Generative Media: sign, metaphor, and experience

Everardo Reyes-García

GENERATIVE MEDIA: SIGN, METAPHOR, AND EXPERIENCE

Everardo Reyes
Université Paris 8 Vincennes-Saint-Denis
ereyes-garcia@univ-paris8.fr

Abstract

This article explores the vivid field of generative media, focusing on the production and semiotic analysis of texts. It uses the broad definition of "text," which encompasses written, visual, and interactive forms, and examines how generative media redefines the roles of content creators and tools. Utilizing Roman Jakobson's communication model, the article highlights the dynamic decision-making process in text production, whether by human or AI. The paper offers a historical review which traces generative media from early computer art in the 1960s, through the advent of digital design tools in the 1980s and 1990s, to contemporary AI techniques like GANs and diffusion models. It identifies the key properties of generative media: synthetic, dynamic, digital, combinatorial, and agentic. The discussion also addresses the shift from unnoticed AI assistance in early tools to the inadvertently AI-generated content in today's media landscape. The last part of the paper categorizes generative media interfaces into three

Digital Age in Semiotics & Communication, Vol. VII, 2024, 62–79
https://doi.org/10.33919/dasc.24.7.4

types: conversational, web UI, and visual programming interfaces, analyzing their semiotic implications. It suggests that understanding these tools requires recognizing their layered technical components and the evolving user experience from magic-like simplicity to complex customization. In conclusion, the articles advocates for a multidisciplinary, process-oriented approach to fully grasp the cultural and communicative transformations driven by generative media, emphasizing the importance of transparency and user agency in AI interactions.

Keywords: generative media, generative AI, user interface, material semiotics, digital semiotics

Introduction

The recent explosion of generative media has attracted a plethora of users from all domains and disciplines. The unrestricted access in form of conversational bots to DALL·E and ChatGPT by OpenAI – in September and November 2022 respectively – proved that high quality content can be generated easily by anyone with a web browser. Many discussions around generative images, generative text, and generative media in general, place the accent on the content generated itself and its potential uses.

This article situates itself at the production stage of texts. A text is understood here in its broadest sense; as a series of syntactic, semantic, and pragmatical traits which cooperate together in the meaning making of instances. In this respect, the distinction between media supports - such as written text, speech, visual images, sounds, video, interactive works - will be useful only in respect to the material form of texts, not to their narrative and discursive content.

In order to produce a text, a sender has to take some decisions first. Following the well-known typology of linguist Roman Jakobson, the text can emphasize its origin, the receiver, the channel, the context, the message itself, or the metalanguage attributes (colours, forms, shapes, materials). In a scenario which considers generative media, the sender has the choice to address explicitly one of those elements, but also to delegate such will to the software program.

The questions which drive this contribution emerge from software artefacts where the capacity of generating texts predominate. How do they enhance or obfuscate the production of texts? What is the communicational relationship between a tool and its user? How can semiotics help to better understand the meaningful dialogues and exchanges which occur between sender and artefact to produce a text?

In the first part of the article, we review the notion of generative media from a historical perspective, especially in relation to the field of digital art, where it has found valuable exponents and perspectives. Then, we analyze current software applications for generative media from a semiotic standpoint. We focus on three types of interfaces: conversational interfaces, web user interfaces, and visual programming interfaces. We demonstrate a correlation of semiotic mechanisms available for each type of interface. While the entry-level is more natural language-based, the complexity of adjusting the capacities of the software become more visible at the mid and advanced levels. This is not surprising according to the levels of description following a semiotic trajectory (Reyes 2017). However, it is interesting to evoke the renewed metaphors and the acts of translation operating at the heart of these artefacts. In the last part, we offer our concluding remarks.

Understanding Generative Media

On the occasion of the 27th Summer School of Semiotics (Sozopol, September 2023), we insisted that current generative media meet five properties. First, it is synthetic in the sense that it is built artificially. Second, it is dynamic which means built on demand. Third, it is digital, i.e. built with computing techniques. Four, it has endless versions which emerge from connecting basic units in infinite combinations. Five, it has agency, from the perspective of reacting to the user input, like a dialogue between human and a multiplicity of artificial agents.

Levels of generation

Since the introduction of computer-assisted design and media software tools in the 1980s, users and media creators have been using AI-assisted technologies, sometimes in an unnoticed manner. For example, the popular "magic wand" tool introduced in Photoshop 1.0 (1989) allowed parts of an image to be selected based on regions of colours. Although the process of selecting regions relied more on image analysis than on AI techniques, the idea was to perform ,"magic" on images. The adjective magic was also present in another tool, the magic eraser, which allowed selected pixels to be converted to transparent upon clicking.

