

# A robotic system for interpreting images into painted artwork

Carlos Aguilar, Hod Lipson

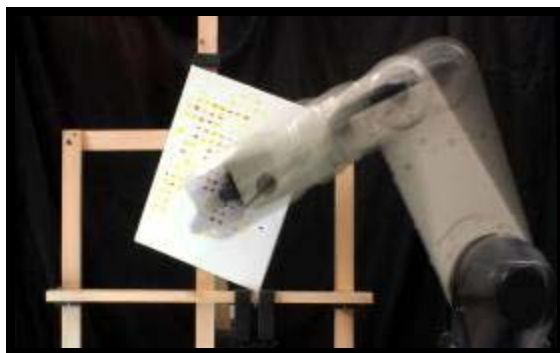
Cornell Computational Synthesis Lab, Cornell University, Ithaca, NY, USA

<http://ccsl.mae.cornell.edu>

e-mail: [cga9@cornell.edu](mailto:cga9@cornell.edu), [hod.lipson@cornell.edu](mailto:hod.lipson@cornell.edu)

## Abstract

We report on a robotic system that can physically produce paintings with a wide range of artistic media such as acrylic paint on canvas. The system is composed of an articulated painting arm and a machine-learning algorithm aimed at determining a series of brushstrokes that will transfer a given electronic image onto canvas. An artist controlling the system is able to influence the resulting art piece through choice of various parameters, such as the palette, brush types and brushstroke parameters. Alternatively, an artist is able to influence the outcome through varying the algorithmic parameters and feedback of the learning algorithm itself. In these results, a genetic algorithm used a painting simulation to optimize similarity between the target and the source images.



(a)



(b)



(c)

*Fig. 1. The robotic painting system: (a) An articulated 6DOF arm holding a paintbrush to a 23 x 30cm canvas; (b) A close-up view of the brush holder; (c) sample painting of a portrait*

## 1. Introduction

### 1.1 Paint and its significance

Even the technological advancements of the post-modern age cannot replace the simplicity or cultural significance of paint and other traditional media. Artists have created representational art from suspended pigment for tens of thousands of years. From cave paintings through the Renaissance to the modern and contemporary eras, paint has never ceased to have a significant role in art. Traditional mediums will always have a power associated with the history of art: every new painted artwork refers implicitly to the history of painted art.

We have developed a robotic system which allows a computer to generate a physical painting from a source image with the aid of the *technartist's* artistic interpretation. Given an input image and environmental parameters, such as brush sizes, the combined computer-artist system is able to interpret the image and the robot then executes the image onto a canvas using paint.

The robotic system is comprised of three main components: a digital paint simulation, a genetic algorithm and an industrial robotic arm. The paint simulation gives the ability to predict how a paint stroke would look if it were painted on canvas. The genetic algorithm utilizes this paint simulation as feedback in order to try to choose a set of brush strokes that will match the input image. Finally the robotic arm automatically executes these strokes onto a physical canvas by manipulating brushes and acrylic paint.

Computer scientists have attempted to transfer the power of paint to the digital era: a whole field of computer graphics is devoted to generating digital images that resemble paintings. Computer scientists have also developed algorithms to simulate the physical behaviour of paint on a computer. Artists have taken advantage of these paint simulations to make original digital paintings.

New media artists have also taken advantage of the artistic significance of paint by developing machines able to produce painted artwork autonomously. Several robotic painting systems will be outlined later in this paper.

### 1.2 An intelligent robotic artist

One goal of this research is to create a robotic system with a self-model based on paint simulation algorithms. This allows the robot to intelligently predict its own behaviour. We present the first machine system able to intelligently interpret an image into an original style of physical artwork.

Genetic algorithms are stochastic algorithms able to solve some open-ended problems in creative and ingenious ways. These algorithms are especially useful when a solution of a problem can be accurately modelled and easily evaluated. GA's can often free a computer from human imposed limitations on a problem and can achieve unexpected results. For these reasons artists have found GA's powerful when trying to create generative art.

Computers have become an integral part of every facet of our lives and art is no exception. Producing or manipulating digital images has become a part of many artists' process. To reach the full effect of an artwork, however it must be taken out of the digital realm and rendered physically, whether displayed on a screen or printed using a printer.

Artists only have a handful of ways to physically produce digital images. Conventional printers and display technologies represent digital images using pixel primitives. A primitive is the most basic unit of any image; they are the atomic elements of an image. Pixels arise as a natural way for a computer to produce a digital image, not because they are appropriate for displaying images in an interesting or artistic way.

Our software implements a standard way in which a computer can represent an image in terms of an abstract primitive; in this case the primitive is a paint stroke. This abstract primitive allows the system to treat the primitive physically, taking advantage of the physical characteristics of a paint stroke.

### 1.3 Implications

People are integral in the artistic process surrounding this robotic painter. The system is not intended to make high level decisions – it relies on human input for these decisions, such as subject matter and palette. This means that a person that might lack the manual dexterity to paint is still able to explore paint as a medium. This research opens the door to a new level of human-machine artistic collaboration.

An artist is able to understand how the algorithm tries to interpret an image, but the algorithm will ultimately add some unrepeatable and unpredictable interpretation of its own.

