Between Chaos and Order

David Schmudde
d@schmud.de
Beyond the Frame, Turin, Italy

Between Chaos and Order delves into the ephemeral aesthetics of cryptography by highlighting the place between input and output – the space of computation. Here the dual nature of encoded communication is on full display: as *source code* facilitating clarity and order and as a *secret code* meant to feign chaos. While the algorithms of secret codes are verifiable, their aesthetic and cultural context can make their physical realization indecipherable. This quality is demonstrated through an examination of three computational artifacts: Edgar Allan Poe's *The Gold Bug*, Alan Turing's notebook on cracking the Enigma machine, and the Electronic Frontier Foundation's publication *Cracking DES*. Each object is appraised through the lens of Information Aesthetics as a way to examine the relative importance of verifiability and aesthetics when communicating the computational process.

Introduction

Within the space of computation – after the instructions are read and before the outcome is provided – exists an actively undecided process. Most of this is hidden behind a screen of interactivity when using today's computers. But if we were to slow it down and watch the electrons move through the semiconductor materials as transistors rapidly switch states, we would see something that would look random and chaotic even though each transistor has a specific reason for being in either binary state at any given time.

We command these movements through machine instructions. These instructions are also known as *code*. While this noun is used in the context of communication (e.g. for communicating a set of regulations or communicating a series of commands), code can intentionally be used to either clarify or obscure a message. The latter is a *secret code*.

Source code, on the other hand, are instructions that have been encoded for machine consumption. Most computer languages strive for human readability and logical coherence. They can presumably be read by any capable interpreter. But instructions encoded using a secret code must be decoded before interpretation; they must be animated before they are consumed.

This aspect of cryptography makes it a unique computational artifact. The difference between random noise and encrypted information cannot be discerned at first glance. This essay considers three artifacts that explore this unique condition. The first is a story by Edgar Allan Poe called *The Gold Bug* which chronicles an obsessive effort to decipher a cryptic series of clues. The second artifact is the notebook that Alan Turing kept while attempting to crack the Enigma machine during World

Keywords Cryptography, Enigma, net.art, Turing, Poe, Preservation, Information Aesthetics, Computational Artifacts.
DOI 10.34626/2024_xcoax_019

War II. And the third artifact is a book published by the Electronic Frontier Foundation called *Cracking DES*. This book provides instructions on how to break a widely-used encryption algorithm – an act which put the researchers involved in legal precarity.

These three artifacts are feats of code breaking. But they are even more important as cultural objects. As such, they have aesthetic properties that can be considered through the theory of Information Aesthetics. Information Aesthetics is "a formalist, mathematical theory of contemporary art that worked to quantify the ratios between order and chaos, information and redundancy" (Patterson 2015, 75). Cryptography is the art of feigning chaos. If order is discovered within a transmission, then the cipher is broken; if it is truly chaotic, then there is no meaning to decipher.

The artifacts that embody the cryptographic algorithms similarly require analysis and deciphering. Are there formal properties of computational artifacts that can help in this effort? The two preeminent theoreticians behind Information Aesthetics, Max Bense and Abraham Moles, sought a method where "the effects of art would not only be programmable but also verifiable" (Quinz 2022), an effort that essentially mirrored the concerns of Information Theory (and later cryptography). All three of these computational artifacts communicate beyond the moment they were created. They are messages for the future. They have at least one verifiable claim – their techniques will correctly decipher encrypted messages coming from a certain cipher. The question is whether or not artifact's complete technical and aesthetic context help or hinder our formal understanding.

The Gold Bug (1843)

Poe's tale is centered on an obsessive treasure seeker who has secured a golden scarabaeus and a piece of parchment with this mysterious cryptogram:

```
53‡‡305))6*;4826)4‡.)4‡);80
6*;48†8¶60))85;1‡(;:‡*8†83(88)
5*†;46(;88*96*?;8)*‡(;485);5*†
2:*‡(;4956*2(5*-4)8¶8*;40692
85);)6†8)4‡‡;1(‡9;48081;8:8‡1
;48†85;4)485†528806*81(‡9;48
;(88;4(‡?34;48)4‡;161;:188;‡?;
```

The code is seven lines of nearly-symmetrical text with no spaces. The story's amateur cryptologist quickly derives the following character frequency chart:

Of the character 8 there are 33.

;	“	26.
4	“	19.
)	“	16.
‡	“	16.
*	“	13.
5	“	12.
6	“	11.
†	“	8.
1	“	8.
0	“	6.
9	“	5.
2	“	5.
:	“	4.
?	“	3.
¶	“	2.
-	“	1.
.	“	1.

