Painting Using Genetic Algorithm with Aesthetic Evaluation of Visual Quality

Sheng-Yu Feng and Chuan-Kang Ting

Department of Computer Science and Information Engineering
National Chung Cheng University, Taiwan
{fsy102m,ckting}@cs.ccu.edu.tw

Abstract. Creating art using artificial intelligence technologies is an emerging research topic. In particular, evolutionary computation has achieved several promising results in generating visual art and music. Evaluation of the items generated by evolutionary algorithms is a key issue at computational creativity. Interactive evolutionary algorithms are widely used to address this issue by incorporating human feedback in the fitness evaluation. However, this manner suffers from fatigue and decreasing sensitivity after long-term evaluation, which is commonly required in evolutionary algorithms. This paper proposes using an aesthetic evaluation of visual quality in the fitness evaluation for genetic algorithm (GA) to create paintings. Specifically, the fitness function considers two features for aesthetics. The generative ecosystemic art system, EvoEco, is applied as a test bench for the proposed method. Experimental results show that the proposed GA can generate satisfactory paintings by using aesthetic evaluation.

1 Introduction

Computational creativity has received increasing attention owing to the advance of artificial intelligence technologies. Creativity involves with the generation of appropriate novelty and represents some ideas based on previous works [2]. Several artificial intelligence technologies have been applied to achieve creativity by computers. In particular, using evolutionary computation to generate artworks, create visual art, and compose music has gained considerable promising results [4,10,13]. Evolutionary algorithms (EAs) are nature-inspired global search approaches and have succeeded in solving a variety of complex optimization problems. EAs manipulate a set of chromosomes representing candidate solutions and change them by mimicking the evolutionary operators in nature, such as selection, crossover, and mutation. The candidate solutions are evaluated by the fitness function. Following Darwin's theory "Survival of the Fittest," the chromosomes with relatively high fitness values are selected to survive into the next generation.

Fitness evaluation plays an important role in EAs because it guides the search direction. In computational creativity using EAs, it ordinarily lacks a specific fitness function for evaluating chromosomes. Human feedback is commonly used as an interactive fitness evaluation to address this issue. That is, human users act like the fitness function to assign fitness values or to select parents and survivors for EAs. This manner is useful since it combines the search capability of EAs and aesthetic taste of human. However, EAs usually require thousands or even million times of fitness evaluation, which makes interactive EAs impractical due to the fatigue and decreasing sensitivity of human beings after long-term evaluation. The interactive EAs for creativity, therefore, have to compromise with small population and short run. Some studies attempt to avoid user's fatigue. For example, Machwe and Parmee [12] used meta-feature clustering to learn and predict user's judgment. Llor et al. [11] applied support vector machine to synthesize the subjective fitness function. Kowaliw et al. [7] developed the EvoEco system to automatically detect and emphasize creative designs. A potential issue at these methods is that users may need to spend more time in fitness evaluation once the prediction results are wrong and needed to be corrected. Some research works on design of fitness functions that can emulate human aesthetic preference by using machine learning technologies such as neural networks [1] and coevolutionary algorithm [5]. Recently, Liu and Ting [9,10] proposed using objective evaluation metrics instead of human feedback to address this issue. They designed the fitness function based on music theory to evaluate the compositions and achieved satisfactory results.

This study aims to generate paintings by EA with aesthetic evaluation. Specifically, we develop a genetic algorithm (GA) and propose using aesthetic visual quality of paintings for fitness evaluation. In performance assessment, we implement the EvoEco system, an EA-based image generator, and replace its interactive EA with the proposed method. The resultant paintings show the effectiveness of the proposed GA with aesthetic evaluation. The remainder of this paper is organized as follows. Section 2 introduces the EvoEco system that uses an interactive EA to generate images. Section 3 presents our proposed GA and describes the structure of chromosomes and the fitness function based on aesthetic visual quality. Section 4 presents and compares the generated paintings. Finally, Section 5 draws the conclusions of this study.

2 Creativity and the EvoEco System

Regarding creativity, Dorin and Korb claimed that "Creativity is the introduction and use of a framework that has a relatively high probability of producing representations of patterns that can arise only with a smaller probability in previously existing frameworks" [3]. According to this definition, they built the EvoEco system to generate images. EvoEco is a multi-agent ecosystemic platform based on interactive EA to evolve into generative art. The system consists of two parts: the ecosystemic (generative) stage transforms a genome into a phenotype (image), and the evolutionary stage selects and evolves chromosomes.

