基于改进蜻蜓算法的多阈值彩色图像分割

鲍小丽 贾鹤鸣* 郎春博
(东北林业大学机电工程学院 黑龙江 哈尔滨 150040)

摘 要 针对传统的蜻蜓算法在处理图像分割问题时收敛速度慢及分割精度低等问题,提出基于混沌初始化和反向学习策略的蜻蜓算法,并将其有效地应用到多阈值彩色图像分割中。改进后的蜻蜓算法 IDA(Improved Dragonfly Algorithm)不仅可以提高初始种群的质量,而且可以扩大种群的搜索范围,更加稳定快速地收敛于全局最优解。为了验证该算法的有效性,在伯克利图像上进行实验,并与其他元启发式算法进行比较。实验结果表明:该算法在保证稳定性的同时使图像分割的精度得到有效提升,而且可以提高算法的寻优效率,具有较强的工程实用性。

关键词 蜻蜓算法 彩色图像分割 多阈值 最大类间方差 混沌初始化 反向学习

中图分类号 TP391 **文献标志码** A DOI:10.3969/j.issn.1000-386x.2020.06.041

MULTI THRESHOLD COLOR IMAGE SEGMENTATION BASED ON IMPROVED DRAGONFLY ALGORITHM

Bao Xiaoli Jia Heming* Lang Chunbo
(*College of Mechanical and Electrical Engineering*, *Northeast Forestry University*, *Harbin 150040*, *Heilongjiang*, *China*)

Abstract Aiming at the problems of slow convergence and low segmentation accuracy of traditional dragonfly algorithm in image segmentation, this paper proposes an improved dragonfly algorithm based on chaotic initialization and opposition-based learning strategy, and then it can be effectively applied to multi threshold color image segmentation. The improved dragonfly algorithm (IDA) not only enhanced the quality of initial population, but also expanded the search range meanwhile convergences to the global optimum with a better speed and steady. In order to verify the effectiveness of the algorithm, experiments were carried out on Berkeley images and compared with other meta-heuristics. Our algorithm can ensure the stability, improve the accuracy of image segmentation, and improve the efficiency of optimization, which has strong project practicability.

Keywords Dragonfly algorithm Color image segmentation Multi threshold Otsu Chaos initialization Opposition-based learning

0 引 言

图像分割是从图像处理到分析的关键步骤,分割的质量会影响到后续的图像分类、模式识别等工作。图像分割的目的在于将图像中的像素点划分为若干个各具相似特性的类别区域并提取出有用的目标[1]。目前常用的图像分割方法有基于阈值的图像分割、基于区域的图像分割、基于边缘检测的图像分割、基于聚类技术的图像分割等[2]。其中阈值分割方法因实现简单、运算效率高和性能较稳定等优点而被广泛应用于很多领域。例如:对医学 CT 图像的分割在临床肺部疾病判断中具有关键作用[3];对森林火灾图像的分割为检测实时火情奠定坚实基础[4];对农田作物图像的分割为后期的病害防治工作提供参考和依据[5]。由于传统的单阈值分割方法在比较复杂的图像分割上难以获得理想的分割效果,因此一些学者利用迭代的方式将其扩展到多阈值图像分割中。

收稿日期:2019-02-23。黑龙江省研究生教育创新工程项目(JGXM_HLJ_2016014)。**鲍小丽**,硕士生,主研领域:图像处理和智能优化。**贾鹤鸣**,副教授。**郎春博**,硕士生。

多阈值 Otsu 图像分割是以类间方差最大为准则来选取最优阈值组合,但随着阈值个数的增加,其计算量也呈指数增长,直接影响了图像分割的效率。因此,一些学者将阈值选取问题视为以准则函数为目标函数的优化问题,并借鉴仿生算法来提高优化效率。例如:由柳新妮等提出的布谷鸟搜索算法在多阈值图像分割中的应用;王钛等提出的基于离散灰狼算法的多阈值图像分割等。但由于优化算法本身的局限性,在分割图像时仍存在稳定性较差,容易陷入局部最优等问题[6]。

蜻蜓算法(Dragonfly Algorithm,DA)是由 Seyedali Mirjalili 在 2015 年提出的一种新颖的群智能优化算法[7]。该算法的灵感来源于自然界中的蜻蜓静态和动态群体行为。在静态群体行为中蜻蜓会分成几个子群体在不同区域中捕食昆虫,其特征为局部移动和飞行路径的突变,这有利于进行局部开发;在动态群体行为中的蜻蜓会聚集成一个大的群体并向着统一的方向飞行,这有利于进行全局搜索。蜻蜓通过分离、结队、聚集、觅食和避敌这五种行为来更新当前所在位置。虽然蜻蜓算法对于优化问题表现出良好的性能,但仍存在求解精度较差、早熟收敛等问题[8]。

本文提出一种改进蜻蜓算法(Improved Dragonfly Algorithm,IDA)并将其应用到多阈值图像分割领域。为解决标准 DA 算法中全局搜索能力较差且后期局部开发受限的问题,在保证算法运行效率的同时提高图像分割的精度,进行了以下两部分改进:1) 混沌初始化蜻蜓的初始位置,提高初始蜻蜓种群的质量;2) 引入反向学习策略增加可选蜻蜓的数量,并择优选取蜻蜓,在增大随机性的同时提高蜻蜓种群的进化速度。实验选取 6 幅伯克利图像,对比 PSO、SCA、BA、HSO 与 ALO 五种优化算法。结果表明,IDA 提高了蜻蜓种群的收敛速度,而且算法的求解精度得到有效提高,其 PSNR、FSIM、SSIM、最优适应度函数值及运行时间等指标均优于 5 种对比算法,为后续的图像处理工作奠定了良好的基础。

1　多阈值 Otsu 图像分割

基于最大类间方差的 Otsu 算法是由日本学者大津提出的一种自适应的阈值选择方法,其基本原理是根据图像的灰度直方图,以目标和背景之间的类间方差最大为准则来确定图像的分割阈值[9]。

