INTERACTIVE DESIGN OF AESTHETIC ABSTRACT TEXTURES BY COMPOSITION PRINCIPLES AND THE LAWS OF CHANCE

Luis Alvarez, Nelson Monzón and Jean-Michel Morel

Luis Alvarez (mathematician, educator), Dep. Informática y Sistemas, Universidad de Las Palmas de Gran Canaria. Campus de Tafira, 35017, Spain. Email: <lalvarez@ulpgc.es> .
Website:< https://sites.google.com/site/luisalvarezsite/>
ORCID: 0000-0002-6953-9587

Nelson Monzón (computer engineer, educator), Centre de Mathématiques et Leurs Applications (CMLA), Ecole Normale Supérieure de Cachan. ENS Cachan. 61, avenue du Président Wilson, 94235 Cachan cédex, France.
Email: <monzon@cmla.ens-cachan.fr>.
Website:<https://www.linkedin.com/in/nelsonmonzonlopez/> ORCID:0000-0003-0571-9068

Jean-Michel Morel (mathematician, educator), Centre de Mathématiques et Leurs Applications (CMLA), Ecole Normale Supérieure de Cachan. ENS Cachan, 61, avenue du Président Wilson, 94235 Cachan cédex, France.
Email: <moreljeanmichel@gmail.com>.
Website:<https://sites.google.com/site/jeanmichelmorelcmlaenscachan/> ORCID: 0000-0002-6108-897X

Submitted: <02/07/2018>

Abstract

The automatic synthesis of abstract textures is, to some extent, envisageable. Indeed, as put forward by abstract art theoreticians, an abstract picture can be conceived of as a tree of elementary shapes interacting by a short list of composition laws such as occlusion, exclusion, bordering, and by rendering rules such as transparency, tessellation, color selection. By randomizing the shape generator and these composition and rendering laws, one obtains an algorithm generating random abstract textures (AAT). We have designed an online user-friendly tool <www.ipol.im/aat> implementing this algorithm.

Keywords: abstract textures, interactive design, grammar algorithms.

In 1917, Jean Arp acknowledged the incorporation of chance into the principles of artistic creation in the following terms: "*the law of chance can only be experienced through complete devotion to the unconscious* (...) *using this process 'according to the law of chance', isn't per se, using chance* (...) I *wanted to create new appearances extracted of man new forms*". Arp explored these ideas creating abstract collages. Fig.1 shows him casually disposing basic shapes on a canvas.

Fig. 1. Jean Arp 1917. *Collage with Squares Arranged According to the Laws of Chance*

Our goal is to develop a fully random creation program using a formal approach and an interactive online computing facility.

The objective of our design facility is to explore randomly all ways to combine shapes on a surface. The fabrication laws being applied uniformly on the image, the result of each random trial is what we call *aesthetic abstract texture* (AAT). We have devoted significant attention to the design of an online interface allowing artists and designers to create/modify easily their own AAT styles. The main challenge is to define an adequate organization of the parameters governing an ATT. Texture parameterization should be

powerful enough to generate a large variety of different new "appearances". At the same time it should be simple enough to be easily understandable and modifiable by a user without programming skills. We have jointly designed the grammar organizing the texture and the online interface that we invite the reader to test at <www.ipol.im/aat>.

One of the first art expressions using random shape distributions is the *Cueva de las Manos* painting (Fig. 2). It illustrates well the problematic of abstract random painting: the hands were disposed randomly but with the intention to cover the whole space. The orientation of the hands is randomly distributed, and they are combined through occlusion or transparency. These basic abstract composition rules will be used, among others, to generate our AATs.

Fig. 2. Cueva de las manos, Argentina, 9.000 years ago.

Ideally, the generator should authorize all possible shapes and shape combination rules, at the very least those observed in abstract painting and decorative arts. This may seem an impossible task. Fortunately, the founders of abstract art have made a remarkable simplification and formalization of the painting generation problem, in syntony with parallel formalizations of mathematical logic or Gestalt theory.

Abstract painting theoreticians were indeed confronted with three new problems that were implicitly resolved in figurative painting. The first one was to invent shapes. The second one was to combine them on a flat surface, and the third one was to attribute them a color. Noticeably, abstract painting did not abandon the idea that basic elements are shapes. There are some exceptions though, like Pollock's endeavour to entrust shape formation to physical processes like spilling.

