



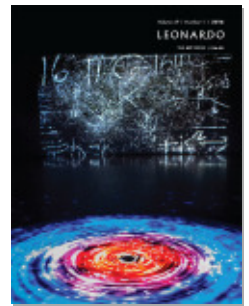
PROJECT MUSE®

Generation of Kandinsky Art

Kang Zhang, Jinhui Yu

Leonardo, Volume 49, Number 1, 2016, pp. 48-54 (Article)

Published by The MIT Press



➔ For additional information about this article
<https://muse.jhu.edu/article/608589>

Generation of Kandinsky Art

KANG ZHANG AND JINHUI YU

ABSTRACT

The authors present a programmed experiment to automatically generate art in the style of Kandinsky during his Bauhaus years. The program the authors developed analyzes the artist's paintings based on his art theories and the authors' own understanding and observations of his artworks. The authors describe the generation process in detail and share and discuss sample generated images styled according to four of Kandinsky's paintings. By pseudorandomizing various parameters, the program is able to make each styled image it generates unique. The authors' approach is highly scalable, limited only by the memory space set in the programming language Processing, which is used for the generation. Potential impacts of the authors' approach are also discussed.

Wassily Kandinsky is widely considered the founder of abstract painting and has influenced generations of abstract artists. His styles went through several stages of changes: from representational to nonrepresentational in his early career; from non-objective to geometric; and finally to what he called "biomorphic" abstraction. During his Bauhaus years (1922–1933), Kandinsky was interested in the composition and interaction of geometrical forms and published his well-known theoretical art book *Point and Line to Plane* [1].

The ability to model and generate abstract art in individualized yet well-known styles automatically has profound implications. First, it could inspire further study and insightful understanding of such styles. Second, various forms of generative art, such as static, animated, flashed displays and so on, would be made possible. Third, this capacity enables commercial applications of abstract art in forms such as logos, advertisements, packaging design and decoration at a negligible cost [2].

Computer graphics, interaction, display and printing technologies are now advanced enough to support automatic generation of sophisticated abstract paintings. One of

the most successful computer-generated art forms has been fractal art [3]. According to coauthor Zhang and colleagues' recently proposed four-level system of classification that rates processes of computer use to generate abstract art in terms of computational power utilized [4], the creation of fractal art is considered a Level 2 process. This classification is based on a measurement of the computational complexity required to create a computer-generated abstract work—i.e. to what extent machine intelligence is utilized in generating the paintings—rather than by the visual complexity, as in Taylor's analysis [5]. Limited attempts have been made at Level 3. An example of a Level 3 process would be using heuristic rules and patterns to encode an artist's style and then generating artist-specific styles of paintings. (This is summarized below in "Related Work"). Only after further investigation and experimentation of sufficient width and depth would reaching Level 4 become a possibility. An example of a Level 4 process would be a program that uses intelligent rules and reasoning capabilities to automatically generate sophisticated aesthetic designs.

In this paper we present our attempt to develop a Level 3 process that can eventually lead to Level 4 sophistication. We report on a scalable and parameterized approach to automatic generation of art in the style of Kandinsky's abstract paintings using the Processing programming language [6]. Processing is built on top of the Java language and designed for artists and designers with the purpose of teaching the fundamentals of computer programming in a visual context. By using Processing, we could easily program basic graphics algorithms and simple rendering techniques for generative art [7].

STYLE ANALYSIS

After reading Kandinsky's two books on art theory [1,8] and searching for images on the Web and published in books, we analyzed the following set of paintings created during Kandinsky's Bauhaus years:

- *Black and Violet* (1923)
- *Composition VIII* (1923)
- *On White II* (1923)

Kang Zhang (educator), Department of Computer Science, University of Texas at Dallas, Richardson, TX 75083-0688, U.S.A. Email: <kzhang@utdallas.edu>.

Jinhui Yu (educator), State Key Laboratory of CAD & CG, Zhejiang University, Hangzhou, 310027, China. Email: <jhyu@cad.zju.edu.cn>.

See <www.mitpressjournals.org/toc/leon/49/1> for supplemental files associated with this issue.

