Engineering Applications of Artificial Intelligence 32 (2014) 124–135

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

From design methodology to evolutionary design: An interactive creation of marble-like textile patterns

Shufang Lu [a,b], P.Y. Mok [b,*], Xiaogang Jin [c]

[a] College of Computer Science and Technology, Zhejiang University of Technology, China
[b] Institute of Textiles and Clothing, The Hong Kong Polytechnic University, Hong Kong
[c] State Key Lab of CAD&CG, Zhejiang University, China

ARTICLE INFO

Article history:
Received 2 June 2013
Received in revised form
1 January 2014
Accepted 25 February 2014
Available online 9 April 2014

Keywords:
Artificial evolution
Design methodology
Textile pattern design
Marbling textiles
Generative art design

ABSTRACT

In this paper, by the integration of design methodology theories with evolutionary computation, a new design system is developed to evolve preferred designs on complex marbling patterns using interactive 'perceptual selection'. The system is formulated in a way to assist the productive–deductive–inductive design reasoning process of the users. Therefore, complex mathematical functions do not cognitively overload the designers, who are released for more critical tasks of aesthetic assessment and new design rules induction. With the implementation on a graphics-processing unit (GPU), real-time complex marbling patterns can be created by the system. The system encourages creativity in the design process and accelerates new design generation. In addition, the resulting patterns fulfil the textile industry requirements of repeat and can be output as vector images.

1. Introduction

Originating from Darwin's theory of natural selection, evolutionary computational techniques are capable of solving long-standing complex optimisation problems (Kicinger et al., 2005; Corne and Bentley, 2001). There are also reported applications in the literature for evolutionary creative designs (Gero, 2002; Lewis, 2008). This paper aims to integrate design research theories with evolutionary computation to propose an effective computer system for creative marbling textile designs.

Design research theories cover a large variety of topics (Kroes, 2002; Darke, 1979; Akin and Lin, 1995; Lawson, 2005; Thomas and Carroll, 1979; Cross, 1984). Among all of them, a strong process orientation is observed in the design methodology literature (Kroes, 2002). Design process is started from the perception of design needs and terminated in a final description of a particular design configuration (Cross, 1984; Finkelstein and Finkelstein, 1983), as illustrated in Fig. 1. Each stage of the design process is itself an iterative sequence of the steps, sub-processes or operations (Zeng and Cheng, 1991). Furthermore, the design process involves both the concepts of population and evolution: multiple designs are assessed and evaluated in the process; successful designs are evolved from previous formulated design criteria or rules. From a philosophical perspective, rational design is a cyclic, iterative process of productive, deductive and inductive (PDI) reasoning (March, 1976). Design generation, the core step in the design process, is to seek certain characteristics in the form of a design proposal to achieve the desired service, based on previous knowledge and some general presuppositions. As depicted in Fig. 1, design generation is accomplished by productive reasoning. After design generation, the performance of the design proposal is deductively predicted by the application of relevant theories to the particular design proposal. Deductive reasoning is used in the design evaluation and analysis stage. Next, the design and its expected characteristics are inductively evaluated to generalise new and modified suppositions for improved proposals in a new round of the design cycle, which is accomplished by inductive reasoning.

As discussed, design process has a number of similar concepts with evolutionary design. This paper aims to integrate productive–deductive–inductive design reasoning with evolutionary computational techniques. Therefore, new designs may emerge and old ones may receive new potential by simulating the iterative design process as artificial evolution. A computer-aided marbling textile design system built with mathematical marbling functions is used to illustrate how the design process can be modelled as artificial evolution, based on the PDI philosophical model of design reasoning. The mathematical functions are used to simulate the productive reasoning in the design process for visualising results of various marbling operations. Thus, designers would not be cognitively overloaded in comprehending the mathematics but be released for more critical tasks – those of the subjective assessment on the aesthetic value of the work by deductive reasoning and of making

* Corresponding author. Tel.: +852 2766 4442; fax: +852 2773 1432.
E-mail address: tracy.mok@polyu.edu.hk (P.Y. Mok).

http://dx.doi.org/10.1016/j.engappai.2014.02.015
0952-1976/© 2014 Elsevier Ltd. All rights reserved.

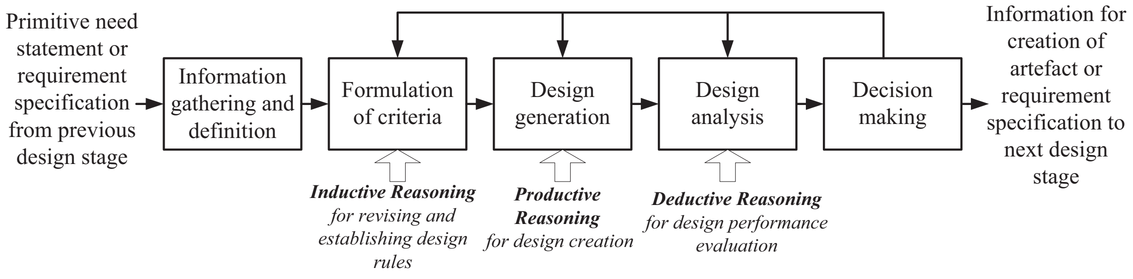

Primitive need statement or requirement specification from previous design stage
Information gathering and definition
Formulation of criteria
Design generation
Design analysis
Decision making
Information for creation of artefact or requirement specification to next design stage
Inductive Reasoning for revising and establishing design rules
Productive Reasoning for design creation
Deductive Reasoning for design performance evaluation

Fig. 1. Design process model and production–deduction–induction design reasoning.

the creative leap in inductive reasoning for generalising new and successful rules. The main contributions of this paper are summarised as follows:

1. This paper integrates design research theories with evolutionary computation to develop an effective computer system for creative marbling textile designs. The system simulates iterative PDI stages of the design process as artificial evolution, in which designers and the system are responsible for different reasoning processes in the loop (with reference to Fig. 1).
2. Mathematical marbling models are combined with evolutionary computation to create marbling textile patterns. To the best of our knowledge, this method is the first and only evolutionary marbling textile environment currently available. The evolutionary computation is based on vector images, which is a little-explored topic.
3. The method improves the mathematical marbling model (Lu et al., 2012) in two respects. Firstly, two new pattern functions are added to the mathematical marbling model. Secondly, the resulting patterns fulfil the textile industry requirements of repeat and can be output as vector images.

2. Related literature

2.1. Computer-aided textile design

Computer techniques are widely used in the textile modelling and visualisation (Grishanov et al., 2011). Computer aided textile designs can be classified into two categories. The first type is simple design such as floral motifs, where designers develop the patterns directly on machines or computer systems. The second type is complex design, such as Shibori, Batik and marbling. The creation of such complex design is obtained by experiment in laboratory. For example of marbling, designers create astonishing vibrant patterns by dropping colour paints onto a liquid surface and stirring the surface with marbling tools. The generated marble pattern is then transferred to the fabric by gently placing a cloth on top of the paints. Such laboratory work on pattern design is time consuming and tedious and places great demands on the designers' experience, skill and effort. The created patterns are difficult to predict as they vary depending on the chemical and physical properties of the materials and the design process must restart from scratch should any mistake be made. Traditional CAD systems scan the artwork obtained from laboratory experiments, and reproduce the designs on CAD systems with copy-and-paste editing and then introduce various repeat structures and colourways by colour separation and recolouring functions (Wilson, 2001). In short, the creation of complex textile patterns still relies on manual experimental work. Traditional CAD systems cannot aid such complex design generation. It is interesting to note a few attempts recently on computer-simulated complex dying and printing processes (Moimoto and Ono, 2010; Wyvill et al., 2004; Shamey et al., 2005; Lu et al., 2012). For example, Akgun (2004) developed a physical-based marbling system by numerically solving complicated fluid flow equations. Real-time design of marbling patterns by such physical simulation method is not possible because the system must resolve the fluid equations for both the velocity field and density field with a new set of parameters at each time step; users have to wait for a long time to see the results.

2.2. Evolutionary design of textures and images

Evolutionary art uses evolutionary computation techniques to evolve creative images or aesthetically pleasing structures (Lutton, 2006; Secretan et al., 2008; PBS, 2008). The evolutionary art creation systems include genetic programming-like art systems (Sims, 1991; Wiens and Ross, 2002; Machado and Cardoso, 2003; Muni et al., 2006), agent-based art systems (McClintock and Yen, 2008; Unemi, 2002) and generative art systems and so on. The genetic programming-like evolutionary method is widely used for the design of raster images. It works as follows: the colour for each pixel of the image is evaluated by its texture-space coordinate under the mathematical formulae or symbolic expression (e.g. lisp function, noise functions). The genotype consists of mathematical functions and terminals. Terminals are usually the variables (x, y) that correspond to the coordinates in the image grid. The phenotype is a raster image of size $w \times h$ (w and h are the width and height of the image). To calculate the phenotype from the genotype, the colour for each pixel is calculated as the function value of the expression for each (x, y) coordinate. Although the genetic programming art system is capable of generating images at any resolution, it costs more space and computational time as the size or resolution of the image increases.

