An Interface for Fitness Function Design

Penousal Machado, Tiago Martins, Hugo Amaro, and Pedro H. Abreu

CISUC, Department of Informatics Engineering, University of Coimbra,
3030 Coimbra, Portugal
{machado,tiagofm}@dei.uc.pt, hamaro@student.dei.uc.pt, pha@dei.uc.pt

Abstract. Fitness assignment is one of the biggest challenges in evolutionary art. Interactive evolutionary computation approaches put a significant burden on the user, leading to human fatigue. On the other hand, autonomous evolutionary art systems usually fail to give the users the opportunity to express and convey their artistic goals and preferences. Our approach empowers the users by allowing them to express their intentions through the design of fitness functions. We present a novel responsive interface for designing fitness function in the scope of evolutionary ant paintings. Once the evolutionary runs are concluded, further control is given to the users by allowing them to specify the rendering details of selected pieces. The analysis of the experimental results highlights how fitness function design influences the outcomes of the evolutionary runs, conveying the intentions of the user and enabling the evolution of a wide variety of images.

Keywords: Ant Paintings, Fitness Function, Autonomous Evolutionary Art

1 Introduction

According to McCormack [13] evolutionary aesthetic search implies two main considerations: (i) The design of a generative system that creates individuals; (ii) The evaluation of the fitness of such individuals. Furthermore, there are two basic approaches to fitness evaluation: (i) Interactive Evolutionary Computation (IEC); (ii) Some form of automated fitness assignment. Although some systems use a combination of these two approaches, the majority focuses on one of them.

IEC allows the users to express their preferences and directly influence the course of evolution, allowing them, in theory, to guide it towards regions of the space that match their goals. However, this comes with a burden: the need to evaluate a vast number of individuals. In practice, more often than not, human fatigue prevents the prolific exploration of the search space. The study of automated fitness assignment may bring insights towards a better understanding of aesthetics, and while it is an effective way of fighting human fatigue. However, to some extent, the solution defeats the purpose since the users are no longer able to express their preferences. Thus, automated aesthetic fitness assignment is vital for the development of an autonomous evolutionary artist. However, it falls short when the goal is to design a creativity support tool that allows the users to express their artistic preferences and intentions. Machine learning techniques

have been used to capture the aesthetic preferences of users with some degree of success (see e.g. [3]). We argue that although such techniques have a role in the development of creativity support tools, the state of the art hasn't reached the level where these techniques would suffice on their own.

Machado and Pereira [12] presented a non-photorealistic rendering (NPR) algorithm inspired by ant colony approaches, where the trails of artificial ants were used to produce a rendering of an original input image. The large number of parameters controlling the behavior of the ants, and the dependencies among parameters, prevented their tuning by hand. As such, an IEC approach was adopted [12]. Instead of being forced to perform low-level changes, users become breeders of species of ants that produce results that they find valuable. The experimental results showed that human fatigue was taking its toll: only the disciplined and patient users were able to guide the algorithm towards non-trivial combinations of parameters. Most users adopted an opportunistic approach, valuing novelty over quality. Additionally, when the users started the process with a specific type of image in mind, e.g. a rendering consisting exclusively of straight lines, they failed to reach their goal.

In this paper a novel interface for the design of fitness functions is described. This frees the users from the need to perform individual assessments allowing them to express their aesthetic and artistic goals by specifying the characteristics they desire or which to avoid. While the ants paint, statistics describing their behavior are gathered, and when the painting is completed, image features are calculated. These behavioral and image features are the basis for the creation of the fitness functions. The exact meaning of each of these features and the interdependencies among them may be difficult to grasp by the common user. To tackle this problem, the interface is responsive, allowing the user to perceive the semantics associated with each feature. Once the evolutionary runs are concluded we further empower the users by letting them to select their favorite phenotypes, apply the associated genotypes to different input images, and control the details of the final rendering.

