2017.xCoAx.org

Lisbon
Computation
Communication
Aesthetics
& X

BETWEEN THE TRIVIAL AND THE IMPOSSIBLE

RECODING AS LEARNING STRATEGY

SUSAN GRABOWSKI
sgrab@informatik.uni-bremen.de

FRIEDER NAKE
nake@informatik.uni-bremen.de

University of Bremen
Germany

Abstract

In recent years, the term "re-coding" (or ReCoding) has been used by Matthew Epler and Marc Webster to develop program code capable of re-creating works of early computer art. The incentive was to save such works from oblivion (as old equipment is no longer available). The task, if solvable, contributes to preserving an important phase of modern cultural heritage. Formally, solving the re-coding problem is equivalent to defining the inverse function of an unknown function of which only a few specimens are known. Here, such specimens are artistic works. We employ techniques of analysis, inspection, intuition, and selection to extract from a few given images as much structural, probabilistic, and geometric data as possible to re-code the class of images the given specimens belong to.

This is a form of inductive and constructive learning in art education. Including algorithmic art in art education may serve as a bridge between the two great topics of media: algorithmics and aesthetics. Recoding may at times be almost trivial, but it may also be impossible. The middle ground of complexity is interesting: imagination is required.

Keywords

Re-coding
Processing
Generative Art
Art Education
Algorithmics
Algorithmic Thinking
Two Cultures

1. INTRODUCTION

On the 7th of May, 1959, physicist and novelist, C.P. Snow (1905-1980), delivered the traditional Rede Lecture[1] at the University of Cambridge. The lecture was later published under the title *The two cultures and the scientific revolution* (Snow 1959).

Snow gave the lecture in times of the Cold War. His message was that there is a deep gap in the West between the scientific and the literary cultures, as he called them. The first was considered to be precise in methods and techniques, experimental, based on proof and logic, and its activity was to explain. The second was ambivalent, speculative, based on interpretation and experience, and its activity was to comprehend. Representatives of the scientific culture were generally under-represented in affairs of running the state, but their work was of utmost importance for the future of the nation's wealth by technological progress. Members of the literary culture, on the other hand, held important positions in state affairs, but did not usually contribute to the wealth of the nation.

Snow was convinced that this unfortunate situation was due to a language barrier between the two cultures. This barrier had to be overcome, Snow's message postulated. In fact, communication between the two camps had essentially collapsed. On the other hand, if we look at our current debates and to activities in education, in the arts, in artistic endeavors, we may gain the impression that art can today no longer thrive without linking up with science and technology and that, whatever you want to engage in, you must build an interdisciplinary team to be mildly successful. The emphasis laid on such doing-something-together-at-all-cost is enormously high. Snow's warning sixty years ago against a cultural split between the humanities and engineering still seems to be relevant and is still not remedied. But could it not also be true that the distance between the scientific and the literary was a dialectical contradiction that should be treated as such, as a contradiction? In the dialectical view of the world, contradictions are reasons for dynamics, movement, change, innovation, invention. Why not celebrate contradiction, instead of artificially gluing it? — This paper takes up a challenge that has emerged in algorithmic art and that may act as a good reason to see how a contradiction may develop into beautiful and satisfactory processes.

1
Inaugurated around 1668.

2. RECODING

Coding is the activity of describing an algorithm[2] in such a form that the compiler or interpreter of a chosen programming language can deal with it. *Re-coding* (also written as ReCoding) must then be the activity of trying to re-construct the code of a drawing — in the case, of course, that drawings are the resulting objects of calculations (which is expressedly the case for us).