Another example of everyday content generation can be seen in web browsers. The list of links that appear when users launch a request is called dynamic content. In early search engines, such as Lycos and Yahoo, the results tended to be similar across computers, but today the links may vary depending on the geographical context, the user navigation history, and other non-visible information exchanges between web services (you can

try the add-on Lightbeam for Firefox to have a glance at those information exchanges). More recently, generative content has taken on the form of recommendations. We see suggested artists and playlist in Spotify, as well as films and videos based on our watching activity on YouTube and Netflix. E-mail clients such as Gmail also generate suggested text as we start typing a new email.

While software users may have not noticed that some of their tools were based on AI, today we consume content without any certainty that it has been generated by AI. Also since the 1980s, computer graphics techniques such as noise algorithms have been used, in order to create "very convincing representations of clouds, fire, water, stars, marble, wood, rock, soap films and crystal" (Perlin 1985). This means that instead of modelling a 3D model or a particle system requiring high amounts of computing power to render, the picture is a 2D image that generates only when needed. In our present time, the same could be said of backgrounds (matte painting), crowd masses, digital actors, prompts, assets, lights, scenes, voices, music, and editing.

A brief history of generative media

The term generative media can be traced back to a time before the introduction of cultural computing in the 1980s and 1990s (Manovich 2013). In the 1950s, early adopters of the term began doing artworks in form of images, films, kinetic sculptures, music, and literature.

Cybernetics is a starting point common to the variety of approaches. Inaugurated by the mathematician Norbert Wiener in 1948, cybernetics is a field dealing with control mechanisms within all types of systems, whether composed of similar, varied or hybrid entities. These control mechanisms are understood in terms of the relationships between components and the type of energy and information they exchange. For Wiener, an important aspect in this process was the external context of the systems. When notions like structure, process, and quantification were later applied in computing systems, they found a representation in visible form.

The impact of cybernetics in generative media was materialized in the 1960s. What is perhaps the first exhibition of computer art, entitled "Generative Computergraphik", was organized in Stuttgart in 1965. It included artworks by early practitioner Georg Ness. Later that year, artist Frieder Nake was also part of the program. At that moment, the word "generative" was associated to "art that was produced from a computer program and, hence, was at least in part produced automatically" (Boden & Edmonds 2009: 23). Generative art referred to "the activation of a set of rules (…) where the

artist lets a computer system take over at least some of the decision-making (although, of course, the artist determines the rules)" (Boden & Edmonds 2009:24). In this respect, other artists in the 1960s employed different terms to characterize their work: Manfred Mohr, called it "generative art", Max Bense "generative aesthetics", and Jack Burnham "process art".

Moving forward in time, in the late 1990s, the term recovered an invigorating momentum. After the explosion of media software such as Illustrator (1987), Director (1988), Photoshop (1989), After Effects (1993), 3D Studio Max (1996), or Maya (1998), artists and designers embraced the use of programming code. The term "generative design" is directly related to 2D and 3D graphics produced with code, rules, and grammars. The application Design By Numbers (DBN), created by scholar and designer John Maeda at MIT, inspired the development of a series of software in generative media: Processing (2001), DesignRobots (2003), NodeBox (2004), ContextFree (2005), and Structure Synth (2007). As we can see, in addition to generated output, a strong importance was placed on the processes, instructions, and computing methods conceived by the producer.

The following trend in our brief account involves AI techniques. In general, AI requires that the analogies between living systems and machines be looked at. An example is the image of artificial neural networks assimilated to an image of the human brain. Other specialized developments include agents, complex systems, artificial life (A-life), and genetic algorithms. An attempt to elaborate a definition of generative art which also includes AI was first proposed by scholar Philip Galanter in 2003 (refined in 2008 and republished in 2016): "Generative art refers to any art practice in which the artist cedes control to a system with functional autonomy that contributes to, or results in, a completed work of art. Systems may include natural language instructions, biological or chemical processes, computer programs, machines, self-organizing materials, mathematical operations, and other procedural inventions" (Galanter 2016: 154).