2 State of the Art

Tzafestas [22] presents a system where artificial ants pick-up and deposit food, which is represented by paint, studying the self-regulation properties of the system and complexity of the resulting images. Ramos and Almeida [18] explore the use of ant systems for pattern recognition purposes. The artificial ants successfully detect the edges of the images producing stylized renderings of the originals. The artistic potential of the approach is explored in later work [17]. Urbano [23] describes a multi-agent system based on artificial ants, presenting the first artificial ant paintings produced using a faithful biological model in [24].

Aupetit et al. [1] introduce an interactive Genetic Algorithm (GA) for the creation of ant paintings. The algorithm evolves parameters of the rules that govern the behavior of the ants. The artificial ants deposit paint on the canvas as they move, thus producing a painting. In a later study, Monmarché et

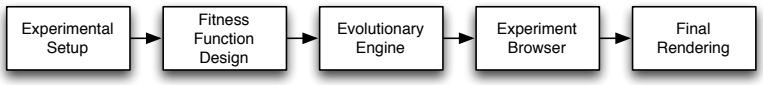

Fig. 1: The pipeline.

al. [14] refine this approach exploring different rendering modes. Semet et al. [21] applied ant colony simulation to produce NPRs of input images. For the same purpose, Fernandes et al. [4] use an approach akin to Ramos and Almeida [18]. Greenfield [5] presents an evolutionary approach to the production of ant paintings and explores the use of behavioral statistics of the artificial ants to automatically assign fitness. Later he adopted a multiple pheromone model where ants' movements and behaviors are influenced (attracted or repelled) by both an environmentally generated pheromone and an ant generated pheromone [6].

The use of evolutionary algorithms to create image filters and NPRs of source images has been explored by several researchers. Focusing on works where there was an artistic goal: Ross et al. [19,15] use Genetic Programming, multi-objective optimization techniques, and an empirical model of aesthetics to automatically evolve image filters; Lewis [9], evolves live-video processing filters through interactive evolution; Machado et al. [10], use GP to evolve image coloring filters from a set of examples; Yip [25] employs GAs to evolve filters that produce images that match certain features of a target image; Collomosse [2] uses image salience metrics to determine the level of detail for portions of the image, and GAs to search for painterly renderings that match the desired salience maps; Hewgill and Ross [7] use GP to evolve procedural textures for 3D objects. Schlechtweg et al. [20] adopt a, non-evolutionary, multi-agent approach for stroke-based rendering.

3 The Framework

Figure 1 describes our pipeline. The users begin by setting up the experiment establishing parameters such as: input image, population size, number of generations, number of runs, crossover and mutation rate. Then, using the responsive interface, they design a fitness function that will be used to guide evolution. Control is then passed to the evolutionary engine. Each genotype of the GA population encodes the parameters of a species of ants. These parameters determine how that ant species reacts to the input image. Each painting, i.e. each phenotype, is produced by simulating the behavior of ants of a given species while they travel across the canvas, leaving a trail of varying width and transparency. When the evolutionary runs are finished, the users select individuals that match their artistic intentions. This can be done by browsing through the populations, but it is usually more effective to consult the "summary" windows that depict the fittest individuals of a run and the fittest individuals of each generation. The chosen individuals are then passed to the final rendering interface, which allows to apply their genotypes to different input images and control several aspects of the final rendering.