The chart arranges the characters in the cryptogram from most frequent to least frequent. After it is established that the person who originally made the cryptogram was an English speaker, it is possible to overlay this chart with a frequency chart of characters in the English language to provide a clue for cracking the cryptogram. This process is enhanced by searching for common character groupings that might represent common words such as “the.” ;48 appears seven times with 8 as the most common character. Asserting that 8 is e, 4 is h, and ; is t provides enough traction to crack the rest of the code.

The first line, `53‡‡†305))6*;4826)4‡.)4‡);80`, thus reads: agoodglassinthebishopshostel, i.e. a *good glass in the bishops hostel.*

The Gold Bug presents a real cipher built for a fictional world. It’s the only cipher presented in this essay without a machine aid. While mechanical ciphers existed for hundreds of years before Poe wrote *The Gold Bug*, their omission highlights the natural link between the human mind, story, and the discipline of cryptography.

Furthermore, the systematic substitution of symbols holds profound implications beyond communication security. Alan Turing, in his groundbreaking 1936 paper “On Computable Numbers, with an Application to the Entscheidungsproblem,” explored automated symbolic processing as a strategy to address fundamental questions in mathematics. His approach demonstrated how the interpretation of symbols could directly influence a machine’s operations, effectively linking abstract mathematical logic with the practical mechanics of computing. This work led to the discovery the Halting Problem, which illustrates an inherent unpredictability in computing by proving that it is impossible to know whether certain programs will conclude without actually running them.

Any machine that can manipulate symbols based on the rules of logic could theoretically manipulate letters and words based on the rules of grammar. Turing successfully delivered a mathematical proof for the Entscheidungsproblem, but as a byproduct, he also created a theoreti-

cal framework with the potential to model aspects of human cognition – if one believes that language is a necessary component of thought.

Turing was not the only person to recognize the power of manipulating written symbols in the 20th century. From the Concrete Poets and Oulipo writers to the theorists in cybernetics, linguistics, and semiotics, these varying explorations of symbol manipulation suggest a broader cultural zeitgeist. Information Aesthetics can be considered as part of this 20th century milieu that studied the increasingly automated and concrete nature of language.

Part of the challenge is that the set of possible rules for modifying symbols is theoretically infinite. What makes Edgar Allan Poe's work in symbolic manipulation so aesthetically appealing is that it follows a simple substitution pattern. But greater sophistication requires meaningful constraints. De Mol, Bullynck, and Daylight argue that folks like Turing pulled from yet another line of thought they deem "Logical Minimalism" as a way to establish a set of minimum operations and axioms.

This balance of flexibility and constraint is at the heart of Turing's 1936 paper which in turn laid the theoretical groundwork for computer science. Turing later speculated on the potential for machine intelligence in his 1950 paper, "Computing Machinery and Intelligence." Such is the power of symbol manipulation and a few simple rules.

In between these pivotal contributions, Turing's expertise in symbolic manipulation made him a critical contributor to the team attempting to crack the Enigma code during World War II.

The Prof's Book (1940)

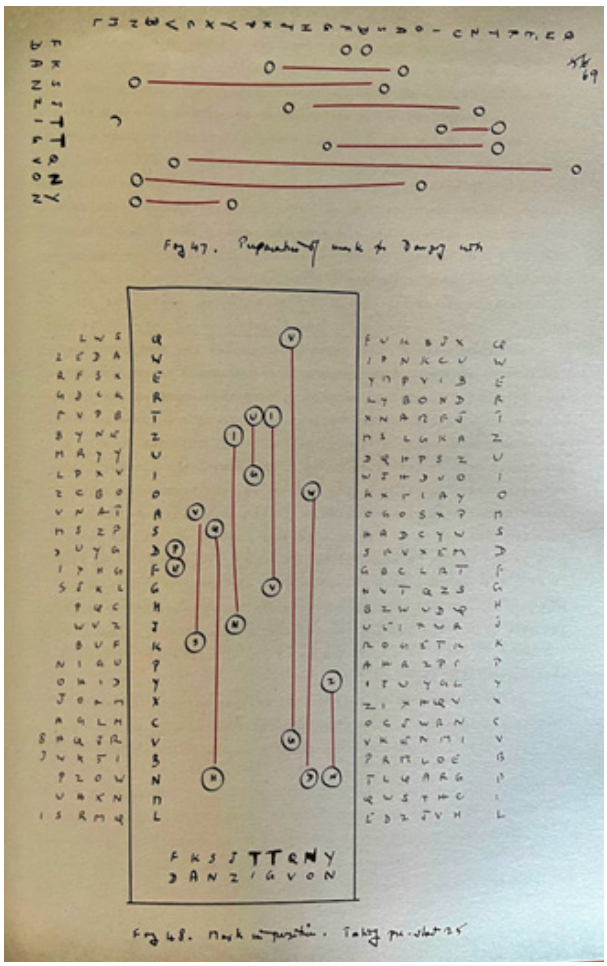

Fig. 1. *The Prof's Book* (Turing 1940, 69).

