Fitness Functions for Ant Colony Paintings

Penousal Machado and Hugo Amaro
CISUC, Department of Informatics Engineering
University of Coimbra
3030 Coimbra, Portugal
machado@dei.uc.pt, hamaro@student.dei.uc.pt

Abstract

A creativity-support tool for the creation of non-photorealistic renderings of images is described. It employs an evolutionary algorithm that evolves the parameters governing the behavior of ant species, and the paintings are produced by simulating the behavior of these artificial ants. The design of fitness functions, using both behavioral and image features is discussed, emphasizing the rationale and intentions that guided the design. The analysis of the experimental results obtained by using different fitness functions focuses on assessing if they convey the intentions of the fitness function designer.

Introduction

Machado and Pereira (2012) presented a non-photorealistic rendering (NPR) algorithm inspired on ant colony approaches: the trails of artificial ants were used to produce a rendering of an original input image. One of the novel characteristics of this algorithm is the adoption of scalable vector graphics, which contrasts with the pixel based approaches used in most ant painting algorithms, and enables the creation of resolution independent images. The trail of each ant is represented by a continuous line of varying width, contributing to the expressiveness of the NPRs.

In spite of the potential of this generative approach, the number of parameters controlling the behavior of the ants and their interdependencies was soon revealed to be too large to allow their tuning by hand. The results of these attempts revealed that only a small subset of the creative possibilities allowed by the algorithm was being explored.

To tackle this problem, Machado and Pereira (2012) presented a human-in-the-loop Genetic Algorithm (GA) to evolve the parameters, allowing the users to guide the algorithm according to their preferences and avoiding the need to understand the intricacies of the algorithm. Thus, instead of being forced to perform low-level changes, the users of this creativity-support tool become breeders of species of ants that produce results that they find valuable. The experimental results highlight the range of imagery that can be evolved by the system showing its potential for the production of large-format artworks.

This paper describes a further step in the automation of the space exploration process and departure from low-level modification and assessment. The users become designers of fitness functions, which are used to guide evolution, leading to results that are consistent with the user intentions. To this end, while the ants paint, statistics describing their behavior are gathered. Once each painting is completed image features are calculated. These behavioral and image features are the basis for the creation of the fitness functions.

Human-in-the-loop in evolutionary art systems are often used as creativity-support tools and thought to have the potential for exploratory creativity. Allowing the users to design fitness functions by specifying desired combinations of characteristics provides an additional level of abstraction, enabling the users to focus on their intents and overcoming the user fatigue problem. Additionally, this approach opens the door for evaluating the system by comparing the intents of the user with the outcomes of the process.

We begin with a short survey of related work. Next, in the third section, we describe the system, focusing on the behavior of the ants and on the evolutionary algorithm. In the fourth section we present experimental results, making a brief analysis. Finally, we draw some conclusions and discuss aspects to be addressed in future work.

State of the Art

In this section we make a survey of related works, focusing on systems that use artificial ants for image generation purposes and on systems where evolutionary computation is employed for NPR purposes.

Tzafestas (2000) presents a system where artificial ants pick-up and deposit food, which is represented by paint, and studies the self-regulation properties and complexity of the system and resulting images. Ramos and Almeida (2000) 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 and smooth transitions between different images. The artistic potential of these approaches is explored in later works (Ramos 2002) and thorough his collaboration with the artist Leonel Moura, resulting in several robotic swarm drawings (Moura 2002). Urbano (2005; 2007; 2011) presents several multi-agent systems based on artificial ants.

Aupetit et al. (2003) introduce an interactive 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 pro-

Figure 1: Screenshot of the graphic user interface. Control panel on the left and current population of ant paintings on the right.

ducing a painting. In a later study, Monmarché et al. (2007) refine this approach exploring different rendering modes. Greenfield (2005) 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 Greenfield (2006) 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.