3.1 Ants' simulation

Our ants live on a 2D environment initialized with the input image and they paint on a painting canvas that is initially empty (i.e., black). The painting canvas is used exclusively for depositing ink. The luminance of an area of the environment is the available energy at that point. During simulation, ants gain energy traveling through bright areas, and this energy is removed from the environment. If the energy of an ant is below a given threshold it dies, if it is above a given threshold it generates offspring. The ants' movement is determined by how they react to light. Each ant has 10 sensory vectors, each with a given direction and length. These sensory organs return the luminance value of the area where each vector ends. To update the position of an ant one performs a weighted sum, calculating the sum of the sensory vectors divided by their norms, multiplied by the luminance of their end point and by the weight the ant gives to each sensor. The result of this operation is multiplied by a scaling factor that represents the ant's base speed. Subsequently, to represent inaccuracy of movement and sensory organs, the direction is perturbed by the addition of Perlin [16] noise to its angle. Each ant has a position, color, deposit transparency and energy; all the remaining parameters are shared by the entire species. Color is determined at birth, each ant assumes the color of the area of the environment where it was born, and does not change throughout its life. Thus, the ants may carry this color to areas of the canvas that possess different colors in the original image. A detailed description of the ants' simulation can be found in [12]. The video `cdv.dei.uc.pt/2014/sim.mov` depicts this simulation showing the environment and painting canvas.

3.2 Extracted Features

During the simulation of each ant species, i.e. the rendering of each phenotype, a series of behavioral statistics is collected, namely: $avg(ants)$ – average number of living ants; $deposited_{ink}$ – total amount of "ink" deposited by the ants; $coverage$ – percentage of the environment visited by the ants; $avg(distance)$ – average euclidean distance between the position where the ant was born and the one where it died; $avg(trail), std(trail)$ – average trail length and the standard deviation of the trail lengths; $avg(life), std(life)$ – average life span of the ants and its standard deviation; $avg(avg(width)), std(avg(width))$ – determined by calculating for each trail the average width, and then the average width of all trails, $avg(avg(width))$, and the standard deviation of the averages, $std(avg(width))$; $avg(std(width)), std(std(width))$ – determined by calculating for each trail the standard deviation of its width, then their average, $avg(std(width))$, and their standard deviation $std(std(width))$; $avg(avg(av))$, $std(avg(av))$, $avg(std(av))$, $std(std(av))$ which are analogous to the features regarding trail width, but pertaining to the angular velocity of the ants;

When the simulation of each ant species ends, the following image features are collected: $complexity$ – the image produced by the ants, I, is encoded in $jpeg$ format, and its complexity estimated using the following formula:

Table 1: Parameters encoded by the genotype

Name	#	Comments
$gain$	1	scaling for energy gains
$decay$	1	scaling for energy decay
$cons_{rate}$	1	scaling for size of circles drawn on the environment
$cons_{trans}$	1	transparency of circles drawn on the environment
$deposit_{rate}$	1	scaling for size of circles drawn on the painting canvas
$deposit_{transp}$	1	base transparency of circles drawn on the painting canvas
$dtransp_{min}$	1	limits for perturbation of deposit transparency when
$dtransp_{max}$	1	offsprings are generated
$initial_{energy}$	1	initial energy of the starting ants
$death_{threshold}$	1	death energy threshold
$birth_{threshold}$	1	generate offspring energy threshold
$descvel_{min}$	1	limits for perturbation of angular velocity when
$descvel_{max}$	1	offsprings are generated
vel	1	base speed of the ants
$noise_{min}$	1	limits for the perlin noise
$noise_{max}$	1	generator function
$initial_{positions}$	$2 * n$	initial coordinates of the n ants placed on the canvas
$sensory_{vectors}$	$2 * m$	direction and length of the m sensory vectors
$sensory_{weights}$	m	weights of the m sensory vectors

$complexity(I) = rmse(I, jpeg(I)) \times \frac{s(jpeg(I))}{s(I)}$, where $rmse$ stands for the root mean square error, $jpeg(I)$ is the image resulting from the $jpeg$ compression of I, and s is the file size function; $fract_{dim}, lac$ – fractal dimension of the ant painting estimated by the box-counting method and its λ lacunarity value estimated by the sliding box method [8], respectively; $similarity$ – similarity between the ant painting and the original image estimated as follows: $similarity = \frac{1}{1+rmse(I,O)}$, where I is the ant painting and O is the original image.

