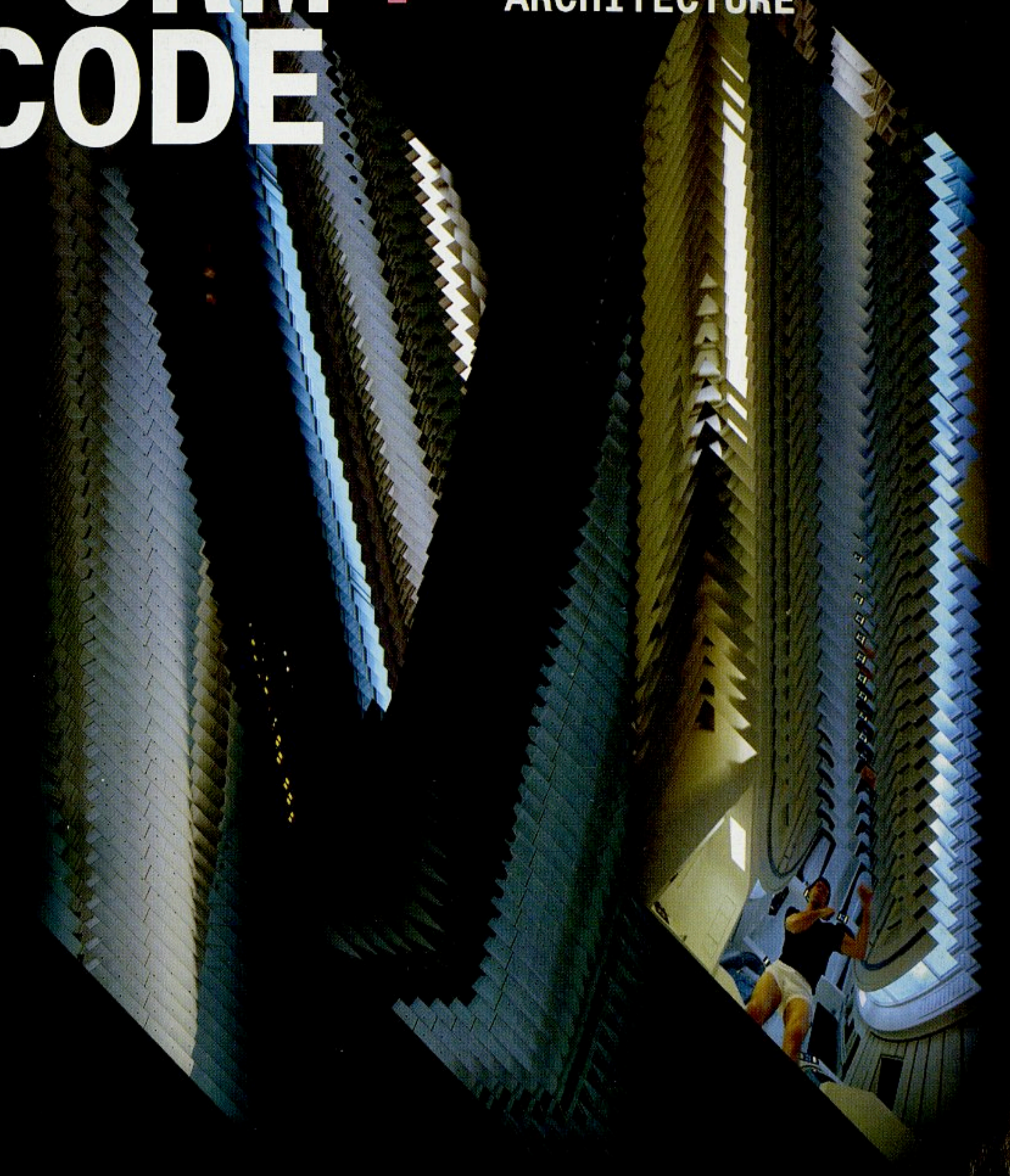
Also in the Design Briefs series from Princeton Architectural Press:

D.I.Y. Design It Yourself, Ellen Lupton, ed.
Elements of Design, by Gail Greet Hannah
Geometry of Design, by Kimberly Elam
Graphic Design Theory, by Helen Armstrong
Grid Systems, by Kimberly Elam
Indie Publishing, by Ellen Lupton
Lettering and Type, by Bruce Willen, Nolen Strals, and Ellen Lupton
Thinking with Type, by Ellen Lupton
Typographic Systems, by Kimberly Elam
Visual Grammar, by Christian Leborg
The Wayfinding Handbook, by David Gibson

FORM+ CODE

IN DESIGN, ART, AND ARCHITECTURE



Casey Reas, Chandler McWilliams, LUST

Princeton Architectural Press / New York

Computers and form have shared history nearly as old as the computer itself

ern personal computer in the late 1970s, “computing machines” were used by designers in the aerospace and automotive industries to perform complex calculations, and by scientists to develop intricate simulations of the physical world. The advantages initially offered by computers came in the form of efficiency and precision. Exploring new possibilities was not a priority; more importantly, they were used to perform calculations in a fraction of the time. The efficiency offered by the computer extended to the production of technical blueprints and allowed complex geometric drawings to be created far more quickly than with conventional techniques. Today, computers are still used as accurate drafting machines, but new ways of using them have opened new territories.

These letters convey the way computers interpret and display typography as a series of points, lines, and curves. For example, Adobe’s PostScript language renders typography with three commands: moveTo, lineTo, and curveTo. To create this typeface, the moveTo command was removed, rendering the letters as a continuous line.

Sketchpad,
Ivan Sutherland, 1963
With Sketchpad, users drew directly on the screen using a light pen. The behavior of the pen was controlled by a group of switches, buttons, and knobs. This image shows Sutherland using his software on a TX-2 computer.