In this paper we present a brief survey of robotic meta-art, explain the painting simulation and genetic algorithm used to implement our robotic system and finally show some simulated and painted results from the robotic painter.

## 2. Problem Statement

Our goal is to take a digital image and have a computer analyze it and produce a physical representation of this image in an original style. This problem is fundamentally different than the problem of painterly rendering in computer graphics.

In computer graphics, arbitrary paint behaviours, paint colors and stroke types can be used to achieve whatever type of affect the user or the algorithm requires. Our research however addresses a physical problem: what instructions can I give a robot to produce a painting? Real paint phenomena must be recreated and predicted in a computer and the result of the algorithm is a set of robotic instructions, not a digital image.

Our work is also materially different than other software based art that can be outputted using a printer. While many software systems can claim original styles of art, these styles are fundamentally digital and can only be transcribed to reality using digital processes which utilize the same digital primitives that a computer uses. One of our central claims is the ability of our system to use the physics of the resulting art in its creative process.

## 3. Background

A number of robotic systems have been developed which produce works of art without the use of complex robotic behaviour such as Sabrina Raaf's *Translator II: Grower* [1]. In *Grower*, a robot slowly traverses a wall and plots in green ink the CO<sub>2</sub> content of the room.

*Grower* contains an interactive element not present in our work and in the rest of the robotic systems discussed in this section. As viewers enter the room they are able to see their affect on the growth of the simulated grass. *Grower* however represents a robotic system whose behaviour is pre-programmed. The most important similarity between *Grower* and our system is that the mechanism for synthesizing the art piece has been automated. The process of painting then becomes an important context for the viewer. The action of painting becomes a significant, and often times publicly displayed portion of the art piece.

Another class of robotic art systems produce artwork with agent robots programmed with individual behaviours. The most compelling example of this class is the work of Leonel Moura. In one of his projects, *ArtSBot* [1] Moura uses a swarm of independent autonomous robots which travel across a canvas depositing ink. The robotic system has no global goals, but each agent has a series of programmed behaviours that depend on the sensor information from its environment: namely the color of the canvas below it and obstacles in its way. The artwork becomes a performance of these simple machines and the resulting drawing is an artefact of the complex dynamics that arise from each agent's simple programmed behaviour.

Our work belongs to a final class of robotic painter that is able to produce representational images. Harold Cohen's *AARON* [3] is probably one of the earliest and most advanced robotic action painters able to produce physical, representational artwork. *AARON* uses a representational model for synthesizing a series of objects: people, rocks, tables and plants. *AARON* is entirely autonomous, choosing its subjects, composition and palette entirely without the input of humans.

While *AARON* produces paintings with an element of randomness, the style and aesthetics of the paintings have been designed and programmed by humans. This work takes the opposite approach to representational art. We do not attempt to automate high level decisions such as subject matter. Our system tries to choose strokes to take advantage of the physical media by predicting the results of the executed painting.

Other robotic representational art includes Robotlab's *Autoportrait* [4], Calin *et al.*'s *A robot Painting a Portrait* [5], Adrian Bower's robotic painter [6] and Jessica Bank *et al.*'s *Fotron 2000* [7]. The media is different in each of these systems: the first two systems use markers, Adrian Bower's system uses paint and the *Fotron 2000* uses light emitting diodes that paint light onto Polaroid film.

All of these systems use image processing techniques to decompose an image. *Autoportrait* features edge finding as well as face recognition algorithms. The *Fotron 2000* also uses an edge finding algorithm but is also able to represent some colors with different lights. Adrian Bower's paint robot uses algorithms to decompose an image into several regions of solid color which the robot then attempts to fill in on the canvas. These systems are different from the system presented in this paper in that they attempt to reverse engineer an image in order to represent it. The result is a human programmed style of representation. Our system, unlike any system before it, does not have any specific instruction on how to paint, just knowledge of how it can affect its environment.

## 4. Materials and Methods

### 4.1 Approach

This project exploits the strength of evolutionary algorithms in solving complex problems with well-defined objectives. In this case, the objective is simply to attempt to reduce color error between our input image and the painted image.

This software takes a bottom up approach to generative art, using the ingenuity of its genetic algorithm to solve for stroke-level style. While other algorithms are better suited for finding solutions for a specific painting goal with a given set of parameters, no other algorithm provides the same flexibility of a GA. For example, the algorithm could make line drawings simply by changing the fitness function and types of strokes used in the algorithm.

In the following sections we will outline a previous version of our robotic painter that did not include a self-model. We then outline the simulation algorithm and genetic algorithm used for the intelligent version of our system.

### 4.2 Brief overview of previous methods

The first version of our robotic painter used the well defined Cyan Magenta Yellow and Key (or black) primaries (CMYK) in order to attempt to paint an image onto canvas. The algorithm calculated the density of each of the channels CMYK using an approximate conversion from the RGB values.

Several iterations of this algorithm were developed. The final version of the algorithm painted first with larger brushes and covered large areas, and covered smaller details with smaller brushes. For the smallest brush, the algorithm would simply paint at the gradients of the channels.

