Software Engineering and Applications 软件工程与应用, 2019, 8(2), 80-88
Published Online April 2019 in Hans. http://www.hanspub.org/journal/sea
https://doi.org/10.12677/sea.2019.82010

Hans汉斯

A Method for Vector Graph Generation Based on Firefly Algorithm

Cui Zhang, He Li, Jingyi Sun

College of Software Engineering, Jilin University, Changchun Jilin
Email: 1761324816@qq.com

Received: Apr. 4th, 2019; accepted: Apr. 19th, 2019; published: Apr. 26th, 2019

Abstract

Firstly, this paper analyzes the principle of vector graph generation, which is essentially an optimization problem. Firefly algorithm is an evolutionary algorithm to solve optimization problems in recent years. In order to improve the performance of vector graph generation, this paper modifies the operation and definition of firefly algorithm to adapt to this problem. The greedy strategy is used to divide the sub-problems to reduce the search space and overcome the inefficiency when the number of graphics is too high and the algorithm is difficult to converge. Experiments verify the reduction ability and compression ability of the algorithm, and compare it with the genetic algorithm, which proves that the method in this paper has better performance in the reduction degree.

Keywords

Vector Graph Generation, Firefly Algorithm, Image Compression

基于萤火虫算法的矢量图生成方法

张　翠，李　贺，孙婧伊

吉林大学软件学院，吉林　长春
Email: 1761324816@qq.com

收稿日期：2019年4月4日；录用日期：2019年4月19日；发布日期：2019年4月26日

摘　要

本文首先分析了矢量图生成问题的原理，其本质上是一个最优化问题，萤火虫算法是最近几年出现的一个解决最优化问题的演化算法，为了改善矢量图生成问题的性能，本文修改萤火虫算法相关操作和定义

文章引用: 张翠, 李贺, 孙婧伊. 基于萤火虫算法的矢量图生成方法[J]. 软件工程与应用, 2019, 8(2): 80-88.
DOI: 10.12677/sea.2019.82010

以适应本问题。具体采用贪心策略划分子问题以缩小搜索空间，克服图形数过高时的效率低下和算法难以收敛的问题。通过实验验证了算法的还原能力和压缩能力，并与采用遗传算法的生成方式进行对比，证明了本文方法在还原度方面性能更优。

关键词

矢量图生成，萤火虫算法，图像压缩

1. 引言

图像在计算机中主要以光栅图(位图)或矢量图的形式存储。位图由像素表示，每个像素都有自己特定的位置和颜色值，而矢量图使用基于数学方程的几何对象来描述图像。相比于位图，矢量图的文件大小只与描述的图形的复杂度有关，占存储空间更小；由于打开过程是软件按图像中存储的算法进行，矢量图在进行图像伸缩时不会失真，易于编辑。这些特性使得矢量图能广泛应用于图像压缩与保真、数据压缩、三维建模等领域。

图像的矢量化方法也引起了学者的广泛讨论和研究。其中的一个方向是，利用多个几何图形的组合来对位图进行拟合。选择怎样的策略优化图形组合成了解决问题的关键。Chris Weller 首先提出使用启发式算法来解决这一问题。他将遗传算法中种群的进化与矢量图中图形元素的优化结合，实现了图形元素组合在视觉上逐渐趋近于原图的效果[1]。但是该算法生成的矢量图丢失了较多原图的细节。后来相继有学者在他做的工作的基础上进行了改进[2] [3]，使矢量图的还原效果有了一定的提高。

Xin-She Yang 在 2009 年提出了一种新型群智能寻优算法——萤火虫算法[4]，该算法通过萤火虫之间的亮度交流来寻找最优解，具有快速的全局搜索能力。因此，本文选择萤火虫算法(FA)作为搜索最优图形组合的策略，以图形进化的方式逐步生成矢量图，使矢量图在还原细节、缩小体积等方面有更优的表现。测试对比结果表明本文方法具有一定有效性与可行性。

