UTOPIA COMPUTER

The »New« in Architecture?

Nathalie Bredella, Chris Dähne,
Frederike Lausch (Eds.)

Forum Architekturwissenschaft
Band 6

Universitätsverlag
der TU Berlin

NETZWERK
ARCHITEKTUR
WISSENSCHAFT

UTOPIA COMPUTER
The “New” in Architecture?

Nathalie Bredella, Chris Dähne,
Frederike Lausch (Eds.)

The scientific series *Forum Architekturwissenschaft* is edited by the Netzwerk Architekturwissenschaft, represented by Sabine Ammon, Eva Maria Froschauer, Julia Gill and Christiane Salge.

The critical concern of the book "Utopia Computer" is the euphoria, expectation and hope inspired by the introduction of computers within architecture in the early digital age. With the advent of the personal computer and the launch of the Internet in the 1990s, utopian ideals found in architectural discourse from the 1960s were revisited and adjusted to the specific characteristics of digital media. Taking the 1990s discourse on computation as a starting point, the contributions of this book grapple with the utopian promises associated with topics such as participation, self-organization, and non-standard architecture. By placing these topics in a historical framework, the book offers perspectives for the future role computation might play within architecture and society.

NETZWERK
ARCHITEKTUR
WISSENSCHAFT

Forum Architekturwissenschaft, Volume 6

UTOPIA COMPUTER

The “New” in Architecture?

Nathalie Bredella, Chris Dähne,
Frederike Lausch (Eds.)

Universitätsverlag
der TU Berlin

→

FRIEDER NAKE AND ARIANNA BORRELLI,
NATHALIE BREDELLA, MADS FRANDSEN
JULIUS WINCKLER

Extracts from an Interview with Frieder Nake

Berlin, December 2017

Interviewers: How did you program your Graphomat?

Frieder Nake: At the time in Stuttgart, the situation was as follows: the Graphomat, engineered by Konrad Zuse, arrived completely without any software. The only thing we had was the announcement by Prof. Dr. Walter Knödel, Head of the Computing Centre at the University of Stuttgart (which was then still a Technische Hochschule): “We’re going to buy one of those machines.” Software did exist for the Graphomat, but it was only compatible with Zuse computers and not with our computer at the Computing Centre (a Standard Elektrik Lorenz Elektronenrechner – SEL ER-56). I couldn’t use any of it, since we didn’t have the software. When Knödel announced the purchase to me, he asked if I could develop the software for the Graphomat, to which I agreed. I’m telling you this anecdote because, to me, it’s a wonderful example of excellent teaching: a professor asks the student to do something—without having any idea himself of what that student may know in the specific case. The professor thereby showed great trust in the student. A first principle of education is mutual trust. That’s why I’m opposed to examinations, where everyone cheats any way and nothing is learnt that way.

I: How can you program a machine that isn’t there yet?

FN: Very easily, it's very simple. Programming means creating descriptions, descriptions for a computer to carry out. The machine only serves to test whether the description suits it and works, or not. All my computer knowledge is based on the assumption that the machine initially also exists only in the form of a description. Back in the 1960s, I programmed on the lowest level—i.e. on the naked machine, which only has one button to start and stop.
Everything in the field of computing consists of descriptions. So I needed nothing but a description of the new Graphomat, which had not arrived yet. Computing itself consists of a world of descriptions: it is a semiotic world. We can regard the descriptions I had to make as text *and* machine, at the same time. They are virtual machines that appear as text and can be read. But those texts, which we call "computer programs," may also be viewed as machines, *text-machines*, we might say. And in that respect, they are a completely new form of text.
For each specific program, you could build a specific computer (which is naturally a rather crazy idea). Each of these computers, though real and actual, would then be—or, better, act as—a virtual machine. Then you wouldn't need to write new programs any more (only when a new program is needed, which is not very unusual). Inversely, it would be possible to run all programs in the world on a single computer. But that would also be extremely impractical because, of course, everyone would want to use it at the same time. That's why we see computers everywhere. No other machine exists as abundantly as the computer and that is its special aspect, a result of its semiotic (sign-based) nature. In principle, that would all be possible in our brains, if suitably trained. However, we humans are not very good at remembering things, so we will continue to write things down.

