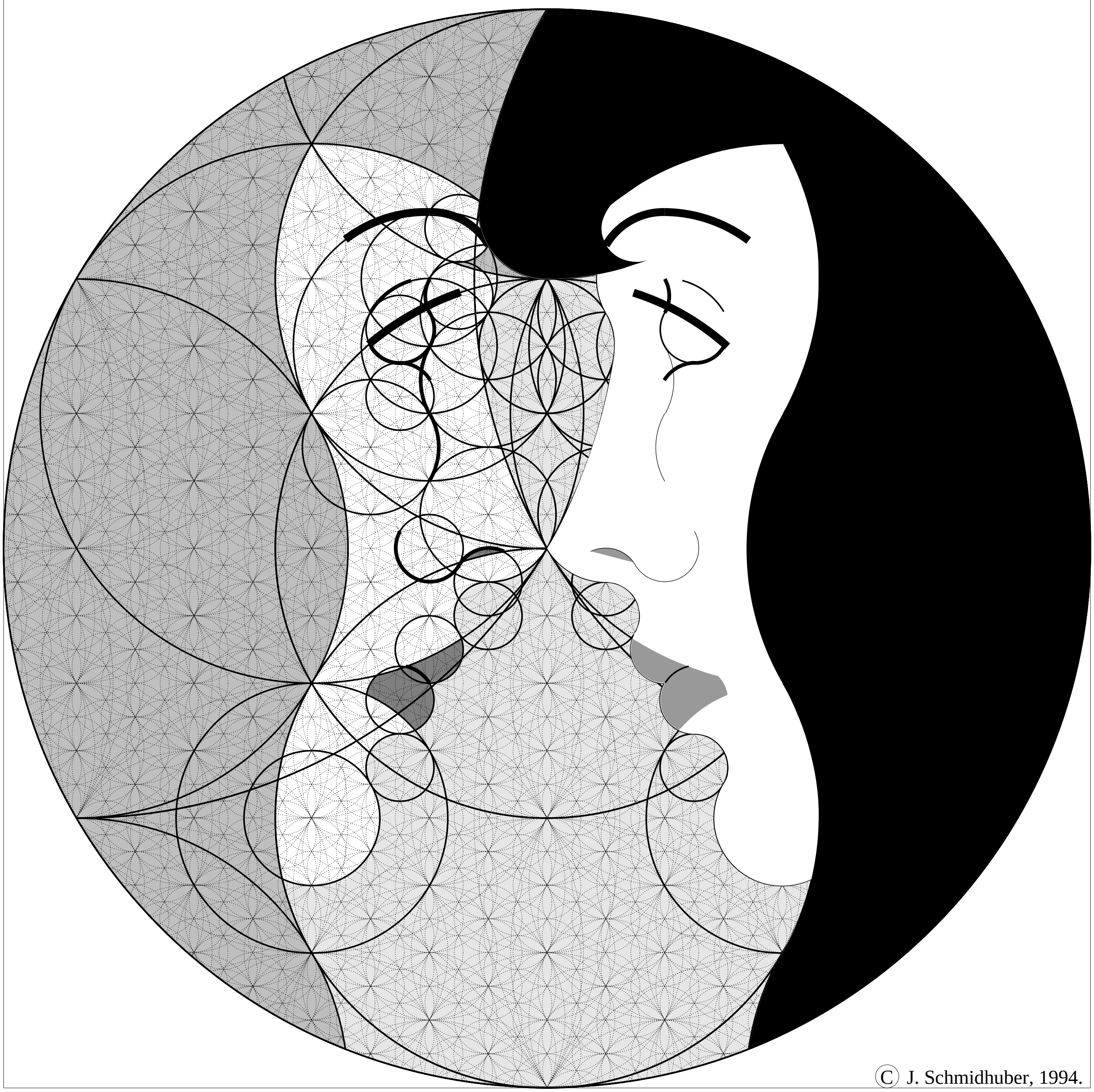
LOW-COMPLEXITY ART

JÜRGEN SCHMIDHUBER



LOW-COMPLEXITY ART

Jürgen Schmidhuber
IDSIA, Corso Elvezia 36
6900 Lugano, Switzerland
juergen@idsia.ch

Leonardo, Journal of the International Society for the Arts,
Sciences, and Technology, vol. 30:2, p. 97–103, MIT Press, 1997.
Note: this version's layout is different from Leonardo's original layout.

Abstract

Many artists when representing an object try to convey its "essence." In an attempt to formalize certain aspects of depicting the essence of objects, the author proposes an art form called low-complexity art. It may be viewed as the computer-age equivalent of minimal art. Its goals are based on concepts from algorithmic information theory. A low-complexity artwork can be specified by a computer algorithm and should comply with two properties: (1) the drawing should "look right," and (2) the Kolmogorov complexity of the drawing should be small (the algorithm should be short) and a typical observer should be able to see this. Examples of low-complexity art are given in the form of algorithmically simple cartoons of various objects. Attempts are made to relate the formalism of the theory of minimum description length to informal notions such as "good artistic style" and "beauty."

1 Introduction

In their introduction to Kolmogorov complexity, Li and Vitányi write:

> We are to admit no more causes of natural things (as we are told by Newton) than such as are both true and sufficient to explain their appearances. This central theme is basic to the pursuit of science, and goes back to the principle known as Occam's razor: "if presented with a choice between indifferent alternatives, then one ought to select the simplest one" [1]. Unconsciously or explicitly, informal application of this principle in science and mathematics abound.

The principle of Occam's razor is not only relevant to science and mathematics, but to fine arts as well. Some artists consciously prefer "simple" art by claiming: "art is the art of omission." Furthermore, many famous works of art were either consciously or unconsciously designed to exhibit regularities that intuitively simplify them. For instance, every stylistic repetition and every symmetry in a painting allows one part of the painting to be described in terms of its other parts. Intuitively, redundancy of this kind helps to shorten the length of the description of the whole painting, thus making it simple in a certain sense.

It is possible to formalize the intuitive notions of "simplicity" and "complexity." Appropriate mathematical tools are provided by the theory of Kolmogorov complexity (or algorithmic complexity) [2]. The Kolmogorov complexity of some computable object is essentially the length (measured in number of bits) of the shortest algorithm that can be used to compute it. The shorter the algorithm, the simpler the object [3].

In this paper, I use basic concepts from the theory of algorithmic complexity to serve as ingredients for a novel form of simple art that I call "low-complexity art." Although the focus in this article will be on black-and-white cartoons, the basic ideas are not limited to this type of application.

2 Basic Concepts

The following are a few basic concepts of the theory of Kolmogorov complexity (see the Appendix for formal details):

Compiler theorem. Informally, this theorem says that any program for a given computer can be compiled into an equivalent program for a given universal computer by a compiler program whose length does not depend on the programs it compiles.

Kolmogorov complexity. The Kolmogorov complexity of a computable object is the length of the shortest program that computes it on a universal computer and halts.