Compared with raster images that operate on pixels, vector images that operate on primitives like points, lines, curves and polygons have a number of advantages, including small file size, scalability and ease of editing. However, evolutionary design of vector images is a little-explored topic. Only a couple of examples of genetic design using vector images are known. den Heijer and Eiben (2012) proposed the use of scalable vector graphics (SVG) to evolve representational images from existing images. Bergen and Ross (2012) developed a system called 'JNetic' to evolve sets of discrete geometric primitives (points, lines, curves and polygons) to compose vector images. However, these methods use evolutionary computation merely for matching the visual aspects of a target image, rather than generating new patterns. By contrast, the method proposed in this paper evolves new textile patterns in the form of vector images.

3. System formulation

3.1. Mathematical marbling pattern functions

Mathematical models such as topological methods are used to represent the textile structures (Grishanov et al., 2009). In this paper, mathematical marbling functions which are based on topological computer graphics are used to simulate designers' productive reasoning in the design of marble textiles. In conventional marbling art (Maurer-Mathison, 1999), designers (marblers) first place the background liquid in a tray and then drop or sprinkle colour paints onto the liquid surface to create an initial design. Next, marblers stir the surface using marbling tools, such as a stylus, brushes and combs. Complex marbling designs emerge as the marblers run the tools back and forth on the liquid surface. Once the marblers are satisfied with the pattern created, they then apply a sheet of paper or fabric on top of the paints to capture the pattern.

Lu et al. (2012) presented mathematical marbling functions to represent initial designs and five marbling operations including tine-line, comb, wavy, circular tine-line and vortex. In order to increase the diversity, two new functions of stylus and ripple are introduced in this paper. A design system is developed using the seven defined mathematical marbling functions as the generative representation. Designers are allowed to visualise the marbling operation effects in real time on computers. For easy reference, these functions are recorded below and details can be found in Lu et al. (2012).

First of all, an initial design is created with a few paint drops. In this paper, vector-based elements including points and lines are used to represent the designs. Paint drops, represented as a series of circles, are the basic shape in the initial design. Given an initial paint drop at centre $C_0(x_0, y_0)$ with radius r_0, the circle can be approximated by an inscribed regular n-gon. The value of n is chosen according to the radius of the paint drop. The points $\boldsymbol{P}(x, y)$ on the circle are calculated by the following equations:

$$\begin{cases} x = x_0 + r_0 \cos\theta \\ y = y_0 + r_0 \sin\theta \end{cases} \tag{1}$$

where $\theta = 2\pi i/n$ $(i = 0, 1, \ldots, n-1)$ and $n = 5r_0$.

Subsequently, additional paint drops are placed on the liquid surface. The points on the original circle are deformed by the new paint drops with radius r_1 and are displaced radially from its centre $\boldsymbol{C}_1$. The points $\boldsymbol{P}(x, y)$ are deformed to new position $\boldsymbol{Q}$ as follows:

$$\boldsymbol{Q} = \boldsymbol{C}_1 + (\boldsymbol{P} - \boldsymbol{C}_1)\sqrt{1 + r_1{}^2/|\boldsymbol{P} - \boldsymbol{C}_1|^2} \tag{2}$$

Once an initial design composed of a number of paint drops is created, marblers use different tools to stir the liquid surfaces to create patterns. These tools are the corresponding mathematical functions of tine-line, comb, wavy, circular tine-line, vortex, stylus and ripple. The inputs for each of these functions are the coordinates of points $\boldsymbol{P}$ and corresponding parameters of the marbling functions and the outputs are the points with new coordinates $\boldsymbol{Q}$.

Tine-line pattern function is responsible for manipulating a pattern by running tine-lines through it in any direction. It is governed by

$$\boldsymbol{Q} = \boldsymbol{P} + \frac{\alpha\lambda}{(d+\lambda)}\boldsymbol{M} \tag{3a}$$

where the scalars α and λ control the maximum shift and sharpness of the shift gradient. $\boldsymbol{M}$ is the unit vector in the direction of the tine-line, which has a start point $\boldsymbol{U}(x, y)$ and an end point $\boldsymbol{V}(x, y)$; $\boldsymbol{N}$ is a unit vector perpendicular to the line. $\boldsymbol{M}$ and $\boldsymbol{N}$ are calculated from $\boldsymbol{U}$ and $\boldsymbol{V}$ of the tine-line. The distance from $\boldsymbol{P}$ to the line is calculated by

$$d = |(\boldsymbol{P} - \boldsymbol{U}) \cdot \boldsymbol{N}|. \tag{3b}$$

Comb pattern function is responsible for designing a pattern by running evenly spaced multiple parallel tine-lines, which move as a rigid assembly. As each tine-line has a mapping function (3a), the number of mapping functions for the comb is the sum of all the tine-lines. The computation time is substantially increased as the number of tine-lines increases. Alternatively, a single function is used to represent the displacement from the set of parallel tine-lines as

$$d' = s/2 - |f \bmod(d, s) - s/2|. \tag{4}$$

$\boldsymbol{U}$ and $\boldsymbol{V}$ are the start point and end point of the first tine-line, d is the distance from $\boldsymbol{P}$ to the line, and s is the spacing between the tine-lines. In this way, the mapping function is computed only once by replacing d in (3b) with d' in (4), regardless of the number of parallel tine-lines.

Wavy pattern function is responsible for generating wavy patterns in any direction t. The mapping is represented as

$$\boldsymbol{Q} = \boldsymbol{P} + f(\boldsymbol{P} \cdot (\sin t, -\cos t))(\cos t, \sin t), \tag{5}$$

where f is the sinusoidal function and its formula is $f(x) = A \sin(\omega x + \psi)$. A, ω and ψ represent the amplitude, wavelength and phase, respectively.

Circular tine-line pattern function is responsible for designing a circular tine-line pattern. Under this operation, points $\boldsymbol{P}$ are mapped to $\boldsymbol{Q}$ by

$$\boldsymbol{Q} = \boldsymbol{C} + (\boldsymbol{P} - \boldsymbol{C})\begin{pmatrix} \cos(\beta\theta) & \sin(\beta\theta) \\ -\sin(\beta\theta) & \cos(\beta\theta) \end{pmatrix}, \tag{6a}$$

$\boldsymbol{C}(x, y)$ is the centre of the circular tine line, the angle subtended at $\boldsymbol{C}$ is $\theta = l/(|\boldsymbol{P} - \boldsymbol{C}|)$, where the length of the displacement arc is $l = \alpha\lambda/(d+\lambda)$ and $d = ||\boldsymbol{P} - \boldsymbol{C}| - r|$. The parameter β controls the direction of the circular tine-line pattern as follows:

$$\beta = \begin{cases} -1 & \text{clockwise}, \\ 1 & \text{counter clockwise}. \end{cases} \tag{6b}$$

Vortex pattern function is responsible for designing vortices patterns. It can be obtained using the same mapping function as the circular tine-line (6a and 6b) except for the displacement term

$$d = |\boldsymbol{P} - C|. \tag{7}$$

Stylus pattern function is a tool to create freehand patterns, such as delicate floral designs. Let $\boldsymbol{U}$ and $\boldsymbol{V}$ be the start and end points of the stylus stroke and $\boldsymbol{L}$ be the line on which $\boldsymbol{U}$ and $\boldsymbol{V}$ lie. In contrast to the tine-line operation where d is calculated as the minimum distance from $\boldsymbol{P}$ to $\boldsymbol{L}$, d is computed as the minimum distance from $\boldsymbol{P}$ to the line segment $\boldsymbol{UV}$. Let $\boldsymbol{T}$ be the perpendicular projection of $\boldsymbol{P}$ onto $\boldsymbol{L}$, then

$$d = \begin{cases} |\boldsymbol{P} - \boldsymbol{U}| \cdot \boldsymbol{N} & \text{if } \boldsymbol{T} \text{ lies between } \boldsymbol{U} \text{ and } \boldsymbol{V}, \\ \min(\boldsymbol{PU}, \boldsymbol{PV}) & \text{Otherwise}. \end{cases} \tag{8}$$

The mapping function is the same as the tine-line pattern function (3a).

Ripple pattern function is a patterning tool used to create a punctuated and undulating design, which looks like rippling waves swaying back and forth on an ocean surface. With f representing the sinusoidal function, the mapping function is defined as:

$$\boldsymbol{Q} = \boldsymbol{P} + f(\boldsymbol{P}_y - \boldsymbol{P}_x) \tag{9}$$

Table 1 summarises the seven mathematical pattern functions and the encoding parameters. Fig. 2 illustrates the seven pattern functions; each function is applied once on the same initial state in Fig. 2(a). Although tine-line has the same mapping function and parameters as stylus, similarly for the case of circular tine-line and vortex, different results are generated. These seven functions can be used together multiple times to create complex marbling textile patterns.

