

Sculpting spaces of possibility

Brief history and prospects of artificial intelligence in design*

Daniel Cardoso Llach

This chapter offers a brief history of, and new directions for, artificial intelligence (AI) in design. Drawing together examples from architecture, music, and the visual arts, it shows how an experimental tradition of computational aesthetics and design underpins present-day approaches to architecture and AI. In exploring this history, the chapter calls attention to two aspects that have frequently been overlooked: first, the materially specific acts of codification that structure these experiments, and second, the reconfigurations of author and user roles that they entail. With these as points of reference, the chapter identifies new directions for creative research on AI and design that eschew simplistic approaches to style transfer or machine autonomy. As illustrations, three recent projects developed at the Computational Design Laboratory at Carnegie Mellon University are discussed. These explore new drawing interfaces that enable users to manually control the weights of a neural network; new approaches to computational urban analysis that reveal morphological gradients at the urban scale; and “distant readings” of architectural data that open up new approaches to design description, analysis, and generation.

Despite the aura of novelty that still accompanies artificial intelligence (AI) methods in popular and academic discourses about design, dreams of adaptive design machines and autonomous creativity have long populated the imagination of computationally minded architects, designers, and artists—and that of artistically inclined mathematicians, scientists, and engineers. Take for example computer scientist Frederick Brooks and his collaborators at the Harvard Computation Laboratory, who developed in 1957 a method for generating musical tunes automatically by numerically encoding and analyzing the structure of 37 sample melodies. Aided by a digital computer, the team counted the number of occurrences of specific melodic sequences, creating frequency tables that described this dataset probabilistically (see Figure 2.1, left). For example, a 0.2 probability of the note A following the musical sequence C-E-G meant that the note A follows two in ten occurrences of the C-E-G pattern in the set. So, if a computer program generates a pseudorandom number, and it falls within the 0 and 0.2 range, it could compose the sequence C-E-G-A (Brooks et al., 1957). Brooks and collaborators created such a program and generated new melodies that fit with the musical pieces in the original dataset. This quintessentially inductive process is nicely illustrated in Figure 2.1, right. By putting frequency tables and rudimentary statistical analysis in the

TABLE I
SORTED OCTOGRAMS, ILLUSTRATING THE FORMATION OF THE
CUMULATIVE PROBABILITY TABLES

Cell	Octo-gram Count	Hepta-gram Count	Relative Frequency	Cumulative Probability
1 2 3 4 5 6 7 8				
36 37 26 27 32 33 26 22	1			
*36 37 26 27 32 33 26 22	2		2/8	2/8
36 37 26 27 32 33 26 27	1			
36 37 26 27 32 33 26 27	2			
36 37 26 27 32 33 26 27	3			
36 37 26 27 32 33 26 27	4			
*36 37 26 27 32 33 26 27	5		5/8	7/8
*36 37 26 27 32 33 26 28	1	8	1/8	8/8
36 37 26 27 32 33 32 33	1			
*36 37 26 27 32 33 32 33	2	2	2/2	2/2
*36 37 32 33 32 33 32 33	1	1	1/1	1/1
36 37 32 33 32 33 36 37	1			
*36 37 32 33 32 33 36 37	2		2/3	2/3
*36 37 32 33 32 33 36 42	1	3	1/3	3/3

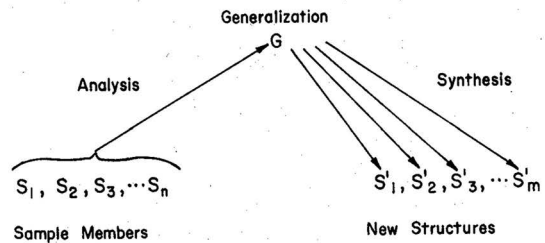


Figure 2.1 Left: Frequency table counting the occurrences of melodic patterns in a dataset of musical compositions. Right: A process diagram explaining the inductive nature of the process (Brooks et al., 1957).

service of generative inference with the intent of producing an artistic result, Brooks and his collaborators anticipated present-day applications of AI—and more specifically machine learning—in art and design. The term “machine learning” was not coined until 1959, and thus it was not available to Brooks and his coauthors. However, invoking computer scientist Claude Shannon, they used a language that unequivocally anticipated this paradigm of computation: “[m]any interesting computer experiments in game playing, ‘learning,’ theorem proving, etc., have been aimed at discovering methods of simulating rudimentary inductive processes with a computing machine” (Brooks et al., 1957, p. 175).

Another early explorer of computer music, US chemist-turned-composer Lejaren Hiller, helps illustrate how the aspiration to simulate inductive reasoning was central to some early expressions of computational art and design. In the late 1950s, Hiller and his collaborators at the University of Illinois modified a computer program used in chemistry for calculating the geometry of molecular bonds, adapting it to generate melodic lines by transposing the structure of tetrahedral carbon bonds into the structure of musical counterpoints (Bewley, 2004). The program generated melodic samples using the Monte Carlo method, a stochastic process weighed to select results conforming to predefined attributes (Hiller & Isaacson, 1959; Zaripov & Russel, 1969). In theory, these attributes could be derived from the statistical analysis of music by other composers—not unlike Brook’s experiment. However, these researchers followed a different approach. They encoded style rules in smaller pieces of code called subroutines, which specified musical characteristics concerning the melody, harmony, and structure of the piece. Random note sequences generated by the program would be checked for compliance with rules specifying, for example, that “no tritones are permitted,” or that “the melody must start and end on middle C,” or that “the range of the melody from its highest to lowest note must not exceed one octave” (Hiller & Isaacson, 1959, p. 75). If the sequence conflicted with any of the rules, it was rejected, and a new sequence was automatically generated.

