

二维 Otsu 自适应阈值选取算法的快速实现

汪海洋^{1,2} 潘德炉¹ 夏德深²

摘要 Otsu 自适应阈值算法作为图像阈值分割的经典算法,在图像领域得到了广泛的应用,在此基础上发展起来的二维阈值法因为计算时间长而制约了其应用.针对二维 Otsu 自适应阈值算法计算复杂度高的缺点,通过消除二维自适应阈值算法中的冗余计算,用迭代的方式得到查询表,从而大大提高了二维阈值算法的计算速度.实验结果表明,该算法不仅计算时间远远小于原始二维 Otsu 算法,并且求得的阈值跟原始的算法一样.

关键词 阈值选取, 类间方差, 图像分割, Otsu

中图分类号 TN911.73

A Fast Algorithm for Two-dimensional Otsu Adaptive Threshold Algorithm

WANG Hai-Yang^{1,2} PAN De-Lu¹ XIA De-Shen²

Abstract As a classical image segmentation method, Otsu adaptive threshold algorithm has been applied widely in image processing. But the long-time computation limits the use of the two-dimensional Otsu threshold algorithm based on the Otsu threshold algorithm. This paper gives a fast algorithm for two-dimensional Otsu adaptive threshold algorithm that overcomes the disadvantage of computational complexity. This fast algorithm gets rid of redundant computation and yields a look-up table by iteration. The experimental result has demonstrated that the computational time of the fast method is not only far less than that of the original two-dimensional one, but also yields the same threshold as that of the original method.

Key words Thresholding, between-class scatter, image segmentation, Otsu

1 引言

在图像处理系统中,图像分割是一项最基本的任务,同时也是一项最重要的任务,因为图像分割的结果将影响以后的特征提取和目标识别.通过阈值实现图像分割是一种非常实用的方法,因为它具有速度快,容易实现等优点.它假定目标和背景各自有明显的灰度分布,即灰度直方图具有两个或多个波峰,阈值分割的任务就是选取阈值把这些峰分离开来.图像分割的结果就是把大于阈值的像素赋给背景色,小于阈值的像素赋给目标色,反之亦然.

阈值选取方法^[1~11]可以分为两类,全局的阈值方法和局部的方法.全局的方法是指整幅图像只使用一个阈值,局部的方法是指整幅图像先被分为很多子块,每块使用不同的阈值. Otsu 阈值化方法^[3]是一种全局的自动非参数无监督的阈值选取算法,它是基于类间方差为最大的测度准则.

Sahoo^[4]等人通过试验证明, Otsu 方法是一种很好的阈值化方法.

然而,在实际图像中,由于噪声干扰等因素的影响,使得图像的灰度直方图分布不一定出现明显的波峰和波谷,此时仅利用灰度直方图得到的阈值并不能使图像分割得到满意的结果,往往还会产生严重的分割错误.这是因为像素灰度值仅仅反映了像素灰度级的幅值大小,并没有反映出像素与邻域的空间相关信息^[2].基于这一点,刘和栗^[2]提出了灰度图像的二维 Otsu 自动阈值分割法.该方法建立了既能反映像素点的灰度分布又能体现像素点与其邻域空间相关信息的灰度均值二维直方图,最佳的阈值是在一个二维的类间方差测度准则取最大值时得到的一个二维矢量,并以此二维矢量作为分割门限进行图像分割,从而提高了 Otsu 法的抗噪声能力.但二维直方图的引入,大大增加了计算复杂性,即便在 P4 2.8 G CPU 计算机上,处理一幅灰度级为 256 的灰度图像也要几十秒.这就在很大程度上限制了该算法的应用范围.为了克服这个缺点,本文提出了一种快速实现算法.该算法首先对原来的阈值算法的阈值求取公式进行变形,然后通过迭代的方法来计算二维直方图的零阶距和一阶距并将它们存入查找表,在最后的阈值选取中,通过从查找表中直接取数值代替原来的数学计算.该算法不仅大大加速了计算速度,因为并没有改变原来的阈值求取公式,所以得到的阈值不会改变.