2. 相关工作

生成与目标光栅图相似的矢量图的主要思路是，将一定数目的基本图形绘制成一幅图像，比较它与原图的相似程度，并以适当的策略不断改变图形位置和形状，逐步提高矢量图对光栅图的还原度，最终形成一幅由基本图形组成的矢量图。如图 1 所示，左侧为矢量图，右侧展示了组成该图的基本图形。

下面介绍该方法的主要步骤：

1) 设置画布：创建一个尺寸与目标图像相同的画布，取一个颜色作为画布的颜色。

2) 向画布添加基本图形：常见的基本图形包括矩形、三角形等[4]。

3) 计算适应度：将基本图形添加到画布上之后，需要评估矢量图的还原效果。通过比较每个基本图形与目标位图对应位置像素的差异，得到生成的矢量图的适应度。适应度越高，矢量图的还原度越好。

4) 图形的移动、变形和变色：为了提升矢量图的还原度，需要对基本图形的位置、形状和颜色等属性进行修改。

综上所述，矢量图生成问题本质上是一个最优化问题，而遗传算法、萤火虫算法等元启发式算法适

合于解决此类问题。Chris 采用遗传算法解决该问题[1]，基本图形信息以基因的形式存储，种群在进化中采取杂交、复制、位突变、交换渲染顺序等操作，以产生还原度更高的个体。

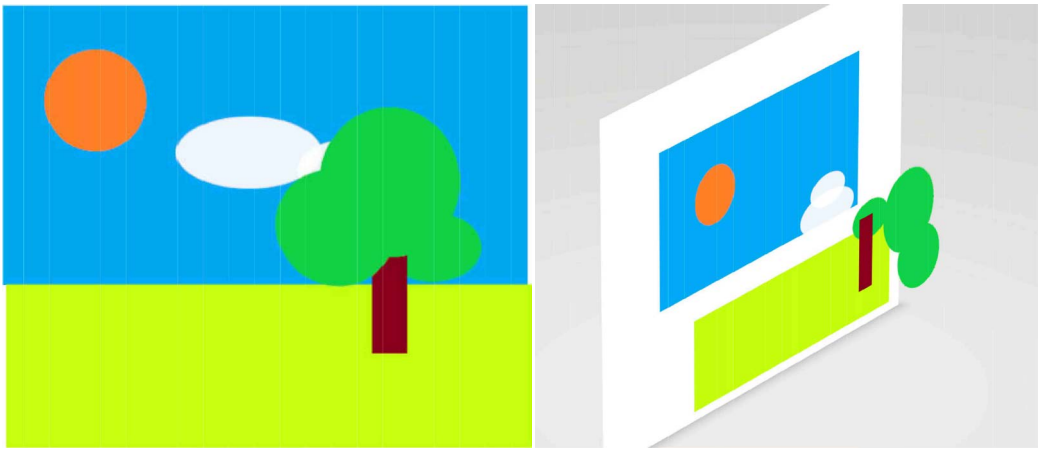

Figure 1. Schematic diagram of vector graph generation method
图 1. 矢量图生成方法原理示意图

Chris 的算法仍存在一些问题。首先，实验表明其算法存在“早熟”问题，在运行数千代后，算法基本陷入局部最优解，适应度难以继续提升，最终对目标图像的还原度不高。这主要是由于图形数量太多，导致搜索空间过大，而遗传算法难以探索如此大的搜索空间。其次，矢量图由大量的基本图形组成，每次计算目标函数都需要渲染全部图形，对性能的消耗很大，算法运行的效率很低。最后，Chris 的算法采用纯色三角形作为基本图形，单个图形中含有的信息较少，不利于进一步压缩体积，且对光滑边缘和柔和色彩的还原能力较差。