At first glance, it is difficult to reconcile Hiller’s work with present-day approaches to machine learning, which tend to relish enormous datasets and seemingly obscure algorithms. After all, instead of generalizing from examples, it appears to rely on a fixed set of predefined style rules (Figure 2.2). However, if we look closely, it becomes clear that the rules themselves result from a process of observation and analysis of examples—which are just other words for “learning.” For example, the rules of strict counterpoint encoded in the program were

EXPERIMENTAL DETAILS

85

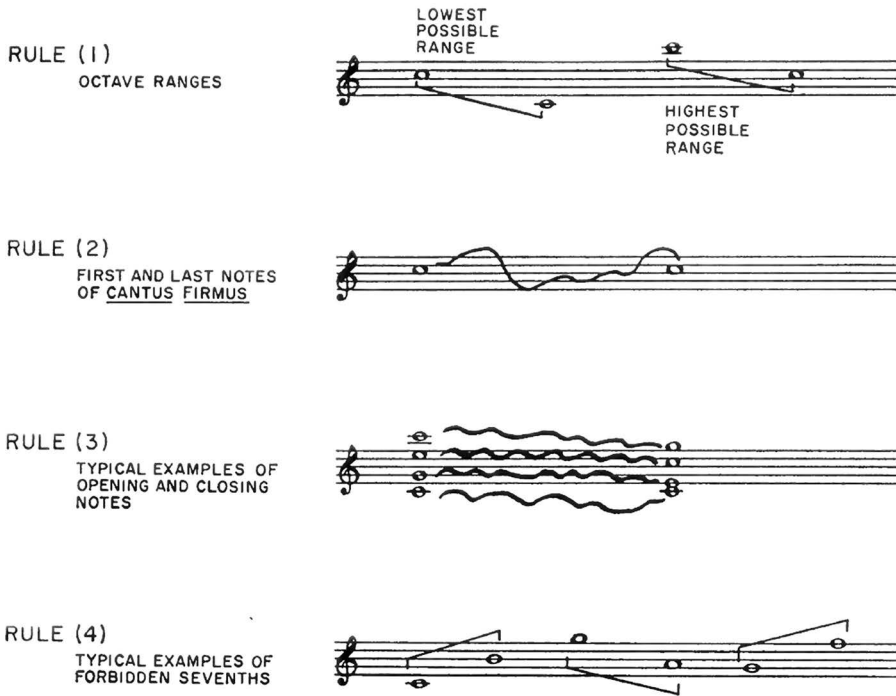


Figure 2.2 Early experiments in rule-based musical composition sought to reproduce stylistic traits identified in a set of melodic samples (Image credit: Hiller & Isaacson, 1959, p. 85).

derived from methods of composition created by 18th-century Italian composer Palestrina (Hiller & Isaacson, 1959). In this precedent from the Renaissance, the US researchers found “a logical abstraction of many important elements of musical structure and form a fundamental basis for handling linear melodic flow” (Hiller & Isaacson, 1959, p. 83). If Brooks used a probabilistic description of a curated set of sample melodies to shape the melodic outcomes of a computer program, Hiller used stylistic rules derived from the work of a composer to constrain brute-force algorithms to the same effect.

Researchers in architecture and the visual arts confronted a similar question: How to shape, or sculpt, the vast spaces of possibility afforded by computational methods and generate meaningful outcomes? However, they dealt with a different material. While musical scores could be represented as one-dimensional sequences of numbers, shapes, colors, and volumes, architecture and the visual arts seemed to demand a different approach. Some of the best-known examples of early computer art and design rely on the codification of the cartesian plane and the exploration of constrained randomness and pen plotters as aesthetic devices (Caplan, 2020). Other theorists and researchers imagined visual and architectural designs as language structures comprising vocabularies of elements and sets of grammatical rules for their combination. US mathematicians and design theorists George Stiny and the late James Gips, for example, defined design computationally as a visual algebra where shapes themselves are the units of calculation (see Figure 2.3)—a theory and a praxis they called

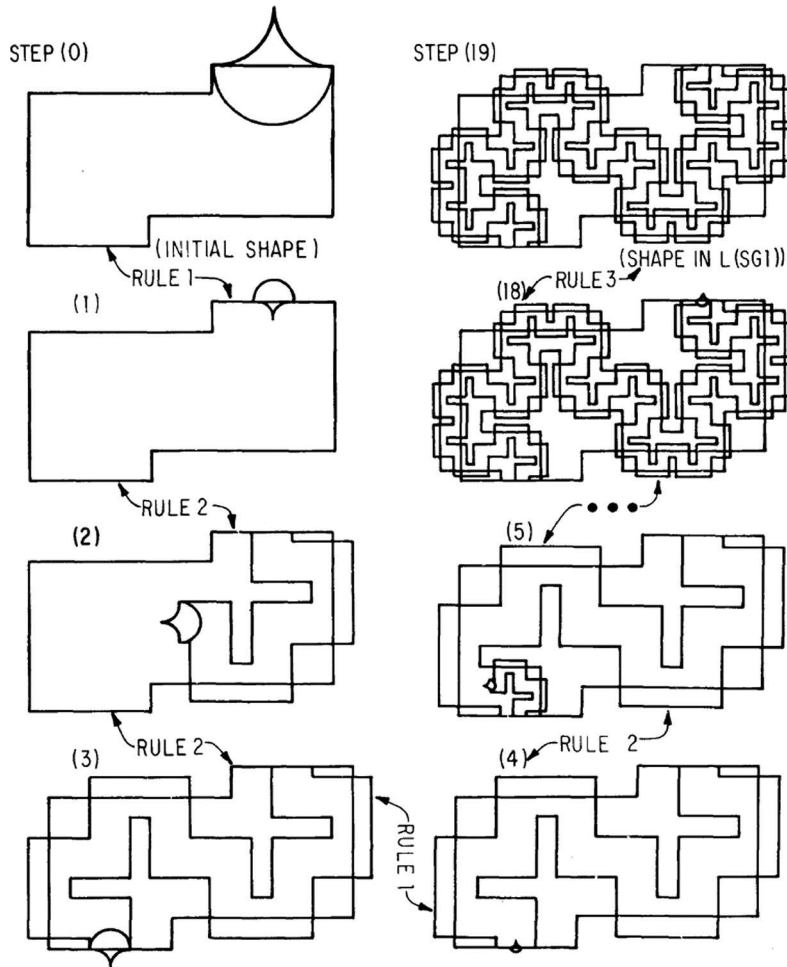


Figure 2.3 United States mathematicians and design theorists George Stiny and James Gips imagined design as a visual form of computation with elements called “shapes,” structured by grammatical rules (Image credit: Stiny & Gips, 1972).

shape grammars (Stiny & Gips, 1972). Like Hiller and Isaacson, Stiny and Gips were familiar not only with earlier experiments in computer art (Bense, 1971) but also (and perhaps more crucially) with earlier developments in mathematics and generative linguistics (Birkhoff, 1933; Chomsky, 2015).

One important difference between shape grammars and the experiments in musical composition considered above is that shapes are themselves the units of calculation. In other words, rather than computing with numerical representations of shapes, as both Hiller and Brooks had done with music, shape grammars compute directly with shapes via spatial relations and transformations (Knight, 1994). Hence, they redefine design as a visual form of computation and, in consequence, as an activity inseparable from perception and interpretation—that is to say, from human experience.

