Construction and Intuition: Creativity in Early Computer Art

Chapter · August 2012

DOI: 10.1007/978-3-642-31727-9_3

Frieder Nake
Universität Bremen

Chapter 3
Construction and Intuition: Creativity in Early Computer Art

Frieder Nake

Abstract This chapter takes some facets from the early history of computer art (or what would be better called "algorithmic art"), as the background for a discussion of the question: how does the invention and use of algorithms influence creativity? Marcel Duchamp's position is positively referred to, according to which the spectator and society play an important role in the creative process. If creativity is the process of surmounting the resistance of some material, it is the algorithm that takes on the role of the material in algorithmic art. Thus, creativity has become relative to semiotic situations and processes more than to material situations and processes. A small selection of works from the history of algorithmic art are used for case studies.

3.1 Introduction

In the year 1998, the grand old man of German pedagogy, Hartmut von Hentig, published a short essay on creativity. In less than seventy pages he discusses, as the subtitle of his book announces, "high expectations of a weak concept" (Hentig 1998). He calls the concept of creativity "weak". This could mean that it is not leading far, it does not possess much expressive power, nor is it capable of drawing a clear line. On the other hand, many may believe that creativity is a strong and important concept.

Von Hentig's treatise starts from the observation that epochs and cultures may be characterised by great and powerful words. In their time, they became the call to arms, the promise and aspiration that people would fight for. In ancient Greece, Hentig suggests, those promises carried names like *arete* (excellence, living up to one's full potential), and *agon* (challenge in contest). In Rome this was *fides* (trust) and *pietas* (devotion to duty), and in modern times this role went to *humanitas*, enlightenment, progress, and performance. Hardly ever did an epoch truly live up to what its great aspirations called for. But people's activities and decisions, if only ideologically, gained orientation from the bright light of the epoch's promise.

F. Nake (✉)
University of Bre Germany
e-mail: nake@informatik.uni-bremen.de

J. McCormack, M. d'Inverno (eds.), *Computers and Creativity*,
DOI 10.1007/978-3-642-31727-9_3, © Springer-Verlag Berlin Heidelberg 2012

If in current times we were in need of a single such concept, "creativity" would probably be considered as one of the favourites. Information, communication, sustainability, ecology, or globalisation might be competing. However, creativity would probably still win. It is a concept full of shining promise. Nobody dares criticise it as plastic and arbitrary. Everybody appears to be relating positively to it. Technofreaks use it as well as environmentalists. No political party would drop it from their rhetoric.

Creativity may be considered as a *means* for activity, or as its *goal*. However, von Hentig is sceptical about the possibility of developing *more* creativity through education and training; he is also sceptical about creative skills independent of the context. Creativity as an abstract, general concept, taken out of context, is unlikely to exist. If a helpful concept at all, creativity is bound to situations and contexts. Only relative to them may our judgement evaluate an activity as creative. Creativity exists only concretely.

Leaving out ancient Greece, the Middle Ages, and the Renaissance, it seems that the way we understand "creativity" *today* is as a US-American invention (Hentig 1998, p. 12). It started with the fabulous definition of an IQ (Intelligence Quotient) and operational tests to measure it by Stern (1912) in Germany. His approach became an operational method in the USA by the end of World War I. J.P. Guilford (1950) and others made clear that IQ tests did not identify anything that might be called "creative". Current creativity research starts from this article. Like any other measure, a test of your IQ may at best say something about a standard behaviour within given boundaries, but not much about crossing boundaries. Often people do what they are *supposed* to do, and they do it well. Others do what they *want* to do, and do it to the dismay of their bosses, teachers, or parents.

When we consider creativity as an attribute, a property, or a feature that we may acquire by taking courses or joining training camps, we put creativity close to a thing, or a commodity. We inadvertently transform a subjective activity or behaviour into an objective thing. We may acquire many or few commodities, cheap or expensive ones. But is quantity important for understanding creativity, or for becoming a creative person? Doesn't it make more sense to associate the term "creativity" with behaviour, activity, situation, and context? The idea of attaching creativity to individuals is probably what we are immediately inclined to think. But it may still not be very helpful. Creativity seems to emerge in situations that involve several people, who interact in different roles with favourable and unfavourable conditions and events.[1]

We may align intelligence with making sense in a situation that makes sense. If we do so, creativity could be viewed as making sense in situations of nonsense. Dream and fantasy are, perhaps, more substantial to creative behaviour than anything else.

[1] We are so much accustomed to thinking of creativity as an individual's very special condition and achievement that we react against a more communal and cooperative concept. It would, of course, be foolish to assume individuals were not capable of creative acts. It would likewise be foolish to assume they can do so without the work of others.

With these introductory remarks I want to announce a sceptical distance to the very concept of creativity. With only little doubt, a phenomenon seems to exist that we find convenient to call by this name. A person engaged in a task that requires a lot of work, imagination, endurance, meetings, walks, days and nights, music, or only a flash in the mind, will use whatever means she can get hold of in pursuit of her task. Even computers and the Internet may be helpful, and they, indeed, often are. If the final result of such efforts is stamped as a "creative" product, is it then sensible to ask the question: what software and other technical means contributed to this creation? Not much, in my view. And certainly nothing that goes beyond their instrumental character. More interesting is to study changes in the role of the instrument as an instrument. The sorcerer's broom is more than a broom only in the eyes of the un-initiated. It is an expression of a human's weakness, not of the instrument's clever strength.

Therefore, I find it hard to seriously discuss issues of the kind: how to enhance creativity by computer? Or: how do our tools become creative? If anything is sure about creativity, it is its nature as a quality. You cannot come by creativity in a quantitative way, unless you reduce the concept to something trivial.

In this chapter, I will study a few examples of early computer art. The question is: How did the use of computers influence creative work in the visual arts? The very size and complexity of the computer, the division into hardware and software must, at the time, have had a strong influence on artistic creativity. The approach will be descriptive and discursive. I will not explain. Insight is with the reader and her imagination, not with the black printed material. I will simply write and describe. I cannot do much more.

The chapter is divided into four narrations. All four circle around processes of art or, in a less loaded expression, around aesthetic objects and processes. The art we will study here is, not surprisingly, algorithmically founded. It is done, as might be said, by *algorists*.[2] They are artists of a new kind: they *think* their works and let machines carry them out. These artists live between aesthetics and algorithmics and, insofar, they constitute a genuinely new species. They do art in postmodern times. When they started in the 1960s, they were often called computer artists, a term most of them hated. Meanwhile, their work is embraced by art history, they have conquered a small sector of the art market, and their mode of working has become ubiquitous.

The first narration will be about a kind of mathematical object. It is called a *polygon* and it plays a very important role. The narration is also about randomness, which at times is regarded as a machinic counterpart to creativity.

Three artists, Vera Molnar, Charles Csuri, and Manfred Mohr, will be the heroes of the second narration. It will be on certain aspects of their work pertaining to our general topic of creativity.

---

[2]There actually exists a group of artists who call themselves, "the algorists". The group is only loosely connected, they don't build a group in the typical sense of artists' groups that have existed in the history of art. The term *algorist* may have been coined by Roman Verostko, or by Jean-Pierre Hébert, or both. Manfred Mohr, Vera Molnar, Hans Dehlinger, Charles Csuri are some other algorists.

Two programs will be citizens first class in the third narration: Harold Cohen's *AARON* stands out as one of the most ambitious and successful artistic software development projects of all time. It is an absolutely exceptional event in the art world. Hardly known at all is a program Frieder Nake wrote in 1968/69. He boldly called it *Generative Art I*. The two programs are creative productions, and they were used for creative productions. Their approaches constitute opposite ends of a spectrum.

The chapter comes to its close with a fourth narration: on creativity. The first three ramblings lead up to this one. Is there a conclusion? There is a conclusion insofar as it brings this chapter to a physical end. It is no conclusion insofar as our stories cannot end. As Peter Lunenfeld has told us, digital media are caught in an aesthetics of the *unfinish* (Lunenfeld 1999, p. 7). I like to say the same in different words: the art in a work of digital art is to be found in the infinite class of works a program may generate, and not in the individual pieces that only *represent* the class.

I must warn the reader, but only very gently. There may occasionally be a formula from mathematics. Don't give up when you see it. Rather read around it, if you like. These creatures are as important as they are hard to understand, and they are as beautiful as any piece of art. People say, Mona Lisa's smile remains a riddle. What is different, then, between this painting and a formula from probability theory? Please, dear reader, enter postmodern times! We will be with you.

3.2 The First Narration: On Random Polygons

Polygons are often boringly simple figures when it comes to the generation of aesthetic, or even artistic objects. Nevertheless, they played an important role in the first days of computer art. Those days must be considered high days of creativity. Something great was happening then, something took on shape. Not many had the guts to clearly say this. It was happening at different places within a short time, and the activists were not aware of each other. Yet, what they did, was of the same kind. They surprised gallery owners who, of course, did not really like the art because, how could they possibly make money with it? With the computer in the background, this was mass production.

If the early pioneers themselves did not really understand the revolution they were causing, they left art critics puzzling even more. "Is it or is it not art?" was their typical shallow question, and: "Who (or what!) is the creator? The human, the computer, or the drawing automaton?" The simplest of those first creations were governed by polygons. Polygons became the signature of earliest algorithmic art. This is why I tell their story.

In mathematics, a *polygon* is a sequence of points (in the simplest case, in the plane). Polygons also exist in spaces of higher dimensions. As a sequence of points, the polygon is a purely mental construct. In particular and against common belief, you cannot *see* the polygon. As a polygon, it is invisible. It shares this fate with all of geometry. This is so because the objects of geometry—points, lines, planes—are pure. You describe them in formulae, and you prove theorems about them.

I cannot avoid writing down how a point, a straight line, and a plane are given explicitly. This must be done to provide a basis for the effort of an artist moving into this field. So the point in three-dimensional space is an unrestricted triple of coordinates, $P = (x, y, z)$. The straight line is constructed from two points, say P_1 and P_2, by use of one parameter, call it t. The values of t are real numbers, particularly those between 0 and 1. The parameter acts like a coordinate along the straight line. Thus, we can describe each individual point along the line by the formula

$$P(t) = P_1 + t(P_2 - P_1). \tag{3.1}$$

Finally, the points of a plane are determined from three given points by use of two parameters:

$$P(u, v) = uP_1 + vP_2 + (1 - u - v)P_3. \tag{3.2}$$

We need two parameters because the plane is spreading out into two dimensions whereas the straight line is confined to only one.

Bothering my readers with these formulae has the sole purpose that they should become aware of the different kind of thinking required here. Exactly describing the objects of hopefully ensuing creativity is only the start. It is parallel to the traditional artist's selection of basic materials. But algorithmic treatment must follow, if anything is going to happen (we don't do this here). The parameters u and v, I should add, can be any real numbers. The three points are chosen arbitrarily, but then are fixed (they must not be collinear).