I: Was it already the Graphomat's task to draw back then?

FN: The Graphomat's task was always and exclusively to draw. There was only one additional function, which was derived from drawing, namely turning lines on paper into incisions in a material, which served as a print template. But the task at hand was to write programs for the computer that would ensure that the Graphomat really did draw something you wanted to be drawn. What was drawn had to be calculated on the computer to create a punched tape that, in turn, controlled the Graphomat. I had to force the computer to draw or, more precisely, to come up with a sequence of commands (a text) that would control the Graphomat in the desired way. The computer didn't want to *draw*, since it was built to *compute*. My task, therefore, was to let the computer compute, but the result of its computing was a drawing. It was a wonderful moment when I realized that this was really what my task required.

I: What was the machine supposed to draw?

FN: We didn't know yet. When I started to design and construct the basic software for Graphomat's drawings, I had to think of a mechanical engineer, physicist, mathematician, architect or sociologist coming along, wanting visual representations of his or her calculations. My software had to work for all of them. So, I had to think of the drawing as such and not of what exactly would be drawn in a specific case. The machine used a system of coordinates. Those coordinates would be used to represent points connected by straight line-segments. My job was to use only the points that the innate Graphomat grid of points could actually get to, and approximate as closely as possible what the architect or engineer wanted to see.
The Graphomat has an available area of about 1.5 x 1.5 metres. Paper was affixed to it. A pen or paintbrush is inserted into a small support. It rests on a mechanical arm, along which the support may move, while the arm itself moves in an orthogonal direction, driven by a spindle. In this manner, the pen can perform moves in a large number of directions. Definitely not *all* directions—we are, after all, in the digital realm, not the analogue! There was a total of 1024 directions the Graphomat could perform precisely. All other directions had to be approximated by zigzag-lines. This

is the price we must pay for trying to create analogue drawings by discrete (digital) means.
The drawing machine reads the punched tape that the computer has delivered. It is absolutely precise in following the text on the tape, never deviating. Only humans could create deviations: stop the machine, move the pen somewhere else by moving the "drawing head" and resume drawing. Of course, that's rather stupid because why would you have the computer calculate a drawing that you would then not allow to be carried out?
This disruptive process could now be called "interaction"! A kind of interaction before its time. The computer "knows" the pens only as numbers 1, 2, 3 and 4. The concrete pen resting in one of these positions (the colour of its ink, the width of its tip—0.2 mm or 5 mm) is not the computer's, nor the Graphomat's concern. You can use a computed punched tape with different graphic equipment to get a completely different drawing. The code of the punched tape is abstract; the pens and colours are concrete.

I: When does the context of art, in which you work, become more significant?

FN: That requires another anecdote. In programming, almost the only way to learn whether your program works correctly is through testing. Research must prove, with mathematical rigour, that a program is correct. But until now, we do not know much. If you want to discover how correct a complex program is, you must apply exhaustive test methods, which is tough and tedious!
I should have tested the 256 directions of each quadrant, amounting to 1024 directions. And then their use in approximating the infinitely many directions when drawing, as well as the smooth curves. Not impossible, but real work. I didn't really fancy that job. So I told myself, "work by chance!" The logic behind it was that, if it looks correct, it may well be correct. That's an advantage of drawing over calculating. We forgive tiny deviations. And that was the launch into art! Take a chance! Give up the absolute precision of digital calculations and trust the slight sloppiness of perception instead!

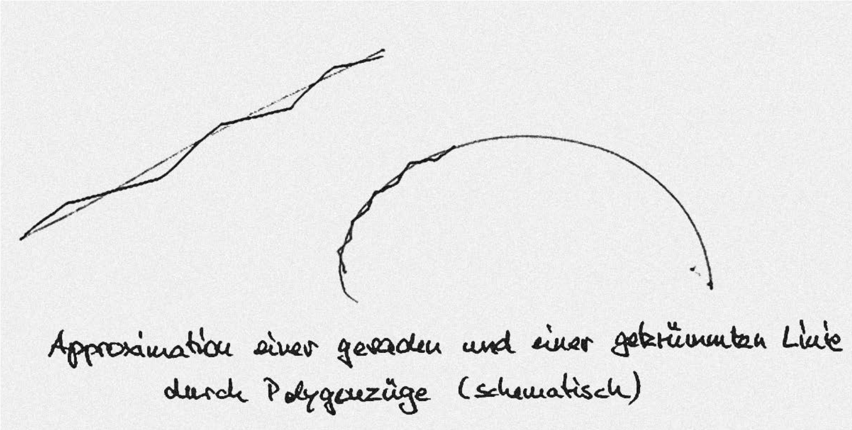

Fig. 1: Stepwise straight lines (polygons) approximating a straight line and a circular line