2 二维 Otsu 阈值分割算法

设图像的灰度等级为 L ,那么像素的邻域平均灰度等级为 L ,计算每个像素点邻域的平均灰度,由此形成一个二元组:像素的灰度值 i 和它的邻域灰度的平均值 j ,设二元组 (i, j) 出现的频数 $f_{i,j}$,则可以定义相应的联合密度 $P_{i,j}$:

$$P_{i,j} = f_{i,j}/N \quad i, j = 1, 2, \dots, L \quad (1)$$

式中的 N 为图像总的像素点数, $N = \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} f_{i,j}$, 并且 $\sum_{i=0}^{L-1} \sum_{j=0}^{L-1} P_{i,j} = 1$.

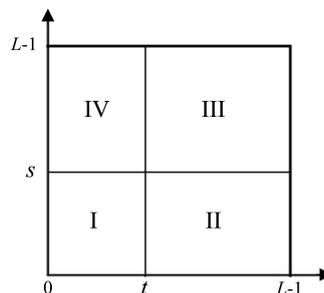


图 1 二维直方图的平面投影示意图的四象限划分
Fig. 1 Four-quadrant partition of the two-dimensional histogram plane

假定二维直方图存在两类 C_0 (I) 和 C_1 (III), 分别代表目标和背景(如图 1 所示), 且具有不同的概率密度函数分布, 设阈值为 (s, t) , 那么两类的概率分别为

$$W_0 = \sum_{i=0}^s \sum_{j=0}^t P_{i,j} = W_0(s, t) \quad (2)$$

收稿日期 2006-3-29 收修改稿日期 2006-6-15
Received March 29, 2006; in revised form June 15, 2006
1. 国家海洋局第二海洋研究所卫星海洋环境动力学国家重点实验室 杭州 310012 2. 南京理工大学计算机系 南京 210094
1. State Key Laboratory of Satellite Ocean Environment Dynamics, Second Institute of Oceanography, State Oceanic Administration, Hangzhou 310012 2. Department of Computer, Nanjing University of Science and Technology, Nanjing 210094
DOI: 10.1360/aas-007-0968

$$W_1 = \sum_{i=s+1}^{L-1} \sum_{j=t+1}^{L-1} P_{i,j} = W_1(s,t) \quad (3)$$

两类对应的均值矢量为

$$\bar{\mathbf{u}}_0 = (u_{0i}, u_{0j})^T = \left(\sum_{i=0}^s \sum_{j=0}^t iP_{i,j}/W_0, \sum_{i=0}^s \sum_{j=0}^t jP_{i,j}/W_0 \right)^T \quad (4)$$

$$\bar{\mathbf{u}}_1 = (u_{1i}, u_{1j})^T = \left(\sum_{i=s+1}^{L-1} \sum_{j=t+1}^{L-1} iP_{i,j}/W_1, \sum_{i=s+1}^{L-1} \sum_{j=t+1}^{L-1} jP_{i,j}/W_1 \right)^T \quad (5)$$

二维直方图上总的均值矢量为

$$\bar{\mathbf{u}}_T = (u_{Ti}, u_{Tj})^T = \left(\sum_{i=0}^{L-1} \sum_{j=0}^{L-1} iP_{i,j}, \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} jP_{i,j} \right)^T \quad (6)$$

假定远离直方图对角线的 $P_{i,j}$ 为零, 即 C_0, C_1 分别落在图 1 中 I、III 象限, II、IV 象限的 $P_{i,j}$ 可以忽略不计, 则可以得到一个类间离散测度 (阈值选取准则)

$$\begin{aligned} S_B(s,t) &= W_0(s,t)(\bar{\mathbf{u}}_0 - \bar{\mathbf{u}}_T)^T(\bar{\mathbf{u}}_0 - \bar{\mathbf{u}}_T) + \\ &W_1(s,t)(\bar{\mathbf{u}}_1 - \bar{\mathbf{u}}_T)^T(\bar{\mathbf{u}}_1 - \bar{\mathbf{u}}_T) + \\ &W_0(\bar{\mathbf{u}}_0 - \bar{\mathbf{u}}_T)^T(\bar{\mathbf{u}}_0 - \bar{\mathbf{u}}_T) + \\ &W_1(\bar{\mathbf{u}}_1 - \bar{\mathbf{u}}_T)^T(\bar{\mathbf{u}}_1 - \bar{\mathbf{u}}_T) = \\ &W_0[(\bar{u}_{0i} - \bar{u}_{Ti})^2 + (\bar{u}_{0j} - \bar{u}_{Tj})^2] + \\ &W_1[(\bar{u}_{1i} - \bar{u}_{Ti})^2 + (\bar{u}_{1j} - \bar{u}_{Tj})^2] \end{aligned} \quad (7)$$

