Evolving Ambiguous Images

Penousal Machado[1] and **Adriano Vinhas**[1] and **João Correia**[1] and **Aniko Ekárt**[2]
[1] CISUC, Department of Informatics Engineering
University of Coimbra, Coimbra, Portugal
[2]Aston Lab for Intelligent Collectives Engineering (ALICE), Computer Science
Aston University, Birmingham, United Kingdom
machado@dei.uc.pt, avinhas@student.dei.uc.pt, jncor@dei.uc.pt, a.ekart@aston.ac.uk

Abstract

This work explores the creation of ambiguous images, i.e., images that may induce multistable perception, by evolutionary means. Ambiguous images are created using a general purpose approach, composed of an expression-based evolutionary engine and a set of object detectors, which are trained in advance using Machine Learning techniques. Images are evolved using Genetic Programming and object detectors are used to classify them. The information gathered during classification is used to assign fitness. In a first stage, the system is used to evolve images that resemble a single object. In a second stage, the discovery of ambiguous images is promoted by combining pairs of object detectors. The analysis of the results highlights the ability of the system to evolve ambiguous images and the differences between computational and human ambiguous images.

1 Introduction

In the field of Evolutionary Art (EA), expression-based systems are, in theory, capable of generating any image of any kind [Machado and Cardoso, 2002; McCormack, 2007]. From a practical point of view, the evolved images depend on the representation scheme in use. Consequently, the results of expression-based EA systems tend to be abstract images. However, since the start of EA, there has been a desire to evolve figurative images. One of the first attempts can be found in the work of Rooke [World, 1996].

Machado and Correia [2012a; 2013] proposed and explored the combination of a general-purpose expression-based GP image generation engine with off-the-shelf object detectors, as well as their own ones, for the purpose of evolving figurative images. The results showed the ability of the system to evolve images containing the desired objects. More precisely, the system was able to evolve images that were classified by the object detector as containing the desired object, although in some cases this object was not clearly recognizable by a human viewer.

This work builds upon the findings of Machado and Correia [2012a; 2013] expanding their approach in order to evolve ambiguous images. In other words, our goal is to evolve images that induce multistable perception, which occurs when the brain (or the computer in our case) is confronted with visual stimulus that can be interpreted in multiple ways. Some famous examples of ambiguous images are: duck/rabbit; Rubin's vase, which can be perceived as a vase or two opposing faces; "My Wife and My Mother-in-Law", which may be interpreted as a young or an old woman (see Figure 1).

We consider ambiguous images and multistable perception fascinating phenomena, worth studying for both scientific and artistic purposes. Some of the questions that motivate the research reported here are: (i) Can ambiguous images be created by fully automated computational means? (ii) Can this be done from scratch (i.e. without resorting to collages or morphing of pre-existent images)? (iii) How do computational ambiguous images look like? (iv) How do they relate to human ambiguous images? (v) How can the dichotomy between human and computational ambiguity be explored for artistic purposes? (vi) Can one explore computer vs. human creativity and perception scientifically via ambiguous images?

To evolve ambiguous images, an incremental approach is followed. First we evolve images containing a single object. Following in the footsteps of Machado and Correia [2012a; 2013] we use an object detector to guide evolution, assigning fitness based on the internal values of the object detection process. Then, using object detectors trained to identify different types of objects, we evolve images containing two distinct objects. Finally, we focus on the evolution of ambiguous images, which is achieved by evolving images containing two distinct objects in the same window of the image.

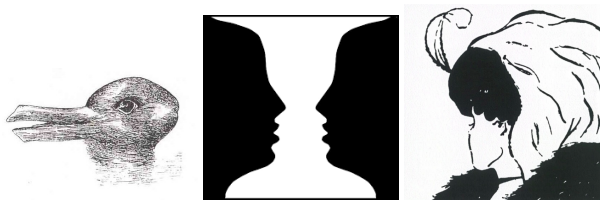

Figure 1: Some well-known examples of ambiguous images, from left to right: duck/rabbit; Rubin's vase; "My Wife and My Mother-in-Law".