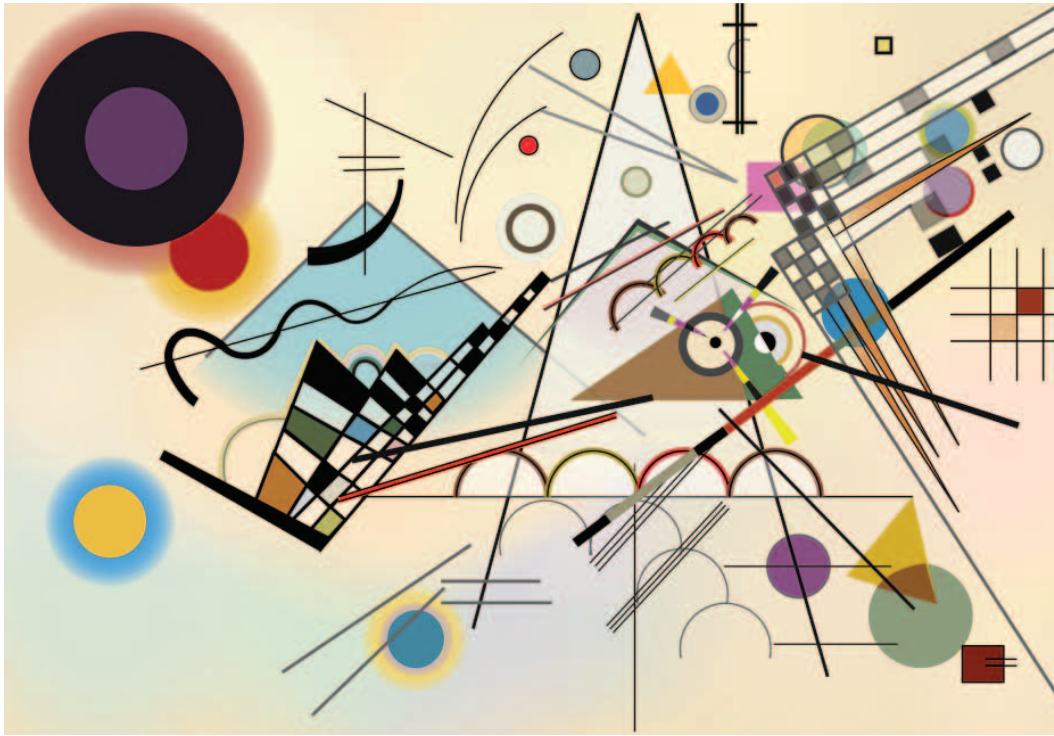


Fig. 1.
Kang Zhang,
program-generated
Composition VIII.
© Dept. of
Computer Science,
University of Texas
at Dallas)

- *Transverse Line* (1923)
- *Still Tension* (1924)
- *Black Circles* (1924)
- *Yellow Red Blue* (1925)
- *Merry Structure* (1926)
- *Several Circles* (1926)
- *Hard But Soft* (1927)
- *Thirteen Rectangles* (1930)

The most representative work during this period is *Composition VIII* [9]. Figure 1 depicts a computer-generated version of this painting created using Processing. Throughout this paper, we refer to the structures composed of geometrical forms that occur frequently in Kandinsky's paintings as *styled patterns*. The process of using a computer to generate Kandinsky-style paintings such as the computer-generated version of *Composition VIII* begins with the observation and analysis of the original painting, followed by the construction of individually styled and aesthetic patterns. The generative program finally assembles the patterns in a pseudorandom fashion into a structure until harmony is achieved.

Kandinsky interpreted the three primary forms that occur frequently in his paintings as being constructed of angular lines [8]:

- acute angles lead to a triangle and have the yellow color within
- right angles lead to a square that is of a plan-like nature and parallel to the red color
- the obtuse angle is passive and related to the light blue tone.

According to Kandinsky, the effect of yellow is emphasized well when combined with a sharp form, such as a triangle. In contrast, the effect of a deeper color (e.g. blue) is reinforced by a rounded form, such as a circle [9].