As indicated above, all this is invisible. As humans, however, we want to see and, therefore, we render polygons visibly. When we do so, we interpret the sequence of points that make up the polygon, in an appropriate manner. The usual interpretation is to associate with each point a location (in the plane or in space). Next, draw a straight line from the first to the second point of the polygon, from there to the third point, etc. A closed polygon, in particular, is one whose first and last points coincide.

To draw a straight line, of course, requires that you specify the colour and the width of your drawing instrument, say a pencil. You may also want to vary the strokeweight along the line, or use a pattern as you move on. In short, the geometry and the graphics must be described explicitly and with utmost precision.

You have just learned your first and most important lesson: geometry is invisible, graphics is visible. The entities of geometry are purely mental. They are related to graphic elements. Only in them, they appear. Graphics is the human's consolation for geometry.

Let this be enough for a bit of formal and terminological background. We now turn to the first years of algorithmic art.[3] It is a well-established fact that between

[3] The art we are talking about, in the mid-1960s, was usually called *computer art*. This was certainly an unfortunate choice. It used a machine, i.e. the instrument of the art, to define it. This had not happened before in art history. *Algorithmic art* came much closer to essential features of the aesthetic endeavour. It does so up to this day. Today, the generally accepted term is *digital art*. But the digital principle of coding software is far less important than the algorithmic thinking in this art, at least when we talk about creativity. The way of thinking is the revolutionary and creative change. Algorithmic art is drawing and painting from far away.

1962 and 1964 three mathematicians or engineers, who on their jobs had easy and permanent access to computers, started to use those computers to generate simple drawings by executing algorithms. As it happened, all three had written algorithms to generate drawings and, without knowing of each other, decided to publicly exhibit their drawings in 1965. Those three artists are (below, examples of their works will be discussed):

- *Georg Nees* of Siemens AG, Erlangen, Germany, exhibited in the Aesthetic Seminar, located in rooms of the Studiengalerie of Technische Hochschule Stuttgart, Germany, from 5 to 19 February, 1965. Max Bense, chairing the institute, had invited Nees. A small booklet was published as part of the famous *rot* series for the occasion. It most likely became the first publication ever on visual computer art (Nees and Bense 1965).[4]
- *A. Michael Noll* of Bell Telephone Laboratories, Murray Hill, NJ, USA showed his works at Howard Wise Gallery in New York, NY, from 6 to 24 April, 1965 (together with random dot patterns for experiments on visual perception, by Bela Julesz; the exhibits were mixed with those of a second exhibition).
- *Frieder Nake* from the University of Stuttgart, Germany, displayed his works at Galerie Wendelin Niedlich in Stuttgart, from 5 to 26 November, 1965 (along with Georg Nees' graphics from the first show). Max Bense wrote an introductory essay (but could not come to read it himself).[5]

As it happens, there may have been one or two forgotten shows of similar productions.[6] But these three shows are usually cited as the start of digital art. The public appearance and, thereby, the invitation of critique, is the decisive factor if what you do is to be accepted as art. The artist's creation is one thing, but only a public reaction and critique can evaluate and judge it. The three shows, the authors, and the year define the beginning of algorithmic art.

From the point of view of art history, it may be interesting to observe that conceptual art and video art had their first manifestations around the same time. Op art had existed for some while before concrete and constructive art became influential. The happening—very different in approach—had its first spectacular events in the 1950s,

[4]The booklet, *rot 19*, contains the short essay, *Projekte generativer Ästhetik*, by Max Bense. I consider it to be the manifesto of algorithmic art, although it was not expressly called so. It has been translated into English and published several times. The term *generative aesthetics* was coined here, directly referring to Chomsky's generative grammar. The brochure contains reproductions of some of Nees' graphics, along with his explanations of the code.

[5]Bense's introductory text, in German, was not published. It is now available on the compArt Digital Art database at compart-bremen.de. Concerning the three locations of these 1965 exhibitions, Howard Wise was a well-established New York gallery, dedicated to avant-garde art. Wendelin Niedlich was a bookstore and gallery with a strong influence in the Southwest of Germany. The Studiengalerie was an academic (not commercial) institution dedicated to experimental and concrete art.

[6]Paul Brown recently (2009) discovered that Joan Shogren appears to have displayed computer-generated drawings for the first time on 6 May 1963 at San Jose State University.

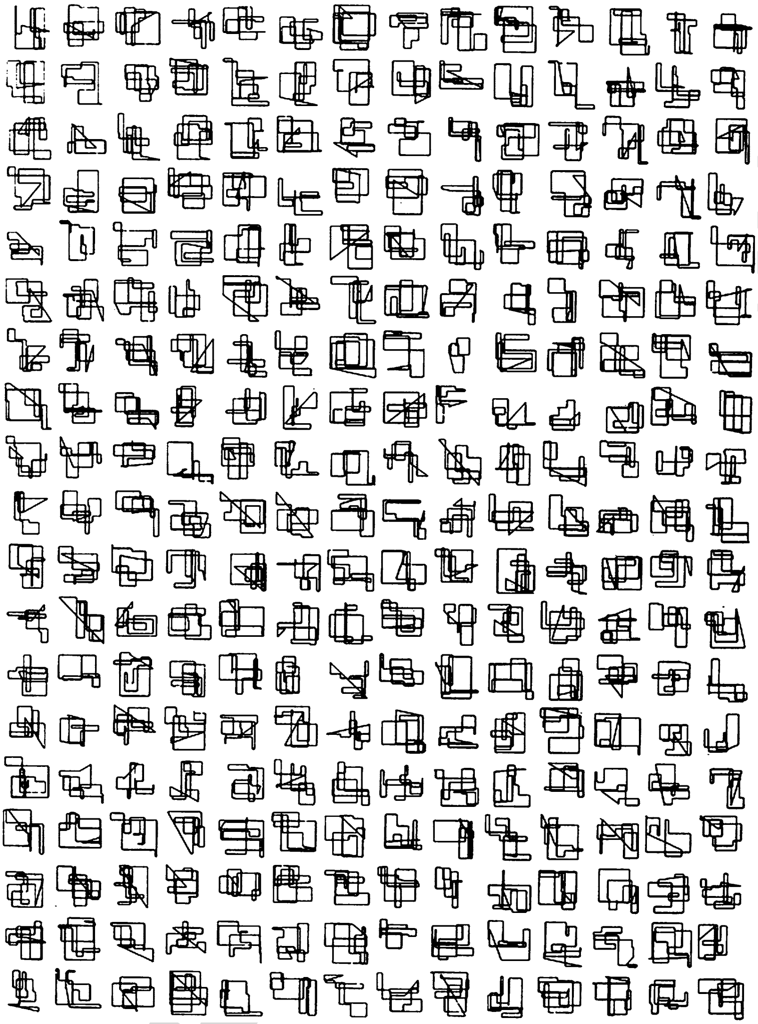

Fig. 3.1 Georg Nees: *23-Ecke*, 1965 (with permission of the artist)

and was continuing them. Pop art was, of course, popular. Serial, permutational, random elements and methods were being explored by artists. Kinetic art and light art were another two orientations of strong technological dependence. Max Bense had chosen the title *Programming the beautiful* (*Programmierung des Schönen*) for the third volume of his *Aesthetica* (Bense 1965), and Karl Gerstner had presented his book *Designing Programs* (*Programme entwerfen*, Gerstner 1963), whose second edition already contained a short section on randomness by computers.

But back to polygons! They appear in the works of the three above mentioned scientists-turned-artists among their very first experiments (Figs. 3.1, 3.2 and 3.3). We will now look at some of their commonalities and differences.

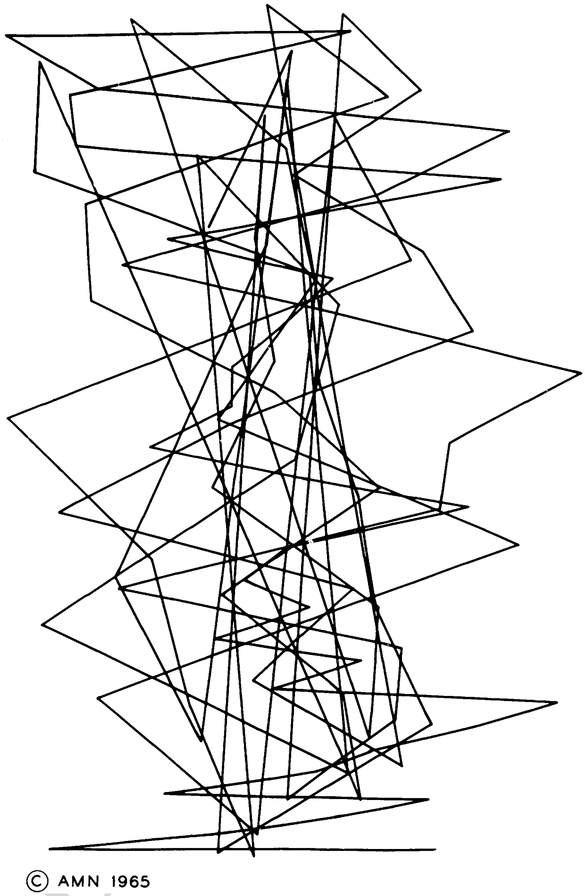

Fig. 3.2 A. Michael Noll: *Gaussian-Quadratic*, 1965 (with permission of the artist)

Assume you have at your disposal a technical device capable of generating drawings. Whatever its mode of operation may be, it is a mechanism whose basic and most remarkable operation creates a straight line-segment between two points. In such a situation, you will be quite content using nothing but straight lines for your aesthetic compositions. What else could you do? In a way, before giving up, you are stuck with the straight line, even if you prefer beautifully swinging curved lines.

At least for a start you will try to use your machine's capability to its very best before you begin thinking about what other and more advanced shapes you may be able to construct out of straight line-segments. Therefore, it was predictable (in retrospect, at least) that Nees, Noll, and Nake would come up with polygonal shapes of one or the other kind.