DRAWING WITH COMPUTERS

In 1963, Ivan Sutherland pioneered the graphical user interface (GUI) with his Sketchpad; this initiated a paradigm shift in how people interacted with computers. Sketchpad's interface consisted of a set of switches and dials, a display, and a light pen—a device used to draw directly on the screen. By pressing switches on a control panel while drawing, the user was able to instruct the computer to interpret the movement of the pen in different ways. Each time the pen touched the screen, a new line was added between the last point and the new one. In this way, the user could draw simple polygons. Another switch was for drawing circles, and another for arcs, etc.; this allowed for fairly sophisticated drawing. Sketchpad gave designers a way to directly manipulate objects on screen without having to first write a numerical, code-based representation of those objects. After the objects were made, they could be duplicated, moved, scaled, and rotated to create new compositions.

Sketchpad was much more than a crude analog of paper and pen; it was a fundamentally new way to design. When drawing in Sketchpad, the designer could make use of constraints in order to form new relationships between elements and to force them to behave in specific ways, for example: snapping the end points of line segments to other end points or lines, keeping lines parallel, or forcing them to have the same length. The user could also create more sophisticated constraints, for example: a constraint could be designed to simulate the load-bearing properties of a bridge.

With the first computer-aided design (CAD) systems, Sutherland's innovations left the lab and entered industry. The software used within the fields of engineering and architecture lacked many of his innovations and served as little more than an analog for pen and paper. They allowed designers to draw using mathematical lines and curves rather than T-squares, drawing boards, and pencils. These "high-powered drafting machines" were hailed for their efficiency, speed, and productivity.[1] Drawings that would have taken days could now be done in hours.

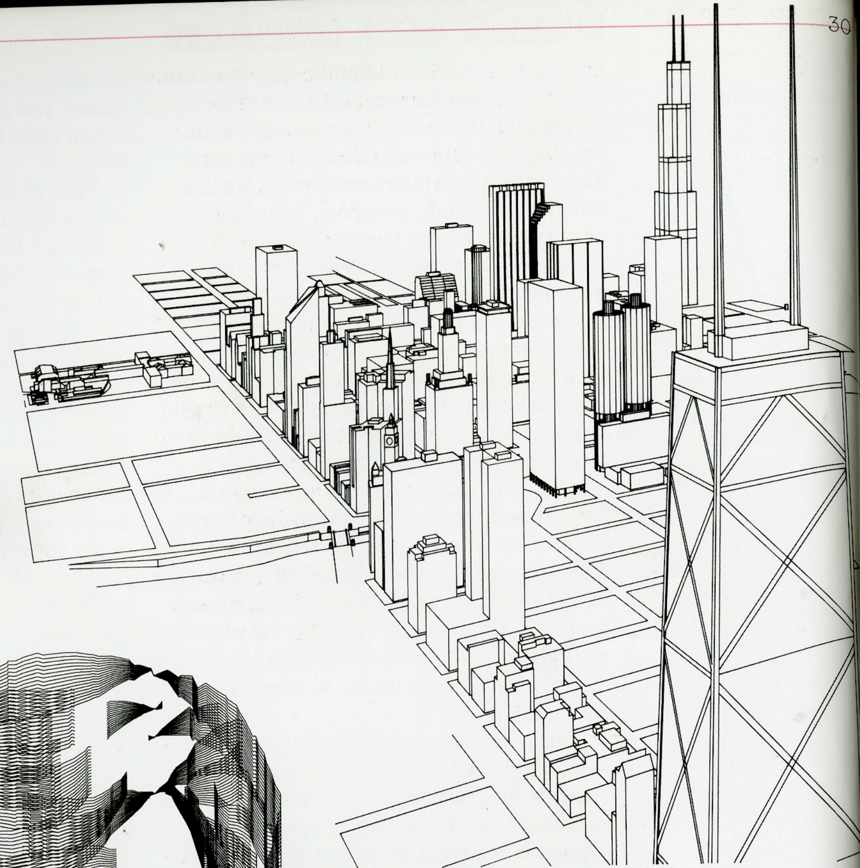
Even in this capacity, drawings made on the computer were considered a poor substitute for hand-drawn sketches and diagrams. Some people felt that the drawings produced by CAD machines were cold and overly technical, preferring the "slightly wobbly line work and imprecise endings of hand-drawn lines."[2] There were other obstacles to integrating CAD systems into industry. Some felt that they presented a new temptation—to never stop editing a drawing or set of plans; others believed that there were too many assumptions in the software that restricted the design possibilities.[3] As a result of these and other problems, computers were considered insufficient for the conceptual stage of design and were often used only at the end of the creative process. The advantages focused primarily on saving the designer's time and increasing productivity.

Within the design industry, however, the field that has been most profoundly transformed by the use of computers is graphic design. The proliferation of the personal computer—and, later on, the laser printer—laid the foundation for desktop publishing. Apple's LaserWriter could reproduce typography and images at much higher resolutions than previous home and small-business printing technologies. Perhaps more importantly, the LaserWriter included PostScript, which made it possible to use a wide array of fonts in the design, because they were now treated as software as opposed to physical metal type or transferrable lettering. This opened the door for designers to create and distribute their own typefaces and to have more control over the final typesetting. These technologies enabled vibrant activity and widespread innovation within the field of visual design in the 1980s and 1990s, ranging from the fonts of

1 Jasia Reichardt, Cybernetic Serendipity: The Computer and the Arts (New York: Praeger, 1969), 67.

2 William John Mitchell and Malcolm McCullough, Digital Design Media: A Handbook for Architects and Design Professionals (New York: John Wiley & Sons Inc., 1991), 129.

3 Christopher Woodward and Jaki Howes, Computing in Architectural Practice (London: Spon Press, 1998), 92.



Sine Curve Man, by Charles A. Csuri, 1967
The geometry for this plotter drawing was created by distorting an image of a face using the values of a sine wave. Like much of Csuri's work, code was manipulated to create an abstraction of the human form.

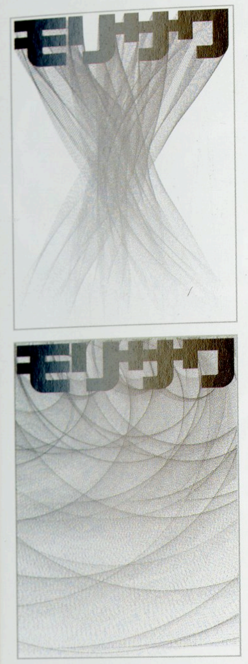
Chicago, by Skidmore, Owings & Merrill (SOM), 1980
With the rise in computing power and the increasing sophistication of CAD software, it became possible to model entire cities. SOM created a wireframe model of downtown Chicago to give viewers a feel for the city's form and massing of buildings.