Invariance theorem. Essentially, the invariance theorem says that the Kolmogorov complexity of some object does not depend of the particular computer used, leaving aside an additive, machine-specific, object-independent constant. This objectivity is due to the compiler theorem.

Noncomputability of Kolmogorov complexity. It can be shown that there is no single algorithm that can generate the shortest program for computing arbitrary given data on a given computer [4].

3 Low-Complexity Art

Suppose an artist's task is to produce a drawing that obeys a set of (possibly informal) predetermined specifications. The goal of low-complexity art is to represent the depicted object's essence by achieving two conflicting goals simultaneously:

> **Goal 1.** Given the specifications, the drawing should "look right".
>
> **Goal 2.** (a) The Kolmogorov complexity of the final design should be provably small. In other words, the algorithm computing the drawing (by generating appropriate instructions for a printer, say) should be short. (b) It should be easy for an informed observer to perceive the algorithmic simplicity of the drawing and to see the object's essence extracted by the low-complexity artist.

It is predicted that the observer will like drawings that achieve both goals. The next section addresses the extent to which both goals are subjective.

3.1 How Subjective is Low-Complexity Art?

Goal 1 is clearly subjective in the sense that it strongly depends on a given observer and the way he or she interprets the (possibly informal) specifications. What "looks right" to an observer from one (sub)culture may "look wrong" to an observer from another (sub)culture (or another time).

Goal 2a depends on the nature of the computer running the algorithm. In what follows, this dependency will be ignored. Ultimately this is justified by the above-mentioned invariance theorem.

Like Goal 1, Goal 2b depends on the observer. But in a sense, Goal 2b is less subjective than Goal 1. This is because intelligent human observers, in principle, can learn to compute anything a digital computer can compute (the reverse is a matter of controversy). In particular, a short algorithm running on a conventional digital machine can be quickly taught to an intelligent human being. Note that if the human observer was another universal computer, then we could immediately apply the invariance theorem, thus (ultimately) eliminating subjectivity from Goal 2b. Then the only remaining subjective aspect of low-complexity art would be that of Goal 1.

3.2 Low-Complexity Art is Hard

This paper describes the goals of low-complexity art, not a way to achieve the goals. The latter requires intuition and, like with any other form of art, a sometimes rewarding but often frustrating struggle to capture "the essence" of what is being depicted. Initial attempts to create a work of low-complexity art will usually have unpredictable results.

The noncomputability of Kolmogorov complexity implies that there is no general method for finding the shortest description of a piece of data [5]. This seems to indicate that low-complexity art will always represent a challenge to any artist willing to pursue it.

3.3 Low-Complexity Design

Regarding the difference between art and design (this difference is important to many artists), I use the expression low-complexity design instead of low-complexity art in cases where no artistic purpose is pursued by a designer trying to achieve Goals 1 and 2.

The next section first defines a fractal [6] coding scheme for encoding drawings in algorithmic form. The coding scheme is easily learned and understood by most humans. Next, I present a set of cartoons, each of which satisfies some pre-determined formal specifications. Furthermore, each cartoon is algorithmically simple—its description (based on the coding scheme implemented on a conventional digital computer) does not require many bits of information. To illustrate Goal 2b, I use text and additional drawings to make the algorithmic simplicity of each cartoon obvious.

4 Fractal Circles for Coding Drawings

The fractal coding scheme I introduce here is general enough for designing arbitrary drawings. It is also sophisticated enough to allow for specification of non-trivial drawings with a limited amount of information. Finally, it is simple enough to be implemented by a short algorithm and to be taught quickly to a typical human observer.

The ancient Greeks considered the circle to be the ideal two- dimensional geometric form. Without necessarily agreeing with the Greeks I have used circles as the basis for designing drawings. One reason is that a circle can be drawn by a very short algorithm. Another reason is that circles are something most humans can relate to: most people know something about circles and their properties. These reasons make it easy to explain the algorithmic simplicity of the drawing to a typical observer (and thus achieve Goal 2b).

Sizes and relative positions of "legal" circles will be greatly limited by the following set of fractal rules.

4.1 Rules for Making Legal Circles

> **Initialization:** *Draw a circle of arbitrary radius and center position. Arbitrarily select a point on the first circle and use it as the center of a second circle with equal radius. The first two circles are defined as legal circles.*

The rules for generating additional legal circles are as follows:

> **Rule 1**. *Wherever two legal circles of equal radius touch or intersect, draw another legal circle of equal radius with the intersection point as its center.*
>
> **Rule 2.** *Within every legal circle with center point p and radius r, draw another legal circle whose center point is also p but whose radius is $r/2$.*

Figure 1 shows the result of a recursive application of the above rules.

4.2 Rules for Making Legal Drawings

A legal drawing is defined by (a) legal arcs or (b) legal areas. The rules for legal arcs and areas are:

Rule 3. *Each legal arc must be a segment of a legal circle.*

Rule 4. *At both endpoints of a legal arc, some legal circles must touch or intersect.*

Rule 5. *The arc width of a legal arc must be equal to the radius of some legal circle.*

Rule 6. *A legal area is an area whose border is a closed chain of legal arcs. Legal areas may be shaded using a small set of grey levels.*

Comments

1. On each legal circle the centers of six legal circles with the same radius can be found.

2. If the radius of the initial circle is defined as 1, the radius of any legal circle can be written as 2^{-n}, and any arc width can be written as 2^{-m}, where n, m are non-negative integers.

3. On a given area, there are about four times as many circles with radius 2^{-n-1}, as there are circles with radius 2^{-n}.

4. Low-complexity art is in no way limited to Rules 1–6. For instance, a drawing that only partly obeys Rules 1–6 may be a work of low-complexity art, as long as the deviations can be uniquely determined by a short algorithm.

4.3 Coding Drawings by Circle Numbers

There are many straightforward schemes for encoding drawings generated by Rules 1–6. Referring again to Fig. 1, let us define the radius of the initial circle (the frame) as 1. A visible circle is any circle wholly or partly covered by the initial circle. Starting with the initial circle, we generate all visible circles; each is given a number. The initial circle is given the number 1. There are 12 visible circles with radius 1 intersecting the initial circle. They are numbered 2, 3, . . . , 13 (in some deterministic fashion—clockwise, for instance). There are 31 visible circles with radius 0.5 (partly) covered by the initial circle. They are numbered 14, 15, . . . , 44, and so on.

Obviously, there are few big circles with small numbers. There are many small circles with large numbers. In general, the smaller a circle, the more bits needed to specify its number.

An unshaded drawing is specified by a set of legal arcs (forget Rule 6 for the moment). For each legal arc l, we need to specify the number of the corresponding circle c_l, the start point s_l, the end point e_l, and the line width w_l. By convention, arcs are drawn clockwise from s_l to e_l. Once we know c_l, we can specify s_l by specifying the number of the circle touching or intersecting c_l in s_l. In general, an extra bit is necessary to differentiate between two possible intersections. Similarly for e_l. Thus, all "pixels" of a legal arc may be compactly represented by a triple of circle numbers, two bits for intersection differentiation, and a few bits for the line width.