### 4.3 Paint Simulation for feedback painting system

#### 4.3.1 Kubelka-Munk

The Kubelka-Munk theory of pigment reflectance [8] is used to simulate paint mixing in a computer paint simulation as was first proposed by Haase *et al* [9]. Kubelka-Munk gives an approximation of the reflectance of light as a function of the ratio of absorption (K) and scattering (S). For an infinitely thick paint sample, the result is Equation 1.

$$R_{\infty} = 1 + \frac{K}{S} - \sqrt{\left(\frac{K}{S}\right)^2 + 2\frac{K}{S}} \quad (1)$$

For a homogenous mixture of several pigments, the total absorption and scattering coefficients are simply weighted by the relative concentrations of each paint in the mixture. We can then find the effective ratio of absorption to scattering as in Equation 2, where  $c$  is a concentration between 0 and 1.

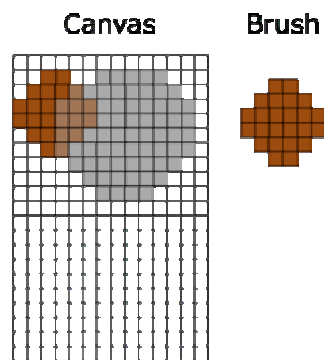
$$\left(\frac{K}{S}\right)_{Mix} = \frac{\sum_1^n c_i K_i}{\sum_1^n c_i S_i} \quad (2)$$

To display the colors on a monitor the reflectance values must be converted to the 1931 CIE color space. The set of discrete reflectance values from (1) represent an approximation for the spectral reflectance curve of the paint mixture. This discretized reflectance curve can be linearly transformed into the tristimulus XYZ primaries as in

Hunt's *Measuring Color* [10]. The tristimulus primaries are then transformed into RGB space using Bruce Lindbloom's *RGB/XYZ Matrices* [11].

Absorption and scattering coefficients are used for twelve paints derived from the physical paint data collected by Jeff Budsberg [12],[13].

#### 4.3.2 Paint Simulation Algorithm



A cell based algorithm similar to Baxter's Impasto **Errore. L'origine riferimento non è stata trovata.** is implemented to produce the paint simulation algorithm. The algorithm is able to simulate paint mixing at up to eight wavelengths. Two digital images are used to represent the canvas and the brush in the simulation. Each pixel represents a cell and the channels of the image keep track of the scattering and absorption coefficients. There is also a channel that keeps track of the amount of paint at each cell. This means that each image can have up to 17 channels.

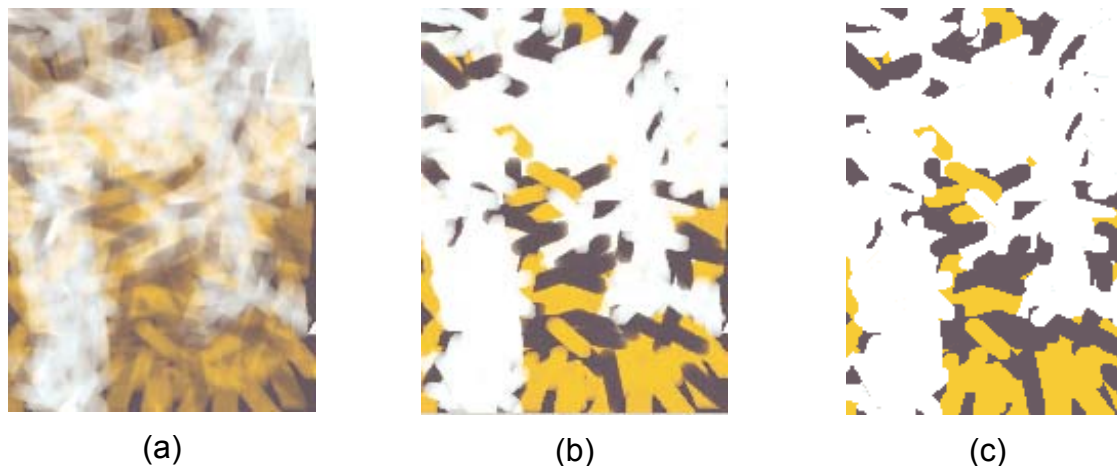
*Fig. 2. Example of a simulated paintbrush-canvas interaction*

To simulate the interaction of the brush on the canvas we register the brush image to a desired location on the canvas image and perform a cell transfer operation on corresponding cells in each image as in Figure 2. We can

then recalculate the absorption and scattering coefficients using (1) and (2).

To model paint layering, a simple mechanism is introduced in the cell transfer operator that allows for different opacities when depositing simulated paint.

To produce an entire stroke, the brush transfer is performed once at each pixel along the path. The results of varying the layer opacity are shown below in Figure 3.



*Fig. 3. Simulated paintings using three different layer opacities: (a)0.1, (b)0.5 and (c)1.0.*

#### 4.3.3 Physical Implementation

The purpose of the paint simulation is to accurately model a dynamic physical system. The approach of a genetic algorithm is effective only if our simulation corresponds closely with the physical system. For this reason the physical system is initially chosen to be as simple as possible.

Acrylic paint is used at a slow deposition rate to insure consistent dimensions and quality of paint strokes. This also allows for strokes to dry before the next layer is deposited. After this simplified painting setup has been developed more complicated and dynamic paint behaviours can be simulated and utilized.

Although this simplified system allows for accurate simulation, it limits the behaviour of a single brush stroke, which in turn makes it much more difficult to use this brush stroke to evolve paintings.