One day, I had to tell myself: “OK, now try the circle.” On the Graphomat, there are no circles. You must approximate its beautifully smooth, calm line by very short straight edges. If they are drawn in quick enough succession, short and shorter in length, they may appear to us as “smooth.” Of course, they are not and never will be. But we are tolerant and can be duped. The ancient Greeks already knew that we can approximate a circle this way. A circular line is created out of a polygon with an ever-increasing number of edges. The sketch above gives an impression of this simple principle (fig. 1).

Let us take a look at the simple image of figure 2. You would, quite likely, agree that we see smoothly curved lines. With a bit of effort, you can see that the sixth line, counting from the outside, resembles a circle. Before it, and then further on to the inside, we see the circle’s “sisters.” They are created from the circle by shifting the regularly spaced points along the circle by randomly chosen amounts, slightly further out or in. The newly determined points are then interpolated. For whatever reason, something slightly sensational has happened to the automatically ongoing process of interpretation: a thin needle punctures the pleasingly adjacent lines. From an aesthetic point of view, the needle is rather well placed and sized. It clearly dominates the image and lends a pleasurable surprise to the image. In some

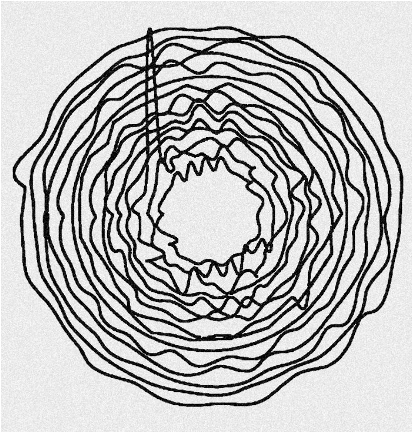

Fig. 2: Frieder Nake, Kreise, 1965

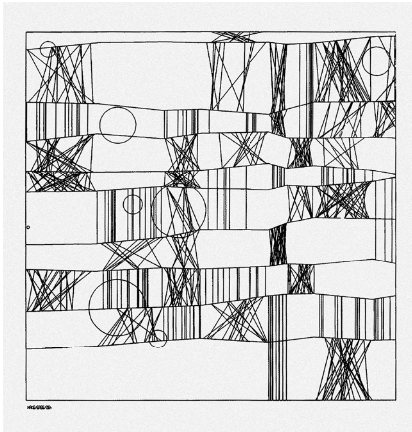

Fig. 3: Frieder Nake, Homage à Paul Klee, 1965

way, the interpolation needed to gain momentum to achieve what the algorithm required of it. A truly remarkable drawing emerged this way (considering this was 1965!). Without the anomaly, the whole drawing would be relatively boring. Chance made it a bit more exciting, don't you think? Similar things happened in other cases again and again. As soon as I had engaged in this kind of process, our good friend—*random chance*—assisted me in achieving a few such gems.

I was particularly fortunate with the program I called *Homage à Paul Klee* (fig. 3). Even though, at close inspection, you will discover a number of aesthetic weaknesses, it immediately attracted attention, positive feedback and even a little admiration and acclaim. It became one of the best-known results of early algorithmic art.

Whatever we *see* must be perceptible to our eyes. It must exist in what is known as the "analogue" (continuous) mode of reality. We don't *see* digital things. We can conceive of them and indeed do so. But such a statement only identifies a general concept, as we realize when pondering differences between analogue and digital!

The image in figure 4 presents my program "Matrizenmultiplikation" (matrix multiplication, from 1967). Details of the way the program works are irrelevant. It suffices to know the following facts for an appreciation.