A first comment on creativity may be in order here. We see, in those artists' activities, the machinic limitations of their early works as well as their creative transcendence. The use of the machine: creative. The first graphic generations: boring. The use of short straight line-segments to draw bending curves: a challenge in creative

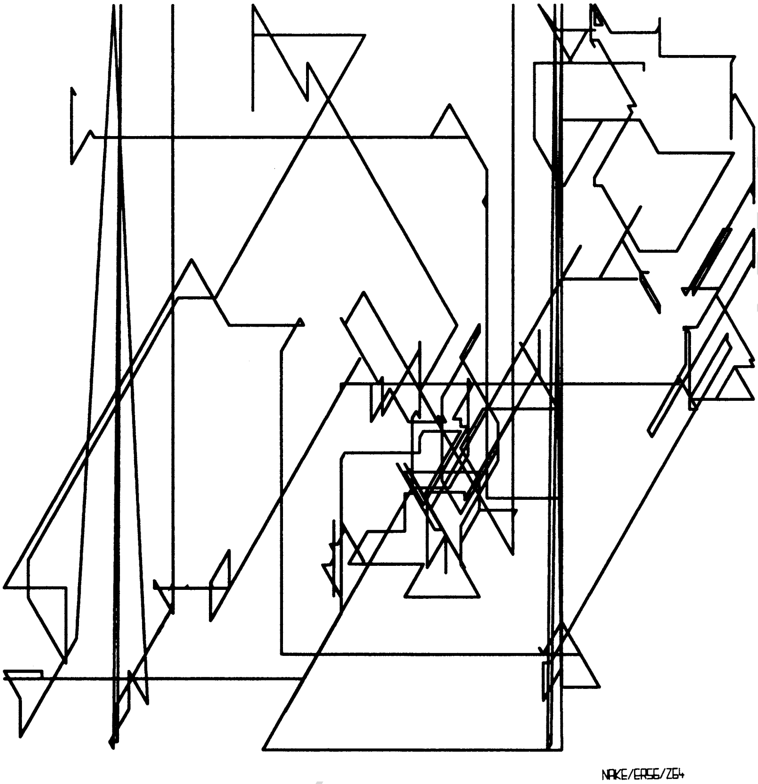

Fig. 3.3 Frieder Nake: *Random Polygon*, 1965

use of the machine. Turning to mathematics for the sake of art: creative, as well as nothing particularly exciting. Throughout the centuries, many have done this. But now the challenge had become to make a machine draw, whose sole purpose was calculation. How to draw when your instrument is not made for drawing?

3.2.1 Georg Nees

Although "polygons" were Nees', Noll's, and Nake's common first interest, their particular designs varied considerably. In six lines of ordinary German text, Nees describes what the machine is supposed to do (Nees and Bense 1965). An English translation of his pseudo-code reads like this:

> Start anywhere inside the figure's given square format, and draw a polygon of 23 straight line segments. Alternate between horizontal and vertical lines of random lengths. Horizontally go either left or right (choose at random), vertically go up or down (also random choice). To finish, connect start and end points by an oblique straight line.

Clearly, before we reach the more involved repetitive design of Fig. 3.1, this basic design must be inserted into an iterative structure of rows and columns. Once a specific row and a specific column have been selected, the empty grid cell located there will be filled by a new realisation of the microstructure just described. As we see from the figure, the composition of this early generative drawing is an invisible grid whose cells contain random 23-gons.

The random elements of Nees' description of the polygon guarantee that, in all likelihood, it will take thousands of years before a polygon will appear equal to, or almost equal to, a previous one. The algorithm creates a rich and complex image, although the underlying operational description appears as almost trivial. The oblique line connecting the first and last points adds a lot to the specific aesthetic quality of the image. It is an aberration from the rectilinear and aligned geometry of the main part of the polygons. This aberration from a standard is of aesthetic value: surprise.

There are $19 \times 14 = 266$ elementary figures arranged into the grid structure. Given the small size of the random shapes, we may, perhaps, not immediately perceive polygons. Some observers may identify the variations on a theme as a design study of a vaguely architectural kind.

The example demonstrates how a trivial composition can lead to a mildly interesting visual appearance not void of aesthetic quality. I postpone the creativity issue until we have studied the other two examples.

When some variable's value is chosen "at random", and this is happening by running a computer program, the concept of randomness must be given an absolutely precise meaning. Nothing on a computer is allowed to remain in a state of vagueness, even if vagueness is the expressed goal. And even if the human observer of the event does not see how he could possibly predict what will happen next, from the computer's position the next step must always be crystal clear. It must be *computable*, or else the program does nothing.

In mathematics, a *random variable* is a variable that takes on its values only according to a probability distribution. The reader no longer familiar with his or her highschool mathematics may recall that a formula like $y = x^2$ will generate the result $y = 16$ if $x = 4$ is given. If randomness plays a role, such a statement could only be made as a probability statement. This means the value of 16 may appear as the result of the computation, but maybe it does not, and the result is, say, 17 or 15.7.

Usually, when in a programming language you have a function that, according to its specification, yields random numbers, these numbers obey a so-called uniform probability distribution. In plain terms, this says that all the possible events of an experiment (like those of throwing dice) appear with the same probability.

But a random variable must not necessarily be uniformly distributed. Probability distributions may be more complex functions than the uniform distribution. In early algorithmic art, even of the random polygon variety, other distributions soon played some role. They simulated (in a certainly naïve way) the artist's intuition. (Does this sound like too bold a statement?)

3.2.2 A. Michael Noll

A. Michael Noll's "Gaussian-Quadratic" graphic makes use, in one direction (the horizontal, viz. Fig. 3.2), of the Gaussian distribution. The coordinates of vertices in the horizontal x-direction are chosen according to a Gaussian distribution, the most important alternative to the uniform distribution. The co-ordinates of vertices in vertical direction are calculated in a deterministic way (their values increase quadratically).

Whereas Nees' design follows a definite, if simple, compositional rule, Noll's is really basic: one polygon whose points are determined according to two distributions. It is not unfair to say that this is a simple visualisation of a simple mathematical process.

3.2.3 Frieder Nake

The same is true of Nake's polygon (Fig. 3.3). The algorithmic principle behind the visual rendition is exactly the same as that of Fig. 3.2: repeatedly choose an x- and a y-coordinate, applying distribution functions F_x and F_y, and draw a straight line from the previous point to the new point (x, y); let then (x, y) take on the role of the previous point for the next iteration.

In this formulation, F_x and F_y stand for functional parameters that must be provided by the artist when his intention is to realise an image by executing the algorithm.[7] Some experience, intuition, or creativity—whatever you prefer—flows into this choice.

The visual appearance of Nake's polygon may look more complex, a bit more like a composition. The fact that it owes its look to the simple structure of one polygon, does not show explicitly. At least, it seems to be difficult to visually follow the one continuous line that constitutes the entire drawing. However, we can clearly discover the solitary line, when we read the algorithm. The description of the simple drawing contains more (or other) facts than we see. So the algorithmic structure may disappear behind the visual appearance even in such a trivial case. Algorithmic simplicity (happening at the *subface* of the image, its invisible side) may generate visual complexity (visible *surface* of the image). If this is already happening in such trivial situations, how much more should we expect a non-transparent relation between simplicity (algorithmic) and complexity (visual) in cases of greater algorithmic effort?[8]

---

[7] Only a few steps must be added to complete the algorithm: a first point must be chosen, the total number of points for the polygon must be selected, the size of the drawing area is required, and the drawing instrument must be defined (colour, stroke weight).

[8] The digital image, in my view, exists as a double. I call them the *subface* and the *surface*. They always come together, you cannot have one without the other. The subface is the computer's view, and since the computer cannot see, it is invisible, but computable. The surface is the observer's view. It is visible to us.

This first result occurred at the very beginning of computer art. It is, of course, of no surprise to any graphic artist. He has experienced the same in his daily work: with simple technical means he achieves complex aesthetic results. The rediscovery of such a generative principle in the domain of algorithmic art is remarkable only insofar as it holds.

However, concerning the issue of creativity, some observers of early algorithmic experiments in the visual domain immediately started asking where the "generative power" (call it "creativity", if you like) was located. Was it in the human, in the program, or even in the drawing mechanism? I have never understood the rationale behind this question: human or machine—who or which one is the creator? But there are those who love this question.

If you believe in the possibility of answering such a question, the answer depends on how we first define "creative activity". But such a hope usually causes us to define terms in a way that the answer turns out to be what we want it to be. Not an interesting discussion.

When Georg Nees had his first show in February 1965, a number of artists had come to the opening from the Stuttgart Academy of Fine Art. Max Bense read his text on projects of generative aesthetics, before Nees briefly talked about technical matters of the design of his drawings and their implementation. As he finished, one of the artists got up and asked: "Very fine and interesting, indeed. But here is my question. You seem to be convinced that this is only the beginning of things to come, and those things will be reaching way beyond what your machine is already now capable of doing. So tell me: will you be able to raise your computer to the point where it can simulate my personal way of painting?"

The question appeared a bit as if the artist wanted to give a final blow to the programmer. Nees thought about his answer for a short moment. Then he said: "Sure, I will be able to do this. Under one condition, however: you must first explicitly tell me how you paint." (The artists appeared as if they did not understand the subtlety and grandeur, really: the dialectics of this answer. Without saying anything more, they left the room under noisy protest.)

When Nietzsche, as one of the earliest authors, experienced the typewriter as a writing device, he remarked that our tools participate in the writing of our ideas.[9] I read this in two ways. First, in a literal sense. Using a pencil or a typewriter in the process of making ideas explicit by formulating them in prose and putting this in visible form on paper, obviously turns the pencil or typewriter in my hand into a device without which my efforts would be in vain. This is the trivial view of the tool's involvement in the process of writing.

The non-trivial view is the observation that my thinking and attitude towards the writing process and, therefore, the content of my writing is influenced by the tool I'm using. My writing changes not only mechanically, but also mentally, depending on my use of tools. It still remains *my* writing. The typewriter doesn't write anything.

[9] Friedrich Kittler quotes Nietzsche thus: "Unser Schreibzeug arbeitet mit an unseren Gedanken." (Our writing tools participate in the writing of our thoughts.) (Kittler 1985), cf. Sundin (1980).

It is me who writes, even though I write differently when I use a pen than when I use a keyboard.

The computer is *not* a tool, but a machine, and more precisely: an automaton.[10] I can make such a claim only against a position concerning tools and machines and their relation. Both, machines and tools, are instruments that we use in work. They belong to the means of any production. But in the world of the means of production, tools and machines belong to different historic levels of development. Tools appear early, and long before machines. After the machine has arrived, tools are still with us, and some tools are hard to distinguish from machines. Still, to mix the two—as is very popular in the computing field where everything is called a "tool"—amounts to giving up history as an important category for scientific analysis. Here we see how the ideological character of so many aspects of computing presents itself.

Nietzsche's observation, that the tools of writing influence our thoughts, remains true. Using the typewriter, he was no longer forced to form each and every letter's shape. His writing became typing: he moved from the continuous flow of the arm and hand to the discrete hits of the fingers. We discover the digital fighting the analog: for more precision and control, but also for standardisation. Similarly, I give up control over spelling when I use properly equipped software (spell-checker). At the same time, I gain the option of rapid changes of typography and page layout.

If creation is to generate something that was not there before, then it is me who is creative. My creation may dwell on a trivial level. The more trivial, the easier it may be to transfer some of my creative operations onto the computer. It makes a difference to draw a line by hand from here to roughly there on a sheet of paper, as compared to issuing the appropriate command sequence, which I know connects points A and B. My thought must change. From "roughly here and there" to "precisely these coordinates".

My activity changes. From the immediate actor and generator of the line, I transform myself into the mediating specifier of conditions a machine has to obey when it generates the physical line. My part has become "drawing by brain" instead of "drawing by hand". I have removed myself from the immediacy of the material. I have gained a higher level of semioticity.

My brain helps me to precisely describe how to draw a line between any two points, whereas before I always drew just one line. It always was a single and particular line: this line right here. Now it has become: this is how you do it, independent of where you start, and where you end. You don't embark on the adventure of actually and physically drawing one and only one line. You anticipate the drawing of any line.

