
Evolving assemblages

Fernando Graça*

Department of Arts and Technologies of Image,
Paris 8 University, 93526 Saint-Denis CEDEX, France
E-mail: fernando.graca@mines-paristech.fr
*Corresponding author

Penousal Machado

CISUC, Department of Informatics Engineering,
University of Coimbra,
Polo II of the University of Coimbra,
3030 Coimbra, Portugal
E-mail: machado@dei.uc.pt

Abstract: ‘Evolving assemblages’ explores the transformation of images or three-dimensional (3D) models in order to create large-scale assemblages of 3D objects. The proposed approach allows the evolution of the type, size, rotation and position of each object, constructing a non-photorealistic transformation of a source image or 3D model. The creator evaluates each individual of the population accordingly to his/hers aesthetics preferences, influencing the final outcome of the process. The evolution of assemblages which are mapped to a 3D model is described. The experimental results presented highlight the generalisation abilities of the evolved individuals and the expressiveness of the evolutionary tool.

Keywords: evolutionary art; evolutionary image filters; non-photorealistic rendering; NPR; assemblage.

Reference to this paper should be made as follows: Graça, F. and Machado, P. (xxxx) ‘Evolving assemblages’, *Int. J. Arts and Technology*, Vol. X, No. Y, pp.000–000.

Biographical notes: Fernando Graça is currently undertaking a PhD at Ecole des Mines de Paris. Previously, he studied for a Master degree at Paris8 in Arts and Technologies of Image. Before coming to Paris, he received his BSc in Informatics Engineering from University of Coimbra. While there, he worked as a researcher at the Cognitive Media Systems Lab at the Centre for Informatics and Systems. His work is focused on generative graphics, and material appearance modelling to generate physically-based renderings.

Penousal Machado teaches Artificial Intelligence and Computational Art at the University of Coimbra and is the Founder and Leader of the Computational Design and Visualization Lab of the University of Coimbra. He is the author of more than 50 refereed journal and conference papers, member of the editorial board of the *Journal of Mathematics and the Arts* and of the *Genetic Programming and Evolvable Machines*, Chair of several scientific events, and recipient of several scientific awards. Recently, his work was featured in *Wired Magazine* UK and included in the ‘Talk to me exhibition’ of the Museum of Modern Art (MoMA).

1 Introduction

The research project ‘evolving assemblages’ (<http://evolving-assemblages.dei.uc.pt/>) started as a study on the development of ornamentation techniques. This initial goal led us to the creation of an interactive evolutionary art tool for the creation of large-scale assemblages of three-dimensional (3D) digital objects. These are characterised by the textures – created through the meshing of objects, by size, and rotation changes. Here, we introduce a new type of assemblage, 3D model assemblages, in which the objects are mapped onto the surface of a 3D model.

The main artistic sources of inspiration for this work are: pointillism, mixed media assemblage of objects, and ornamentation techniques (e.g., similar to the ones found in Gustav Klimt works). From a scientific point of view, areas such as evolutionary non-photorealistic rendering and artistic filter evolution are of particular relevance. The evolved individuals take as input a 3D mesh, and produce as output the type, scale, rotation and placement of the objects, which will be placed on the surface of the mesh. In this way, the result is an assemblage of 3D objects, which constitutes a non-photorealistic portrayal of the source model. Finally, the 3D assemblage is rendered using a raytracer.

The evolutionary process leading to the creation of assemblages of 3D digital objects can be succinctly described as follows:

- 1 the programme receives as input a 3D model
- 2 the user creates and selects a library of 3D objects to generate the assemblage
- 3 through a user-guided evolutionary process, the user evolves the type, rotation, size and placement of the objects:
 - a an initial random population of object assemblages is created
 - b the user indicates those that better match his/hers ideas
 - c the next population is created through the recombination and mutation of the genetic code of the selected assemblages
 - d the process is repeated from Step (b) until an assemblage that satisfies the preferences of the user is found.
- 4 a raytracer render the 3D scene.

The proposed approach can be seen as an instance of computer-aided creativity (Machado et al., 2007), in the sense that the tool takes care of several aspects related with the technical execution of the piece, allowing the user to focus on the creative aspects of the task. It allows the creation of novel artworks that would be (nearly) impossible to produce by conventional means. In addition, it provides mechanisms that allow the exploration of a search space of potential pieces, guided by the artistic and aesthetics preferences of the user. Due to the stochastic nature of the process, serendipity plays an important role, sometimes leading the artist to explore unforeseen paths and diverting the artist from his/hers original ideas. On the other hand, user guided-breeding promotes the recombination of the individuals that are closer to the aesthetics preferences of the user, leading to the successive refinement of the populations. As such, ‘evolving assemblages’ has an impact on the creative process, the artist is no longer responsible for the idea: the idea rises via the interaction between the artist and tool. Nevertheless, it is important to ensure that she/he can still convey her/his aesthetic preferences and views by the use of

the tool. The artists should be able to significantly influence the final outcome of the process, express themselves through the use of this tool, and recognise their *signature* in the evolved works, thus connecting with them at an emotional and artistic level.

The paper is structured as follows. In Section 2, we make a brief survey of related work. In Section 3, we present an overview of the different modules of the system. Next, in Section 4, we describe the evolutionary engine, giving particular emphasis to the representation, genetic operators, and genotype-phenotype mapping, and we describe the production of 3D model assemblages. In Section 5, we present and discuss some of the experimental results attained in the course of our research. Finally, we draw some conclusions and discuss aspects to be addressed in future work.

2 Related work

Through time, evolution has created a wide variety of species adapted to their environment. Some of these species – e.g., humans – exhibit intelligent and creative behaviour. According to Darwin (1859), evolution is based on two fundamental principles: selection, and reproduction with variation. Selection ensures that fitter individuals are more likely to reproduce. The descendants of these individuals inherit characteristics from the progenitors – which implies that they tend to be fit – but they are not exact copies, which allows evolution. The reinterpretation of Darwin’s ideas in light of Mendel’s genetics, i.e., Neo-Darwinism, explains how the characteristics are inherited and how and why changes occur. Natural selection occurs at the phenotypic level, while reproduction acts on the genotype (Dennett, 1995). The characteristics of the individuals are not directly inherited. Instead, the genes that codify these characteristics and those that enabled their development are inherited. Variation results from copying errors – i.e., mutations – and from the recombination of the genetic material of the progenitors.