Establishing theoretical relationships between lines, planes and colors, he also proposed three primary contrasting pairs of elements, as illustrated in Fig. 2:

1. Straight line vs. curved line—one or more straight lines intersecting with a curved line, and individual lines and curves. Some lines are filled with segmented colors.
2. Triangle vs. circle—one or more triangles overlapping on a circle or concentric circles, one or more half circles in a row, and individual filled triangles and circles. Some triangles are filled with segmented colors.
3. Yellow vs. blue—circles filled with yellow or blue, having an outer circle in blue or yellow gradually fading outward. Straight lines intersect at the right angles to form one or more filled rectangles.

Most of Kandinsky's paintings of the Bauhaus years are composed of various forms of these three pairs of elements







<p>Straight line</p> 	<p>Curved line</p> 
<p>Triangle</p> 	<p>Circle</p> 
<p>Yellow</p> 	<p>Blue</p> 

Fig. 2. Kang Zhang, contrasting pairs of elements.
© Dept. of Computer Science, University of Texas at Dallas)

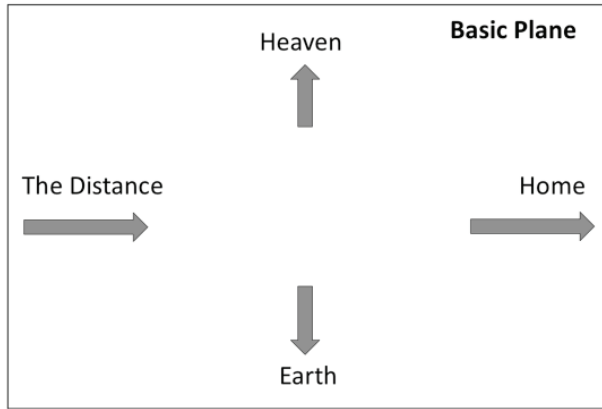


Fig. 3. Kang Zhang, four boundaries of the basic plane (BP).
 (© Dept. of Computer Science, University of Texas at Dallas)

and the interaction between the pairs. Forms “grow outwards from many points to envelop the whole (in a manner similar to the way new skin grows over a wound)” [10] rather than being developed from a single center.

Using the artist’s own terminology, the canvas on which abstract paintings are drawn is considered a *basic plane* (BP). The four sides of the BP and the visual representations of their tensions of movement and “literary” interpretations are illustrated in Fig. 3.

MODELING FOUR PAINTINGS

To derive a detailed encoding and generation of Kandinsky’s style, we selected four paintings from his Bauhaus period: *Composition VIII* (1923), *Black and Violet* (1923), *On White II* (1923) and *Several Circles* (1926).

The Generation Flow

With the exception of *Composition VIII*, which we generated by hard-coding the colors, sizes and positions of all the objects according to the original painting (see Fig. 1), all images (such as Fig. 4) were fully automatically generated by randomizing all attributes (e.g. size, color, location, etc.) following the process we describe in the remaining part of this section. As depicted in Fig. 5, the generation process consists of the following major steps:

1. We select a painting to be modeled.
2. The program carries out the setup, during which it prepares all the necessary variables or parameters and their sizes, ranges and random variation ranges.
3. The program generates the background colors based on certain characteristics of the painting.
4. A selection of styled patterns are pre-coded in Processing.
5. The program renders styled patterns based on the parameters randomly set in Step 2. This is the key step in the generation process.
6. When there are no more patterns left to render, the program saves the entire image rendered on the digital canvas.

The following subsections describe these steps in more detail.

Style Encoding

Based on the aforementioned analysis and our understanding of Kandinsky’s art theory, we have made the following observations from the four selected paintings:

- Semi-circles are most often drawn with their curved lines pointing toward the top of the canvas.
- Equilateral triangles are most often drawn with their points facing to the top of the canvas, and with only their two sides and an open bottom.
- Thin horizontal and vertical lines set the base and are intersected by small, angled lines.
- When forms overlap, the overlapping is reflected through the use of different colors.
- Prominent full circles—large with contrasting colors—are often surrounded by gradual changing shades of contrasting colors, most often yellow and blue.
- Grid-like forms are filled with interleaving colors. Some are formed by parallel lines and others by