Emigre and FUSE, to the radical work of April Greiman, David Carson, and many others.

With PostScript solidifying its place as the de facto standard, Adobe introduced Illustrator as its new visual development tool. With Illustrator, anyone could draw and lay out text and graphics without having to know the intricacies of the PostScript language. Eventually Illustrator, along with Adobe's Photoshop and InDesign applications, became nearly ubiquitous among graphic designers. Interestingly, all three of these applications have introduced scripting languages in recent years that allow users to extend the tools by writing code.

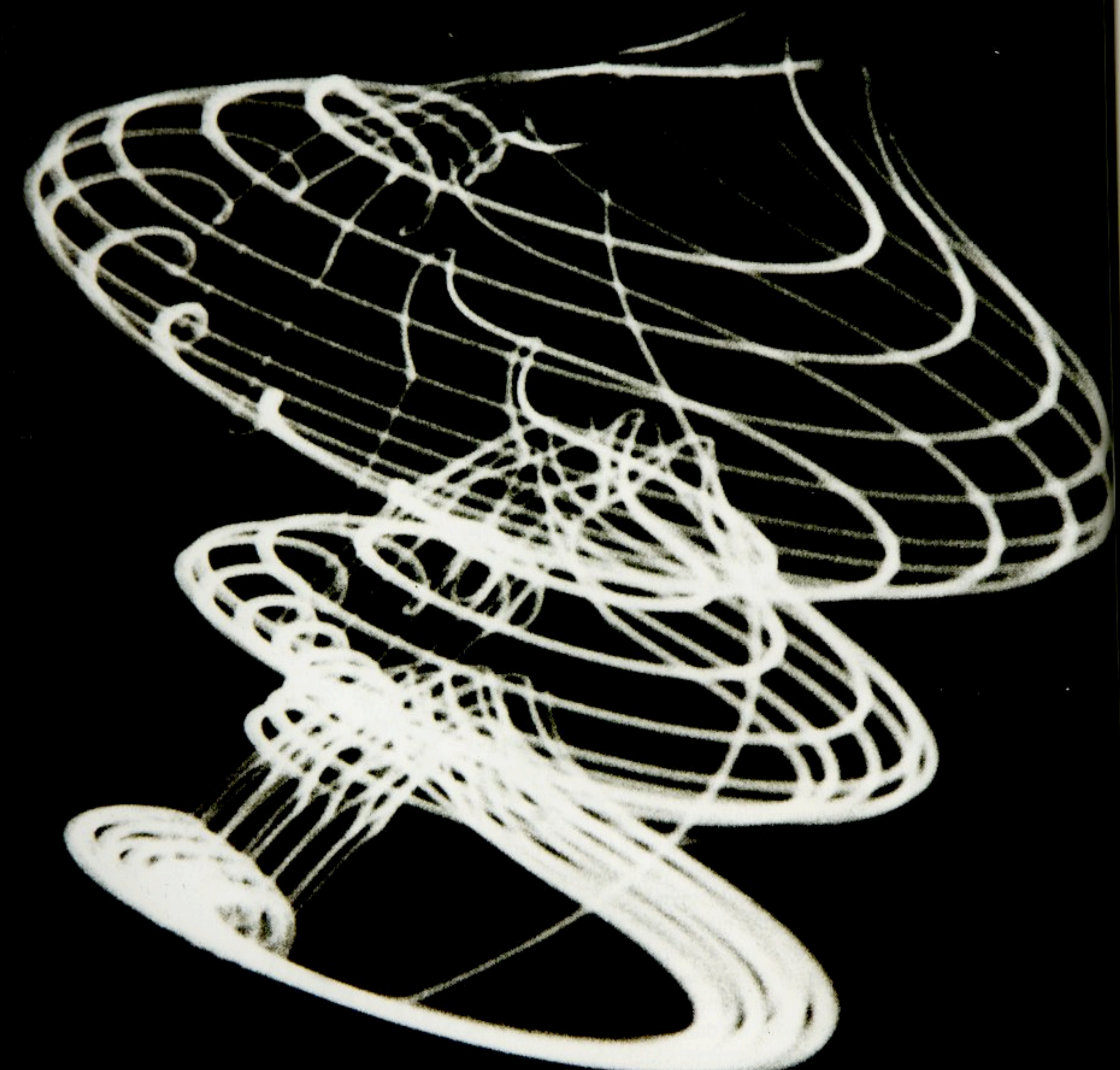
Following the birth of the Internet and other networking technologies, the computer increasingly became a tool for collaboration. Global computer networks called into question the need for centralized offices, in favor of an organization consisting of individuals spread around the globe. This has had a massive impact on the open-source software movement, where large and sophisticated applications are often built by a loose collection of individuals united by a shared interest. These new ways of working also had an impact on the way forms were created. In a distributed environment, different individuals work simultaneously on different parts of the same piece, seeing the whole only after the parts are stitched back together.

Having begun to represent objects and, with the aid of Sketchpad, relationships and behaviors, the real potential for the role of computers in design started to assert itself. If computers could be used to model what we know, then perhaps we could also use them to simulate what we don't know. French architect and philosopher Bernard Cache summed up the history of CAD systems by saying they "have certainly increased the productivity of the idea, but fundamentally they offer no advances over the work done by hand. Now, we can envisage second-generation systems in which objects are no longer designed but calculated."[4]



[4] Bernard Cache, Earth Moves: The Furnishing of Territories (Writing Architecture) (Cambridge, MA: MIT Press, 1995), 88.

Morisawa Posters, by John Maeda, 1996
Rather than rely on existing software tools, Maeda wrote his own code to manipulate typographic form. This allowed him to explore a unique graphic language for the ten posters he created for Morisawa. Each poster performs an algorithm to transform the logo of the company.