From Holland’s (1975) work onwards, natural evolution has also become the basis for several artificial intelligence approaches, usually referred to as evolutionary computation (EC) whose goal can be synthesised as follows:

“How do we turn Darwin’s ideas into algorithms?” [Holland, (2000), p.377].

The idea of using EC for artistic purposes can be traced back to Richard Dawkins. In his book *The Blind Watchmaker*, Dawkins (1987) delineates a programme which allows the evolution of the morphology of ‘virtual creatures’ or *biomorphs*. More precisely, each *biomorph* is a drawing, the appearance of which depends on the values of a set of parameters encoded in a string, the genotype. The biomorphs of the current population are displayed on the screen, and the user indicates his/her favourite ones. In other words, the user guides the genetic algorithm (GA), which circumvents the need to develop a computational fitness function.

This seminal work, along with the influential works of Sims (1991), and Todd and Latham (1992), led to the emergence of a new research area, evolutionary art, which is characterised by the use of nature-inspired computing for artistic purposes. A thorough survey on the application of biological-inspired techniques to visual art can be found in Lewis (2007).

The use of evolutionary algorithms to create image filters and non-photorealistic renderings 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

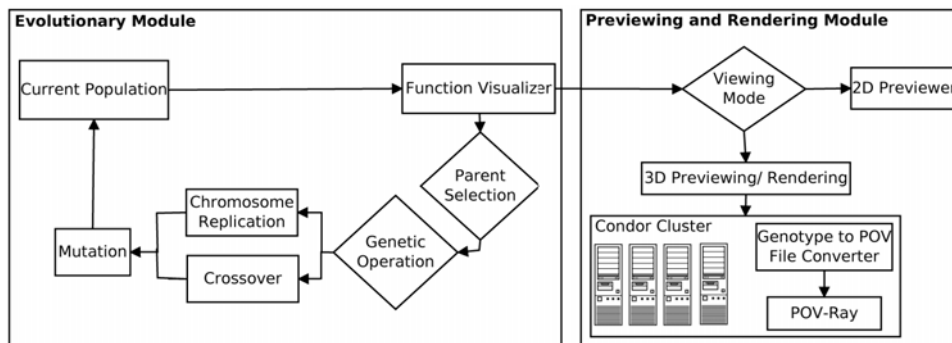
et al. (2006) and Neufeld et al. (2007), where genetic programming (GP) (Koza, 1992), multi-objective optimisation techniques, and an empirical model of aesthetics are used to automatically evolve image filters; Lewis (2004), which evolved live-video processing filters through interactive evolution; Machado et al. (2002), where GP is used to evolve image colouring filters from a set of examples; Yip (2004), which employs GAs to evolve filters that produce images that match certain features of a target image; Collomosse and Hall (2005) and Collomosse (2006, 2007), which use 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.

For the distribution of objects onto a surface, several researchers have already covered the problem: Hausner (2001) places squared-tiles in a planar surface using direction fields and Voronoi diagram to orient and distribute the tiles; Kim and Pellacini (2002) extend the previous method using a collection of tiles of arbitrarily-shape to fill an input container image. In addition to this area, a significant research effort is in progress on the context of 3D models. Fleischer et al. (1995) create patterns based on biologically inspired phenomena (e.g., physical processes of collision and adhesion). Each element is designated as a cell, which interacts to form cellular textures, this interaction arises whether automatically or the user specifies the cell's behaviour.. The texture patterns arise from the interactions of discrete elements. Prusinkiewicz et al. (1994) extend the Lindenmayer systems in a manner suitable for simulating the interactions between a developing plant and its environment. Zhou et al. (2006) propose an algorithm for 3D texture synthesis, the main idea is to repeat and stitch patterns of geometry to cover the surface of the input model. However, it is necessary a post processing step to buttress the integrity of the geometry. The work of Ma et al. (2011) uses also repetitive elements to synthesise element textures. Each element is represented by multiple samples. They encode the attributes position, size, shape, and orientation in each sample. Then, they place the samples using a neighbourhood metric. The overall distribution of the samples is minimised using an energy function through an iterative optimisation solver. Lai et al. (2006) cover a 3D mesh with rectangular tiles based on the characteristics of the surface, using also a global optimisation approach to evenly distribute the objects. A fascinating line of research by Gal et al. (2007) involves the composition of 3D shapes in order to fill the volume of an input object. The algorithm fits the elements, from a database, into the target shape. The fitting algorithm considers the partial shape similarity between the target and the element which is more likely to be the proper match. The algorithm searches in the database elements in such way that some portion of the element locally approximates the region around a given point. The fitting scheme evaluates the quality of each match based on a score function. In addition, the elements must also satisfy the following constraints: visibility, overlap limit, and proportions. Finally, Szeliski et al. (1991) use particle systems and its interactions to shape the surfaces. The system is well generalised for a wide variety of situations: it adapts to the transformations of the surface (for instance, it is possible to create particles to fill the gaps when the surface is stretched), and it allows the creation a 3D surface from a collection points. Several other works exist, however a thorough survey is beyond the scope of this article.

3 Overview of the system

Figure 1 presents the architecture of the system, which is composed of two main components: an *evolutionary* module and a *previewing and rendering* module. The evolutionary module is an expression-based GP (Sims, 1991) interactive breeding tool, which is responsible for the evolution of a population of assemblages.

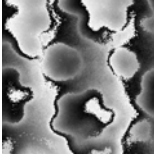



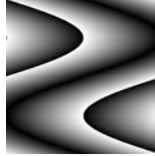
Figure 1 The architecture of the system



The user selects two parents, which generates offspring through crossover and mutation. Crossover recombines the genetic code of the parents, being, therefore, responsible for the exploitation of the characteristics that are already present in the current population. The mutation operator induces small changes in the genetic code, promoting exploration. The users preserve and select variations of the individuals, evolving images that match their preferences. Although randomness plays an important role, as evolution progresses the choices of the user are continuously steering the algorithm to a particular style of imagery, making the images increasingly refined and unique. The evolutionary module comprises a *function visualiser* (see Figure 2) that depicts a greyscale visualisation of the individuals' expression trees. As is usually the case in expression-based GP, the greyscale value of a pixel at the (x, y) coordinates is determined by the output value of the individuals' expression trees for (x, y) . Each individual is an assemblage of 3D objects. Therefore, usually, this visualisation mode does not provide enough information to allow educated choices by the user.