Clearly, the larger the circles used, the fewer the number of bits needed to specify the corresponding legal arcs and the simpler (in general) the drawing. By using very many very small circles (beyond the resolution of the human eye), anything can be drawn (using Rules 4 and 5) so that it looks "right." This would not be very impressive, however, because a lot of information would be required to specify the drawing. It would be more impressive if it were possible to draw something non-trivial that looks right using only legal arcs defined by a few large circles. In a way, this would be related to capturing an object's essence, provided one agrees that the essence of an object is inherent in the sortest algorithm describing the object. Such compact representation can be difficult, however. I found that it is much easier to come up with acceptable complex drawings than with acceptable simple drawings of given objects.

Why use fractal circles instead of fractal squares? Since I prefer to sketch living objects as opposed to inanimate objects, and since I found it hard to convincingly sketch living objects without using curved lines of some kind or another, I decided to use circles as a basis for my fractal scheme.

Often the low-complexity artist will use drawing-specific symmetries and the like to further compress the description of a drawing. The next section will present examples of this.

5 Examples of Cartoons Based on Rules 1–4

The following is a set of informal specifications of the cartoons I will present:

Figure 2a: The (informal) goal was to draw a butterfly approaching a vase with a flower.

Figure 3a: The goal was to design a woman's profile.

Figure 4a: The goal was to design a logo for a gym based on a weight lifter's upper body.

Figures 2a, 3a and 4a are examples of cartoons designed by using Rules 1–6 only. Instead of providing each drawing's somewhat opaque coding sequence, I refer the reader to Figs 2b, 3b and 4b, which graphically illustrate the algorithmic simplicity of the corresponding cartoons. In conjunction with Fig. 1, each graphic illustration allows the algorithmic simplicity of its corresponding cartoon to be described quickly to the human observer.

I should mention, however, that simplicity of expression is a defining feature of cartoons. Although great artists often are distinguished by their ability to communicate the most expressive content using the fewest strokes of brush or pen, successful cartooning using low-complexity drawings does not necessarily translate to other artistic disciplines.

The figures demonstrate that the circle scheme is quite flexible. In terms of bits, it is cheaper to encode all cartoons (Figs 2a, 3a and 4a) simultaneously than to encode each cartoon separately because the algorithm for generating legal circles and their numbers is shared by all three cartoons. In the terminology of algorithmic information theory, the cartoons share a nontrivial amount of mutual algorithmic information. The circle scheme can be viewed as something like a common recognizable artistic style.

A drawing that can be computed by a short algorithm typically exhibits strong relationships between the whole and its parts, essentially because the same short pieces of code have to be used repeatedly to generate all parts of the drawing. In the terminology of algorithmic information theory, there is a great deal of mutual algorithmic information (coherence) between different parts of the drawing and also between the whole and its parts. The circle cartoons exemplify this: many of their details are based on identical or similar shapes and arcs, computed by the same subprogram executed again and again. Likewise, many parts of the drawings are self-similar.

6 On Beauty and Minimum Description Length

Sometimes an artist is appreciated for a distinctive style. Sometimes certain works of art are perceived as beautiful. This section attempts to relate the formalism of the theory of minimum description length (MDL) [8] to informal notions such as "beauty" and "good artistic style." This section, however, is not intended to address all aspects of beauty and interest, but only those related to "capturing the essence."

6.1 What is a Beautiful Drawing?

What is beautiful? What is not? There clearly are no objective answers to these questions. What is considered beautiful by one observer may be regarded as ugly by another observer. Ideals of beauty are different in different cultures and subcultures, they have changed over the centuries, and they are not even stable with respect to a single individual. Therefore, any theory of beauty has to take the observer into account.

Following common sense, I assume that a typical human observer tries internally to represent input data in terms of what is familiar. Regarding the observer's subjectivity, I assume that the Church-Turing thesis is true (everything that can be computed by a human being can be computed by an appropriate program for a general-purpose computer) and postulate the following setting. At a given time, a human observer's current knowledge about visual scenes can be described as a coding algorithm. This algorithm maps input data (such as retinal activity caused by a work of art in the visual field) onto internal representations of the data. The coding algorithm C, the data D and its internal representation D' can be written as strings of symbols from a finite alphabet. If D' conveys all information about D, but the length of D' is less than the length of D, then D is compressible or redundant with respect to the observer's knowledge. The observer already knew something about D. Similar statements can be made in cases where D' allows only for partial reconstruction of D.

The observer's subjectivity is embodied by the coding algorithm C. One may be tempted to define the beauty of a drawing with respect to C. In the following preliminary attempt to do so (inspired by the MDL approach), I assume that "beauty" simply corresponds to "high conditional probability given C": Given C, the best way of selecting a drawing s from a set or class S of possible drawings satisfying certain specifications may be to maximize $P(s \mid C)$, the conditional probability of s, given C. Bayes' formula tells us

$$P(s \mid C) = \frac{P(C \mid s)P(s)}{P(C)}, \tag{1}$$

or, equivalently,

$$-logP(s \mid C) = -logP(C \mid s) + logP(C) - logP(s). \tag{2}$$

Let us interpret this. Since C is given, $P(C)$ may be viewed as a normalizing constant. It can be disregarded. $-logP(C \mid s)$ can be interpreted as the information (or length of the observer's shortest algorithm) required to compute C from s. $P(s)$ is given by some *a priori* distribution on the drawings. For simplicity, let us assume that this prior is uniform. Then, given C, $s \in S$ is optimal (most likely, most "beautiful") if the information required to compute C from s is minimized.

How can this be related to human experience? The following example attempts to establish such a relationship.

"Beautiful" faces. Human beings appear to have a certain coding scheme for storing faces in memory. This scheme is certainly different from the circle scheme described earlier. It is probably based on previous experiences with many different faces, and it is probably adapted to code many faces efficiently. One way of doing so is to store a prototype face and code new faces by coding only the deviations from the prototype.

The principle of minmal description length suggests that the "ideal" (most likely) prototype F_P maximizes $P(F_P \mid F)$, thus minimizing

$$-logP(F \mid F_P) - logP(F_P),$$

where F is a given set of all faces to be coded. In other words, the optimal prototype minimizes the sum of the description lengths of all faces relative to the prototype, as well as of the description length of the prototype itself (relative to the observer's remaining knowledge about visual scenes).

Assuming that all faces are equally likely to appear in the visual field, the formalism above predicts that the most beautiful face is the one that can be most easily computed from the coding scheme. It seems reasonable to assume that the information required to specify the coding scheme is dominated by the information required to specify the prototype face. If the current face looks like the prototype face, then there is very little additional information to compute. This would imply that the prototype face is perceived as the most beautiful one.

Previous work on attractive faces. The statement above seems compatible with results presented by Langlois and Roggmann [9], who claim that the "average face" (computed by digital blending of numerous photos of real faces) is perceived as the most attractive one. Perrett, May and Yoshikawa [10] partly dispute this claim, however. Their test subjects also appreciate average faces computed by blending [11] but prefer "attractive average faces" constructed from faces perceived as attractive. Indeed, the most attractive faces are caricatures obtained by digitally exaggerating the deviations between "average" and "attractive average."