3.2. System overview

The system is constructed based on the design process theory. Design theory reveals that design is some kind of evolution instead of a single instance of inspiration. Preferred designs evolve from repeated trial-and-error practices approved by a designer's judgement and collected through his experience. The PDI philosophical model of rational design involves 3 iterative stages: productive reasoning creates a novel composition; deductive reasoning predicts and evaluates design performance; and inductive reasoning accumulates habitual notions and evolves and generalises rules for improved designs. The mathematical functions presented in Section 3.1 accomplish the production stage because they can create and visualise pattern designs instantly on computers. The next essential step of the design process is the evaluation of designs by deductive reasoning. It is preferable that human artists or designers, instead of computers, carry out the design evaluation because artistic evaluation is often subjective and typically based on aesthetic appeal (Bentley, 1999). By comparing different designs, designers accumulate rules and modify compositions by inductive reasoning, for improved designs in the next cycle of the process. In this stage, a population of diverse designs is necessary for designers to induce rules for the preferred pattern creation.

Table 1
Functions and their corresponding parameters.

Operation	Corresponding parameters
Tine-line	α, λ, $\boldsymbol{U}$, $\boldsymbol{V}$
Comb	α, λ, $\boldsymbol{U}$, $\boldsymbol{V}$, s
Wavy	A, ω, ψ, t
Circular tine-line	α, λ, r, $\boldsymbol{C}$, β
Vortex	α, λ, r, $\boldsymbol{C}$, β
Stylus	α, λ, $\boldsymbol{U}$, $\boldsymbol{V}$
Ripple	A, ω, ψ

The population concept and the iterative design improvement favour modelling design activity as an evolution process. In this paper, an evolutionary design system is developed to create marble textile patterns, enabling the 'evolution' of pattern designs using interactive 'perceptual selection'. The following subsection describe in detail the design representation, design generation with 5 genetic operations (random alteration, function recombination, swapping, changing initial state, and lock and undo) and design evolution. The design generation and design evolution are formulated according to the design practices and design process theory, thus to encourage creativity in design and accelerate the generation of new designs. Fig. 3 shows the workflow of the marbling textile design generation.

Step 1: Generate an initial population of individual designs with a population size 9.
Step 2: User evaluates the aesthetic performance and selects preferred designs.
Step 3: Generate new designs using operations: random alteration, function recombination, swapping, or changing initial state, sometimes apply with lock function.
Step 4: Repeat the process from Step 2 until the user is satisfied with the generated designs.

3.3. Design representation

The mathematical functions described in Section 3.1 are powerful and flexible tools for marble pattern design. As shown in Fig. 2, the final shape of a pattern is determined by both the initial state and the sequence of marbling operations executed onto the initial design. The representation of individual marble patterns consists of (a) terminals and (b) a set of pattern functions, as shown in Fig. 4. Terminals are the initial design, which include points and lines. The second part of the design representation is a sequence of pattern functions with their parameter values. The value range and data type of each parameter for the seven pattern functions (Table 1) are shown in Table 2.

Users can decide the number of pattern functions to apply on the initial design. A new marbling pattern, in vector images, can be

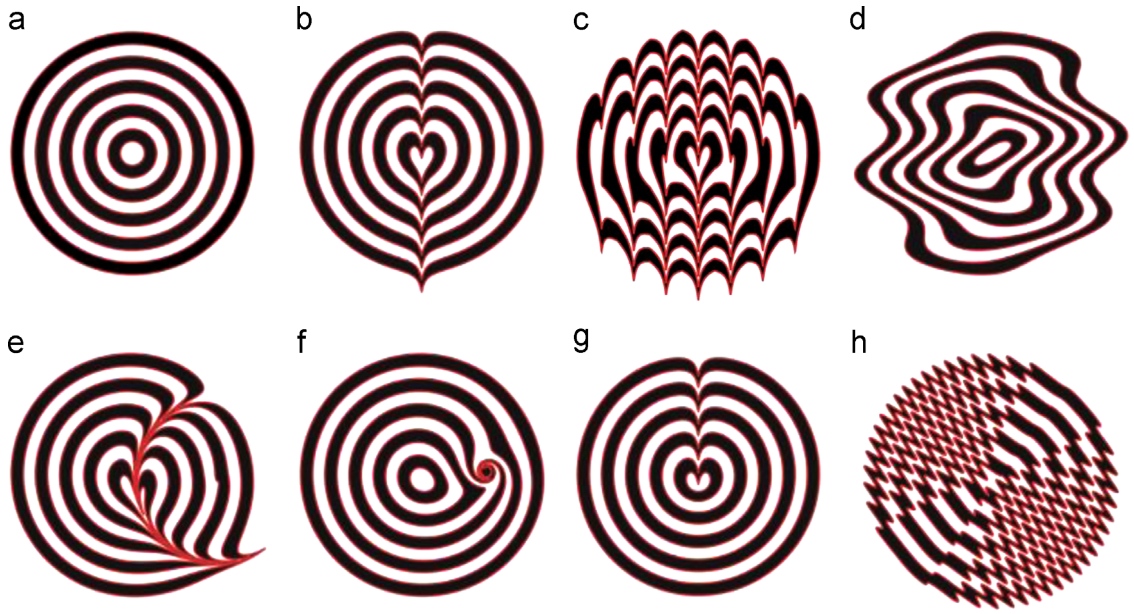


Fig. 2. (a) An initial state, (b) tine-line result on the initial state, (c) comb result on the initial state, (d) wavy result on the initial state, (e) circular tine-line result on the initial state, (f) vortex result on the initial state, (g) stylus result on the initial state, and (h) ripple result on the initial state.

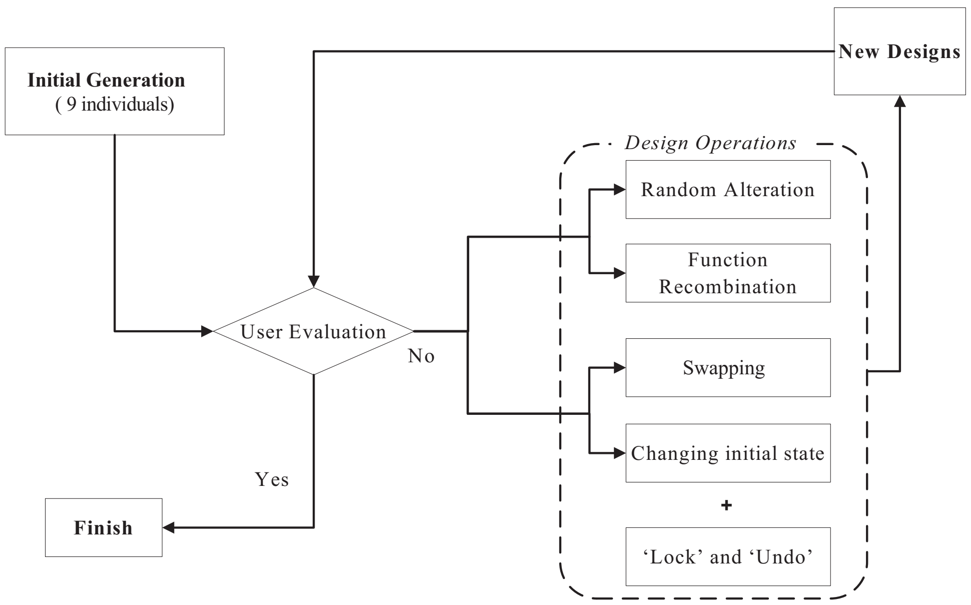


Fig. 3. The workflow of marbling textile pattern design generation.

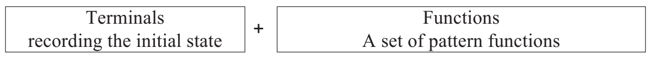


Fig. 4. Representation of a design by concatenating the terminals part and the functions part.

created by applying the set of pattern functions to the initial state. Each point in the initial state is transformed from its current position to a new position defined by the function value of the relevant mathematical function. Fig. 5 shows an example of (a) an initial state, (c) a set of 10 pattern functions and (b) the resulting pattern.

3.4. *Population size and initial population*

In a design process, particularly in the early stages, alternative ideas of design should be presented. Designers should consider different proposals and iteratively modify the designs in order to decide on the best solution to the design problem. In the proposed system, users work collaboratively with the system. A reasonable response time is necessary for any interactive design system. More computation time is required with an increasing number of individuals or increasing complexity of individual designs, thus user may need to wait for a long time to see the designs. Moreover, interactive design systems would not prefer a large population size because of the size constraint of human–computer interface. It is also because user may be cognitively overloaded for assessing a large number of designs. Alternatively, a small population size has the drawback of limited coverage of the search space. As suggested by Bentley (1999), users can quickly judge all the individuals in every generation by using population size less than 10. To achieve a good balance between the quality of the solution and user requirements in terms of response time, user interface and user fatigue, a 3-by-3 grid layout with population size of nine is chosen (see Fig. 12).