To see the generated assemblages the user can use a previewer. The previewer runs on the master computer, it evaluates the genotypes and places the objects accordingly. Due to the complexity of the assemblages, we have available a 3D renderer that uses a Condor-base (Tannenbaum et al., 2001) render farm. The master creates and submits several Condor jobs for each individual of the population. Each job is responsible for: converting the genotype in a persistence of vision (POV) 3D script-scene file (<http://www.povray.org/>); rendering a slice of the resulting 3D scene using POV-ray; transferring the rendered image slice to the master. The master gathers and merges the rendered image slices, displaying the images as they become available. By changing the settings of the POV-ray initialisation file, the user can adjust the quality and size of the renderings, thus also adjusting the speed.

Figure 2 Chromosomes of a sample genotype and the visualisation of the corresponding functions over $[0, 1]$

<i>Type</i>	<i>Rotation</i>	<i>Size</i>	<i>x-position</i>	<i>y-position</i>
Max(1.79, +(image, x))	Min(x, -(1.8, sin(max(y, 1.9))))	Min(y, -(min(x, x), sin(max(x, 1.9))))	abs(x)	-(sin(y), x)
				

Note: In the left, *image* is a zero-arity operator that returns the value of the x, y pixel of the source image.

4 Evolutionary engine

In this section, we describe the evolutionary module, focusing on aspects such as representation, genetic operators and genotype-phenotype mapping.

4.1 Representation

The genotype of each individual has five chromosomes (Machado and Graça, 2008; Graça and Machado, 2008): $\langle type, rotation, size, x-position, y-position \rangle$ identified respectively by the index $i \in \{1, \dots, 5\}$. Each chromosome i is an expression-tree calculated for each entry $a_{xy}^{(i)}$ of the matrix $M^{(i)} \in \mathfrak{R}^{n \times n}$, where $0 \leq a_{xy}^{(i)} \leq 1$, each tree encodes a particular aspect of the assemblage, as follows:

- *type* – the output value of the *type* expression-tree determines what object, from a pool of available ones, will be placed
- *rotation* – the rotation what will be applied to the object
- *size* – the scaling applied to the object
- *x-position* and *y-position* – the x and y coordinates where the object will be placed.

In the experiments mentioned herein, the function set is:

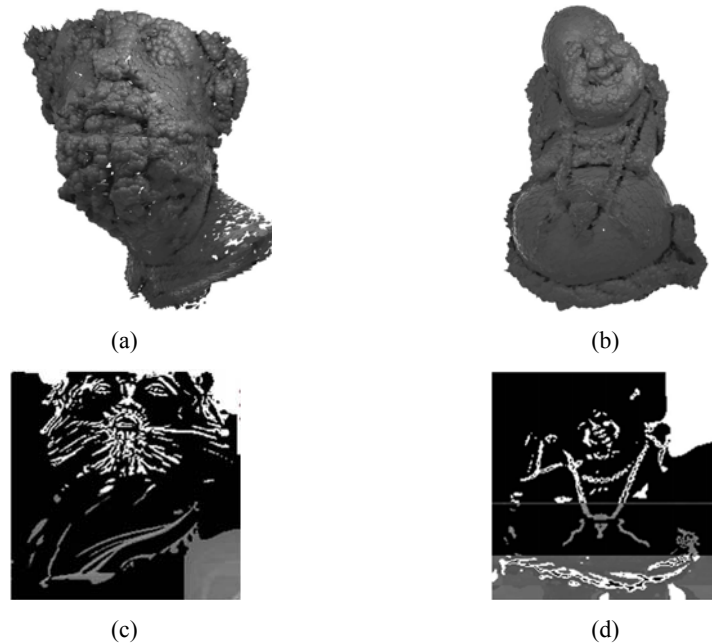
$$\{\sin, \cos, atan, \max, \min, abs, +, -, \times, \%, pow, \exp, avg\}$$

where \sin , \cos and $atan$ are the usual trigonometric operations; \max and \min take two arguments returning respectively, the maximum and minimum value; abs returns the absolute value; $\{+, -, \times\}$ are the standard arithmetic operations; $\%$ the protected division operator (Fogel et al., 1966); pow receives two arguments, returns the base raised to the power; \exp is a exponential function; avg , a function that takes two arguments (x, y) , and returns the average of the values within a square window of size m centred at (x, y) , m is specified by the user. The terminal set is:

$\{x, y, \text{getElliptical}, \text{getHyperbolic}, \text{random}_{constants}\},$

where x and y are variables; $\{\text{getElliptical}, \text{getHyperbolic}\}$, exploit the curvature of the surface (see Figure 3), and return the weight of the vertices that are locally convex and locally saddle shape at (x, y) position; $\text{random}_{constants}$ returns a random floating point value between 0 and 1. Instead of processing the geometry each time that we evaluate the individual, the values of the curvature are previously calculated and saved in a two dimensional matrix $n \times n$, Figures 4(c) and (d) show the result of the computation. The operators $\{\text{getElliptical}, \text{getHyperbolic}\}$ are obtained using the Gaussian curvature K , Kobbelt et al. (2000). In this way, we can classify the points into categories depending on the sign of K . A positive Gaussian curvature, $K > 0$, means the surface is either a peak or valley – *elliptical* points, see Figure 4(d). A negative value, $K < 0$, means the surface has saddle points – *hyperbolic* points, see Figure 4(c).

Figure 3 (a) and (b) show the result of using the operators getElliptical and getHyperbolic in the chromosome type on two different meshes. The evolved type chromosome (c) and (d) almost detects the contours of the input model, and associates the white, black and grey regions to spheres, cubes and pyramids



Note: Here, each entry of the matrix related to *type* chromosome was multiplied by 255 grey levels.