对于一幅图像(大小为 $M \times N$),灰度级为 L,各个灰度值为 0 到 $L-1$,灰度值为 i 的像素总数为 n_i,像素总数可以表示为:

$$N = \sum_{i=0}^{L-1} n_i \tag{1}$$

任意像素灰度值为 i 出现的概率为:

$$p_i = \frac{n_i}{N} \tag{2}$$

设阈值 t 将整幅图像分为前景和背景两部分,则类间方差 ${\sigma_B}^2$ 是 t 的函数:

$${\sigma_B}^2(t) = P_0 \times (m_0 - m_G)^2 + P_1 \times (m_1 - m_G)^2 \tag{3}$$

式中:$P_0 = \sum_{i=0}^{t} p_i$;$P_1 = \sum_{i=t+1}^{L-1} p_i$;$m_0 = \frac{1}{P_0}\sum_{i=0}^{t} ip_i$;$m_1 = \frac{1}{P_1}\sum_{i=t+1}^{L-1} ip_i$;$m_G = \sum_{i=0}^{L-1} ip_i$。$P_0$ 和 P_1 分别表示像素点被分到前景和背景两类中的概率;m_0 和 m_1 分别表示前景和背景的平均灰度;m_G 表示整幅图像的平均灰度;使 ${\sigma_B}^2(t)$ 达到最大时的 t^* 即为最优阈值。

将其推广到多阈值中,假设要将图像分成 n 部分,则阈值可以表示为 $T = (T_1, T_2, \cdots, T_{n-1})$。对于这 n 个部分来说,每部分总像素出现概率分别是 P_0,$P_1, \cdots, P_{n-1}$。其中:

$$P_k = \sum_{i=T_k+1}^{T_{k+1}} p_i \tag{4}$$

$$\mu_k = \frac{1}{P_k}\sum_{i=T_k+1}^{T_{k+1}} ip_i \tag{5}$$

图像的灰度均值为:

$$\mu = \sum_{i=0}^{L-1} ip_i \tag{6}$$

类间方差为:

$${\sigma_B}^2(T) = \sum_{i=0}^{n-1} P_i(\mu_i - \mu)^2 \tag{7}$$

图像的类间方差越大,分割效果越好,所以使 ${\sigma_B}^2(T)$ 达到最大的阈值组合即为所求的最优阈值。Otsu 方法的本质是采用穷举法选取最佳阈值组合,其不足是随着阈值个数的增加,计算量会以指数形式增长,为提高算法的运行效率,现采用基于混沌初始化和反向学习策略的改进蜻蜓算法对选取多阈值的过程加以优化。

2　蜻蜓算法

为了使蜻蜓算法更快地收敛于全局最优,其搜索半径会随迭代次数的增加而线性递增,直到整个蜻蜓群体全部相邻。

利用 DA 算法求解优化问题的思路为:在 N 维空间中初始化产生个体数量为 m 的蜻蜓种群,种群中第 $i(i=1,2,\cdots,m)$ 个蜻蜓的位置表示 $X_i = (X_{i1}, X_{i2}, \cdots,$

X_{iN}),通过计算蜻蜓种群对应的适应度函数值$f(x)$并加以比较,选择当前适应度值最大的蜻蜓位置作为食物源的位置,适应度值最小的蜻蜓位置作为天敌的位置,然后根据蜻蜓的5种行为更新当前个体所在位置。具体公式如下:

1)分离行为:避免蜻蜓和相邻个体之间发生碰撞。

$$S_i = \sum_{j=1}^{W} (X - X_j) \tag{8}$$

式中:X是当前蜻蜓所在位置;X_j是第j个相邻蜻蜓所在位置;W是邻近蜻蜓的个数。

2)结队行为:相邻个体之间趋于相同的速度。

$$A_i = \frac{\sum_{j=1}^{W} V_j}{W} \tag{9}$$

式中:V_j是第j个相邻蜻蜓的速度。

3)聚集行为:蜻蜓倾向于向相邻个体的中心聚拢。

$$C_i = \frac{\sum_{j=1}^{W} X_j}{W} - X \tag{10}$$

4)觅食行为:食物对蜻蜓的吸引力。

$$F_i = X^+ - X \tag{11}$$

式中:X^+是食物源所在位置。

5)避敌行为:蜻蜓对天敌的排斥力。

$$E_i = X^- + X \tag{12}$$

式中:X^-是天敌所在位置。

步长向量指示蜻蜓的步长和运动方向,其定义如下:

$$\Delta X_{t+1} = (sS_i + aA_i + cC_i + fF_i + eE_i + \omega \Delta X_t) \tag{13}$$

式中:s、a、c、f、e分别对应于5种行为的权重;ω是惯性权重;t是当前迭代次数。

当两个蜻蜓之间的欧氏距离小于搜索半径r,则认为两个蜻蜓是相邻的;反之则不相邻,此时蜻蜓通过随机游走的方式围绕搜索空间飞行。

搜索半径r的定义如下:

$$r = \Delta b/4 + (\Delta b \times (t/max\_iteration) \times 2) \tag{14}$$

式中:$\Delta b = ub - lb$,ub和lb分别表示搜索半径的上界和下界;$max_iteration$为最大迭代次数。

蜻蜓位置向量的计算如下:

当有邻近蜻蜓时:$X_{t+1} = X_t + \Delta X_{t+1}$ (15)

当无邻近蜻蜓时:$X_{t+1} = X_t + Levy(d) \times X_t$ (16)

式中:t是当前迭代次数;d表示位置向量的维度。

$$Levy(d) = 0.01 \times \frac{r_1 \times \sigma}{|r_2|^{\frac{1}{\beta}}} \tag{17}$$

式中:r_1和r_2是[0,1]内的随机数;β为常数,这里取值为1.5。

$$\sigma = \left(\frac{\Gamma(1+\beta) \times \sin\left(\frac{\beta\pi}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right) \times \beta \times 2^{\left(\frac{\beta-1}{2}\right)}} \right)^{\frac{1}{\beta}} \tag{18}$$