那么最佳的阈值 (s', t') 满足下式

$$S_B(s', t') = \max \{S_B(s, t)\}$$

3 二维 Otsu 自适应阈值选取算法的快速实现

由式 (4) 和式 (5) 可得

$$\begin{aligned} \bar{\mathbf{u}}_0 &= (u_{0i}, u_{0j})^T = \\ &\left(\sum_{i=0}^s \sum_{j=0}^t iP_{i,j}/W_0, \sum_{i=0}^s \sum_{j=0}^t jP_{i,j}/W_0 \right)^T = \\ &\left(\sum_{i=0}^s \sum_{j=0}^t iP_{i,j}, \sum_{i=0}^s \sum_{j=0}^t jP_{i,j} \right)^T / W_0 \end{aligned} \quad (8)$$

$$\begin{aligned} \bar{\mathbf{u}}_1 &= (u_{1i}, u_{1j})^T = \\ &\left(\sum_{i=s+1}^{L-1} \sum_{j=t+1}^{L-1} iP_{i,j}/W_1, \sum_{i=s+1}^{L-1} \sum_{j=t+1}^{L-1} jP_{i,j}/W_1 \right)^T = \\ &\left(\sum_{i=s+1}^{L-1} \sum_{j=t+1}^{L-1} iP_{i,j}, \sum_{i=s+1}^{L-1} \sum_{j=t+1}^{L-1} jP_{i,j} \right)^T / W_1 \end{aligned} \quad (9)$$

$$\text{令 } X_0 = \sum_{i=0}^s \sum_{j=0}^t iP_{i,j}, Y_0 = \sum_{i=0}^s \sum_{j=0}^t jP_{i,j},$$

$$X_1 = \sum_{i=s+1}^{L-1} \sum_{j=t+1}^{L-1} iP_{i,j}, Y_1 = \sum_{i=s+1}^{L-1} \sum_{j=t+1}^{L-1} jP_{i,j}.$$

则

$$\bar{\mathbf{u}}_0 = (u_{0i}, u_{0j})^T = (X_0, Y_0)^T / W_0 \quad (10)$$

$$\bar{\mathbf{u}}_1 = (u_{1i}, u_{1j})^T = (X_1, Y_1)^T / W_1 \quad (11)$$

建立三个查找表

$$P(m,n) = \sum_{i=0}^m \sum_{j=0}^n P_{i,j} \quad (12)$$

$$X(m,n) = \sum_{i=0}^m \sum_{j=0}^n iP_{i,j} \quad (13)$$

$$Y(m,n) = \sum_{i=0}^m \sum_{j=0}^n jP_{i,j} \quad (14)$$

则

$$X_0 = X(s,t) \quad (15)$$

$$Y_0 = Y(s,t) \quad (16)$$

$$X_1 = X(L-1, L-1) - X(L-1, t) - X(s, L-1) + X(s, t) \quad (17)$$

$$Y_1 = Y(L-1, L-1) - Y(L-1, t) - Y(s, L-1) + Y(s, t) \quad (18)$$

$$W_0 = P(s,t) \quad (19)$$

$$W_1 = P(L-1, L-1) - P(L-1, t) - P(s, L-1) + P(s, t) \quad (20)$$

所以我们可以直接通过二维直方图的零阶距 ($P(m, n)$) 和一阶距 ($X(m, n), Y(m, n)$) 的查找表由式 (15) ~ (20) 来求取类内均值, 大大加快了速度. 三个查找表可以通过以下三个迭代公式计算得到

$$P(m,n) = P(m-1, n) + P(m, n-1) - P(m-1, n-1) + P_{mn} \quad (21)$$

其中 $P(0, 0) = P_{00}$.

$$X(m,n) = X(m-1, n) + X(m, n-1) - X(m-1, n-1) + n \times P_{mn} \quad (22)$$

其中 $X(0, 0) = 0 \times P_{00}$.

$$Y(m,n) = Y(m-1, n) + Y(m, n-1) - Y(m-1, n-1) + m \times P_{mn} \quad (23)$$

其中 $Y(0, 0) = 0 \times P_{00}$.