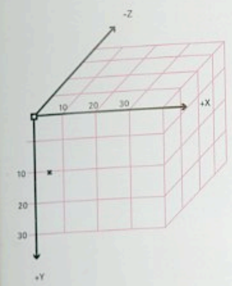


Electronic Abstraction 6, by Ben F. Laposky, 1952
Laposky used an oscilloscope—a technical device for viewing changes in voltage—to produce his abstract images. Starting in the early 1950s, he refined a process of modulating electronic waveforms to produce stunning and diverse images.

CONTROLLING FORM

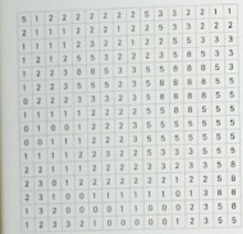
Understanding the ways that code is used in the production and creation of form requires a general knowledge of how form is manipulated by the computer. Outside the computer, form itself is physical and intuitive—it is the curve of a line on a page, the texture of paint, or the slope of a hillside. To manipulate form in the world, we don't need to understand the mathematics behind how things are put together, and we can specify where things are in relative terms, like “over there” or “next to me.” If a piece of clay is close enough to touch, then it can be directly molded and shaped. In contrast, computers rely on the ability to specify everything in numerical terms.

COORDINATES



The computer needs to know the position of every mark it draws, either on the screen or with a printer. To do this, we typically use Cartesian coordinates. If you imagine laying a large piece of graph paper over the screen, an x-axis runs from left to right, and a y-axis goes from top to bottom. These axes allow us to specify a precise position on the grid using a pair of numbers, normally the x-value followed by the y-value. For example, a point at (5, 10) is 5 lines from the left edge of the screen, and 10 lines down from the top.

SHAPE



Placing a piece of graph paper on the screen is more than just a metaphor. The screen is, in fact, composed of a grid of points called pixels. One way to draw a form on-screen is to lay the grid of pixels over an image of the form and measure the color value at each pixel in the grid. This method of representing an image makes what is called a raster image.

A raster image, which is sometimes referred to as a bitmap, is a complete description of what is shown on-screen at a given resolution. Resolution refers to how many points make up an image for a given physical size. If an image has a resolution of 800 × 600 pixels, there is a total of 480,000 pixels in the image, therefore requiring 480,000 numbers, with each one representing the color of one pixel. Resolution can be thought of as an image composed of tiles. There are two ways to make a tiled image look better: make the tiles smaller or move farther away from the image. On a computer, the two are related. Since the screen has a set size, lowering the resolution is like increasing the size of the tiles; alternately, you can keep the screen resolution constant and make the image smaller on-screen.

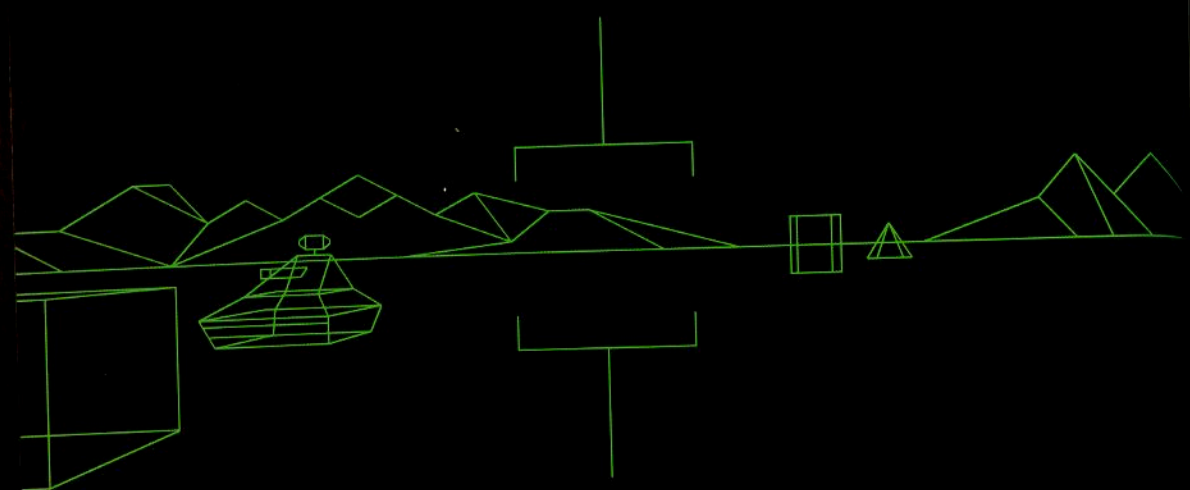
As explained earlier, an image's form, color, and shape must be converted to numbers in order for it to be useable on a computer. As a result, these qualities often lose the continuity that we have become accustomed to in the world. Every image on the computer has a resolution, or a width-by-height measurement, in pixels, but how many pixels are enough? The ever-increasing number of megapixels available in digital cameras and the popularity of high-definition television (HDTV) suggest that there are never enough.

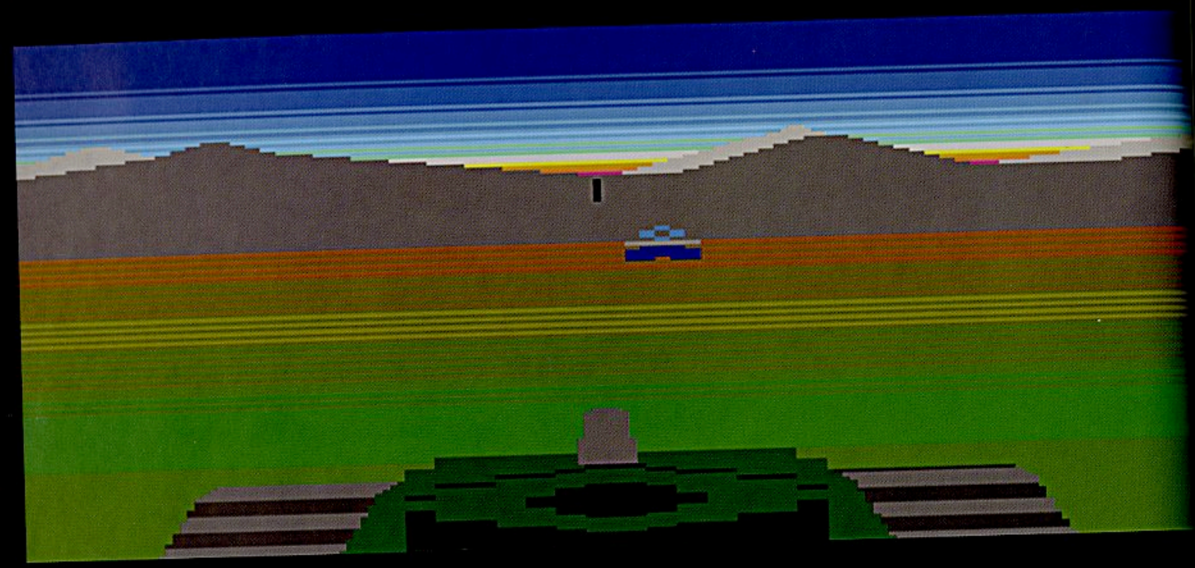
A megapixel is one million pixels. It refers to the total number of pixels in an image. In other words, the width of the image in pixels is multiplied by the height of the image in pixels so that an image measuring 2,048 by 1,536 is said to have 3.1 megapixels (2,048 × 1,536 = 3,145,728). Our eyes see a continuous analog stream of colors. The best we can do to represent that digitally is to increase the resolution to fool the eye into thinking that the image is continuous. But the fact remains that this is just a simulation, and it requires a lot of processing and memory to store enough information for the illusion to hold.