The use of evolutionary algorithms to create image filters and NPRs of source images has been explored by several researchers. Focusing on the works where there was an artistic goal, we can mention the research of: Ross et al. (2006) and Neufeld et al. (2007), where Genetic Programming (GP), multi-objective optimization techniques, and an empirical model of aesthetics are used to automatically evolve image filters; Lewis (2004), evolves live-video processing filters through interactive evolution; Machado et al. (2002), use GP to evolve image coloring filters from a set of examples; Yip (2004) employs GAs to evolve filters that produce images that match certain features of a target image; Collomosse (2006; 2007) 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 (2003) use GP to evolve procedural textures for 3D objects; Machado and Graça (2008) employ GP to evolve assemblages of 3D objects that are an artistic representation of an input image.

The Framework

The system is composed of two main modules: the evolutionary engine and the painting algorithm. A graphic user interface gives access to these modules (see Fig. 1). 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 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.

In the following sections we describe the framework. First, we present the painting algorithm. Next, we describe the evolutionary component. Finally, we detail the behavioral and image features that are gathered.

Figure 2: On the left, an ant with five sensory vectors. On the middle, the living canvas of an ant species. On the right, its painting canvas.

The Painting Algorithm

Our ants live on the 2D world provided by the input image and they paint on a painting canvas that is initially empty (i.e., black). Both living and painting canvas have the same dimensions and the ants move simultaneously on both canvas. The painting canvas is used exclusively for depositing ink and has no interference with the behavior of the ants. Each ant has a position, color, deposit transparency and energy; all the remaining parameters are shared by the entire species. If the energy of an ant is bellow a given threshold it dies, if is is above a given threshold it generates offspring.

The luminance of an area of the living canvas represents the available energy, i.e. food, at that point. Therefore, ants may gain energy by traveling through bright areas. The energy consumed by the ant is removed from the living canvas, as will be explained later in detail.

The ants' movement is determined by how they react to light. Each ant senses the environment by "looking" in several directions (see Fig. 2). We use 10 sensory vectors, each vector has a given direction relative to the current direction of the ant and a length. The 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 scalar 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 (1985) noise to its angle.

The ant simulation algorithm is composed of the following steps:

1. Initialization: n ants are placed on the canvas on pre-established positions; Each ants assumes the color of the area where it was placed; Their energy and deposit transparencies are initialized using the species parameters;
2. For each ant:
   (a) Update the ant's energy;
   (b) Update the energy of the environment;
   (c) Place ink on the painting canvas;
   (d) If the ant's energy is bellow the death threshold remove the ant from the colony;
   (e) If the ant's energy is above the reproduction threshold generate an offspring; The offspring assumes the color of the position where it was created and a percentage of the energy of the progenitor (which loses this energy); The offspring inherits the velocity of the parent, but a perturbation is added to the angular velocity by randomly choosing an angle between $descvel_{min}$ and $descvel_{max}$ (both values are species' parameters); Likewise, the deposit transparency is inherited from the progenitor but a perturbation is included by adding a randomly choosen a value between $dtransp_{min}$ and $dtransp_{max}$;
   (f) Update ant's position;
3. Repeat from 2 until no living ants exist;

Steps (b) and (c) require further explanation. The consumption of energy is attained by drawing on the living canvas a black circle of size equal to $energy * cons_{rate}$ of a given transparency ($cons_{trans}$) . Ink is deposited on the paining canvas by drawing a circle of the color of the ant – which is attributed when the ant is born – with a size given by ($energy * deposit_{rate}$) and of given transparency ($deposit_{transp}$). Fig. 2 depicts the living and painting canvas of an ant species during the simulation process. It is important to notice that the color of an ant is determined at birth. 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 painting algorithm can be found in Machado and Pereira (2012).