With AI and machine learning techniques, the term "generative" is used with a more technical background. For instance, the popular Generative Adversarial Networks (GANs) were introduced in a paper in 2014 to describes generative models and generators. Well-known artists such as Refik Anadol, Mario Klingemann, Trevor Paglen, and Memo Akten adopted GANs, in order to create images and films. Further advances in image synthesis have led to StyleGAN and diffusion models which have been implemented in popular tools such as Midjourney. Today, it is claimed that 34 million images are generated daily by all kinds of users across all AI services and platforms (Valyaeva 2023). Hence, the notion of "generative

AI" has gained attention, since it encompasses the capability of producing high-quality results out of endless connections and possibilities of the data that has been fed to the model.

Meanings of generative media

Throughout the history of generative art, artists and designers have reflected on the use of computing techniques to create their works. Broadly speaking, the computer has helped to better understand their own artistic practice and to think differently the expressive message to convey.

While computers and networks are becoming increasingly complex and distributed environments to create artworks, one of the first and most important contributions was the programming and generative possibility of computing systems. Early adopters such as Vera Molnar, Ernest Edmonds, and Frieder Nake created their own software procedures using the programming language Fortran. It was a novel perspective to consider an artwork as determined by its endless variations: the combination and recursion of visual parameters and, its responsiveness to the public or to external events occurring in the spatial context (changes in the temperature, the light, the heat, the amount of people in the room). Moreover, computing behaviours were also associated with cultural values. For example, "randomness", which basically adds a disordered series of events and numbers to a program, for an artist it was also seen as "chaotic" (Molnar), "indeterminate" (Edmonds), and "unpredictable" (Nake).

Returning to Jakobson's work, it is possible to locate meaning intentions at the level of each component of the communicational typology. On the side of the "sender" (i.e. the artist using digital tools), the intentions have been to "stretch the imagination, expand consciousness, and inspire others to new levels of creativity and invention" (Shanken 2009:15). With regard to the "context", artists are sensitive to the world they live in, criticizing the effect of technologies in culture and society. The "channel" (networks and media artefacts) is often repurposed for the sake of motivating strong aesthetic experiences. The "receiver" is the main goal to be reached, the observer to be seduced, the player to be touched, the visitor to be captivated, and the mind to be baffled.

Within a generative context, the originality of the artwork resides beyond a final outcome. Not only can variations be endless, but digital practices such as remix, glitch, and fork also make explicit the possibility of reusing materials to create further versions. Taking a look at generative art enables us to recall the importance of processes and procedural systems. In these terms, the meaning of writing procedures is to enforce "rules to gen-

erate some kind of representation, rather than authoring the representation itself" (Bogost 2007).

As already mentioned, early adopters had to made interventions at the level of the programming language, in order to express their visions. Adopting code as creative tool and choosing a programming language are choices related to questions of budget, platforms, and operating systems, but they are also a subjective decision. They influence what can be done and how easy or difficult it is to do it. In other words, they have the potential to frame a way of thinking. For our current discussion, we will suggest that the syntax of programming instructions may be somehow related to the syntax of prompts as they exist in recent generative AI services. Prompts have become a valuable asset in the production of generative media. We recently talked about prompt engineering, prompt programming, prompt design, and prompt modifiers. Despite the fact that one of the critiques that can be made to text-based prompting techniques is the necessity to use natural language to describe a desired output, the environment is based precisely on Large Language Models (LLM) and Natural Language Processing (NLP) techniques, in order to analyze and interpret text sequences. As we will explore, there are of course Large Visual Models (LVM) that can be used for different tasks and scenarios (Bommasani 2021). There is also a series of transformations which can be used to encode and decode text, and to align it to images and other media formats.

Semiotics of Generative Media

In this paper, we have made reference to cultural software which became predominant in the 1980s. We also mentioned web-based trends and quickly alluded to programming languages. In this section, our intention is to continue an exploration of tools for generative media. We ask the following questions: How are generative media created? Which software tools? Can we identify trends? What is the impact of such tools at the social and cultural levels? In our view, the transformation of a tool into a media involves the action of meaning-making processes that fall within the scope and import of semiotics. We relate three kinds of user interface to three levels of meaning implication with the tool, that is, generative media as sign, as metaphor, and as experience.

Digital tools for generative media

Well before the arrival of computers, artists and designers were well acquainted with random values and chance. Various artists in the abstract, conceptual, and contemporary movements employed dices, for example, in order to obtain aleatory results which were then translated into the artwork

proper. Stimuli from the environment were also a source of generating noise and new values (blowing wind, flowing water, animal movements). For our purposes, it will important to say that stochastic methods and other numerical techniques are considered digital tools when the internal system of a computer enables the translation of analogue signals into binary information.