Raster graphics are an ideal way to store and manipulate photographic imagery, but they suffer from the confines of the resolution at which they are created. If we scale a bitmap image up to make it larger, the blocks of color must also be enlarged. This makes raster graphics a less than ideal way

Coordinates
Most computer graphics use a square grid with a horizontal x-axis and a vertical y-axis. An additional x-axis is used to draw 3-D forms.

Raster
A raster is a grid of pixels. The color of each pixel is controlled to create an image.





BattleZone, Atari, 1980 and 1983
The graphics in the arcade version of BattleZone were drawn with vector lines on an oscilloscope. The Atari 2600 home version of BattleZone used raster graphics, because it was displayed on a television set. It was more colorful, but the graphics were lower in resolution.

to store drafting or drawing information that often needs to be moved, scale, rotated, and reworked. For this, there are vector graphics.

Vectors graphics use the same Cartesian grid as bitmaps, but instead of storing the value for every pixel in the image, they store a list of equations that define the image. This is ideal for drafting and precision drawing, where any shape available to geometry—lines, circles, rectangles, and curves—can be combined to create a composition. Because the forms are described using geometric equations, they can be scaled and transformed easily and without losing detail. The scalable nature of vector graphics makes them an essential element in the production of printed matter. A printer may have a resolution many times greater than that of a monitor, and without vector graphics it would be very difficult to create smooth lines and crisp type. Furthermore, fabrication technologies, such as laser cutting and computer numerical controlled (CNC) milling rely on the detail and precision offered by vectors.

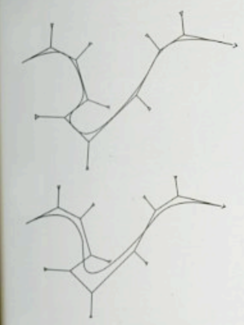
Objects in three-dimensional modeling software, such as Rhinoceros and Autodesk Maya, are commonly represented using vectors. In addition to the two-dimensional curves, points, and lines we are familiar with, these applications allow designers to create a number of different objects, such as meshes, NURBS (Non-Uniform Rational B-Splines), and subdivision surfaces.

COLOR

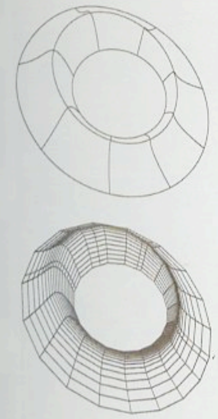
Unlike paint, color on-screen is additive, meaning that the more colors you add together, the closer you get to white. Additive color systems use the primary colors red, green, and blue to create the colors we see on-screen. The common 24-bit color depth allows each base color to be assigned a value from 0 to 255, giving a total of 16,777,216 possible colors—that's more than can be distinguished by the naked eye. For example, pure yellow has a red value of 255, a green value of 255, and a blue value of 0. Light brown has a red value of 140, a green value of 98, and a blue value of 0. Changing this blue value to 255 produces an electric purple.

REALISM

Like the history of European painting until the end of the nineteenth century, the history of computer graphics has prioritized a realistic depiction of the natural world. The bridge between the crude wireframe engineering models produced in the 1960s and the naturalistic form, lighting, and textures of today's rendering tools has spanned over thirty years of focused research. (This transition can be traced, in part, using the dates of the work included in this book.) One of the first effects mastered was the illusion of a third dimension rendered on a flat screen. After that came the hidden-surface algorithm for hiding the lines at the back of a model and making it appear solid rather than composed of wire. Similar to how shading in a pencil drawing helps produce depth and continuity, shading algorithms were developed to create the appearance of smooth surfaces from the hard edges of flat polygon models. Over time, new and better techniques were developed to accurately depict textures and, more importantly, light reflecting off surfaces. Beyond these algorithms, a mathematical model of a camera is at the core of most rendered software images. The parameters of these models imitate those of real lenses, such as focal length, field of view, and aperture. When the image is rendered, the calculated lens optics determines how near or distant the objects appear and distorts the geometry to create perspective. The development of ever-more-realistic rendering techniques continues, but in recent years there's been a renewed interest in non-photorealistic rendering. These techniques make geometric models look as if they were painted or built from clay.

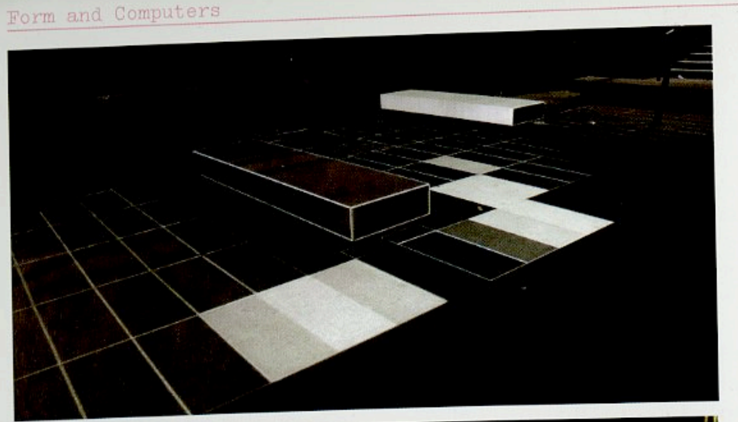


Splines
A spline is a type of curve, with a shape defined by the position of control vertices. Splines have a dimensionality that affects how close the curve fits to each vertex.