The most famous code breaking effort in the 20th century – the cracking of the German Enigma in World War II – thwarted a dynamic form of character substitution. For example, the characters DAEDAQOZSIQM-MKBILGMPWHAIV deciphered to KEINEZUSAETZEZUMVORBERICHT (*keine Zusätze zum Vorbericht)* in one particular setting of the Enigma on one particular day (Turing 1940, 97). But this string of characters could mean something totally different on a different day.

This success was first achieved with the help of a crib, a segment of cipher text where both the encrypted and the decrypted texts are known. By August 1939, it was determined that approximately 70% of the cribs used by the British were accurate (Turing 1940, 138). But this wasn't enough information to generate a key for the Enigma. Unlike the simple substitution cipher used in *The Gold Bug*, the Enigma's encoding process involved multiple rotors, each with an initial setting (the *Grundstellung*), rotor offset configurations (the *Ringstellung*), and plug boards settings (the *Stecker*). Decrypting a message required accurate knowledge of the Ringstellung, along with the correct Stecker configuration and the message's unique Grundstellung. Each keystroke adjusted the mechanical rotors' settings by a certain number of positions, ensuring that each character's substitution was dynamically set for the entire message.

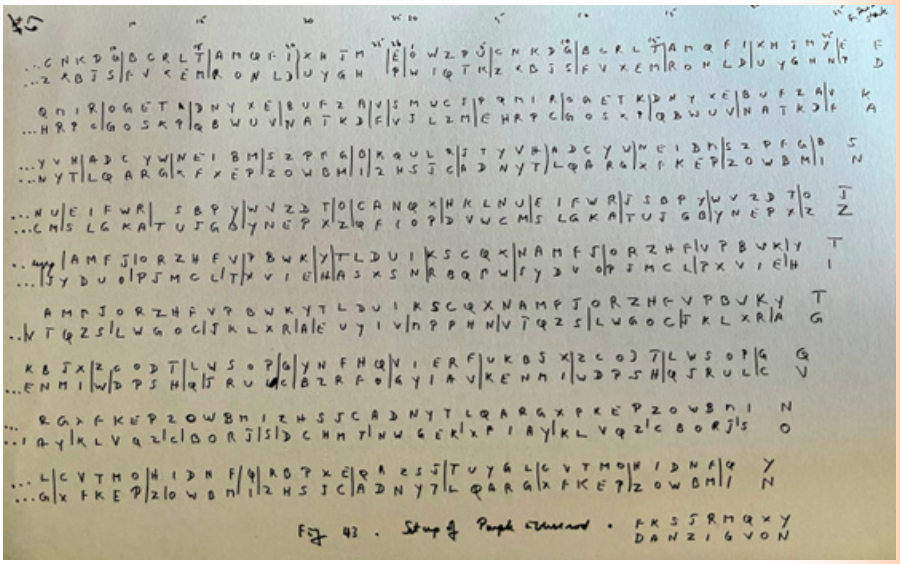

Fig. 2. *The Prof's Book* (Turing 1940, 67).

Early versions of the Enigma machine, many of which lacked a Stecker, were particularly vulnerable to cribs. The Polish codebreakers successfully exploited this weakness before the war. As Turing explains:

> *The Poles found the keys for the 8th of May 1937, and as they found that the wheel order and the turnovers were the same as for the end of April they rightly assumed that the wheel order and Ringstellung had remained the same during the end of April and the beginning of May. This made it easier for them to find the keys for other days at the beginning of May and they actually found the Stecker for xxxxx the 2nd, 3rd, 4th, 5th and 8th and read about 100 messages.* (Turing 1940, 136)

Further intelligence revealed that the Grundstellungen were not chosen at random and that Germans communicated numbers which

were spelled out in full. This pattern was crucial for decrypting additional messages.