The historical recollection presented in this paper has also a more technical approach which clarifies the relationship between software and hardware. To give an example, among the several programming languages used by renowned artist Frieder Nake in his career, ALGOL 60 served to create the custom drawing software Walk-through-Raster. In order to produce a visible picture, the procedure was tightly linked to a Graphomat Z64. Introduced by engineer Konrad Zuse in 1961, this machine was "an automatic drawing board that was controlled by punched cards" (Burbano & Garcia 2016). From this example, we want to make explicit several interconnected parts in the production of a generative artwork: a programming language, a software, punched cards, and a computer able to interpret them.

The further development of computing machinery led to what are known as compiler programs. Their aim is to translate instructions written in one programming language into assembly language. An assembly language is used for developing system software such as operating systems (OS), and it is specific to the type of hardware in the machine (for example, a certain kind of processor). An additional operation comes from the translation by assemblers into machine language or binary code. Although this description may sound far from the typical level of the users, the processes which occur below the interface are interrelated. We see their impact in form of incompatibility and obsolescence. Apple users, for example, noticed recently that some applications cannot be opened with the new Sonoma OS, since this is built for the new Silicon processors of the M series.

It can be said that such technical issues are hidden to common users thanks to software applications. Moreover, as the web becomes an increasingly powerful environment, browsers like Firefox and Chrome have become the main place where users access can share, store, and process information. Vendors using cloud services offer now online versions of their applications. We see an increasing number of online services which demand intense computing power not often available in local machines. In the following table, we summarize some of the most important components at different technical levels based on a web environment. For us, this account is important because the material levels determine different types of meaning implication.

Table 1: Material levels based on a web environment

Material levels	**Example of components**
Screen devices	Computer displays, tablets & smartphone screens
Screen objects	Media texts: visual text, visual images, videos… Graphical interface: windows, buttons, pointers…
Web development environment	Frameworks: React, Bootstrap… APIs: Google Maps, OpenWeatherMap… Libraries: Docker, Socket… Hubs: AWS, Github, Hugging Face…
Machine learning services	Models: LLMs, VLMs… Datasets Data Algorithms: sorting, search, hashing…
Middleware	Message Brokers, Databases, Servers, API Gateways…
Hardware	CPU, GPU, RAM, Hard Drive…

Levels of meaning

Starting from the definition of a sign in the Peircean tradition, it is well known that semiosis is the produce of three parts: the sign itself (representamen), the object, and the interpretant. Additionally, each part can be further described in three subparts, yielding to three trichotomies. In the second trichotomy, the signs in relation to its object, Peirce further elaborated a subtrichotomy for the icons, calling them hypoicons to signify that there is a substructure supporting an icon. Quoting Peirce: "Hypoicons may roughly /be/ divided according to the mode of Firstness which they partake. Those which partake of the simple qualities, are *images*; those which represent the relations, mainly dyadic, or so regarded, of the parts of one thing by analogous relations in their own parts, are *diagrams*; those which represent the representative character of a representamen by representing a parallelism in something else, are *metaphors*" (Peirce Edition Project 1998: 274). Figure 1 schematizes these 15 categories.

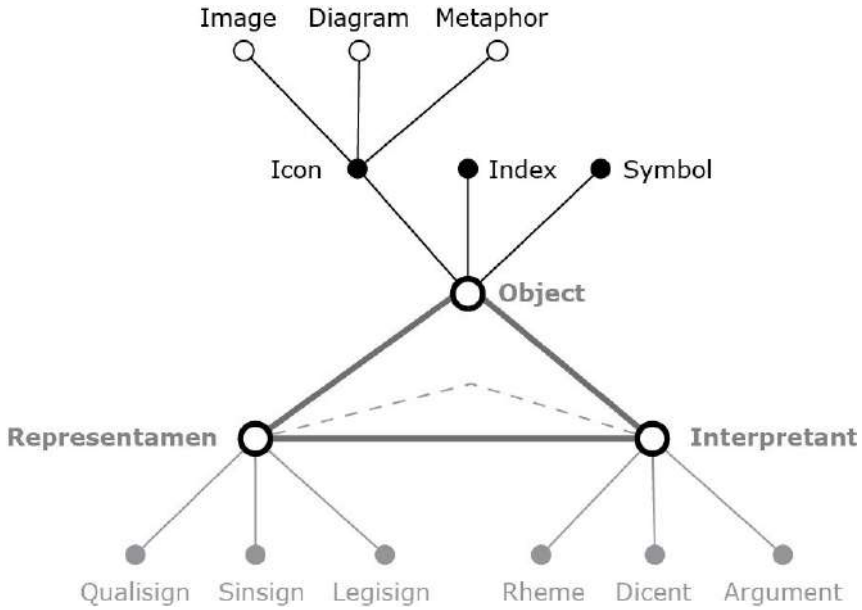

Figure 1: Peirce's trichotomies including 15 categories of signs

From this theoretical background, we can identify three levels of meaning implication in the production of generative texts with AI tools. From the perspective of the expression plane (i.e. the part of the meaning process that is perceptible), our levels go from computer signs (representamen), to digital metaphors (object), then to user experience (interpretant).