为了解决这些问题，本文在以下方面做出改进：

首先，以萤火虫算法进行优化，尝试利用萤火虫只在其感知范围内相互吸引的特点，避免“超级个体”带来的早熟收敛，增大探索范围。其次，利用贪心策略划分子问题，缩小搜索空间，同时减少一次迭代计算的图形个数，提升算法效率。最后，重新设计合适的基本图形，该基本图形由 4 个顶点组成，由折线或曲线围成，并由渐变颜色填充，如图 2 所示。这种设计使得图形相比单色图形包含的信息更多，有利于减少图形数量，压缩体积，并且曲线围成的图形对图像中光滑边缘和柔和色彩的还原能力更强。

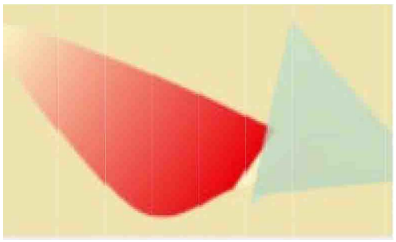

Figure 2. The basic graphics of vector graph generation process
图 2. 基本图形

3. 基于萤火虫算法的矢量图生成方法

3.1. 萤火虫算法的仿生原理[5]

萤火虫算法由自然界中萤火虫发光的一系列特性发展而来。该算法主要利用萤火虫在某个区域内寻

找荧光亮度更强的伙伴并向其聚拢的行为特点，实现基于群体搜索的随机寻优。

为了提取与算法相关的生物学概念，首先说明三个理想化条件：

1) 不考虑性别对于萤火虫吸引力的影响。

2) 萤火虫的吸引力与自身亮度成正比，与萤火虫之间的距离成反比。亮度最高的萤火虫最具有吸引力，使得在一定距离范围内的萤火虫向它移动。亮度相同时，萤火虫随机移动。

3) 萤火虫的荧光亮度反映一个解对待优化问题的满足程度。萤火虫亮度越高，说明萤火虫所在位置对目标的满足值越高。

萤火虫算法将解决问题的各个要素与自然界萤火虫的相关概念对应起来，并借鉴萤火虫的活动规律形成了完整有效的搜索策略。萤火虫算法将搜索空间中的点看作自然界中萤火虫的个体，用待优化问题的目标函数评价萤火虫个体所处位置的好坏，将搜索更优解的过程融入成萤火虫个体受其他个体吸引移动位置的过程。在该算法中，随着萤火虫种群中个体的位置更新，好的可行解逐渐替代较差可行解，最终实现对目标问题的解的优化。

3.2. 萤火虫算法相关操作定义

面向矢量图生成问题，本文对萤火虫算法的相关操作进行了定义，以适应该问题的解决，具体定义如下。

1) 萤火虫位置

上述矢量图像由多个基本矢量图形组合而成，故萤火虫的位置可以由 n 维向量表示：

$$X=\left(s_1,s_2,\cdots,s_n\right)^T \tag{1}$$

其中 $s_k\left(k\in\{1,2,\cdots,n\}\right)$ 为一个矢量图形。定义图形之间的差

$$s_1-s_2=t \tag{2}$$

其中 t 为 s_2 向 s_1 的变换。

为衡量变换的幅度(图形间的差异)，定义 $|t|$ 为该变换的等价长度，其值由图形 s_1 和 s_2 在位置、形状、颜色上的差异决定。于是可以定义图形的变换

$$s_2+kt=s_2',k\in[0,1] \tag{3}$$

k 表示 s_2 遵循变换 t 的程度，当 $0<k<1$ 时，s_2' 为一个介于 s_1 与 s_2 之间的图形。

2) 萤火虫移动

定义飞行向量

$$T_{ji}=X_i-X_j=\left(t_1,t_2,\cdots,t_n\right) \tag{4}$$

表示在搜索空间中萤火虫由位置 X_j 变为 X_i 的飞行方向。定义位置 X_j 与 X_i 之间的距离为

$$\left|T_{ji}\right|=\sqrt{\left|t_1\right|^2+\left|t_2\right|^2+\cdots+\left|t_n\right|^2} \tag{5}$$