The paper is structured as follows: In the next section a brief overview of related work is presented; Section 3 describes our approach for the evolution of ambiguous images, including an overview of the approach, the Genetic Programming (GP) engine, the object detection system, and the fitness assignment scheme. The experimental setup, the results attained and their analysis are presented in section 4. Finally, in section 5 we draw overall conclusions and indicate future research.

2 State of the Art

The evolution of figurative images has been an ambitious challenge pursued by several researchers since the early days of Evolutionary Art. Although many efforts have been made in this direction, most of the known approaches found in the literature focus on evolving specific types of objects, namely human faces. Baker [1993] evolved line drawings using a Genetic Algorithm (GA), where each line could have different characteristics. Johnston and Caldwell [1997] made use of GAs to evolve human faces using different portions of them, in a trial to create an artist capable of constructing criminal sketches. Frowd *et al.* [2004] also employ a GA to evolve human faces, using Principal Component Analysis and eigenfaces.

While some authors tried to evolve photographic human faces, others tried to evolve cartoon faces [Nishio *et al.*, 1997] and cartoon faces animations [Lewis, 2007] using GAs. In the latter, the evolution of human figures was also explored.

In contrast to the works described above, Ventrella [2010] and DiPaola and Gabora [2009] built a general purpose evolutionary art tool. They addressed the problem of evolving images towards a given goal image, as suggested by Sims [1991]. This problem can be interpreted as a symbolic regression problem where the fitness function is a similarity degree between the evolved images and the target image. For DiPaola and Gabora [2009], the final goal was to evolve abstract portraits of Darwin. Despite taking into account expressiveness in their fitness function, which gave sense to their work from an artistic point of view, the results were disappointing in terms of resemblance to the goal object, considering figurative art.

In interactive evolutionary art, Rooke [1996] evolved several images reminiscent of African masks, Machado and Cardoso [2002] evolved images that are evocative of female bodies, faces, etc. More recently, Secretan *et al.* [2008] created an user-guided collaborative evolutionary engine, named picbreeder. Some of the images evolved by the users resemble figurative images such as cars, butterflies and flowers.

3 Overview of the Approach

The approach is based on the framework of Machado and Correia [Machado *et al.*, 2012a; Correia *et al.*, 2013]. Figure 2 presents an overview of the framework, which is composed of two main modules, an evolutionary engine and a classifier. The execution of the framework proceeds as follows:

1. Randomly initialize the population;
2. Render the individuals, i.e., genotype-phenotype mapping;

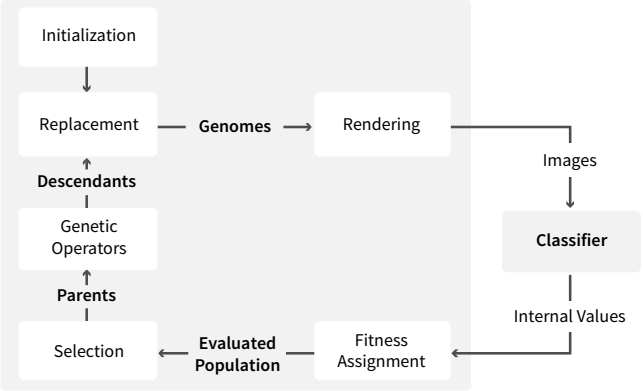

Figure 2: Overview of the system.

3. Apply the classifier to each phenotype;
4. Use the results of the classification to assign fitness; This requires assessing internal values and intermediate results of the classification, to assign fitness;
5. Select progenitors; Apply genetic operators, create descendants; Use the replacement operator to update the current population;
6. Repeat from step 2 until some stopping criterion is met.

For the purpose of this paper, the framework was instantiated with a general-purpose GP-based image generation engine – described in section 3.1 – and with a cascade classifier as an object detector – described in section 3.2. To create a fitness function able to guide evolution it is necessary to convert the binary classification output of the object detector to one that can provide a suitable fitness landscape. This is achieved by accessing internal results of the classification task that provide an indication of the degree of certainty in the classification. This process is described in section 3.3.