Following the EA framework, an image is represented as a chromosome in EvoEco. A chromosome is composed of k agents and each agent is in charge of painting during its lifetime. The evolutionary process of EvoEco to generate images is as follows:

1. Randomly determine the background color of each chromosome.
2. Chromosomes serve as agents to paint during their life time.
3. Mutation and crossover are performed on the chromosomes.

Note that in EvoEco the fitness of images is evaluated by users. When the generated image is good, users can terminate the system. Figure 1 shows the EvoEco system and some evolving images.

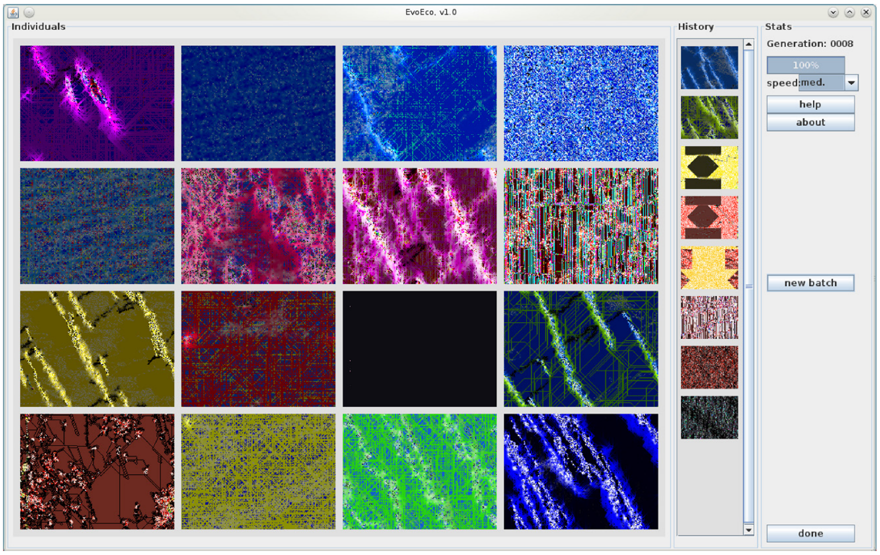


Fig. 1. Screenshot of the EvoEco system. The left frame shows the population of 16 chromosomes. The middle frame shows the history of previous eight choices. The right frame shows the control panel.

3 The Proposed Method

Genetic algorithm (GA) is a well-known EA and has shown its effectiveness on a variety of optimization problems. The general principle of GA is to simulate the mechanisms of natural evolution, such as selection, crossover, and mutation [6]. Based on Darwin's theory "Survival of the Fittest," GA is believed to be capable of evolving candidate solutions into better ones. To solve an optimization problem, GA represents candidate solutions as chromosomes, for which the encoding scheme is essentially related to the problem to be solved. Compared to conventional search methods, GA manipulates a set (population) of chromosomes in the search process. The fitness function is used to evaluate the quality (fitness) of candidate solutions (chromosomes). For a maximization problem, higher fitness implies better solution quality.

The evolutionary process of GA begins with initialization of the population. Then GA embarks on reproduction to generate new candidate solutions: First, the selection operator picks two chromosomes from the population to serve as parents. The two parents carry out the crossover operator to produce their offspring, where the crossover rate defines the probability to perform crossover. Afterward, mutation is performed with a probability, i.e., mutation rate, on the offspring to slightly change some genes. The above reproduction process repeats until the set of offspring is filled. Following "Survival of the Fittest," the survivor operator picks the fittest chromosomes out of the offspring population with (or without) the primitive population to survive and constitute the population for the next generation.

Design of GA for a given problem concerns representation, fitness function, selection, crossover, and mutation operators. The following describes our design of GA for generation of paintings.

3.1 Representation

A chromosome in the GA represents multiple agents to generate images. Formally, a chromosome I is represented by

$$I = (k, (ind_H, ind_S, ind_B), a^1, \ldots, a^k),$$

where $k \in \{2, \ldots, 6\}$ is the number of agents, (ind_H, ind_S, ind_B) indicates the background color, and $a^1, \ldots, a^k$ are painters.