4.2 Genetic operators

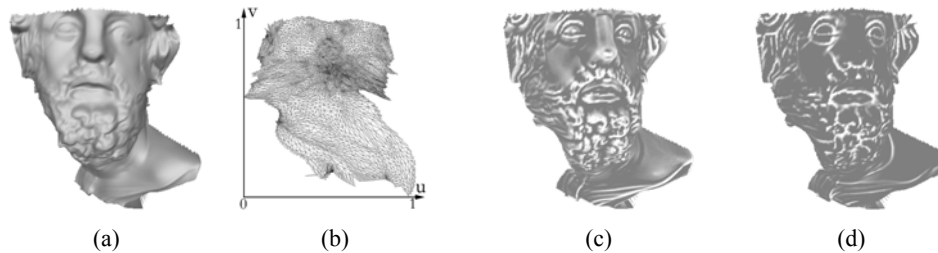
We have at our disposal three genetic operators: crossover, mutation and chromosome replication. The crossover is based on the standard GP subtree exchange crossover (Koza, 1992). It randomly selects a sub-tree from each parent and exchanges them, creating two descendants. This operator is applied to each homologous chromosome pair, meaning it

can only exchange subtrees between similar chromosomes (e.g., a subtree of the chromosome encoding size cannot be replaced by a subtree of chromosome encoding rotation). For each homologous chromosome pair there is a probability of occurrence of crossover. The mutation operator randomly selects a subtree and replaces it by a randomly created one. The chromosome replication operator propagates a specific chromosome through the entire population, allowing the user to test it in different conditions. For example, the user may feel particularly pleased with the rotations applied to the objects in one individual, and desire to use the rotation expression in all individuals. Alternatively, the user may wish to test small variations of a specific chromosome without changing the remaining ones. To address these needs, the chromosome replication:

- a copies the chromosome selected by the user to all individuals, replacing the corresponding ones
- b applies, to each individual, a mutation to the copied chromosome.

Since we have a set of five chromosomes, the evolution of the assemblages can be difficult and slow. For this reason, he/she is able to select the set of chromosomes which he/she wants to evolve at each new generation.

Figure 4 The original triangle mesh (a) and the application of parametrisation with free boundary (b). The computation of the curvature of the surface results in the identification of hyperbolic regions (c) and local convex points (d)

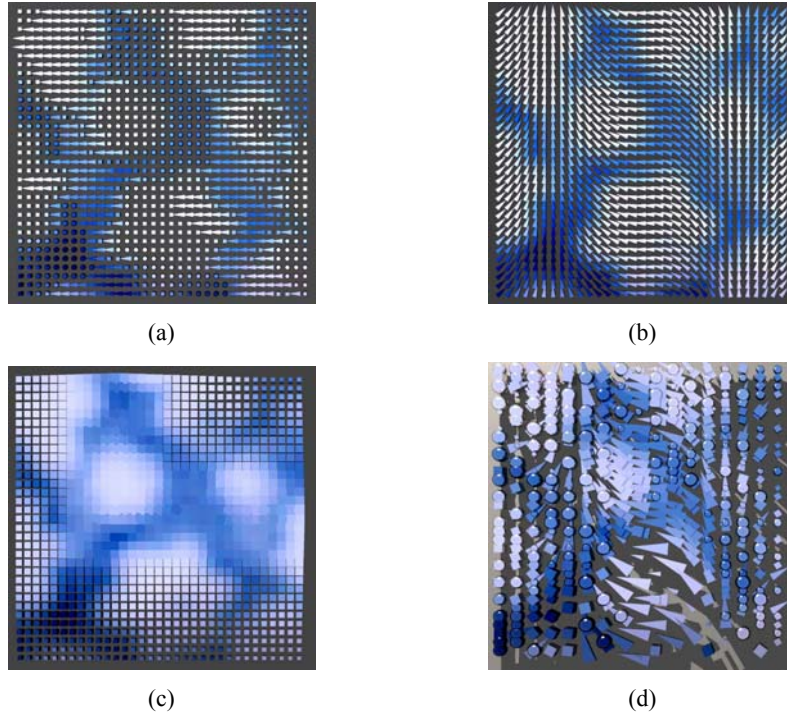


4.3 Genotype-phenotype mapping

In this section, we describe how the genetic code of an individual yields an assemblage of 3D objects. To illustrate our explanation, we resort to the genotype presented in Figure 2. For the time being, we will assume that the objects are placed following a regular 32×32 grid, and that three types of objects are available: cubes, spheres and pyramids.

The first chromosome, *type*, determines which type of object is placed. In this case, values in $]0, 0.33]$ correspond to cubes, in $[0.33, 0.66]$ to spheres, and in $[0.66, 1[$ to pyramids. The application of this chromosome, alone, would produce the 3D scene depicted in Figure 5(a). Likewise, the *rotation* chromosome determines the rotation applied to each object, and size determines the scaling applied. Figures 5(b) and 5(c) depict the results of independently applying these chromosomes, using, respectively, pyramids and cubes for easier viewing.

Figure 5 Assemblages resulting from applying to the source image the chromosomes (a) *type*, (b) *rotation*, (c) *size*. Assemblage (d) results from the simultaneous application of $\langle \textit{type}, \textit{rotation}, \textit{size}, \textit{x-position}, \textit{y-position} \rangle$ (see online version for colours)



The regular placement of the objects on a predetermined grid has characteristics that we wish to avoid, namely:

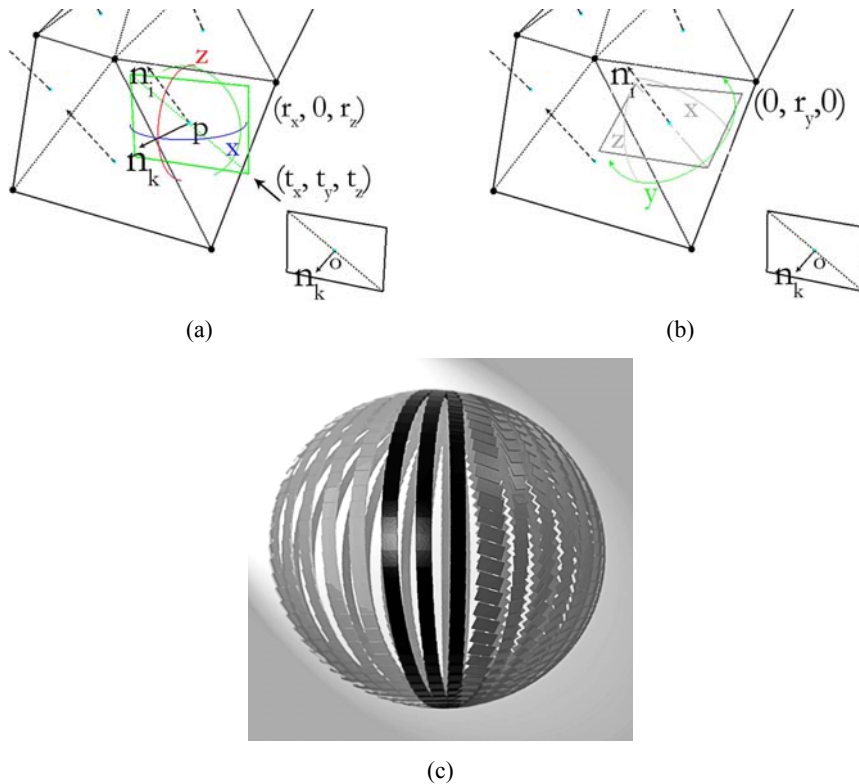
- 1 the regularity of the grid can become a visual distraction
- 2 it only allows a homogeneous distribution of the objects, making it impossible to ignore regions of the input image or 3D object, or to clutter objects on certain regions.

To overcome this limitation, we resort to the *x*- and *y*-position chromosomes. These determine the coordinates where the objects are placed. In Figure 5(d), we present the result of the application of the entire genotype, i.e., $\langle \textit{type}, \textit{rotation}, \textit{size}, \textit{x-position}, \textit{y-position} \rangle$. In this work (Machado and Graca, 2008), we present several alternative object placement strategies, including the use of dither masks to control the number of objects placed on the canvas, and we make an assessment of the experimental results attained through their use. In the present paper, object placement is entirely determined by the *x*- and *y*-position chromosomes, no dither masks are used. Up to now, we considered only the planar case (see Figure 6). Our next step is to extend the previous approach to the 3D surface (see Figures 7 and 8).

Figure 6 Source image and the corresponding planar assemblage (see online version for colours)



Figure 7 (a) The placement of the objects on the surface* (b) the chromosome *rotation* dictates the amount of rotation in y-axis of the object** (c) an example of placement of objects in the surface of a sphere (see online version for colours)



Notes: *First, the square object is placed in the origin o of the scene, then it is translated (t_x, t_y, t_z) to the position p . **This process is applied to every object introduced in the scene.

The use of parametrisation suggests a convenient way to represent the 3D surface in two dimensional coordinates. Our strategy is based on free-boundary parametrisation, whereas fixing the boundary may introduce distortions to the placement of the objects. The present work uses the least squares conformal maps method described in the literature on mesh parametrisation, Hormann et al. (2007). What follows is the description of how the objects are placed in the 3D surface. First, we start by parameterising the input mesh, which allows to attach a two dimensional coordinate system to the object [see Figure 4(b)]. Second, we iterate (for any x and $y \in 1, \dots, n$) through the matrices $M^{(iv)}$ and $M^{(v)}$, related to the x - and y -position. For each pair $(a_{xy}^{(iv)}, a_{xy}^{(v)})$ – we recall that it corresponds to the coordinates (u, v) in parameter space – we check if it is in the range of the texture coordinates of the input mesh, then we translate the object to the corresponding point p in 3D coordinates, see Figure 7(a). Third, the normal of the object n_k is aligned according to the normal n_i of the face where the point p is [see Figure 7(b)]. Fourth, we link the (u, v) coordinates to the entries of the matrices of the chromosomes $\langle type, rotation, size \rangle$ using the standard bilinear interpolation in order to extract: the scalars rotation (r), size, and the type of object to use.

Figure 8 The 3D model of a hand and the 3D assemblage



Note: In the right, the conversion of the POV script-scene to the 3D format.

5 Discussion of results

The analysis of the experimental results attained by evolutionary art systems, specially user driven ones, entails a high degree of subjectivity: our approach is thought for large-scale formats. Therefore, it is somewhat difficult to adequately convey the real look of the evolved assemblages in the space and format available for their presentation (see the web page of the project for high resolution of the results). Considering these difficulties, we focus on the presentation of assemblages created by our approach, highlighting the influence of the artistic decision in making the final outcomes. Further, a more detailed implementation issues are explored.

To characterise the strengths and limitations of the proposed method we used different input models. Figure 9 depicts the object library used in the course of the experiments. This library is composed, mainly, by shapes that are evocative of tapestry. The creation of these structures followed the rule: include objects of varying size and complexity. In this regard, the rationale is that large objects could be used to create the raw shape of the input model, whereas small and thin objects could be used to provide detail to areas of the input model that we wished to emphasise.

Figure 9 Object library used in experiments



The experimental results confirm our previous findings, that is, the user is able to guide the evolutionary algorithm to regions of the search space that are a good match to her/his aesthetic preferences. Moreover, the process is not effortless: several generations – usually 40 to 60 – are needed in order to produce assemblages that convey the artistic views of the user.