## 4.4 Genetic Algorithm

### 4.4.1 Introduction to Genetic Algorithms

Genetic algorithms use mechanisms inspired by biological evolution in order to find an a solution to a problem. The first step is encoding the problem's solution into a series of numbers or bits that the computer can manipulate; this encoding represents the genotype of the solution. The algorithm then creates a random population of solutions: in our case, this population consists of randomly encoded paintings. Most of these random paintings will not correlate at all with the input image, however some will undoubtedly be better than others. The population is then modified using recombination and mutation operators and a new generation of paintings is born. Each generation the best paintings of the new generation replace the least fit paintings according to some fitness criteria and a selection method.

It is noted that this is not the first attempt to use evolutionary algorithms to evolve paintings. John Collomosse and Chakraborty *et al*[15] both report on code able to evolve a set of brush strokes to an input image. Both of these evolutions, however are entirely non-physical. In the evolution they allow the algorithm to layer an arbitrarily large number of strokes with an arbitrary opacity and arbitrary RGB color. The result is a digital image with a painterly quality, but differs greatly from the physical problem we attempt to solve.

### 4.4.2 Representation and Population Initialization

A painting is represented as a list of brush strokes. Each brush stroke consists of five parameters: an index representing the paint color, an index representing the brush size, a length, an angle and a position. To convert the brushstroke list into a painting, the list is first sorted by brush width and then paint color.

A population of strokes is initialized by producing random strokes of a user-defined maximum stroke length along a grid that is also defined by the user.

### 4.4.3 Fitness

The HSV color error between the input image and the simulated painting are used as the fitness criteria:

$$E_{HSV} = \sqrt{(v_1 - v_2)^2 + (s_1 * \cos(h_1) - s_2 * \cos(h_2))^2 + (s_1 * \sin(h_1) - s_2 * \sin(h_2))^2} \quad (3)$$

The HSV color error is defined as the vector distance between two colors when represented in the conical HSV color space. The error of the simulation versus input image is then summed over all of the pixels.

#### 4.4.4 Selection

Deterministic crowding is used for selection. Every generation each painting is paired with a randomly selected painting for crossover. After crossover and mutation, each offspring is then paired with the parent that contributed the most strokes to it. The offspring then competes with the paired parent to see which will get inserted into the population. The parent is replaced if its color error is larger than that of the child.

#### 4.4.5 Crossover

A box region crossover is used to produce two offspring paintings from two parent paintings. Each child will take all the strokes in a random boxed region from one parent, and will take all of the strokes outside of this region from the second parent.

Figure 4 demonstrates the crossover operator.

#### 4.4.6 Mutation

The mutation operator modifies a single stroke in a painting. The mutation operator modifies one of the following parameters with equal probability.

- Position – the position is moved a distance  $D$  and by an angle  $A$ .  $A$  is chosen random to be between  $0-2\pi$ .  $D$  is chosen randomly to be between zero and the grid size used to initialize the population.
- Stroke color – either adds one or subtracts one to the paint color index.
- Brush width – either adds one or subtracts one to the brush index.
- Angle – change the angle by  $a$ , where  $a$  is chosen to be a random number between  $-\pi$  and  $+\pi$ .
- Stroke length – scale the brush length by a random number between 0.5 and 1.5

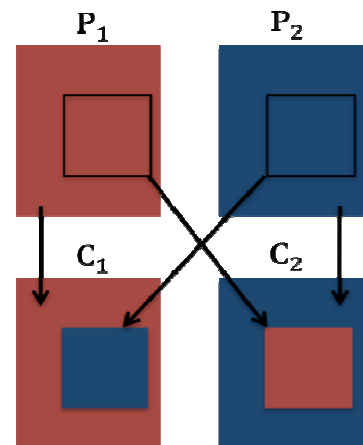


Fig. 4. Example crossover operation.

## 5. Results

### 5.1 Results from Human Designed paint algorithm

An early version of the robotic painter was a semi-deterministic painter that used described in section 4.2. In this version of our system there was no feedback of any sort, so simulated results are not available. Several results from several versions of the human designed painting algorithm are shown in Figure 5.





(a)



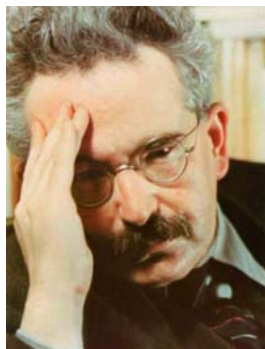
(b)



(c)



(d)



(d)

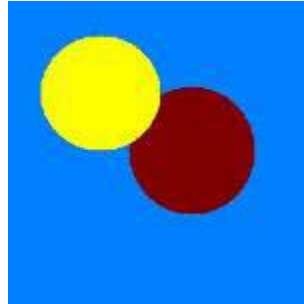


(e)

*Fig. 5. Robotic paintings from human-designed painting algorithm: (a) Dan Whoarly's 'Jimi' [17]; (b) 10 x 16.5 cm robotic painted acrylic on canvas; (c) flower image by André Karwath [18]; (d) robotic painted 15 x 20 cm acrylic on canvas; (e) Picture of Walter Benjamin [19] (f) 28 x 31.5 cm robotic painted acrylic on canvas*

## 5.2 Evolved Simulation Results

Each result of an evolution is unique and cannot be replicated. However, the evolution has several mechanisms by which the user can affect the result. The initial population size, the mutation ratio and the crossover ratio will affect the performance and efficiency of the algorithm. Most importantly the number of strokes, the stroke maximum length and the paint colors affect potential styles of the resulting painting. We first look at a simple input image given in Figure 6.