I am the creative one, and I remain the creator. However, the stuff of my creation has changed from material to semiotic, from particular to general, from single case to all cases. As a consequence, my thinking changes. I use the computer to execute a program. This is an enormous shift from the embodied action of moving the pencil. Different skills are needed, different thinking is required and enforced. Those who

[10]Cf. Sundin (1980).

claim the computer has become creative (if they do exist) have views that appear rather traditional. They do not see the dramatic change in artistic creation from material to sign, from mechanics to deliberate semiotics.

What is so dramatic about this transformation? Signs do not exist in the world. Other than things, signs require the presence of human beings to exist. Signs are established as relations between other entities, be they physical or mental. In order to appear, the sign must be perceived. In order to be perceivable, it must come in physical form. That form, however, necessary as it is, is not the most important correlate of the sign. Perceivable physical form is the necessary condition of the sign; the full sign, however, must be constituted by a cognitive act.

Semiotics is the study of sign processes in all their multitudes and manifestations. One basic question of semiotics is: how is communication possible? Semiotic answers to this question are descriptive, not explanatory.

3.3 The Second Narration: On Three Artists

It has often been pointed out that computer art originates in the work of mathematicians and engineers. Usually, this is uttered explicitly or implicitly with an undertone on "only mathematicians and engineers".

The observation is true. Mathematicians and engineers are the pioneers of algorithmic art, but what is the significance of this observation? Is it important? What is the relevance of the "*only* mathematicians" qualification? I have always felt that this observation was irrelevant. It could only be relevant in a sense like: "early computer art is boring; it is certainly not worth being called art; and no wonder it is so boring—since it was not inspired by *real* artists, how could it be exciting"?

Frankly, I felt insulted a bit by the "only mathematicians" statement.[11] It implies a vicious circle. If art is only what artists generate, then how do you become an artist, if you are not born an artist? The only way out of this dilemma is that everyone is, in fact, born an artist (as not only Joseph Beuys has told us). But then the "only mathematicians" statement wouldn't make sense any more.

People generate objects and they design processes. They do not generate art. Art, in my view, is a product of society—a judgement. Without appearing in public and thus without being confronted with a critique of historic and systematic origin, a work remains a work, for good or bad, but it cannot be said to have been included in the broad historic stream of art. Complex processes take place after a person decides to display his or her product in publicly accessible spaces. It is only in the public domain that art can emerge (as a value judgement!). Individuals and institutions in mutual interdependence are part of the processes that may merge to the judgement that a work is assessed and accepted as a work of "art"—often enough, as we all know, sparking even more controversy.

[11] This should read "mathematicians or engineers", but I will stick to the shorter version.

In the course of time, it often happens that an individual person establishes herself or himself stably or almost irrevocably in the hall of art. Then she or he can do whatever they want to do, and still get it accepted as "art". But the principle remains.[12]

The "only mathematician" statement is relevant only insofar as it is interpreted as "unfortunately the pioneers were only mathematicians. Others did not have access to the machines, or did not know how to program. Therefore we got the straight-line quality of early works."

However, if we accept that a work's quality as a work of *art* is judged by society anyhow, the perspective changes. Mathematician or bohemian does not matter then. There cannot be serious doubt that what those pioneering mathematicians did caused a revolution. They separated the generation of a work from its conception. They did this in a technical way. They were interested in the operational, not only mental separation. No wonder that conceptual art was inaugurated at around the same time. The difference between conceptual and computational art may be seen in the computable concepts that the computer people were creating.

However, when viewed from a greater distance, the difference between conceptual artists and computational artists is not all that great. Both share the utmost interest in the idea (as opposed to the material), and Sol LeWitt was most outspoken on this. The early discourse of algorithmic art was also rich about the immaterial character of software. Immaterial as software may be, it does not make sense without being executed by a machine. A traditionally described concept does not have such a surge to execution.[13]

The pioneers from mathematics showed the world that a new principle had arrived in society: the algorithmic principle! No others could have done this, certainly not artists. It had to be done by mathematicians, if it was to be done at all. The parlance of "only mathematicians" points back to the speaker more than to the mathematician.

Trivial to note is that creative work in art, design, or any other field, depends on ideas on one hand, and skills on the other. At times it happens that someone has a great idea but just no way to realise it. He or she depends on others to do that. Pushing things a bit to the extreme, the mathematics pioneers of digital art may not have had great ideas, but they knew how to realise them.

[12]Marcel Duchamp was the first to talk and write about this: "All in all, the creative act is not performed by the artist alone; the spectator brings the work in contact with the external world by deciphering and interpreting its inner qualification and thus adds his contribution to the creative act. This becomes even more obvious when posterity gives a final verdict and sometimes rehabilitates forgotten artists." (Duchamp 1959). This position implies that a work may be considered a work of art for some while, but disappear from this stage some time later, a process that has often happened in history. It also implies that a person may be considered a great artist only after his or her death. That has happened, too.

[13]It is a simplification to concentrate the argument on conceptual vs. algorithmic artists. There have been other directions for artistic experiments, in particular during the 1960s. They needed a lot of technical skill and constructive intelligence or creativity. Recall op art, kinetic art, and more. Everything that humans eventually transfer to a machine has a number of precursors.

On the other hand, artists may have had great ideas and lots of good taste and style, but no way of putting that into existence. So who is to be blamed first? Obviously, both had to acquire new and greater knowledge, skills, and feelings. They had to learn from each other. Turning the argument around, we come up with "unfortunately, some were only artists and therefore had no idea how to do it." Doesn't this sound stupid? It sounds as stupid the other way around.

So let us take a look at what happened when artists wanted, and actually managed, to get access to computers. As examples I have chosen Vera Molnar, Charles Csuri, and Manfred Mohr. Many others could be added. My intent, however, is not to give a complete account, a few cases are enough to make the point.

3.3.1 Vera Molnar

Vera Molnar was born in Hungary in 1924 and lived in Paris. She worked on concrete and constructive art for many years. She tried to introduce randomness into her graphic art. To her great dismay, however, she realised that it is hard for a human to avoid repetition, clusters, trends, patterns. "Real" randomness does not seem to be a human's greatest capability.

So Vera Molnar decided that she needed a machine to do parts of her job. The machine would not be hampered by the human subjectivity that seems to get in the way of a human trying to do something randomly. The kind of machine she needed was a computer that, of course, she had no access to. Vera Molnar felt that systematic as well as hazardous ways of expressing and researching were needed for her often serial and combinatorial art. Since she did not have the machine to help her to do this, she had to build one herself. She did it mentally: "I imagined I had a computer" (Herzogenrath and Nierhoff 2006, p. 14). Her *machine imaginaire* consisted of exactly formulated rules of behaviour. Molnar simulated the machine by strictly doing what she had told the imaginary machine to do.

In 1968, Vera Molnar finally gained access to a computer at the Research Centre of the computer manufacturer, Bull. She learned programming in Fortran and Basic, but also had people to help her. She did not intend to become an independent programmer. Her interests were different. For her, the slogan of the computer as a tool appears to be justified best. She allowed herself to change the algorithmic works by hand. She made the computer do what she did not want to do herself, or what she thought the machine was doing more precisely.[14]

Figure 3.4 (left)[15] shows one of her early computer works. She had previously used repertoires of short strokes in vertical, horizontal, or oblique directions, sim-

[14]The catalogue (Herzogenrath and Nierhoff 2006) contains a list of the hardware Vera Molnar has used since 1968. It also presents a thorough analysis of her artistic development. The catalogue appeared when Molnar became the first recipient of the *d.velop digital art award.* A great source for Molnar's earlier work is Hollinger (1999).

[15]This figure consists of two parts: a very early work, and a much later one by the same artist. The latter one is given without any comment to show an aspect of the artist's development.

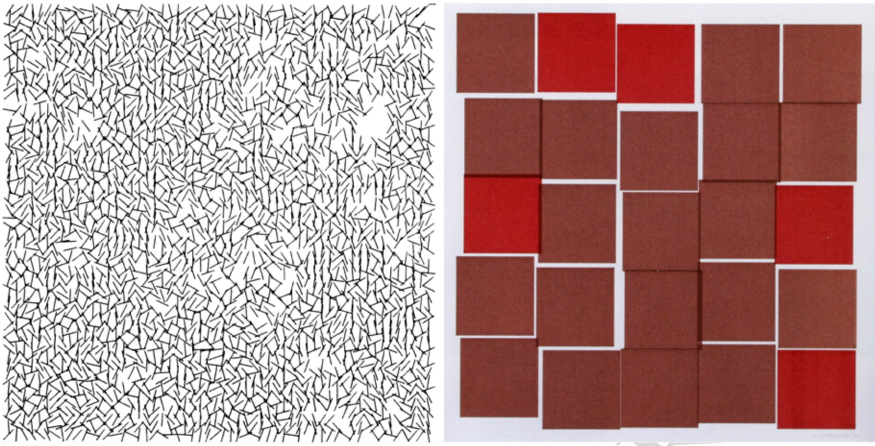

Fig. 3.4 Vera Molnar. *Left*: *Interruptions*, 1968/69. *Right*: *25 Squares*, 1991 (with permission of the artist)

ilar in style to what many of the concrete artists had also done. The switchover to the computer gave her the opportunity to do more systematic research. ("Visual research" was a term of the time. The avantgarde loved it as a wonderful shield against the permanent question of "art". Josef Albers and others from the Bauhaus were early users of the word.)

The *Interruptions* of Fig. 3.4 happen in the open spaces of a square area that is densely covered by oblique strokes. They build a complex pattern, a texture whose algorithmic generation, simple as it must be, is not easy to identify. The open areas appear as surprise. The great experiment experienced by pioneers of the mid-1960s shows in Molnar's piece: what will happen visually if I force the computer to obey a simple set of rules that I invent? How much complexity can I generate out of almost trivial descriptions?

3.3.2 Charles Csuri

Our second artist who took to the computer is Charles Csuri. He is a counter example to the "only mathematicians" predicament. Among the few professional artists who became early computer users, Csuri was probably the first. He had come to Ohio State University in Columbus from the New York art scene. His entry into the computer art world was marked by a series of exceptional pieces, among them *Sine Curve Man* (Fig. 3.5, left), *Random War*, and the short animated film *Hummingbird* (for more on Csuri and his art, see Glowski 2006).