Random initial population is used in most evolutionary design systems reported in the literature. In this paper, two methods are adopted to generate the initial population. One is by random generation and the other is by loading existing successful designs, which can be developed by the interactive mathematical marbling systems (Lu et al., 2012). Such formulations of the initial population agree with the design practice that designers are often asked to adapt and develop, based on ideas or existing designs from their customers.

Table 2
The parameter value range and data type.

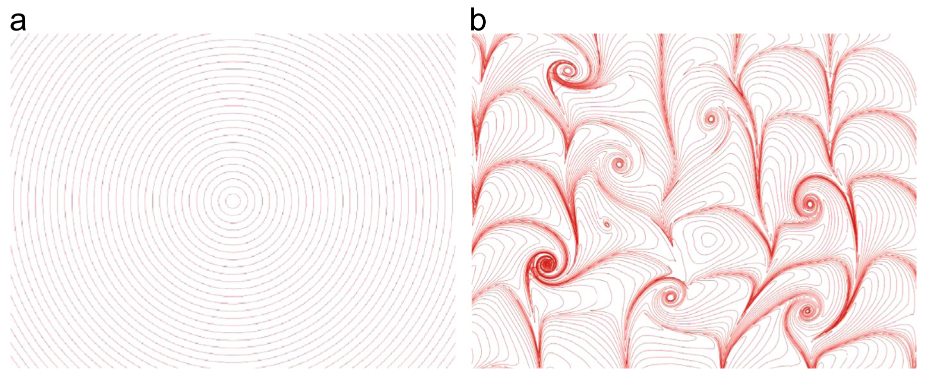
Parameter	Value range	Type
A	[80, 200]	Integer
λ	[8, 30]	Integer
$\boldsymbol{U}(x,y)$	[(0, 0),(800, 800)]	(integer, integer)
$\boldsymbol{V}(x,y)$	[(0, 0),(800, 800)]	(integer, integer)
S	[40, 160]	Integer
A	[3, 50]	Integer
ω	[0.03, 0.3]	Float
ψ	[0.0, 2π]	Float
T	[0.0, 2π]	Float
R	[10, 200]	Integer
$\mathbf{C}(x,y)$	[(0, 0),(800, 800)]	(integer, integer)
B	$-1/1$	Integer

3.5. *Design generation*

In evolutionary computation, genetic operators such as crossover and mutation are used to introduce variations to the population and reproduce a new generation of individuals, which imitate biological evolution. The method of design is analogous to evolution in nature that new designs are created by introducing something 'new' to what already exists. With reference to the design presentation (Section 3.3), new designs can be created by altering either the terminals part or the pattern function part. We propose four operations, including random alteration, function recombination, swapping and changing initial state to simulate design rules for generating new designs. The random alterations as well as the function recombination operations create new designs by introducing variations to the pattern function part of the individuals. Comparatively, the swapping and the changing initial state operations introduce variations to the terminals part (i.e. initial state) instead of the pattern function part. Two function tools 'lock' and 'undo' are developed to support the design operations.

3.5.1. *Random alteration*

In evolutionary computation, mutation operator is used to guarantee that the new generation is not simply a mixture of inherited characters. Similarly, a 'random alteration' is developed in our system to introduce variations to any selected designs.



c

Pattern Functions									
1. Tine-line	2. Tine-line	3. Vortex	4. Circular tine-line	5. Circular tine-line	6. Vortex	7. Vortex	8. Circular tine-line	9. Circular tine-line	10. Vortex

Fig. 5. (a) Is the initial state of the pattern, (b) is the resulting pattern, which is obtained by applying the set of pattern functions in (c) to the initial state.

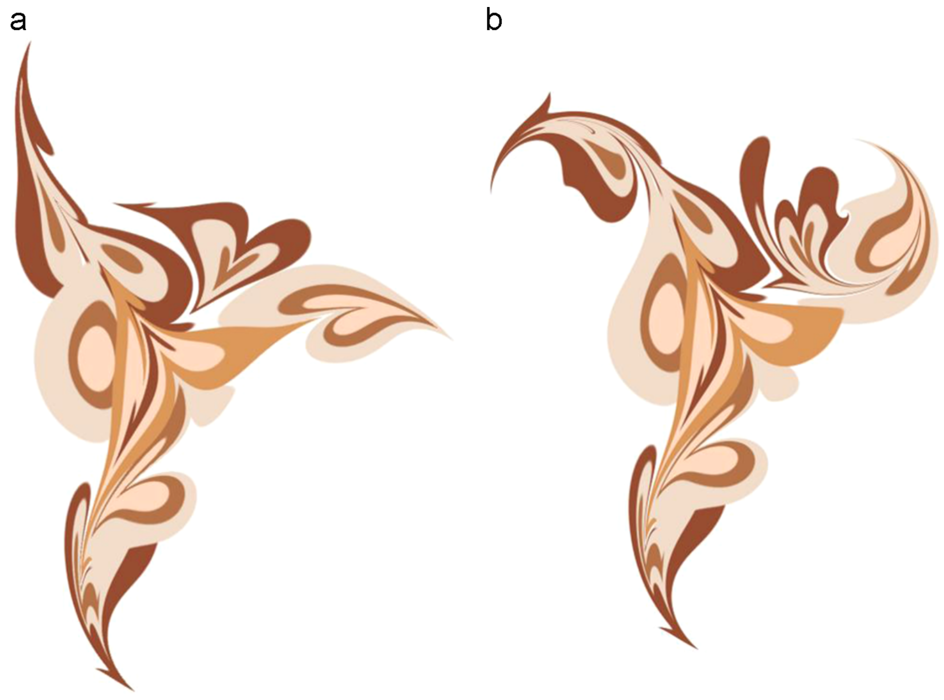


Fig. 6. Random alteration: pattern in (a) is altered to pattern in (b).

The operation is similar to uniform mutation operator of traditional evolutionary algorithms, the pattern function of selected design is altered by replacing its parameters with another set of parameters. The new set of parameters is randomly generated with respect to its valid ranges, as shown in Table 2. Otherwise, the pattern function is copied without any change. Fig. 6 shows an example of random alteration; the pattern in Fig. 6(a) is changed to the pattern in Fig. 6(b) and the parameter values of the pattern functions for the two patterns are listed in Table 3.

3.5.2. *Function recombination*

In evolutionary computation, crossover is used to mix the genes of two selected parents to create two offspring. Following the similar concept of genetic crossover, a 'function recombination' operator is developed to exchange some pattern functions of two selected parents at a randomly selected position. A recombination result is shown in Fig. 7 where the exchange of function parameter values is shown in Fig. 8.

If more than two designs are selected, a scheme is needed to match two individuals for the creation of new designs. In evolutionary computation, the matching is accomplished by mating operator, which randomly matches any two individuals based on the fitness values of the individuals. Therefore, some individuals may be selected more than once whereas some may not be selected at all. To give users more control over the design process, we reduce the level of randomness. Moreover, to ensure diversity of the designs, we consider as many different possible combinations of the selected individuals as possible. To do so, assuming n individuals (designs) are selected, the total number of possible combinations is C_n^2 reproducing $2C_n^2$ new designs. In our method, all possible combinations are included in the recombination operation. If $2C_n^2$ is a value larger than the population size, the first nine results are displayed. Users usually select the most

Table 3
The parameter values of the pattern functions for the two mutation patterns in Fig. 5.