The situation is simple and clear. We are given a drawing, perhaps even more than one. We know or assume that they are results of the execution of a program. But we don't have a clue of the program itself; at any rate, it is not available to us. In case of several given drawings, we assume they came from one and the same program. In such a situation the task is: Design code that is capable of generating images similar to the given images, plus an unlimited number of more. We are *not* interested in *copying* the given images. We consider the task to be solved if the new code generates images that come close enough to the given ones. Images must not coincide in each and every detail. It is enough if a person,

2
We assume the reader is at least somewhat familiar with this very important term.

comparing the given to the re-coded images, concludes that the two evidently belong to the same class of images, however that class would be defined. Intuitively, we seem to be capable of judging such vague similarity. [3]

More than such class-similarity we cannot hope for. If we defined the task more strictly by requesting that one of the re-coded images is equal in all its detail to the given one, we could always provide a trivial re-coding: scan the given image, store it, and output it on request. If not only one, but several images were given at the start, the trivial code would have to be slightly more complex. We would scan all the given ones, store them, and prepare code containing a switch that randomly selects and outputs one of the stored cases. As said before, re-coding is more subtle than copying.

We conclude that the task of re-coding always allows for a super-trivial solution that is, however, not based on constructive code. The trivial new coding in fact evades the problem of coding altogether. An acceptable solution to the re-coding problem requires that the new image be constructed in a more or less new way where this new encoding only requires a good measure of similarity between the given sample results from the unknown program and the output of the new code. To solve this task, we will have to perform an analysis of basic elements, structures and superstructures, measurements, and other analytic considerations of the given sample. The job of re-tracing an unknown algorithm of generative art surprisingly puts us into the position of an extremely accurate analytics of art.

We see a good chance for introducing generative art into art education. We see a good chance also for curating exhibitions of algorithmic art, and even for some modest but fruitful overcoming of the two cultures gap. We see chances for kids, students, and adults to engage in artistic and algorithmic activities offering new and rewarding experiences. Such activities have the potential of helping us remain true human beings whilst the environment around us is accelerating the digital race. We see a chance for slow lingering against faster and faster glimpsing. In late fall of 2012, the US American Matthew Epler came up with the idea of re-coding works of early algorithmic art. Epler considers himself a creative technologist, "specializing in creating one-of-a-kind interactive projects" (Epler 2013).

On the more or less abandoned website of his ReCode Project he says:

> **The ReCode Project is a community-driven effort to preserve computer art by translating it into a modern programming language (Processing). ... The focus of the ReCode Project is three-fold: 1. Bring historic works of computer art back into the public eye. 2. Make it accessible and useable. 3. Save the code. (Epler 2013)**

Not much later, British Mark Webster suggested to Epler to organize a series of events, including the idea of cities declaring themselves to be ProcessingCities. [4] Whereas Epler concentrated on the web-site inviting people to re-code early computer art, Webster wanted people to connect in actual spaces experiencing history in a new way by learning from pioneers. It seems that besides a challenging but quite limited Re-coding workshop and lecture in 2013 in Bordeaux, France, not much has actually happened. As far as we know, there are no publications about this approach, and after a series of rather trivial re-codings of computer-generated images taken from Grace Hertlein's short-lived magazine *Computer Graphics and Art* (Hertlein 1976-1978) nothing tangible seems to have come from the idea.

3
Copy, imitate, simulate may be considered different levels of creating similarity; but also getting inspiration.

4
"Processing" here refers to the programming system of Processing, (Reas & Fry 2014).

The very idea of re-coding, however, caught on in our research group at the University of Bremen in Germany. Different, perhaps, from original intentions, and not without critical analysis of the potentials of this approach, we have over a number of years gathered a fair amount of hands-on experience in projects of re-coding. We will later indicate an example. As part of our long-term project on algorithmic art (called *comp*Art[5]), we have used various seminars for individual re-coding efforts of various kinds. So even if Matthew Epler's first impulse may have had only little resonance, via Mark Webster's French connection the idea of re-coding fell on fertile ground in the North of Germany.

5
The roots of projects under the title of *comp*-Art reach back to the mid 1960s.

3. EXTREMES

To formulate it once more, the re-coding problem is this: Given is a non-empty finite set of images. They are of algorithmic origin and from one and the same source program. Find an image-generating algorithm whose set of generated results contains the given set of images.