In mathematics, a “matrix” is a quadratic (e.g. a square) arrangement of numbers that are organised in rows and columns (never mind that there are also rectangular matrices). In our example, all these numbers are between 0 and 1. I subdivide the interval from 0 to 1 into seven short intervals, one immediately following its predecessor, without any gaps. A colour is assigned to each of the intervals. Scanning the numbers of the matrix and replacing its numbers by the intervals they belong to and, furthermore, putting the interval’s colour at the location of the number, creates the coloured array. The numbers of the matrix are clearly digital. Their corresponding colours are analogue. “Matrizenmultiplikation” is (among other things) a machine to translate from the digital and discrete to the analogue and continuous, even though the structure of the small squares is, again, discrete.

Let us take a closer look at the colours of the image. Take the top horizontal row. As you can see when moving from left to right, there are repeated densifications, roundish darker shades of the same hue. This effect is clearly visible at the far left in the second green, and later, in the violet and orange fields. I should tell you that the machine’s upper row is drawn from left to right. The pen (5 mm wide) is lowered down onto the paper, then draws a 5 mm long stroke, and is lifted up again. It is followed by the next elementary step and the next one, changing pens and colours. Now, when the pen is stopped in order to be lifted up, this takes a tiny amount of time. It is enough to allow a little more ink to flow onto the paper. The result are those blotches in forms that are completely uncontrollable, blots, dark patches. A purely analogue process is added to the image of the digital, while in our perception (the realm of aesthetics!), it may gain special attraction.

I wish to add that, today, such an effect is no longer possible on slick (and *more* digital!) computer screens, unless you make the pointless effort of simulating the analogue digitally. Which may be rather crazy!

There is another aspect I want to make you aware of. Take a look along the second row, but now going from right to left! Can you see it? The darker spots are now situated to the left in the small colour-fields, not to the right as before. On the way back,

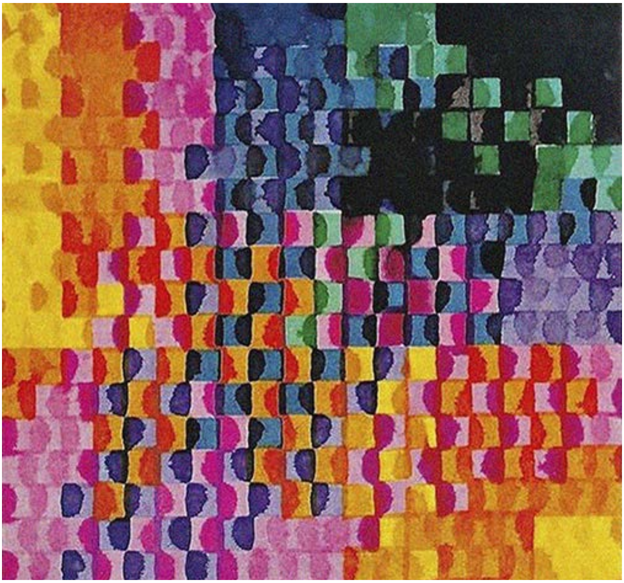

Fig. 4: Frieder Nake, Matrix multiplication series 31, Detail, 1967

the machine was programmed to draw from right to left, so the pen is always lifted on the left-hand side of the little squares. This effect creates some attraction—don't you agree? I certainly do. The algorithmic (and digital) image that I force the computer to produce appears in analogue mode in a way that simply does not appear in its programmed or algorithmic form of existence. Forgive me if I'm too proud of this earliest period. The digital and the analogue were still friends back then. They loved each other. And I loved them.

I: What is the relationship between visual insight and mathematical formulae?

FN: Our example offers a good way to explain that. The program is called "matrix multiplication." That's a bit mathematical. Matrices can be multiplied. Whatever that means and however you do it, it is precisely defined. So, my thought was this: take something from mathematics, take a matrix, and make it the source of images (fig. 5).