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 living canvas
$cons_{trans}$	1	transparency of circles drawn on the living canvas
$deposit_{rate}$	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 offsprings are generated
$dtransp_{max}$	1	
$initial_{energy}$	1	initial energy of the starting ants
$death_{threshold}$	1	death energy treshold
$birth_{threshold}$	1	generate offspring energy threshold
$descvel_{min}$	1	limits for perturbation of angular velocity when offsprings are generated
$descvel_{max}$	1	
vel	1	base speed of the ants
$noise_{min}$	1	limits for the perlin noise generator function
$noise_{max}$	1	
$initial_{positions}$	$2 * n$	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

Evolutionary Engine

As previously mentioned, we employ a GA to evolve the ant species' parameters. The genotypes are tuples of floating point numbers which encode the parameters of the ant species. The size of the genotype depends on the experimental settings. 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.

The Features

During the simulation of each ant species the following behavioral statistics are collected:

$avg(ants)$ Average number of living ants;

$coverage$ Proportion of the living canvas visited by the ants; An area is considered to be visited if, at least, one ant consumed resources from that area;

$deposited_{ink}$ The total amount of "ink" deposited by the ants; This is calculated by multiplying the area of each circle drawn by the ants by the opacity (i.e. $1 - transparency$) used to draw it.

$avg(trail), std(trail)$ The average trail length and the standard deviation of the trail lengths, respectively;

$avg(life), std(life)$ The average life span of the ants and its standard deviation, respectively;

$avg(distance)$ The average euclidean distance between the position where the ant was born and the one where it died;

$avg(avg(width)), std(avg(width))$ For each trail we calculate its average width, then we calculate the average width of all trails, $avg(avg(width))$, and the standard deviation of the averages, $std(avg(width))$;

$avg(std(width)), std(std(width))$ For each trail we calculate the standard deviation of its width, then we calculate their average, $avg(std(width))$, and their standard deviation $std(std(width))$;

$avg(avg(av)), std(avg(av)), avg(std(av)), std(std(av))$ These statistics are analogous to the ones regarding trail width, but pertaining to the angular velocity of the ants;

When the simulation of each ant species ends we calculate the following image features:

$complexity$ the image produced by the ants, *I*, is encoded in $jpeg$ format, and its complexity estimated using the following formula:

$$complexity(I) = rmse(I, jpeg(I)) \times \frac{s(jpeg(I))}{s(I)},$$

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

$fract_{dim}, lac$ The fractal dimension of the ant painting estimated by the box-counting method and its λ lacunarity value estimated by the Sliding Box method (Karperien 2012), respectively;

$inv(rmse)$ The similarity between the ant painting and the original image estimated as follows:

$$inv(rmse) = \frac{1}{1 + rmse(I, O)},$$

where I is the ant painting and O is the original image;

Experimental results

The results presented in this section were obtained using the following experimental setup: Population Size = 25; Tournament size = 5; Crossover probability = 0.9; Mutation Probability = 0.1 (per gene); Initial Position of the ants = the image is divided in 3×3 rectangles of the same size and one ant is placed at the center of each of these rectangles; Initial number of ants = 9; Maximum number of ants = 250; Maximum number of simulation steps 1000. Thus, when the drawing stage starts each ant species is represented by nine ants. However, these ants may generate offspring during simulation, increasing the number of ants in the canvas.

Typically, interactive runs had 30 to 40 generations, although some were significantly longer. The runs conducted using explicit fitness functions lasted 50 generations. For each fitness function we conducted 10 independent runs.

User Guided Runs

Machado and Pereira (2012) describe and analyze results attained in the course of user guided runs. In Fig. 3 we depict some of the individuals evolved in those runs, with the goal of giving a flavor of the different types of imagery that were evolved.

Figure 3: Examples from user guided runs.