3 改进蜻蜓算法

3.1 混沌初始化

种群初始化对DA算法的求解精度和收敛速度起着至关重要的作用。标准DA算法初始化时一般是采取随机初始化的方法来确定蜻蜓的位置和移动步长,存在的弊端是当蜻蜓远离食物源时会影响收敛速度。混沌运动具有规律性、随机性和遍历性等特点,这个过程不收敛但有界,并对外部参数和初始值极其敏感,所以基于混沌的种群初始化可以在一定范围内按自身规律不重复地遍历搜索空间,使算法在求解精度和收敛速度方面得到显著的提升[10]。这里选用Logistic映射方程进行种群初始化操作,其表达式如下:

$$cx_{n+1} = \mu cx_n(1 - cx_n) \tag{19}$$

式中:$\mu \in (0,4]$,$cx_n \in (0,1)$,μ越大混沌性越高,$\mu = 4$时系统完全处于混沌状态。

$$x(i,j) = x_{\min} + rand(i,j) \times (x_{\max} - x_{\min}) \tag{20}$$

混沌初始化种群的基本原理:首先给定一个cx初值,通过式(19)Logistic映射方程产生初始化所需要的混沌变量,然后利用混沌变量取代标准初始化方程中的随机数,即式(20)中的$rand(i,j)$,直到遍历完所有蜻蜓个体的每一维度为止。

3.2 反向学习策略

反向学习是由Tizhoosh在2005年提出的一种优化学习策略,同时根据概率定理证明每个随机产生的候选解有50%的概率比它的反向解更靠近最优解[11]。本文应用反向学习策略的主要思想是:在DA算法迭代寻优过程中设置一个反向概率p,每次迭代执行完蜻蜓种群更新后,产生一个0到1的随机数r_3,若$r_3 > p$则结合反向学习策略优化种群。在搜索空间中通过当前个体产生反向个体,并从两者中选择较优秀的个体进入下一代,从而吸引种群向最优个体逼近,在增大随机性的同时降低算法复杂度,公式如下:

$$x_{ij}^* = k(a_j + b_j) - x_{ij} \tag{21}$$

式中:x_{ij}^*为种群中第i个蜻蜓在第j维度上的反向解;a_j和b_j分别为第j维度的最小值和最大值;参数k决定反向学习类型。

3.3 算法描述

混沌初始化策略能够提高初始蜻蜓种群的质量和

种群分布的随机性,反向学习策略能够增加蜻蜓种群多样性,通过比较选取其中适应度较好的个体,可以增强算法的寻优能力,提高种群进化速度和算法的运行效率,其流程如图 1 所示。

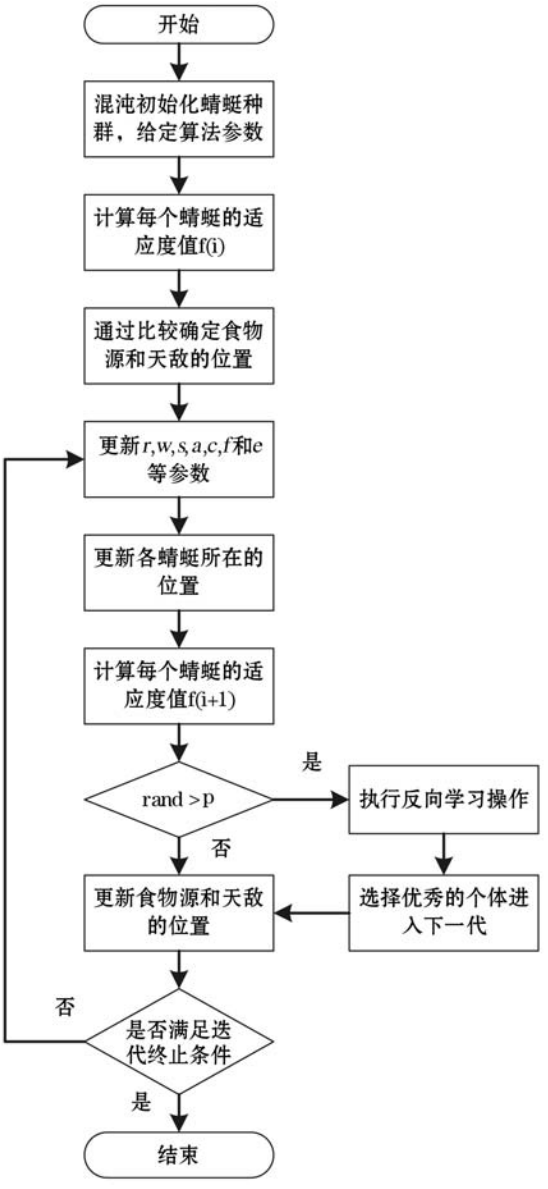

图 1　改进蜻蜓算法流程图

改进蜻蜓算法(IDA)的基本步骤如下:

Step 1　随机生成种群规模为 m 初始种群,利用式(19)和式(20)混沌初始化每个粒子的位置,预先设定算法执行所需参数。

Step 2　计算出每个蜻蜓的适应度值 $f(i)$,从中选出最优个体的位置定义为食物源的位置,其适应度值为 f_{food};选出最差的个体的位置定义为天敌的位置,其适应度值为 f_{enemy}。

Step 3　更新参数,其中包括搜索半径 r,惯性权重 w,分离度权重 s,对齐度权重 a,凝聚度权重 c,食物吸引度权重 f 和天敌驱散度权重 e。

Step 4　计算步长向量,当有邻近蜻蜓时根据式(15)更新蜻蜓的位置;当无邻近蜻蜓时根据式(16)更新蜻蜓的位置,同时将其限制在最大范围[0,L - 1]内。

Step 5　将更新后的各蜻蜓的位置代入适应度函数得到新的适应度值 $f(i+1)$。

Step 6　若 $rand > p$,则根据式(21)执行反向学习操作,并在当前个体与反向个体中择优选出进入下一代的个体。

Step 7　将个体的适应度值分别与最优值 f_{food} 和最差值 f_{enemy} 比较,若优于 f_{food} 则将其更新食物的位置,若差于 f_{enemy} 则更新天敌的位置。