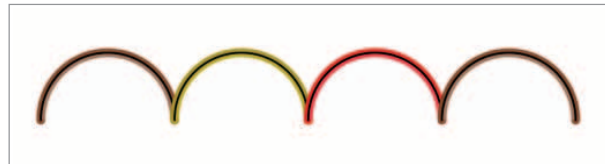


Fig. 4. An example of styled patterns.
 (© Dept. of Computer Science, University of Texas at Dallas)

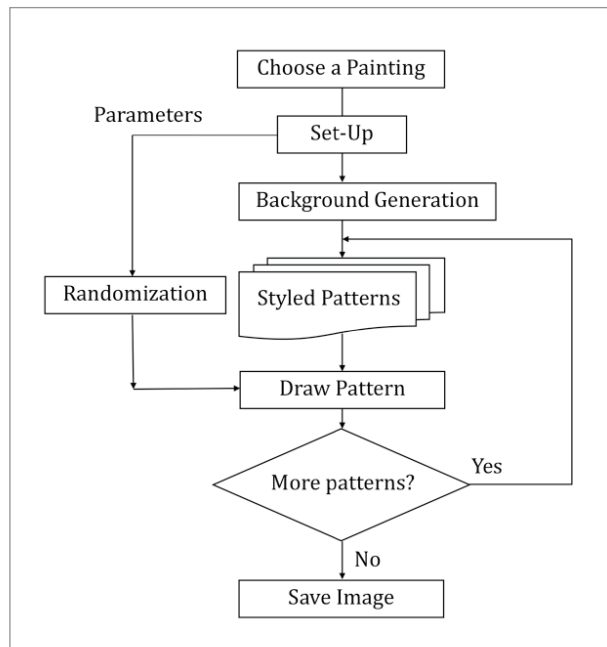


Fig. 5. Kang Zhang, steps of automatic generation.
 (© Dept. of Computer Science, University of Texas at Dallas)

angled lines; some meet at the end, forming sharp triangles.

- Curved lines that do not form circles are rare but, when used, are placed in dominant positions with colors contrasting with the background.
- Dark boundaries are filled with light colors. Black is occasionally used to fill out forms for additional emphasis.
- Red and black are used together, creating a highlighting effect.
- The background of the canvas usually consists of non-uniform colors.
- Typical neighboring colors include green and blue; red, yellow and black; and red and blue.

We could make more detailed observations, but we believe that the list above is sufficient for us to derive concrete encoding to model the four selected paintings. Altogether, we have encoded 32 styled patterns derived from these observations.

Figure 4 shows an example of a styled pattern. The pattern consists of a number of half circles (also called *arcs*) along a line, each drawn in a different color. The program segment implementing the pattern is shown in Fig. 6. The parameters that may be set by the user include the number of arcs in the row (`numOfArcs`), the position (`centerX`, `centerY`) and rotation angle (`rotAngle`) of the row, the color to be filled (`fillColor`), and the level of transparency of the filled color (`transparency`). The arcs can also be set to gradually change in size from large (`largeRadius`) to small (`smallRadius`). In this implementation, the colors of the arcs are determined by the program rather than the user. The program could easily be changed to let the user set the arc colors and then let the program decide any of the other parameters if needed.

Setup and Background

The setup step includes the definition of canvas size, color ranges, variables and their possible ranges, and the setting of their initial values. This step also generates a pseudorandom set of coordinates on the canvas, a pseudorandom set of colors, and a pseudorandom range for each parameter applied to a styled pattern.