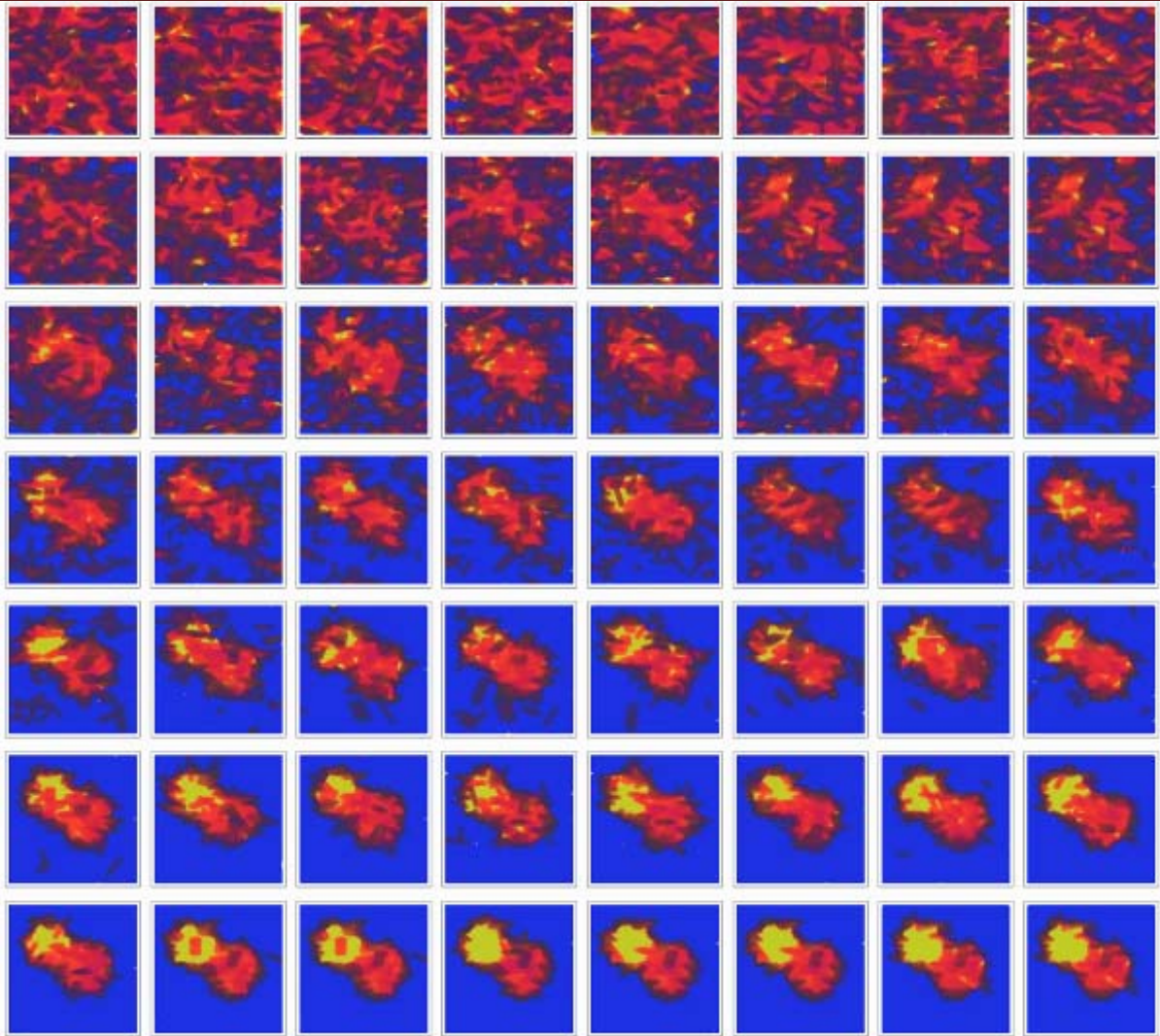
*Fig. 6. Two circles input image*

This case is a simple test to demonstrate the functionality of the evolutionary algorithm. The circles are chosen because there is nothing in the primitive (a straight stroke) that implies it should be able to represent a circle trivially. A non-physical paint palette is used, shown in Figure 7. The paint colors were invented to somewhat correlate with the input image, but do not trivially match the input image's colors. The layer opacity is set to .25 to allow for paint mixing and the initialization grid is set tightly so that the evolution can take advantage of paint mixing. Six hundred brush strokes is chosen to demonstrate that the algorithm is able to converge a large number of parameters, and also to ensure that there are enough strokes to represent the input image.



*Fig. 7. An artificial palette used for painting evolution*

Each generation of the evolution (generations 1-56) are shown in Figure 8. The evolutionary algorithm can successfully map the three colors in the input image in under 60 generations.