	Pattern (a)	Pattern (b)
1. Tine-line	(100, 20, [473, 669], [273, 242])	(100, 20, [473, 669], [273, 242])
2. Stylus	(106, 20, [598, 25], [122, 784])	(106, 20, [598, 25], [122, 784])
3. Circular tine-line	(104, 15, 130, [369, 204], 1)	(104, 15, 130, [369, 204], 1)
4. Circular tine-line	(105, 13, 10, [297, 738], −1)	(**132, 8, 80, [455, 551], 1**)
5. Circular tine-line	(106, 13, 20, [538, 376], −1)	(**146, 9, 88, [163, 575], 1**)

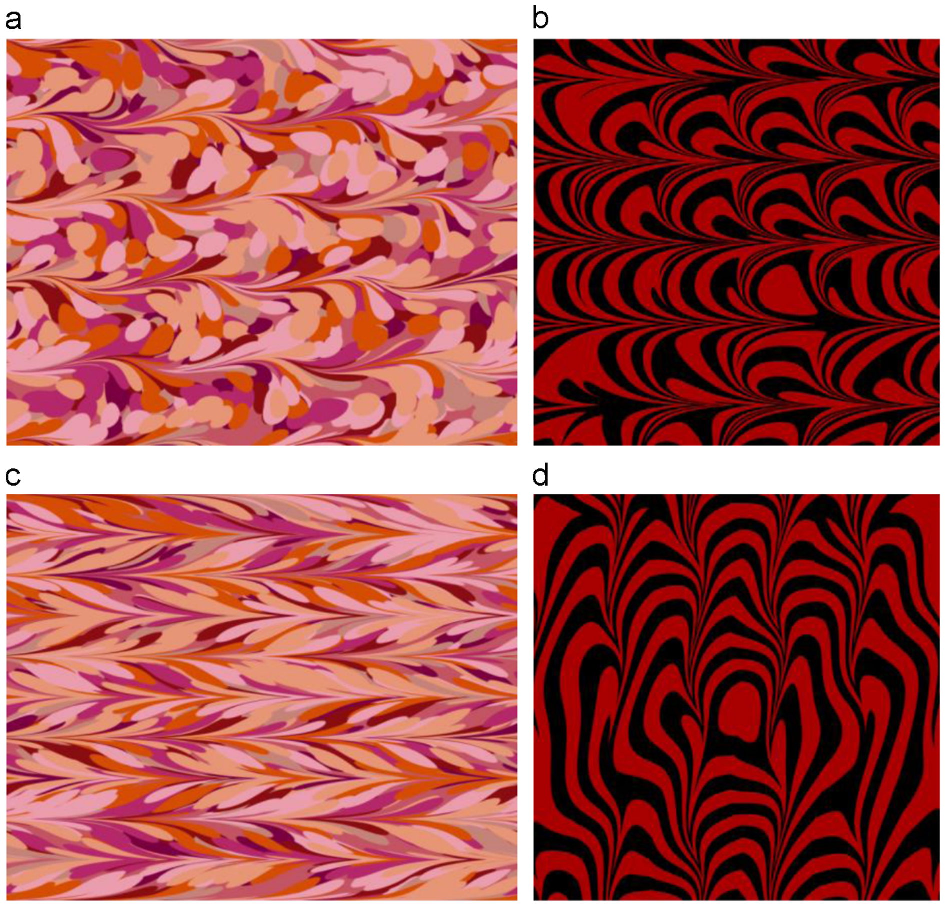


Fig. 7. Function recombination: (a) design 1 pattern; (b) design 2 pattern; (c) the resulting new design 1 pattern; and (d) the resulting new design 2 pattern.

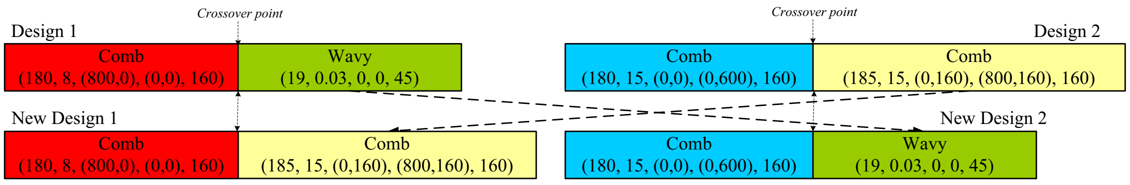


Fig. 8. Schematic illustration of function recombination operation.

preferred ones first, then the less preferred ones. Thus, the recombination results of the most preferred ones can often be retained as a new population of designs.

3.5.3. *Swapping head and tail*

Swapping operation has similar concept as the function recombination operation; the selected designs will swap their initial states (head) and pattern function part (tail), as illustrated in Fig. 9. Swapping is applied in a similar manner as recombination to ensure wide coverage of different possible combinations between selected designs. The operation is analogue to the fixed position crossover operator in traditional evolutionary computation.

3.5.4. *Changing initial state*

As mentioned earlier, pattern shape is determined by both its initial state and the pattern functions used. Apart from introducing

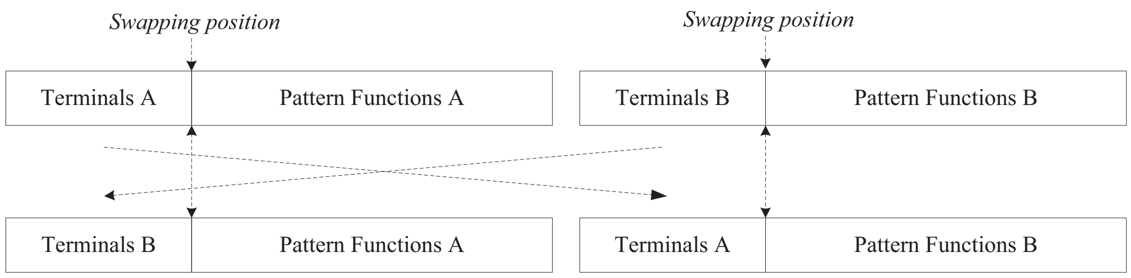


Fig. 9. Schematic illustration of swapping operation.

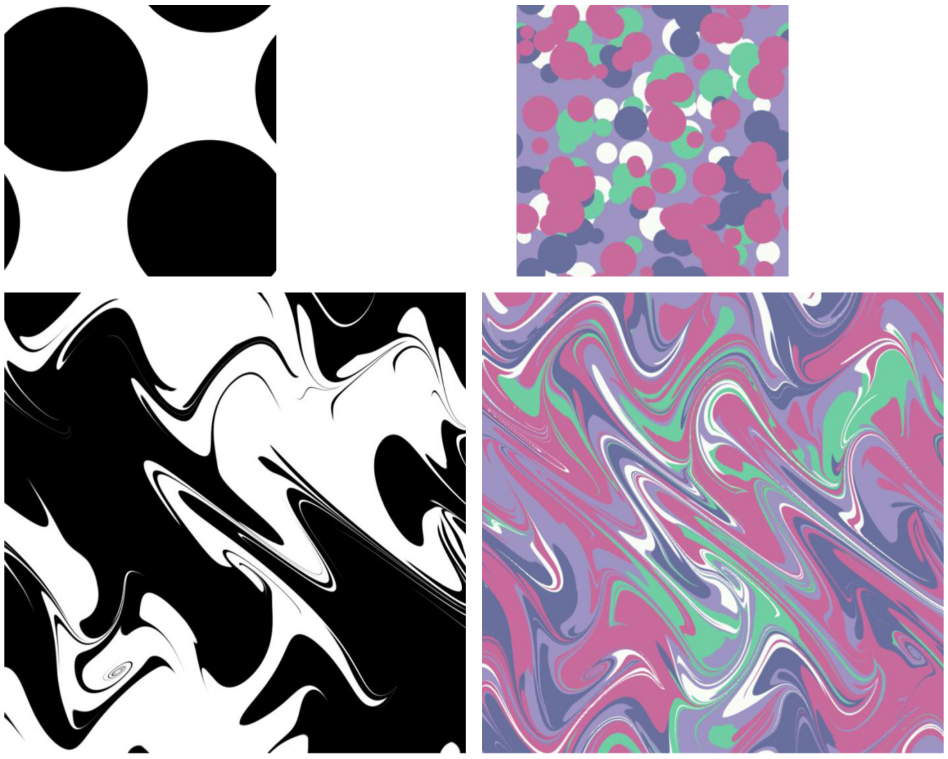


Fig. 10. Different patterns (bottom row) are generated based on different initial states (top row).

changes to the pattern functions, new designs can be derived by introducing variations to the initial design. Users can use the 'changing initial state' tool to replace the initial state of selected designs with a new one. Designers can adapt existing designs or ideas, from which to derive new designs. This is a common practice in design discipline. Fig. 10 shows the resulting patterns from two initial designs using the same set of pattern functions, in which both ripple and wavy functions are applied.

3.5.5. Lock and undo functions

In supporting the design, two functions named 'Lock' and 'Undo' are introduced. In recombination and swapping operations, all the individuals in the current population are allowed to join in the selection process and thus, all individuals would be possibly changed. The 'lock' function is provided for users to lock individual designs in the current population. The 'lock' mimics the elitist strategy (Back and Hoffmeister, 1991) in evolutionary computation. In avoiding the loss of good individuals in the evolution process, the elitist strategy preserves good genes for later generations. In our system, the locked design will not be changed in the design generation. 'Undo' function is also provided for users to testing out different rules of design generations.

3.6. Design evolution: user-in-the-loop evaluation and rule induction

The major difference between the proposed design theory inspired evolutionary design system and traditional evolutionary design system lies in the way that the evolution process is formulated. In traditional evolutionary design, genetic operators, sequential manner in order to produce a new generation of offspring. The application of these genetic operators is a stochastic process governed by numerical probabilities. In this paper, instead of simulating biological evolution, design operations, analogue to genetic operators, are developed to model different ways of creating new designs. As such, users create complex marbling textile patterns without understanding the underlying mathematical functions. These design operations simulate production stage of the design process. The trial-and-error design production helps users to accumulate and generalise design rules for identifying which operations can generate their preferred designs. If users are allowed to assess design after a sequence of operations, it is difficult to comprehend which operations create their preferred shapes. Therefore, users would prefer to assess the effectiveness of every single design operation so as to induce design rules by iterative selection.