3.3 Evolutionary Engine

A GA is used to evolve the ant species' parameters. The genotypes are tuples of floating point numbers which encode the parameters of the ant species. Table 1 presents an overview of the encoded parameters. We use a two point crossover operator for recombination purposes and a Gaussian mutation operator. We employ tournament selection and an elitist strategy, the highest ranked individual proceeds – unchanged – to the next population.

4 The Anatomy of a Fitness Function

The fitness functions assume the form of a weighted sum. For each feature the user may indicate a weight, w_i, and the intention to minimize, maximize or make the feature match a target value. Specifying a target value implies that the fitness component associated with that feature is:

$$f_i = \frac{1}{1 + |target_{value_i} - feature_{value_i}|}$$

When the goal is to maximize a given feature, we use:

$$f_i = abs\left(\frac{feature_i}{offlinemax(feature_i)}\right),$$

where *offlinemax* returns the maximum possible value found for that feature. This value can be established analytically for certain features, e.g. fractal dimension never exceeds 2, and it was found empirically for the remaining ones. To prevent the evolutionary algorithm from focusing exclusively on a given feature we employ a logarithmic scale so that the evolutionary advantage decreases as the feature value becomes higher, promoting the discovery of individuals that use all features employed in the fitness function. This is accomplished using: $f'_i = log(1 + f_i)$. For minimization we use the same formulas as for maximization, but f'_i returns a negative value: $f'_i = -log(1 + f_i)$. Considering all of the above the fitness functions are given by: $\sum_{i=1}^{features} w_i \times f'_i$, where w_i is a value in the $[-1, 1]$ interval.

5 Fitness Function Design Interface

The interface is composed of a set of responsive "icons", one for each of the features. The vertical slider on the right of each icon specifies the weight associated with each feature. Pressing the rightmost button beneath each icon indicates maximization of the corresponding feature, while pressing the leftmost button indicates minimization. Specifying a target value for a feature is accomplished using the plus and minus buttons. for most user indicating a specific value for something as the *complexity* of an image would be meaningless. To circumvent this problem all values are specified in the $[0, 1]$ interval and then mapped to $[offlinemin(feature_i), offlinemax(feature_i)]$. Thus, specifying a target value of, e.g., 0.8 for a given feature and a positive weight indicates the wish to reach a value close to its maximum; conversely, indicating a value of 0.2 indicates the wish to reach a value close to its minimum. Specifying a negative weight indicates the wish to deviate from the target value.

To give the user a better grasp of the semantics associated with each feature the icons are responsive, in the sense that the displayed image changes in accordance with the changes of value for that feature. For instance, increasing the average number of ants increases the number of points displayed in the corresponding icon, increasing the $avg(distance)$ increases the distance between the start and end points of each trail, etc. Furthermore, since the features are not independent, the changes on one value may affect the appearance of other icons; e.g. increasing the number of ants while leaving the coverage and amount of deposited ink unchanged implies that each ant visits a smaller area of the canvas and deposits a smaller amount of ink. The interface reflects these dependencies by decreasing the radius and opacity of the circles of the icons corresponding to deposited ink and coverage.

Figure 2 depicts the interface. The rationale behind the fitness function being specified can be described as follows: maximizing *coverage* and *similarity* promotes paintings where the ants visit the entire canvas and that closely match the original image; in what concerns line width, maximizing, $avg(std(width))$ and $std(avg(width))$ promotes high variations of width and heterogeneous widths among lines, respectively, minimizing $avg(avg(width))$ promotes thin lines; Line