Step 8　判断是否满足终止迭代的条件,若满足则退出循环,否则转向 Step3。

4　伯克利经典图像分割实验

4.1　实验背景

多阈值的图像分割可以看成一种单目标优化问题,结合群智能优化算法的寻优过程实际上是求解 Otsu算法多阈值适应度函数的最优阈值组合。实验选取 6 幅伯克利经典图像进行分析研究,并与 SCA、PSO、BA、HSO 和 ALO 算法进行比较。本文通过峰值信噪比(PSNR)、特征相似性(FSIM)、结构相似性(SSIM)、适应度函数值及运行时间这 5 种指标来评估 IDA 算法与其他 5 种算法的性能差异。

峰值信噪比(PSNR)是基于对应像素点间的误差来衡量图像失真程度的指标,PSNR 的值越大,图像的失真程度越小[12]。其定义如下:

$$PSNR = 10\lg\left(\frac{255^2}{MSE}\right) \tag{22}$$

式中:MSE 表示原图像与分割图像的均方误差。

特征相似性(FSIM)是一种基于图像的相位一致性和与图像的空域梯度特征来评价图像质量的指标[13]。其定义如下:

$$FSIM = \frac{\sum_{x\in\Omega} S_L(x) \times PC_m(x)}{\sum_{x\in\Omega} PC_m(x)} \tag{23}$$

式中:Ω 表示图像的整个空域;$S_L(x)$ 用来评价图像的相似性;$PC_m(x)$ 表示相位相似性。

$$PC_m(x) = \max(PC_1(x), PC_2(x)) \tag{24}$$

式中：$PC_1(x)$和$PC_2(x)$分别表示参考图像和被评测图像的相位一致性。

$$S_L(x) = [S_{PC}(x)]^{\alpha} \cdot [S_G(x)]^{\beta} \tag{25}$$

式中：

$$S_{PC}(x) = \frac{2PC_1(x) \times PC_2(x) + T_1}{PC_1^{\ 2}(x) \times PC_2^{\ 2}(x) + T_1} \tag{26}$$

$$S_G(x) = \frac{2G_1(x) \times G_2(x) + T_2}{G_1^2(x) \times G_2^2(x) + T_2} \tag{27}$$

其中：$S_{PC}(x)$表示图像的特征相似性；$S_G(x)$表示图像的梯度相似性；$G_1(x)$和$G_2(x)$分别表示参考图像和被评测图像的梯度幅值、α、β、T_1和T_2均为常量。

结构相似性(SSIM)是利用图像的亮度，对比度和结构信息的幂乘积作为测度来客观评价图像的质量，其范围为0到1，其值越大，说明分割后的图像与原图像越相似[14]。其定义如下：

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{28}$$

式中：μ_x和μ_y分别为原图像和分割图像平均灰度；σ_x^2和σ_y^2分别为两幅的方差；σ_{xy}为原图像和分割图像的协方差；$c_1 = 6.502\ 5$，$c_2 = 58.522\ 5$。

算法的实验环境为 Windows10-64bit，1.6 GHz 处理器和8 GB 内存，编程环境为 MATLAB 2014b。为了客观对比 IDA 算法与其他5种算法的综合性能，充分考虑算法的随机性对实验结果的影响，实验时的各参数设置如下：粒子数目 $n = 30$，最大迭代次数$t_{\max} = 500$，各算法特性参数设置如表1所示。

表1 各算法参数设置

算法	参数	取值
IDA	β	1.5
	p	0.5
	μ	4
SCA[15]	r_2	$[0,2\pi]$
	r_3	$[0,2]$
	r_4	$[0,1]$
PSO[16]	ω	$[0.4,0.9]$
	$v_{\max}$	25.5
	c_1	2
	c_2	2
ALO[17]	w	$[2,6]$
	PAR	0.3
HSO[18]	$HCMR$	0.95
	BW	25.5
BA[19]	β	$[0,1]$

4.2 实验设计

为了验证本文优化算法在处理多阈值图像分割问题时的优势，在伯克利图像库中随机选取6幅标准图像，灰度图像如图2所示。以类间方差函数作为每种算法的适应度函数，同时利用优化算法对6幅图像多阈值分割时的阈值选取过程加以优化。本次实验对每幅图像分别进行4个、6个、8个和10个这4种类型的多阈值图像分割。

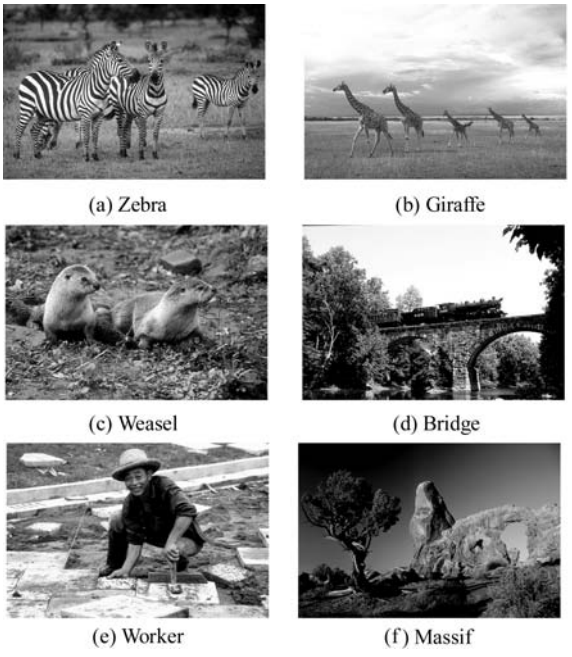

图2 伯克利灰度图像

5 实验结果与分析

5.1 图像分割精准度

本文选用PSNR、FSIM、SSIM和最优适应度值4个指标对各算法图像分割的精准度进行定量分析，数据结果如表2－表5所示。首先分析峰值信噪比 PSNR 指标，它是基于图像的灰度值来衡量图像失真程度的指标，PSNR 的值越大，图像的失真程度越低。当分割阈值个数为4或6时，各种优化算法的 PSNR 的值相差较小，但 IDA 算法仍具有一定优势。当阈值分割个数为8或10时，对于 Worker 和 Massif 等这类复杂的图像，IDA 算法得到了很有竞争力的结果。例如：对 Massif 图像进行10阈值分割时 IDA、PSO、SCA、BA、HSO 和 ALO 对应的取值分别为 28.693 9、26.366 9、27.388 1、26.363 9、27.020 2，直观地表现出基于 IDA 算法分割的图像与其他5种算法相比失真程度小。这说明 IDA 算法不仅能够解决多维函数极值求解的问题，而且在图像分割领域也有很强的工程实用性。