A canonical example of a shape grammar helps reveal their capacity to elicit design inductively from examples. In the late 1970s, Stiny and the late Australian architectural theorist

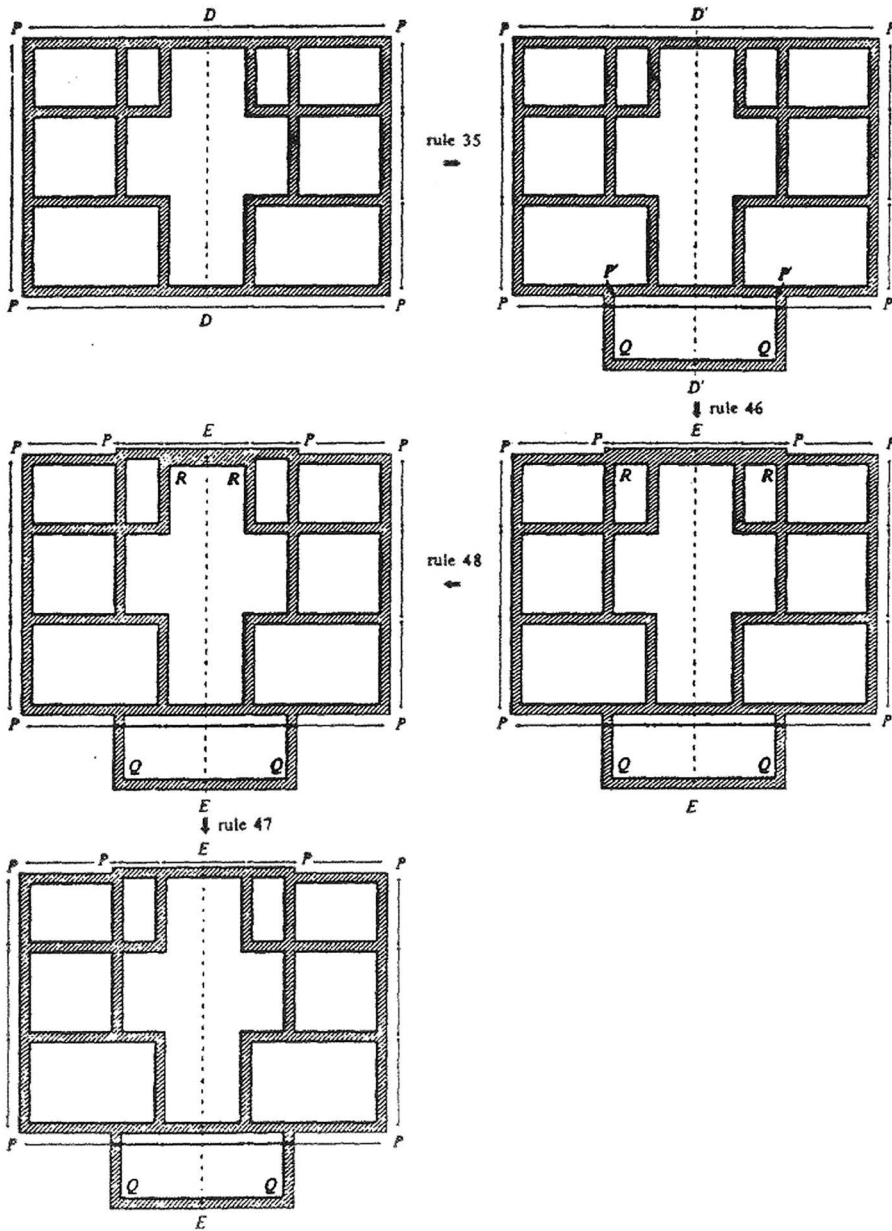


Figure 2.4 The Palladian grammar by George Stiny and William Mitchell is a generative description of all of Andrea Palladio's villas (Image credit: Stiny & Mitchell, 1978).

William J. Mitchell set out to codify the design principles of Italian Renaissance architect Andrea Palladio into a shape grammar. They noted that, while Palladio made many of his design principles explicit in his writings, many remained implicit. So, to recast Palladio “in a modern, generative form,” they examined both Palladio’s writings and his built work and created a set of rules (see, for example, Figure 2.4). Unlike Brooks’ experiments in musical composition, the rules were not the result of statistical analysis but from a careful and

judicious study by the two computationally inclined design and architecture scholars. In that sense, Stiny and Mitchell's Palladian grammar is closer to Hiller and Isaacson's experiment—in that they abstract compositional principles from a particular body of work. In the language of symbolic AI, Stiny and Mitchell were “knowledge engineers”—laboriously “extracting” expertise, to the extent that this is possible, from Palladio's documented works and encoding it in a formal system. Both buildings and words comprised what in present-day machine learning parlance we may refer to as the “training set” for their Palladian grammar.

However, Stiny and Mitchell's ambitions went beyond mere style transfer. An effective grammar, they wrote, should perform descriptive, analytic, and synthetic functions (Stiny & Mitchell, 1978). In other words, it should (a) characterize the traits shared by all Palladian buildings, (b) assess whether a given design, real or invented, fits within the constraints of the Palladian style, and (c) be able to generate entirely new Palladian designs (Ibid.). They emphasized the open-ended nature of the resulting grammar, and on the extent to which it codified the Palladian style, and not so much on its potential to be automated. For these reasons, the resulting grammar is a work of architectural history as much as one of computational design.

An important difference between Stiny and Gips' work and Hiller's experiments in music is that, as mentioned, shape grammar units were not numerical, and thus they were not conventionally computable. As visual elements, their pliability was difficult to reproduce digitally. As many a shape grammarian would attest, the structured nature of computational descriptions contrasts with the plasticity of visual perception. In this sense, shape grammars can be seen as a critique of numerocentric approaches to computational design and, in fact, as a provocative redefinition of *both* design and computation. Stiny and the late Gips spent the better part of their academic careers clarifying this point—decoupling computation from computers—and investigating a fascinating (and difficult) question: How to compute with shapes?

Despite these important differences, the experiments in musical composition by Hiller, Brooks and their collaborators, and the mathematically tractable visual computations envisioned by Stiny and Gips, share the same ambition: to define open-ended and yet stylistically consistent aesthetic systems. Their authors sought to ensure this consistency by imposing restrictions through statistical, probabilistic, or rule-based methods. Crucially, by selecting stylistic traits and curating a set of precedents, these authors set the boundaries of a space of possibilities: a *design space*.

While architects have been using rules for a long time—think of Vitruvius or Durand—open-ended design worlds such as those described through grammatical rules or probabilistic descriptions express a distinctive 20th-century imaginary of design linked to information theory and mathematical abstraction. They express a desire to codify design knowledge to make it computable and malleable in new ways. Crucially, they inscribe new conceptualizations of the design process and, in particular, new modes of authorial agency in design. These sensibilities remain at the core of present-day efforts to harness AI and machine learning methods in design. In what follows, we will examine transitions and overlaps between rule-based and data-intensive design systems, pondering what kind of design “knowledge” these systems encode—and what different kinds of “learning” they rely upon and facilitate.

