:]=0$
For $k\in[1,2,3,\cdots,K]$ do
    Element $e_k=\{x_k,c_k,k,0\}$
    push e_k on priority queue Q
End for
While Q is not empty do
  Pop Q to get e_i

if $L[x_i]$ is 0 then
$L[x_i]$ is k_i
Update centroid $C[k_i]$ online with x_i and c_i
for Each connected neighbor x_j of x_i do
Create element $e_j=\{x_j,c_j,k,d_{j,k_i}\}$
if $L[x_j]$ is 0 then
Push e_j on Q
end if
end for
end if
end while
Return L

2　基于 Kmeans＋＋的自适应超像素投点方法

2.1　Kmeans＋＋初始化聚类中心

Kmeans＋＋算法是 Kmeans 算法的改进，其主要区别在于初始化确定初始聚类中心。Kmeans 算法是随机确定初始聚类中心，而 Kmeans＋＋则是根据当前样本点到已有中心点的距离给出该样本点成为下一个聚类中心的概率，距离越大则概率越大，然后根据概率大小抽取下一个聚类中心，不断重复直至抽取 K 个聚类中心。具体步骤如下：

(1)随机确定第一个聚类中心。

(2)更新每一个样本到已有聚类中心的最小距离。

(3)根据距离大小确定每个样本成为下一个聚类中心的概率。

(4)根据概率大小抽取下一个聚类中心。

(5)重复步骤 2 至 4，直至抽取 K 个聚类中心。

Kmeans＋＋算法初始化确定初始聚类中心的方法很大程度上提高了聚类效果。同时，对于超像素而言，种子点等同于超像素聚类中心，那么将一副图像所有像素点看作为样本点，就可以类比得到自适应确定超像素初始种子点的方法思路，即利用图像中像素点到已有种子点的距离来衡量该像素点成为下一个种子点的概率。

2.2　自适应确定超像素初始种子点方法

Kmeans＋＋初始化确定聚类中心方法能够得到良好的初始种子点分布，但用于超像素自适应确定初始种子点上还需要考虑两个问题：算法复杂度和图像噪声的影响。

Kmeans＋＋每确定下一个聚类中心时，需要计算所有样本点到已有聚类中心的最短距离。假定当前样本点有 N 个，已有距离中心 K 个，那么求下一次聚类中心时就需要计算 $N(N-K)$ 次，应用到超像素中，像素点数目和超像素数目多时，复杂度很高。一方面，可以采用存储最小距离的方式，将样本点到已有聚类中心的最短距离记录下来，每次只需要比较记录其中最短距离和上一个确定聚类中心的距离，如果与上一个确定的聚类中心距离更近则更新记录中的距离，这样求下一个聚类中心的距离只需要计算$(N-K)$次即可。另一方面，对于图像来说，当确定一个新的聚类中心后，那么也仅导致其周围的像素点到已有聚类中心的最短距离发生变化，所以不需要考虑所有像素点，只需要计算当前确定的聚类中心附近像素点的最短距离变化，可以通过缩小更新范围来进一步减少计算量。

同时，图像的噪声应该考虑其中，由于是依靠 Labxy 五维空间距离，如果图像中出现了噪声点，那么很大概率会被选为下一个种子点，这对于后续生成超像素是非常不利的。不过，图像噪声可以采用一定方式滤除，常用的方法就是滤波。常见的滤波器有均值滤波器、高斯滤波器、中值滤波器等，不同的滤波器可以带来不同的效果。并且在对图像滤波时可以进行下采样，然后在下采样的图像中选取种子点，之后对应回原图像，这样处理既能够避免噪声点影响，同时大大减少了计算量。

除此之外，Kmeans＋＋确定聚类中心是根据距离远近按照概率抽取聚类中心，距离越大则概率越大。在确定超像素种子点时可以直接简化为选取距离最大点作为下一个种子点，这样也可以避免随机抽取到噪声点。而且，Kmeans＋＋第一个聚类中心是随机抽取的，用于图像中随机选择第一个种子点将存在一定的缺陷，为了保证第一个种子点选择合理，在图像的梯度图像上选择最小值对应的点作为第一个起始种子点。另外可以发现 Kmeans＋＋算法中确定聚类中心的数目是固定的，迭代停止条件是直至抽取 K 个聚类中心。这里可以通过建立当前像素点到种子点最短距离与初始种子点之间的距离关系来改变迭代停止条件，从而自适应确定种子点数目。