表 2　各算法 PSNR 指标

图像	K	PSNR					
		IDA	PSO	SCA	BA	HSO	ALO
Zebra	4	17.727 8	17.668 5	17.283 3	17.684 9	17.655 4	17.727 8
	6	20.133 1	20.127 6	20.128 6	20.123 9	19.852 4	20.042 2
	8	21.983 6	21.969 1	20.946 0	21.540 4	21.953 3	21.983 1
	10	24.062 5	24.054 0	23.991 4	23.925 4	23.911 2	23.749 5
Giraffe	4	15.241 2	15.225 2	14.829 5	15.114 5	14.868 0	15.241 2
	6	18.695 5	18.695 3	18.226 3	18.289 5	18.688 5	18.666 0
	8	21.040 7	20.807 2	20.962 4	20.788 1	20.531 7	20.987 7
	10	29.623 7	26.821 3	27.208 9	25.278 4	29.087 9	23.445 4
Weasel	4	17.948 1	17.948 1	17.848 3	17.885 9	17.942 3	17.948 1
	6	20.951 7	20.877 2	20.710 4	20.697 8	20.937 7	20.951 7
	8	23.258 5	22.690 0	23.006 1	22.961 1	22.235 2	23.162 9
	10	27.803 2	26.690 8	27.671 6	25.794 7	24.362 2	26.584 0
Bridge	4	27.803 2	26.690 8	27.671 6	25.794 7	24.362 2	26.584 0
	6	27.220 4	27.174 2	27.194 7	26.912 3	26.973 1	27.193 7
	8	29.382 7	29.380 2	28.850 1	27.438 8	29.130 8	29.365 0
	10	31.263 9	30.915 9	30.761 0	29.617 4	31.202 3	31.213 2
Worker	4	20.159 6	20.119 0	19.622 1	20.061 4	20.441 5	20.159 6
	6	22.593 6	22.247 8	22.171 7	22.031 1	22.267 2	22.411 3
	8	24.879 3	23.829 2	24.848 8	24.482 9	24.634 5	23.948 1
	10	27.027 2	24.980 7	25.329 2	25.340 2	26.055 4	26.481 8
Massif	4	19.595 9	19.595 9	19.453 1	19.454 9	19.413 0	19.595 9
	6	23.276 3	23.235 2	23.064 8	22.863 1	22.914 8	23.076 8
	8	26.106 3	25.067 2	25.054 3	25.079 6	25.699 9	25.555 0
	10	28.693 9	26.366 9	27.388 1	26.363 9	27.020 2	27.611 1

表 3　各算法 FSIM 指标

图像	K	PSNR					
		IDA	PSO	SCA	BA	HSO	ALO
Zebra	4	0.756 7	0.756 3	0.751 4	0.756 4	0.756 6	0.756 7
	6	0.817 6	0.817 1	0.814 3	0.817 0	0.814 3	0.816 3
	8	0.859 9	0.856 0	0.851 6	0.848 4	0.857 1	0.859 4
	10	0.893 0	0.892 7	0.888 8	0.867 1	0.885 0	0.881 3
Giraffe	4	0.667 5	0.666 6	0.643 6	0.656 1	0.664 4	0.667 5
	6	0.717 0	0.716 3	0.694 1	0.709 7	0.713 7	0.708 7
	8	0.771 0	0.766 6	0.756 5	0.736 1	0.759 0	0.764 8
	10	0.869 9	0.848 5	0.824 5	0.800 1	0.866 0	0.825 5
Weasel	4	0.751 2	0.751 2	0.748 4	0.750 3	0.751 0	0.751 2
	6	0.832 6	0.831 6	0.816 6	0.829 1	0.832 0	0.832 6
	8	0.879 4	0.871 1	0.874 3	0.874 9	0.864 6	0.877 6
	10	0.926 1	0.915 4	0.912 1	0.894 5	0.896 8	0.913 5
Bridge	4	0.846 6	0.845 2	0.845 0	0.844 6	0.838 1	0.846 6
	6	0.913 8	0.913 6	0.916 6	0.910 3	0.907 9	0.913 7
	8	0.939 2	0.936 1	0.938 4	0.932 9	0.937 1	0.935 2
	10	0.956 7	0.951 3	0.953 5	0.933 7	0.954 0	0.950 3
Worker	4	0.782 2	0.782 1	0.777 6	0.782 3	0.783 0	0.782 2
	6	0.860 1	0.847 6	0.839 3	0.854 5	0.856 3	0.857 8
	8	0.909 3	0.896 4	0.899 9	0.880 8	0.898 1	0.897 8
	10	0.939 4	0.918 8	0.925 4	0.906 8	0.921 8	0.925 4
Massif	4	0.785 5	0.785 5	0.778 3	0.770 8	0.770 1	0.785 5
	6	0.847 2	0.841 1	0.837 2	0.841 0	0.841 2	0.842 2
	8	0.881 7	0.876 1	0.886 0	0.869 4	0.876 5	0.879 2
	10	0.925 4	0.891 5	0.905 3	0.884 4	0.892 1	0.904 1