3.1 Genetic Programming Engine

The GP-based image generation engine used in this experiment is inspired by the work of Sims [1991], and is similar to the work of Machado [2007]. It was built upon ECJ,[1] a java-based Evolutionary Computation research system, that contains several evolutionary algorithms and primitives. It is a general purpose, expression-based, Genetic Programming image generation engine that allows the evolution of populations of images. The genotypes are expression trees where the functions include mathematical and logical operations and the terminal set is composed of two variables, x and y, and random constant values. The phenotypes are images, rendered by evaluating the expression trees for different values of x and y, which serve both as terminal values and image coordinates. In other words, the value of the pixel of coordinates (i, j) is calculated by assigning i to x and j to y and evaluating the expression tree.

[1] ECJ – `http://cs.gmu.edu/ eclab/projects/ecj/`

3.2 Object Detection

The object detector used in the framework is based on the work of Viola and Jones [2001]. Two object detectors were trained, by building datasets of faces and flowers. This training procedure was attained using OpenCV API.[2] For the datasets for each classifier we specified a negative dataset, composed of images that did not contain any of the objects being evolved and was common for the trained object detectors. Additionally, a positive dataset was used according to the desired object detector that we wanted to produce, where each image had an annotation regarding the object location. The process was similar to the one used by Correia *et al.* [2013] where more insight about building the datasets can be found. After the training phase, the object detector can be used to detect whether an image contains an object. The detection algorithm can be summarized in the following steps:

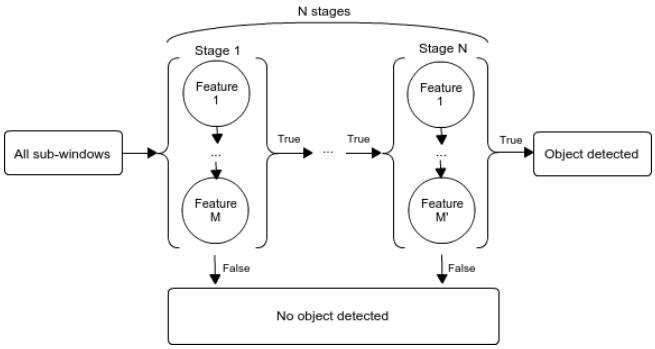

Figure 3: Cascade of classifiers with N stages.

1. Define a window of size w (e.g. 20×20).
2. Define a scale factor s greater than 1. For instance 1.2 means that the window will be enlarged by 20%.
3. Define W and H as the width and height, respectively, of the input image.
4. From $(0, 0)$ to (W, H) define a window of the image with a starting size of w for calculation.
5. For each window apply the cascade classifier. The cascade has a group of weak stage classifiers, as presented in Figure 3. Each stage is composed, at its lower level, of a group of low level features (Haar features). Apply each feature of each stage to the window. If the overall resulting value is lower than the stage threshold the window does not have an object and the search terminates for the window. If it is higher, the search continues to the next stage. If all stages are passed, the window contains an object.
6. Apply the scale factor s to the window size w and repeat step 5 until window size exceeds the image in at least one dimension.

This algorithm has key aspects that have impact on the evolutionary process: (i) the whole process is a summation of multiple binary classifications on windows of the input image; (ii) for each window, the detection involves multiple stage classifiers and feature extraction. Aspect (i) allows us to explore different parts of the individual and evaluate its potential. Aspect (ii) provides us with access to several internal values of the classifier that can be used to assign fitness. Both aspects allow the design of an appropriate fitness function to guide evolution which is described in the following section.

[2]OpenCV – `http://opencv.org/`

3.3 Fitness Assignment

An important aspect of the framework, as for any Evolutionary Computation algorithm, is the fitness assignment scheme. The fitness assignment process determines the rank given to each evolved solution, determining which individuals should survive for the next generation. Thus, one must design a fitness function able to guide evolution. In our particular case, it is necessary to convert the binary output of the object detector(s) to one that can provide a suitable fitness landscape. This is achieved by accessing internal results of the classification task. As such, images that are immediately rejected by the classifier will have lower fitness values than those that are close to the detection of the object. Based on the work of Machado and Correia [2012a; 2013], we applied the following fitness function:

$$f(x) = \sum_{i}^{cstages_x} (stagedif_x(i) * i) + cstages_x * 10. \quad (1)$$

Variables $cstages_x$ and $stagedif_x(i)$ are extracted from the object detection algorithm. The rationale is that an image that passes several stages has a higher $cstages$ value and is likely to be closer to being recognized as having a object than one that passes fewer stages. Images that are clearly above the thresholds associated with each stage have higher $stagedif$ values. As such, these images are preferred over ones that are only slightly above the threshold. More details regarding the fitness function are available in [Machado *et al.*, 2012a].