基于上述描述，可以得到自适应确定超像素初始种子点的方法，步骤如下：

(1)将待分割图像进行滤波和下采样得到处理

后的图像。

(2)用处理后的图像做梯度图像,选取梯度最小的点作为第 1 个初始种子点。

(3)计算处理后图像所有种子点到第 1 个初始种子点的距离,选取距离最大对应的像素点作为第 2 个初始种子点。记第 1 个初始种子点和第 2 个初始种子点之间的距离为 D。

(4)更新新确立种子点范围内所有像素点到已有种子点的最短距离,选择距离最大的像素点成为下一个种子点。

(5)不断重复步骤(4),直至得到 K 个种子点或当前最短距离小于一定比例的 D。

(6)将在处理后图像上得到的种子点映射回原图像,得到良好的初始种子点分布。

自适应确定初始种子点方法伪代码如下:

Get small image by filtering and downsampling origin image

The first seed point C_1 is selected as the point with the minimum value on the gradient image of the small image for all pixels i in $R\times R$ region around C_1 do

Calculate D_{iC_1} and store in $D[i]$

end for

The second seed point C_2 is selected as the point $i=\text{argmax}(D[i])$ and $D=D_{iC_1}$, $k=2$

while (condition=$k<K$; $\max(D[i]<m\cdot D)$)

for all pixels i in a $R\times R$ region around C_k do

Calculate D_{iC_k}

if$D_{iC_k}<D[i]$

$D[i]=D_{iC_k}$

end if

Select the point $i=\text{argmax}(D[i])$ as C_{k+1}

$k=k+1$

end for

end while

Mapping seed point C_k to origin image

其中 condition 可以设置为 $k<K$,此时达到设置对应的超像素数目 K 完成投点,还可以设置为 $\max(D[i])<m\cdot D$,此时算法还可以自适应地确定超像素数目,其中 m 的值设置在 0.05～0.1 较为合适。$D[i]$表示像素点 i 到已有种子点的最短距离,D_{iC_k} 表示像素点 i 到种子点 C_k 的距离,计算公式如下:

$$D_{iC_k}=(1-r)\cdot d_c+r\cdot n\cdot d_s \tag{4}$$

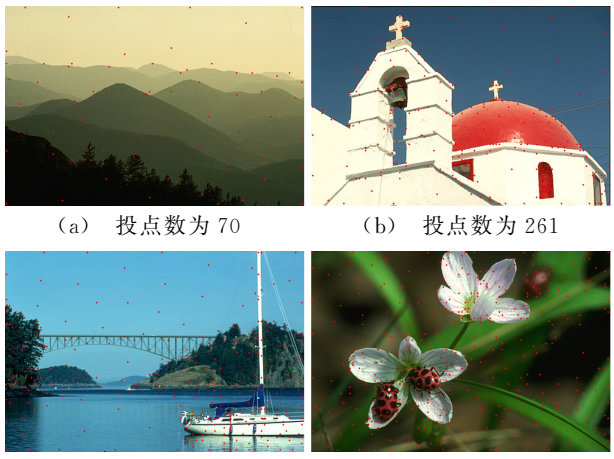
其中,r 式代表颜色距离与位置距离的比值,取 0.4～0.6,$n=255/\sqrt{h^2+w^2}$,255 为 Lab 颜色空间的最远距离,$\sqrt{h^2+w^2}$ 为图像位置空间的最远距离。

3 超像素分割实验

实验分为两个部分,第一部分实验是比较自适应投点与规则网格投点、随机投点的区别,通过对不同图像进行投点来比较结果。第二部分实验是比较使用自适应投点优化后的 SNIC 算法,与传统规则网格投点、随机投点或测地线投点的 SNIC 算法在生成超像素上的差别,分析边界召回率、欠分割错误率、平均 CPU 时间等相关评价参数。

3.1 自适应确定初始种子点实验

为检验方法能够自适应确定超像素个数,算法迭代终止条件设置为 $\max(D[i])<m\cdot D$。图 1 是对 BSDS500 数据库中的图片进行自适应投点的结果。子图标题表示为投点数目(即超像素的数目)。图中红点即为种子点位置。



(a) 投点数为 70　(b) 投点数为 261

(c) 投点数为 303　(d) 投点数为 372

图 1 自适应确定初始种子点结果

通过图 1 结果可以发现,图 1(a)中目标普遍较大,则需要的投点数也较少,随着图像越来越复杂,细小目标越来越多,颜色变化越大,投点的数量也就越多,由此看来,本文提出的算法能够很好地实现投点数量自适应,对于简单图像投点少,在保证分割质量的同时减少计算量;对于复杂的图像投点多,可减少欠分割发生。不仅如此,通过进一步观察投点结果,可以发现种子点具有良好的分布。图 1(c)中天空的区域颜色变化较少,当目标单一时,投点稀疏,