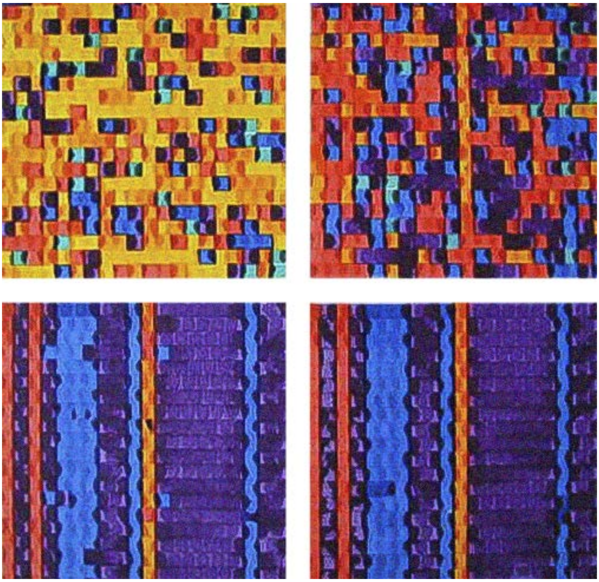

Fig. 5: Frieder Nake, Matrix multiplication, 1968. Four states shown: up left, up right, down left, down right.

I choose a matrix at random; call it *A*. Multiply it by itself, thus producing its "powers." First, you get A^2, next A^3 and so on. When the program is running and producing the powers, one after the other, it occasionally takes the current state and puts it aside, perhaps A^2, A^5, A^{10}, A^{20}. These states are translated into images, the coloured interpretations of the matrix of numbers. Once in a while, the state that the matrix has just achieved is visualised. It is a relatively simple process. As you can guess, the definition of the intervals between two visualizations has a great influence on the visual nature of the image. That is where the artist resides, the concrete form of visualisation!

I: Are the visualised matrix powers different in the images?

FN: They can vary from image to image and the colour intervals can also be different. Parameters abound: the number of colours to use, the choice of colours themselves, the division of the interval [0, 1] into subintervals, the size of the square matrix, the random first choice of the matrix, the number of states, the sequence of states to be visualized. It became clear that you need some sensitivity to make a good or fruitful choice of all those. You need to experiment.

I should add an important secret that has tremendous influence on the appearance of the images. The secret is that the matrices are so-called *stochastic matrices*. Think of something called a "state system." In every moment, such a system is in one of many possible specific states. It is irrelevant for us how a change of state occurs. We only know that, from its current state, the system assumes its next state through probabilities. We use the term "transition probabilities." The process of getting from a current state, say σ_{i}, to the next state, say σ_j, is controlled by transition probability p_{ij}. Such processes are called *Markov Processes*. The matrices with which I played are ones that describe such Markov state systems. Now we see a bit more of what their fate is, from a higher perspective. This interaction between precise mathematics and random choice is art!

I could talk for a long time about this, but that would be boring. Think of this: you see a medieval painting, or an image from the Renaissance period. Without much hesitation, you may say: aha, that's Mary, in the background a shy Joseph, and the little baby Jesus in the foreground is so sweet. Petals raining from the sky, angels exulting, and more. Clearly, they are all just blotches of colour. But each of us interprets and immediately recognises them. Hardly anyone says: Of course, that's the devil, or: Look, that's my mum with me. It's always Mary and the baby Jesus.

But when we look at the "matrix multiplication" images, we don't immediately say: sure, a Markov System, in an advanced state. We don't say something like that because it isn't part of our observed reality or our heritage of stories. It doesn't have to be that way, but that's how it is.

Behind such a Markov matrix of transition probabilities lies the following. If we create a sequence of powers of the initial matrix and go pretty far with it, the matrix rows stand for the probabilities at which the system transforms from its original state into any of the other states within so and so many steps, as defined by the powers of the matrix. However, the entire system strives to reach a limit state. Mathematics shows that, in the end, it is completely irrelevant in which state the system began. It can go wildly up and down, over and under, but ultimately that makes no difference, since all states eventually become stochastically equivalent. As I said, that is what happens towards the end. The end may be infinitely far away. That is what the images of Matrix Multiplication show us. Just like Jesus and Mary, always the same.

I: Even the selection of the states?