表 4　各算法 SSIM 指标

图像	K	PSNR					
		IDA	PSO	SCA	BA	HSO	ALO
Zebra	4	0.617 8	0.616 6	0.603 4	0.616 0	0.616 2	0.617 8
	6	0.730 2	0.730 1	0.729 9	0.729 6	0.722 5	0.728 6
	8	0.801 1	0.795 9	0.800 6	0.782 9	0.792 7	0.801 0
	10	0.853 0	0.851 9	0.849 2	0.818 6	0.832 6	0.830 6
Giraffe	4	0.570 2	0.569 5	0.566 4	0.565 5	0.565 1	0.570 2
	6	0.677 8	0.677 3	0.670 0	0.669 1	0.676 0	0.670 0
	8	0.749 0	0.747 0	0.732 8	0.731 4	0.732 1	0.742 9
	10	0.849 8	0.833 4	0.807 5	0.775 2	0.846 9	0.808 4
Weasel	4	0.645 0	0.645 0	0.637 0	0.644 1	0.644 1	0.645 0
	6	0.770 0	0.767 8	0.757 7	0.760 6	0.768 0	0.770 0
	8	0.838 4	0.824 6	0.833 9	0.832 1	0.811 7	0.837 3
	10	0.902 1	0.888 0	0.892 1	0.867 3	0.864 4	0.889 5
Bridge	4	0.808 0	0.807 9	0.801 6	0.801 7	0.800 8	0.808 0
	6	0.882 7	0.882 1	0.880 3	0.882 0	0.874 9	0.882 5
	8	0.908 6	0.900 9	0.900 9	0.917 0	0.903 6	0.905 0
	10	0.930 0	0.915 1	0.926 0	0.917 3	0.929 8	0.921 9
Worker	4	0.677 0	0.676 3	0.669 4	0.674 4	0.672 5	0.677 0
	6	0.772 1	0.758 5	0.746 7	0.765 8	0.766 9	0.770 0
	8	0.857 9	0.820 0	0.833 4	0.810 0	0.824 6	0.822 1
	10	0.892 3	0.850 8	0.856 5	0.841 9	0.876 8	0.871 8
Massif	4	0.675 7	0.675 7	0.651 7	0.647 3	0.647 9	0.675 7
	6	0.759 9	0.751 0	0.759 1	0.754 9	0.753 9	0.758 0
	8	0.838 5	0.817 7	0.833 5	0.805 5	0.817 4	0.831 0
	10	0.875 6	0.833 4	0.873 6	0.838 6	0.841 9	0.862 4

表 5　各算法最优适应度函数值

图像	K	PSNR					
		IDA	PSO	SCA	BA	HSO	ALO
Zebra	4	1 632.934 8	1 632.932 5	1 629.374 2	1 632.883 2	1 632.510 1	1 632.934 8
	6	1 679.622 4	1 679.621 7	1 663.494 3	1 675.738 4	1 678.915 6	1 678.616 5
	8	1 699.653 9	1 696.916 2	1 677.678 3	1 694.351 1	1 697.241 8	1 699.647 5
	10	1 709.754 7	1 709.561 3	1 691.533 6	1 689.693 3	1 704.867 3	1 709.731 5
Giraffe	4	4 259.930 3	4 259.923 3	4 254.303 8	4 258.706 2	4 259.074 4	4 259.930 3
	6	4 305.247 5	4 305.203 5	4 266.091 8	4 288.424 2	4 300.447 6	4 305.242 2
	8	4 323.172 6	4 322.357 8	4 300.084 4	4 310.882 1	4 319.484 1	4 320.771 2
	10	4 328.932 9	4 327.115 3	4 308.538 5	4 325.145 8	4 327.363 7	4 327.214 1
Weasel	4	1 319.949 1	1 319.949 1	1 315.330 3	1 318.981 1	1319.591 6	1 319.949 1
	6	1 369.021 1	1 368.996 9	1 357.148 4	1 366.784 1	1 367.994 5	1 369.021 1
	8	1 390.030 5	1 387.400 5	1 374.322 4	1 389.262 2	1 387.409 5	1 389.181 4
	10	1 399.400 9	1 393.736 2	1 384.490 8	1 384.950 9	1 396.923 4	1 399.022 9
Bridge	4	8 287.049 2	8 286.998 1	8 281.841 7	8 286.150 4	8 286.328 3	8 287.049 0
	6	8 364.721 5	8 364.354 5	8 358.147 2	8 359.714 9	8 362.533 8	8 364.710 0
	8	8 393.577 5	8 392.261 4	8 370.830 2	8 371.781 1	8 391.932 3	8 393.457 5
	10	8 408.015 3	8 407.144 3	8 385.345 3	8 378.978 7	8 404.031 2	8 407.747 7
Worker	4	3 591.468 5	3 591.467 5	3 588.285 4	3 591.164 3	3 590.548 6	3 591.468 5
	6	3 667.041 2	3 666.993 6	3 644.316 7	3 653.045 9	3 666.276 1	3 667.002 4
	8	3 697.186 1	3 697.056 3	3 676.040 8	3 677.342 5	3 690.626 1	3 696.193 8
	10	3 712.164 6	3 697.056 3	3 676.040 8	3 677.342 5	3 690.626 1	3 696.193 8
Massif	4	2 424.570 8	2 424.570 8	2 421.220 3	2 424.513 9	2 424.294 5	2 424.570 8
	6	2 487.008 4	2 482.0920	2 478.411 6	2 485.504 5	2 483.861 7	2 487.006 4
	8	2 512.305 3	2 510.934 4	2 483.706 8	2 501.175 2	2 507.558 3	2 512.023 6
	10	2 525.609 4	2 519.899	2 501.622 5	2 509.817 3	252 0.138 9	2 525.232 7