Using Features Individually

To test the evolutionary algorithm we performed runs where each feature, with the exception of $frac_{dim}$ and lac, was used as fitness function. Maximizing the values of fractal dimension and lacunarity would lead to results that we find uninteresting. Therefore, we established for these features by measuring the fractal dimension and lacunarity of one of our favorite ant paintings evolved in user guided runs, 1.5 and 0.95, respectively, and the maximum fitness is obtained when these values are reached. For these two features, fitness is assigned by the following formula:

$$fitness = \frac{1}{1 + |target_{value} - feature_{value}|}$$

In Fig. 4 we present the evolution of fitness across the evolutionary runs. To avoid clutter we only present a subset of the considered fitness functions. In general, the evolutionary algorithm was able to find, in all runs and for all features, individuals with high fitness in relatively few generations. Unsurprisingly, and although it is subjective to say it, the runs tended to converge to ant paintings that, at least in our eyes, are inferior to the ones created in the course of interactive runs. Fig. 5 depicts the individuals that obtained the maximum fitness value for the corresponding image features. These individuals are representative of the imagery evolved in the corresponding runs.

It worth to notice that high $complexity$ is obtained by evolving images with abrupt transitions from black to white. This results in high frequencies that make $jpeg$ compression inefficient, thus resulting in high complexity estimates. The results attained with $lacunarity$ yield paintings with "gaps" between lines, revealing the black background,

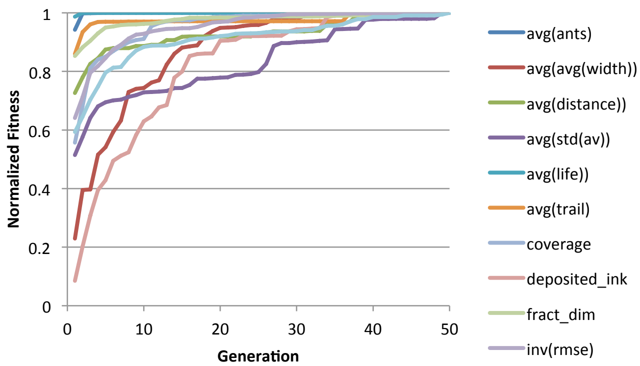

Figure 4: Evolution of the maximum fitness. The results are averages of 10 independent runs. The results have been normalized to allow the presentation of the results using distinct fitness functions in the same chart.

which matches the texture of the image from where the target $lacunarity$ value was collected. This contrasts with the results obtained using $fract_{dim}$, while the algorithm was able to match the target fractal dimension value, the images produced are radically different from the target's image. The $inv(rmse)$ runs revealed images that reproduce the original with some degree of fidelity, showing that this feature can promote similarity between the painting and the original.

The results obtained using a single behavioral feature are uninteresting in the context of NPR. They tend to fall in two categories, either they constitute "poor" variations of the original or they are unrecognizable versions of it.

Combining Behavioral and Image Features

From the beginning it was clear that it would be necessary to combine several features to attain our goals. To make the fitness function design process easy to understand, and thus allow inexperienced users to design their own fitness functions, we decided that all fitness functions should assume the form of a weighted sum.

Since different features have different ranges of values, it is necessary to normalize them, otherwise some features would outweigh the others. Additionally, angular velocity may be negative, so we should consider the absolute values. Considering these issues, normalization is attained by the following formula:

$$norm(feature) = abs\left(\frac{feature}{offlinemax(feature)}\right),$$

where $offlinemax$ returns the maximum value found in the course of the runs described in the previous section for the feature in question.

This modification is not sufficient to prevent the evolutionary algorithm to focus exclusively on a single feature. To minimize this problem, we consider 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 as follows:

$$lognorm(feature) = log(1 + norm(feature))$$

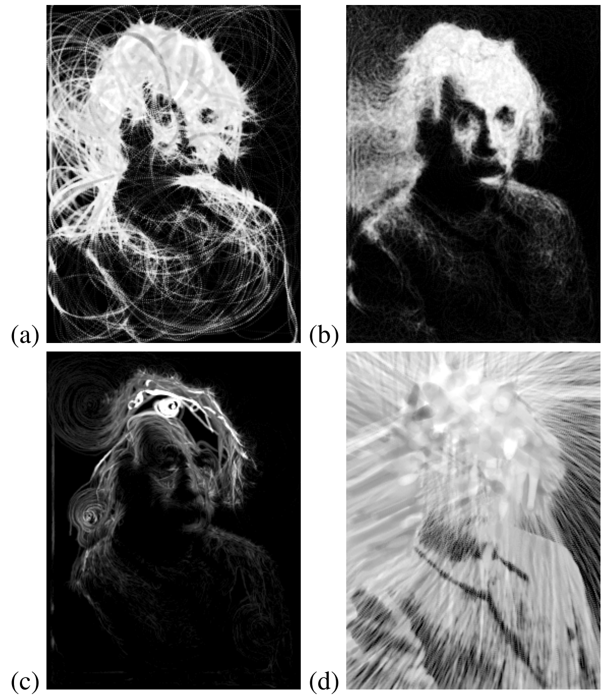

Figure 5: The individuals that obtained the maximum fitness value for: (a) $Complexity$; (b) $inv(rmse)$; (c) lac; (d) $fract_{dim}$.

All the fitness functions that combine several features are weighted sums of the $lognorm$ of each of the features used. However, for the sake of simplicity we will only mention the feature names when writing their formulas. From here onwards $feature$ should be read as $lognorm(feature)$.

Next we describe several fitness functions that combine a variable number of features. The analysis of the experimental results of evolutionary art systems is subjective by nature. As such, more than presenting measures of performance that would be meaningless when considering the goals of our system, we focus on describing the intentions behind the design of each fitness function, and make a subjective analysis of the results based on the comparison between the evolved paintings and our original design intentions.

f1: $coverage + complexity + lac$

The design of this fitness function was prompted by the results obtained in previous tests. The goal is to evolve ant paintings where the entire canvas is visited, with high $complexity$, and with a $lacunarity$ value of 0.95.

As it can be observed in Fig. 6 the evolved paintings successfully match these criteria. By comparing them with the ones presented in Fig. 5 one can observe how $lacunarity$ influences texture, $complexity$ leads high frequencies, and coverage promotes visiting most of the canvas.

f2: $inv(rmse) - 0.5 * complexity$

The rationale for this fitness function is obtaining a good approximation to the original image while keeping the complexity low. Thus, we wish to obtain a simplified version of

Figure 6: Two of the fittest images evolved using **f1**.

Figure 7: Two of the fittest images evolved using **f2**.

Figure 8: Two of the fittest images evolved using **f3**.

the original. Preliminary tests indicate the tendency of the algorithm to focus, exclusively, on minimizing $complexity$, which was achieved by producing images that were entirely black. Since this sort of image exists in the initial populations of the runs, this is a case of premature convergence. To circumvent it we decreased the weight given to $complexity$, which allowed the algorithm to escape this local optimum.

Although the results are consistent with the design (see Fig. 7) they do not depict the degree of abstraction and simplification we intended. As such, they should be considered a failure since they do not match our design intentions.

f3: $avg(std(width))+std(avg(width))-avg(avg(width))+inv(rmse)$

Here we focus on the width of the lines being drawn promoting the evolution of ant paintings with lines with high variations of width, $avg(std(width))$, heterogeneous widths among lines, $std(avg(width))$, and thin lines, $-avg(avg(width))$. To avoid radical deviations from the original drawing we also value $inv(rmse)$.

The experimental results, Fig. 8, depict these characteristics, however to fully observe the intricacies of the ant paintings a resolution higher than the space constraints of this paper allows would be required.

Figure 9: Two of the fittest images evolved using **f4** (first row), **f5** (second row) and **f6** (third row).

f4: $avg(std(av)) + inv(rmse) + coverage$
f5: $avg(avg(av)) - avg(std(av)) + inv(rmse) + coverage$
f6: $-avg(avg(av)) + avg(std(av)) + inv(rmse) + coverage$

When designing **f4-f6** we focused on controlling line direction. In **f4** we use $avg(std(av))$ to promote the appearance of lines that often change direction. In **f5** we use $avg(avg(av)) - avg(std(av))$ to encourage the appearance of circular motifs (high angular velocity and low variation of velocity). Finally, **f6** is a refinement of **f4** with

$-avg(avg(av))$ preventing the appearance of circular patterns, valuing trails that curve in both directions, attaining an average angular velocity close to zero.