Concerning computer signs, they are the fundamental units of expression. They store bits in form of bytes, which then form data types (integers, strings, boolean). At a higher level, they can also be seen as arrangements which give form to building blocks (data structures, visual primitives, visual descriptors, interface elements).

Regarding digital metaphors, in terms of Peirce we consider them related to a subpart of iconic signs. In form of screen objects, they are configurations made out of the entities encountered at lower material levels (table 1). Thus, we talk about interface assemblies or configurations of interface elements. As such, they are described in terms of their constitutive parts: How are they made? How do they follow or challenge the expected experience of the user? How do they establish a communicative and semiotic relationship between human and non-human actors? This is paramount to us because screen objects host, locate, and define modalities of interaction: to believe, to do and to know (for an example of such modalities applied to face-recognition software, see Reyes 2021). Therefore, studying metaphors in the

digital domain requires not only an study of the material aspects of indexical icons (i.e. the graphical user interface), but also the conventions and ideologies that are evoked. Marianne Van den Boomen precisely situated discourse metaphors in the digital praxis (2014): more than just conceptual metaphors like cyberspace or electronic highway, discourse metaphors are also material metaphors in connection to technological artefacts (email, chatbots, generative AI tools). Metaphors reflect the pragmatical perspective of Peirce. They are active agents which bring something into being, rather than merely expressing a fixed idea for the sake of poetic or rhetorical effect. Similarly, Joanna Zylinska follows Matthew Cobb's suggestion that our scientific horizon depends on the technological metaphors we use at a given time. She goes further to ask: "Can we learn to see ourselves and our world better once we have changed our metaphors?" (Zylinska 2023: 151).

Finally, the experience level of meaning implication emphasizes the pragmatic aspect of the life of signs. Pierce claimed that once a sign spreads among the users, its meaning grows by use and experience. Scholar Thomas L. Short comments: "Experience thus leads to modifications of meaning. By unexpected consequences, good and bad, we learn more about what to expect from, or that can be done with, things of the type signified. Meaning is added to meaning" (Short 2007: 286). In our typology, experience enhances a mode of discovering, of learning, and making conclusions. In this respect, interface assemblies can be seen not only as interface features, but also as arguments, statements, and visions that reflect upon practices influenced by larger semiotic spheres, from artistic, educational and professional strategies, to digital culture and the digital way of life.

Generative media interfaces: three cases for analysis

Our typology of levels of meaning implication is derived from our own practice in the study of web-based tools and uses. In this part we have selected three cases for a semiotic analysis. Each case is associated to a level of meaning as shown in table 2. With this distinction that we do not claim a unique relationship between both sides. We rather sustain that these, and other levels of meaning, exist in a hierarchical mode. For our purposes, we believe that conversational interfaces, as they are ubiquitous in digital culture, simplify discussing the experience of a user that has access to a generative media tool like ChatGPT. However, a more knowledgeable user who dares to go further in the creation of generative texts, is rapidly confronted with environments like web user interfaces. Ultimately, it is also possible to discover more technical subtleties if a user unveils advanced tools where the computing signs can be manipulated with more flexibility.

Table 2: Cases and levels of meaning

Case study	**Level of meaning**
Conversational interface	Experience of users
Web user interfaces	Digital Metaphors
Visual programming interfaces	Computer Signs

Conversational interfaces

Conversational interfaces share similarities with prompt-based services. These are tools which accept the input of a user, commonly in form of natural-language text sequences. The output generated by the computer is in form of text or another media (as in text-to-image tools, for example).

Text-based and command-line interfaces (CLI) are in fact one of the first modes of human-computer interaction. They were introduced in the 1960s to replace punched cards. Today, we still use CLI every time we manage files or launch batch operations from the Terminal application in OSX, or with programming languages that include a CLI mode: Python, R, MATLAB, LISP.