From rule-making to data-wrangling

The origins of today's machine learning techniques are often traced to the convergence of AI research and statistics in the 1980s (Park, 2015). More recently, a series of technological

shifts such as the availability of increasingly large datasets and processing power, and the resurgence of neural networks, has infused new life into the promise of using statistical analyses to inform computational reasoning. Machine learning researchers, however, do not conform to a unitary voice, and different schools of thought, each with their guiding theories and preferred algorithmic tools, converge in a diverse (and rapidly evolving) field. What seems to best unite this community and their tools is the notion that computational methods might help automate something akin to inductive reasoning: to draw generalizations, to infer patterns, and to predict future outcomes based on prior experience encoded as data. But data are never neutral—they are cultural artifacts situated in social and material settings (Drucker, 2011; Boyd & Crawford, 2012; Paglen, 2016; Crawford & Paglen, 2019; Loukissas, 2019; D’Ignazio & Klein, 2020). This means that data themselves are not inherently truthful but *constructed*, and thus their production and the processes for collecting, curating, and visualizing them are acts of design.

A closer look at some machine learning techniques helps illustrate this. US psychologist Frank Rosenblatt’s formulated artificial neural networks in 1958. Based on a loose analogy of the human brain, neural networks use a combination of simple calculating units, each able to compute a very simple output from a very simple input. Trained on a sufficiently large number of examples, a neuron can, for example, react to particular details or features in pictures, such as edges, shadows, or colors. Convolutional neural networks (CNNs), one of the most commonly used techniques for visual recognition, are multilayered neural networks able to detect patterns in images at different scales and levels of complexity. CNNs were first demonstrated in the 1990s to be able to interpret handwriting in a computationally efficient way (LeCun et al., 1998). Their first major application was to interpret handwriting in checks (LeCun, 2014). Today, a common use of CNNs is the classification of images based on resemblance—for example, they are the backbone of Google’s image search feature. Groups of neurons can recognize (or represent) images with shared features or infer those features on any given image (Mordvinsteve, 2015). A neural network trained on, for example, the work of an expressionist painter can transfer this particular style into any image. However, unlike its stochastic and grammar-based predecessors, this type of work resembles a Photoshop filter more than a generative device.

Projects exploring the aesthetic potential of CNNs include experiments in style transfer (Gatys et al., 2015; Kogan, 2015): a search engine that finds places on satellite photographs based on visual likeness (Levin et al., 2016); and the at-times disturbing imagery produced by Deep Dream, a CNN developed at Google that can be directed to recursively infer and emphasize a variety of patterns in images—such as faces, colors, or edges (Mordvinsteve, 2015; Mordvinsteve et al., n.d.). Recent work on recurrent neural networks (RNNs) suggests a different type of production system. RNNs are a type of neural network that computes its own output. US computational artist and author Kyle McDonald shows that RNNs can be used to classify images by similarity (Figure 2.5) or to generate new text that resembles the style of a set of examples (McDonald, 2016). For example, an RNN trained on the written works of author Zadie Smith can produce text that maintains a stylistic resemblance with her works—despite being mostly nonsensical (Karpathy, 2015a, 2015b). Unlike CNNs, which are designed to work on image maps, RNNs are designed to process sequential inputs of arbitrary length, which makes them suitable for processing text.

Compared with the Cold-War era experiments in music and design we discussed at the beginning of the chapter, the training sets for these experiments are larger—in some cases millions of examples versus a few dozen—and their computational techniques more sophisticated—distributed, multiple “neural” processes vs. stochastic or rule-based ones.

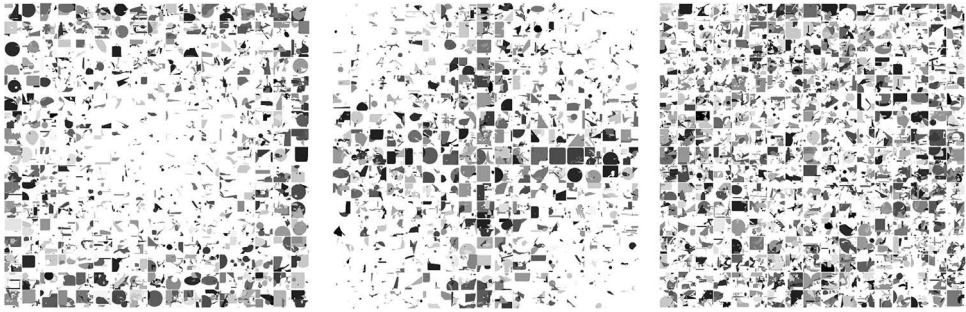


Figure 2.5 In *Innards*, US computational artist Kyle McDonald uses a recurrent neural network to generate emoji from vectorial graphics using a character-level recurrent framework. The result evokes Allison Parrish’s work “smiling face withface” (Image credit: McDonald, 2016).

And yet, despite a well-engrained cultural fascination with “big data,” not all approaches to machine learning in design involve large datasets. An alternative approach is exemplified by London-based US computer scientist and musician Rebecca Fiebrink, who develops tools that employ machine learning methods for artistic performance and creative design. Her free and open-source software Wekinator allows users to train machine learning algorithms to respond to different types of input such as facial gestures and bodily motions (Fiebrink et al., 2009). Wekinator users can input a series of gestures into the system via a webcam and associate these to, for example, sound responses. In this way, users craft a small dataset of gestures that the system can use to classify a much wider range of hand gestures during a performance. The result is a highly versatile gestural instrument. Fiebrink’s idea of “data as a user interface” nicely illustrates an aesthetic and philosophical commitment to the potential of open-ended and embodied interactions, rather than those of symbolic manipulation, in creative practice (Fiebrink, 2011, 2015). Here, the quality and craft of the dataset are more important than its size.

Both the user-centered work exemplified by Rebecca Fiebrink’s work just discussed and the data-focused explorations exemplified by the projects outlined before share with their Cold-War era predecessors an interest in the configuration of stylistically consistent aesthetic systems. Be it through rules, probabilistic descriptions, stochastic processes, neural networks, or grammars, they rely on the codification of aesthetic materials to analyze, describe, generate, or perform. Despite their different technical underpinnings, they also share a long-standing fascination with the potential of computational methods to work inductively: to derive aesthetic results from the analysis of examples.

From this brief overview of machine learning in design, I would like to propose two conceptual moves to understand these technologies’ potentials in the design domain. One: understanding the construction of the data, the design of the rules, and the curation of the examples, as *acts of design*. Two: to consider not just outcomes but how these systems *reconfigure relationships* between designers, authors, users, and audiences.