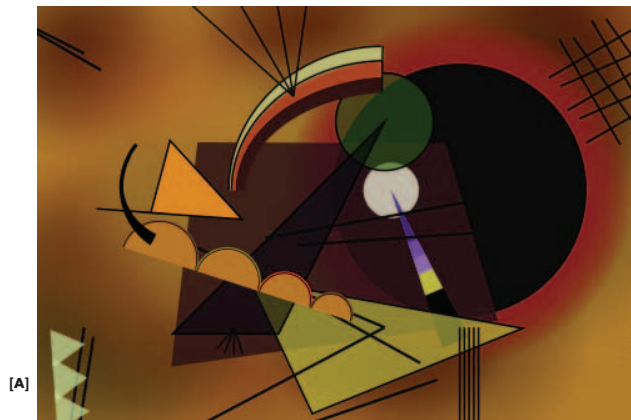
We divide the entire display area (canvas) into two regions: the central region and the peripheral region surrounding the central region. This simple division serves two purposes. First, some objects, such as groups of lines, are typically located in the peripheral region, while others, such as large triangles and quadrilateral shapes, are located in the central region. Second, some of Kandinsky's paintings, such as *Yellow, Red, Blue* (1925), have slightly lighter background colors in the central region than in the peripheral region. A set of *X* and *Y* coordinates within a predefined margin are randomly selected from both regions. Random sets of other parameters, such as length and radius, are also generated within predefined ranges during the setup phase.

Typically the background color in Kandinsky's paintings is non-uniform yet dominated by one color shade. The background color shade at a specific location is sometimes influenced by the foreground object shape at that location. Based on this observation, we first fill the canvas background with a uniform color randomly determined within a predefined range according to the style of the painting. Gradient color shades within a given range of variations from the uniform color are then applied to different locations on the canvas. The peripheral region may be filled with slightly darker shades, as was the case for the paintings *Yellow, Red, Blue* (1925) and *Black and Violet* (1923).

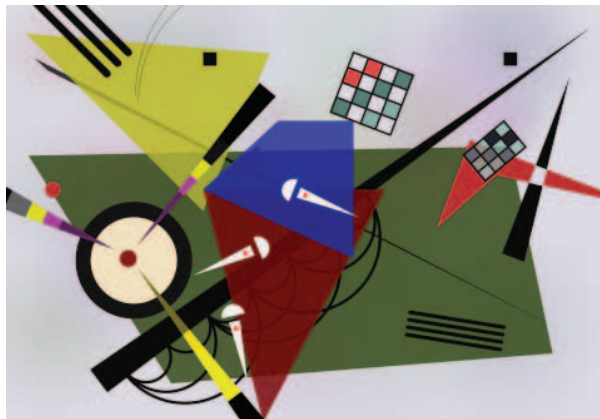
Schlegel anticipated the aesthetic appeal of patterns and

```
void arcs(int centerX, int centerY, int numOfArcs, int largeRadius, int smallRadius, int fillColor, int transparency, int rotAngle){
    int inc = (largeRadius-smallRadius)/numOfArcs, offsetX=0, r, g, b;
    translate(centerX, centerY);
    rotate(radians(rotAngle));
    fill(fillColor, transparency);
    for (int i=0; i<numOfArcs; i++) {
        r=(i%3)*40+130; g=(i+1)%3*75; b=(i+2)%3*25;
        strokeWidth(4);
        stroke(r,g,b);
        arc(offsetX, 0, largeRadius-i*inc, largeRadius-i*inc, PI, TWO_PI);
        strokeWidth(1);
        stroke(0);
        arc(offsetX, 0, largeRadius-i*inc, largeRadius-i*inc, PI, TWO_PI);
        offsetX += largeRadius-i*inc;
        if (inc>0) offsetX -= 4;
    }
    strokeWidth(0);
    rotate(radians(-rotAngle));
    translate(-centerX, -centerY);
}
```

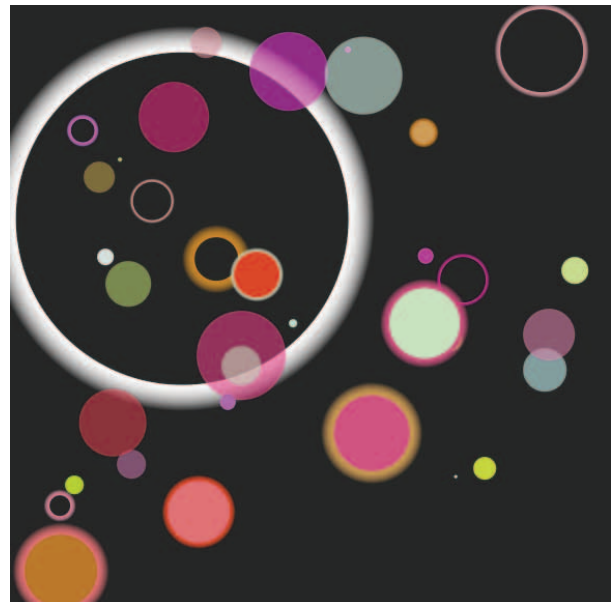
Fig. 6. The program segment implementing the pattern. © Dept. of Computer Science, University of Texas at Dallas