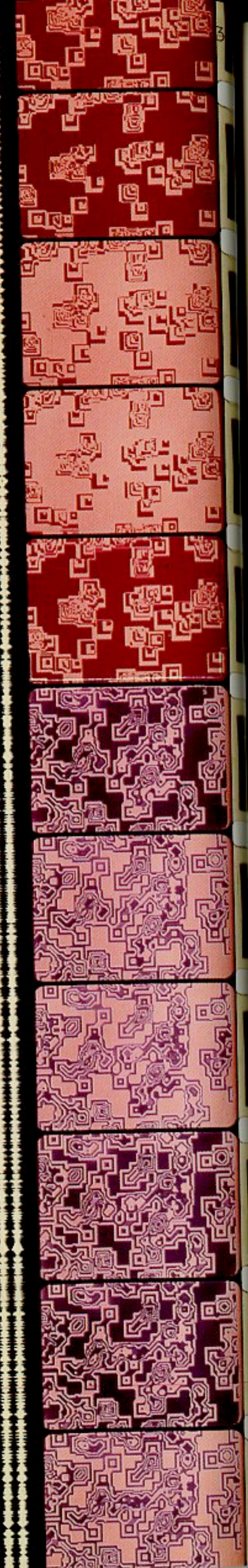


Advanced Geometry
Surfaces can be mathematically defined in a number of different ways. A triangle mesh is a set of connected triangles; NURBS are smooth surfaces created with splines; and subdivision surfaces use recursion to make a fine mesh to represent curvature.

Toward realism
The history of algorithms in computer graphics follows a path toward realism, from coarse outlines t
smooth, solid surfaces



Entramado,
by Pablo Valbuena, 2008
Valbuena combines virtual 3-D models with precisely placed projectors to augment physical space with a closely choreographed sequence of light, which appears to follow and modify the space itself.



Pixillation,
by Lillian Schwartz, 1970
Schwartz worked with Ken Knowlton at Bell Labs to produce the computer-generated sequences of this abstract film.

PRODUCING FORM

An important aspect of the relationship between form and code is how the abstract, immaterial, and imperceptible world of code comes into contact with our senses. Understanding how color is represented is a part of this relationship, but there are other processes by which the numerical representation of form can be transformed into something that we can perceive, such as light, pigment, or material structure.

LIGHT

Long before the ubiquity of full-color displays, the oscilloscope served as the primary device for real-time visual output from the computer. Despite its low-quality monochrome image, systems like Sketchpad and early video games made excellent use of this device.

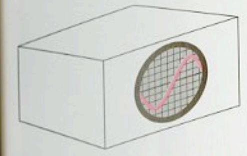
The full-color cathode ray tube (CRT) in the form of the television was targeted as the primary display device for early home video game systems, such as ColecoVision and the Atari 2600. The CRT consists of an electron gun and fluorescent screen enclosed in a vacuum tube. The gun fires electrons at the screen in a left-to-right, top-to-bottom pattern. When the electrons strike the screen, the fluorescent material glows. As a result of this process, the images on CRT screens have a very distinctive appearance.

The invention of the framebuffer was crucial to the widespread use of the full-color CRT, and to computer graphics as a whole, opening the door for digital painting programs, photo manipulation, and texturing. First developed at Xerox Palo Alto Research Center (now called PARC Inc.) in 1972, the framebuffer stored the entire contents of the screen in memory. Prior to this, only vector graphics could be drawn on-screen, because it was impossible to manage the amount of memory necessary to work with rasterized images.

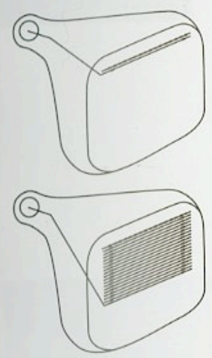
Increasingly, the most common computer displays in use today are liquid crystal displays (LCD). LCDs have numerous advantages over the CRT. They use less power and are smaller, which makes them ideal for mobile computing. They can also update their image faster to provide a more vivid experience. Because LCDs can be made in a range of sizes, from the handheld to a large television, they can be used to create both intimate and public experiences. In addition, they can be modified to make touch screens and to provide physical feedback.

Modern digital projectors allow content to be seen by a large group of people at once. Beyond this basic use, projectors offer a way to immerse the viewer in imagery, augment a physical space, or create nonstandard display shapes such as circles. The front-projection setup, where the image is projected onto the front side of a screen, is the most common. A rear-projection setup, with the image projected onto the back of a semitransparent screen, is a good way to allow viewers to approach the image without worrying about casting shadows or otherwise interfering with the image.

Appearing in everything from key chains to coffee makers to animated billboards; light-emitting diodes (LEDs) are a staple of contemporary everyday life. An LED is an electronic component that creates light when a current is applied to it. Compared to traditional means of generating light, LEDs are far more energy efficient and last longer. In the context of form making, they are interesting for their highly variable appearance and small size. It is possible to create displays of nearly any size or shape by piecing a large number of LEDs together. In this way, each LED can act as a pixel in a raster display. These custom displays are then controlled using hardware and software that make them behave like traditional screens.



Oscilloscope
Oscilloscopes use voltages to control the movement of an electron beam. The movement from left to right is sometimes fixed to a clock, while the movement up and down is controlled by an electrical signal. This setup makes it easy to visualize regular signals like sine waves.



Cathode ray tube (CRT)
Electrons are fired through a vacuum tube at a phosphorescent screen, causing it to glow on impact. In raster displays, the beam moves from left to right, top to bottom.