于是在 k 时刻，由位置 X_1^k 向 X_2^k 飞行的萤火虫的位置可表示为

$$X_1^{k+1}=X_1^k+\beta\left(X_2^k-X_1^k\right) \tag{6}$$

其中 β 为吸引度，当 $\beta=0$ 时，下一时刻萤火虫的位置 $X_1^{k+1}=X_1^k$；当 $\beta=1$ 时，萤火虫位置 $X_1^{k+1}=X_2^k$。

β 的值与两萤火虫间的距离成平方反比，结合空气吸收效应的效果，可近似为

$$\beta = \frac{\beta_0}{1+\gamma|T|^2} \tag{7}$$

其中 β_0 为 $|T|=0$ 时的吸引度，取 $\beta_0=1$； γ 为空气对光的吸收率，本文将其初始值取为

$$\gamma = \left(\frac{|T_{\max}|}{4}\right)^2 \tag{8}$$

其中 $T_{\max}$ 为理论上两位置间距离的最大值，上述 γ 表示当两萤火虫距离为 $\frac{T_{\max}}{4}$ 时，两萤火虫的间的吸引程度下降为距离最近时的一半。

萤火虫在 k 时刻的随机运动可表示为

$$X^{k+1} = X^k + \alpha T_{\text{rand}} \tag{9}$$

其中 α 为步长， T_{rand} 为一随机的飞行向量。

至此，萤火虫 i 受到萤火虫 j 吸引的运动行为可以表示如下：

$$X_i = X_i + \frac{\beta_0}{1+\gamma|X_j - X_i|^2}\left(X_j - X_i\right) + \alpha T_{\text{rand}} \tag{10}$$

3) 适应度计算

对于本文中的矢量图生成算法，计算适应度需先将解 X 中图形在画布上依次绘制，得到一幅图像 P，计算 P 与目标图像 P' 的对应像素在 RGB 颜色空间中的距离平方和，如下式

$$\Delta = \sum\left[\left(R_i - R_i'\right)^2 + \left(G_i - G_i'\right)^2 + \left(B_i - B_i'\right)^2\right] \tag{11}$$

该式可以反映对应像素的颜色差异，进而反映两幅图像的差异程度，和越大则图像差异程度越大。最终适应度可由

$$\text{Fitness} = \frac{k\left(\Delta_{\max} - \Delta\right)}{\Delta_{\max}} \tag{12}$$

得出，其中 $\Delta_{\max}$ 为对于一个目标图像而言可能的最大差异值。当生成图形与目标完全一致时，适应度为 k，而生成图形与目标图像完全不一致时， $\Delta = \Delta_{\max}$，此时适应度为 0。

3.3. 以划分子问题的方式运行算法

当图形数量较大时，搜索空间过大，萤火虫算法容易陷入局部最优解，为解决这个问题，本文将矢量图生成问题划分为数个子问题，对每个子问题分别运行优化算法，每个子问题的结果通过贪心策略连接成为整个问题的完整解。这种方式有效地减小了搜索空间，使得萤火虫算法在每个子问题的解决中发挥了较好的效果；同时该方法减小了每代需要变换和绘制的图形数量，提升了性能。

整体解决方式与划分子问题方式的对比如表 1 所示。在表 1 所示情况下，“生成 n 个基本图形”的问题被划分为 k 个“生成 n/k 个基本图形”的子问题，并串行执行，依次将图形加入最终的解向量中，直至图形数量达到要求，整个问题解决。

3.4. 基于萤火虫算法的矢量图生成算法

综上所述，算法步骤如下，流程图如图 3。

(1) 初始化相关变量。根据输入的目标图像、图形个数、子问题个数，初始化画布、最大差异值 $\Delta_{\max}$、

矢量图 S 、萤火虫种群规模 N 。

Table 1. Difference between overall solution and problem-dividing solution
表 1. 整体解决方式与划分子问题方式的区别