Sine Curve Man won him the first prize of the *Computer Art Contest* in 1967. Ed Berkeley's magazine, *Computers and Automation* (later renamed to *Computers and People*), had started this yearly contest. It was won in 1965 by A. Michael Noll,

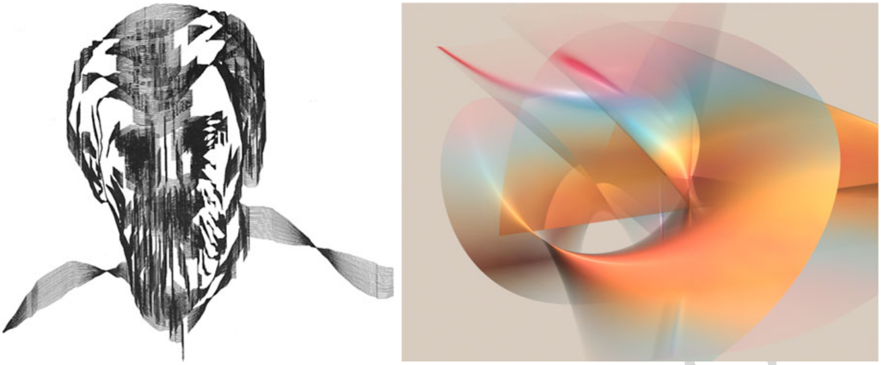

Fig. 3.5 Charles Csuri. *Left*: *Sine Curve Man*, 1967. *Right*: *yuck 4x3*, 1991 (with permission of the artist)

1966 by Frieder Nake, and then by Csuri, an educated artist for the first time. This award, by the way, never gained high esteem. It took many more years, until 1987, when the now extremely prestigious *Prix Ars Electronica* was awarded for the first time.

For his first programming tasks, Csuri was assisted by programmer James Shaffer. Similar to Vera Molnar, we see that the skill of programming may at the beginning constitute a hurdle that is not trivial to master. If time plays a role, an artist willing to use the computer, but still unable to do it all by himself, has almost no choice but to rely on friendly cooperation. Such cooperation may create friction with all its negative effects. As long as the technical task itself does *not* require cooperation, it is better to acquire the new technical skill. After all, there is no art without skillful work, and a steadily improved command of technical skills is a necessary condition for the artist. Why should this be different when the skill is not the immediate transformation of a corporeal material by hand, but instead the description only of relations and conditions, of options and choices of signs?

Csuri's career went up steeply. Not only did he become the head of an academic institute but even an entrepreneur. At the time of a first rush for large and leading places in computer animation, when this required supercomputers of the highest technological class and huge amounts of money, he headed the commercial Cranston Csuri Productions company as well as the academic Advanced Computing Center for the Arts and Design, both at Columbus, Ohio. In the year 2006, Csuri was honoured by a great retrospective show at the ACM SIGGRAPH yearly conference.

Sine Curve Man is an innovation to computer art of the first years in two respects: its subject matter is figurative, and it uses deterministic mathematical techniques rather than probabilistic. There is a definite artistic touch to the visual appearance of the graphic (Fig. 3.5), quite different from the usual series of precise geometric curves that many believe computer art is (or was) about.

The attraction of *Sine Curve Man* has roots in the graphic distortions of the (old?) man's face. Standard mathematics can be used for the construction. A lay person may, however, not be familiar with such methods. Along the curves of an original drawing, a series of points are marked. The curves may, perhaps, have been extracted from a photograph. The points become the fixed points of interpolations by sums of *sine* functions. This calculation, purely mathematical as it is, and without any intuitive additions triggered by the immediate impression of a seemingly half-finished drawing, is an exceptional case of the new element in digital art.

This element is the dialectics of aesthetics and algorithmics. *Sine Curve Man* may cause in an observer the impression that something technical is going on. But this is probably not the most important aspect. More interesting is the visual (i.e. aesthetic) sensation. The distortions this man has suffered are what attracts us. We are almost forced to explore this face, perhaps because we want to read the curves as such. But they do not allow us to do this. Therefore, our attention cannot rest with the mathematics. Dialectics happens, as well as semioses (sign processes): jumping back and forth between semantics and syntactics.

3.3.3 Manfred Mohr

Manfred Mohr is a decade younger than the first two artists. They belong to the first who were accepted by the world of art despite their use of computers. Do they owe anything to computers? Hard to say. An art historian or critic will certainly react differently if he doesn't see an easel in the artist's studio, but a computer instead.

The artist doesn't owe much to a computer. He has decided to use it, whatever the reason may have been. If to anything, he owes to the programs he is using or has written himself. With those programs, he calls upon work formerly spent that he now is about to set in action again. The program appears as canned labour ready to be resuscitated.

The relation between artist and computer is, at times, romanticised as if it were similar to the close relation between the graphic artist and her printer (a human being). The printer takes pride in getting the best quality out of the artist's design. The printing job takes on artistic quality itself. The computer, to the contrary, is only executing a computable function. It should be clear, that the two cases are as different as they could ever be.

If we characterise Vera Molnar, in one word, as the grand old lady of algorithmic art, and Charles Csuri as the great entrepreneur and mover, Manfred Mohr would appear as the strictest and strongest creator of a style in algorithmic art. The story says that his young and exciting years of searching for his place in art history were filled with jamming the saxophone, hanging out in Spain and France, and with hard edge constructivist paintings. Precision and rationality became and remained his values. They find a correspondence and a balancing force in the absolute individual freedom of jazz. Like many of the avant-garde artists in continental Europe during the 1960s, he was influenced by Max Bense's theory and writing on aesthetics, and when he read in a German news magazine (Anon 1965) that computers had appeared in fine art, he knew where he had to turn to.

K.R.H. Sonderborg and the art of *Informel*, Pierre Barbaud and electronic music, Max Bense and his theory of the aesthetic object constitute a triad of influences from

Book ID: 303794_1_En, Date: 2012-07-30, Proof No: 1, UNCORRECTED PROOF

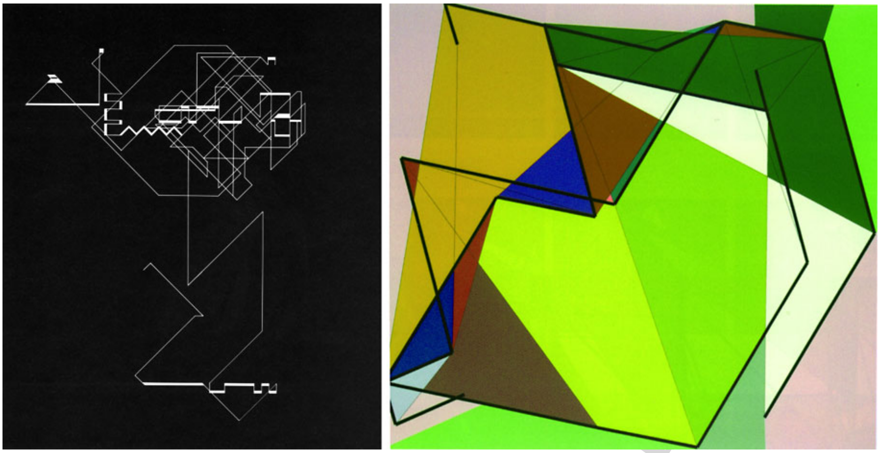

Fig. 3.6 Manfred Mohr. *Left*: *P-18* (*Random Walk*), 1969. *Right*: *P-707-e1* (*space.color*), 1999–2001 (with permission of the artist)

which Mohr's fascinating generative art emerged. From his very first programmed works in 1969 to current days, he has never betrayed his striving for the greatest transparency of his works. Never did he leave any detail of his creations open to hand-waving or to dark murmurs. He discovered the algorithmic description of the generative process as the new creation. The simplest elements can become the material for the most complex visual events.

After about four years of algorithmic experiments with various forms and relations, Manfred Mohr, in 1973, decided to use the cube as the source of external inspiration. He has continued exploring it ever since. There are probably only a few living persons who have celebrated and used the cube more than him (for further information see Keiner et al. 1994, Herzogenrath et al. 2007).

Figure 3.6 shows one event in the six-dimensional hypercube (right), and one of the earliest generative graphics of Mohr's career (left).

When we see a work by Mohr, we immediately become aware of the extraordinary aesthetic quality of his work. His decisions are always strong and secure. The random polygon of Fig. 3.6 is superior to most, if not all, of the others one could see in the five years before. The events of the heavier white lines add an enormous visual quality to the drawing, achieved in such strength here for the first time.

The decision, in 1973, to explore the three-dimensional cube as a source for aesthetic objects and processes, put Manfred Mohr in a direct line with all those artists who, at least for some part of their artistic career, have explored one and the same topic over and over again. It should be emphasised, however, that his interest in the cube and the hypercube[16] does not signify any pedagogical motif. He does not intend to explain anything about spaces of higher dimensions, nor does he visualise

[16] The hypercube is analogous to a three-dimensional cube in four or more dimensions. It is recursively defined as an intricate structure of cubes.

cubes in six or eleven dimensions. He takes those mental creatures as the rational starting points for his visual creation. The hypercube is only instrumental in Mohr's creative work; it is not the subject matter.

The cube in four or more dimensions is a purely mental product. We can clearly *think* the hypercube. But we cannot visualise it. We may take the hypercube as the source of visual aesthetic events (and Mohr does it). But we cannot show it in a literal sense of the word. Manfred Mohr's mental hikes in high dimensions are his inspiration for algorithmic concrete images. For these creations, he needs the computer. He needs it even more when he allows for animation.

Manfred Mohr's work stands out so dramatically because it cannot be done without algorithms. It is the most radical realisation of Paul Klee's announcement: *we don't show the visible, we make visible.* The image is a visible signal. What it shows is itself. It has a source elsewhere. But the source is not shown. It is the only reason for something visible.

Creativity? Yes, of course, piles of. Supported by computer? Yes, of course, in the trivial sense that this medium is needed for the activity of realising something the artist is thinking of. In Manfred Mohr's work (and that of a few others whose number is increasing) generative art has actually arrived. The actuality of his work is its virtuality.

3.4 The Third Narration: On Two Programs

Computer programs are, first of all, *texts*. The text describes a complex activity. The activity is usually of human origin. It has before existed as an activity carried out by humans in many different forms. When it becomes the source of an algorithmic description, it may gradually disappear as a human activity, until in the end, the computer's (or rather the program's) action appears as the first and more important than the human activities that may still be needed to keep the computer running: human-supported algorithmic work.

The activity described by a computer program as a text may be almost trivial, or it may be extremely complex. It may be as trivial as an approximate calculation of the *sine* function for a given argument. Or it may be as complex as calculating the weather forecast for the area of France by taking into account all available atmospheric measurements collected around the world.

The art of writing computer programs has become a skill of utmost creativity, intuition, constructive precision, and secrets of the trade. Donald Knuth's marvellous series of books, *The Art of Computer Programming*, is the best proof of this (Knuth 1968). These books are one of the greatest attempts to give an in-depth survey of the entire field of computing. It is almost impossible to completely grasp this field in totality, or even to finish writing the series of books. Knuth is attempting to do just this.

Computer programs have been characterised metaphorically as tools, as media, or as automata. How can a program be an automaton if it is, as I have claimed, a text? The answer is in the observation that the computer is a *semiotic machine* (Nadin 2011, Nöth 2002, Nake 2009).

The computer is seen by these authors as a semiotic machine, because the stuff it processes is of a semiotic nature. When the computer is running, i.e. when it is working as a machine, it is executing a program. It is doing this under the control of an operating system. The operating system is itself a program. The program, that the computer is executing, takes data and transforms it into new data. All these creatures—the operating system, the active program, and data—are themselves of semiotic nature. This chapter is not the place to go deeper into the semiotic nature of all entities on a computer.[17] So let us proceed from this basic assumption.