The basic abstract shapes have been often reduced to the simplest geometric shapes like polygons, circles, smooth curves, or thick strokes. This has never been considered a big drawback by the likes of Kandinsky, Klee, Kupka or Malevich. Indeed, even when working with the simplest basic shapes, many degrees of freedom are left to the painter, like their size, color, orientation and relative disposition. Enough complexity emerges from the interactions of these basic elements. Nevertheless, abstract painting faces the same physical constraints as figurative painting. The addition of a new shape with its color on the canvas can be either subtractive or additive. In the subtractive case the new shape hides its background; in the additive case, the color of the new added shape is combined with the background's colors. This creates the classic transparency effect, caused in the physical world by light shafts, shadows or glasses.

Our facility combines basic abstract shapes by applying randomly the most classic shape composition laws. In that way, the styles of many abstract painters can be casually retrieved, just because their parameters are straightforward shape combination techniques like the ones of Figs. 1 and 2..

Our algorithm extends the numerous algorithms reproducing specific styles by mimicking the shapes and composition rules used by one or the other abstract painter. For example Kirsch et al. [1] proposed algorithms to emulate some of the Diebenkorn and Miró works. Tao et al. [2] reproduce Malevich's style, Zhang et al. [3]

Kandinsky's style and Taylor et al. [4] Pollock's style.

Barla et al. [5] proposed to create new images by random spatial arrangements of specific patterns, a task which requires modeling shape interactions. Similarly, we do not aim at reproducing any particular painter style. Our goal is to enable artists and designers to test new styles, based on random combinations of shapes, composition and rendering rules. In [6] we presented the mathematical description of a texture generator based on an organization in successive layers. As we argue next, the tree organization that we propose in this paper is far more general. In [7] we show a number of collections of textures made with the ATT generator.

Shapes: Beyond the very basic abstract shapes like circles, rectangles, triangles, etc.., we have added in the generator more sophisticated ones like for instance rosettes used from the antiquity as decorative ornaments, or active lines as described by Paul Klee in [8]: "*An active line on a walk, moving freely, without goal. A walk for a walk's sake*".

Composition laws: In addition to the basic shape combination effects by *occlusion* and *transparency* our facility algorithm authorizes *tessellation*, *light spot* and shape *contouring*. *Tessellation*, used by painters like Paul Klee, Robert Delaunay, Joaquín Torres García or Bruce Gray is based on the assignment of independent colors to each connected component generated by the boundaries of intersecting shapes. *Light spot* consists in an attenuation of the colors intensity according to the distance to a point named the light spot center. *Contouring* shape boundaries in black is a classic boundary enhancement technique used in cartoons and by artists like Mondrian or Picasso. It produces a stained glass effect. To illustrate the bridge between abstract and decorative art, our algorithm also authorizes a setup of the image similar to a scarf or a Persian carpet where several different ATTs build successive borders surrounding a central ATT.

In general, the computer generation of AAT's requires programming skills and tools. This is the case for the Processing programming language [9] designed for artists and designers. The potential of creative coding can also be improved using code bending [10]. The design with our online facility does not require programming skills. We propose an experiential learning model. In a few minutes users can create new AAT's by editing one of a predefined collection of AAT's or by creating a new one from scratch using the automatic creation facility. The full understanding of the AAT parameters can be done through experiments and it only requires an hour's training.

FORMALIZATION OF THE AAT DESIGN

To formalize the design of an AAT, we argue that the painting's or image's natural organization is a tree where the root is the whole image, and where the general construction rule is to combine new shapes inside -or around- a parent shape. In that way any painting can be formalized as a tree of shapes. What makes the result different is the choice of shapes, the choice of colors and the shape composition rules at each tree node. Hence, an AAT is specified when a tree data structure is given, and when at each node shape, color and composition rules are specified.

For instance in the AAT presented in Fig. 3 the first level of the tree is composed by the image subdivision given the central rectangle and one border strip band. The next levels of the tree specify, for each node, the way it is filled with children shapes respecting a number of composition rules. The user never specifies manually the choice and location of a shape in the AAT. The shapes are automatically sorted in the parent shape following composition rules and random sampling.