Let us formalize this. Let P be an image-generating source (a "program"). We denote by $\Gamma(P)$ the set of all images that P can generate. Let the non-empty finite set $I \subset \Gamma(P)$ of images be a subset of the set of all possible images of P. Find an algorithm A such that its coded version A' satisfies the condition $I \subset \Gamma(A')$. I.e., we want to find among the images generated by algorithm A (really, by its coded version A') all the given images (as already indicated in the previous section).[6]

So the re-coding problem is solved by an algorithm. The algorithm will, in general, be capable of producing infinitely many images. The parameters of the algorithm determine the generated images in their different appearance.

The structure and basic functions of the algorithm determine the shared characteristics of those images. Re-coding suggests to take the constructed A' for the unknown P.

Let us look at a few examples to see more concretely what the task is. In figures 1 and 2, we see two of the earliest drawings of computer art. They are by A. Michael Noll and Georg Nees, who were responsible for the first two exhibitions of computer art.[7] As we will see, they are easy to re-code.

Not without effort, we may detect that for both drawings consist of just one continued line (polygon). This property is the key to easily re-code both images. In fact, both drawings, different as they look, and different as their aesthetics must therefore be, there is one simple algorithm generating both of them:

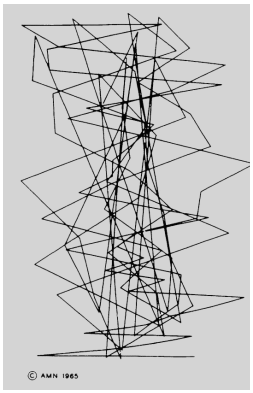
> **Within the given image format, choose at random a starting point; call it $P0 = (x0, y0)$. Randomly choose a direction α and a length d for the next line-segment. Compute the coordinates of the next point $(P1)$: $x1 = x0 + d*\cos\alpha$, $y1 = y0 + d*\sin\alpha$. Draw a straight line from $P0$ to $P1$, using the currently active color and line attributes. Redefine $P0$ by giving it the coordinates of $P1$. Repeat this as often as a finishing parameter n (the number of edges of the polygon) requires.**

We assume that the reader does not necessarily start from the assumption that the two drawings are structurally equal, since their visual appearances are definitely quite different. In Noll's drawing, the orientations of edges are varying widely and rapidly, whereas in Nees's example they strictly alternate between

6
We neglect the fact, mentioned before, that any necessary comparison is relative to a measure of similarity. That may be in the results (the *surface*), or the algorithms (the *subface*).

7
Georg Nees: *Computer Grafik*. 5 to 19 Feb., 1965, at Max Bense's Studiengalerie, University of Stuttgart / A. Michael Noll (with Bela Julesz): *Computer-generated pictures*. 6 to 24 April, 1965, at the Howard Wise Gallery in New York City.

horizontal and vertical. At least for some of us it comes as a surprise that the same structure allows for such diversity in appearance. This results from the parameters that determine what it means to do something "at random". For early computer art, as well as for today's, randomness and, thus, probability distributions play an enormous role.



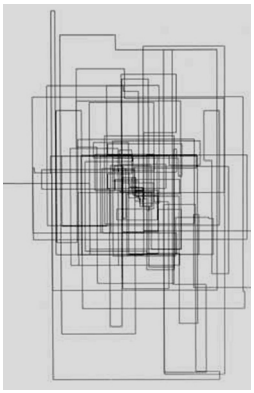


Fig. 1
A. Michael Noll:
Gaussian quadratic 1965.
Credit: A.M. Noll.

Fig. 2
Georg Nees: *Irrweg* 1965.
Credit: G. Nees.

In Georg Nees's drawing, the probability distribution for the direction of the next edge must be such that only four cases are permitted: up, down, left, right. Each one of them must be assigned a probability. But the newly chosen direction depends on the previous one for the alternating behavior.

Not easy to discover is the important property of the choice of coordinates in Noll's image. Its name hints at it: *Gaussian quadratic*. How many of us would conclude that one of the coordinates progresses quadratically, whereas the other is chosen from a Gaussian (normal) distribution? We consider this feature as impossible to detect by pure inspection, whereas the overall dramatic structural equality is rather simple to discover, but still contains an interesting aesthetic lesson on abstraction.