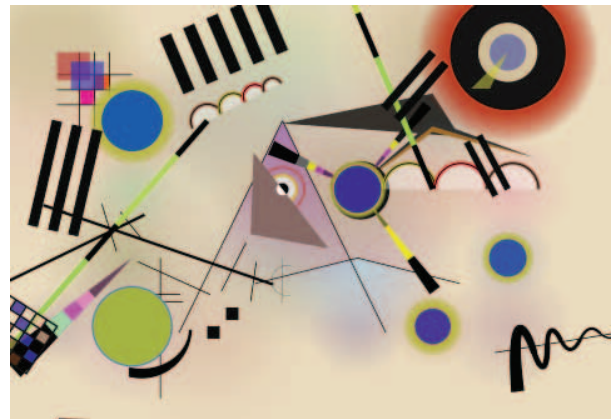
[A]



[B]



[C]



[D]

Fig. 7. Samples of automatically generated modelings of Kandinsky's paintings (a) *Black and Violet*, (b) *On White II*, (c) *Several Circles* and (d) *Composition VIII*. (© Dept. of Computer Science, University of Texas at Dallas)

randomness in paintings such as Kandinsky's as "structured, artistic chaos" [11]. We therefore used a pseudorandom process to generate the aforementioned parameters, each within a range obtained through experiments and selected through our observations.

Results

We now use *Black and Violet* as an example to explain the generation process. The program first divides the canvas into 50 smaller areas. The background is first filled with the color set by randomly generating the color code of (R, G, B) within the ranges [150, 210], [120, 130] and [10, 30]. One may note that the central region is lighter than the peripheral region in the background of *Black and Violet*. The four sides of the background are colored by subtracting 30 from each of the R, G and B values.

Each small area in the background is drawn with a circle that is randomly sized between 120 and 250 pixels and filled with a color of a small variation from the above R, G and B values, gradually fading outward. The area size and circle size are chosen so that the circles overlap with each other a number of times sufficient to yield smooth yet non-uniform color variations. The iterative process of fine-tuning all these parameters and random ranges, and then comparing the re-

sults with the original painting, is indeed time consuming and tedious. This process is necessary, however, to ensure that the program-generated image is close enough to the original painting.

The generation program then selects 0–5 variations of each styled pattern. The variations are realized by parameterization and randomization. For example, some patterns, such as the large black curve (top-left in Figure 7[a]), implemented by connecting eight cubic Bézier curves), are set with the rotation angles and sizes corresponding to their positions in respect to the four boundaries. Other patterns, such as open-bottom equilateral triangles (gradient light green and gradient light blue patterns in the central area, such as in *Composition VIII*), are positioned upward and appear to be growing out of the "earth"-like mountains.

The entire generated image is finally saved as a .tiff file; its size is determined by the canvas size set in the setup step. This generation program was used to generate Figs 7(b)–7(d).

POTENTIAL IMPACTS

Although the patterns encoded are all styled based on Kandinsky's paintings of his Bauhaus years, our approach is equally applicable to the generation of other styles of paintings. It demonstrates the possible impact of computer-

simulated human aesthetic production. By adding more details and styled patterns, and introducing aesthetic rules, we could cause programmed generative processes to mimic or reproduce many types of art. Our approach also has great potential in its broader applications. For example, Kandinsky's paintings expose strong inner dynamics of geometrical objects. The ability to automatically generate his style of abstract painting could enable abstract representation and animation of dynamic systems such as music and human movement [12,13].

This approach could also be extended into the graphic design field by encoding design rules and generating desired designs. Such an approach would assist and extend designers' creativity through machine intelligence and automation, leading to the generation of more sophisticated aesthetic forms at Level 4 [4]. The analysis of Kandinsky's aesthetics would certainly benefit the future design of information visualization [14].