Critique of previous work. The studies above, however, do not say much about the plausibility of the algorithms used to compute average faces. Let us assume that the brain does indeed support face-processing by an ideal (in the information theoretic sense) prototype face. It would be naive to assume that the ideal face equals the one computed by blending. There are many plausible algorithms for computing prototypes, based on many plausible metrics for "distances" between faces. Therefore the studies above, including the statement that the average face is not the most attractive one, have to be judged with skepticism. The presented claims depend on the definition of "average" and the corresponding nature of the blending algorithms, which may not be very closely related to a hypothetical method the brain might be using for generating the "optimal" prototype F_D. Unfortunately, at the present time it seems impossible to analyze the way the brain stores representations of objects. Therefore it also seems impossible to test the predictions made by the formalism presented above.

Comments

1. Beauty and evolution. One may continue to speculate as follows: A society with a distribution of faces corresponding to an algorithmically simple prototype face may have an evolutionary advantage. This is because face recognition (based on the given hardware, the brain) may be more successful or efficient in such a society. Evolutionary pressure may favor beautiful prototypes, where beauty is defined by the nature of the computations our brain handles well. On the other hand, the nature of these computations is influenced by typical face recognition tasks to be solved. It is difficult to analyze such mutual dependencies.

2. Learning coding schemes. Most certainly, the fractal circle scheme I discuss is different from the typical human coding scheme. Therefore, the most beautiful cartoons relative to the circle scheme will be different from the most beautiful cartoons relative to the coding scheme of most humans. Unfortunately, I cannot extract the latter (although many artists implicitly try to guess it, I believe). However, humans can learn new coding schemes. In particular, it is not hard to learn the circle scheme. Therefore I hope that some of the cartoons in this paper will be easily accessible to some readers.

3. Something "beautiful" needs not be "interesting". Interest has to do with the unexpected. But not everything that is unexpected is interesting—just think of white noise. One reason for the interestingness (for some observers) of some of the pictures shown here may be that they exhibit unexpected structure. Certain aspects of these pictures are not only unexpected (for a typical observer), but unexpected in a regular, non-random way. The formalization of "interestingness" requires an extension of the formalism above. This, however, is beyond the scope of this discussion.

6.2 What is Good Artistic Style?

I made use of the fact earlier that humans can learn new coding algorithms. In a way, our innate coding algorithm is universal enough to allow for implementing new coding algorithms. One important subgoal of low-complexity art is to devise good coding algorithms. What does this mean? A new coding scheme may be considered "good" by a given observer if (1) it does not require many bits to be specified (given the observer's previous knowledge), and (2) many different drawings (satisfying typical specifications) can be encoded efficiently by it. In that case the drawings share a non-trivial amount of algorithmic information, and the coding scheme represents something like a common style. A given coding scheme may be representative for a given artist, which will make that artist stylistically recognizable.

More formally, the quality of an artistic style or coding scheme C may be evaluated as follows. An optimal style C maximizes $P(C \mid S)$, the conditional probability of the style, given a set of drawings S (defined by a set of specifications). Equivalently, an optimal style C minimizes

$$-logP(C \mid S) = -logP(S \mid C) + logP(S) - logP(C). \tag{3}$$

Since S is given, $P(S)$ may be viewed as a normalizing constant, and it may be ignored. The term, $-logP(S \mid C)$, can be interpreted as the information required to compute all elements in S from C. The term, $P(C)$, is given by some a priori distribution on the coding schemes and depends on the observer. $-logP(C)$ can be interpreted as the information necessary to specify C, given the observer's knowledge. Thus, given S, C is optimal (most likely) if the sum of two terms is minimized: (1) The information required to compute S from C, and (2) the information required to compute C from the observer's previous coding scheme.

The circle scheme is easy to teach, which may be another way of saying that not much information is required to compute it from typical human knowledge. In this case, the second term appears negligible. Therefore, given the drawings presented in this paper, the circle scheme appears to correspond to a "good" (although probably non-optimal) artistic style.

7 Final Remarks

7.1 On the Difficulty of Creating Low-Complexity Art

This paper discusses (and exemplifies) the nature of low-complexity art without providing a general method for creating it. Although it is trivial to redraw the concrete examples shown here, they seem to appear "out of the blue," without giving indication of how they were discovered. Nor do they give many clues about how to draw other objects. No universal algorithm for generating low-complexity art is known. At the moment, a human artist is still required.

I found it difficult to discover acceptable but algorithmically simple cartoons. I found it easier to come up with acceptable cartoons that appeared to be algorithmically complex.

7.2 Relation to Previous Work

This paper certainly is not the first to introduce formal rules in art. For instance, Lyonel Feininger writes:

> *"Aber die Erkenntnis ist mein, dass es in der ganzen Welt, in allen Welten, nichts Ungesetzliches, nichts Zufälliges, nichts ohne Form und Rhythmus gibt noch geben kann. Warum dann gerade in der Kunst? Soll diese nicht dann, indem sie des Menschen schöpferischen Willen offenbart, gerade voll Form, voll Gesetz und Geist sein?"*
>
> English translation (free translation by the author — subtleties in the original text source may be lost): *"But the insight is mine, that in the whole world, in all worlds, nothing unlawful, nothing random, nothing without form and rhythm exists nor can exist. Why then in the arts? Shouldn't the arts, by exhibiting man's creative will, be filled with form, law and spirit?" [12]*

The ancient Greeks, Leonardo da Vinci, Albrecht Dürer, LeCorbusier and many others devised formal rules to draw things. Most rules are based on simple proportions—for example, "The distance between the eyes should equal the eye-width" (origin unknown), and "The ratio of the distance from toes to navel and the distance from toes to top of the head should equal the harmonic proportion" (LeCorbusier). (The harmonic proportion is obtained when a straight line of length a is divided into two segments of lenghts b and c, such that $\frac{a}{b} = \frac{b}{c}$. One solution is $b = a\frac{\sqrt{5}-1}{2}$.)

I do not claim to be the first to present examples of low-complexity art, however. Certain representations of artistically interesting and aesthetically pleasing representations of fractal objects may be regarded as works of low-complexity art, such as pictures of hills, coastlines, etc. [13]. This is because they are based on relatively short and understandable algorithmic descriptions. Also, certain data-compression methods can be used to generate patterns that can be regarded as works of low-complexity art [14]. Similar statements could be made about certain other simplifying computer models for representing objects [15]. Also, many graphic designers consciously or unconsciously use their tools to come up with algorithmically simple designs.

Apparently, however, nobody so far has identified low-complexity art and low-complexity design as such in its most general form. An aim of this paper is to make explicit the nature of low-complexity art and low-complexity design—the creation of understandable works of art (or designs) with low Kolmogorov complexity. This also provides a framework for the categorization of previous work.