Since we are interested in the evolution of images that are evaluated by several object detectors, the combined fitness function described in Equation 2 was used:

$$combined(x) = \prod \log_2(f_i(x) + 2), \quad (2)$$

where $f_i(x)$ is the ith single fitness function (1), per classifier of the combination.

4 Experimentation

The experimentation was divided into three steps. First we reproduced the previous work of [Correia *et al.*, 2013], by performing evolutionary runs with a single object detector. Next, we combined the two detectors to test the ability of our approach to evolve images that simultaneously depict both objects. Finally, we combined the detectors enforcing an overlap between the objects, hoping to achieve ambiguous images.

4.1 Experimental Setup

Table 1 describes the parameters used in the GP engine while Table 2 contains the parameters of the object detection algorithm.

Table 1: Parameters of the GP engine.

Parameter	Setting
Population size	100
Number of generations	1000
Crossover probability	0.8 (per individual)
Mutation probability	0.05 (per node)
Mutation operators	sub-tree swap, sub-tree replacement, node insertion, node deletion and mutation
Initialization method	ramped half-and-half
Initial maximum depth	5
Mutation max tree depth	3
Function set	$+$, $-$, $\times$, $/$, min, max, abs, neg, warp, sign, sqrt, pow, mdist, sin, cos, if
Terminal set	x, y, random constants

Table 2: Detection parameters used.

Parameter	Setting
Min. window width	90
Min. window height	90
Image Width	128
Image Height	128
Scale Factor	1.1
Image pre-processing	Otsu's Binarization

Regarding the general object detection parameters, there are some differences from the approach of Machado and Correia [2012a; 2013]. The size of the individuals rendering, in pixels, for evaluation is substantially increased from 64×64 to 128×128. This forces the evolved individuals to contain larger objects, and be more robust and less noisy. The minimum size window, or minimum object size, was also increased to promote the appearance of objects at the center of the image. Another relevant difference is that the images are all pre-processed, for training or detection, with a binarization algorithm. We used Otsu's binarization algorithm [1979] to transform the images. This way, we highlight the difference between the object and everything around it. We conducted tests with and without binarization of the output of the GP system.[3] Overall, we found that binarized images tended to be clearer to humans. This is likely to be related to the image equalization and "normalization" operations performed by the object detectors before classification, which may highlight features that are hard to see in the original images.

Three different experimental environments were prepared for this work. In the first one we used the two classifiers – faces and flowers – individually to guide evolution. We then focused on the evolution of images that could simultaneously evoke faces and flowers. In the second experimental setting, we assigned fitness based on the results of the face and flower detectors and we reduced the minimum window size parameter to (40×40). This promoted the evolution of images containing both objects, but it did not require the faces and the flowers to overlap. Finally, in the third experimental setting, we used both classifiers to assign fitness and a minimum window size of (90×90), which forced an overlap between the windows detecting both objects. For each combination of parameters, we performed 30 independent evolutionary runs using different random seeds. To promote readability we normalised all fitness values by dividing the raw value by the maximum value found in the course of the experiments. This also means that the values used for the plots are averages of 30 runs.

[3] We are binarizing the output of a floating point GP system. Using a binary GP approach would avoid this intermediate step.