*Fig. 8. 56 generations of an evolution*

For the next case, a more complicated image is chosen. The first simulations using Budsberg's physical paint data are performed on a picture of German philosopher Walter Benjamin, shown above in Figure 5. Walter Benjamin is chosen for his influential essay relevant to the field of representational art and the study of artistic media: *The Work of Art in the Age of Mechanical Reproduction* [20].

The first evolutions use just twenty brush strokes to attempt to represent the input image. Two different palettes: one which we thought would be appropriate for the picture, the second palette simply contained all of the colors for which there was physical data.

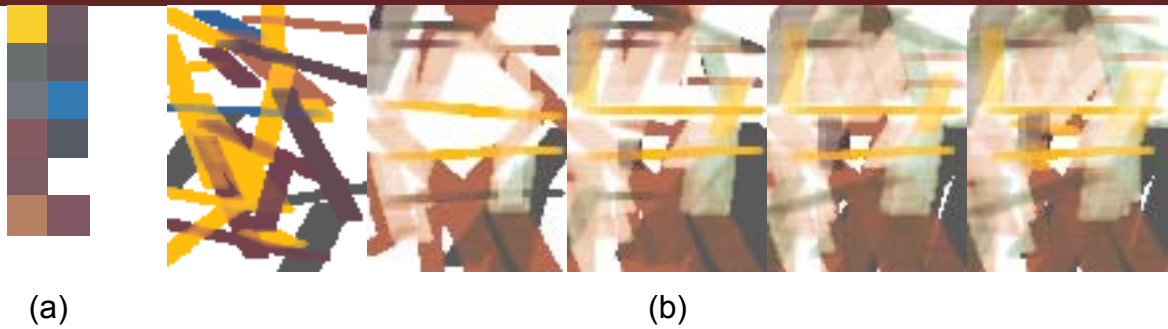


Fig. 9. Abstract painting of Walter Benjamin with layer opacity of .1: (a) The palette used; (b) the best simulated painting after 0, 50, 100, 200 and 300 generations.

Additionally different amounts of paint mixing are allowed in the simulations. A layer opacity of 0.1 corresponds with a system where more paint mixes in the canvas or where the paint layers are thin and translucent. A layer opacity of .5 on the other hand does not allow for a significant amount of paint mixing. The progressions of one such evolution are given in Figure 9. The final result of the abstract painting evolutions are shown again in Table 1.







Layer Opacity	0.5	0.1
Palette		
		
		

Table. 1. Sumarized results from abstract evolutionary paintings.

Finally the algorithm is tested on a much more difficult problem with the same input image of Walter Benjamin. For this evolution a simulation corresponding to the physical system described in section 4.3.3 is used. Here virtually no paint mixing is allowed, using a layer opacity of .5. Also the dimensions of the strokes is reduced and the density of strokes is greatly increased. A total of 1223 strokes are used in the simulation. The results are shown in the figure below.

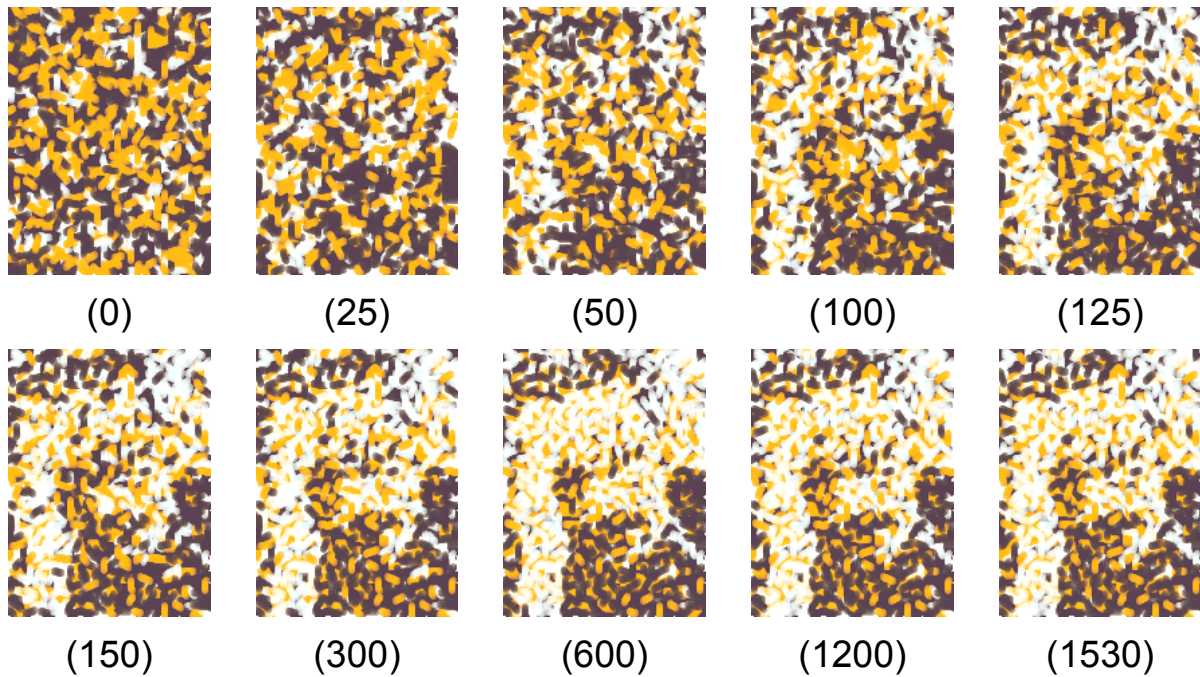


Fig. 10. Progression of an evolution of Walter Benjamin. The generation number is given in parentheses.