Volume, by United Visual Artists, 2006
This light and sound sculpture is composed of a series of custom-designed LED columns that respond to the motion of viewers.

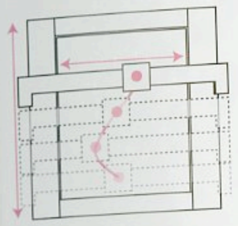
Hylozoic Grove, by Philip Beesley, 2008
Beesley used a laser cutter to create two different types of form for this sculpture. The structure and mechanisms were cut from rigid plastics; while light, flexible plastics were used to create the delicate featherlike elements.

PRINTING

In the early days of computer graphics, images were printed on paper using a plotter in order to make the details, which appeared vague on the extremely limited displays, appear clear. A plotter is a machine that moves a pen over a drawing surface. The pen is given commands to control the direction and speed of movement, making it possible to vary the quality of the lines. By changing the material of the drawing surface or swapping the pen for a pencil, brush, or other drawing instrument, many interesting results have been created.

In the mid-1980s, the first laser printers designed for home use began to appear. Laser printers use a combination of electric charge and focused light to fuse toner to paper. This technique allows them to print 300 dots per inch (dpi), which is considerably higher than the 72 dpi available with the common dot matrix printer.

Though laser printers excel at printing on paper, the invention of the inkjet printer expanded the range of possible mediums and inks available. The basic spray-nozzle design of the inkjet is so flexible that it is now possible to print on diverse types of paper, plastic, and fabric. Even entire circuit boards can be "printed" using conductive ink.



FABRICATION

Fabrication is a catchall term used to describe a host of new technologies that are capable of producing physical objects out of digital representations. In a far more drastic way than printers and screens, various fabrication techniques are used for vastly different purposes and require new ways of thinking about code, space, and structure. The most common and straightforward fabrication tool is the laser cutter, which is mechanically similar to a plotter, except that a laser, rather than a pen, is positioned on an arm that can move in two dimensions. The computer moves the laser along the x- and y-axis of the bed to cut the material. Often, laser cutters have restrictions on the size, thickness, and type of material that can be cut. In addition to movement in two directions, the power of the laser cutter can be adjusted to etch metal and create intricate burn patterns on wood. Though laser cutters are limited to working in two dimensions, many architects, designers, and sculptors have found inventive ways to cut sections (similar to topographic maps) that are then reassembled to create intricate 3-D objects.

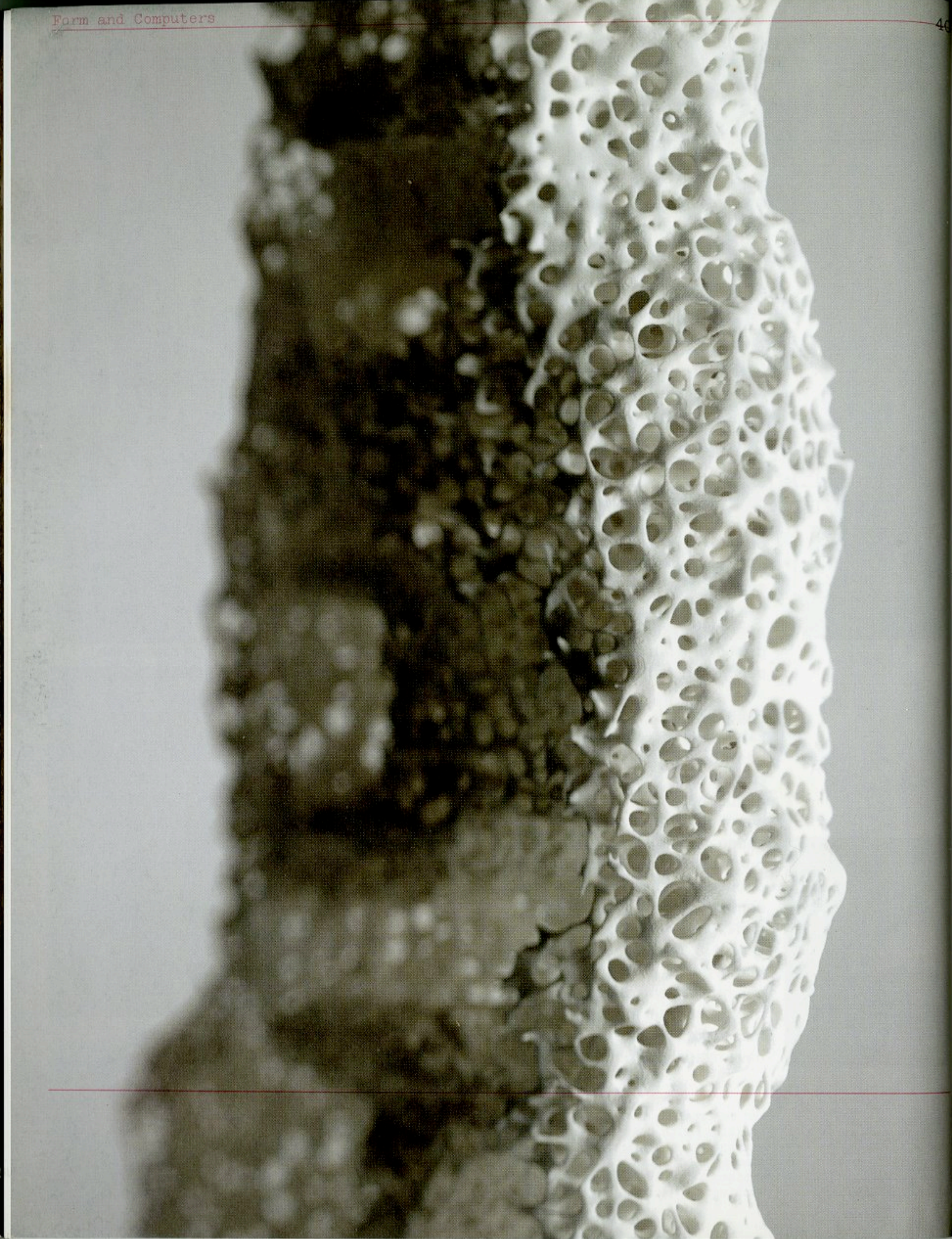
CNC milling, Selective Laser Sintering (SLS), stereolithography, and 3-D printing are just a few of the ways to create fully three-dimensional objects; that is, objects whose representations on the computer screen include information for x, y, and z axes, which are used to control the output device. A CNC-milling machine is similar to a plotter or laser cutter, but with the added flexibility of a continuous up-and-down motion. For example, a router bit is moved over a block of material, and as the bit moves, it cuts away a small amount of material, leaving behind a sculpted surface. In a three-axis machine, the router bit can only move directly up-and-down, making it difficult to sculpt objects from all sides. Some machines mount the block of material on a lathe, which rotates the surface facing the bit in order to provide additional flexibility.