在船及其倒影的区域颜色变化较多；当目标较多时，投点则密集。这样的种子点分布能够很好地解决传统规则网格投点数量少时容易发生欠分割的问题。由此看来，自适应投点能够得到良好的初始种子点分布，可根据图像复杂程度不同，自适应地确定种子点数目与位置。

为进一步比较规则网格投点、随机投点与自适应投点的区别，在相同数量下得到投点结果以及超像素分割结果。为更好地对比超像素个数相同下，各个评价参数的区别，将此时自适应投点的迭代终止条件设置为种子点达到指定数目停止投点。图 2 是三种投点结果比较，以及对应的 SNIC 超像素分割结果，包括分割细节对比图。

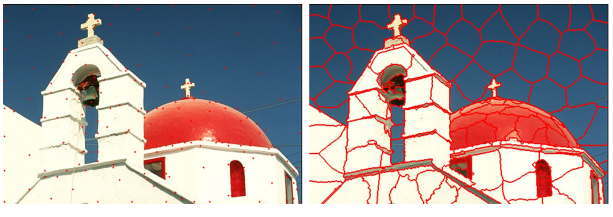
从图 2(a)，(b)中可以发现规则网格投点时，各个种子点之间的间距是固定比例的，在一些细小目标上(例如图中建筑物上的窗户等小目标)没有投点，产生的超像素大小几乎一致。从图 2(c)，(d)中可以发现种子点随机排布，在一定概率上能够在细小目标上进行投点，不过随机投点在大目标区域(例如图中的天空部分区域)可能会投点密集，超像素大小不一致。图 2(e)，(f)展示了自适应投点以及超像素分割结果，可以发现自适应投点在大目标区域的投点稀疏，超像素较大，在小目标区域及附近投点密集，超像素较小，能够很好地分割出细节。通过图 2(g)，(h)，(i)的超像素分割细节对比能够进一步看出，规则网格投点与随机投点均会产生欠分割，而自适应投点产生的超像素则能够很好地避免出现欠分割。



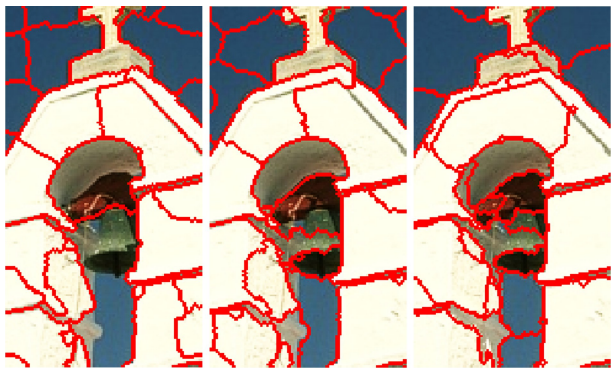
(a)　规则网格投点　　(b)　规则网格投点＋SNIC



(c)　随机投点　　(d)　随机投点＋SNIC



(e)　自适应投点　　(f)　自适应投点＋SNIC



(g) 规则网格投点分割 (h) 随机投点分割 (i) 自适应投点分割

图 2　三种投点结果以及对应的超像素分割结果

3.2　采用自适应投点改进的 SNIC 超像素算法与传统投点算法比较

为了进一步展现自适应投点带来的效果，采用自适应投点替代传统的规则网格投点来改进 SNIC 算法，并且比较改进后得到的 ASNIC(Adaptative Simple Non-Iterative Clustering)与传统投点方法(规则投点、随机投点和测地线投点)生成超像素的评价指标。

实验测试了 BSDS500 数据库测试集的 200 张图片，用上述四种算法分别进行不同数目的超像素分割实验，比较了边界召回率、欠分割错误率，以及算法平均 CPU 时间。边界召回率与欠分割错误率都是衡量超像素分割的重要指标，其中边界召回率衡量真实边缘落在超像素边界的程度。边界召回率高，则说明真实边缘没有被遗漏。欠分割错误率衡量了超像素区域跨越真实边界的程度，欠分割错误率高，说明某一个超像素区域有多个不同的真实区域，此时超像素未将不同区域分割开来。所以良好的超像素分割需要较高的边界召回率和欠分割错误率。由于处理方式不同，目前欠分割错误率有很多不同版本[16-17]，文献[17]对不同版本的所有参数进行了分析，本文选择最受肯定的欠分割错误率。