Let us turn to figure 4, a picture by the Spaniard Manuel Barbadillo. It is trivial to describe. One and the same basic module is arranged repeatedly on a 4-by-4 grid. The module allows for four rotations by 90 degrees. Each of the rows contains the four positions of the module, as well as each of the columns and each of the four quadrant squares. There are symmetries between neighboring rows and columns: a lot to be detected.

The image is purely combinatoric. A program and, thus, the computer, is not needed. To use a computer to carry out the picture is, in a way, a luxury. Some of the manual skill may be shifted on to the computer in executing the pattern. Not more to be said. In terms of the re-coding problem: trivial.

How about Manfred Mohr's picture from 1970 (figure 3), when this grandmaster of algorithmic art was still doing his first steps into the new world. The picture looks complex. A lot is happening in it. Small creatures appear everywhere taking on graphic forms of straight lines, dots, circles, triangles, hooks, ranging from shortest line marks to quite complex assemblages. We detect areas of emptiness, and areas where those creatures amass. *Êtres graphiques*, Manfred Mohr has called them: graphic beings.

In some places of the image, it looks as if a grid was supporting the chaotic world of those beings. In other areas this view is not directly backed up. On the other hand, the locations of the beings appear to be too orderly determined to suggest they were just randomly thrown at the canvas. But at least one first statement can be ventured describing this image: It is based on a large repertoire of simple and complex geometric shapes that are placed at certain locations.

If we accept this statement as a starting point for a continued analysis, we have now two subtasks to solve. One is to describe all the elements of the repertoire. The second is to find rules governing the choice of the next location. Once these two tasks are solved, the description of the image is simple:

> **Randomly select one element from the given repertoire of elements. Additionally, some geometric transformation may be required (e.g., size adjustment). Determine the next location according to the rules for locations. Put the chosen element at the determined location. Repeat this sequence of operations until a pre-determined number of elements is reached.**

For the task of determining the next location, we may start from the hypothesis of a regular grid underlying all the placements. If we engage in some measurements, we will find the hypothesis verified. Determining the many individual options that are open for the elements of the repertoire, a lot of work must be done, but it can be done.

In his early times of getting into the algorithmic world of generative art, Manfred Mohr was particularly interested in issues of reperoire, i.e. of those sets that build the basis for an artist's choice in constructing an image. This question of the repertoire is to the heart of the aesthetics of figure 3.

What do we learn from these four examples? We learn first that there are trivial cases whose re-coding can be done instantly. The combinatorial picture of figure 4 is such a case, and with a bit more of analytical effort, figure 1 and 2 are also. We learn, second, that an image may allow for a trivial top structure description (figure 3) that relies on a more complex analytical step which, in itself, is not trivial, but solvable. With this sort of images, we enter a realm of art where the method of re-coding becomes interesting for art education. We have identified *a lower boundary for re-coding*. This trivial level develops into a domain of tough challenges. At the lower boundary re-coding is trivial because it is immediate. Interesting for education are the challenging cases. They require effort to master, development of new skills: algorithmic thinking. This is new! By the way, we have arrived at this position by analytically dealing only with computer-generated pieces. We may, of course, as well start from works without a computational background. The resulting algorithm would then not be a re-coding but a coding. Everything else would be the same. This may motivate us to search for a better name than *re-coding*. After all, our examples share this feature:

> **Take some works of art, radically analyze them and, perhaps, abstract from some given features; create an algorithmic description that can be transformed into code.**

Having established this, what about an *upper bound of re-coding*? The pictures of figures 5 and 6 are examples of which we claim they are *impossible to re-code*, if we take re-coding as the activity of constructing algorithms and programs capable of generating given sets of pictures.

The statement of impossibility is dangerous. Why should not someone appear with an advanced background in algorithmic art who, given enough time, would eventually find the level of analysis where our claim of unbreakability breaks down? We indicate this level in the case of figure 5.