4 算法复杂度和实验结果

假设图像的灰度级为 L , 在原始的二维 Otsu 算法中, 需要对每个 s, t 的组合计算一个离散度矩阵的迹, 然后取离散度矩阵的迹最大时的 s, t 为分割阈值. 在原始算法中需要 s 和 t 的双重循环, 而每次计算离散度矩阵的迹, 需要对 $3st + 3(L-s)(L-t)$ 个点做累加运算, 总累加次数为

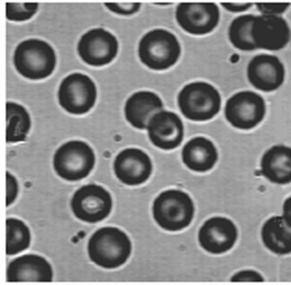
$$A_1 = \sum_{s=0}^{L-1} \sum_{t=0}^{L-1} [3st + 3(L-s)(L-t)] = O(L^4) \quad (24)$$

而本文提出的快速算法, 由于采用了查找表, 每次计算离散度矩阵的迹, 仅需要 9 次累加运算, 而采用式 (21) ~ (23)

建立查找表需要 $(L - 1) \times (L - 1) \times 9$ 次累加运算, 所以总的累加次数为

$$A_2 = 9 \times L \times L + \sum_{s=0}^{L-1} \sum_{t=0}^{L-1} 9 = 18 \times L \times L = 18 \times L^2 = O(L^2) \quad (25)$$

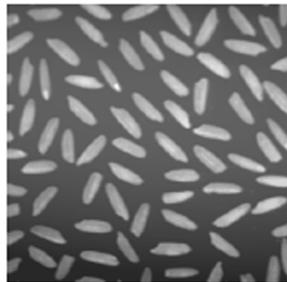
从二者的时间复杂度可以看出, 本文提出的算法加快了计算速度. 另外由于本文并没有改变阈值选取的式 (7), 所以可以保证求得的阈值与原始算法一样, 不会改变图像的分割效果. 下面将给出试验结果, 如图 2 所示.



(a) 血液
(a) Blood



(b) 照相
(b) Camera



(c) 大米
(c) Rice

图 2 试验图

Fig. 2 Experimental images

表 1 是在 P4 2.8 G CPU 512 M 内存台式机 VC6.0 下的运行结果, 可以看出本文的算法比原始的算法的速度提高了 1000 多倍, 而得到的结果是一样的. 表 1 中的计算时间是指求解式 (7) 所用的时间, 并不包括图像的预处理和计算二维直方图的时间. 从算法的实现原理和时间复杂度分析可以看出, 算法的计算时间仅与图像的大小, 灰度分辨率, 计算机的性能有关, 与具体的图像无关. 表 2 给出了在 PM 2.0 G CPU 1 G 内存笔记本 VC6.0 下图 2 的处理结果, 可以得出同样的结论.

表 1 性能比较 (台式电脑)

Table 1 Performance comparison (desktop computer)

图例	原始算法		本文算法	
	阈值	计算时间 (s)	阈值	计算时间 (s)
(a)	(90,147)	49.532	(90,147)	0.047
(b)	(66,140)	49.188	(66,140)	0.032
(c)	(141,89)	49.078	(141,89)	0.031

从上面的分析, 我们可以得到在计算机性能等同和图像

表 2 性能分析 (笔记本电脑)

Table 2 Performance comparison (portable computer)

图例	原始算法		本文算法	
	阈值	计算时间 (s)	阈值	计算时间 (s)
(a)	(90,147)	46.75	(90,147)	0.016
(b)	(66,140)	46.75	(66,140)	0.015
(c)	(141,89)	46.813	(141,89)	0.015

大小等同的条件下, 二维直方图阈值选取算法主要受灰度分辨率 L 的影响. 为了说明本文提出的快速算法的实用性, 在计算机性能等同和图像大小等同但灰度分辨率不同的条件下, 表 3 给出了原始算法和本文算法各自的时间代价; 表 4 给出了本文算法所需的额外空间代价. 从表 3 可以得出当灰度分辨率比较小 (≤ 128) 的时候, 原始算法有一定的实际应用价值, 但当灰度分辨率比较大 (>128) 的时候, 原始算法基本上没有什么实际应用价值, 特别是在有实时性要求的应用中; 本文算法即使当灰度分辨率为 1024 (大部分灰度图像的灰度分辨率为 256), 所用的时间也仅为 0.594, 虽然它消耗的额外空间为 3.145728 MB (表 4), 相对于现在的计算机的内存, 也只是一个很小的开销, 所以本文提出的算法具有很强的实际应用价值.