> *It was never necessary to make a rack because when the 1938 messages were read it was found that the word EINS [the German word for one] occurred very frequently. We therefore made a catalogue of the encoded values of EINS at every possible starting position, and arranged the encoded values in alphabetical order. The unanalysed catalogue was made by enciphering first E at every possible position, then I, N and S. This was done with the automatic typewriting enigmas.* (Turing 1940, 140)

Note the use of mechanized aids to brute force combinations even before the invasion of Poland. Also note the inadvertent clues left behind by the Germans through their language and systemic tendencies. Much of this was initially mitigated by the additional security features added to the Enigma leading up to the war. The subsequent combinatorial explosion sent researchers scrambling for testable hypotheses that could at least generate some kind of confirmation or refutation.

Fig. 3. rmh40. "Enigma Machine at the Deutsches Museum." Photograph. Flickr, August 19, 2013. Accessed February 11, 2024.

New automation machinery was essential for reducing the problem space. While the Bombe remains the most ambitious, declassified documents detail compliments such as the Spider, which showed the "permissible wirings" between the different rotors (the Ringstellung) and plugboard settings (the Stecker). The codebreakers could then rapidly test different settings using the Bombe machine. The automated process was called "firing" the machine at the Enigma settings to see if deciphered characters were produced.

These code-breaking machines were destroyed after the war to ensure their secret technologies remained secret. What remains are the markings of the mind that created them. Alan Turing's notebook does not contain a mathematical treatise. It is a collection of narrative fragments and problem-solving schemes that are much less cohesive than Poe's fictional story. But the aesthetic qualities of these artifacts and any

machines that have been rebuilt continue to inform the interplay between culture and code.

This situation has been experienced by many digital artists, scientists, and archivists. As the net.art artist Cornelia Sollfrank opined, "It is very conceivable to transfer the algorithm [of the physical artifact] to a different and more up-to-date software. This would be one way to keep [my work] alive. Furthermore, we have many documents related to it, texts, this book, prints, a video, so the idea will remain alive" even though it "will no longer be the 'authentic piece'" (Sollfrank and Soon 2021).

Sollfrank made sure to mention the algorithm in the same breath as texts, prints, and video and separate from the concern of maintaining the original source code on original hardware. An engineer's schematics and programmer's code may seem like a complete guard against entropy but the artifacts are incomplete – and often even incomprehensible – without context.

The preservation of computational artifacts is a concern situated somewhere between Information Theory and Information Aesthetics. Time, the forcing function of preservation, is omnipresent in Information Theory. But the transmission of information across decades and centuries becomes an increasingly aesthetic concern because of the massive unpredictability of the receiver.

An algorithm by itself may be considered timeless. Its validity is often verifiable. But when it takes shape in our world – whether as a piece of fictional literature or in the reality of warfare – then it is subject to all the complexity of its context. This is true whether it is an exercise of the mind, as in Poe's story, or a mechanical process, as in Turing's notebook. Poe's simple cryptogram, for example, has been reproduced with numerous errors which have been perpetuated across various reprints through the decades (Giordano 2019). An Information Aesthetics framework would observe that the cryptogram itself uses many redundant symbols arranged in mostly novel patterns; the cipher is not chaotic in any sense. Hence why it is relatively simple to unlock the patterns and decode the message. But as the many erroneous *Gold Bug* reprints suggest, a verifiably incorrect cipher has little impact on the aesthetic enjoyment or popularity of the work.

Both Poe's place in culture and the readers themselves have changed significantly since the author was publishing cryptographic challenges in popular daily newspapers. The aesthetic value of the surrounding work – Poe's story and Turing's notebook – along with the contextual reputation of their authors are the great carries of the ideas behind their computational artifacts, even if the artifacts themselves fall victim to the passage of time. And this is something that Information Aesthetics cannot capture. Whereas Information Theory is successful because it eschews the complex analysis of a complete final message, Information Aesthetics operates exclusively on the complex final artifact (Nake 2012). Therefore we will always arrive at a place where a collection of analytical observations don't necessarily say anything substantive about the piece as a whole.

While Information Aesthetics is an incomplete tool, the exercise of assessing components of an computational artifact within a complete cultural context remains valuable. The computational aspect should be verifiable and contextualized if it is to be meaningful to the person encountering the artifact. The piece that perhaps gets closest to fulfilling

this aspiration is the Electronic Frontier Foundation's book *Cracking DES: Secrets of Encryption Research, Wiretap Politics, & Chip Design* (1998).