The agents paint by assigning colors at some pixels. An agent can be defined by

$$a = (sD, delay, (a_H, a_S, a_B), a_{dir}, life),$$

where $sD \in \{0, \ldots, 8\}$ denotes the starting direction, $delay \in [0, 0.5]$ determines whether the agent paints this pixel or not, and $life$ is the agent's moving time. The terms (a_H, a_S, a_B) and a_{dir} is generated by a GP-tree, which indicates the color to paint at the current pixel and the direction for the subsequent move.

In the beginning, each agent randomly generates a GP-tree. Then, agents conduct the following actions at every time step:

1. Collect data from its local neighborhood as input.

   An agent will collect data from its Moore neighborhood (see Fig 2). These data, including eight directions and eight colors, are used to calculate the following input variables:
   - (H_c, S_c, B_c): The hue (H), saturation (S), and brightness (B) values of the pixel at the agent's current location
   - (H_p, S_p, B_p): The hue, saturation, and brightness values of the pixel at the agent's previous location
   - $(H_{\text{mean}}, S_{\text{mean}}, B_{\text{mean}})$: The mean hue, saturation, and brightness values of the agent's neighborhood
   - (ind_H, ind_S, ind_B): The chromosome's initial background color
   - d_p: The agent's previous direction

- $B_{\max}, B_{\min}$: Maximum and minimum brightness values among the agent's neighborhood
- *rand*: a uniformly random number in the range $[0, 1]$

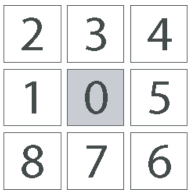


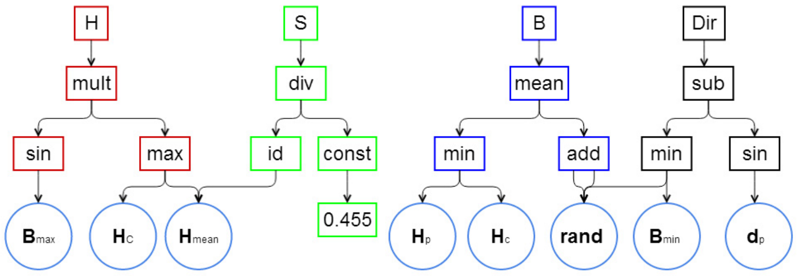
Fig. 2. Moore neighborhood and directions

2. Calculate output values from the inputs.

   The output values are computed by the agent's GP-tree and the 16 inputs. In this study, we introduce uncertainty in determining the direction to diversify the generated patterns. The output values include new hue a_H, saturation a_S, brightness a_B, and direction a_{dir}, which are used to determine the agent's next pixel to move to. Note that the hue, saturation, and brightness values might be out of bounds $[0, 1]$ and are thus truncated by discarding the integer part and negative sign. The direction is also regularized to $\{0, \ldots, 8\}$ by the remainder of dividing $(100 \cdot a_{dir})$ by nine. Table 1 lists the functions used for the GP-trees. An example GP-tree is given in Fig. 3.

Table 1. Function set used in the GP

Function	Action (on two inputs x and y)
add	$x + y$
sub	$x - y$
mult	xy
const	returns the associated constant value
max	$\max(x, y)$
min	$\min(x, y)$
sin	$\sin(x)$
id	x
div	$\begin{cases} 1 & y < 0.0001 \\ x/y & \text{otherwise} \end{cases}$
incdec	$\begin{cases} x + 1/8 & y > 0.5 \\ x - 1/8 & \text{otherwise} \end{cases}$
mean	$\frac{x+y}{2}$



$$
\begin{aligned}
a_H &= \sin(B_{\max}) * \max(H_c, H_{\text{mean}})\\
a_S &= H_{\text{mean}}/0.455\\
a_B &= \frac{1}{2}(\min(H_p, H_c) + rand + B_{\min})\\
a_{dir} &= \min(rand, B_{\min}) - \sin(d_p)
\end{aligned}
$$

Fig. 3. An example GP-tree for the four outputs a_H, a_S, a_B, and a_{dir}

3. Color the current pixel according to the output variables.

   For a pixel, each agent will probabilistically decide whether to paint it or not. The color to paint for the current pixel is determined by (a_H, a_S, a_B) from step 2.