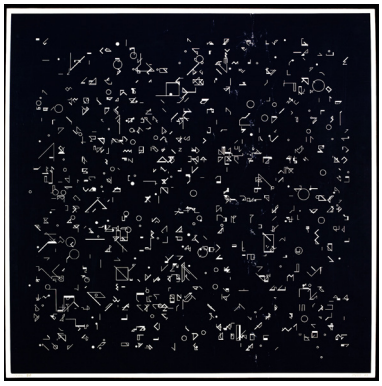


Fig. 3
Manfred Mohr:
A formal language (P-49)
1970. Credit: V&A
Museum London

Fig. 4
Manuel Barbadillo:
Composición modular
1965. Credit: compArt

We have developed the concept of *algorithmic sign* (Nake 2004) and, closely related to it, the concept of *surface and subface* (Grabowski & Nake 2005; Nake 2008). Everything that exists in a computational environment can be described as the *double of surface and subface*. Surface is perceivable by humans, subface is computable by software. Things in the world of experience are by necessity duplicated when they go through the doors of computability. Since almost every phenomenon nowadays gets computerized, there seems to be no escape. This perspective is particularly enlightening when we are dealing with works of art.

The image of figure 5 owes its existence to a Markov Process. Whatever that may be, the very moment we disclose this fact, our intelligent and diligent analyst, trying to derive an appropriate algorithm, sees a bright light shining on his problem. He will still need a lot of radical analysis before he finds a re-coding algorithm. But the hint contained in the name "Markov" radically changes his way of thinking.

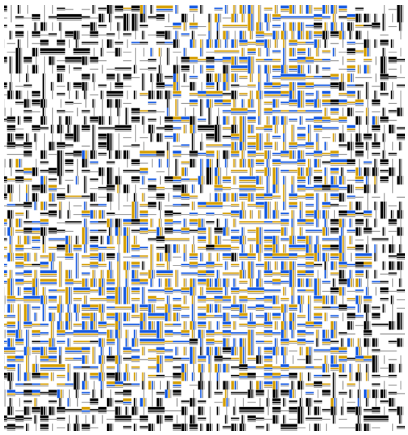


Fig. 5
Frieder Nake:
Abteiberg 2005

Fig. 6
Casey Reas: *Process 14 (Software 3)* 2012.
Credit: C. Reas

We do not want to describe any detail of this, particularly not since figure 5 originates in a third version of an algorithm one of us wrote in 1966. That algorithm, christened as *Walk through raster*, was exceptionally powerful insofar as the repertoire of signs to be used was itself a parameter (Nake 1974). This opened the algorithmic generation of pictures to a vast variety. But even more powerful, the underlying Markov logic was also parameterized into transition probabilities that depend on time (the Markov chain is non-stationary). The formal properties of the algorithm are such that a rather strong local control is capable of letting emerge global structural or compositional features. We believe, this was, fifty years ago, a remarkable discovery. Casey Reas's image of figure 6 is another example beyond the simplistic assumptions of naive re-coding. Without clues from the artist, no analyst has a chance to re-code it. For educational purposes, we have now identified the bracketing extremes, within which an exciting kind of new art education may happen! Neither the trivial nor the (almost) unattain-

able are interesting for education. Interesting is the possible, the challenging but realistic. That's the middle ground between the extremes.

4. EXPERIENCE

At the University of Bremen, as well as the University of the Arts in Bremen, we have offered seminars, workshops, and lab classes to students of digital media, integrated design, fine art, education, and computer science. Usually students of the same class come from different backgrounds. For their practical work, we ask them to do various kinds of re-coding projects.

We should not take the term, "re-coding", all too literally. In the context of art education, we mean it more in a sense of: Choose an artist from the algorithmic realm of artistic production, or from the more traditional side; carefully study some of his or her work; generalize just a bit of what you observe; take the result of your analysis as the starting point for your own algorithmic creation.