3.6.1. *User evaluation and selection*

Designers have varied preferences and thus aesthetic assessment is often subjective. Due to the lack of objective aesthetic criteria, design evaluation is hard to be automated by computer. Subjective user assessment is used in our method. In some interactive applications, ranking is often employed by users to define fitness for every individual in the population, which facilitates fitness-dependent selection. If users are required to rank every single design in each cycle of the design process, it may increase the risk of user fatigue. Bush and Sayama (2011) suggested that human user can play a more central role in the control flow of evolutionary process beyond that of fitness evaluator. In our method, instead of ranking, user selects an arbitrary number of individual designs from the population based on his/her own preference.

3.6.2. *Design evolution*

After selection, users interactively apply operations of random alteration, function recombination, swapping or changing initial state to create new designs. The involvement of these design operations is not a sequential process of random operations. To conclude effective design rules, users are given more control over the design process: users are allowed to define the operations to use and alter parameters (e.g. probability of random alteration operation) throughout the design process. The workflow of the marbling textile design generation is shown in Fig. 3.

By the PDI framework, designers can induce rules for the preferred pattern creation. This is done in the trial-and-error design process. As the final marbling patterns are determined by the initial state and the sequence of operations. If designers want to reproduce similar patterns, they can apply changing initial state operators based on previously stored good designs. In the design operation, by experimenting different operations and testing out with 'undo' function, e.g. tuning the probability of mutation, or crossover, new patterns are generated; designers learn how the operations alter the patterns and gradually cumulative design rules to generate their preferred designs.

4. Textile properties

4.1. *Vector graphics*

Scalability is an important feature in textile design. Usually, it is necessary to scale the already defined small size pattern to a bigger size for printing onto fabrics. Pattern designs can be represented as raster images or vector images. A raster image is represented by a matrix of pixels and each pixel is a tiny coloured square. Therefore, a large number of colours are required to render a raster image for accurate reproduction of the original source artwork. In contrast to raster images, vector images are composed of geometrical primitives, such as points, lines, curves and polygons. Unlike raster images, vector images possess several advantages including: scalability, ease of editing, small file size, text searching of graphics and compatibility. Therefore, vector graphics are preferred and widely used in the textile and printing industry. The patterns produced by our design system are output to the SVG format, which is the W3C standard (SVG, 2006).

4.2. *Repeat*

Repeat is another basic principle of textile pattern design (Wilson, 2001). Textile designers need to have a good idea of how the designs can repeat and are going to look as a finished fabric. Therefore, the computer-generated patterns should support various repeat structures. The system formulation described in Section 3 generates single patterns. Because the pattern is created by the deformation of points in the initial state, some parts of the

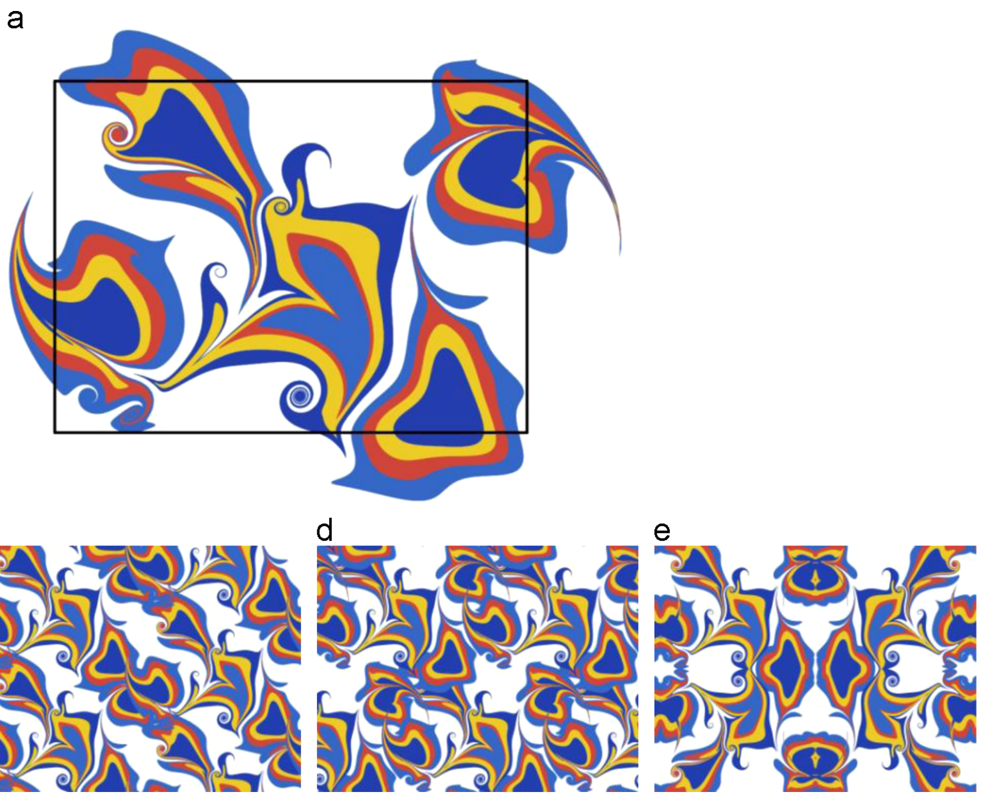


Fig. 11. The original design (a) and the resulting designs after repeating processes with (b) straight repeat structure, (c) half-drop repeat structure, (d) tile repeat structure and (e) mirrored repeat structure.

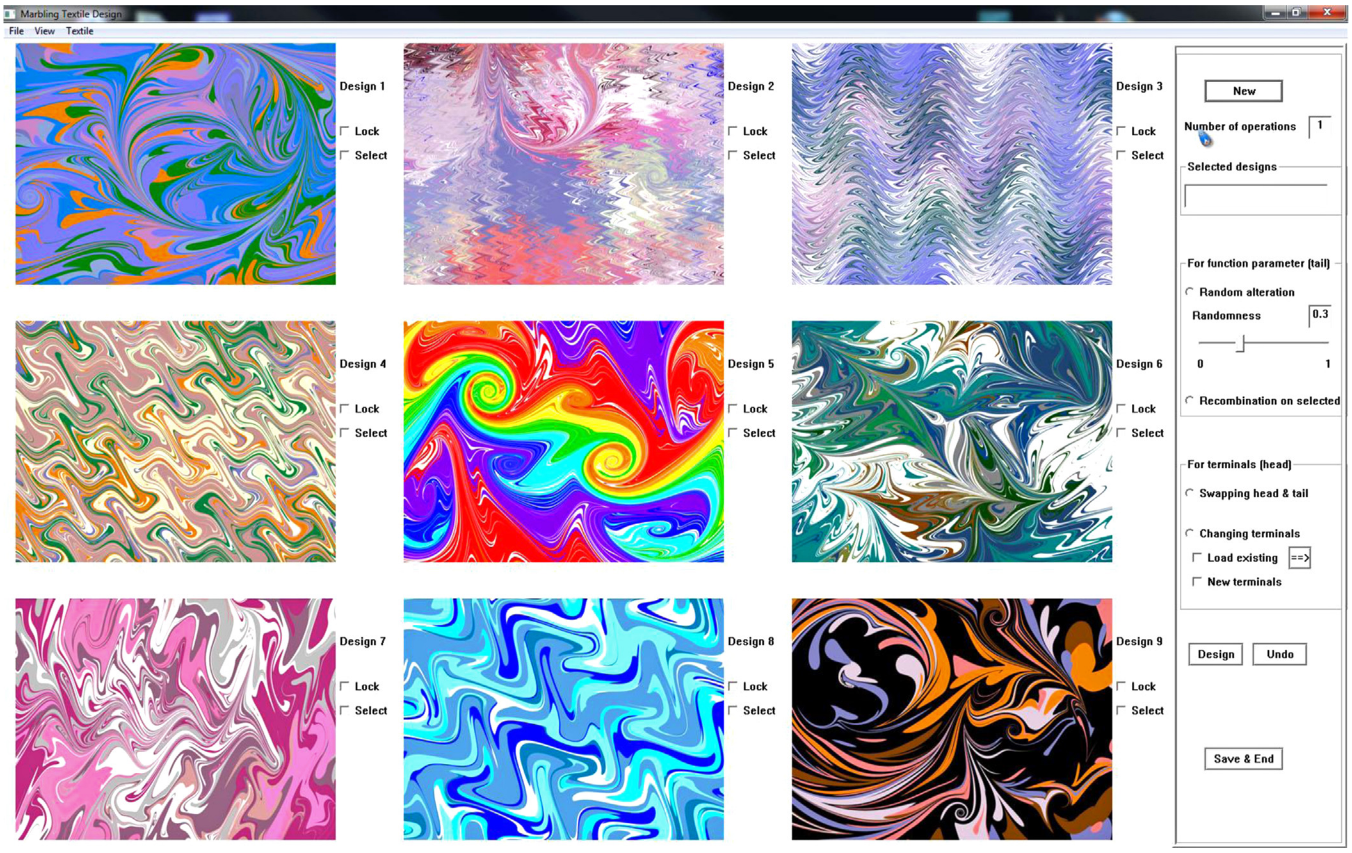


Fig. 12. Interface of the system.