Our generation program is highly scalable, not only with the size of the generated image (using Processing's "scale" statement) but also with the number and size of any individual styled pattern determined by a set of parameters. The only limitation to the image size is the machine's memory space. Using an Apple iMac with 4 GB memory and an Intel Core 2 Duo (1 processor) at 3.06 GHz, (the Processing language required 1 GB of memory), we have been able to generate images of 10880×7514 pixels. Such an image requires a storage space of 245.3 MB. The generation process for an image of this size takes about 4–5 minutes, but the program has not been optimized for either speed or memory.

RELATED WORK

Artificial intelligence researchers have long been making creative computers for art generation. One example is AARON, developed by Harold Cohen [15]. On the other hand, computer graphics researchers and digital artists have used the computer to generate abstract art works based on fractals [3,16]. There has, however, been limited effort toward automatic generation of abstract paintings based on specific artists' styles. The notable example is that of Taylor [17,18], who used fractals to model and analyze Jackson Pollock's drip style of paintings and generated remarkable results. There have also been a few attempts at automatically generating

Mondrian-style abstract paintings [19], which can be traced back to the pioneering work of Noll [20].

Kirsch and Kirsch [21] first proposed to model Miro's style by storing typical Miro shapes in a database and then using them to manually analyze the target composition. They did not encode the analysis in any grammar or program but offered some preliminary discussion. Barnett and Barnett [11] analyzed Kandinsky's compositions, while Price [22] applied Kandinsky's aesthetics to Java programming.

Zhang and previous coauthors [4] made an initial attempt to automatically generate Kandinsky-style abstract paintings, which included very few styled patterns and generated uniform backgrounds and no sophisticated combination of geometric objects. We have since conducted an in-depth and comprehensive experiment on generating several styles of Kandinsky's paintings. This paper is a reworked and extended version of a SIGGRAPH Asia 2013 art paper [23] and includes a more detailed and elaborate discussion, as well as more interesting images generated.

CONCLUSION

This paper has introduced our recent attempt at automatic generation of Kandinsky-style abstract paintings. The approach used is scalable and generic in generating pseudo-random images resembling the artist's style. As with any automated approach, our approach has its limitations in generating detailed artistic styles that match the artist's interpretation. Automatically generating nongeometrical but nonrepresentational images resembling the artist's style is much more challenging but possible and requires much more detailed modeling and encoding. More styled patterns could be added, and the styled encoding can also be further fine-tuned, to generate patterns with additional details of the artist's style.

We believe that there should be many opportunities to further refine and extend the work and apply it to commercial and design arenas. An immediate future extension would be to include other artistic styles and model Kandinsky's paintings from his other artistic periods (such as his work with non-geometric shapes) and other abstract artists' works. Since the original submission of this paper, we have applied this technique to automatically generate Jackson Pollock's drip style [24] and Kazimir Malevich's geometric abstract style [25].

Acknowledgments

We thank the anonymous reviewers for their insightful and constructive comments, which helped us significantly improve our paper. This work is partially supported by the National Science Foundation (NSF) of China (No. 61379069).

References

- 1 W. Kandinsky, *Point and Line to Plane* (originally published 1926), translated by H. Dearsynte and H. Rebay, Dover Publications Inc., New York, 1979.
- 2 J. Fogarty, J. Forlizzi and S.E. Hudson, "Aesthetic Information Collages: Generating Decorative Displays That Contain Information,"

Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology (UIST'01), ACM Press, pp. 141–150.

- 3 M.F. Barnsley, *Fractals Everywhere*, Morgan Kaufmann, 1993.
- 4 K. Zhang, S. Harrell and X. Ji, "Computational Aesthetics—On Complexity of Computer-Generated Paintings," *Leonardo*, Vol. 45, No. 3 (2012) pp. 243–248.
- 5 R.P. Taylor, "Pollock, Mondrian and Nature: Recent Scientific Investigation," *Chaos and Complexity Letters* 1, 2004, 29.
- 6 C. Reas and B. Fry, *Processing: A Programming Handbook for Visual Designers and Artists*, 2007, MIT Press.
- 7 M. Pearson, *Generative Art: A Practical Guide Using Processing*, Manning Publications, July 2011.