FN: Even that, with a bit of luck. We may ask the program to display first, fifth, fifteenth and twentieth state. We don't know in advance wow close, by that time, the system may already be to an almost stable phase of its behaviour.
I have produced perhaps 40 or 50 images with the program. But I could have continued exploring for a much longer time by systematically varying the large set of input parameters. Mathematics may be a relatively abundant source. That's why the program allows so many experiments. The process of intelligently and sensitively engaging with it becomes art. An abstract intuition is as erotic as any other.

I: Did you have any idea at all what the results of the program would look like?

FN: Not at all. Well, naturally I knew a bit about the structure. But only the general structure. Not the appearance.

I: What else existed in the field of visualisation and mathematics at the time?

FN: Naturally, I never really thought in terms of "visualisation." I don't even like using the word today, but nowadays everyone uses it all the time. Today, everything must be a visualisation. I thought that was dumb, because my aim was to create art. Is that a bit arrogant? I am afraid, it is.

Paul Klee had a wonderful description of this aspect: “Art does not reproduce the visible; rather it makes visible.” We don’t create an image of what everyone can see already; we rather create images of what nobody usually sees. I adopt that motive. I didn’t produce any other images in the strict sense of, *now I’ll take, as my source, something from mathematics*. But mathematics, all by itself, is not visible. In that sense, the images generated by the program “Matrizenmultiplikation” are testimonies to Paul Klee. Mathematics is always present when we write a program. It is not usually as explicit as in this case.

Let me continue with the statement that the aesthetics of images may be described on the basis of rational processes. In 1968/69, when I was in Toronto as a guest of Leslie Mezei, I roughly followed a line I wish to explain now. By that time, I believed in the radical-rational approach to aesthetics that Max Bense was trying to develop.

Let’s assume we have ten different available criteria. Images would then be assessed according to those. So we start by analysing the image in terms of criterion number one. If it is a quantitative criterion, the result is a number. We do the same with criterion number two and carry on until the last criterion, number ten. Images are thus represented by a “point” in a ten-dimensional space.

The image of figure 6 was created by a program that I proudly called *Generative Aesthetics No. 1*. The program’s task was this: take all the aesthetic measures known to me at the time (as defined by Helmar Frank and Rul Gunzenhäuser in the early 1960s) and construct an image that should fulfil the numerical conditions defined using the given criteria. More concretely, this could, for instance mean: the *measure of prominence* for blue should be between 0.2 and 0.3, while the *measure of surprise* for yellow should be about 0.7, and similar conditions should hold for other colours. Subject to these constraints, maximize the information-aesthetic measure!

To cut a long story short: by the end of that year, I was done with this work. I was quite curious when I started this project in summer 1968. Would I be able to solve the mathematics? Solving the mathematics and developing the program took me the entire year.

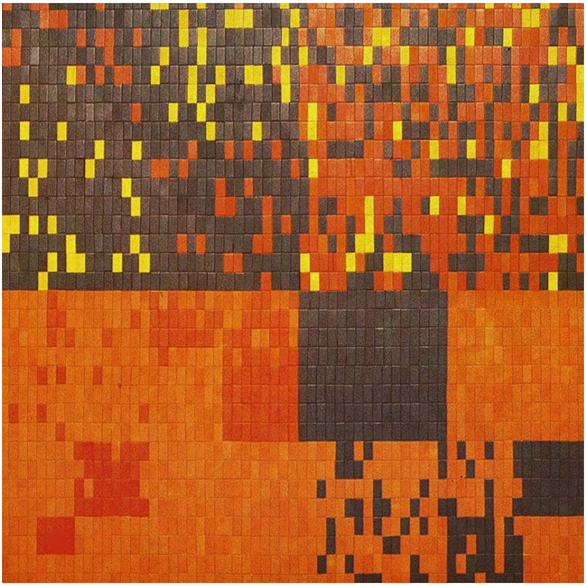

Fig. 6: Frieder Nake, Generative Aesthetics I, 1969

I was proud when the year was over and my program said: your conditions do not allow for any solution, or: there are solutions, and here is one. I had succeeded in developing a high dimensional optimization problem with constraints whose result was a probability distribution of the given colours according to numerically given constraints.

But more than this: The program's second part created a distribution of the colours according to the probability distribution determined by the first step.