Such an approach may give freedom to students. They may, if they wish, try to get as close to the original as possible, or pay an homage to the artist, or generate something totally free and expressionist. Our hidden agenda, however, is that we want students to engage in processes between computation and intuition, between algorithmics and aesthetics, between determined structures and chance detail, prediction and surprise, strict and loose, open and closed. In other words, we see art education as a marvellous chance to approach the world as it is: Caught between poles of necessity and possibility. That's our view of reality, and it is not even our view alone. It is just real. Our experience is encouraging. [8] Students start on an analytical job trying to find out the parameters that control a certain image. These parameters become the class parameters determining the range over which the pictorial events may happen. Having established such analytical preparation, students are ready for the generative part of their task: They develop an algorithmic description of a potentially infinite set of images. This may range from almost trivial combinatoric pieces to the intractable where analysis fails (figures 5 and 6).

Susan Grabowski has lead students in an interdisciplinary seminar on art education and algorithmic art to develop analogue techniques that kids could use in their early years at school to simulate parts of Manfred Mohr's path in inventing repositories from which to create algorithmic works. Those kids would, of course, not know that they were simulating anything. They just had fun doing what they were doing, enjoying themselves as they were naively creating (figures 7 and 8).

We take up Manfred Mohr's early algorithmic experiments under the title, "A Formal Language" (figure 4). We have used them in a number of cases. We have already said that their structure belongs to the trivial end of algorithmic dialectics. But their elemental detail requires surprisingly detailed effort.

At the beginning of his path into the world of computation, Manfred Mohr wrote: "... old techniques of drawing and imagination are not to be imposed on the machine." (Mohr 1971) Computers should rather be integrated into the aesthetic system. This amounts to a position that accepts the computer as a partner (similar to Harold Cohen's position at the end of his life). Mohr formulates four premises:

1. A precise idea of an aesthetical problem.
2. The need to break this idea into parts ...
3. A steady control of the computing process to take full advantage of the

8
To describe this in detail, we don't here have enough space.

machine-human dialogue.
4. The need for the logic of the events to become perceptible. (Mohr 1971)

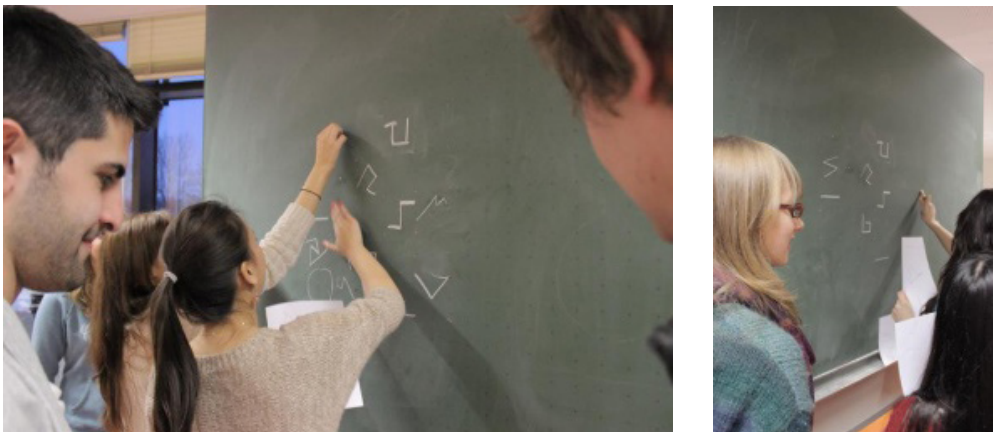


Fig. 7, 8
Students working toward a re-coding of Manfred Mohr's *A Formal Language*, University of Bremen 2012

In that same essay, Mohr concludes that single images are no longer in focus of an artist's work. The work is now determined rather as a set of images. That's the set an algorithm stands for.

One of our students, Kerstin Bub, has demonstrated how detailed and deep an analysis must be if we really want to get close to an artist's concept before we can start (re-)coding images in his honor. Mohr's images of the *A Formal Language* series are characterized at their bottom level by a finite repertoire of graphic elements. That's nothing very special or difficult. It belongs to the good heritage of early computer art. However, Manfred Mohr did not succumb to the seemingly narrow limits of early computing equipment. He did not restrict his own decisions to a small repertoire only of simple visual primitives. He rather thought of the computer under any computational command as a machine to explode into variety. And thus, his repertoire, taken as a schema, is simple, but it is complex when taken as an actual collection of *êtres graphiques*.