4. Move to the next pixel in the direction given by output.

   Each agent will move to the next pixel according to direction a_{dir}, no matter the current pixel is colored or not. This study develops two strategies for the moving direction, i.e., deterministic and uncertain direction. The first strategy moves the agent to the pixel in the direction of a_{dir} deterministically. In the second strategy, the nine directions can be selected for the agent to move, where the a_{dir} direction has a high probability 1/2 while each of the remaining eight directions has a low probability 1/16 to be selected as the moving direction. This uncertainty in the moving direction is expected to diversify the generated patterns.

3.2 Genetic Operators

The genetic operators of GA include parent selection, crossover, mutation, and survival selection. In this study, we employ the k-tournament selection in view of its recognized performance. The proposed GA uses a 5-tournament selection, which selects the best of 5 randomly picked chromosomes as a parent. Performing this selection twice gains a pair of parents for subsequent crossover and mutation.

This study adopts the well-known uniform crossover for the GA. Considering the fact that the two parents may have different numbers of agents, the crossover operation is performed with the smaller number of agents between the two parents. Then the uniform crossover sequentially determines at random which parent for a gene to inherit from. The two offspring thus hold the characteristics of both parents.

For the mutation, we adopt the uniform mutation. This mutation scans the genes of offspring and decides to change a gene to a random value according to the mutation rate. For the genes representing agents, if mutation occurs, it will randomly generate a new agent to replace the old one. This mutation is expected to increase the diversity.

As for survival selection, this study utilizes the $(\mu+\lambda)$ strategy to keep elitists among the population.

3.3 Fitness Function

In this study, we propose using two measures of visual quality, i.e., edge distribution and line intersection, in place of human feedback to evaluate the quality of painting.

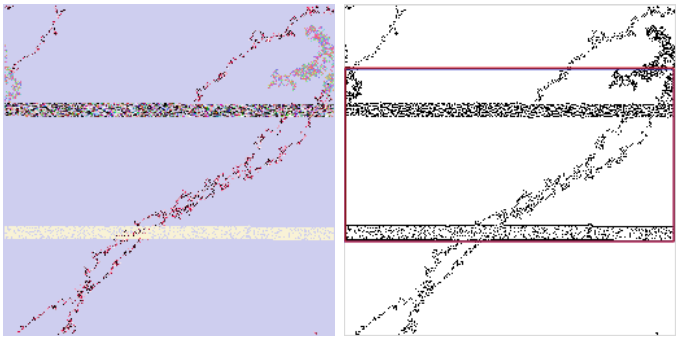
Edge Distribution

In evaluating the fitness of a painting, we first consider that the important regions should have more focus points. Li et al. [8] have validated the utility of this feature. Given that our painting comprises the results from several agents coloring the canvas pixel by pixel and thus has interactions, we remove the Gaussian smoothing filtering used in the original method and keep the edge noises to increase edge points in the generated paintings.

Specifically, we convert the image from HSB scheme to gray channels and then apply a 3×3 Laplacian filter on it (cf. Fig. 4). Afterward, we calculate the area of the smallest bounding box and its ratio to the whole area. The bounding box is defined as the rectangle that contains 81% edge points [8]. Figure 5 shows the original image and the image after applied with Laplacian filter and its bounding box. According to the test in [8], high-quality paintings have a low ratio, whereas low-quality paintings have a high ratio. Therefore, we define the complement of

-1	-1	-1
-1	8	-1
-1	-1	-1

Fig. 4. Discrete approximation to the Laplacian filter


(a) The better image that has a smaller bounding box

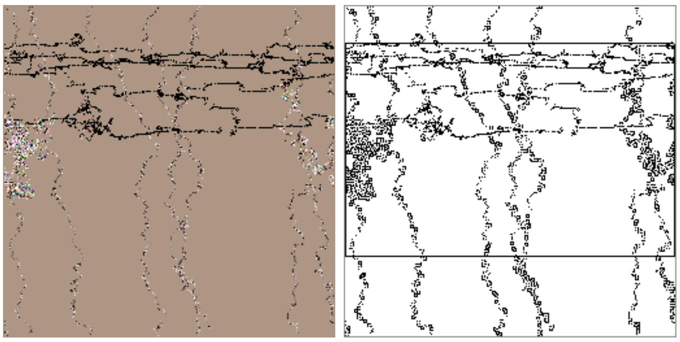

(b) The worse image that has a larger bounding box

Fig. 5. Laplacian filter example

this ratio as the first part of fitness evaluation and compute it by

$$f_{\text{edge}} = 1 - \frac{W_b H_b}{WH},$$

where W_b and H_b denote the width and height of the bounding box, and W and H are the width and height of the image, respectively.