Finally, it should be mentioned that there have been attempts to use classic information theory [16] to formalize what is aesthetically pleasing [17]. Another contribution of this paper is to offer an alternative approach based on algorithmic information theory.

7.3 Outlook

Possibilities for artistic expression depend on available technology. The cave artists from the stone age did not have the technology for creating the colors that made impressionism possible. The impressionists did not have today's computer graphics. This does not imply that the best cave drawings are less perfect than the best impressionist paintings, nor that the best impressionist paintings are less perfect than the best computer graphics. Each age, however, tends to have its preferred means of artistic expression. What can we expect with regard to the future of low-complexity art?

There will be tools that will simplify the creation of low-complexity art. I expect significant extensions of the programs used to speed up and simplify the creation of the drawings shown here. In particular, I expect "virtual ateliers" implemented on powerful machines. A virtual atelier will be accessible via stereoscopic virtual-reality interfaces. It will allow complex three- or higher-dimensional objects to be quickly composed from simpler ones by hand movements (perceived by the machine via data gloves or similar devices). For instance, with a virtual atelier it will be easy to extend the circle scheme to an analogous sphere scheme or bubble scheme. The algorithmic atelier will permit the artist to quickly generate sequences of three-dimensional sketches of sculptures, to evaluate them with respect to their artistic value and to discard them or refine them.

In principle, the technology for building virtual ateliers is available. Given the current inflation of cheap computing power, we may expect that it will not take long before many artists will have access to acceptable virtual ateliers.

8 Appendix: Technical Details of Kolmogorov Complexity

This appendix represents a formal equivalent of the basic concepts discussed in the article.

Each Turing machine (TM) C (mapping bitstrings to bitstrings, without loss of generality) computes a partial function $f_C : \{0,1\}^* \rightarrow \{0,1\}^*$ (f_C is undefined where C does not halt).

Compiler theorem. There is a universal TM U with the following property: for every TM C there exists a constant prefix μ_C (a bitstring) such that $f_C(p) = f_U(\mu_C p)$ for all bitstrings p. μ_C is the compiler that compiles programs for C into equivalent programs for U.

Kolmogorov complexity. The Kolmogorov complexity $K_U(s)$ of a finite string s is the length of the shortest program p that computes s on a universal Turing machine U and halts, where the set of possible halting programs forms a prefix code (no halting program can be the prefix of another one):

$$K_U(s) = \min_p \{ \mid p \mid : \ f_U(p) = s \},$$

where $\mid p \mid$ denotes the length of p. In general, $K_U(s)$ is noncomputable.

Invariance theorem. Due to the compiler theorem, $K_{U_1}(s) = K_{U_2}(s) + O(1)$ for two universal machines U_1 and U_2. We choose one particular universal machine U and write $K(s) = K_U(s)$.

9 Acknowledgements

Thanks to Ulrike Krommer, Christian Freksa, Mark Ring, Christof Schmidhuber, Kurt Prelinger, Karolin Schmidhuber, Gerhard Weiss, Astrid Kiehn, Rohan Baxter, Margit Kinder, Martin Eldracher, Daniel Prelinger, Sepp Hochreiter, Gary Greenfield, and Charles Ames, for useful comments.

10 References

1. M. Li and P.M.B. Vitanyi, An Introduction to Kolmogorov Complexity and Its Applications (Springer-Verlag, New York, 1993). Page v in preface.

2. A.N. Kolmogorov, "Three Approaches to the Quantitative Definition of Information," Problems of Information Transmission 1 (1965) pp. 1–11; G.J. Chaitin, "On the Length of Programs for Computing Finite Binary Sequences: Statistical Considerations," Journal of the ACM 16 (1969) pp. 145–159; and R.J. Solomonoff, "A Formal Theory of Inductive Inference, Part 1," Information and Control 7 (1964) pp. 1–22.

3. See Li and Vitanyi [1] for the best overview. See J. Schmidhuber, "Discovering neural nets with low Kolmogorov complexity and high generalization capability", Neural Networks (1997, in press), for a machine learning application. A short version of this text has appeared in A. Prieditis and S. Russell, editors, Machine Learning: Proceedings of the Twelfth International Conference, pages 488–496, Morgan Kaufmann Publishers, San Francisco, CA, 1995.

4. See Kolmogorov [2]; Chaitin [2]; and Solomonoff [2].

5. See Kolmogorov [2]; Chaitin [2]; and Solomonoff [2].

6. B. Mandelbrot, The Fractal Geometry of Nature (San Francisco: Freeman, 1982).

8. See Kolmogorov [2]; Chaitin [2]; and Solomonoff [2]; L.A. Levin, "Laws of Information (Nongrowth) and Aspects of the Foundation of Probability Theory," Problems of Information Transmission 10, No. 3, 206–210 (1974); C.S. Wallace and D.M. Boulton, "An Information Theoretic Measure for Classification," Computer Journal 11, No. 2, 185–194 (1968); and J. Rissanen, "Modeling by Shortest Data Description," Automatica 14 (1978) pp. 465–471.

9. J.H. Langlois and L.A. Roggman, Attractive Faces Are Only Average, Psychological Science 1 (1990) pp. 115–121.

10. D.I. Perrett, K.A. May and S. Yoshikawa, "Facial Shape and Judgments of Female Attractiveness," Nature 368 (1994) pp. 239– 242.

11. P.J. Benson and D.I. Perrett, "Extracting Prototypical Facial Images from Exemplars," Perception 22 (1993) pp. 257–262.

12. Lyonel Feininger, translated by J. Schmidhuber.

13. See Mandelbrot [6]; and H.-O. Peitgen and P.H. Richter, The Beauty of Fractals: Images of Complex Dynamical Systems (Springer-Verlag, New York, 1986).

14. K. Culik and J. Kari, "Parallel Pattern Generation with One-Way Communications," in J. Karhumaki, H. Maurer and G. Rozenberg, eds., Results and Trends in Theoretical Computer Science (Springer-Verlag, New York, 1994) pp. 85–96.

15. For work on face animation, see I.A. Essa, T. Darrell and A. Pentland, "Modeling and Interactive Animation of Facial Expressions Using Vision," Technical Report, MIT Media Lab Perceptual Computing Section TR 256, 1994. See also Mengxiang Li, "Minimum Description Length Based 2-D Shape Description," in IEEE 4th Int. Conference on Computer Vision (May 1992) pp. 512–517.

16. C.E. Shannon, "A Mathematical Theory Of Communication (Parts 1 and 2)", Bell System Technical Journal 27 (1948) pp. 379–423.

17. F. Nake, Ästhetik als Informationsverarbeitung (Springer-Verlag, Berlin, 1974).

18. See Kolmogorov [2]; Wallace and Boulton [8]; and Chaitin [2].

Manuscript received 2 January 1995.

Fig. 1. Fractal result of an iterative application of Rules 1 and 2 (following initialization). This self-similar figure builds the basis of all the figures that follow. On each legal circle there are the centers of six legal circles with equal radius. For clarity, each legal circle is drawn with a width proportional to the logarithm of its radius. Parts of legal circles located outside the initial circle (the "frame") are not shown. Also, circles with radius less than 1/16 the radius of the frame are not shown.