Kerstin Bub worked hard to reconstruct the repertoire from one of the realized pieces of *A Formal Language*. With and without the help of digital equipment, she defined levels of signs that shared certain geometric or graphic characteristics. Only on those seperate levels was she able, in the end, to come up with the complete repertoire that Mohr's algorithm is using. Figures 9 to 11 show steps from the analytic work of hers.

5. ALGORITHMICS IN ART EDUCATION

Algorithmic Art is meanwhile more than fifty years old, if we restrict the term to such works that have purposefully been designed and executed by dividing labor between the human inventor, designer, and selector on one hand, and the computer as a machine capable of carrying out semiotic processes on the other hand.[9] Early Algorithmic Art was called *Computer Art*, a term that now has become popular again. Fifty years were enough that today we often find doctoral theses and art historic as well as systematic research being done on this strong and influential sector of the world of art. Curators are no longer shy to include computer-generated contributions in their shows. In short, computer art is no longer an ugly child hiding in a corner. It is present everywhere. Ignoring it is a sign of stubborn conservatism. However, schools and high schools still seem to be a bit reluctant when it comes to letting algorithms appear in the art room. Those algorithms become the subject matter that must be designed in order to then itself engage

9
The computer is considered here as the semiotic machine, a view that is well justified, however only on a simple level of semiosis.

in designing works of visual art. There is an irritating indirection in the creative act. When art traditionally may be a wonderful hands-on experience, it now has been removed from immediate experience by an intervening algorithmic level. That's disturbing.

A hesitation may, thus, well be understood before schools are opening up the art room to let in the state of affairs as it has grown in the rest of social reality. We understand the well-justified reluctance against giving up a last refuge of emotion, intuition, empathy, wholistic views of the world. Such feelings may, at times, be due to an experience of insecurity or incompetence on the side of teachers.

Justified as such feelings may be, societies and their media have decades ago left the stable and trusted paths of security. Technical media have become ubiquitous and wearable. A look to the sidewalks of cities will show this overwhelmingly. Almost everybody there is playing around with their smartphone. How boring! In China, they subdivide sidewalks into parts with and without smartphone.

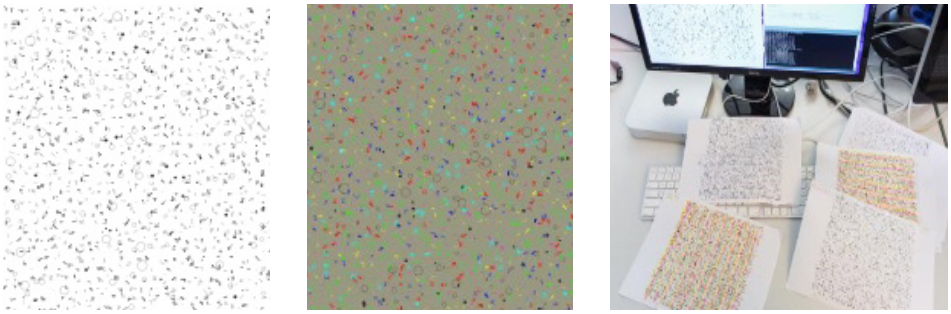


Fig. 9, 10, 11
Kerstin Bub: Steps from her scrupulous determination of Mohr's repertoire in *A Formal Language*. Credit: K. Bub

Digital media are characterized by computability, interactivity, and connectivity. They have become aspects of daily life for most of us, and certainly for the young generations. But must we not all understand much deeper what is happening when we voluntarily and happily jump on the next band-waggon, if we want to remain independent enough, at least in the self-propelled image of ourselves? Unless we all gain insight into what we have to do in order to make a computer do what we want it to do, we will be at the mercy of powers that we may not want to rely on. It is true that, even if we know what is happening, how and why, we are not yet saved from being kind of enslaved or, at least, dealt with like idiots. But without knowing the processes going on behind the scenes, we have almost no chance to stay independent from hidden mystic powers.