In all cases, the addition of $inv(rmse)$ and $coverage$ serves the goal of evolving ant paintings with some similarity to the original and that visit a large portion of the canvas.

In Fig 9 we present some of the outcomes of this experiences. As it can be observed the evolved images closely match our expectations and, as such, we consider them to be some of the most successful runs.

Once the individuals are evolved the ant species may be applied to different input images, hopefully resulting in ant-paintings that share the characteristics that we value. This is one of the key aspects of the system: although finding a valuable ant species may be time consuming, once it is found it can be applied with ease to other images producing large-scale NPR of them. In Fig. 10 we present ant paintings created by this method.

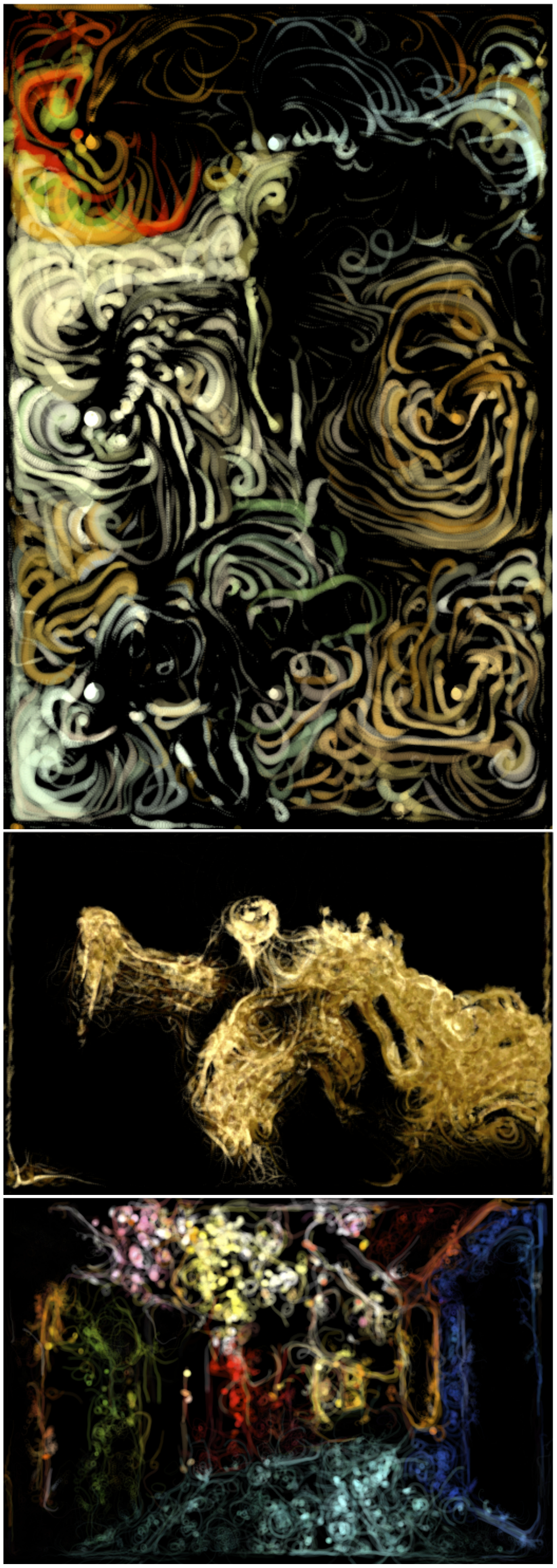

Figure 10: Results obtained by applying an individual from the **f4** runs to different input images.

Conclusions

We presented a creativity-support tool that aids the users by providing a wide variety of paintings, which are arguably consistent with the intentions of the users, and which they would be unlikely to imagine on their own. While using this tool the users become designers of fitness functions, which are built using a combination of behavioral and image features. We reported the results obtained, focusing on the comparison between the evolved ant-paintings and the design intentions that led to the development of each fitness function.