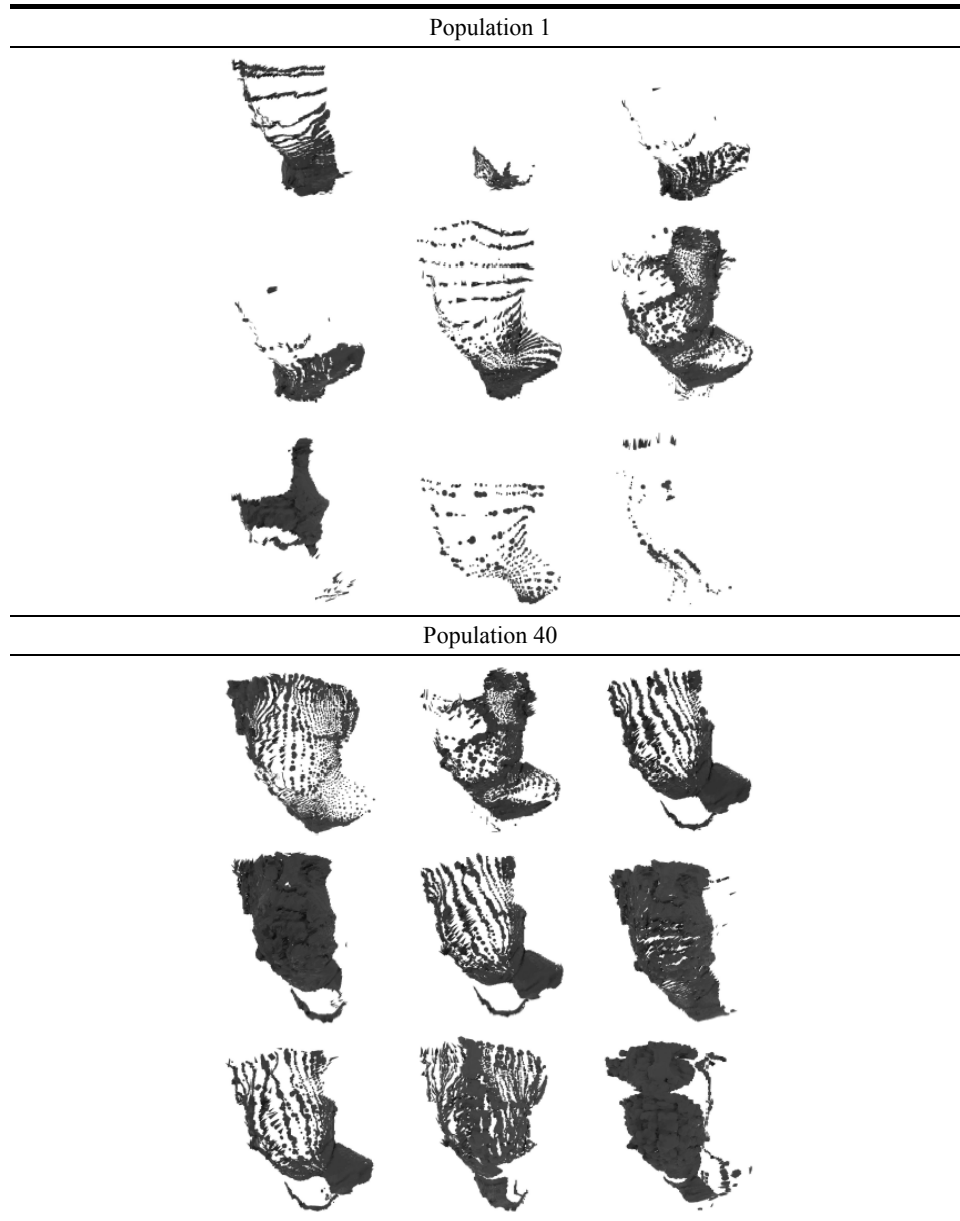
As it can be observed in Figure 10, there is an improvement in the distribution of the objects. Due to the simplicity of the x - and y -position chromosomes, the first population shows a sparse distribution of the objects. While a more interesting and even placements start to emerge in the 40th population. During the evolution phase, we always preserved the individuals that covered more area of the surface. In addition, it was interesting to notice that in the individuals that generate balanced distributions, one of the chromosomes x - or y -position tends to display an almost perfect gradient from 0 to 1, and the other one a more complex pattern. This characteristic allows an even exploration of the spatial distribution.

With respect to the issue of performance, the experiments were performed on an Intel Core2Duo, 2.8GHz, Windows machine. The computational effort to render each assemblage is considerable, and it depends mainly on the amount of objects present in the assemblage. For instance, in the first population, depicted in Figure 10, the second individual in the top row has 1,154 objects, while the 1st individual in the middle row of the 40th population has 113,349 objects. The remaining individuals have in average approximately 27,000 objects. The complexity of each individual may be overwhelming (see for instance Figure 12). As such, during the evolution process, we use a simplified version of the objects, and we compute low quality renders. Taken together, the rendering of each individual can vary approximately from 10 seconds to 4 minutes, for low and high number of objects.

It is interesting to note that like planar assemblages (Machado and Graça, 2008; Graça and Machado, 2008), the richness of the object library plays an important role. While in planar assemblages the overarching object's colour discerns the image in the assemblage, in 3D model assemblages it can be discarded, which opens new possibilities

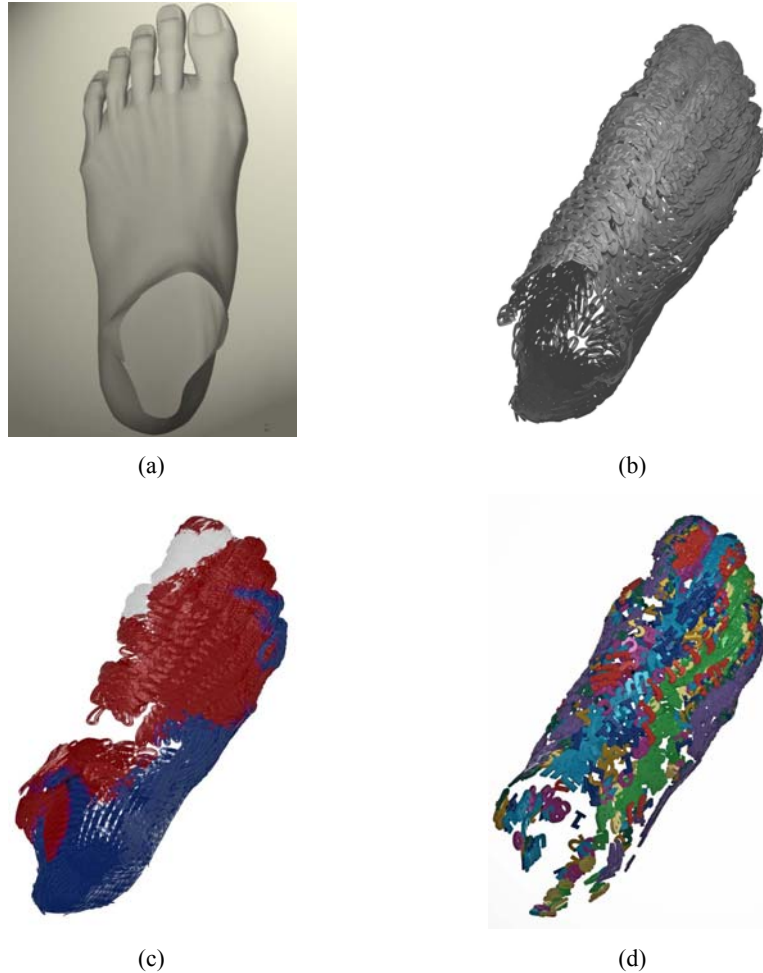
[see Figures 11(c) and 11(d)]. Furthermore, the computation of the curvature allows the evolved program to adapt to the curvature characteristics of the mesh.