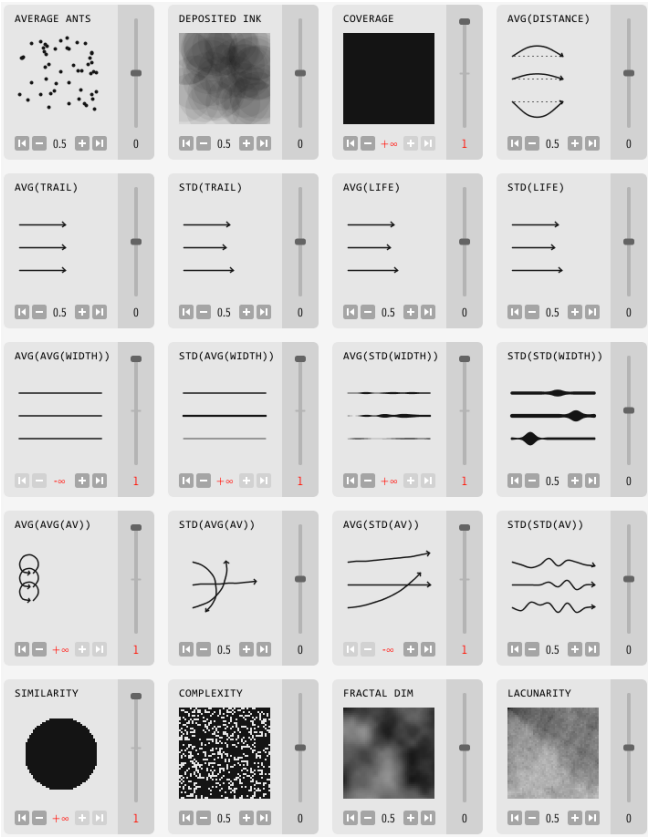

Fig. 2: The interface for fitness function design.

direction is controlled by maximizing $avg(avg(av))$ while minimizing $avg(std(av))$, which promotes the appearance of circular motifs (high angular velocity and low variation of angular velocity). The video `cdv.dei.uc.pt/2014/int.mov` illustrates the responsiveness of the icons.

6 Final Rendering Interface

One of the novel characteristics of our approach is the adoption of scalable vector graphics, which contrasts with the pixel based approaches used in most ant painting algorithms, and allows us to create resolution independent paintings. For this purpose, during simulation the trails of the ants are created by drawing circles of a given color and transparency as the ants move. The resulting image is then saved in bitmap format and scalable vector graphics, in this case as PNG and PDF files, respectively. This approach has two major drawbacks: (i) the PDF files may become rather large; (ii) the individual circles become visible at high zoom levels (see figure 3, left).

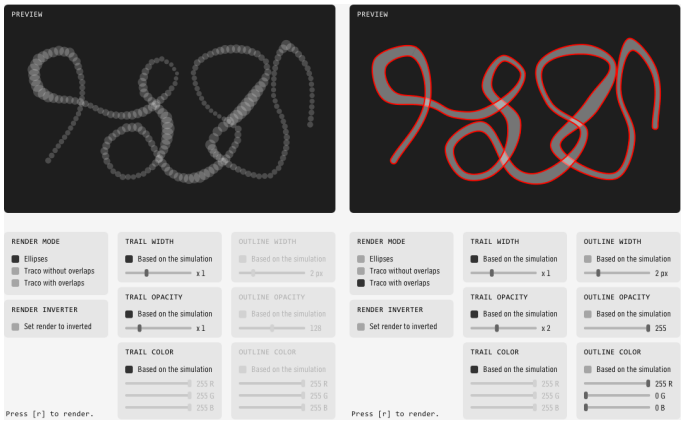

Fig. 3: On the left, the default rendering mode used during simulation. On the right, a more elaborated rendering mode.

To overcome this problem we developed a rendering algorithm that converts the set of circles of each trail into a line of variable width. The algorithm deals with self intersecting lines and enables the specification of additional rendering options: rendering mode – default, line, line with increased opacity on self-intersections; trail width – which can be set to a given value or based on the simulation, in this case the user may specify a multiplier to increase or decrease line width; trail opacity – which has an analogous behavior to trail width; and trail color, which can be based on the simulation or set to a specific color; The user may also indicate an outline for the lines, specifying the width of the outline, its opacity and its color, which can all be specified by assigning a fixed value or by inheriting the values from the simulation (see figure 3). Finally, the invert option reverses the logic of the painting algorithm, making ants react to darkness instead of luminance.