However, the popularity of prompt-based interfaces is due to supporting a conversational scheme of communication. Conversation has the property of being "a context-dependent social activity that implies a potentially terrifying immediacy" (Hall 2018: 10). As one of the most important human activities, conversation as interface puts real-time and natural language interaction at the center of the design practice. We can see this characteristic in a prominent example already developed in the early years. The NLP (natural language processing) program ELIZA, developed by computer scientist Joseph Weizenbaum at MIT in 1964, consisted of a conversational interaction between a human user and a programmed simulated psychotherapist. Participants were immersed in the conversation partly because the systems was always available to interact.

Conversation has been present in chatrooms, email clients, games, SMS, and text messaging applications. In her study, designer Erika Hall (2018) makes the difference between texting and writing and how they define particular modes of conventions (the former being more informal, while the latter being more compositional). Typing can be seen as "fingered conversation". A similar trend is noted by computer scientist Jensen Huang (CEO of Nvidia), who sees the interaction with chatbots as an act of programming: "everybody can program ChatGPT… because human language, natural language, is the best programming language" (Huang 2023).

When features such as immediacy, real-time responses, ubiquity of access (mobile, web, desktop, sunglasses, IoT), permanent availability (anytime), language understanding (any topic, but also disregarding input errors or misformulations from the user), voice recognition, voice synthesis, are included in conversational interfaces targeted to general public, a commercial message is to think of the user experience as magic.

As we saw in previous sections, the adjective magic already existed in tools like the "magic wand" in late 1980s. Today, web apps and digital tools increasingly include the magic icon as a visual element that indicates the presence of AI-driven features, a special functionality meant to surprise and delight the user and to enhance the overall experience. The "magic wand" conveys broadly creation, transformation and content generation. Besides this indexical icon, we can cite others. The "sparkling pen" that stands for text generation or enhancement, the "starburst palette" that suggests artistic or design generation, the "glowing microphone" for AI-generated audio or voice content, and the "enchanted book" that represents AI-generated storytelling or content creation.

Web user interfaces

While using commercial generative AI tools might indeed feel like magic, the second type of interface we bring forward concerns the production of generative media with web-based UI tools. These tools confront the user with a more complex environment which offers the benefit of adjusting some parameters that exist under the hood of the magic hat. In this scenario of use, the role of the sender is typically played by a company, or a research team, or an educational group who wants to customize the predefined behaviour of a given solution. For example, it could be possible to design a tool with specific knowledge in a given discipline, or to integrate a different model, or training data. In such situations, the producer assembles the technical possibilities of the tool in form of a graphical interface and makes them available to its readers (members of the company, the research lab, or students in class, for example). We agree with scholar Peter Bøgh Andersen with regard to the role of the designer as sender of computer-based signs: "a designer is an indirect sender, creating the structure for processes in which the interface is manifested" (Bøgh 1997: 184).

Nowadays, graphical interfaces are seldom hard-coded. The common practice is to use a toolkit that puts together pre-defined groups of elements. Let us consider a popular open source framework like Bootstrap, originally developed by Twitter in 2011. It exploits style rules in CSS together with interactive events in JavaScript and offers various blocks of

containers, layouts, forms, and other components. In web environments, the HTML elements are ultimately consistent, i.e. input fields, buttons, text areas, while their appearance changes according to the framework in use. In this line, ChatGPT has never publicly detailed its framework; it rather evokes a combination of them: Tailwind CSS, React, and Bootstrap.

With the introduction of the web as platform, robust online development environments have appeared. From creative coding (e.g. p5.js) to machine learning (e.g. Jupyter), generative media interfaces can be designed directly on the web. In this domain, a well-know user-interface library is Gradio, established in 2019 as an open source project, then acquired by Hugging Face in 2022. As a Python library, Gradio can be used in other services such as Google Colab or Jupyter Notebooks.

A quick look at Gradio's rationale allows to see that its prebuilt interface components are specially configured for AI tasks that output the result in a web-supported media format (text, image, video, audio). Interface components are grouped in blocks that maintain a functional coherence, for example a text box and a button in the same bind convey interrelated functions. Gradio also dedicates a high-level component to create chatbots. The Gradio framework has gained popularity because it integrates an environment that allows to create, test, access, copy, and edit the source code, to share the interface, to test different models, and to store it free at Hugging Face. By June 2024, Hugging Face reported more than 500 thousand spaces created by the community of users. Moreover, besides LLM applications and conversational design, Gradio is extensibly used in LVM (Large Vision Models). To mention an example, Gradio is the framework used by Automatic 1111, the standard user interface for Stable Diffusion WebUI. Image generators are a case that challenges complex and customizable support for UI frameworks.