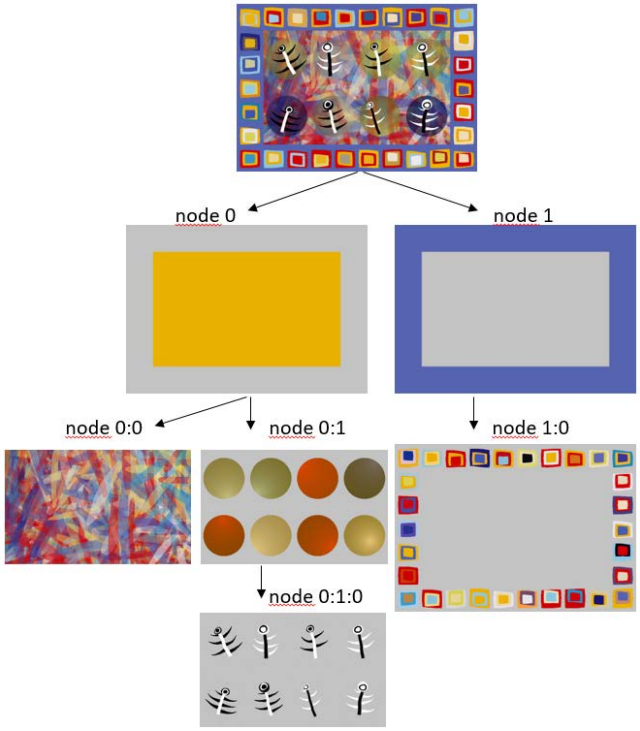

Fig 3. Illustration of the tree data structure of an aesthetic abstract texture (AAT). In each node, composition rules, shapes and rendering properties are fixed randomly and independently.

In this tree organization the texture parameters are easily modified by a friendly and intuitive interface. Roughly speaking, at each tree node the generator chooses by random sorting (and the user can manually modify each choice):

1. The kind of basic shape used in the node
2. The spatial relation of the shapes with respect to the parent node. Shapes can be distributed in several random distribution rules or following a choice of regular distributions inside or around the parent shape.
3. The geometric transformation (rotation, scaling, etc..) applied to the basic shapes to obtain the shapes to include in the texture
4. The shape interaction. The shapes can be allowed or forbidden to intersect, to exceed the parent shape, to meet the boundary of the parent shape, etc.
5. Coloring: each shape is attributed a random colour taken from a palette, possibly a contour and a light spot.
6. The shape color's rendering with respect to former shapes: (transparency, tessellation, occlusion)

AAT generation configuration

There are two ways to select the basic shapes:

1. Automatic shape generation. The algorithm generates automatically basic shapes such as triangles, circles, rectangles or more complex ones such as rosettes, Polllock-like active lines, Kandisky eyes, etc.
2. The list of basic shapes can be increased at will by the user: Shapes can be obtained from a silhouette drawn in an image (see Fig. 4). In the online interface, users can pick a single shape or a collection of shapes.

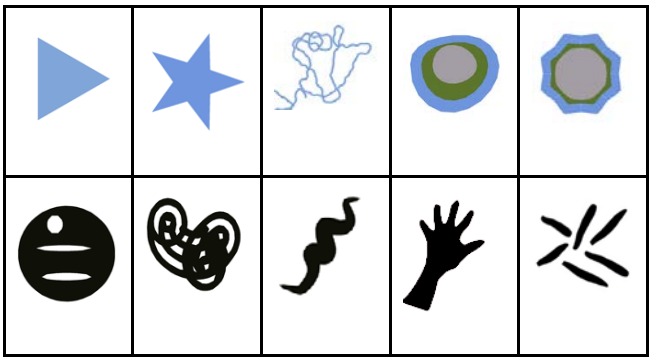

Fig. 4. Illustration of some shapes generated automatically (in blue) and some obtained from silhouettes (in black).

We also need to define the way the shapes are distributed with respect to the shapes in the parent node. In the current version of the generator, shapes can be distributed as follows:

1. Random distribution. The locations of the shapes are distributed randomly inside the shapes of the parent nodes.
2. Regular distribution. The shapes are distributed inside the parent shapes following a regular lattice.
3. Contouring. The shapes are distributed along the boundary of the parent shapes.
4. Mondrian type. The parent shape is filled using Mondrian type rectangle distribution. (This more peculiar distribution was picked to illustrate the versatility of the designing structure.)

Alternatively, Loi et al. [11] have proposed shape distributions based on partitioning, mapping and merging operators.

Once the way the shapes are distributed is fixed, we define a default shape diameter which determines the default shape size and the number of shapes. The number of shapes is fixed to cover the parent node area using shapes with the default shape size. The lower the default shape diameter, the larger the number of shapes to include.