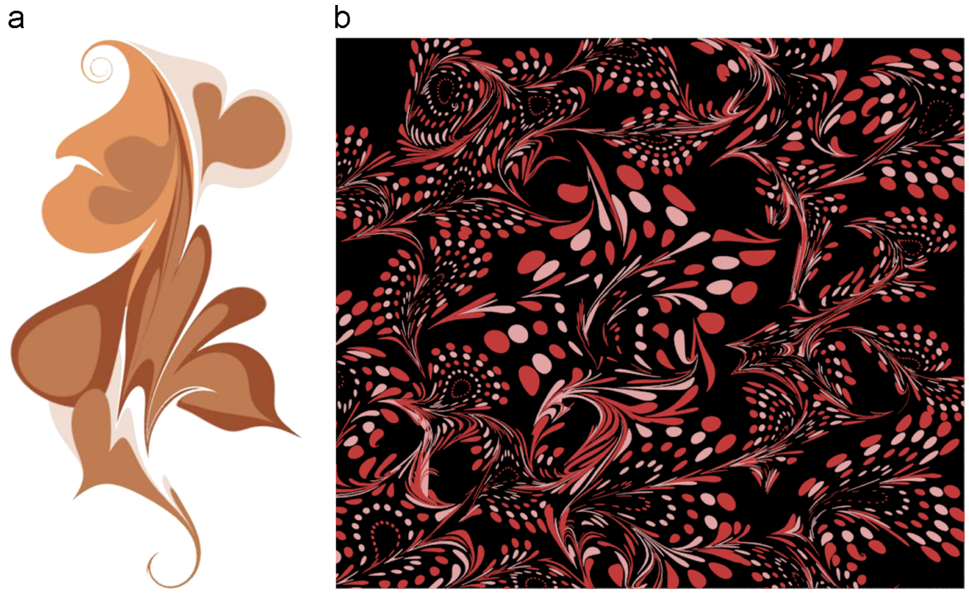


Fig. 13. Two marbling textile patterns generated by the system.

pattern would run out of the predefined cell (pattern size) in the deformation process, as shown in Fig. 11(a). To generate a periodic pattern, nine single patterns are placed on the canvas with locations defined by the repeat structures and the simulation is run again. The centre cell in the canvas is the periodic pattern. A few repeat structures that commonly used in the textile industry are incorporated in our system, thus repeatable patterns can be obtained automatically. Fig. 11(b–e) shows a few patterns generated after the repeat simulation.

5. System implementation and discussion

In our method, the initial state consists of vector-based elements including points and lines. A marbling pattern is formed by deforming points in the initial state with a set of pattern functions. This process is known as a front tracking method, which offers a precise representation of the pattern free from grid resolution. By applying a design operation to a set of selected designs, each individual design is recalculated separately according to its initial state and the set of pattern functions. This is a time-consuming process depending on the number of points in the initial state and the number of pattern functions involved. Fortunately, this process is highly parallel as all the points are deformed by the same pattern functions in each individual. GPUs are good at performing parallel mathematical operations. The Compute Unified Device Architecture (CUDA) framework (NVIDIA, 2011) is the dominating platform for general-purpose computation on graphics processing units (GPGPU). The calculation of pattern functions is a GPGPU problem, so CUDA is employed in our method. All the points of each individual are generated and deformed in parallel. For displaying and rendering the pattern, Direct3D 10 API (Blythe, 2006) is used, which supports the interoperability with CUDA. In this way, no extra data transfer is required between CUDA and Direct3D. The interface of the system is shown in Fig. 12.

Fig. 13 shows two marbling textile patterns generated by our design system. Fig. 13(a) is a simple marble pattern and Fig. 13 (b) is a complex one. The operations adopted and the required computation times are listed in Table 4. The number of points in the pattern is defined by the initial state. As described in Section 3, two design operations (changing initial state and swapping) are used to derive new designs by introducing variations to the initial state. Therefore, the number of points in the pattern will be changed once any of the two operations is applied. Two examples are Steps 2 and 3 in Table 4. The operations are calculated on the CPU to generate design representation for each new design. The generation and deformation of points is performed on the GPU. Most of the computation time is from the calculation of design operations on the CPU. The computation on the GPU costs very little time even with a large number of points, for example Step 3 for pattern of Fig. 13(b) in Table 4. The computation time in the table represents the time necessary for the current pattern. The total processing time is the time needed for the creation of all nine patterns in the population.

Traditionally, the manual process of creating a marbling textile pattern may take several hours. It is not a trivial task and the design process must be restarted from scratch when any mistake is made. In comparison, speed is an obvious advantage of the proposed system. Design alterations, which could take hours to do manually, can now be performed on computers with a few clicks. It is demonstrated that the system can generate creative marbling designs. Fig. 14 shows some results of marbling textile patterns used in home decoration and fashion design.

Commercial software Corel Painter supplies users with the function of generating marbling patterns interactively bases on image editing (Grossman, 2009). Corel painter supports two modes including low quality and high quality to create marbling textiles. However, it requires several seconds to render a complex and high-quality patterns because it is implemented on the CPU, hence it is not real time. As a result, our system which runs on GPU outperforms Corel painter in terms of speed. Moreover, the patterns produced by Corel Painter do not meet the textile industry requirements such as repeat and vector output. Fig. 15 are the results of tiling four single bouquet patterns in straight repeat structure. The pattern produced by Corel Painter has obvious artifacts in the upper region. On the contrary, the similar

Table 4
The design operations and computation time of the pattern in Fig. 13.

Pattern in Fig. 13(a)	1	2	3
Design operations	Recombination	Swapping	Random alteration
Number of points/ functions	4865/6	3900/6	3900/6
Computation time (s)	0.0055	0.0041	0.0030
Pattern in Fig. 13(b)	1	2	3
Design operations	Random alteration	Recombination	Changing initial state
Number of points/ functions	5335/24	5335/24	44910/24
Computation time (s)	0.0049	0.0064	0.0044

System configuration: 2.93 GHz Intel Core 2 Quad Q9550 CPU, NVIDIA GeForce GTX 275 GPU.

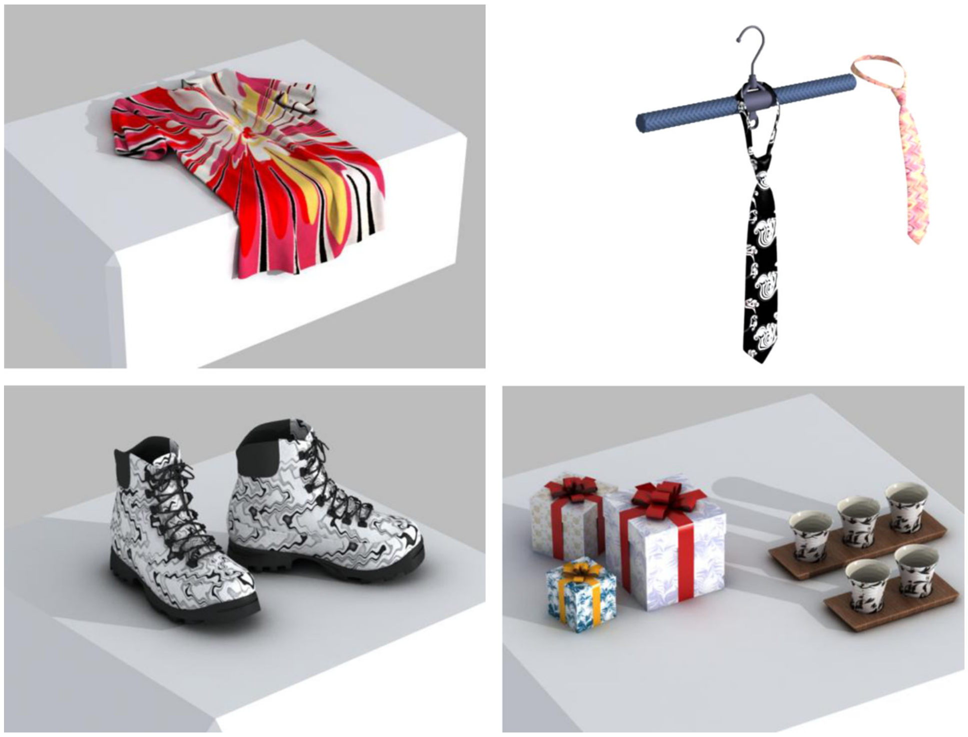


Fig. 14. Textile patterns used in home decoration and fashion design. The 3D objects were rendered with ray tracing in 3D Studio Max.