Cracking DES (1998)

The publication of *Cracking DES* emerged from a backdrop of prolonged governmental efforts to suppress cryptography research in the United States. This situation escalated when researchers sought to publish evidence that proved the vulnerability of the federally-approved and widely adopted Data Encryption Standard (DES). According to the researchers, the National Security Agency and the Federal Bureau of Investigation "pressured agencies such as the Commerce Department, State Department, and Department of Justice to not only subvert their oaths of office by supporting these unconstitutional laws, but to act as front-men in their repressive censorship scheme, creating unconstitutional regulations and enforcing them against ordinary researchers" (Electronic Frontier Foundation 1998, 4-1).

The United States' export controls forced researchers to register as arms dealers before publishing cryptographic techniques. Only after landmark cases like Bernstein v. US Department of Justice in 1995 were cryptographers able to openly discuss their work. But even then, the freedom to publish electronically on the World Wide Web or via File Transfer Protocol remained restricted into the turn of the millennium.

Cracking DES is the result of this absurd situation. This 272 page book contains the code and diagrams for all components needed to break DES. Since it was illegal to distribute the book electronically, a researcher would have to manually type in all the code to reproduce and verify the results. But one quick look at the original published C code will immediately reveal some peculiar syntax (Electronic Frontier Foundation 1998, 5-39):

```
cdaf5a
e1af5a
8538e5 /*
8f13e5 .* .Run the search. Uses the search parameters in the
ffec91 .* ......global linked list CHIP_ARRAY and keeps its
context info
c140a5 .* ......in the global CTX.
7c495d .*/
2fb622 void RunSearch(FILE *ctxFile) {
2944bc ..CHIP_CTX *cp;
2d049e ..SEARCH_CTX *ctx = &CTX;
79c4fb ..int halt = 0;
d4ceca ..time_t startTime, lastReportTime, t;
8cd6eb ..long loopCount = 0;
95431e ..char buffer[128];
c3af5a
c9fbd6 ..if (!QUIET) printf("Loading search context file…
\n");
578e14 ..OpenSearchContext(ctxFile, ctx);
45af5a
da37ac ..printf("Initialization Successful - Beginning
search.\n");
09a530 ..if (QUIET) printf("Quiet mode: Press ? for help
```

```
during search.\n");
7c2a59 ..if (FP_LOG && VERBOSE) fprintf(FP_LOG, "--- Begin-
ning search ---\n");
46ec5d ..for (cp = CHIP_ARRAY; cp != NULL; cp = cp->nextChip)
e4084a ..InitializeChip(cp, ctx);
9abe63 ..startTime = time(NULL);
155889 ..lastReportTime = 0
1daf5a
b005cf ..while (halt == 0) {
5ffb77 ....t = time(NULL); ............................/*
report every 5 seconds */
97eba6 ....if (t/5 != lastReportTime/5) {
e24d90 ......sprint(buffer, "%7ld blocks done, %7ld left,
%4ld running (time=%7ld).",
c347d2 ............ctx->totalFinishedKeyBlocks, ctx->to-
talUnstartedKeyBlocks +
16efa5 ............ctx->totalPendingKeyBlocks, ctx->to-
talPendingKeyBlocks,
db00a9 ............(long)(t - startTime));
889596 ......if (!QUIET) printf(">%s ('?'=help)\n", buffer);
751c3a ......if (FP_LOG && VERBOSE) fprintf(FP_LOG, "Report:
%s\n', buffer):
e61ab3 ......lastReportTime = t;
b36fe7 ....}
```

The rows of dots that precede each line helped the machines of the day make sense of how far to indent when using Optical Character Recognition to automatically input the code. *Chapter 4: Scanning the Source Code* from *Cracking DES* provides guidance to the reader and even suggests a specific set of scanning tools provided by Pretty Good Privacy, Inc – the same team behind the now-ubiquitous cipher suite.

This is the nature of code made concrete and ready to be automated, where even the spaces are meaningful and must be made literal. The physical book goes to great lengths to provide everything necessary to build a complete digital automatic code cracker. In this effort, the authors managed to make an artifact that embodies the technological, social, and regulatory environment of its time – arguably essential parts of any program that are too often overlooked (Lonati et al. 2022).

The code above provides a hint of what it's like to use the software. After loading a context file, it tells the operator that the computer is "Beginning search" and then updates the person on their progress. The software is searching a subset of all possible keys to try and find the right key. The technique is not unlike the Bombe built by Turing and his compatriots. In both cases, the number of tries in the best case scenario is reduced thanks methods that reduced the number of possible solutions.