其次对特征相似性 FSIM 指标进行分析,它是基于图像的相位一致性特征与图像的空域梯度特征来评价图像的质量的一种指标。从表 3 中的数据可以看出,基于 IDA 算法的多阈值图像分割得到了较高的 FSIM 值,随着阈值个数的增加,IDA 算法的优越性越来越显著。例如,在对 Worker 这幅图像进行 4 个、6 个、8 个和 10 个这 4 种类型的分割时,IDA 算法得到的 FSIM 值分别为 0.782 2、0.860 1、0.909 3、0.939 4,得到的值越来越接近于 1,其他算法与 IDA 算法的差距也与越来越明显,表现出 IDA 算法寻优精度高的优点,可以得到较好的分割阈值组合,从而获得高质量的多阈值彩色分割图像。

然后对结构相似性 SSIM 进行分析,它是一种利用亮度、对比度和结构信息的幂乘积作为测度来评价图像的指标。从表 4 中可以看出,对同一幅待分割的图像,本文算法得到了最高值。同时,随着分割阈值个数的增加,SSIM 的值在不断增大,各算法可以更加准确地分割出图像中的主要目标,而且包含更多信息,从视觉上更加接近原图像,其中 IDA 算法的效果最为显著。例如:IDA、PSO、SCA、BA、HSO 和 ALO 算法对 Giraffe的 6 阈值分割结果分别为 0.677 8、0.677 3、0.670 0、0.669 1、0.676 0、0.670 0;对 Weasel 的 10 阈值分割结果分别为 0.902 1、0.888 0、0.892 1、0.867 3、0.864 4、0.889 5。再次表现出基于 IDA 算法的多阈值图像分割可以获得与原图像更加相似的图像,能很好地完成图像分割的任务。

最后对各算法的最优适应度值加以分析:由表 5 可知,本文算法在这 6 种算法中所获得的值最大。适应度函数值表示的是图像分成的若干个部分间的类间方差,最优适应度值越大,图像分割的质量也就越高,抗噪性也会不断提高。例如,IDA、PSO、SCA、BA、HSO 和 ALO 算法对 Bridge 图像进行 4 阈值分割的最优适应度函数值分别为 8 287.094 2、8 286.998 1、8 281.841 7、8 286.150 4、8 286.328 3、8 287.049 0。虽然 ALO 算法在阈值个数较少时,与 IDA 算法差异很小,获得的图像分割质量较好,但随着阈值个数的增加,ALO 易陷于局部最优,影响图像分割的效果。这说明改进后的蜻蜓算法的寻优能力明显提升,能够有效地跳出局部最优,更稳定快速地收敛于全局最优解,从而获得理想分割阈值。

5.2 运行时间

由表 6 可知,对于同一幅待分割的图像,在设置相同的阈值个数时,本文算法运行时间最短,SCA 算法所消耗的时间略高于 IDA,ALO 算法的运行时间最长。随着阈值个数的增加,各算法的运行都在不断增加,但 IDA 算法仍具有一定优势,说明本文算法可以有效提高效率,能够较快地完成多阈值彩色图像分割任务。

表 6 各算法运行时间

图像	K	时间/s					
		IDA	PSO	SCA	BA	HSO	ALO
Zebra	4	5.467 0	5.784 3	5.471 6	5.936 1	5.469 5	6.267 0
	6	6.043 7	6.195 8	6.057 2	6.382 6	6.062 8	7.243 7
	8	6.808 7	6.831 7	6.818 7	6.993 8	6.824 9	8.208 7
	10	7.664 2	7.708 1	7.702 8	7.809 8	7.703 3	9.194 2
Giraffe	4	5.231 0	5.310 9	5.236 7	5.678 9	5.240 0	6.012 0
	6	5.909 9	6.067 8	6.011 3	6.189 9	6.010 0	7.133 4
	8	6.712 3	6.931 2	6.756 2	6.990 2	6.765 4	8.189 7
	10	7.549 8	7.608 9	7.586 9	7.745 3	7.589 9	9.167 5
Weasel	4	5.108 9	5.198 9	5.109 9	5.278 5	5.114 5	6.100 8
	6	5.893 0	5.921 3	5.901 4	6.082 6	5.978 9	7.121 1
	8	6.623 4	6.673 4	6.639 8	6.990 8	6.632 4	8.009 9
	10	7.421 3	7.490 4	7.503 8	7.601 1	7.435 6	9.101 1
Bridge	4	5.123 0	5.156 9	5.130 5	5.248 2	5.172 3	5.918 3
	6	5.678 9	5.689 7	5.679 9	5.893 8	6.694 3	6.923 4
	8	6.435 6	6.485 4	6.446 0	6.562 1	6.498 8	8.012 1
	10	7.128 9	7.246 9	7.130 2	7.359 0	7.209 0	9.021 7
Worker	4	5.334 7	5.339 9	5.335 6	5.440 6	5.359 2	6.012 5
	6	6.013 8	6.021 6	6.017 4	6.128 4	6.035 9	6.826 0
	8	6.801 4	6.808 9	6.806 9	6.811 0	6.808 3	7.628 1
	10	7.656 9	7.660 3	7.660 6	7.665 9	7.659 5	8.445 0
Massif	4	5.258 0	5.303 0	5.263 4	5.324 9	5.278 2	5.935 6
	6	5.987 0	6.010 3	5.990 2	6.013 1	5.992 6	6.743 7
	8	6.708 3	6.726 4	6.710 3	6.826 1	6.813 0	7.540 6
	10	7.525 8	7.589 1	7.528 3	7.631 6	7.689 2	8.293 7

综上所述,本文从图像分割精度和运行时间这两方面对 6 种优化算法的性能进行对比分析,SCA 算法的运行时间虽然和 IDA 算法差距较小,但其分割精度并不理想;ALO 算法虽然可以获得精度较高的分割图像,但其运行时间不占优势。本文提出的 IDA 算法的优越性在于不仅提高了多阈值图像分割的质量,而且算法的运行效率也显著提高,从而验证了 IDA 算法在彩色图像分割问题中的有效性和正确性,能够胜任复杂图像的处理任务。

6 结 语

通过引入混沌初始化和反向学习策略,本文提出了一种基于改进蜻蜓算法的多阈值彩色图像分割技术,以类间方差函数作为 IDA 算法的适应度函数,利用蜻蜓种群的两种行为:觅食行为和迁徙行为来快速搜寻图像多阈值分割时的最佳阈值。实验结果表明,对同一待分割的图像,在设置相同的阈值分割个数时,IDA 算法与其他 5 种算法相比,在全局搜索和快速收敛能力、图像分割精度、运行时间等方面都表现出显著

的优越性。此次实验随机地从伯克利图库中随机选取 6 幅图像,可以看出本文算法对不同类型的彩色图像能够达到实时性处理与分析的要求,具有很好的实用价值。

参 考 文 献

[1] 邢致恺,贾鹤鸣,宋文龙. 基于莱维飞行樽海鞘群优化算法的多阈值图像分割[J/OL]. 自动化学报:1 - 15[2019 - 02 - 23]. https://doi.org/10.16383/j.aas.c180140.