	基本图形数	种群规模	子问题个数	解向量维数/每代渲染图形数
整体解决	n	m	1	n
划分子问题			k	n/k

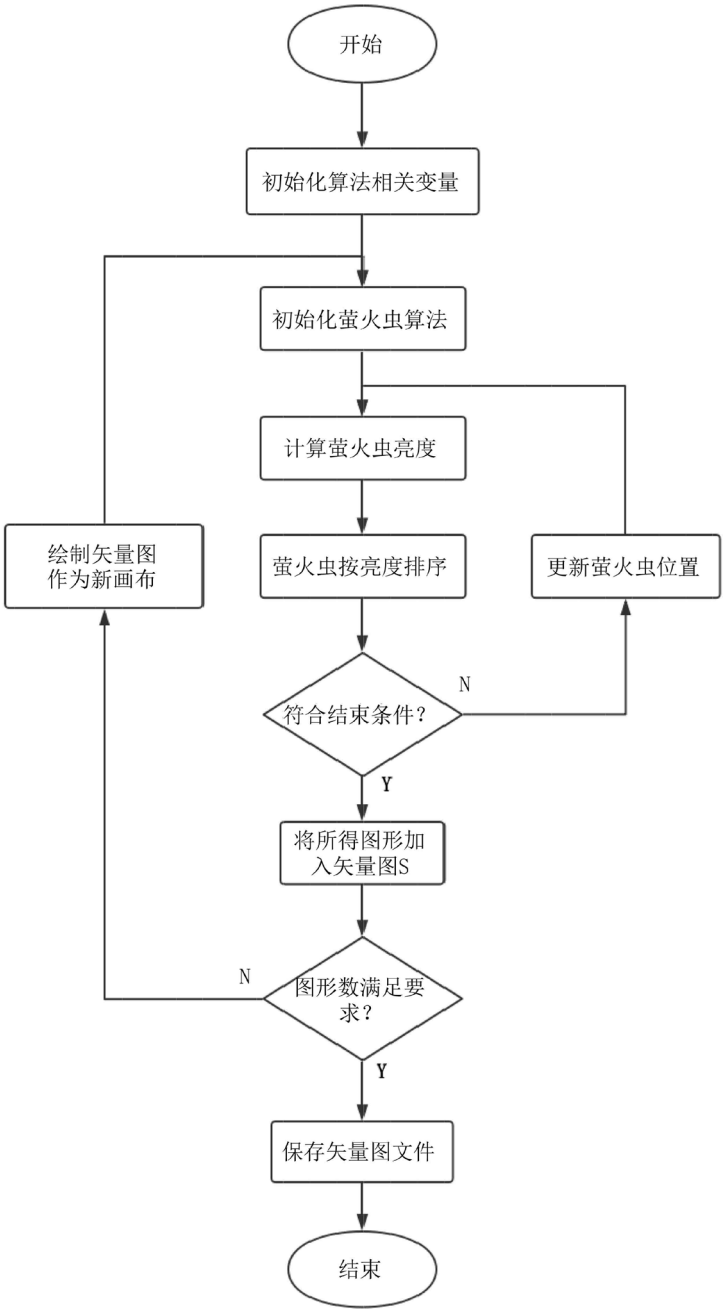

Figure 3. Flow chart
图 3. 算法流程图

(2) 初始化萤火虫算法。生成萤火虫种群及相关参数。

(3) 计算萤火虫亮度并排序。计算每个萤火虫位置 X_i 的适应度，作为萤火虫的亮度，并按亮度排序，亮度最高的萤火虫位置为 X_M。

(4) 判断是否满足萤火虫算法结束条件。若适应度达到要求或迭代次数达到上限，萤火虫算法结束，输出 X_M 中图形，转到步骤(5)，否则转到步骤(4)。

(5)更新萤火虫位置。萤火虫受到比它更亮的萤火虫的吸引，位置根据式(10)更新，转到步骤(3)。

(6) 将输出的图形加入矢量图 S，并绘制 S，作为新的画布。

(7) 判断图形数是否满足要求。若此时 S 中图形数满足要求，将矢量图 S 保存到文件中，否则转到步骤(2)。

4. 实验与对比分析

为了验证算法效果，本文选用“Stuffed Toy”、“Sailboats”、“Union Jack”、“Pumpkin”和“Dog”五张图作为实验对象进行适量图像生成，实验的软硬件环境如表 2 所示。