With these, we might explore directions forward that challenge naïf fantasies of design automation, and simplistic approaches to style transfer, and open up analytical and generative potentials for incorporating machine learning methods in design.

The following section discusses three research projects developed at the Computational Design Laboratory at Carnegie Mellon University that help illustrate the conceptual moves stated above and outline some emerging potentials and limitations of machine learning in the design domains.

Some new directions

Granular control over algorithmic actions

The first project, developed by computational designer and artist Erik Ulberg, explores new ways of interacting with convolutional neural networks that afford users fine-grain control over the algorithm's behavior. Inspired by the notion of "craft," Erik sought to create an algorithm that enabled the direct, iterative connection artists have with materials when producing work. He developed two computational tools, allowing users to edit and visualize, at different levels of abstraction, a neural network to make line drawings. Together, the tools "provided a crude form of perception for a generative line drawing system to allow it to dynamically respond to preexisting images or emergent structures" (Ulberg, 2020). The first tool, the "Kernel Tuner," allows users to control the weights of a neural network's first layer of convolution, where basic features are extracted from drawings. By adjusting the kernels, users could calibrate the tool to respond to a drawing's specific features—in this case, a line's end. Testing how the Kernel Tuner responded differently to different kinds of drawings, Erik observed that, rather than a general-purpose drawing system, the tool was highly specific to a certain kind of drawing. The process of calibrating the tool seemed to reveal aspects of the network's internal operation. Noting that often many kernels activated at any given pixel, he made the following observation about a network's way of dealing with visual ambiguity:

...the internal abstractions in a network are better thought of in terms of adjectives on continuous scales rather than one-hot vectors of nouns. Neural networks do not produce clean, easy-to-understand signals. A pixel is not just a "corner." It is "corner"-like, but also "vertical line"-like and "horizontal line"-like. Adjusting the parameters manually facilitates human-led jumps through the search space, but these results suggest we should incorporate computational fine tuning to overcome the complexity of balancing competing activations.

(Ulberg, 2020)

The second tool, the "Network Builder," works at a higher level of abstraction. Its visual interface allows users to engage with more complex visual concepts such as "house" or "bottle," correcting the weights to privilege certain activations. Working in conjunction with the Kernel Tuner, the Network Builder can produce intriguing compositions based on these visual concepts (Figure 2.6). The tools did not fully accomplish Erik's goal to facilitate an engagement with neural networks, resembling that of an artist with their material. However, they did offer a new type of algorithmic scaffolding for visual experimentation, and intriguing visual results. In addition, the visualization offered insight into the workings of the algorithm itself—for example, about how CNNs handle visual ambiguity. Erik's project foreshadows future developments into computational methods that experiment with the very architecture of machine learning algorithms, and with their user interfaces, for expressive purposes.

Architectural distant readings

The second project asks how machine learning methods might enable the analysis of large collections of architectural data. Researchers in the digital humanities have developed the concept of "distant reading" to describe analyses of large datasets enabled by computational

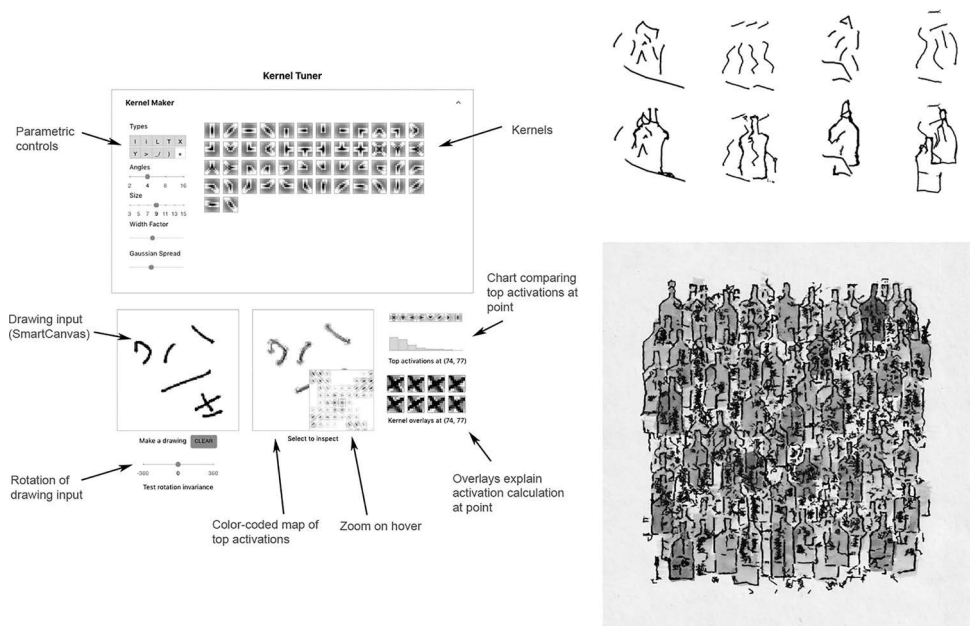


Figure 2.6 Tools to visualize and manually alter the behavior of a neural network in real time allows for a new kind of image manipulation (Image credit: Ulberg, 2020).

methods. These researchers contrast this approach with that of “close reading,” which typically focuses on a single literary, visual, or technical work. For example, instead of focusing on one work by a single author, such as a novel or a memoir, or even the entire works of a single author, researchers using this concept have explored entire genres or periods of literary production. Relying on computational techniques such as text mining and network analysis, distant reading practices seek to identify continuities or variations across vast collections that would be difficult to perceive with the conventional toolkit of the literary scholar (Underwood, 2017). With similar ideas in mind, architect and computational designer Cecilia Ferrando and collaborators conducted an experiment on “architectural distant reading” wherein architectural characteristics such as visual and spatial connectivity can be read across a large body of buildings, potentially identifying typological and configurational variations and similarities (Ferrando et al., 2019). To test this idea, the team laboriously composed a small dataset of religious buildings, which they represented as graphs—with nodes as individual spaces and links as indicators of spatial connectivity. They implemented a graph-based kernel to quantify the similarity between graphs based on the length of random walks and clustered the buildings in the dataset by the similarity of their spatial structures. This enabled them to find correlations between spatial structure and typological or functional traits. For example, the team observed that buildings cataloged as mosques in the dataset had a relatively horizontal, or shallow, spatial structure compared to monasteries. Their graphs resembled “flowers,” with a central space and a number of connected spaces forming on a level of spatial “depth” (Figure 2.7). In contrast, monasteries were more hierarchical, with spaces often chained into sequences. The clustering inspired the team to think of the concept of a “typological gradient,” which helps to think about building types not as exclusive categories but rather as overlapping ones.