Line Intersection

In the light of the proposed painting method, it is desired to have more interaction between agents. The interaction level of agents is reflected on the number of line intersections in the image. Accordingly, we define the second part of fitness evaluation by

$$f_{\text{intersect}} = \frac{\#\text{intersections}}{WH}.$$

The fitness value of a painting is defined by

$$F = f_{\text{edge}} + f_{\text{intersect}}.$$

Table 2. Parameter setting

Parameter	Value
Paint size	256×256
Representation	Agents
Population size	10
Selection	5-tournament
Crossover	Uniform
Crossover rate P_c	1.0
Mutation	Uniform
Mutation rate P_m	5/chromosome_length
Survivor	$\mu + \lambda$
Termination	10 generations

The objective of GA is to find the paintings that maximize the above fitness value considering edge distribution and line intersection.

4 Simulation Results

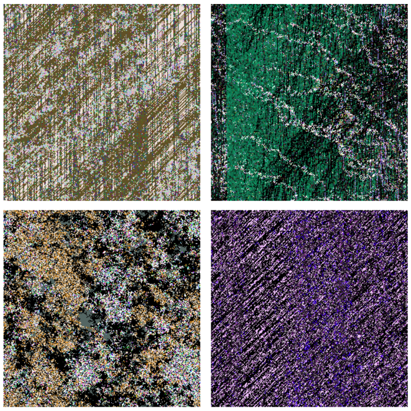
This study conducts simulations to examine the performance of the proposed GA-based painting system. Table 2 lists the parameter setting for the GA. Two factors are taken into account in the simulations: The first is the use of smooth rule; the second is the presence of uncertainty in the moving direction. The study investigates the results from the four combinations of these two factors.

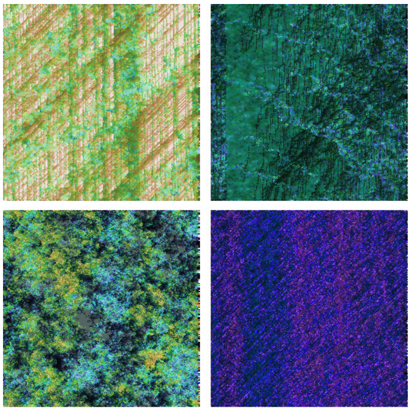
1) No Smoothing + Deterministic Direction

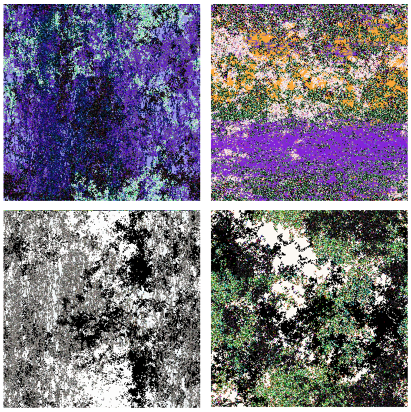
We first experimented with drawing a pixel in a deterministic direction without performing the smoothing rule at each step. The color of a pixel is calculated according to its neighbors' information. The simulation results in Fig. 6a shows that the generated patterns lack diversity. This is caused by the use of deterministic direction for drawing pixels, which results in many straight lines in the patterns. To address the monotony issue, we adopt the smoothing rule and enable uncertainty in the direction to move.

2) Smoothing + Deterministic Direction

Figure 6b presents the results of drawing a pixel in a deterministic direction with smoothing at each step. The generated patterns are similar with Fig. 6a because the moving directions are deterministic. However, the gaps between pixels are moderated by smoothing among neighbors. It helps to generate smoother and more harmonious patterns in Fig. 6b than the patterns in Fig. 6a.


(a) no smoothing + deterministic direction


(b) smoothing + deterministic direction


(c) no smoothing + uncertain direction

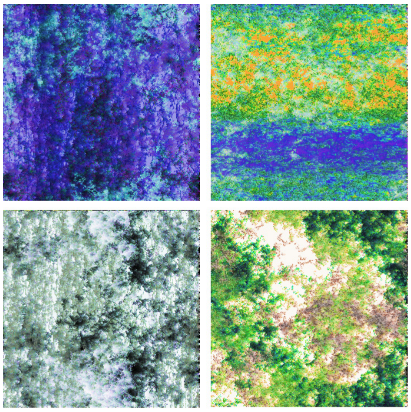

(d) smoothing + uncertain direction

Fig. 6. Simulation results

3) No Smoothing + Uncertain Direction

The results in Fig. 6c are generated by drawing pixels in uncertain directions without smoothing. The effect of applying uncertainty in moving direction is obvious: Comparing the patterns in Figs. 6a–6c, uncertain directions lead to high variety and fewer straight lines in the generated patterns. This outcome is preferable for visual experience.