- 8 W. Kandinsky, *Concerning the Spiritual in Art* (originally published in 1914), translated with an introduction by M.T.H. Sadler, Dover Publications Inc., New York, 1977.
9. See W. Kandinsky, *Composición VIII* (1923), Solomon R. Guggenheim Museum, New York.
- 10 V.E. Barnett and P.H. Barnett, "The Originality of Kandinsky's Compositions," *The Visual Computer*, Vol. 5, No. 4, July 1989, pp. 203–213.
- 11 G.R. Thompson, *The Art of Authorial Presence: Hawthorne's Provincial Tales*, Duke University Press, 1993, p. 21.
- 12 A. Ione, "Klee and Kandinsky: Polyphonic Painting, Chromatic Chords and Synesthesia," *Journal of Consciousness Studies*, Vol. 11, No. 3–4, 2004, pp. 148–58.
- 13 S.S. Snibbe and G. Levin, "Interactive Dynamic Abstraction," *Proceedings of the 1st International Symposium on Non-photorealistic Animation and Rendering (NPAR'00)*, Annecy, France, 2000, pp. 21–29.
14. K. Zhang, "From Abstract Painting to Information Visualization," *IEEE Computer Graphics and Applications*, May/June 2007, pp. 12–16.
- 15 See <www.kurzweilcyberart.com/>.
- 16 L. Ammeraal and K. Zhang, *Computer Graphics for Java Programmers*, Second Edition, John-Wiley & Sons, 2007.
- 17 R.P. Taylor, "Order in Pollock's Chaos," *Scientific American*, December 2002, pp. 116–121.
- 18 R.P. Taylor, A.P. Micolich and D. Jones, "The Construction of Pollock's Fractal Drip Paintings," *Leonardo* Vol. 35, No. 2 (2002) p. 203.
- 19 M. Fogleman, "Procedurally Generating Images in the Style of Piet Mondrian," <<http://fogleman.tumblr.com/post/11959143268/procedurally-generating-images-in-the-style-of-piet>>, 2011.
- 20 A.M. Noll, "Human or Machine: A Subjective Comparison of Piet Mondrian's 'Composition with Lines' and a Computer-Generated Picture," *The Psychological Record*, Vol. 16. No. 1, January 1966, pp. 1–10.
- 21 J.L. Kirsch and R.A. Kirsch, "The Anatomy of Painting Style: Description with Computer Rules," *Leonardo*, Vol. 21, No. 4 (1988) pp. 437–444.
- 22 C.B. Price, "From Kandinsky to Java (The Use of 20th Century Abstract Art in Learning Programming)," *ITALICS*, Vol. 6, No. 4, October 2007, pp. 35–50.
- 23 K. Zhang and J.H. Yu, "Generating Abstract Paintings in Kandinsky Style," *Proceedings of the 6th ACM SIGGRAPH Asia (Art Papers)*, Hong Kong, China, 19–22 November 2013, ACM Press.
- 24 Y. Zheng, X.C. Nie, Z.P. Meng, W. Feng and K. Zhang, "Layered Modeling and Generation of Pollock's Drip Style, The Visual Computer" (May 2014). Available online: <www.utdallas.edu/~kzhang/Publications/VisComp2014.pdf>.
- 25 W.Y. Tao, Y.X. Liu, and K. Zhang, "Automatically Generating Abstract Paintings in Malevich Style," in *Proceedings of the 13th IEEE/ACIS International Conference on Computer and Information Science (ICIS'2014)*, Taiyuan, China, 4–6 June 2014, pp. 201–205.

Manuscript received 30 December 2013.

KANG ZHANG is a professor of computer science and a professor of arts and technology at the University of Texas at Dallas. His research interests include information visualization, visual languages and aesthetic computing.

JINHUI YU is a professor of computer science at the State Key Lab of CAD & CG, Zhejiang University, China. His research interests include image-based modeling, nonphotorealistic rendering, computer animation and computer art.