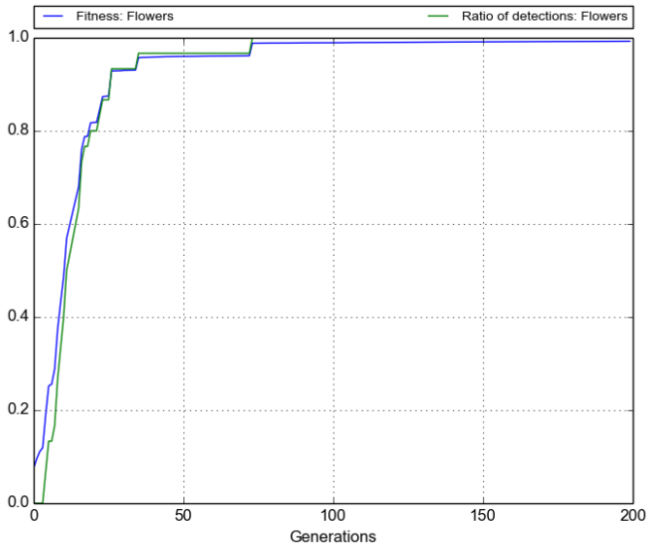

Figure 4: Evolution of the fitness of the best individual across generations and of the percentage of best individuals where a flower was detected. The results are averages of 30 runs.

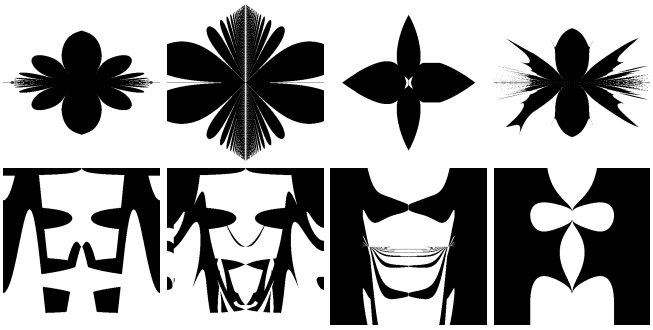

Figure 5: Some examples of evolved images containing flowers (top row) and faces (bottom row).

4.2 Experimental Results

The results obtained when evolving images containing single objects confirm previous work in this field [Machado *et al.*, 2012a; Correia *et al.*, 2013]. In all runs and for all classifiers, evolution was able to produce images where the object was detected. In most situations this was accomplished in fewer than 50 generations.

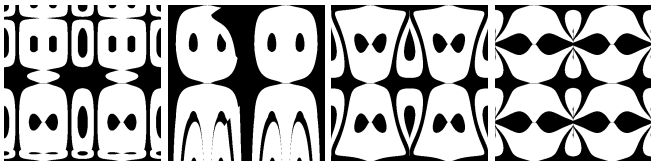

Figure 6: Examples of images containing non overlapping faces and flowers.

Figure 4 depicts the evolution of the fitness of the best individual when evolving flowers, as well as the percentage of those individuals where a flower was detected. As can be observed, by the 70th generation all runs had already produced individuals containing flowers. Also confirming previous results in the area, although all runs evolved images where the object in question was detected, the visibility of these objects to a human observer is questionable in some of the cases. Figure 5 presents some examples of the evolved images. As can be observed, while the flowers are easy to identify, seeing the faces is not obvious in all the cases. This can be explained by the fact that the detection of flowers relies heavily on the contour of the shape, while the detection of faces relies on the presence of a combination of features that can be identified as eyes, eyebrows, lips, nose, chin, face contour,[4] which may be obfuscated by other image artifacts.

We then focused on the evolution of images containing faces and flowers simultaneously, without enforcing the overlap between the regions where these objects were identified. Figure 6 depicts examples of the results obtained in this setting. In all of the examples presented, the system was able to evolve images where the object detectors found faces and flowers. Interestingly, some of the evolved images (e.g. the two rightmost images of Figure 6) depict the same type of optical illusion as Rubin's vase (see Figure 1). In this case, although we do not promote the overlap between the detection windows and although these could be completely non-overlapping, the solutions found by the EC engine often take advantage of the similarities between visual features of the objects. This is particularly evident in the rightmost image of Figure 6 where the eyes of the faces serve as petals for the flowers, and vice-versa. As such, we can state that in some of the evolutionary runs the algorithm evolved images that are ambiguous both from a computational and human perspective, in the sense that both computer and human are able to recognize simultaneously a face and a flower in the same region. As a side-note, it is also interesting to note that some of these images constitute tiling patterns, which is an unexpected outcome.