Table 2. Experimental environment
表 2. 实验环境

参数	值
编程语言	C#
操作系统	Windows 10 64位
CPU主频	2.4 GHZ
内存	8 GB
硬盘	500 GB

在最近吸引度 $\beta_0 = 1$，步长因子 $\alpha = 0.3$，萤火虫个数 $N = 5$ 的情况下，得到的输出结果如图 4 所示，图中从左自右分别为原图、本文方法输出图像和 Chris Weller 遗传算法[1]输出图像。

针对以上五张图片，对其执行本算法前后体积大小进行记录并计算压缩情况如下：(表 3)

Table 3. Data record
表 3. 数据记录

图片	位图体积	矢量图体积	压缩倍数	图形数量
Stuffed Toy	300 KB	16.84 KB	17.81	80
Sailboats	268 KB	7.01 KB	38.23	40
Union Jack	201 KB	4.03 KB	49.87	20
Pumpkin	188 KB	9.15 KB	20.54	50
Dog	186 KB	11.32 KB	16.43	60

在还原度方面，我们采用 Percent Correct [1]进行描述，该值计算方式类似于式(12)，可以线性地描述对目标图像的还原程度，计算方法为：

$$\text{Percent Correct} = \frac{\Delta_{\max} - \Delta}{\Delta_{\max}} \times 100\% \tag{13}$$

Figure 4. Experimental result
图 4. 实验结果

本文方法与 Chris Weller 遗传算法的对比如表 4 所示。从表中数据可以得出，普遍情况下本文算法可以进一步提高图片的还原度。

Table 4. Percent correct comparison
表 4. 还原度对比

图片	本文方法 Percent Correct	遗传算法 Percent Correct	图形数
Stuffed Toy	94.30%	90.84%	80
Sailboats	96.40%	92.30%	40
Union Jack	95.82%	81.37%	20
Pumpkin	95.10%	93.24%	50
Dog	90.70%	86.04%	60

综合以上实验结果及数据，本文方法在矢量图生成问题上具有可行性，在还原度方向上较 Chris Weller 的遗传算法有一定量的提高，同时可以达到满意的压缩效果。

5. 结语

为了将光栅图转化为矢量图达到压缩体积的目的，本文研究了一种矢量图生成方法[2]的原理和特性，提出了一种基于萤火虫算法并结合了贪心策略的矢量图生成方法。最后对该方法进行了实验验证，并将结果与采用遗传算法的矢量图生成方法对比，表明本文方法能更有效地解决矢量图生成问题。

参考文献

[1] Weller, C. (2009) Generation of Vector-Based Graphics from Existing Bitmap Images by Means of the Genetic Algorithm.

[2] Wilkens, S. (2005) Rendering Non-Photorealistic Images by Means of a Genetic Algorithm. Unpublished Student Project.

[3] Bergen, S. and Ross, B.J. (2012) Automatic and Interactive Evolution of Vector Graphics Images with Genetic Algorithms. *Visual Computer*, **28**, 35-45. https://doi.org/10.1007/s00371-011-0597-4

[4] Izadi, A., Ciesielski, V. and Berry, M. (2010) Evolutionary Non Photo-Realistic Animations with Triangular Brushstrokes. In: Li, J., Ed., *AI* 2010: *Advances in Artificial Intelligence*. *AI* 2010. *Lecture Notes in Computer Science*, Vol. 6464, Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-17432-2_29

[5] Yang, X.S. (2009) Firefly Algorithms for Multimodal Optimization. *International Symposium on Stochastic Algorithms*, **5792**, 169-178. https://doi.org/10.1007/978-3-642-04944-6_14