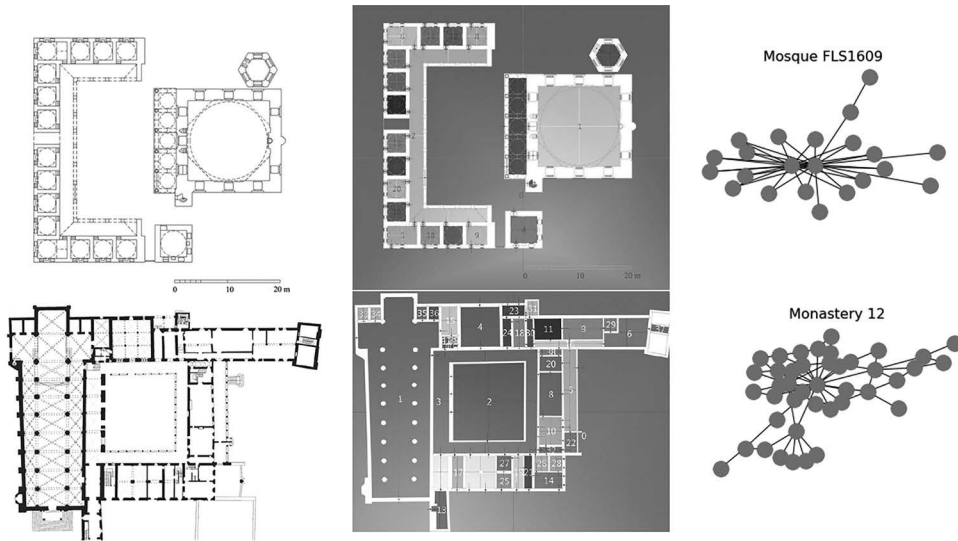


Figure 2.7 An experiment in “architectural distant reading.” A graph-based spatial analysis helps analyze “typological gradients” in collections of architectural data (Image credit: Ferrando et al., 2019).

Cecilia and team’s findings suggest that spatial structure in conjunction with machine learning methods can be used as an effective typological classifier—they were able to classify the buildings in the dataset with 93% accuracy. But, more importantly, they sketch out a way for machine learning methods to support architectural analysis and design—not as replacements of but as resources for architects and researchers.

New urban cartographies

The third project offered here for discussion is Jinmo Rhee’s study of Pittsburgh’s urban fabric using machine learning algorithms (Rhee et al., 2019). Despite being a relatively small city, Pittsburgh’s fabric is notably heterogeneous, with many neighborhoods, each with a distinctive urban character. Noting how research into computational aids to design has historically fallen short of acknowledging the importance of urban contexts, Jinmo developed a computational method to help understand the city’s rich patchwork of urban fabrics. The way architectural and urban data is conventionally represented digitally in urban data—without any link between urban context and architectural unit—makes this type of analysis difficult. To address this limitation, Rhee designed a data structure combining information about architectural units, such as their height, function, dimensions, occupancy, and their immediate urban contexts (Rhee, 2018). Working with GIS data provided by the city’s and county’s governments, Rhee generated a new dataset of images of each one of Pittsburgh’s 48,913 buildings, surrounded by its immediate urban context. Using t-distributed stochastic neighbor embedding, a dimensionality reduction technique, he was able to cluster buildings based not only on their individual characteristics but also on their contexts. Mapped back into space, this clustering produces a new type of morphological plan that redistricts the city based on subtle distinctions in its urban fabric (see Figure 2.8).

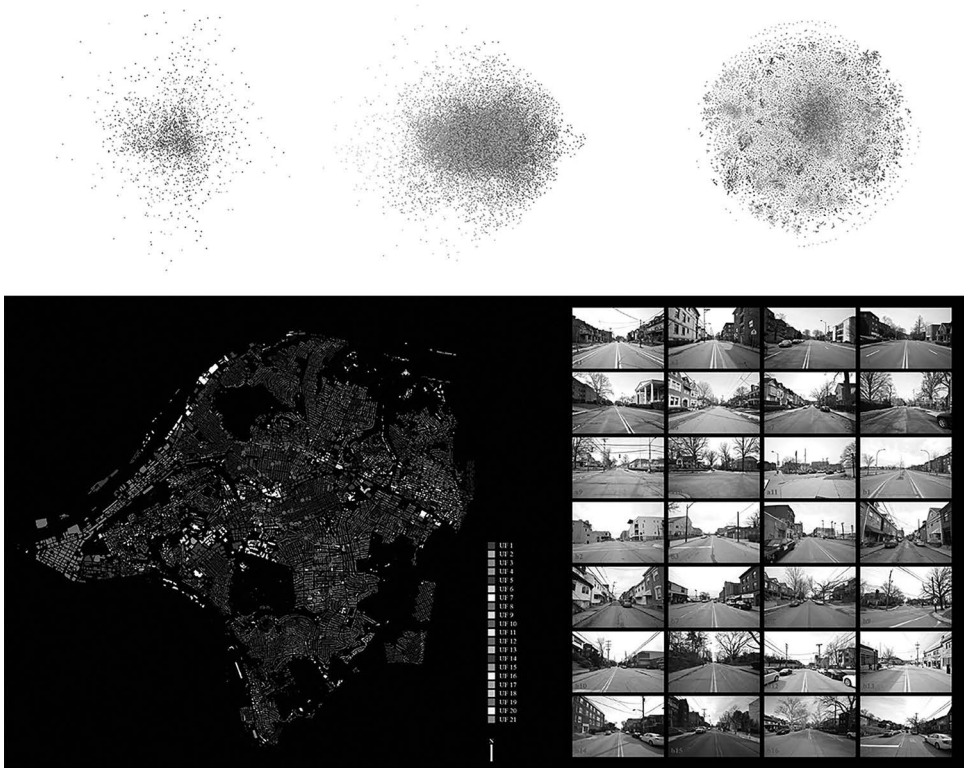


Figure 2.8 New data structures comprising building and urban data allow for fine-grain morphological analyses using machine learning techniques (Image credit: Rhee et al., 2019).

Jinmo supplemented his study with field visits to what appeared to be boundary conditions in the dataset, and with visits to the city archives, enriching his analysis with qualitative observations. These on the one hand helped confirm that the boundaries identified by his method corresponded with observable urban phenomena and on the other helped enrich his study with a deeper understanding of the historical origins of Pittsburgh’s morphological diversity. The study, which Jinmo is currently expanding as part of his PhD, opens up both analytical and generative opportunities. On the one hand, the enriched urban dataset affords the detailed study of a city’s morphology, which opens up nuances and suggests the possibility of comparative analyses of urban form. On the other, these techniques may be developed into a design toolkit allowing designers to, for example, assess architectural or urban design hypotheses in terms of their fit with existing, or desired, contexts.