4) Smoothing + Uncertain Direction

Finally, Fig. 6d shows the patterns generated by drawing a pixel in an uncertain direction with smoothing at each step. The results reflect the effects of smoothing and uncertainty: The generated patterns have fewer straight lines due to uncertainty in moving; furthermore, smoothing contributes to harmonious and moderate coloring.

5 Conclusions

This study proposes using visual quality as the fitness evaluation criterion to address the issues of fatigue and decreasing sensitivity at interactive EAs for computational creativity. The proposed fitness function is based on two measures of visual quality, i.e., edge distribution and line intersection. The edge distribution considers the ratio of bounding box area to the whole area. The line intersection reflects the interaction level of agents.

Simulations are conducted on the EvoEco system to investigate the effects of the proposed method. The simulation results show that smoothing among neighbors helps to bring about harmonious coloring. In addition, the uncertainty in moving directions leads to fewer straight lines and thus increases the variety and diversity of generated patterns. In general, these two factors and the proposed fitness function based on visual quality achieve satisfactory patterns, which validate the effectiveness of EAs and aesthetic evaluation on computational creativity.

Future work includes some directions. First, the fitness function can consider more aesthetic evaluation metrics. Second, design of genetic operators based on aesthetic evaluation is promising for EAs to generate satisfactory artworks.

References

1. Baluja, S., Pomerleau, D., Jochem, T.: Towards automated artificial evolution for computer-generated images. Connection Science 6(2), 325–354 (1994)
2. Boden, M.A.: The Creative Mind: Myths and Mechanisms. Basic Books (1991)
3. Dorin, A., Korb, K.B.: Improbable creativity. In: Proceedings of the Dagstuhl International Seminar on Computational Creativity (2009)
4. Fernandes, C.M., Mora, A.M., Merelo, J.J., Rosa, A.C.: Kants: A stigmergic ant algorithm for cluster analysis and swarm art. IEEE Transactions on Evolutionary Computation 44(6), 843–856 (2013)
5. Greenfield, G.: Co-evolutionary methods in evolutionary art. In: The Art of Artificial Evolution, pp. 357–380 (2008)
6. Holland, J.: Adaptation in Natural and Artificial Systems. University of Michigan Press (1975)
7. Kowaliw, T., Dorin, A., McCormack, J.: Promoting creative design in interactive evolutionary computation. IEEE Transactions on Evolutionary Computation 16(4), 523–536 (2012)
8. Li, C., Chen, T.: Aesthetic visual quality assessment of paintings. IEEE Journal of Selected Topics in Singal Processing 3(2), 236–252 (2009)
9. Liu, C.-H., Ting, C.-K.: Polyphonic accompaniment using genetic algorithm with music theory. In: Proceedings of the 2012 IEEE Congress on Evolutionary Computation (2012)
10. Liu, C.-H., Ting, C.-K.: Evolutionary composition using music theory and charts. In: Proceedings of the 2013 Computational Intelligence for Creativity and Affective Computing (2013)

11. Llorà, X., Sastry, K., Goldberg, D.E., Gupta, A., Lakshmi, L.: Combating user fatigue in igas: Partial ordering, support vector machines, and synthetic fitness. In: Proceedings of the 2005 Conference on Genetic and Evolutionary Computation, vol. 2, pp. 1363–1370 (2005)
12. Machwe, A.T., Parmee, I.C.: Reducing user fatigue within an interactive evolutionary design system using clustering and case-based reasoning. Engineering Optimization 41(9), 871–887 (2009)
13. Valdez, M.G., Guervós, J.J.M., Trujillo, L., de Vega, F.F., Romero, J.C., Mancilla, A.: Evospace-i: a framework for interactive evolutionary algorithms. In: Proceedings of the 2013 Genetic and Evolutionary Computation Conference, pp. 1301–1308 (2013)