Since the process is computationally intensive, the final rendering is typically an independent process. Once the evolutionary runs are over the users select their favorite individuals for final rendering, using the final rendering interface to set the rendering options. To further empower users and give them a finer degree of control, during final rendering, we allow them to add ants to points of the canvas by pointing and clicking with the mouse. Adding ants to certain regions may increase the rendering detail of that area, or introduce ants in areas of the canvas that were not being visited.

7 Experimentation

The design of our interfaces was guided by user feedback. The appearance of the icons was validated individually. For each icon we performed evolutionary runs: maximizing the value of the associated feature, minimizing it, and setting as target an intermediate value. We confronted the users with the visual outcomes of these runs and adjusted or re-designed the icons accordingly. This process was

Fig. 4: Examples of phenotypes resulting from each of the 15 fitness functions.

repeated interactively eventually leading to the interface described in section 5. Once developed, the fitness design interface was given to eleven users that performed multiple evolutionary runs providing additional feedback. A detailed analysis of user friendliness and user experience is beyond the scope of the paper. Nevertheless, it is worthwhile mentioning the most common complaints: (i) users are puzzled by the meaning of the negative weights; (ii) they have difficulties in grasping the meaning of the fractal dimension and lacunarity icons; (iii) they dislike the similarity icon. By analyzing the fitness functions created by the users, it becomes obvious that negative weights and specific target values were rarely used. When questioning the users we realized that when they use these options their motivation was to "see what it does" rather than a specific outcome. The same applies to the use of fractal dimension and lacunarity. Considering this feedback we are likely to redefine the intervals for the weights setting them to [0,1]. The implication is that the users will no longer be able to indicate the intention to deviate from a given target value.

Giving an accurate portrayal of the results obtained by all users is close to impossible due to space limitations, therefore we focus on the results obtained by one of them: a graphic designer that was not familiar either with the inner working of the system or with the interface. After a short explanation of the workflow we asked him to create five different fitness functions and conducted ten evolutionary runs for each of these functions. Population size was set to 25

Fig. 5: The same genotype applied to different input images.

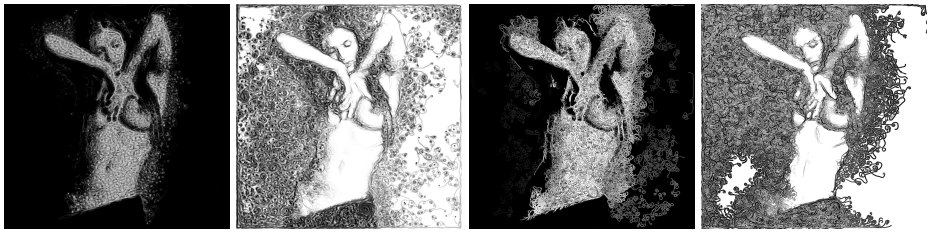

Fig. 6: The same genotype rendered with different final rendering options.

and the number of generations to 50, the other GA parameters are similar to the ones used in [11]. The input image was selected by the user, a photo of Angelina Jolie taken by Annie Leibovitz. Once these runs were finalized the user reviewed the results, selected his favorite individuals, and was asked to design an additional set of five fitness functions. This process was repeated iteratively resulting in a total of 15 fitness functions and 150 evolutionary runs. Figure 4 summarizes the results of these experiments by showing one individual per fitness function, with each row corresponding to one iteration. The results highlight not only the diversity of the results, but also the progress of the user through time.

One of the key aspects of our approach is the ability to apply selected genotypes to different input images. Thus, over time the users compile ant species that match their preferences and intentions, then applying these species to create NPRs of several images. In figure 5 we show the results of applying the genotype corresponding to the rightmost image of the bottom row of figure 4 to different input images. These results indicate that although the ant species are sensitive to the environment, i.e. input image, the characteristics of the painting, e.g. curviness of the lines, are inherent to the ant species. Therefore, applying the same ant species to different input images tends to result in ant paintings with similar aesthetic qualities.