The assumption becomes obvious when we take a look at a program as a text. Leaving aside all detail, programming starts from a more or less precise specification of what a program should be doing. Then there is the effort of a group of programmers developing the program. Their effort materialises in a rich mixture of activities. Among these, the *writing of code* is central. All other kinds of activities eventually collapse into the writing of code.

The finished program, which is nothing but the code for the requested function, appears as a text. During his process of writing, the programmer must read the text over and over again. And here is the realisation: the computer is also reading the text! The kind of text that we call "computer program" constitutes a totally new kind of poetry. The poetics of this poetry reside in the fact that it is written for two different readers: one of them human, the other machine.

Their fantastic semiotic capabilities single out humans from the animal kingdom. Likewise, the computer is a special machine because of its fantastic semiotic capabilities. Semiotic animal and semiotic machine meet in reading the text that is usually called a *program*.

Now, reading is essentially interpreting. The human writer of the program materialises in it the specification of some complex activity. During the process of his writing, he is constantly re-reading his product as he has so far written it. He is convinced of the program's "correctness". It is correct as long as it does what it is supposed to do. However, how may a text be actively *doing* anything?

The text can do something only if the computer is also reading it. The reading, and therefore interpreting, of the program by the computer effectively transforms the text into a machine. The computer, when reading the program text (and therefore: interpreting it), cannot but execute it. Without any choice, reading, interpreting, and executing the text are one and the same for the computer. The program as a text is interesting for the human only insofar as the computer is brought to execute it. During execution, the program reveals its double character as text-and-machine, both at the same time. So programs are executable texts. They are texts as machine, and machine as text.

After this general but also concrete remark about what is new in postmodern times, we take a look at two specific and ambitious, albeit very different programs.

[17] A book is in preparation that takes a fundamental approach to this topic: P.B. Andersen & F. Nake, *Computers and signs. Prolegomena to a semiotic foundation of computing*.

We don't look at their actual code because this is not necessary for our discussion of creativity in early computer art. Harold Cohen's famous *AARON* started its astonishing career in 1973, and continued to be developed for decades. Frieder Nake's *Generative Aesthetics I* was written, completed, then discarded in the course of one year, 1968/69.

3.4.1 Harold Cohen: AARON

AARON is a rule-based system, an enormous expert system, one of the very few expert systems that ever made it to their productive phase (McCorduck 1990). In the end it consisted of so many rules that its sole creator, Cohen, was no longer sure if he was still capable of understanding well enough their mutual dependencies.

Everything on a computer must be rule-based. A rule is a descriptive element of the structure: `if` C `then` A, where C is a condition (in the logical sense of "condition"), and A is an action. In the world of computing, a formal definition must state precisely what is accepted as a C, and what is accepted as an A. In colloquial terms, an example could be: `if (figure ahead) then (turn left or right)`. Of course, the notions of "figure", "ahead", "turn", "left", "right" must also be described in computable terms, before this can make any sense to a computer.

A rule-based system is a collection of interacting rules. Each rule is constructed as a pair of a condition and an action. The condition must be a description of an event depending on the state (value) of some variables. It must evaluate to one of the truth-values `true` or `false`. If its value is `true`, the action is executed. This requires that its description is also given in computable form. The set of rules making up a rule-based system may be structured into groups. There must be an order according to which rules are tested for applicability. One strategy is to apply the first applicable rule in a given sequence of rules. Another one determines all applicable rules and selects one of them.

Cohen's AARON worked for many years during which it produced a large collection of drawings. They were first in black and white. Later, Cohen coloured them by hand according to his own taste or to codes also determined by AARON. The last stage of AARON relied on a new painting machine. It was constructed such that it could mimic certain painterly ways of applying paint to paper.

During more than three decades, AARON's command of subjects developed from collections of abstract shapes to evocations in the observer of rocks, birds, and plants, and to figures more and more reminiscent of human beings. They gave the impression of a spatial arrangement, although Cohen never really entered into three dimensions. A layered execution of figures was sufficient to generate a low-level of spatial impression.

Around the year 2005, Cohen became somewhat disillusioned with the figural subjects he had gradually programmed AARON to better and better create. When he started using computers and writing programs in the early 1970s, he was fascinated

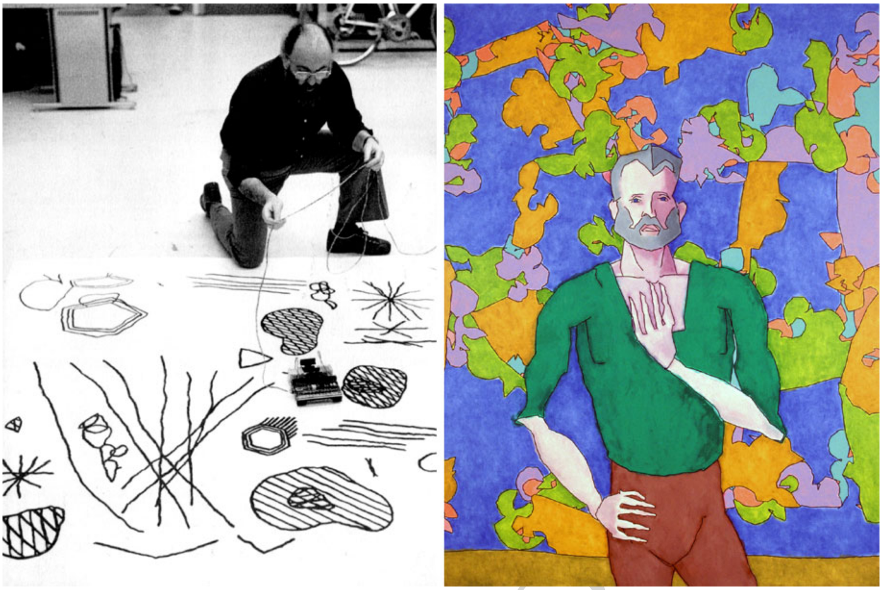

Fig. 3.7 Harold Cohen. *Left*: Early drawing by *AARON*, with the artist. *Right*: Drawing by *AARON*, 1992 (with permission of the artist)

by the problem of representation. His question then was: just how much, or little, does it take before a human observer securely recognises a set of lines and colours as a figure or pattern of something? How could a painting paint itself? (Cohen 2007).

But Harold Cohen has now stopped following this path any further. He achieved more than anyone else in the world in terms of creating autonomous rule-based art systems. He did not give up this general goal. He decided to return to pure form and colour as the subject matter of his autonomous rule-based system.

For a computer scientist, there is no deep difference between an algorithm and a rule-based system. As Cohen (2007) writes, it took him a while to understand this. The difference is one of approach, not of the results. Different approaches may still possess the same expressive power. As Cohen is now approaching colour again in an explicitly algorithmic manner, he has shifted his view closer to the computer scientist's but without negating his deep insight into the qualities of colour as an artist.

This is marvellous. After a long and exciting journey, it sheds light on the alleged difference between two views of the world. In one person's great work, in his immediate activity, experience, and knowledge, the gap between the "two cultures" of C.P. Snow fades. It fades in the medium of the creative activity of one person, not in the complex management of interdisciplinary groups and institutes. The book must still be written that analyses the Cohen decades of algorithmic art from the perspective of art history.

Cohen's journey stands out as a never again to be achieved adventure. He has always been the lonely adventurer. His position is unique and singular. Artificial

Intelligence people have liked him. His experience and knowledge of rule-based systems must be among the most advanced in the world. But he was brave enough to see that in art history he had reached a dead-end. Observers have speculated about when would AARON not only be Cohen's favourite artist, but also its own and best critic. Art cannot be art without critique. As exciting as AARON's works may be, they were slowly losing their aesthetic appeal, and were approaching the only evaluation: oh, would you believe, this was done by computer? The dead-end.

Harold Cohen himself sees the situation with a bit more skepticism. He writes:

> It would be nice if AARON could tell me which of them [its products] it thinks I should print, but it can't. It would be nice if it could figure out the implications of what it does so well and so reliably, and move on to new definitions, new art. But it can't. Do those things indicate that AARON has reached an absolute limit on what computers can do? I doubt it. They are things on my *can't-do-that* list. . . (Cohen 2007).

The *can't-do-that* list contains statements about what the computer can and what it cannot do. During his life, Cohen has experienced how items had to be removed from the list. Every activity that is computable must be taken from the list. There are activities that are not computable. However, the statement that something cannot be done by computer, i.e. is not computable, urges creative people to change the non-computable activity into a computable one. Whenever this is achieved after great hardship, we don't usually realise that a new activity, a computable one, has been created with the goal in mind to replace the old and non-computable.

There was a time, when Cohen was said to be on his way to becoming the first artist of whom there would still be new works in shows after his death. He himself had said so, jokingly with a glass of cognac in hand. He had gone so far that such a thought was no longer fascinating. The Cohen manifesto of algorithmic art has reached its prediction.

But think about the controversial prediction once more. If true, would it not be proof of the computer's independent creativity? Clearly, Cohen wrote AARON, the program, the text, the machine, the text-became-machine. This was *his*, Cohen's creative work. But AARON was independent enough to then get rid of Cohen, and create art all by itself. How about this?

In a trivial sense, AARON is creative, but this creativity is a pseudo-creativity. It is confined to the rules and their certainly wide spectrum of possibilities. AARON will forever remain a technical system. Even if that system contained some meta-rules capable of changing other rules, and meta-meta-rules altering the meta-rules on the lower level, there would always be an explicit end. AARON would not be capable of leaving its own confines. It cannot cross borders.

Cohen's creativity, in comparison, stands out differently. Humans can always cross borders. A revolution has happened in the art world when the mathematicians demonstrated to the artists that the individual work was no longer the centre of aesthetic interest. This centre had shifted to descriptions of processes. The individual work had given way to the *class of works*. Infinite sets had become interesting, the individual work was reduced to a by-product of the class. It has now become an instance only, an index of the class it belongs to.

No doubt, we need the instance. We want to literally see something of the class. Therefore, we keep an interest in the individual work. We cannot see the entire class. It has become the most interesting, and it has become invisible. It can only be thought.

I am often confronted with an argument of the following kind. A program is not embedded into anything like a social and critical system, and clearly, without a critical component, it cannot leave borders behind. So wait, the argument says, until programs are embedded the proper way.

But computers and programs don't even have bodies. How then should they be able to be embedded in such critical and social systems? Purpose and interest are just not their thing. Don't you, my dear friends, see the blatant difference between yourself and your program, between you and the machine?

Joseph Weizenbaum dedicated much of his life to convincing others of this fundamental difference. It seems to be very tough for some of us to accept that we are not like machines and, therefore, they are not like us.

3.4.2 Frieder Nake: Generative Aesthetics I

A class of objects can never itself, as a class, appear physically. In other words, it cannot be perceived sensually. It is a mental construct: the description of processes and objects. The work of art has moved from the world of corporeality to the world of options and possibilities. Reality now exists in two modes, as actuality and virtuality.