Overall, user interfaces for generative media adhere in part to the notion of visibility of the process. They show a combination of numeric values and technical vocabulary to have a glance at the variables that affect an output.

Visual programming interfaces

Our third case involves a closer interaction with computer signs. In this case, senders extend their role as producers, developers, designers, and creators of texts. Following the variety of material layers underlying a generative media process (table 1), a deeper involvement with computer signs adopts the principle that components can be considered as texts. This means that a sender is not only able to articulate prompts, but also user interfaces and potentially datasets, frameworks, models, and APIs.

In our view, one way to take a look at the manner in which the components of material layers are interrelated is to observe tools and environments which reflect the state of the program and the flows of data among its components. For this matter, visual programming interfaces provide a rich field of study. Indeed, visual representations have been of interest to computer scientists working on CGI, image processing, computer vision, and spatial hypertext. Common visual objects for visual programming include flowcharts, blocks, nodes, curves, and shapes that can be manipulated in two dimensions. In terms of their utility, many studies focus on their more natural way of interacting, especially for educational purposes (Jiang 2023): constructing instead of writing a program; drawing lines instead of typing coordinates. Other studies refer to their added value to specialized users: a visual program tends to be a higher-level description of more abstract processes difficult to see. Such a program could also present more information about the state, the variables, and data structures in use (Myers 1990).

As we recall from precedent sections, node-based tools became popular in the early 2000s. They appeared as stand-alone generative design applications (e.g. NodeBox in 2004), and as modules integrated in media software (e.g. Grasshopper for Rhino in 2007). Today, node-based programs remain widely used in the creative community (Pure Data, MAX/MSP, TouchDesigner, Blender Nodes, Unity VFX Graph, among many others). In its visual form, a program is diagram made of nodes interconnected with lines. Nodes and lines are graphical elements that can be used as visual variables (shapes, colours, position), representing meaningful computing concepts (classes, operations, relations, imbrications)

In the arena of generative AI, it is not surprising that text-to-image and image-to-image models, with their applications in design, illustration, creative imagery, and marketing, have adopted graphical programming techniques. While major services such Midjourney, Dall-E, Runway, or Leonardo AI propose a commercial, web-based, GUI-based environment, Stable Diffusion gained acceptance among the open source community when the company Stability AI released version 1 in Open RAIL license in 2022. Besides DreamStudio and Stable Assistant, two products commercialized by Stability AI, Stable Diffusion can be installed locally to generate unlimited numbers of workflows and images for free. As already mentioned, Automatic 1111 is a common GUI, but ComfyUI rapidly became the preferred interface to create complex workflows yet in the intuitive logic of visual programming.

To give an example, each box in ComfyUI represents a node. There are several categories of nodes: input nodes (text, images), processing nodes (KSampler, CLIP Text Encode), output nodes (Save image, Preview image), model nodes (Load Checkpoint), among others. The left and right sides of each node have one or more inlets and outlets. A user can draw a line from one inlet to an outlet. Lines have different colours depending on the data value. Furthermore, inside the box of each node, there is a list of parameters and values that can be adjusted. All these visual options have been customized to meet the machine learning routines. In the background, ComfyUI's interface is based on LiteGraph, a Javascript library to create graphs. Although ComfyUI is based on Python, it launches in a virtual server accessible from a web browser.

Routines constructed in ComfyUI can be saved and shared. They can be executed in the same way, on condition that all libraries and models exist in the local machine. This is important to note because users of this environment need to be well informed about deeper material layers. Installation requires instructions to be run in the command terminal but also to have modern GPU, CPU, RAM, OS, and disk space available (the size of the recently announced Stable Diffusion 3 is more than 20 GB including text encoders, which are used to manage the translation from text to image in the latent space).

Conclusion

In this paper we have explored generative media from different perspectives: as digital material, as meaning implication, and within a historical context that started in the 1960s, which has risen a series of interpretations and experiences. We adopted the point of view of the producer, i.e. the creator and sender of messages.

We criticized the idea of generative AI as magic, also reinforcing the notion of computer systems as a black box machine. At the present moment, one of the most advanced levels of user interaction in generative media implies to know exactly what is happening in the generation process. However, one difficulty which emerges when dealing with GPT models is their increasingly production of unexpected results. Computer scientist and author Ronald Kneusel has referred to some "happy accidents" that were not intended by the GPT-4 model but are nonetheless accurate (like generating computer code in a language that has little presence on the web) (Kneusel 2023).