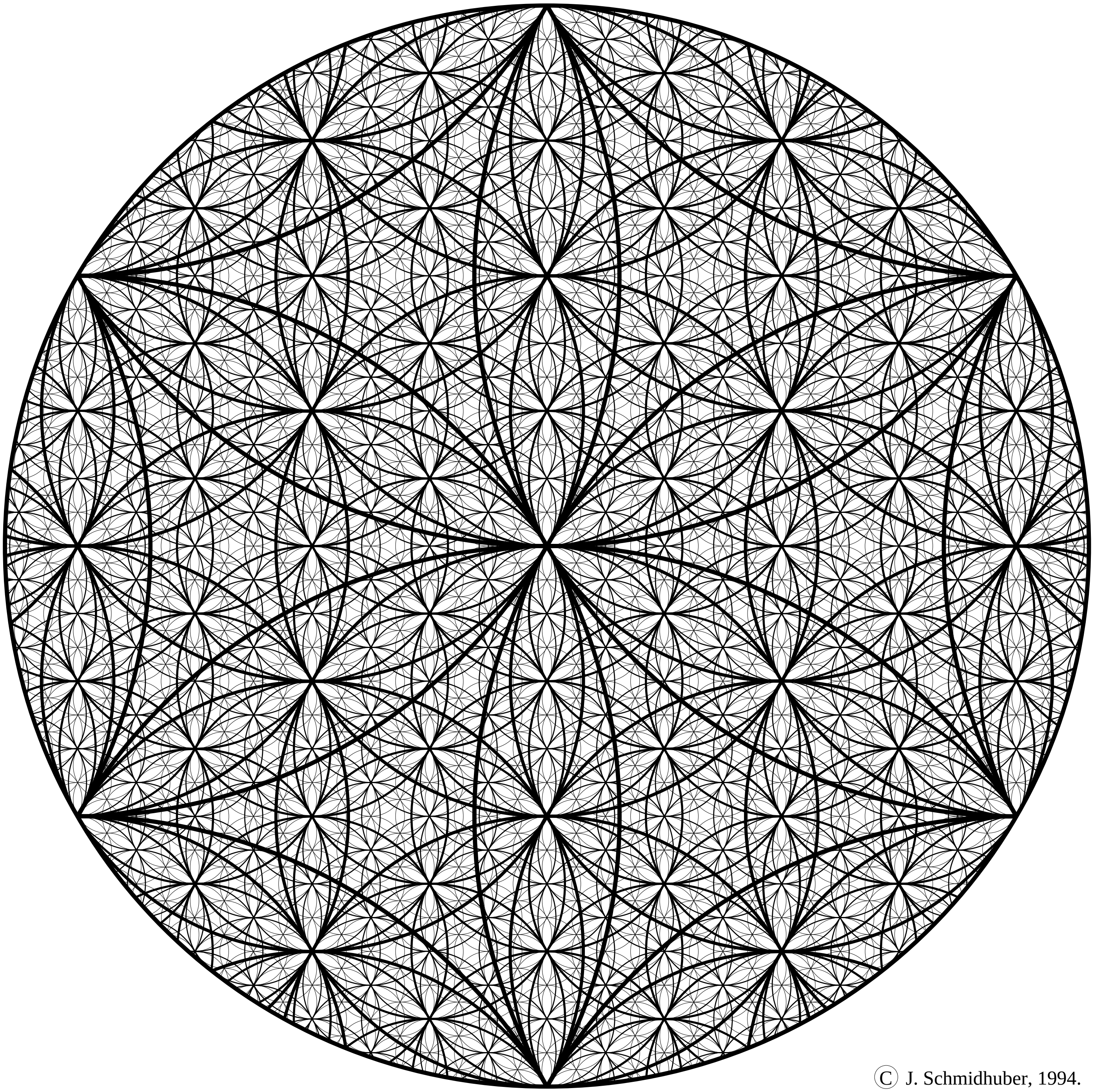


Fig. 2a. Butterfly approaching a vase with a flower. Only Rules 1–6 were used to design the drawing.