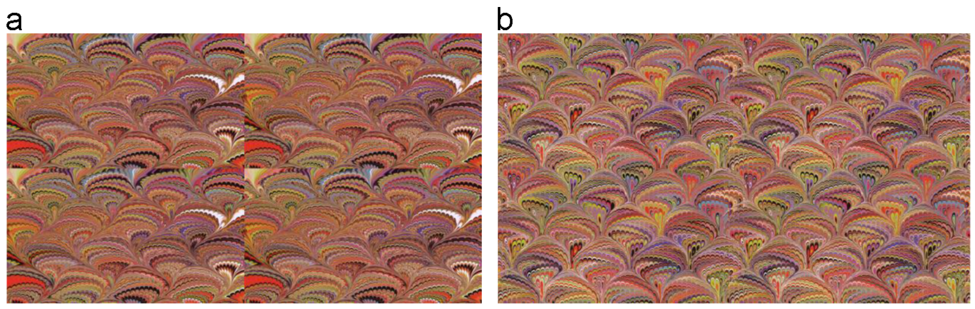


Fig. 15. Comparison between Corel Painter and our system: (a) and (b) are the results of tiling in straight-repeat structure four single bouquet patterns produced by Corel Painter and our system, respectively.

results produced by our method support various kinds of repeat structure and can be output as vector images.

6. Conclusion

In this paper, design methodology theories are integrated with evolutionary computation to construct an effective design system for marbling textiles. It is the first and only evolutionary marbling textile design environment currently available. The formulation of the design system, from the representation of individual designs, to the construction of design operations and design evolution, are all based on design research theories. It assists designers' PDI reasoning in the design process. The system encourages creativity in the design process and accelerates design generation. The resulting marble textile patterns fulfil the textile industry requirements of repeat and can output design in vector graphics. The system is being introduced to the textile industry, and very positive feedback has been received that the system accelerates design generation and creates novel designs. A Japanese textile company named MIYASHITA ORIMONO CO., LTD. (2011) has successfully adopted our system and applied the designed textiles to garments and accessories.

Acknowledgements

The authors would like to thank all anonymous reviewers for constructive comments. The work described in this paper was financially support by The Hong Kong Polytechnic University (Project code: A-PL18). Shufang Lu was supported by Zhejiang Provincial Natural Science Foundation of China (Grant No. LQ14F020004) and the Open Project Program of the State Key Lab of CAD&CG (Grant No. A1426), Zhejiang University. Xiaogang Jin was supported by the National Natural Science Foundation of China (Grant no. 61272298) and the Major Science and Technology Innovation Team (Grant no. 2010R50040).

Appendix A. Supplementary Information

Supplementary data associated with this article can be found in the online version at http://dx.doi.org/10.1016/j.engappai.2014.02.015.

References

Akin, O., Lin, C., 1995. Design protocol data and novel design decision. Des. Stud. 16, 211–236.
Akgun, B.T., 2004. The digital art of marbled paper. Leonardo 37 (1), 49–52.
Back, T., Hoffmeister, F., 1991. Extended selection mechanisms in genetic algorithms. In: Proceedings of the 4th International Conference on Genetic Algorithms. pp. 92–99.
Bergen, S., Ross, B., 2012. Automatic and interactive evolution of vector graphics images with genetic algorithms. Vis. Comput. 28, 35–45.
Bentley, P., 1999. An introduction to evolutionary design by computers. Evol. Des. Comput., 1–79
Bush, B., Sayama, H., 2011. Hyperinteractive evolutionary computation. IEEE Trans. Evol. Comput. 15, 424–433.
Blythe, D., 2006. The Direct3d 10 System. ACM SIGGRAPH, pp. 724–734.
Corne, D., Bentley, P., 2001. Creative Evolutionary Systems. Morgan Kaufmann
Cross, N., 1984. Developments in Design Methodology. John Wiley & Sons
Darke, J., 1979. The primary generator and the design process. Des. Stud. 1, 36–44.
den Heijer, E., Eiben, A., 2012. Evolving pop art using scalable vector graphics. Evol. Biol. Inspired Music, Sound, Art Des., 48–59
Finkelstein, L., Finkelstein, A., 1983. Review of design methodology. Phys. Sci., Meas. Instrum. Manag. Educ.-Rev., IEE Proc. A 130 (4), 213–222.
Gero, J.S., 2002. Artificial Intelligence in Design'02Springer
Grishanov, S., Siewe, F., Cassidy, T., 2011. An application of queuing theory to modelling of melange yarns. Part II: a method of estimating the fibre migration probabilities and a yarn structure simulation algorithm. Text. Res. J. 81 (8), 798–818.
Grishanov, S., Meshkov, V., Omelchenko, A., 2009. A topological study of textile structures. Part I: an introduction to topological methodsText. Res. J. 79 (8) 702–713.
Grossman, R., 2009. Digital Painting Fundamentals with Corel Painter 11. Course Technology PTR
Kicinger, R., Arciszewski, T., Jong, K., 2005. Evolutionary computation and structural design: a survey of the state-of-the-art. Comput. Struct. 83, 1943–1978.
Kroes, P., 2002. Design methodology and the nature of technical artefacts. Des. Stud. 23, 287–302.
Lawson, B., 2005. How Designers Think: Demystifying the Design Process. Architectural Press, Jordan Hill, GBR
Lewis, M., 2008. Evolutionary visual art and design. Art Artif. Evol., 3–37
Lu, S., Jaffer, A., Jin, X., Zhao, H., Mao, X., 2012. Mathematical marbling. IEEE Comput. Graph. Appl. 32, 26–35.
Lutton, E., 2006. Evolution of fractal shapes for artists and designers. Int. J. Artif. Intell. Tools 15, 651–672.
Machado, P., Cardoso, A., 2003. NEvAr–System Overview. In: Proceedings of Generative Art 2003. http://dx.doi.org/10.1.1.75.2020.
March, L., 1976. The Architecture of Form. Cambridge University Press
Maurer-Mathison, D., 1999. The Ultimate Marbling Handbook: A Guide to Basic and Advanced Techniques for Marbling Paper and Fabric. Watson-Guptill Crafts
McClintock, J., Yen, G., 2008. A two-tiered, agent based approach for autonomous, evolutionary texture generation. IEEE World Congress on Computational Intelligence, pp. 3220–3227.
MIYASHITA ORIMONO CO., LTD., 2011. ⟨http://miyashita-orimono.jp/index.html/⟩.
Moimoto, Y., Ono, K., 2010. Computer-generated tie-dyeing using a 3d diffusion graph. Adv. Vis. Comput., 707–718
Muni, D., Pal, N., Das, J., 2006. Texture generation for fashion design using genetic programming. In: Proceedings of the 9th International Conference on Control, Automation, Robotics and Vision (ICARCV), pp. 1940–1944.
NVIDIA, C., 2011. Nvidia Cuda Compute Unified Device Architecture Programming Guide. ⟨http://developer.nvidia.com/cuda⟩.
PBS, 2008. NOVA: Fractals – Hunting the Hidden Dimension. DVD.
Secretan, J., Beato, N., D'Ambrosio, D., Rodriguez, A., Campbell, A., Stanley, K., 2008. Picbreeder: collaborative interactive evolution of images. Leonardo 41, 98–99.
Shamey, R., Zhao, X., Wardman, R., 2005. Numerical simulation of dyebath and the influence of dispersion factor on dye transport. In: Proceedings of the 37th Conference on Winter Simulation, pp. 2395–2399.
Sims, K., 1991. Artificial evolution for computer graphics. Comput. Graph. (ACM) 25, 319–328.
SVG, 2006. ⟨http://www.w3.org/Graphics/SVG/⟩.
Thomas, J., Carroll, J., 1979. The psychological study of design. Des. Stud.1, 5–11.
Unemi, T., 2002. Sbart 2.4: an iec tool for creating 2d images, movies, and collage. Leonardo 35, 189–191.
Wiens, A., Ross, B., 2002. Gentropy: evolving 2d textures. Comput. Graph. 26, 75–88.
Wilson, J., 2001. Handbook of Textile Design: Principles, Processes and PracticeCRC Press
Wyvill, B., van Overveld, C., Carpendale, M., 2004. Rendering cracks in batik. NPAR, 61–149
Zeng, Y., Cheng, G., 1991. On the logic of design. Des. Stud.12, 137–141.