[2] 于月娜,梁光明,刘任任,等. 基于图割算法的宫颈细胞分层次分割[J]. 计算机应用与软件,2018,35(12) :233 - 236,329.

[3] 周茂,曾凯,杨奎,等. 肺部 CT 图像分割方法研究[J]. CT 理论与应用研究,2018,27(6):3 - 11.

[4] 胡加鑫,贾鹤鸣,邢致恺,等. 基于鲸鱼算法的森林火灾图像多阈值分割[J]. 森林工程,2018,34(4):1 - 6.

[5] 刁智华,刁春迎,袁万宾,等. 基于改进型模糊边缘检测的小麦病斑阈值分割算法[J]. 农业工程学报,2018,34(10): 147 - 152

[6] 张杰,张末含,李述特,等. 基于图像处理的橡胶树割痕提取与死皮识别[J]. 计算机应用与软件,2019,36(1):259 - 262,329.

[7] Mirjalili S. Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems[J]. Neural Computing & Applications, 2016, 27(4):1053 - 1073.

[8] 赵齐辉,杜兆宏,刘升,等. 差分进化的蜻蜓算法[J]. 微电子学与计算机,2018,35(7):101 - 105.

[9] 李阳,归伟夏. 一种基于改进小生境遗传算法的图像分割方法[J]. 计算机应用与软件,2017,34(4):202 - 206,248.

[10] 许栋,崔小欣,王田,等. 基于 Logistic 映射的混沌随机数发生器研究[J]. 微电子学与计算机,2016,33(2):1 - 6.

[11] 汪慎文,丁立新,谢承旺,等. 应用精英反向学习策略的混合差分演化算法[J]. 武汉大学学报,2013,59(2):111 - 116.

[12] 蒋刚毅,黄大江,王旭,等. 图像质量评价方法研究进展[J]. 电子与信息学报,2010,32(1):219 - 226.

[13] 张晓琳,刘直芳,代金波,等. 基于视觉感知的梯度结构相似度图像质量评价[J]. 计算机应用研究,2011,28(6):2348 - 2351.

[14] 王正友,李振兴,林维斯,等. 结合 HVS 和相似特征的图像质量评估方法[J]. 仪器仪表学报,2012,33(7):168 - 174.

[15] 郎春博,贾鹤鸣,邢致恺,等. 基于改进正余弦优化算法的多阈值图像分割[J/OL]. 计算机应用研究,2019,37(4):1 - 7.

[16] 韦苗苗, 江铭炎. 基于粒子群优化算法的多阈值图像分割[J]. 山东大学学报(工学版), 2005(6):122 - 125.

[17] 李宗妮,吴伟民,林志毅. 一种采用改进蚁狮优化算法的图像增强方法[J]. 计算机应用研究,2018,35(4):304 - 306,318.

[18] 刘立群,王联国,火久元. 基于改进和声搜索算法的玉米叶片病害图像分割算法[J]. 计算机应用与软件,2016,33(4):189 - 192.

[19] 王智杰,牛硕丰,刘相兴,等. 蝙蝠算法优化二维熵的变电设备红外图像分割应用研究[J]. 电子设计工程,2018,26(18):89 - 93.

(上接第 233 页)

[5] 王松,刘复昌,黄骥,等. 基于卷积神经网络的深度图姿态估计算法研究[J]. 系统仿真学报,2017,29(11):2618 - 2623.

[6] 梅峰,刘京,李淳芃,等. 基于 RGB-D 深度相机的室内场景重建[J]. 中国图象图形学报,2018,20(10):1366 - 1373.

[7] Yamaguchi K, Mcallester D, Urtasun R. Efficient joint segmentation, occlusion labeling, stereo and flow estimation[C]//ECCV: European Conference on Computer Vision, 2014:756 - 771.

[8] Saxena A, Chung S H, Ng A Y. Learning depth from single monocular images[C]//Advances in Neural Information Processing Systems 18, MIT Press, 2006:1161 - 1168.

[9] Liu M, Salzmann M, He X. Discrete-continuous depth estimation from a single image[C]//Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, 2014:716 - 723.

[10] Eigen D, Puhrsch C, Fergus R. Depth map prediction from a single image using a multi-scale deep network[EB]. arXiv:1406.2283, 2014.

[11] Liu F, Shen C, Lin G. Deep convolutional neural fields for depth estimation from a single image[C]//2015 IEEE Conference on Computer Vision and Pattern Recognition(CVPR). IEEE,2015.

[12] Wang P, Shen X, Lin Z, et al. Towards unified depth and semantic prediction from a single image[C]//2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2015.

[13] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]//IEEE Conference on Computer Vision & Pattern Recognition. IEEE Computer Society, 2016.

[14] Chakrabarti A, Shao J, Shakhnarovich G. Depth from a single image by harmonizing overcomplete local network predictions[EB]. arXiv:1605.07081, 2016.

[15] 李耀宇,王宏民,张一帆,等基于结构化深度学习的单目图像深度估计[J]. 机器人,2017,39(6):812 - 819.

[16] Li B, Dai Y, Chen H, et al. Single image depth estimation by dilated deep residual convolutional neural network and soft-weight-sum inference[EB]. arXiv: 1705.00534, 2017.