The second part was important. For it constructed an actual image according to the frequencies the first part had determined to satisfy the constraints formulated in terms of the radical information-aesthetics. The first part, determining a frequency distribution of colours, became the necessary *statistical pre-selector*, as I called it.

I became fully aware of the fact that information-aesthetics always only regards things from the point of view of statistics! This follows from the fact that it is based on the measure of information according to Claude Shannon. Following him, the image is nothing but a perpetual source of visual (even aesthetic) information. If I wanted to create an image from the first selection of probabilities (better: frequencies), I would have to address a topological-geometric task. This became the starting point for part 2 of the program.

I invented a suitable data structure that was simultaneously and independently developed for other purposes at two other institutes. This data structure became known as the "quadtree."

For me, the structure was to distribute onto the image the amounts of yellow, red and blue, or whatever other colours there were, in accordance with the calculated frequencies. To that aim, the entire image was split into its four quadrants. In the next step, each of those four quadrants was split into its four sub-quadrants, etc. down to a smallest size of quadrants.

In each step, the entire mass of colour available here was distributed down, etc. A simple, procedure, rather free of any considerations of context, a scheme, not more. But nobody had done this before. The image of figure 6 was created this way, as you can no doubt assume.

Generative Aesthetics no. 1 firstly followed a principle of distribution, and secondly a principle of topology. Not a bad move. The aesthetic criteria applied during distribution are relatively weak. But they are rational, numeric, quantitative criteria. They are blind. They don't know what blue is like. They only know "Colour 1" and "Colour 2" etc. But adjacent colours would have to be considered in order to approach aesthetic appeal. The incredibly powerful machines of today can do that. It was impossible for a single person in 1968.

The computer in Toronto generated many images for me using *Generative Aesthetics no. 1*. Since there was no drawing machine that could create proper coloured drawings from the printed output, I had to accept distributions of stars, short lines, slashes and other printer symbols on continuously folded sheets of paper as the results of these efforts.

Back in Stuttgart, I worked with a few students from the Academy of Fine Arts. We did our tedious work by hand, sticking small pieces of coloured cardboard onto a large panel, following what the computer printouts told us. In doing so, we quite happily turned ourselves into the servants of the Toronto computer. We finished two of those panels. One of them was acquired by the great collector Etzold, who passed it on to Abteiberg Museum in Mönchengladbach, Germany, where it has been displayed several times. The other one was lost. Rumour has it, my mother didn't like it.

I think, this was the height of information-aesthetics. In some way, a triumph. Shortly after, I gave up and moved away from information-aesthetics to real computing. This would be another story.

Bibliographic information published by the Deutsche Nationalbibliothek

The German National Library lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at http://dnb.dnb.de.

Universitätsverlag der TU Berlin, 2023

https://verlag.tu-berlin.de
Fasanenstr. 88, 10623 Berlin
Tel.: +49 (0)30 314 76131
E-Mail: publikationen@ub.tu-berlin.de

Editing: Daniela Petrosino
Proofreading: Clara Dick
Translation: Ben Liebelt
Picture Editing: Jürgen Schreiter, Darmstadt
Layout: Stahl R, www.stahl-r.de
Typesetting: Julia Gill, Stahl R
Print: docupoint GmbH

ISBN 978-3-7983-3270-6 (print)
ISBN 978-3-7983-3271-3 (online)

ISSN 2566-9648 (print)
ISSN 2566-9656 (online)

Published online on the institutional repository of the Technische Universität Berlin:
DOI 10.14279/depositonce-15964
http://dx.doi.org/10.14279/depositonce-15964

The critical concern of the book “Utopia Computer” is the euphoria, expectation and hope inspired by the introduction of computers within architecture in the early digital age. With the advent of the personal computer and the launch of the Internet in the 1990s, utopian ideals found in architectural discourse from the 1960s were revisited and adjusted to the specific characteristics of digital media. Taking the 1990s discourse on computation as a starting point, the contributions of this book grapple with the utopian promises associated with topics such as participation, self-organization, and non-standard architecture. By placing these topics in a historical framework, the book offers perspectives for the future role computation might play within architecture and society.

Universitätsverlag der TU Berlin
ISBN 978-3-7983-3270-6 (print)
ISBN 978-3-7983-3271-3 (online)