Everything in the world is in waves being computerized, which means: Processes and things are duplicated into surfaces and subfaces, perceivables by humans and computables by machines. What can be more beautiful, agreeable, and enjoyable than learning about social processes of enormous reach and depth, than in the context of art and artistic production!

Taking up algorithmics in the domain of aesthetics is a wonderful chance for art education. Not to ignore the algorithmics in aesthetics (and vice versa!) is a chance of historic dimension. It is a chance to bridge Snow's dichotomy. A chance to play with contradiction. A chance to view the world as what it is: always moving.

6. CONCLUSION

Algorithmic thinking is thinking in terms of algorithms, and that means that the end of algorithmic thinking is algorithms. They are precise, unambiguous, and effectively operational descriptions of complex activities. This sounds like the contrary of aesthetic activities, which are often associated with spontaneity, intuition, emotion, and expression. The history of art, however, offers quite a bit of rational methods and techniques also. One of the most glamorous cases was the invention of central perspective transformation, a mathematical procedure that is nowadays implemented in software.

The idea of bringing back to our attention algorithms that are now forgotten, but were once used to create early examples of algorithmic art, and, by doing this, save the beginnings of algorithmic art from being totally lost, is a nice and friendly idea. But it is helpless. Only for certain types of aesthetic objects can it work. We have identified them as trivially re-codable. On the other hand, many interesting cases cannot be re-coded if nothing extra is known. The re-code idea alone is not fruitful.

No causal relation requires to re-code in art education. It is not more than one approach among many. We don't suggest at all that the algorithmic dimension must enter art education. Re-coding may nevertheless be used in art education if we drop the idea of "save our cultural heritage". Instead, coding after analysis of given works has the potential of, at the same time, allowing for utmost precision and most open interpretation. Art education as an exercise in careful study and algorithmic formulation as the result of thinking algorithmically in the context of artistic production appears to us as a fruitful path into a kind of art education that is open for an art as expression of social developments and processes in contradiction.

REFERENCES

Epler, Matthew. http://mepler.com/CONTACT-BIO, w.d.

Epler, Matthew. http://mepler.com/The-ReCode-Project, 1973 (?)

Hertlein, Grace. Computer Graphics and Art. A magazine published quarterly from 1976 to 1978.

Mohr, Manfred. Computer graphics. Catalog of the exhibition "Une esthétique grogrammée". Musée d'Art Moderne de la Ville de Paris. 1971, p. 36-40

Nake, Frieder. *Ästhetik als Informationsverarbeitung*. Wien, New York: Springer, 1974.

Nake, Frieder. Das algorithmische Zeichen und die Maschine. In Hansjürgen Paul& Erich Latniak (eds.). *Perspektiven der Gestaltung von Arbeit und Technik. Festschrift für Peter Brödner*. München: Rainer Hampp, 2004, p. 203-233.

Nake, Frieder. Surface, interface, subface. Three cases of interaction and one concept. In Uwe Seifert, Jin Hyun Kim, Anthony Moore (eds.): *Paradoxes of Interactivity. Perspectives for media theory, human-computer interaction, and artistic investigations*. Bielefeld: transcript, 2008, p. 92-109.

Nake, Frieder, and Susanne Grabowski. Zwei Weisen, das Computerbild zu betrachten. Ansicht des Analogen und des Digitalen. In Martin Warnke, Wolfgang Coy, Georg Ch. Tholen (eds.): *HyperKult II. Zur Ortsbestimmung analoger und digitaler Medien*. Bielefeld: transkript, 2005, p. 123-149.

Reas, Casey, and Ben Fry. *Processing. A Programming Handbook for Visual Designers and Artists*. Cambridge, MA: MIT Press, 2nd ed., 2014.

Snow, Charles Percy. *The two cultures and the scientific revolution*. Cambridge: Cambridge University Press, 1959