Discussion

From rule-based music and synthetic Palladian villas to distant readings of architectural collections, handcrafted neural networks, and data-rich urban analyses, this chapter has delineated an experimental tradition in computational art and design based on inductive processes. Via rules, statistical controls, or machine learning algorithms, this tradition has focused on shaping spaces of design possibility. But I have also tried to show that the units of

these computational transactions (encoded melodies, shapes, “data”) have a materiality and a plasticity of their own, that they are irreducibly contingent upon human labor and creativity, and that, it is precisely there, on those materialities, labors, and creative acts, that the critical and aesthetic potentials of machine learning in design are realized. And yet, these labors are commonly hidden by rhetoric centered on the supposed autonomy of computational systems. By discussing a series of recent experiments, I have tried to show how the poetics of machine learning is also the poetics of data—how we as researchers, willingly or unwillingly, design and construct them. In other words, practices of data structuring, collection, and manipulation are acts of design at the root of the new aesthetics emerging around AI and machine learning methods.

Accordingly, the projects discussed suggest the emergence of new authorial roles related to how data is collected, curated, processed, and visualized. The explicit and implicit labors of designing and constructing a dataset are not only the defining component of the process. They also appear as the key arena for both critical and creative inquiry.

To finalize, drawing from the examples provided, I would like to offer for discussion three specific arenas where these practices open up opportunities for creative and critical engagements with AI and machine learning in design.

Codification and “translation” as creative practices

As recent studies of science and technology remind us, rather than objective and neutral, data are laboriously designed and constructed. They are enmeshed with and at the same time shape their social and material settings. It is thus important to recognize codification—the construction of these data—as a creative practice worthy of critical consideration. In encoding Palestrina’s compositional principles into computer subroutines, for example, Hiller and collaborators constructed a specific kind of dataset on which to train their stochastic system. In encoding 37 musical pieces as numerical strings, Brooks and collaborators not merely translated but also took a position about the nature of music as media, making it computable as a space of probabilistic inference. In encoding Palladian architecture in a finite set of visual rules, Stiny and Gips adopted (or reinforced) a stance about orthographic architectural representations’ capacities to codify what is important about architecture. These decisions should not be seen as innocent but as key *design* decisions crucially structuring design spaces. Of course, no translation is transparent nor leaves original meanings untouched. For example, in Cecilia Ferrando’s “architectural distant readings,” the decision to codify buildings as graphs shaped the kind of questions she and her coauthors were able to ask of the dataset—privileging notions of spatial structure and hierarchy (and overlooking other, arguably essential, factors of architectural quality). In Jinmo Rhee’s study of Pittsburgh, the analytical and generative potential of the machine learning framework is realized in the design of a new data structure specifically designed to codify aspects of a building’s urban context. As these examples show, critically engaging with codification practices can unlock opportunities for creative engagement in a design culture increasingly defined by data.

Disciplinary and methodological hybridity

Computational methods have always worked in conjunction with other knowledge domains and draw from a broad range of conceptual and methodological toolkits. Computational design practices are not an exception. In Hiller’s experiments in musical

composition, for example, molecular structures served as conceptual scaffoldings for me-
lododic ones. In Stiny and Gips' work, a centuries-old tradition of orthographic architec-
tural representations was used to produce a generative description of an entire body of
architectural work. This conceptual openness imposes critical demands on computational
design practices, because they are never safely contained within a disciplinary or method-
ological bucket. Failing to recognize these demands will likely result in yet another sim-
plistic attempt to style transfer, yet another naïf fantasy to automate design, or yet another
misguided effort to quantify architectural quality. The rigor with which the different do-
mains are explored and articulated will determine the quality of both the creative outputs
and critical insights offered.

There is no formula for creatively putting different forms of expertise in motion “in
the wild” nor for accessing methodological or disciplinary hinterlands effectively through
computational methods. However, the chapter offers some hints: Jinmo's attention to his-
torical archives and to evidence collected “on the field” productively puts architectural
and urban sensibilities in conversation with computational expertise. Erik's artistic sensi-
bility and practice not only motivated his inquiry but also shaped the two computational
tools he developed, as well as their outputs. Cecilia's architectural training and scholarly
understanding of spatial analysis methods, along with her deep understanding of machine
learning, shaped the choices she made about codification as well as the questions she asked
of the dataset.

Slowdowns and interruptions at the interface

The design of interfaces for users to interact with data in machine learning contexts of-
fers opportunities for creativity and innovation. Interfaces developed with particular users
or user groups in mind can help support new kinds of creative exploration. In Rebecca
Fiebrink's “Wekinator,” for example, we see an early expression of interfaces allowing
musicians, visual artists, and other creative practitioners to easily create and manipulate
training datasets. Using their bodies to iteratively sculpt the training data, users shape these
data and create frameworks of inference that suit their expressive goals—a computational
design space. Similarly, Erik Ulberg's “Kernel Tuner” and “Network Builder” engineer a
type of interaction that affords artists low-level, granular control over the algorithmic ma-
chinery of a convolutional neural network. Importantly, the interface actually *slows down*
a computational process that is typically in the background and creates opportunities for
interruptions and adjustments, which enable artistic experimentation. Of interest here is
that the focus is not placed on the design of new, faster algorithms, but on the design of an
interaction that facilitates new kinds of engagement with a computational process—making
it less rigid, and less predictable.

Acknowledgments

I am thankful to the graduate students in the Computational Design Laboratory at the
School of Architecture at Carnegie Mellon University with whom we discussed many of
the ideas in this paper and in particular to Jinmo Rhee, Erik Ulberg, Cecilia Ferrando,
and Ardavan Bidgoli. I am also thankful to Rebecca Fiebrink for insightful discussions
about her work.

Note

- * An earlier version of this article was published in German in: *Machine Learning: Medien, Infrastrukturen und Technologien der Künstlichen Intelligenz*, edited by Christoph Engemann and Andreas Sudmann. Bielefeld: Transcript, 2017.