Overall the results indicate that it is possible, to some extent, to convey design intention through fitness functions, leading to the discovery of individuals that match these intentions. This allows the users to operate at a higher level of abstraction than in user guided runs, circumventing the user-fatigue problem typically associated with interactive evolution. The analysis of the results also reveals the discovery of high-quality ant paintings that are radically different from the ones obtained through interactive evolution.

Although the system serves the user intents, different runs converge to different, and sometimes highly dissimilar, images. Each fitness function can be maximized in a multitude of ways, some of which are quite unexpected. As such, we argue that the system opens the realm of possibilities that are consistent with the intents expressed by the user, often surprising him/her in the process.

On the downside, as the **f2** runs reveal, in some cases the design intentions are not fully conveyed by the evolved ant paintings. It is also worth mentioning that interactive runs allow opportunistic reasoning, which may allow the discovery of unexpected and highly valued ant paintings.

The adoption of a semi-automatic fitness assignment scheme, such as the one presented by Machado et al. (2005), is one of the directions for further research. It also become obvious that we only began to scratch the vast number of possibilities provided by the design of fitness functions. In the future, we will invite users that are not familiar with the system to design their own fitness functions, which will allow us to assess the difficulty of the task for regular users.

Acknowledgements

This research is partially funded by the Portuguese Foundation for Science and Technology in the scope of project SBIRC (PTDC/EIA–EIA/115667/2009) and of the 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

Aupetit, S.; Bordeau, V.; Monmarché, N.; Slimane, C.; and Venturini, G. 2003. Interactive Evolution of Ant Paintings. In *IEEE Congress on Evolutionary Computation*, volume 2, 1376–1383.

Collomosse, J. P. 2006. Supervised genetic search for parameter selection in painterly rendering. In *Applications of Evolutionary Computing, EvoWorkshops 2006*, 599–610.

Collomosse, J. 2007. Evolutionary search for the artistic rendering of photographs. In Romero, J., and Machado, P., eds., *The Art of Artificial Evolution: A Handbook on Evolutionary Art and Music*. Springer Berlin Heidelberg. 39–62.

Greenfield, G. 2005. Evolutionary methods for ant colony paintings. In *Applications of Evolutionary Computing, EvoWorkshops2005: EvoBIO, EvoCOMNET, EvoHOT, EvoIASP, EvoMUSART, EvoSTOC*, volume 3449 of *LNCS*, 478–487. Lausanne, Switzerland: Springer Verlag.

Greenfield, G. 2006. Ant Paintings using a Multiple Pheromone Model. In *Bridges*.

Hewgill, A., and Ross, B. J. 2003. Procedural 3d texture synthesis using genetic programming. *Computers and Graphics* 28:569–584.

Karperien, A. 2012. Fraclac for imagej, version 2.5. In *http://rsb.info.nih.gov/ij/plugins/fraclac/FLHelp/Introduction.htm*.

Lewis, M. 2004. Aesthetic video filter evolution in an interactive real-time framework. In *Applications of Evolutionary Computing, EvoWorkshops2004: EvoBIO, EvoCOMNET, EvoHOT, EvoIASP, EvoMUSART, EvoSTOC*, volume 3005 of *LNCS*, 409–418. Coimbra, Portugal: Springer Verlag.

Machado, P., and Graça, F. 2008. Evolutionary pointillist modules: Evolving assemblages of 3d objects. In *Applications of Evolutionary Computing, EvoWorkshops 2008: EvoCOMNET, EvoFIN, EvoHOT, EvoIASP, EvoMUSART, EvoNUM, EvoSTOC, and EvoTransLog, Naples, Italy, March 26-28, 2008. Proceedings*, volume 4974 of *Lecture Notes in Computer Science*, 453–462. Springer.