In our third experimental setting the overlap between the regions where faces and flowers are detected becomes a requirement. Figure 7 shows the evolution of the fitness of the best individual. In addition to the combined fitness value, we also present the fitness scores according to each classifier. As previously, these results have been normalized by dividing by the highest corresponding value found in the course of all the experiments. The percentage of the best individuals where

[4]Simultaneous presence of all of these features is not necessary.

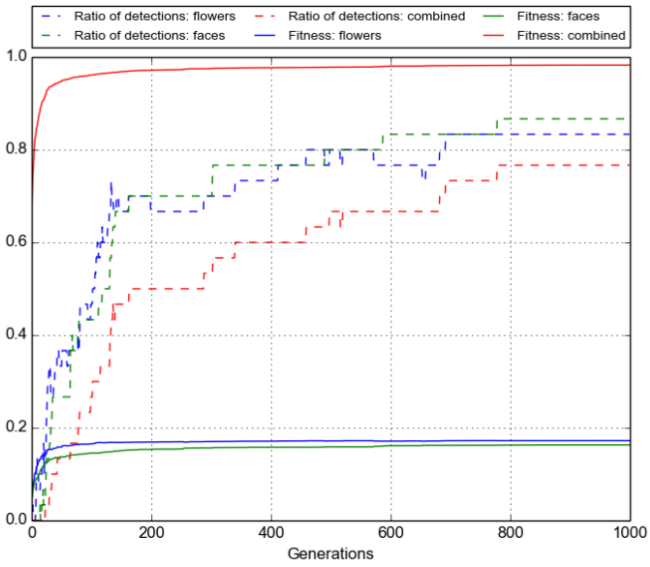

Figure 7: Fitness of the best individual and percentage of the best individuals containing an overlap between a face and a flower. In addition to the overall fitness and detection ratios, the partial fitness and the ratios of each of the detectors is also presented. The results are averages of 30 runs' objects.

faces and flowers were simultaneously detected in overlapping regions is also depicted, as well as the percentage of the best individuals where faces and flowers were detected. All results are averages of 30 runs.

As can be observed, although there is an abrupt increase of fitness during the first generations, improving fitness beyond that point is extremely difficult. Moreover, maximizing the response of the face detector is harder than maximizing the response of the flower detector. This outcome was expected since the same behaviour was observed when evolving images containing a single object. In 76.6% of the runs, the algorithm was able to evolve images where overlapping faces and flowers were detected. However, when we compare the fitness values obtained by each of the two object detectors with those obtained when evolving single objects, we arrive at the conclusion that the components of the combined fitness are far from their maximum values. This can be observed by contrasting the value reached by the fitness component regarding flowers of Figure 7, with the value attained when evolving flowers only, which is depicted in Figure 4. Therefore, although overlapping faces and flowers were detected in 76.6% of the runs, the difficulties found in maximizing the individual fitness components indicate that these detections are probably not robust.

An analysis of the resulting images reveals that although the majority of the runs evolved images where both objects were detected in the same window, which can, as such, be considered computationally ambiguous, most of the images found are not evocative of both objects (see Figure 8). Thus, in most cases they do not induce a multistable interpretation. Nevertheless, in some cases, images that are also ambiguous

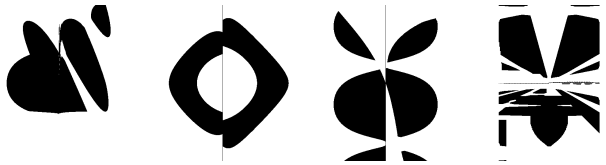

Figure 8: Example evolved images that are computationally ambiguous, but fail to induce, in our opinion, multistable interpretation in humans.