References

- Bense, M. (1971). The projects of generative aesthetics. In *Cybernetics, Art and Ideas* (ed. Jasia Reihardt) (pp. 57–60). Studio Vista. http://www.computerkunst.org/Bense_Manifest.pdf
- Bewley, J. (2004). *Lejaren A. Hiller: Computer Music Pioneer*. <http://library.buffalo.edu/libraries/units/music/exhibits/hillerehibitssummary.pdf>
- Birkhoff, G. D. (1933). *Aesthetic Measure*. Harvard University Press.
- Boyd, D., & Crawford, K. (2012). Critical Questions for Big Data. *Information, Communication & Society*, 15(5), 662–679. <https://doi.org/10.1080/1369118X.2012.678878>
- Brooks, F. P., Hopkins, A. L., Neumann, P. G., & Wright, W. V. (1957). An Experiment in Musical Composition. *IRE Transactions on Electronic Computers, EC-6*(3), 175–182. <https://doi.org/10.1109/TEC.1957.5222016>
- Caplan, L. (2020). The Social Conscience of Generative Art. *Art in America, January 2020*, 108, 50–57.
- Cardoso Llach, D. (2015). *Builders of the Vision: Software and the Imagination of Design*. Routledge.
- Chomsky, N. (2015). *Syntactic Structures*. Martino Fine Books.
- Computational Design Laboratory. (2016). *Code Lab*. <http://code.arc.cmu.edu/>
- Crawford, K., & Paglen, T. (2019, September 19). *Excavating AI: The Politics of Images in Machine Learning Training Sets*. <https://www.excavating.ai>
- D'Ignazio, C., & Klein, L. F. (2020). *Data Feminism*. The MIT Press.
- Domingos, P. (2015). *The Master Algorithm: How the Quest for the Ultimate Learning Machine Will Remake Our World*. Basic Books.
- Drucker, J. (2011). Humanities Approaches to Graphical Display. *Digital Humanities Quarterly*, 5(1). <http://www.digitalhumanities.org/dhq/vol/5/1/000091/000091.html>
- Ferrando, C., Cardoso Llach, D., Dalmasso, N., & Mai, J. (2019). Architectural Distant Reading: Using Machine Learning to Analyze Architectural Traditions. *CAAD Futures 2019: Hello, Culture*. CAAD Futures 2019, Daejeon, Korea.
- Fiebrink, R. (2011). *Real-time Human Interaction with Supervised Learning Algorithms for Music Composition and Performance*. Princeton University.
- Fiebrink, R. (2015, July 9). Data as Design Tool. How Understanding Data as a User Interface Can Make End-User Design More Accessible, Efficient, Effective, and Embodied, While Challenging Machine Learning Conventions. *International Conference on Auditory Display (ICAD) 2015*. <http://iem.kug.ac.at/icad15/icad15.html>
- Fiebrink, R., Trueman, D., & Cook, P. R. (2009). *A Meta-Instrument for Interactive, On-the-Fly Machine Learning—Semantic Scholar*. NIME - New Interfaces for Musical Expression, Carnegie Mellon University. /paper/A-Meta-Instrument-for-Interactive-On-the-Fly-Fiebrink-Trueman/0652e6120d-72dc0162a17b5b0e49e9f5e12bc668
- Gatys, L. A., Ecker, A. S., & Bethge, M. (2015). A Neural Algorithm of Artistic Style. *ArXiv:1508.06576 [Cs, q-Bio]*. <http://arxiv.org/abs/1508.06576>
- Karpathy, A. (2015a). *Char-rnn*. GitHub. <https://github.com/karpathy/char-rnn>
- Karpathy, A. (2015b, May 21). *The Unreasonable Effectiveness of Recurrent Neural Networks*. <http://karpathy.github.io/2015/05/21/rnn-effectiveness/>
- Knight, T. W. (1994). *Transformations in Design: A Formal Approach to Stylistic Change and Innovation in the Visual Arts*. Cambridge University Press.
- Kogan, G. (2015). *Style Transfer*. Genekogan.com. <http://www.genekogan.com/works/style-transfer.html>
- Le Cun, Y. (2014, June 2). Convolutional Network Demo from 1993. https://www.youtube.com/watch?v=FwFduRA_L6Q
- LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86(11), 2278–2324. <https://doi.org/10.1109/5.726791>

- Levin, G., Newbury, D., & McDonald, K. (2016, May 24). *Terrapattern_alpha*. Terrapattern.Com. <http://www.terrapattern.com/>
- Loukissas, Y. A. (2019). *All Data Are Local: Thinking Critically in a Data-Driven Society*. The MIT Press.
- McDonald, K. (2016, October 7). *A Return to Machine Learning*. Medium. <https://medium.com/@kcimc/a-return-to-machine-learning-2de3728558eb>
- Mordvinsteve, A. (2015, July 13). *Inceptionism: Going Deeper into Neural Networks*. Research Blog. <https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html>
- Mordvinsteve, A., Tyka, M., & Olah, C. (n.d.). *Deep Dream*. GitHub. Retrieved January 5, 2017, from <https://github.com/google/deepdream>
- Paglen, T. (2016, December 8). *Invisible Images (Your Pictures Are Looking at You)*. The New Inquiry. <http://thenewinquiry.com/essays/invisible-images-your-pictures-are-looking-at-you/>
- Park, J. H. (2015). *Synthetic Tutor: Profiling Students and Mass-Customizing Learning Processes Dynamically in Design Scripting Education* [Thesis, Massachusetts Institute of Technology]. <http://dspace.mit.edu/handle/1721.1/101544>
- Rhee, J. (2018). *Diagrammatic Image Dataset Pittsburgh (DID-PGH)*. Jinmo Rhee. http://www.jinmorhee.net/jinmorhee_2-multiimage/didpgh.html
- Rhee, J., Cardoso Llach, D., & Krishnamurti, R. (2019). Context-rich Urban Analysis Using Machine Learning—A case study in Pittsburgh, PA. *Architecture in the Age of the 4th Industrial Revolution - Proceedings of the 37th ECAADe and 23rd SIGraDi Conference*, 3, 343–352. http://papers.cumincad.org/data/works/att/ecaadesigradi2019_550.pdf
- Stiny, G., and Gips, J. (1972). “Shape Grammars and the Generative Specification of Painting and Sculpture.” In *IFIP Congress, Information Processing* 71, 1460–65.
- Stiny, G., & Mitchell, W. J. (1978). The Palladian Grammar. *Environment and Planning B*, 5, 5–18.
- Ulberg, E. (2020). *Crafting the Weights of a Convolutional Neural Network to Make a Drawing* [MS]. Carnegie Mellon University.
- Underwood, T. (2017). A Genealogy of Distant Reading. *Digital Humanities Quarterly*, 11(2). <http://www.digitalhumanities.org/dhq/vol/11/2/000317/000317.html>
- Zaripov, R. Kh., & Russell, J. G. K. (1969). Cybernetics and Music. *Perspectives of New Music*, 7(2), 115–154. <https://doi.org/10.2307/832298>