The final rendering interface gives an additional degree of control to the users, allowing them to fine tune rendering options and explore alternative rendering modes. Figure 6 illustrates how different combinations of parameters affect the visual outcome. Since the details of the rendering are difficult to perceive in small format the video `cdv.dei.uc.pt/2014/ren.mov` illustrates the final rendering process.

8 Conclusions and Future Work

An interface for fitness function design in the scope of evolutionary ant painting system was presented. This interface allows the users to operate at a higher level of abstraction than in IEC and circumvents the user-fatigue problem. Nevertheless, unlike other automated fitness assignment schemes, the users are able to express their artistic and aesthetic preferences.

Although the system serves the user intents, different runs converge to different, and sometimes highly dissimilar, images. As such, we argue that the system opens the realm of possibilities that are consistent with the intents expressed by the users, often surprising them in the process. Moreover, while browsing the outcomes of evolutionary runs users often find images that they consider appealing due to their novelty and/or aesthetic properties, but which do not maximize the fitness function they specified. In future work we will use machine learning techniques to automatically define fitness functions from a set of such images. The automatic discovery of fitness functions may be a complement to the user interface and bring insights to understand the preferences of the users. The further refinement of the interface and the inclusion of additional features based on image analysis, more specifically related with color analysis, will also be addressed.

Acknowledgments This research is partially funded by: QREN/COMPETE, under contract 22997 (SI & IDT-CO-PROMOÇÃO); the Portuguese Foundation for Science and Technology, project PTDC/EIA–EIA/115667/2009; iCIS project (CENTRO-07-ST24-FEDER-002003), which is co-financed by QREN, in the scope of the Mais Centro Program and European Union's FEDER.

References

1. Aupetit, S., Bordeau, V., Monmarché, N., Slimane, C., Venturini, G.: Interactive Evolution of Ant Paintings. In: IEEE Congress on Evolutionary Computation. vol. 2, pp. 1376–1383. Canberra (8-12 december 2003)
2. Collomosse, J.P.: Supervised genetic search for parameter selection in painterly rendering. In: Applications of Evolutionary Computing, EvoWorkshops 2006. pp. 599–610. Budapest, Hungary (2006)
3. Ekárt, A., Sharma, D., Chalakov, S.: Modelling human preference in evolutionary art. In: EvoApplications (2). pp. 303–312 (2011)
4. Fernandes, C.M., Isidoro, C., Barata, F., Rosa, A.C., Guervós, J.J.M.: From pherographia to color pherographia: Color sketching with artificial ants. In: IEEE Congress on Evolutionary Computation. pp. 1124–1131 (2011)
5. Greenfield, G.: Evolutionary methods for ant colony paintings. In: Applications of Evolutionary Computing, EvoWorkshops2005: EvoBIO, EvoCOMNET, EvoHOT, EvoIASP, EvoMUSART, EvoSTOC. LNCS, vol. 3449, pp. 478–487. Springer Verlag, Lausanne, Switzerland (2005)
6. Greenfield, G.: Ant Paintings using a Multiple Pheromone Model. In: Bridges. London, England (2006)
7. Hewgill, A., Ross, B.J.: Procedural 3d texture synthesis using genetic programming. Computers and Graphics 28, 569–584 (2003)