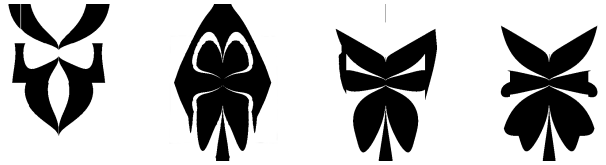

Figure 9: Examples of evolved images considered ambiguous by both humans and computers.

from a human perspective were evolved. Figure 9 depicts some of these exceptions to the norm. For instance, looking at the leftmost image of Figure 9 we can identify eyebrows, the white oval shapes create the illusion of eyes, while the remaining symmetrical black shapes create the illusion of a face contour. Simultaneously, the white regions can be interpreted as petals of flowers. Looking at the rightmost image, one can recognize a flowery pattern, but one can also interpret the top two "petals" as eyebrows and the middle shapes as eyes, which immediately evoke a face, and then one will probably interpret the bottom petals as a beard or mustache.

Humans have evolved to quickly recognize faces, which is simultaneously advantageous and problematic in this context. On the one hand, our ability to recognize faces even when only a subset of the features is present makes the task more feasible. On the other hand, the same ability makes the analysis of the experimental results more subjective. The shared left-right symmetry of faces and flowers also plays an important role in the evolution of ambiguous images. We are currently conducting experiments using other objects, some of which are not symmetric (e.g. profile faces). Although we are able to evolve computational ambiguous images, it is hard for humans to see both objects, particularly the non-symmetric one. Our tentative explanation is that since detection of symmetry plays an important role in human image perception, humans tend to be drawn towards the symmetric object overlooking the non-symmetric one.

5 Conclusions

In this paper we explored the generation of ambiguous images by evolutionary means. We used a general purpose expression-based GP image generation engine and several object detectors. The fitness is assigned by utilizing values from the detection phase. We used a framework presented in previous work and further explored it, while addressing some challenges of this research. The experimental results also highlight the differences between human ambiguous and computational ambiguous images.

At first, several object detectors were used to assign fitness and evolve images that resemble faces and flowers. The results from 30 runs per classifier showed that it is possible to evolve images that are detected and resemble, from a human perspective, the object. Next, we focused on the combination of flower and face detectors, and evolved images that contained both objects. The results showed the ability of the system to evolve images where both object detectors found their respective object. Some of the evolved images depicted optical illusions, with shared visual features and tiling patterns. In the final experiment, the object detectors were parameterized to detect larger objects, forcing the overlapping of the objects in the evolved images. In several runs the system was able to evolve images where the two objects are detected by the respective object detectors. Although the evolution of computational ambiguous images was frequent, only a portion of these images are evocative of both objects to humans. These evolved images can be considered ambiguous to humans, capable of inducing multistable perception.

Although the results obtained so far are not of the same level as human-designed ambiguous images, we consider them inspiring. They also demonstrate the feasibility of the approach and open new avenues for research.

The next steps will be the following: perform experiments considering a wider set of classes of objects; further explore the evolution of images with partial and total overlap of object detectors and explore the generation and evolution of ambiguous tiling patterns. As previously mentioned, the experimental results contain many false positives. In a different line of research (see, e.g., [Machado *et al.*, 2012b]) we are exploring the ability of the GP approach to find false positives to improve the quality of the training sets, and hence the robustness of the object detectors.

Acknowledgments

This research is partially funded by: Fundação para a Ciência e Tecnologia (FCT), Portugal, under the grant SFRH/BD/90968/2012; project ConCreTe. The project ConCreTe acknowledges the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET grant number 611733. The authors are grateful to Harry Goldingay for proofreading the paper.

References

[Baker, 1993] Ellie Baker. Evolving line drawings. Technical Report TR-21-93, Harvard University Center for Research in Computing Technology, 1993.

[Correia *et al.*, 2013] João Correia, Penousal Machado, Juan Romero, and Adrián Carballal. Evolving figurative images using expression-based evolutionary art. In *Proceedings of the fourth International Conference on Computational Creativity (ICCC)*, pages 24–31, 2013.

[DiPaola and Gabora, 2009] Steve R. DiPaola and Liane Gabora. Incorporating characteristics of human creativity into an evolutionary art algorithm. *Genetic Programming and Evolvable Machines*, 10(2):97–110, 2009.