Figure 10 The nine individuals of populations 1 and 40



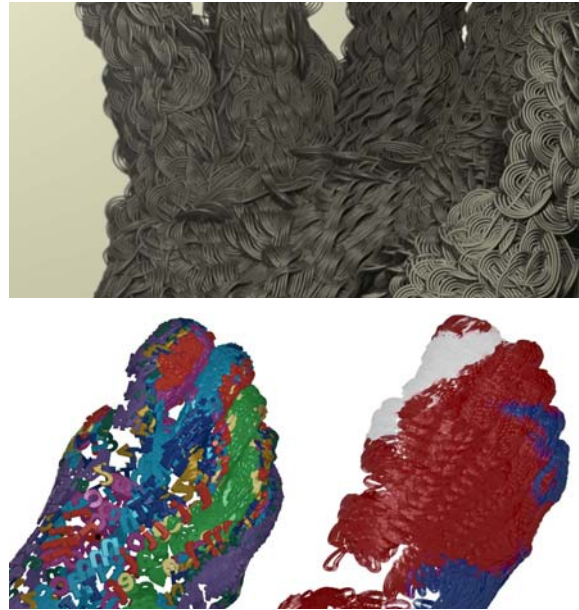
Notes: For the purposes of demonstration and to afford a better visibility of the placement, rotation, and scaling transformations we used simple primitives (cube, sphere and pyramid) in the assemblages. For this particular evolution, we used the following parameters: population size 9; probability crossover 0.25; probability of mutation 0.01; resolution to preview the individuals: 320×240 .

Figure 11 A top view rendering of the 3D model (a) and three assemblages produced using different individuals and object libraries (b, c, d) (see online version for colours)



Note: In the case of the assemblages (c) and (d) each library object has a predetermined colour.

Concerning to the placement of the objects on the surface of the mesh, the present method relies on the use of the texture coordinates to place the 3D objects. This technique can be restrictive, to see similar results with the same genotype in different 3D models, it is necessary to unfold the mesh always in same fashion. In addition, all objects with coordinates that do not belong to the UV space of the input mesh are discarded, which prevents the complete exploration of the chromosomes. However, the texture coordinates are very useful to encompass certain areas of the model, and simplifies the representation of the mesh.

Figure 12 Detail of the Figure 8 and Figures 11(c) and 11(d) (see online version for colours)

6 Conclusions

The fundamental premise being pursued in this article concerns the use of an interactive evolutionary tool for the generation of assemblages of 3D objects. In summary, the experimental results show that the initial choices done by the user – selection of the type of input and object library – have a significant impact in the final outcome. In addition, the interactive evolutionary process allows the users to explore the search space of assemblages, and to guide evolution in accordance to their artistic intent, eventually finding regions of the space that match his/hers preferences. In this way, the users are able to recognise their artistic signature in the evolved assemblages and to develop a sense of authorship of the evolved artworks.

In planar assemblages, Figure 6, the source image becomes perceptible mainly due to the colour assigned to each object and due to the applied rotations. Although the objects provide the texture, it is mostly their colouring and rotation that allows the viewer to perceive the original image. In 3D model assemblages, the colour is not required for the recognisability of the object and its use can, therefore, be freely explored. The developed system uses the parametric space. A number of alternative strategies are possible, for instance the use of the mesh's volume (Theobalt et al., 2007), which may overcome the limitations of the texture coordinates. Another approach involves the use of a particle system similar to the work of Pastor et al. (2003). The particles are fixed to the surface of the 3D models; each particle indicates a potential location of an object. To the extent that this approach succeeds, the distribution of objects dynamically adapts to the surface of the input object. Any of these strategies potentially improves the distribution of the objects.

Some of the best evolved genotypes proved their generality by producing engaging assemblages when applied to images and 3D models different that those used during the evolutionary process. These generic abilities can be explained by the functional dependence of the assemblage on the input image. But we have not foreseen them. In future work, we intend to conduct experiments with the specific goal of promoting the generalisation abilities of the genotype.

Currently, the produced artworks are printed on large-scale printer. We wish however to explore other media, that may convey better the 3D nature of the pieces. Namely, we are currently exploring the possibility of using computerised numerical control mills to create sculptures.

Acknowledgements

We would like to acknowledge the contribution of Paulo Marques to provide valuable support in the setup of the clusters and Sara Silva that presented our work at GECCO 2010. This research is partially funded by the Portuguese Foundation for Science and Technology, project PTDC/EIA-EIA/115667/2009.