The generator applies a random geometric affine map to the selected basic shape database to generate new shapes. Affine maps are parameterized by the following 2x2 matrix that composes four intuitive operations: a rotation, an horizontal squeeze, a second rotation of the squeezed shape, and a zoom:

$$\mathbf{A} = \mathbf{s}\begin{pmatrix} \cos\alpha_2 & \sin\alpha_2 \\ -\sin\alpha_2 & \cos\alpha_2 \end{pmatrix}\begin{pmatrix} \mathbf{s}_x & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{pmatrix}\begin{pmatrix} \cos\alpha_1 & \sin\alpha_1 \\ -\sin\alpha_1 & \cos\alpha_1 \end{pmatrix}$$

For each shape the algorithm selects randomly the parameters of the affine transformation in a reasonably wide range of values (which, again, can be modified by the user).

We also use the following shape interaction rules when including new shapes in the tree:

- Inclusion with respect to parent shape: a new shape is added only if it is included in the parent shape.
- Exclusion with respect to brothers: new shapes are added only if they do not intercept brother shapes associated to the same parent shape
- Exclusion with respect to other shapes in the node: A new shape is added only if it does not intercept other shapes in the same node level but associated to different shapes.

Once the shape tree structure is created, an important issue is the way it is rendered to create the AAT result. The following rendering configurations provide a variety of rendering options (which are again chosen randomly and can be modified manually with the online interface).

Texture rendering configuration

- Color palette. Associate to each shape a random color taken in a palette given by an image.
- Transparency. Each shape is associated a transparency factor.
- Light spot. When activated, the intensity of the colors in the points of the shape is attenuated accordingly to the distance to a point sorted inside the shape.
- Tessellation. The image tessellation generated by the object boundaries is used to render the texture “behind” the new drawn shape. Colors are associated independently to each new connected component of the image.
- Shape border delineation. Optionally, a black stripe border is delineated along the shape contour.
- Drawing child shape outside its parent. This option decides if a shape extends or not outside its parent shape.

INTERACTIVE ONLINE AAT GENERATOR

The ATT generator is fully automatic. This means that by clicking "run", a random ATT will be generated. An ATT is both a texture and a texture style. Indeed, the same ATT can be "rerun" by changing any of its parameters, but also only its seed random parameters. Then a "similar" but everywhere distinct image will be generated.

We wanted users to create and modify AAT's in an easy and intuitive way. To develop this online interface we used the Image Processing On Line (IPOL) facilities. IPOL is a journal enabling the creation of sophisticated online interfaces well adapted to many image processing tasks. All information required to reproduce an AAT is stored in a parameter configuration file that can be edited on line using the interface. The usual starting point for a user is to select a particular AAT model among the ones proposed as examples, or to generate automatically a new AAT by clicking "run".. Once the AAT model is created, the users can modify it in two ways: they can modify directly the text parameter file (expert mode) or they can modify the most significant texture parameters using buttons and sliders of the interface. These parameters are the following:

- The color palette used to color the texture
- The basic shapes used to generate the AAT (users can upload their own collection of shapes given by silhouettes)
- Resolution : by changing the resolution users change the texture scale (the lower the resolution the smaller the shapes)
- Reshuffle shapes : this option generates a similar AAT changing the random shape positions
- Reassign colors : without changing the palette, the assignment of colors to shapes is randomly modified.
- Change drawing order : change the order the shapes are drawn.

We show below a variety of AAT's obtained using the online interface. The examples of Figs 5 to 7 were not modified by a user: they came out by a random choice of all parameters. Fig 8 is the result of modifications using the online facility; In these examples we can easily recognize elements from the styles of Mondrian, Pollock, Miró and Kandinsky. These textures can include elements of known styles but more importantly the combination of shapes and composition rules in a tree organization allows the creation of new styles of AAT's.

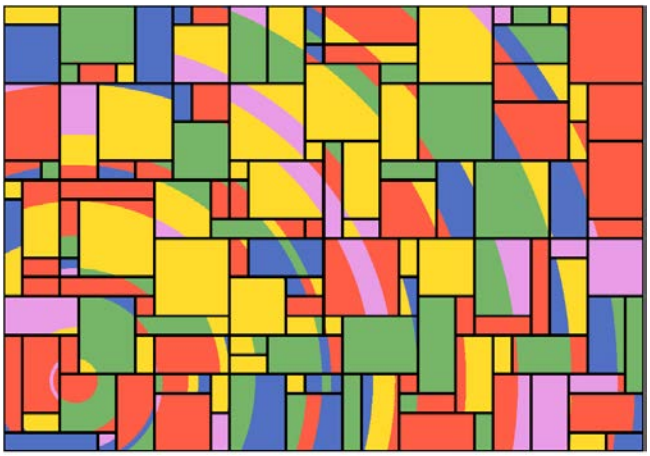

Fig 5. AAT created using the online interface with recognizable elements of Mondrian's style.