### 5.3 Evolved Paintings Physical Results

A painting was physically executed of an evolution similar to Figure 10. The input image, the simulated result and the physical result are shown in Figure 11.

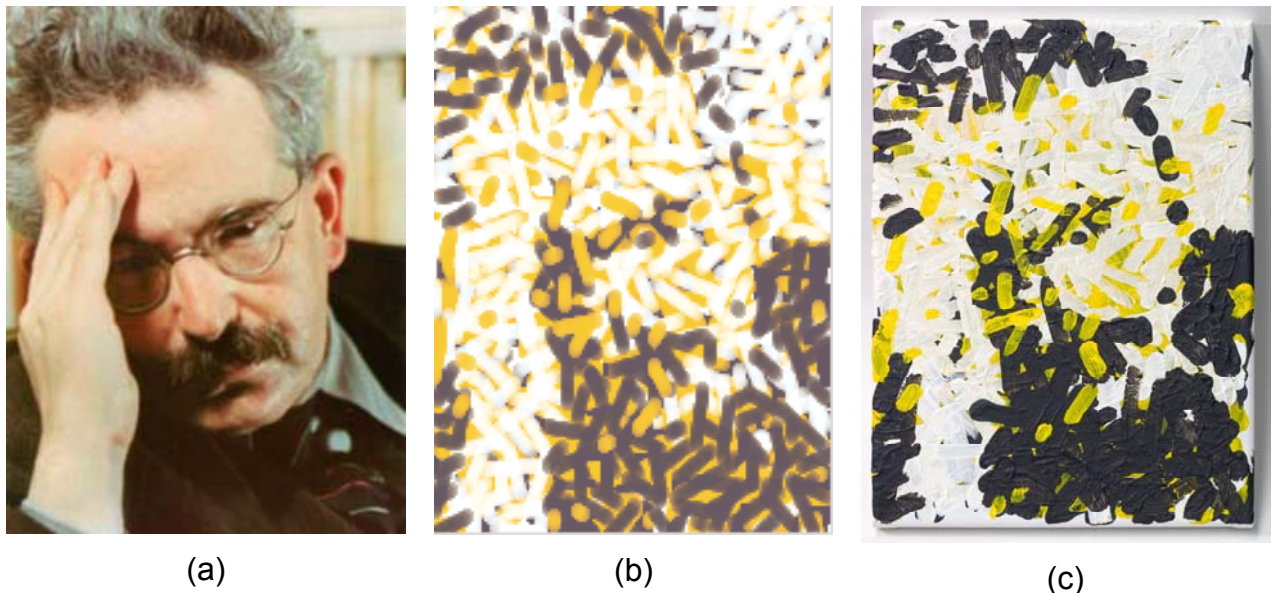


Fig. 11. A physically painted evolved painting. (a) shows the input image, (b) shows the simulated evolved painting and (c) physical painting 23 x 30cm robotic painted acrylic on canvas.

## 6. Discussion

The simulation results demonstrate the diverse styles and aesthetics our robotic painter is able to conceive. The first painting executed from a simulated evolution (Figure 11) is a good test of the flexibility of the genetic algorithm. Since minimal paint mixing is allowed in this particular evolution, determining a painting to match the input image becomes difficult. The result however is still clearly a unique interpretation of the input image.

The effect of the primitive in determining the result and style of a painting is demonstrated in the abstract evolved paintings in Table 1. In these short evolutions we see that given a more flexible primitive, the evolution is able to achieve more accurate representation consistently. Specifically if paint mixing is allowed then we are able to obtain images that more closely resemble the source image.

The ability of the algorithm to find a good representation depends greatly on the palette. For example if we only use two similar colors that are not represented in the source image the algorithm will not be able to represent the input image well.

The dependence of representational ability on palette stems from the fitness function. A broader palette and an ability to mix the colors in this palette together allows the algorithm to better match the colors to the input image.

Alternative fitness functions based on shape and frequency could address this limitation. Using the Fourier Transforms of an image has been considered as a fitness that would allow a wider variety of palettes to represent a given image.

It is important to note that while the goal given to the algorithm is to produce the most accurate reproduction of an input image, this is not necessarily the final goal of the controlling artist. The most interesting facet of these results is not how well we can reproduce a given image, but how fundamental parameters and human choices affect the way the algorithm chooses to interpret the image. Collaboration between the artist and the machine becomes necessary because their goals are at odds with each other.

While a significant amount of time was spent improving the algorithm to be able to evolve more precise representations, the most compelling results might be the abstract representations in Figure 9, although this evolution only took 1% of the time to evolve when compared to the more precise painting in Figure 10. The abstract results were also more unpredictable. While the small stroke, high precision problem often times resulted in paintings that looked very similar, the abstract evolutions often produced unexpected and varied results.

The face has proven a powerful tool for abstraction. While the image of Walter Benjamin is a complicated image with large gradients, and small details, human ability to recognize and interpret faces allows us to recognize the image even with the high levels of abstraction.