表 3 时间代价 (秒)

Table 3 Time cost (s)

灰度分辨率	64	128	256	512	1024
原始算法	0.093	1.609	26.375	453.484	7017.86
本文算法	<0.001	0.015	0.031	0.0125	0.594

表 4 空间代价 (字节)

Table 4 Space cost (Byte)

灰度分辨率	64	128	256	512	1024
额外的空间代价	12288	49152	196608	786432	3145728

5 总结

针对二维 Otsu 算法运算时间长, 应用面较窄的缺点, 提出了一种快速算法. 该算法首先对求解阈值的公式进行变形, 然后通过建立查找表的方法消除公式中的冗余计算, 把计算复杂度由 $O(L^4)$ 提高到了 $O(L^2)$, 大大提高了计算速度, 而代价只是比原算法多用了三个查找表的空间.

References

- 1 Sonka M, Hlavac V, Boyle R. *Image Processing: Analysis and Machine Vision*. Beijing: Posts & Telecom Press, 2003
- 2 Liu Jian-Zhuang, Li Wen-Qing. The automatic thresholding of gray-level picture via 2D Otsu method. *Acta Automatica Sinica*, 1993, **19**(1): 101~105 (刘建庄, 栗文清. 灰度图像的二维 Otsu 自动阈值分割法. *自动化学报*, 1993, **19**(1): 101~105)
- 3 Otsu N. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 1979, **9**(1): 919~926
- 4 Sahoo P K. A survey of threshold techniques. *Computer Vision Graphic, Image Process*, 1988, **41**(2): 233~260

- 5 Sezgin M, Sankur B. Survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic Imaging*, 2004, **13**(1): 146~165
- 6 Wu S, Amin A. Automatic thresholding of gray-level using multi-stage approach. In: Proceedings of IEEE International Conference on Document Analysis and Recognition. IEEE, 2003. 1238~1242
- 7 Tsai D M. A fast thresholding selection procedure for multimodal and unimodal histograms. *Pattern Recognition Letters*, 1995, **16**(6): 653~666
- 8 Yin P Y, Chen L H. A new method for multilevel thresholding using symmetry and duality of the histogram. In: Proceedings of 1994 International Symposium on Speech, Image Processing and Neural Networks. IEEE, 1994. **1**: 45~48
- 9 David J E, Albert T A, Chenoweth D L. Two-dimensional entropic segmentation. *Pattern Recognition Letters*, 1999, **20**(3): 329~336
- 10 Albuquerque M P, Esquef I A, Gesualdi M A R. Image thresholding using Tsallis entropy. *Pattern Recognition Letters*, 2004, **25**(9): 1059~1065
- 11 Sahoo P K, Arara G. A thresholding method based on two-dimensional Renyi's entropy. *Pattern Recognition*, 2004, **37**(6): 1149~1161

汪海洋 博士研究生. 主要研究方向为图像处理, 计算机视觉, 模式识别. 本文通信作者. E-mail: why_haiyang@126.com

(WANG Hai-Yang Ph.D. candidate. His research interest covers image processing, computer vision, and pattern recognition. Corresponding author of this paper.)

潘德炉 教授. 主要研究方向为海洋水色遥感, 卫星遥感, 模式识别. E-mail: pandelu@sio.hzcnc.net

(PAN De-Lu Professor. His research interest covers remote sensing of ocean color, satellite remote sense image analyzing, and pattern recognition.)

夏德深 博士, 教授. 主要研究方向为图像处理, 卫星遥感, 模式识别. E-mail: deshens_x@263.net

(XIA De-Shen Ph.D., professor. His research interest covers image processing, satellite remote sense image analyzing, and pattern recognition.)