8. Karperien, A.: Fraclac for imagej, version 2.5. In: http://rsb.info.nih.gov/ij/plugins/fraclac/FLHelp/Introduction.htm (2012)
9. Lewis, M.: Aesthetic video filter evolution in an interactive real-time framework. In: Applications of Evolutionary Computing, EvoWorkshops2004: EvoBIO, EvoCOMNET, EvoHOT, EvoIASP, EvoMUSART, EvoSTOC. LNCS, vol. 3005, pp. 409–418. Springer Verlag, Coimbra, Portugal (5-7 Apr 2004)
10. Machado, P., Dias, A., Cardoso, A.: Learning to colour greyscale images. The Interdisciplinary Journal of Artificial Intelligence and the Simulation of Behaviour – AISB Journal 1(2), 209–219 (2002)
11. Machado, P., Amaro, H.: Fitness functions for ant colony paintings. In: Proceedings of the 4th International Conference on Computational Creativity. pp. 32–39 (2013)
12. Machado, P., Pereira, L.: Photogrowth: non-photorealistic renderings through ant paintings. In: Soule, T., Moore, J.H. (eds.) Genetic and Evolutionary Computation Conference, Philadelphia, PA, USA, July 7-11, 2012. pp. 233–240. ACM (2012)
13. McCormack, J.: Facing the future: Evolutionary possibilities for human-machine creativity. In: Romero, J., Machado, P. (eds.) The Art of Artificial Evolution: A Handbook on Evolutionary Art and Music, pp. 417–451. Springer (2007)
14. Monmarché, N., Mahnich, I., Slimane, M.: Artificial art made by artificial ants. In: Romero, J., Machado, P. (eds.) The Art of Artificial Evolution: A Handbook on Evolutionary Art and Music, pp. 227–247. Springer (2007)
15. Neufeld, C., Ross, B., Ralph, W.: The evolution of artistic filters. In: Romero, J., Machado, P. (eds.) The Art of Artificial Evolution: A Handbook on Evolutionary Art and Music, pp. 335–356. Springer (2007)
16. Perlin, K.: An image synthesizer. In: Cole, P., Heilman, R., Barsky, B.A. (eds.) Proceedings of the 12st Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1985. pp. 287–296. ACM (1985)
17. Ramos, V.: On the implicit and on the artificial - morphogenesis and emergent aesthetics in autonomous collective systems. In: ARCHITOPIA Book, Art, Architecture and Science, pp. 25–57 (2002)
18. Ramos, V., Almeida, F.: Artificial ant colonies in digital image habitats - a mass behaviour effect study on pattern recognition. In: From Ant Colonies to Artificial Ants - 2 nd Int. Wkshp on Ant Algorithms. pp. 113–116 (2000)
19. Ross, B.J., Ralph, W., Hai, Z.: Evolutionary image synthesis using a model of aesthetics. In: Proceedings of the 2006 IEEE Congress on Evolutionary Computation. pp. 1087–1094. IEEE Press, Vancouver, BC, Canada (2006)
20. Schlechtweg, S., Germer, T., Strothotte, T.: Renderbots – multi agent systems for direct image generation. Computer Graphics Forum 24(2), 283–290 (2005)
21. Semet, Y., O'Reilly, U.M., Durand, F.: An interactive artificial ant approach to non-photorealistic rendering. In: Genetic and Evolutionary Computation - GECCO 2004, Genetic and Evolutionary Computation Conference, Seattle, WA, USA, June 26-30, 2004, Proceedings, Part I. LNCS, vol. 3102, pp. 188–200 (2004)
22. Tzafestas, E.: Integrating drawing tools with behavioral modeling in digital painting. In: Ghandeharizadeh, S., Chang, S.F., Fischer, S., Konstan, J.A., Nahrstedt, K. (eds.) ACM Multimedia Workshops. pp. 39–42. ACM Press (2000)
23. Urbano, P.: Playing in the pheromone playground: Experiences in swarm painting. In: EvoWorkshops. LNCS, vol. 3449, pp. 527–532. Springer (2005)
24. Urbano, P.: The *t. albipennis* sand painting artists. In: EvoApplications (2). Lecture Notes in Computer Science, vol. 6625, pp. 414–423. Springer (2011)
25. Yip, C.: Evolving Image Filters. Master's thesis, Imperial College of Science, Technology, and Medicine (2004)