Fig 6. AAT created using the online interface with recognizable elements of Pollock's style.

Fig 7. AAT created by the online interface with recognizable elements of Miró's style.

Fig 8. AAT created using the online interface with recognizable elements of Kandinsky's style. This one was modified manually in the online interface. .

CONCLUSION

In this paper we presented a new grammar for the design of aesthetic abstract textures (AAT). Each texture is organized as a collection of composition rules, shapes and rendering properties stored using a data tree structure. The advantage of this approach is that the information is simple to manage and modify but at the same time powerful enough to create a large variety of textures implicitly containing many well known styles. We have devoted significant attention to the design of a friendly online interface <www.ipol.im/aat> to create/modify textures using this formal organization. Our aim is that artists and designers, without programming skills, can easily use this online facility to create/modify their own AAT's

We can finish this description with a number of limitations for our approach. We are aware that a much more complete shape generator should be built, based on abstract principles. Complex shapes could be constructed by combining elementary shapes with concatenation rules such as bordering, periodization, symmetrization, recursive tree structures, etc. as developed for example in Klee's lecture notes [8].

We have not yet put all reasonable image tiling schemes. For example recursive subdivision schemes or random tessellations with straight lines are popular ways to subdivide a shape, not yet implemented, with an exception: we did include a Mondrian like subdivision scheme. A last caveat is that the textual tree structure does not completely describe the ATT: a drawing order must be specified, and by default it is the order in which the tree's nodes are being read.

Could our random image generator create realistic images? From a probabilistic viewpoint realistic painting is very unlikely. Indeed, the natural forms created by geological or biological processes and by humans are very specific. Thus, realistic compositions are very unlikely with our generator, while familiar abstract set-ups do arise spontaneously. Certain natural textures are, however, prone to random mathematical modeling, as shown in [12].

Bibliographical information:

Luis Alvarez is a professor at the Universidad de Las Palmas de Gran Canaria, Spain. His research areas are computer vision and mathematics.

Nelson Monzón is a postdoctoral researcher and software engineer at Ecole Normale Supérieure Paris-Saclay, France. His main scientific interests are computer vision and image/video processing.

Jean-Michel Morel is a professor of mathematics at Ecole Normale Supérieure Paris-Saclay, France. His research areas are image processing and computer vision.

References and Notes

1. Joan L. Kirsch and Russel A. Kirsch, *The Anatomy of Painting Style: Description with Computer Rules*, Leonardo **21**, No. 4 (1988) pp.437–444.

2. Wenyuan Tao, Yaxuan Liu and Kang Zhang, *Automatically Generating Abstract Paintings in Malevich Style*, 2014 IEEE/ACIS 13th International Conference on Computer and Information Science (ICIS), Taiyuan, 2014, pp. 201-205.

3. Kang Zhang and Jinhui Yu, *Generation of Kandinsky Art*, Leonardo **49,** No. 1 (2016) pp. 48-54.

4. Richard P. Taylor, Adam P. Micolich and David Jonas, *The Construction of Jackson Pollock's Fractal Drip Paintings*. Leonardo **35,** No. 2 (2002) pp. 203-207.

5. Pascal Barla, Simon Breslav, Joëlle Thollot, François Sillion, and Lee Markosian. *Stroke pattern analysis and synthesis*, In Computer Graphics Forum, **25**, (2006) pp. 663–671

6. Luis Alvarez, Yann Gousseau, Jean-Michel Morel and Agustín Salgado, *Exploring the Space of Abstract Textures by Principles and Random Sampling*, Journal of Mathematical Imaging and Vision **53**, No. 3, (2015) pp. 332-345

7. Luis Alvarez and Jean-Michel Morel, *Image Design by Principles and Random Sampling*, <http://www.ctim.es/ImageSynthesis/>

8. Paul Klee, *Pedagogical Sketchbook*, (Praeger Publishers, 1972)

9. C. Reas and B. Fry, *Processing: A Programming Handbook for Visual Designers and Artists*, 2007, MIT Press.

10. Ilias Bergstrom and R. Beau Lotto, *Code Bending: A New Creative Coding Practice*, Leonardo **48,** No. 1 (2015) pp. 25-31.

11. Hugo Loi, Thomas Hurtut, Romain Vergne and Joëlle Thollot, *Programmable 2D Arrangements for Element Texture Design*, ACM Transactions on Graphics **36**, No. 3 (2017)

12. Carola-Bibiane Schönlieb and Franz Schubert, "*Random simulations for generative art construction–some examples*." Journal of Mathematics and the Arts 7.1 (2013): 29-39.