AARON's generative approach is activity-oriented. The program controls a drawing or painting tool whose movements generate, on paper or canvas, visible traces for us to see. The program *Generative Aesthetics I*, however, is algorithm-oriented. It starts from a set of data, and tries to construct an image satisfying conditions that are described in the data.

You may find details of the program in Nake (1974, pp. 262–277). The goal of the program was derived from the theory of information aesthetics. This theory starts by considering a visual artefact as a sign. The sign is really a *supersign* because it is usually realised as a structure of signs.

The theory assumes that there is a repertoire of elementary or primitive signs. Call those primitive signs: $s_1, s_2, \ldots, s_r$. They must be perceivable as individual units. Therefore, they can be counted, and relative frequencies of their occurrence can be established. Call those frequencies, $f_1, f_2, \ldots, f_r$.

In information aesthetics, a schema of the signs with their associated relative frequencies is called a *sign schema*. It is a purely statistical description of a class of images. All those images belong to the class that use the same signs (think of colours) with the same frequencies.

In Shannon's information theory, the statistical measure of information in a message is defined as

$$H = -\sum_{i=1}^{r} p_i \log p_i . \tag{3.3}$$

The assumption for the derivation of this formula in Shannon and Weaver (1963) is that all the p_i are probabilities. They determine the statistical properties of a source sending out messages that are constructed according to the probabilities of the source.

This explanation may not mean much to the reader. For one, information theory is no longer popular outside of certain technical contexts. Moreover, it was overestimated in the days when the world was hoping for a great unifying theory. The measure H gives an indication of what we learn when one specific event (out of a set of possible events) has occurred, and we know what the other possible events could be.

Take as an example the throwing of dice in a typical board game. As we know, there are six possible events, which we can identify by the numbers 1, 2, 3, 4, 5, and 6. Each one of the six events occurs with the same probability, i.e. 1/6. Using Shannon's formula for the information content of the source "dice", we get

$$H = -\log(1/6) = -(\log 1 - \log 6) = \log 6 \approx 2.6 \tag{3.4}$$

(the logarithm must be taken to the base of 2). The result is measured in bits and must be interpreted thus: when one of the possible results of the throw has appeared, we gain between two and three bits of information. This, in turn, says that between two and three decisions of a "yes or no" nature have been taken. The Shannon measure of information is a measure of the uncertainty that has disappeared when and because the event has occurred.

Information aesthetics, founded by Max Bense and Abraham A. Moles (Bense 1965, Moles 1968) and further developed in more detail by others (Gunzenhäuser 1962, Frank 1964), boldly and erroneously ignored the difference between frequency and probability. To repeat, probabilities of a sign schema characterise an ideal source. Frequencies, however, are results of empirical measurement of several, but only finitely many messages or events (images in our case). As such, frequencies are only estimates for probabilities.

Information aesthetics wanted to get away from subjective value judgement. Information aesthetic criteria were to be objective. Aspects of the observer were excluded, at least in Max Bense's approach. Empirical studies from the 1960s and later were, however, not about aesthetic sources, but about individual pieces. In doing so, the difference of theory and practice, of infinite class and individual instance, of probability and frequency, had to be neglected by replacing theoretical probability by observed frequency, thus $p_i = f_i$. This opened up the possibility to measure the object without any observer being present. However, the step also gave up aesthetics as the theory of sensual perception.

Now, the program *Generative Aesthetics I* accepted as input a set of constraints of the following kind. For each sign (think of colour), a measure of surprise and a measure of conspicuity (defined by Frank in 1964) could be constraint to an interval of feasible values. Such requirements defined a set of up to $2r$ constraints. In addition, the aesthetic measure that Gunzenhäuser had defined as an information-theoretic analogue to Birkhoff's famous but questionable measure of "order in complexity" (Birkhoff 1931) could be required to take on a maximum or minimum value, relative to the constraints mentioned before. Requesting a maximum to be the goal of construction put trust on the formal definition of aesthetic measure actually yielding a good or even beautiful solution. Requesting a minimum, to the contrary, did not really trust the formalism.

With such a statement of the problem, we are right into mathematics. The problem turns out to be a non-linear optimisation problem. If a solution is possible, it had to be a discrete probability distribution. This distribution represents all images satisfying the constraints. it was called "the statistical pre-selector," since it was based only on a statistical view of the image. In a second step, a topological pre-selector took the sign schema of the previous step and created the image as a hierarchical structure of colour distribution, according to the probabilities determined before.

The type of structure used for this construction of the image was, in computer science, later called a *quadtree*. A quadtree divides an image into four quadrants of equal size. The generative algorithm distributes the probabilities of the entire image into the four smaller quadrants such that the sum total remains the same. With each quadrant, the procedure is repeated recursively, until a quadrant is covered by one colour only, or its size has reached a minimal length.

Generative Aesthetics I thus bravely started from specifying quantitative criteria that an image was to satisfy. Once the discrete probability distribution was determined as a solution to the set of criteria, an interesting process of many degrees of freedom started to distribute the probabilities into smaller and smaller local areas of the image but such that the global condition was always satisfied. Aesthetics happened generatively and objectively, by running an automaton.

The program was realised in the programming language PL/I with some support from Fortran routines. Its output was trivial but fast. I was working on this project in Toronto in 1968/69. Since no colour plotter was available, I used the line printer as output device. The program's output was a list of measures from information aesthetics plus a coded printout of the generated image. I used printer symbols to encode the colours that were to be used for the image.

This generative process was very fast, which allowed me to run a whole series of experiments. These experiments may constitute the only ones ever carried out in the spirit of generative aesthetics based on the Stuttgart school of information aesthetics. The program was intended to become the base for empirical research into generative aesthetics. Regrettably, this was not realised.

With the help of a group of young artists, I realised by hand only two of the printouts. From a printer's shop we got a set of small pieces of coloured cardboard. They were glued to a panel of size 128 × 128 cm. One of those panels has been lost (Fig. 3.8). The other one is in the collection Etzold at Museum Abteiberg in Mönchengladbach, Germany.

Besides the experience of solving a non-trivial problem in information aesthetics by a program that required heuristics to work, I did this project more like a scientist

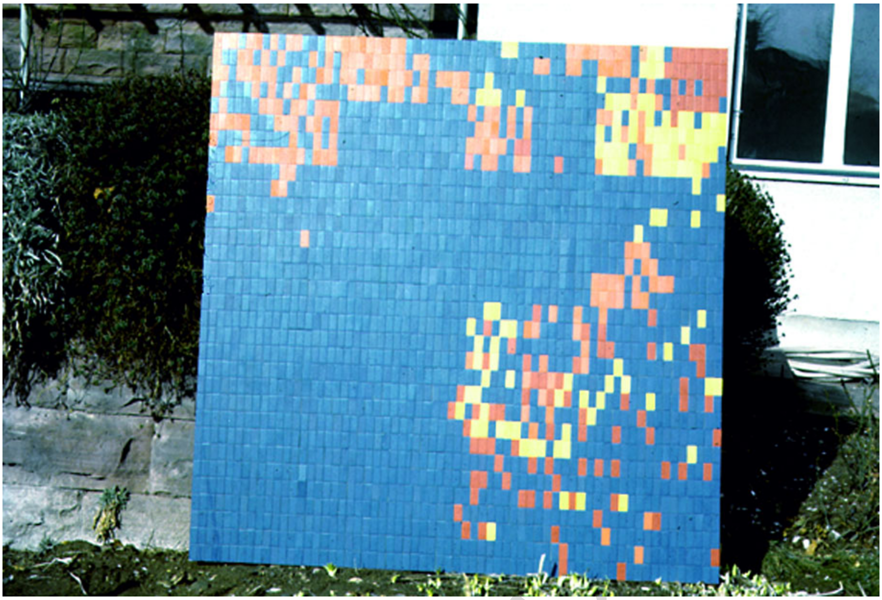

Fig. 3.8 Frieder Nake: *Generative Aesthetics I, experiment 4a.1*, 1969

than an artist. An artist would have organised, well in advance, a production site to transform the large set of the generated raster images into a collection of works. This collection would become the stock of an exhibition at an attractive gallery. A catalogue would have been prepared with the images, along with theoretical and biographical essays. Such an effort to propagate the most advanced and radically rational generative aesthetics would have been worthwhile.

Instead, I think I am justified in concluding that this kind of formally defined generative aesthetics did not work. After all, my experiments with *Generative Aesthetics I* seemed to constitute an empirical proof of this.

Was I premature in drawing the conclusion? It was the time of *Cybernetic Serendipity* in London, *Tendencies 4*, and later *Tendencies 5* in Zagreb. In Europe one could feel some low level, but increasing attention being paid to computer art. A special show was in preparation for the 35th Biennale in Venice, bringing together Russian constructivists, Swiss concrete artists, international computer artists, and kids playing. Wasn't this an indication of computer art being recognised and accepted as art. Premature resignation? Creativity not recognised?

I am not so sure any more. As a testbed for series of controlled experiments on the information-aesthetic measures suggested by other researchers, *Generative Aesthetics I* may, after all, have possessed a potential that was not really fathomed. The number of experiments was too small. They were not designed systematically. Results were not analysed carefully enough. And other researchers had not been invited to use the testbed and run their own, most likely very different, experiments.

It may well be the case that the project should be taken up again, now under more favourable conditions, and different awareness for *generative design*.

3.5 The Fourth and Last Narration: On Creativity

This chapter finds its origins in a Dagstuhl Seminar in the summer of 2009. Schloss Dagstuhl is a beautiful location hidden in the Southwest of Germany, in the province of Saarland. Saarland is one of the European areas where over centuries people from different nations have mixed. After World War II, Saarland belonged to France for some time until a public vote was taken (in 1955) about where people preferred to live, in West Germany or France. Was their majority decision in favour of the German side an act of collective creativity?

Mathematicians in Germany and beyond have had a wonderful institution ever since 1944, the *Mathematical Research Institute of Oberwolfach*. It is located at Oberwolfach in the Black Forest. Mathematicians known internationally for their interest in a specialised field, meet there to pursue their work. They come in international groups, with an open agenda leaving lots of time for spontaneous arrangements of discussion, group work, and presentations.

The German *Gesellschaft für Informatik*, after having established itself as a powerful, active, and growing scientific association in the field of computing, became envious of the mathematicians and decided that they also wanted to have such a well-kept, challenging and inviting site for scientific meetings of high quality. Soon enough, they succeeded. Was this creativity or organisation?

So Dagstuhl became a place for scientists and others, from computer science and neighbouring disciplines, to gather in a beautiful environment and work on issues of a specialised nature. They are supposed to come up with findings that should advance theory and practice of information technology in the broadest sense.

A week at a Dagstuhl seminar is a great chance to engage in something that we usually find no opportunity to do. The topic at this particular occasion was computational creativity—a topic of growing, if only vague interest these days.

Inspired by some of the debates at the seminar, I have tried in this chapter, to recall a few aspects from the early history of algorithmic art as a case from the fringes of computing that we would usually consider a case for creativity. We usually assume that for art to emerge, creativity must happen. So if we see any reason to do research into the relation between creativity and computers, a study of computer art seems to be a promising case.