[Frowd *et al.*, 2004] Charlie D. Frowd, Peter J. B. Hancock, and Derek Carson. EvoFIT: A holistic, evolutionary facial imaging technique for creating composites. *ACM Transactions on Applied Perception*, 1(1):19–39, July 2004.

[Johnston and Caldwell, 1997] Victor S. Johnston and Craig Caldwell. Tracking a criminal suspect through face space with a genetic algorithm. In Thomas Bäck, David B. Fogel, and Zbigniew Michalewicz, editors, *Handbook of Evolutionary Computation*, pages G8.3:1–8. Institute of Physics Publishing and Oxford University Press, Bristol, New York, 1997.

[Lewis, 2007] Matthew Lewis. Evolutionary visual art and design. In Juan Romero and Penousal Machado, editors, *The Art of Artificial Evolution: A Handbook on Evolutionary Art and Music*, pages 3–37. Springer Berlin Heidelberg, 2007.

[Machado and Cardoso, 2002] Penousal Machado and Amílcar Cardoso. All the truth about NEvAr. *Applied Intelligence, Special Issue on Creative Systems*, 16(2):101–119, 2002.

[Machado *et al.*, 2007] Penousal Machado, Juan Romero, and Bill Manaris. Experiments in computational aesthetics: an iterative approach to stylistic change in evolutionary art. In Juan Romero and Penousal Machado, editors, *The Art of Artificial Evolution: A Handbook on Evolutionary Art and Music*, pages 381–415. Springer Berlin Heidelberg, 2007.

[Machado *et al.*, 2012a] Penousal Machado, João Correia, and Juan Romero. Expression-based evolution of faces. In *Evolutionary and Biologically Inspired Music, Sound, Art and Design First International Conference, EvoMUSART 2012, Málaga, Spain, April 11-13, 2012. Proceedings*, volume 7247 of *Lecture Notes in Computer Science*, pages 187–198. Springer, 2012.

[Machado *et al.*, 2012b] Penousal Machado, João Correia, and Juan Romero. Improving face detection. In Alberto Moraglio, Sara Silva, Krzysztof Krawiec, Penousal Machado, and Carlos Cotta, editors, *Genetic Programming - 15th European Conference, EuroGP 2012, Malaga, Spain, April 11-13, 2012. Proceedings*, volume 7244 of *Lecture Notes in Computer Science*, pages 73–84. Springer, 2012.

[McCormack, 2007] Jon McCormack. Facing the future: Evolutionary possibilities for human-machine creativity. In Juan Romero and Penousal Machado, editors, *The Art of Artificial Evolution: A Handbook on Evolutionary Art and Music*, pages 417–451. Springer Berlin Heidelberg, 2007.

[Nishio *et al.*, 1997] Kenichi Nishio, Masayuki Murakami, Eiji Mizutani, and Nakaji Honda. Fuzzy fitness assignment in an interactive genetic algorithm for a cartoon face search. In Elie Sanchez, Takanori Shibata, and Lotfi A Zadeh, editors, *Genetic Algorithms and Fuzzy Logic Systems: Soft Computing Perspectives*, volume 7. World Scientific, 1997.

[Otsu, 1979] Nobuyuki Otsu. A Threshold Selection Method from Gray-level Histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 9(1):62–66, 1979.

[Secretan *et al.*, 2008] Jimmy Secretan, Nicholas Beato, David B. D Ambrosio, Adelein Rodriguez, Adam Campbell, and Kenneth O. Stanley. Picbreeder: Evolving pictures collaboratively online. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 1759–1768, New York, NY, USA, 2008. ACM.

[Sims, 1991] Karl Sims. Artificial evolution for computer graphics. *ACM Computer Graphics*, 25:319–328, 1991.

[Ventrella, 2010] Jeffrey Ventrella. Self portraits with mandelbrot genetics. In *Proceedings of the 10th international conference on Smart graphics*, SG'10, pages 273–276, Berlin, Heidelberg, 2010. Springer-Verlag.

[Viola and Jones, 2001] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 1:511, 2001.

[World, 1996] Linda World. Aesthetic selection: The evolutionary art of Steven Rooke. *IEEE Computer Graphics and Applications*, 16(1), 1996.