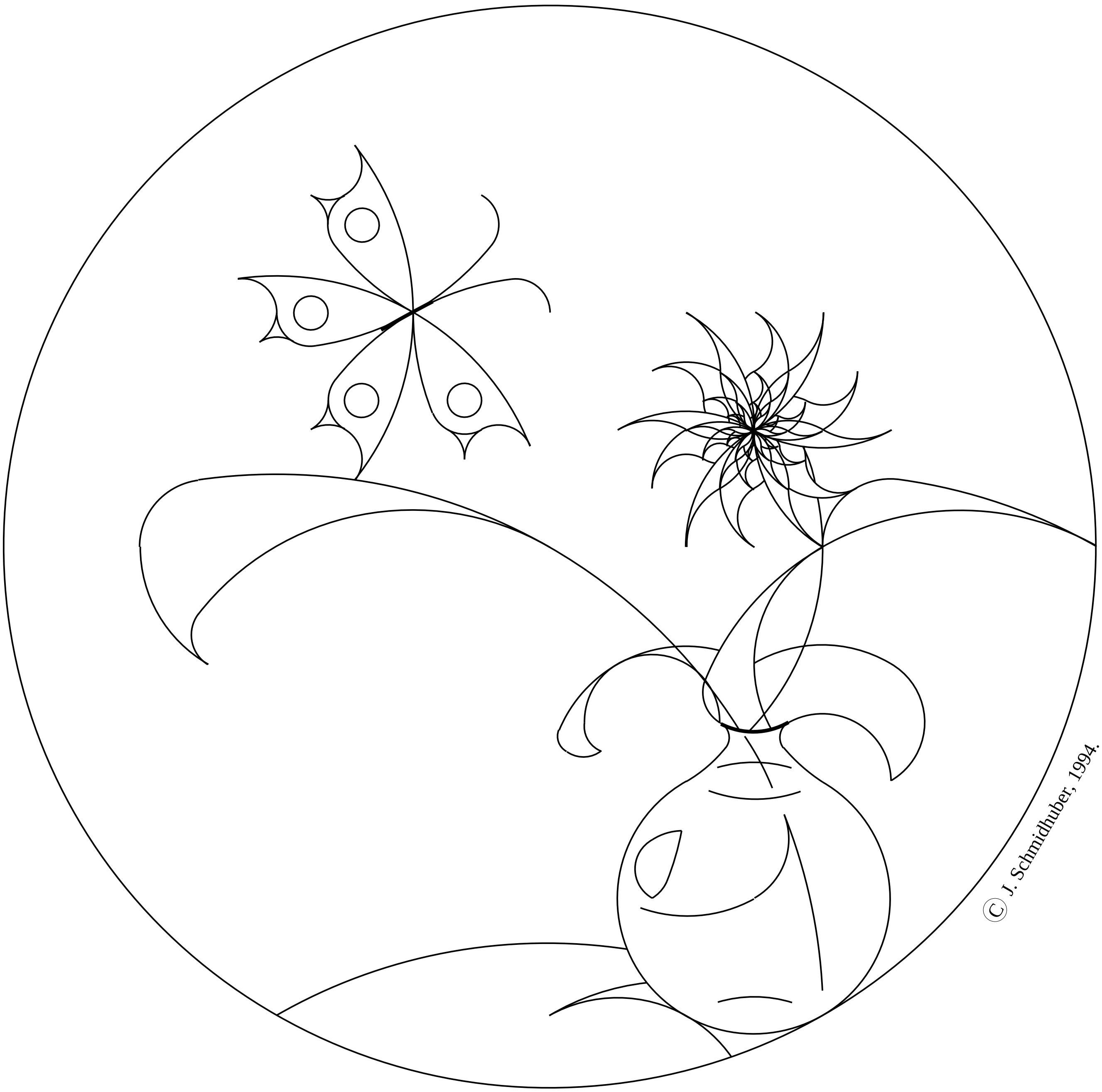
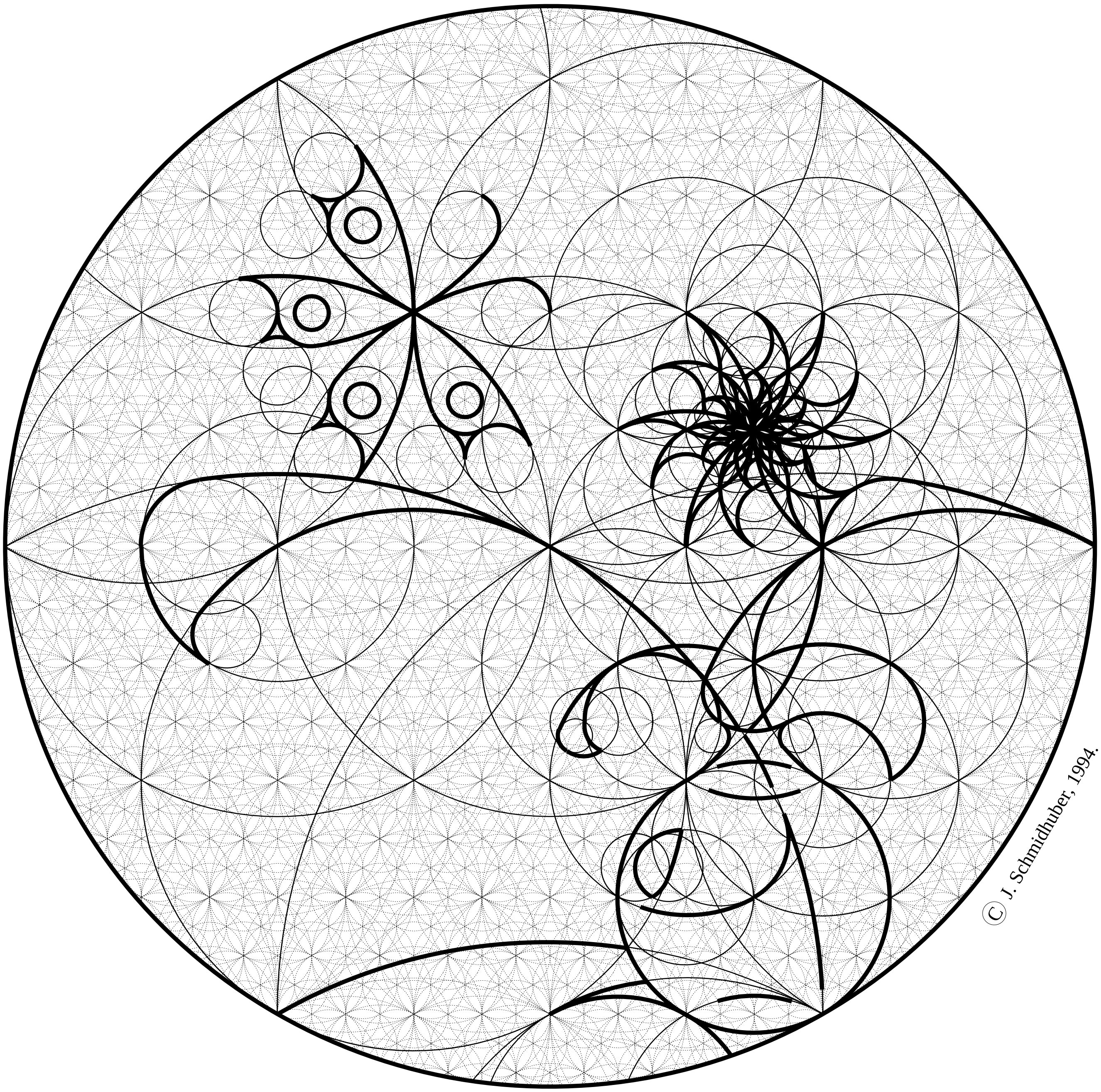


Fig. 2b. Illustration of the low Kolmogorov complexity (or algorithmic simplicity) of Fig. 2a. All circles shown are taken from Fig. 1; very few of them, however, are needed to specify this drawing. Many of the arcs are large and do not require many bits to be specified.



Woman's profile. Again, only Rules 1–6 were used to design the drawing.

Fig. 3b. Illustration of the low Kolmogorov complexity (or algorithmic simplicity) of Fig. 3a. All circles shown are taken from Fig. 1. Most of the circles used (drawn with thicker lines) are comparatively large—their unique specification requires only a few bits. Note the symmetries in nose, lips and chin — many of the same contours reappear, thus "going together well".