Our contribution builds upon a material perspective of signs and cultural traces, and it calls for a multidisciplinary and experimental attitude, in order to grasp the complexity of digital reality and culture. We believe it

necessary to encroach upon the field of more specialized users, in order to acquire a more complete vision of digital culture. A process-oriented view is of particular help. It looks at software in connection with human learning and communication, and in relation to the use context: "system objects are seen as the expression plane of signs whose content is generated in the work context" (Bøgh 1997: 186). In an attempt to better understand interfaces, it seems productive to consider them as texts. First, interface elements exist as programming code. Second, visible texts in the GUI are shown in natural language. Third, interfaces follow a model that is only perceived through their visual organization and the actions it allows to do. Interfaces can then be seized as enunciation acts, as objects that reflect design choices, as pieces of culture that will endure (or not) as the system is used.

Acknowledgements

Research in this article was first presented as an invited lecture at the Summer School "Conceptualizing Digital Reality through Metaphors: Semiotic and Interdisciplinary Perspective," organized by Professor Kristian Bankov in Sozopol, Bulgaria, on September 7, 2023. I appreciate the enriching discussions that helped develop this text. Special thanks to the ERUA staff for their administrative support.

References

Boden, M., E. Edmonds. 2009. "What Is Generative Art?". *Digital Creativity,* Vol. 20, No. 1–2, 21–46. Available at: https://doi:10.1080/14626260902867915.

Bøgh Andersen, P. 1997. *A Theory of Computer Semiotics*. Cambridge University Press.

Bommasani, Rishi, et al. 2021. "On the Opportunities and Risks of Foundation Models." Report. Center for Research on Foundation Models (CRFM), Stanford Institute for Human-Centered Artificial Intelligence. arXiv preprint arXiv:2108.07258. Available at: https://doi.org/10.48550/arXiv.2108.07258.

Burbano, A., E. G. Bravo. 2016. "Konrad Zuse: Enabler of Computational Arts?." In *Archiving and Questioning Immateriality, Proceedings of the 5th Computer Art Congress*. Paris: Europia, 190–203.

Franco, F. 2022. *The Algorithmic Dimension: Five Artists in Conversation*. Cham: Springer.

Galanter, P. 2016. Generative Art Theory. In Paul, C. (ed.). *A Companion to Digital Art*. Hoboken: Wiley. 146–180.

Hall, E. 2018. *Conversational Design*. A Book Apart.

Huang, J. 2023. *NVIDIA Keynote at SIGGRAPH 2023*. [Video] Available at: https://www.youtube.com/watch?v=Z2VBKerS63A.

Jiang, P. 2023. "Positional Control in Node-Based Programming." In *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems*, 1–7.

Kneusel, R. 2023. *How AI Works: From Sorcery to Science*. London: No Starch Press.

Manovich, L. 2013. *Software Takes Command*. New York-London: Bloomsbury.

Myers, B. 1990. "Taxonomies of Visual Programming and Program Visualization." *Journal of Visual Languages & Computing*, Vol. 1, No. 1, 97–123.

Peirce Edition Project. 1998. *The Essential Peirce: Selected Philosophical Writings Volume 2 (1893–1913)*. Indiana University Press.

Perlin, K. 1985. An Image Synthesizer. In *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '85)*. New York: ACM, 287–296. Available at: https://doi.org/10.1145/325334. 325247.

Reyes, E. 2017. *The Image-interface: Graphical Supports for Visual Information*. Hoboken: Wiley.

Reyes, E., S. Labelle. 2018. Seeing Through the Web: Tools, Practices, Transformations. In *Proceedings of the 2nd International Conference on Web Studies*, 1–4.

Reyes, E. 2021. Face Value: Analyzing and Visualizing Facial Data. In *Lexia. Rivista di semiotica, 37–38*, 467–483.

Shanken, E. 2009. *Art and Electronic Media*. London: Phaidon Press.

Short, T. 2007. *Peirce's Theory of Signs*. Cambridge University Press.

Valyaeva, A. 2023. "AI Has Already Created As Many Images As Photographers Have Taken in 150 Years. Statistics for 2023". *Everypixel Journal*. Available at: https://journal.everypixel.com/ai-image-statistics (accessed 31 May 2024).

Van den Boomen, M. 2014. *Transcoding the Digital: How Metaphors Matter in New Media*. Amsterdam: Institute of Network Cultures.

Zylinska, J. 2023. *The Perception Machine*. Cambridge: MIT Press.