表 1～3 分别展示了边界召回率、欠分割错误率和平均使用 CPU 时间，为了更直观地比较，将数据绘制为图 3～5。

可以看到，与其他投点方法相比，ASNIC 在边界召回率、欠分割错误率上均有较大提升。SNIC 实际上是基于区域生长的一种算法，没有强制连通。

可以发现，自适应投点对 SNIC 算法带来的提升是十分明显的。ASNIC 的边界召回率明显高于其他方法，而且欠分割错误率明显更低。由此可见，自适应投点对于超像素分割算法是一种更优的确定初始种子点的策略。

最后比较所用 CPU 时间。可以看出，自适应投点算法缩小了种子点的更新范围，大大减少了计算量，与其他方法所耗时间基本没有太大差异。

表 1 边界召回率

种子点数	边界召回率			
	规则网格投点	随机投点	测地线投点	ASNIC
100	0.589 0	0.573 5	0.600 9	0.632 4
200	0.661 4	0.652 9	0.676 5	0.709 5
300	0.701 2	0.697 8	0.715 5	0.752 4
400	0.739 1	0.729 6	0.745 4	0.779 2
500	0.757 7	0.751 3	0.771 0	0.802 9
600	0.776 9	0.769 9	0.787 6	0.821 0
700	0.795 0	0.786 1	0.803 0	0.836 1
800	0.809 7	0.800 2	0.818 8	0.848 8
900	0.823 4	0.814 0	0.832 5	0.858 8
1 000	0.837 1	0.825 5	0.843 5	0.869 2

表 2 欠分割错误率

种子点数	欠分割错误率			
	规则网格投点	随机投点	测地线投点	ASNIC
100	0.164 8	0.179 1	0.163 1	0.149 8
200	0.125 1	0.132 7	0.121 3	0.109 7
300	0.108 2	0.112 1	0.104 6	0.092 9
400	0.095 1	0.100 2	0.093 8	0.083 7
500	0.088 8	0.093 3	0.086 0	0.076 9
600	0.083 6	0.087 6	0.081 2	0.072 3
700	0.079 2	0.082 8	0.077 2	0.068 6
800	0.075 1	0.079 3	0.072 9	0.065 7
900	0.072 1	0.075 8	0.069 9	0.063 5
1 000	0.069 0	0.073 3	0.067 4	0.061 3

表 3 平均使用 CPU 时间

种子点数	平均使用 CPU 时间/s			
	规则网格投点	随机投点	测地线投点	ASNIC
100	0.135 7	0.132 8	0.140 2	0.140 8
200	0.134 4	0.134 7	0.140 9	0.140 6
300	0.134 5	0.137 1	0.141 9	0.143 4
400	0.142 3	0.133 3	0.136 2	0.147 3
500	0.134 5	0.135 9	0.142 3	0.147 6
600	0.142 1	0.136 0	0.145 1	0.145 0
700	0.140 3	0.132 1	0.143 1	0.153 1
800	0.135 8	0.138 0	0.145 1	0.150 3
900	0.139 4	0.136 7	0.149 0	0.153 5
1 000	0.142 7	0.143 8	0.143 0	0.154 9