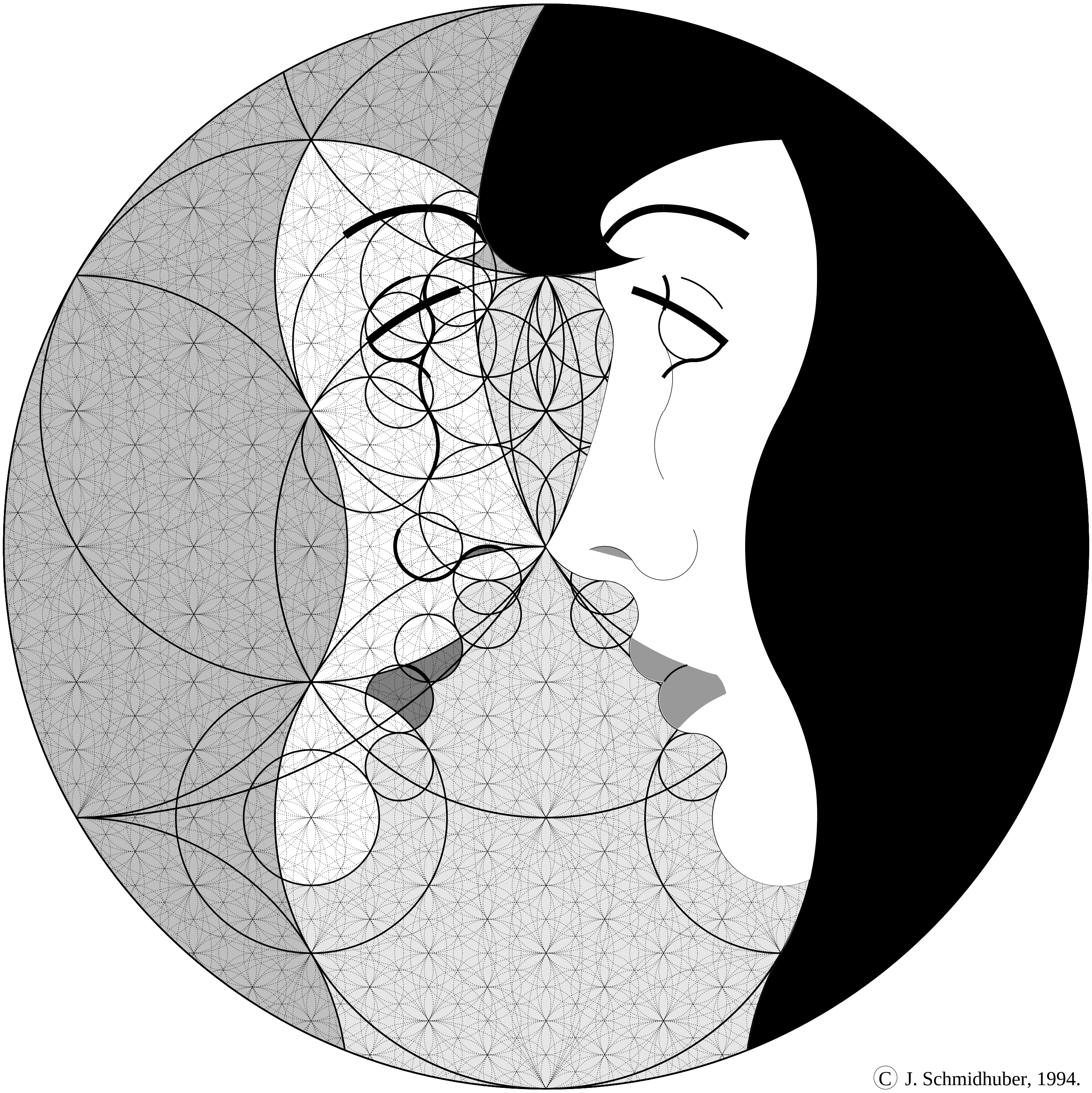


Fig. 4a. Cartoon of a weight lifter's upper body, designed for the logo of a gym. Again, only Rules 1–6 were used to specify the cartoon.

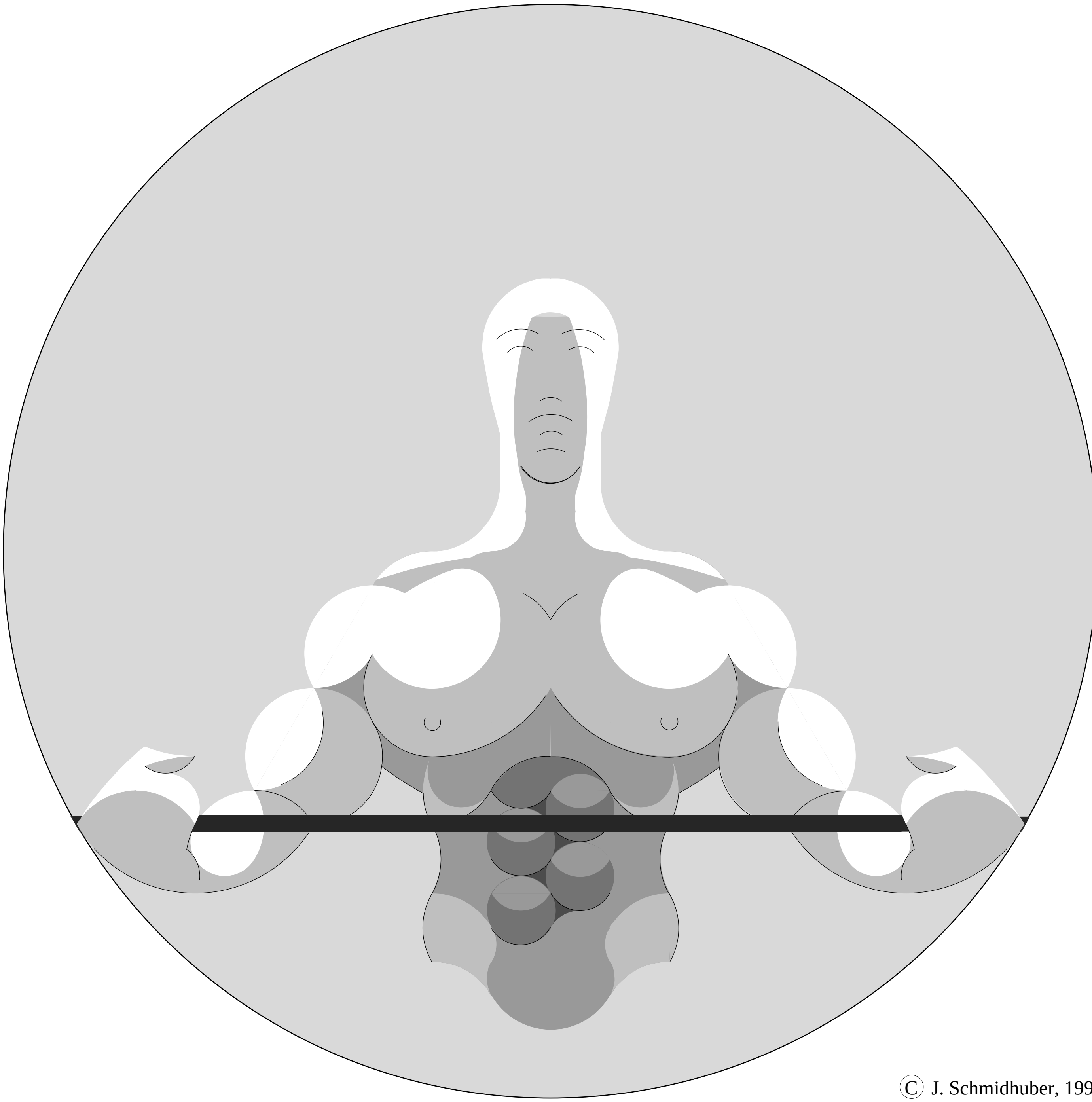


Fig. 4b. Illustration of the low Kolmogorov complexity (or algorithmic simplicity) of Fig. 4a. All circles shown are legal circles taken from Fig. 1. Note that shoulders and biceps/triceps are shaped by circles of equal size. This may be viewed as an idealization of what can be observed in certain human weight lifters. The same circle size is used for many additional features, such as the top of the head, parts of the chest, etc. Mirror symmetry is broken only for the abdominal muscles. The visible part of the dumbbell belongs to a legal circle with nearly infinite radius (drawn with a huge but legal arc width).