## 7. Future Work

The simulated image still deviates from the painted result in Figure 11. Moving forward, the greatest improvement that can be made to our work is to improve the paint simulation. This will have the resulting painting match the simulation more closely. More importantly it is desirable to expand to expand the simulation in order

to predict a wider variety of paint behaviours as well as curved strokes. As we teach the computer to model more complicated shapes and behaviours of paint strokes, the system will be able to develop a wider variety of unique styles.

## 8. Conclusion

Our painting robot is the first machine to intelligently reproduce an image in an original artistic style. From the results we have seen that the program has determined several original styles in which to paint in simulation.

Harold Cohen's 1974 essay *On Purpose* predicted: "[W]ithin the next two or three decades... the computer will have come to be regarded as a fundamental tool by almost every conceivable profession. The artists may be among them. That will be the case, obviously, only if it shows itself to have something of a non-trivial nature to offer to the artist; if it can forward his purposes in some significant way" [21].

This project attempts to forward the artist's purpose significantly by adding intelligent machine collaboration. We have avoided the viewing of robotic art as an entirely stand-alone agent of art production. Our system is not designed to act entirely on its own. Our system does not produce the subject matter because a computer cannot choose an input image with any understanding of its significance. Therefore giving it this ability seems trivial and does not further the artist's purpose.

This project formulates art in a way which computers can powerfully contribute: as an optimization problem. Optimization is similar to some low level human artistic processes: how can one best represent a certain form, possibly using only a certain type of basic element? Or how could one represent a certain value or color using a limited palette? These questions are not completely analogous to computer optimization because the human optimization is inevitably interwoven with many higher-level processes such as emotional intelligence.

For example, Georges Seurat's technical thought process approached the problem of representing color by trying to maximize the averaging effects of his pixel-like primitives. Similarly, Picasso thought technically about the most basic shapes that might be able to represent a scene in many of his abstract paintings.

While a computer cannot produce the original artistic styles or creative breakthroughs of Seurat or Picasso, the described low level technical thought process is achievable in computers.

## Acknowledgement

This research has been supported in part by US National Science Foundation (NSF) Grant IIS-0428133.

## References

- [1] Sabrina Raaf, "*Grower*", robotic installation, 2004.
- [2] Leonel Moura, "*ArtSBots*", robotic vehicles, ink and canvas, 2003.
- [3] Harold Cohen, "*AARON*", robotic painting machine, 1973.
- [4] Robotlab, "*Autoportrait – portrait drawing robot*", robotic arm, dry erase marker on white board, 2002.

- [5] S. Calinon, J. Epiney and A. "Billard", IEEE-RAS International Conference on Humanoid Robots, 2005.
- [6] Adrian Bowyer, "A robot-rendered *Giaconda*", IBM SCARA robot paint on canvas.
- [7] Jessica Banks, Jonathan Bachrach and Daniel Paluska, "Fotron 2000", robotic sketch artist, leds and polaroid film, 2000.
- [8] Paul Kubelka and Franz Munk, "Ein Beitrag Zur Optik der Farbanstriche", *Zeitschrift fur technische Physik*, 1931.
- [9] Chet Haase and Gary Meyer, "Modeling Pigmented materials for realistic image synthesis", ACM Transactions on Graphics, 1992
- [10] R. Hunt. Measuring Colour (2nd ed), Fountain Press, Tolworth, UK, 1999.
- [11] Bruce Lindbloom. "RGB/XYZ Matrices". Accessed November 2008.  
[http://www.brucelindbloom.com/index.html?Eqn\\_RGB\\_XYZ\\_Matrix.html](http://www.brucelindbloom.com/index.html?Eqn_RGB_XYZ_Matrix.html)
- [12] Jeffrey B. Budsberg, Donald P. Greenberg, and Stephen R. Marschner. "Reflectance Measurements of Pigmented Colorants", Technical Report PCG-06-02. Cornell University, Ithaca, NY, September, 2006.
- [13] Jeffrey B. Budsberg, "Pigmented Colorants: Dependence on Media and Time", Master's thesis, Cornell University, January 2007.
- [14] W. Baxter, J. Wendt, and M. Lin, "IMPASTO: A Realistic, Interactive Model for Paint", In the proceedings of NPAR 2004, The 3rd International Symposium on Non-Photorealistic Animation and Rendering, Annecy, France, June 7-9 2004,
- [15] John P. Collomosse, "Supervised Genetic Search for Parameter Selection in Painterly Rendering", In the proceedings of Advances in Knowledge Discovery and Data Mining, 10th Pacific-Asia Conference, 2006.
- [16] U. Chakraborty and H. Kang, "Stroke-based Rendering by Evolutionary Algorithm", Proc. IEEE Indicon Conference, 2004.
- [17] Dan Whoarly, "Jimi", 2004 acrylic on canvas.
- [18] *André Karwath*, "Phalaenopsis", digital photograph, 2005.
- [19] "Walter Benjamin" Image accessed November 2008.  
<http://www.nndb.com/people/073/000039953/>
- [20] Walter Benjamin, "The Work of Art in the Age of Mechanical Reproduction", 1935.
- [21] Harold Cohen, "On Process: An Enquiry into the Possible Role of the Computer in Art", 1974.