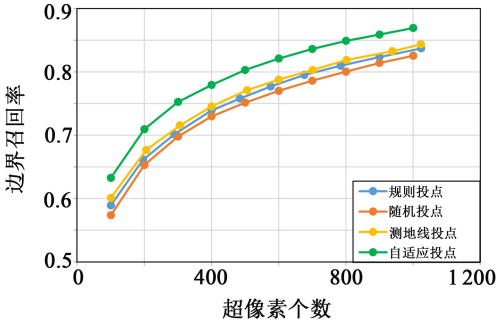


图 3 边界召回率

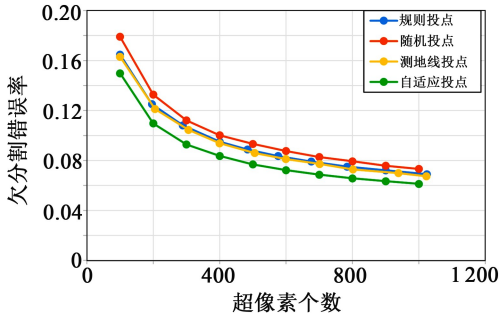


图 4 欠分割错误率

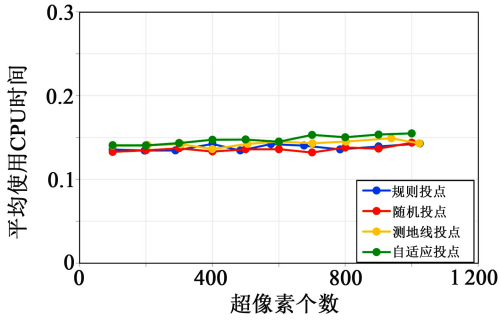


图 5 平均 CPU 时间

总体来说，自适应投点相比于规则网格投点以及随机投点能够得到更好的初始种子点分布。在不耗费大量计算资源的情况下，采用自适应投点的 ASNIC 相比传统投点方法能够有效提升超像素分割质量。

4 结论

本文提出了基于 Kmeans++的自适应确定初始种子点的方法，相比传统规则网格投点与随机投点，能够得到更好的初始种子点分布。同时，该方法可以用于改进基于聚类与区域生长的超像素生成算法，在不消耗大量计算资源下可以有效提高超像素分割质量，减少欠分割。

参考文献：

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

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

[3] Moore A P, Prince S J D, Warrell J, et al. Superpixel lattices [C]// 2008 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR 2008), 2008: 1-8.

[4] Comaniciu, Dorin, Meer, et al. Mean shift: A robust approach toward feature space analysis[J]. IEEE Trans. on Pattern Analysis & Machine Intelligence, 2002, 24(5): 603-619.

[5] Vedaldi A, Soatto S. Quick shift and kernel methods for mode seeking[C]// European Conf. on Computer Vision, 2008: 705-718.

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

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

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

[9] Achanta R, Susstrunk S. Superpixels and polygons using simple non-iterative clustering[C]// IEEE Conf. on Computer Vision & Pattern Recognition, 2017.

[10] 赵宇晴，陈广胜，景维鹏. 对简单线性迭代聚类算法改进的遥感影像超像素分割方法[J]. 东北林业大学学报，2020，48(11)：66-71.
Zhao Y Q, Chen G S, Jing W P. An improved super pixel segmentation method for remote sensing image based on simple linear iterative clustering algorithm[J]. J. of Northeast Forestry University, 2020, 48(11): 66-71.

[11] 王季峥，尹丽菊，咸日常，等. 基于改进 SLIC 算法的电力设备故障区域分割方法[J]. 计算机应用与软件，2021，38(1)：222-226，237.
Wang J Z, Yin L J, Xian R C, et al. Fault area segmentation method for power equipment based on improved SLIC algorithm [J]. Computer Applications and Software, 2021, 38(1): 222-226, 237.

[12] 胡志立，郭　敏. 基于 SLIC 的改进 GrabCut 彩色图像快速分割[J]. 计算机工程与应用，2016，52(2)：186-190，270.
Hu Z L, Guo M. Improved GrabCut color image fast segmentation based on SLIC[J]. Computer Engin. and Application, 2016, 52(2): 186-190, 270.

[13] 孙玮婕，杨　军. 改进的简单非迭代聚类的遥感影像分割研究[J]. 计算机工程与应用，2021，57(13)：185-192.
Sun W J, Yang J. Research on remote sensing image segmentation based on improved simple non iterative clustering [J]. Computer Engineering and Application, 2021, 57(13): 185-192.

[14] 韩纪普，段先华，常　振. 基于 SLIC 和区域生长的目标分割算法[J]. 计算机工程与应用，2021，57(1)：213-218.
Han J P, Duan X H, Chang Z. Target segmentation algorithm based on SLIC and region growing[J]. Computer Engineering and Application, 2021, 57(1): 213-218.

[15] Arthur D, Vassilvitskii S. K-means＋＋: The advantages of careful seeding[C]// Proc. of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA) Society for Industrial and Applied Mathematics, Philadelphia, 2007: 1027-1035.

[16] Neubert P, Protzel P. Superpixel benchmark and comparison [J]. Forum Bildverarbeitung, 2012, 6: 1-12.

[17] Stutz D, Hermans A, Leibe B. Superpixels: An evaluation of the state-of-the-art[J]. Computer Vision & Image Understanding, 2018, 166: 1-27.

作者简介：

杨志立(1995－)，男，江苏省泰州市人，硕士研究生，主要研究方向为医学图像处理；

张　东(1963－)，男，湖北省武汉市人，博士，教授，博士生导师，研究方向为信号处理与微系统。