People are, of course, curious to learn about human creativity in general. A special interest in the impact of computing on creativity must have its roots in the huge machine. As already indicated, I see the computer as a semiotic machine. The subject matter of computational processes must always already belong to the field of semiotics. The subject matter computers work on is of a relational character more than it is "thing-like".

This important characteristic of all computing processes exactly establishes a parallel between computable processes and aesthetic processes. But to the extent that computable processes are carried out by machinery, those processes cannot really reach the pragmatic level of semiosis. Pragmatics is central to purpose. Purpose is what guides humans in their activities. The category of purpose is strongly connected to interest.

I don't think it could be proved—in a rigorous mathematical meaning of the word "prove"—that machines do not (and can never) possess any form of interest and, therefore, cannot follow a purpose. On the other hand, however, I cannot see any common ground between the survival instinct governing us as human beings, and the endless repetition of always the same, if complex, operations the machine is so great and unique at. There is just nothing in the world that indicates the slightest trace of an interest on behalf of the machine. Even without such proof, I do not see any reason or situation where I would use a machine, and this machine developed anything I would be prepared to accept as "interest" and, in consequence, a purposeful activity.

What above I have called an interpretation by the machine is, of course, an interpretation only in a purely formal sense of the word. Clearly, the agent of such interpretation is a machine. As a machine, it is constructed in such a way that it has no freedom of interpretation. The machine's interpretation is, in fact, of the character of a determination: it must determine the meaning of a statement in an operational way. When it does so, it must follow strict procedures hard-wired into it (even if it is a program called a *compiler* that carries out the process of determination). This does not allow a comparison to human interpretation.

3.6 Conclusion

The conclusion of this chapter is utterly simple. Like any other tool, material, or media, computer equipment may play important roles in creative processes. A human's creativity can be enhanced, triggered, or encouraged in many ways. But there is nothing really exciting about such a fact other than that it is rather new, it is extremely exciting, it opens up huge options, and it may trigger super-surprise.

In the year 1747, Julien Offray de La Mettrie published in Leiden, the Netherlands, a short philosophical treatise under the title *L'Homme Machine* (The Human Machine).[18] This is about forty years before the French Revolution, in the time of the Enlightenment. La Mettrie is in trouble because of other provocations he published. His books are burned, and he is living in exile.

In *L'Homme Machine*, La Mettrie undertakes for the first time the radical attempt to reduce the higher human functions to bodily roots, even to simple mechanical explanations. This essay cannot be the place to contribute to the ongoing and, perhaps, never ending discourse about the machinic component in humans. It has been demonstrated often enough that we may describe certain features of human

[18] I only have a German edition. The text can easily be found in libraries.

behaviour in terms of machines. Although this is helpful at times, I do not see any reason to set both equal.

We all seem to have some sort of experienced understanding of construction and intuition. When working and teaching at the Bauhaus, Paul Klee observed and noted that "We construct and construct, but intuition still remains a good thing."[19] We may see construction as that kind of human activity where we are pretty sure of the next steps and procedures. Intuition may be a name for an aspect of human activity about which we are not so sure.

Construction, we may be inclined to say, can systematically be controlled; intuition, in comparison, emerges and happens in uncontrolled ways. Construction stands for the systematic aspects of work we do; intuition for the immediate, non-considerate, and spontaneous. Both are important and necessary for creation. If Paul Klee saw the two in negative opposition to each other, he was making a valid point, but from our present perspective, he was slightly wrong. Construction and intuition constitute the dialectics of creation. Whatever the unknown may be that we call intuition, the computer's part in a creative process can only be in the realm of construction. In the intuitive capacities of our work, we are left alone. There we seem to be at home. When we follow intuitive modes of acting, we stay with ourselves, implicit, we do not leave for the other, the explicit.

So at the end of this mental journey through the algorithmic revolution (Peter Weibel's term) in the arts, the dialectic nature of everything we do re-assures itself. If there is anything like an intuitively secure feeling, it is romantic. It seems essential for creativity.

In the first narration, I presented the dense moment in Stuttgart on the 5th of February, 1965, when computer art was shown publicly for the first time. If you tell me explicitly, Georg Nees told the artist who had asked him—if you tell me explicitly *how* you paint, then I can write a program that does it. This answer concentrated in a nutshell, I believe, the entire relation between computers, humans, and creativity.

The moment an artist accepts the effort of describing how he works, he reduces his way of working to that description. He strips it of its embedding into a living body and being. The description will no longer be what the artist does, and how he does it. It will take on its separate, objectified existence. We should assume it is a good description, a description of such high quality concerning its purpose that no other artist has so far been able to give. It will take a lot of programming and algorithmic skill before a program is finished that implements the artist's rendition. Nevertheless, the implementation will not be what the artist really does, and how he does it. It will, by necessity, be only an approximation.

He will continue to work, he will go on living his life, things will change, he will change. And even if they hire him as a permanent consultant for the job of his own de-materialisation and mechanisation, there is no escape from the gap between

[19](Klee 1928) Another translation into English is: "We construct and construct, but intuition is still a good thing."

Book ID: 303794_1_En, Date: 2012-07-30, Proof No: 1, UNCORRECTED PROOF

a human's life and a machine's simulation of it. Computers just don't have bodies. Hubert Dreyfus (1967) has told us long ago why this is an absolute boundary between us and them.

The change in attitude that an artist must adapt to if he or she is using algorithms and semiotic machines for his or her art is dramatic. It is much more than the cozy word of "it is only a tool like a brush" suggests. It is characterised by explicitness, computability, distance, decontextualising, semioticity. None of these changes is by itself negative. To the contrary, the artist gains many potentials. His creative capacities take on a new orientation exactly because he or she is using algorithms. That's all. The machine is important in this. But it is not creative.

The creation of a work that may become a work of art may be seen as changing the state of some material in such a way that an idea or intent takes on shape. The material sets its resistance against the artist's will to form. Creativity in the artistic domain is, therefore, determined by overcoming or breaking the material's resistance. If this is accepted, the question arises what, in the case of algorithmic art, takes on the role of resistant material. This resistant material is clearly the algorithm. It needs to be formed such that it is then ready to perform in the way the artist wants it to do. So far is this material removed from what we usually accept under the category of form, that it must be built up to its suitable form rather than allow for something to be taken away. But the situation is similar to writing a text, composing a piece of music, painting a canvas. The canvas, in our case, turns out to be the operating system, and the supporting program libraries appear as the paints.

Acknowledgements My thanks go to the people who have worked with me on the compArt project on early digital art and to the Rudolf Augstein Stiftung who have supported this work generously. I have never had such wonderful and careful editors as Jon McCormack and Mark d'Inverno. They have turned my sort of English into a form that permits reading. I also received comments and suggestions of top quality by the anonymous reviewers. All this has made work on this chapter a great and enjoyable experience.

References

Anon (1965). Bald krumme linien. In *Der Spiegel* (pp. 151–152).

Bense, M. (1965). *Aesthetica. Einführung in die neue Ästhetik*. Baden-Baden: Agis. This is a collated edition of four books on aesthetics that appeared between 1954 and 1960. Aesthetica has been translated into French and some other languages.

Birkhoff, G. (1931). A mathematical approach to aesthetics. In *Collected mathematical papers* (Vol. 3, pp. 320–333). New York: Am. Math. Soc.

Cohen, H. (2007). Forty-five years later.... http://www.sandiego.gov/public-library/pdf/cohencatalogessay.pdf.

Dreyfus, H. (1967). Why computers must have bodies in order to be intelligent. *The Review of Metaphysics*, *21*, 13–32.

Duchamp, M. (1959). The creative act. In R. Lebel (Ed.), *Marcel Duchamp* (pp. 77–78). New York: Paragraphic Books.

Frank, H. (1964). *Kybernetische Analysen subjektiver Sachverhalte*. Quickborn: Schnelle.

Gerstner, K. (1963). *Programme entwerfen*. Teufen: Arthur Niggli. Second ed. 1968, third ed. 2007 in English under the title *Designing programmes*. Baden: Lars Müller.

Glowski, J. M. (Ed.) (2006). *Charles A. Csuri: beyond boundaries, 1963-present*. Columbus: Ohio State University.
Guilford, J. P. (1950). Creativity. *American Psychologist*, *5*, 444–454.
Gunzenhäuser, R. (1962). *Ästhetisches Maß und ästhetische Information*, Quickborn: Schnelle.
Hentig, H. v. (1998). *Kreativität. Hohe Erwartungen an einen schwachen Begriff*. München: Carl Hanser.
Herzogenrath, W., & Nierhoff, B. (Eds.) (2006). *Vera Molnar. Monotonie, symétrie, surprise*. Bremen: Kunsthalle. German and English.
Herzogenrath, W., Nierhoff, B., & Lähnemann, I. (Eds.) (2007). *Manfred Mohr. Broken symmetry*. Bremen: Kunsthalle. German and English.
Hollinger, L. (Ed.) (1999). *Vera Molnar. Inventar 1946–1999*. Ladenburg: Preysing Verlag.
Keiner, M., Kurtz, T., & Nadin, M. (Eds.) (1994). *Manfred Mohr*. Weiningen-Zürich: Waser Verlag. German and English.
Kittler, F. (1985). *Aufschreibesysteme 1800/1900*. München: Fink. English: Kittler, F. (1990). *Discourse networks 1800/1900*. Stanford, with a foreword by David E. Wellbery.
Klee, P. (1928). Exakte versuche im bereich der kunst.
Knuth, D. E. (1968). *The art of computer programming*. Reading: Addison-Wesley. Planned for seven volumes of which three appeared from 1968 to 1973. Resumed publication with part of Vol. 4 in 2005.
Lunenfeld, P. (1999). *The digital dialectic. New essays on new media*. Cambridge: MIT Press.
McCorduck, P. (1990). *AARON's code: meta-art, artificial intelligence, and the work of Harold Cohen*. New York: Freeman.
Moles, A. A. (1968). *Information theory and esthetic perception*. Urbana: University of Illinois Press. French original 1958.
Nadin, M. (2011). *Semiotic machine*. An entry of the Semiotics Encyclopedia Online. http://www.semioticon.com/seo/S/semiotic_machine.html.
Nake, F. (1974). *Ästhetik als Informationsverarbeitung*. Vienna: Springer.
Nake, F. (2009). The semiotic engine. Notes on the history of algorithmic images in Europe. *Art Journal*, *68*, 76–89.
Nees, G., & Bense, M. (1965). *Computer-grafik* (19th ed.) Stuttgart: Walther.
Nöth, W. (2002). Semiotic machines. *Cybernetics & Human Knowing*, *9*, 5–21.
Shannon, C. E., & Weaver, W. (1963). *The mathematical theory of communication*. Chicago: University of Illinois Press.
Stern, W. (1912). *The psychological methods of intelligence testing*. Baltimore: Warwick and York. Transl. from the German.
Sundin, B. (Ed.) (1980). *Is the computer a tool?* Stockholm: Almqvist & Wiksell.