Conclusion

The theory of Information Aesthetics considers the structure, complexity, and the mix of order and novelty an object presents to a consumer. Although mid-century theorists were considering works of contemporary art, one can admire the cryptographic artifacts in this paper through a similarly rigorous aesthetic lens. This shared spirit is rooted in the fact

that cryptography, as practiced, is concerned with the manipulation of symbols. The outcome may be theoretically secure. But attackers are clever and vulnerabilities may exist through the entire chain of enciphering, transmission, and deciphering. The context is just as important as the content.

The Gold Bug depicts a man deciphering a text that has presumably never been deciphered. The successful decryption required ample global context but the key itself was simple; it matched the letter frequency of the English language. *Cracking DES* also presents a complete solution where the key can be found to decipher any symbols encrypted with DES. This is a dynamic process that must be run with every new text. Running the software again in ten months or ten years requires ample global context that the artifact attempts to provide.

Turning's notebook does not present a solution, but it is the most pure rendering of a mind engaged with computation. It is filled with promising dead ends and many false starts. In this way it is a dynamic artifact searching the key space in real time – something that *Cracking DES* abstracts away into computer code.

When Fernando Domínguez Rubio ruminated "On the Discrepancy Between Objects and Things," he came to the conclusion "that things are constantly falling out of place." Sometimes they are valuable *objects* like a computer. But when the computer breaks it becomes a *thing* that goes into the garbage. Random text is just a thing until you discover that it is actually an *object* to be solved. This exact scenario is depicted by Poe in *The Gold Bug*.

Countless people have jobs separating objects from things. Museums are one example where this happens on an institutional level. But even with the best intentions, this process is somewhat arbitrary. The status of many artifacts are decided long after the people that made them are gone. Computational artifacts are extremely abstract and the software component is entirely metaphysical. Although no value is objectively self-evident, the cultural resonance of a computational artifact is easier to understand if its aesthetic properties help convey a more complete story. This is the least we owe the people that inherit our work. Otherwise, artifacts risk becoming obscured by a veil of entropy. Not because the artifacts no longer has meaning, but because that meaning can no longer be deciphered.

References

De Mol, Liesbeth, Maarten Bullynck, Edgar G. Daylight. 2018. "Less is more in the Fifties. Encounters between Logical Minimalism and Computer Design during the 1950s." *IEEE Annals of the History of Computing, Institute of Electrical and Electronics Engineers*. IEEE Annals of the History of Computing, 40(1):19-45. https://doi.org/10.1109/MAHC.2018.012171265.

Electronic Frontier Foundation. 1998. *Cracking DES: Secrets of Encryption Research, Wiretap Politics & Chip Design*. Sebastopol, CA: O'Reilly & Associates, Inc.

Giordano, Robert. 2019. "The Gold Bug Cryptogram." Poe Stories, August 21, 2019. https://poestories.com/discuss/the-gold-bug-cryptogram.

Lonati, Violetta, Andrej Brodnik, Tim Bell, Andrew Paul Csizmadia, Liesbeth De Mol, Henry Hickman, Therese Keane, Claudio Mirolo, and Mattia Monga.
2022. "What We Talk About When We Talk About Programs." *ITiCSE-WGR '22: Proceedings of the 2022 Working Group Reports on Innovation and Technology in Computer Science Education*, 117-164. December 29, 2022. https://dl.acm.org/doi/10.1145/3571785.3574125

Nake, Frieder.
2012. "Information Aesthetics: An heroic experiment." *Journal of Mathematics and the Arts*, vol. 6, no. 2-3, 65-75. https://doi.org/10.1080/17513472.2012.679458

Patterson, Zabet.
2015. *Peripheral Vision: Bell Labs, the S-C 4020, and the Origins of Computer Art*. Cambridge, MA: The MIT Press.

Poe, Edgar Allan.
1843. The Gold Bug. https://poestories.com/read/goldbug.

Quinz, Emanuele.
2022. "From Gestalt to Gestaltung: A Conversation with Giovanni Anceschi." In *Design, Gestaltung, Formatività*, August 2022, 133-146. http://doi.org/10.1515/9783035622447-010.

Rubio, Fernando Domínguez.
2016. "On the Discrepancy between Objects and Things: An Ecological Approach." *Journal of Material Culture*, vol. 21, no. 1 https://doi.org/10.1177/1359183515624128.

Sollfrank, Cornelia, Winnie Soon.
2021. *Fix My Code*. Eeclectic.

Turing, Alan.
2021. *The Prof's Book: Alan Turing's Treatise on the Enigma*. Kronecker Wallis.