References

- Collomosse, J.P. (2006) ‘Supervised genetic search for parameter selection in painterly rendering’, in *Applications of Evolutionary Computing, EvoWorkshops 2006*, pp.599–610, Budapest, Hungary.
- Collomosse, J.P. (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*, pp.39–62, Springer, Berlin, Heidelberg.
- Collomosse, J.P. and Hall, P.M. (2005) ‘Genetic paint: a search for salient paintings’, in *Applications of Evolutionary Computing, EvoWorkshops 2005*, pp.437–447, Lausanne, Switzerland.
- Darwin, C. (1859) *On the Origin of Species*, John Murray, London.
- Dawkins, R. (1987) *The Blind Watchmaker: Why the Evidence of Evolution Reveals a Universe Without Design*, W.W. Norton and Company, Inc., New York.
- Dennett, D. (1995) *Darwin’s Dangerous Idea*, Penguin Books, London.
- Fleischer, K.W., Laidlaw, D.H., Currin, B.L. and Barr, A.H. (1995) ‘Cellular texture generation’, in *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH ’95*, pp.239–248, ACM, New York, NY, USA.
- Fogel, L.J., Owens, A.J. and Walsh, M.J. (1966) *Artificial Intelligence through Simulated Evolution*, John Wiley & Sons, New York.
- Gal, R., Sorkine, O., Popa, T., Shefr, A. and Cohen-Or, D. (2007) ‘3d collage: expressive non-realistic modeling’, in *Proceedings of the 5th International Symposium on Non-photorealistic Animation and Rendering, NPAR ’07*, pp.7–14, ACM, New York, NY, USA.
- Graça, F. and Machado, P. (2008) ‘Evolving assemblages of 3D objects’, in Ochoa, A., Ortiz, M., and Santana, R. (Eds.): *Procs. of the New Creativity, The 11th Biennial Symposium on Arts and Technology*, pp.212–217, New London, USA.
- Hausner, A. (2001) ‘Simulating decorative mosaics’, *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH’01*, New York, NY, USA, ACM, pp.573–580.

- Hewgill, A. and Ross, B.J. (2003) 'Procedural 3d texture synthesis using genetic programming', *Computers and Graphics*, Vol. 28, No. 4, pp.569–584.
- Holland, J.H. (1975) *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, MI.
- Holland, J.H. (2000) 'Building blocks, cohort genetic algorithms, and hyperplane-defined functions', *Evolutionary Computation*, Vol. 8, No. 4, pp.373–391.
- Hormann, K., Lévy, B. and Sheffer, A. (2007) 'Mesh parameterization: theory and practice', in *ACM SIGGRAPH Course Notes*.
- Kim, J. and Pellacini, F. (2002) 'Jigsaw image mosaics', in *ACM Transactions on Graphics*, pp.657–664.
- Kobbelt, L., Bischoff, S., Botsch, M., Kähler, K., Rössl, C., Schneider, R. and Vorsatz, J. (2000) 'Geometric modeling based on polygonal meshes', in *Eurographics 2000*, pp.15–21.
- Koza, J.R. (1992) *Genetic Programming: On the Programming of Computers by Natural Selection*, MIT Press, Cambridge, MA.
- Lai, Y-K., Hu, S-M. and Martin, R.R. (2006) 'Surface mosaics', *Vis. Comput.*, Vol. 22, Nos. 9–11, pp.604–611.
- Lewis, M. (2004) 'Aesthetic video filter evolution in an interactive real-time framework', in *Applications of Evolutionary Computing, EvoWorkshops 2004, EvoBIO, EvoCOMNET, EvoHOT, EvoIASP, EvoMUSART, EvoSTOC, LNCS*, Vol. 3005, pp.409–418, Springer Verlag, Coimbra, Portugal.
- Lewis, M. (2007) 'Evolutionary visual art and design', in Romero, J. and Machado, P. (Eds.): *The Art of Artificial Evolution: A Handbook on Evolutionary Art and Music*, pp.3–37, Springer, Berlin, Heidelberg.
- Ma, C., Wei, L-Y. and Tong, X. (2011) 'Discrete element textures', *ACM Transactions on Graphics*, Vol. 30, No. 62, pp.1–10.
- Machado, P. and Graca, F. (2008) 'Evolutionary pointillist modules: evolving assemblages of 3D objects', in Giacobini, M., Brabazon, A., Cagnoni, S., Di Caro, G., Drechsler, R., Ekárt, A., Esparcia-Alcázar, A., Farooq, M., Fink, A., McCormack, J., O'Neill, M., Romero, J., Rothlauf, F., Squillero, G., Uyar, A. and Yang, S. (Eds.): *Applications of Evolutionary Computing, Lecture Notes in Computer Science*, Vol. 4974, pp.453–462, Springer, Berlin/Heidelberg, Naples, Italy.
- 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*, Vol. 1, No. 2, pp.209–219.
- Machado, P., Romero, J., Santos, A., Cardoso, A. and Pazos, A. (2007) 'On the development of evolutionary artificial artists', *Computers & Graphics*, Vol. 31, No. 6, pp.818–826.
- 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*, pp.335–356, Springer, Berlin, Heidelberg.
- Pastor, O.M., Freudenberg, B. and Strothotte, T. (2003) 'Real-time animated stippling', *IEEE Comput. Graph. Appl.*, Vol. 23, No. 4, pp.62–68.
- Prusinkiewicz, P., James, M. and Měch, R. (1994) 'Synthetic topiary', *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH'94*, New York, NY, USA, ACM, pp.351–358.
- 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*, pp.1087–1094, IEEE Press, Vancouver, BC, Canada.
- Sims, K. (1991) 'Artificial evolution for computer graphics', *ACM Computer Graphics*, Vol. 25, No. 4, pp.319–328.

- Szeliski, R., Szeliski, R., Tonnesen, D. and Tonnesen, D. (1991) ‘Surface modeling with oriented particle systems’, in *Computer Graphics*, Vol. 26, No. 2, pp.185–194.
- Tannenbaum, T., Wright, D., Miller, K. and Livny, M. (2001) ‘Condor – a distributed job scheduler’, in Sterling, T. (Ed.): *Beowulf Cluster Computing with Linux*, MIT Press.
- Theobalt, C., Rössel, C., de Aguiar, E. and Seidel, H-P. (2007) ‘Animation collage’, in *ACM SIGGRAPH EUROGRAPHICS Symposium on Computer Animation 2007*.
- Todd, S. and Latham, W. (1992) *Evolutionary Art and Computers*, Academic Press, Winchester, UK.
- Yip, C. (2004) *Evolving Image Filters*, Master’s thesis, Imperial College of Science, Technology, and Medicine.
- Zhou, K., Huang, X., Wang, X., Tong, Y., Desbrun, M., Guo, B. and Shum, H-Y. (2006) ‘Mesh quilting for geometric texture synthesis’, *ACM Trans. Graph.*, Vol. 25, No. 3, pp.690–697.