Machado, P., and Pereira, L. 2012. Photogrowth: non-photorealistic renderings through ant paintings. In Soule, T., and Moore, J. H., eds., *Genetic and Evolutionary Computation Conference, GECCO '12, Philadelphia, PA, USA, July 7-11, 2012*, 233–240. ACM.

Machado, P.; Romero, J.; Cardoso, A.; and Santos, A. 2005. Partially interactive evolutionary artists. *New Generation Computing – Special Issue on Interactive Evolutionary Computation* 23(42):143–155.

Machado, P.; Dias, A.; and Cardoso, A. 2002. Learning to colour greyscale images. *The Interdisciplinary Journal of Artificial Intelligence and the Simulation of Behaviour – AISB Journal* 1(2):209–219.

Monmarché, N.; Mahnich, I.; and Slimane, M. 2007. Artificial art made by artificial ants. In Romero, J., and Machado, P., eds., *The Art of Artificial Evolution: A Handbook on Evolutionary Art and Music*. Springer Berlin Heidelberg. 227–247.

Moura, L. 2002. Swarm paintings – non-human. In *ARCHITOPIA Book, Art, Architecture and Science*. 1–24.

Neufeld, C.; Ross, B.; and Ralph, W. 2007. The evolution of artistic filters. In Romero, J., and Machado, P., eds., *The Art of Artificial Evolution: A Handbook on Evolutionary Art and Music*. Springer Berlin Heidelberg. 335–356.

Perlin, K. 1985. An image synthesizer. In Cole, P.; Heilman, R.; and Barsky, B. A., eds., *Proceedings of the 12st Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1985*, 287–296. ACM.

Ramos, V., and Almeida, F. 2000. Artificial ant colonies in digital image habitats - a mass behaviour effect study on pattern recognition. In *In Dorigo, M., Middendorf, M., Stuzle, T. (Eds.): From Ant Colonies to Artificial Ants - 2 nd Int. Wkshp on Ant Algorithms*, 113–116.

Ramos, V. 2002. On the implicit and on the artificial - morphogenesis and emergent aesthetics in autonomous collective systems. In *ARCHITOPIA Book, Art, Architecture and Science*. 25–57.

Ross, B. J.; Ralph, W.; and Hai, Z. 2006. Evolutionary image synthesis using a model of aesthetics. In Yen, G. G.; Lucas, S. M.; Fogel, G.; Kendall, G.; Salomon, R.; Zhang, B.-T.; Coello, C. A. C.; and Runarsson, T. P., eds., *Proceedings of the 2006 IEEE Congress on Evolutionary Computation*, 1087–1094. Vancouver, BC, Canada: IEEE Press.

Tzafestas, E. 2000. Integrating drawing tools with behavioral modeling in digital painting. In Ghandeharizadeh, S.; Chang, S.-F.; Fischer, S.; Konstan, J. A.; and Nahrstedt, K., eds., *ACM Multimedia Workshops*, 39–42. ACM Press.

Urbano, P. 2005. Playing in the pheromone playground: Experiences in swarm painting. In Rothlauf, F.; Branke, J.; Cagnoni, S.; Corne, D. W.; Drechsler, R.; Jin, Y.; Machado, P.; Marchiori, E.; Romero, J.; Smith, G. D.; and Squillero, G., eds., *EvoWorkshops*, volume 3449 of *Lecture Notes in Computer Science*, 527–532. Springer.

Urbano, P. 2007. Mimetic variations on stigmergic swarm paintings. In Monmarché, N.; Talbi, E.-G.; Collet, P.; Schoenauer, M.; and Lutton, E., eds., *Artificial Evolution*, volume 4926 of *Lecture Notes in Computer Science*, 62–72. Springer.

Urbano, P. 2011. The *t. albipennis* sand painting artists. In *EvoApplications (2)*, volume 6625 of *Lecture Notes in Computer Science*, 414–423. Springer.

Yip, C. 2004. Evolving Image Filters. Master's thesis, Imperial College of Science, Technology, and Medicine.