CNC milling is a subtractive process; that is, material is cut away from a larger block in order to create the object. In contrast, SLS, 3-D printing, and stereolithography are additive processes that build up the final object by adding or fusing material together. Additive techniques have the distinct advantage of being able to create hollow spaces, undercuts, and overhangs, which are difficult to do using a three-axis CNC machine.

In a 3-D printer, a model is created by layering and fusing successive cross sections of material. Layers of powdered material, such as plaster, resin, or even cornstarch or sugars, are deposited and then selectively fused together by

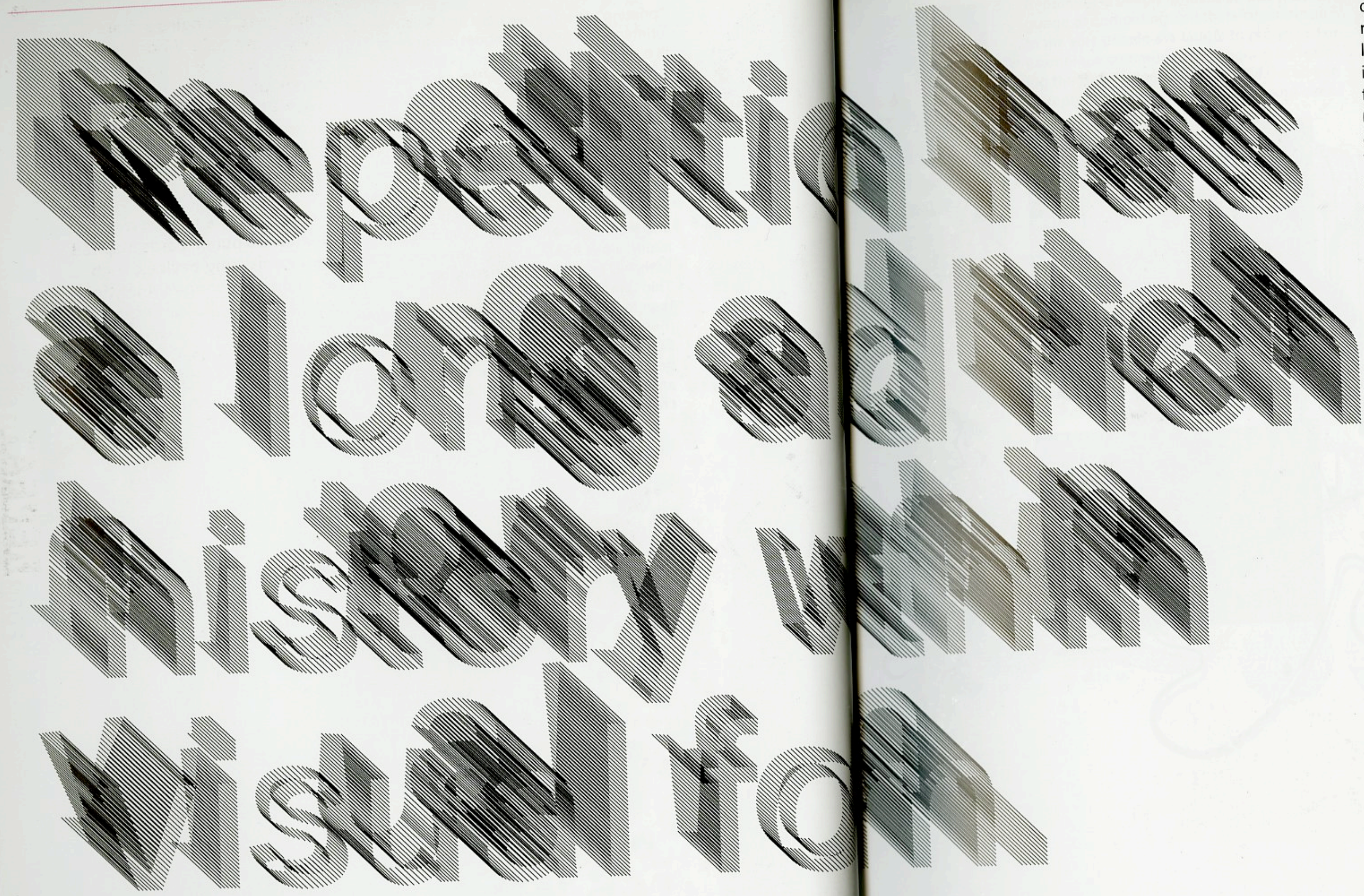
Plotter
Plotters work by moving a drawing implement, typically a pen or brush, over a surface to control its horizontal and vertical positions with two motors. A similar mechanism is used in laser cutters and CNC-milling machines.

Cirrus 2008, by Zaha Hadid Architects, 2008
This sculptural seat, built by Associated Fabrication, is made of milled sheets of Formica and medium density fiberboard.



"printing" an adhesive from an ink-jet-like printer head. After the model is complete, it is excavated from the excess powder, which is then recycled for the next model. Stereolithography and SLS both employ variations of this additive technique. In stereolithography, thin layers of a photopolymer resin are deposited and then cured with an ultraviolet laser to harden the areas where it is focused. Once all of the layers are complete, the remaining liquid is drained and the model undergoes additional curing in ultraviolet light. SLS combines ideas from both 3-D printing and stereolithography. Thin layers of powder are deposited and then fused together using a laser to build the model layer by layer. A distinct advantage of SLS is the wide variety of materials that can be used, including nylon, ceramics, plastic, and metals, making it possible to quickly create prototypes of working machine parts.

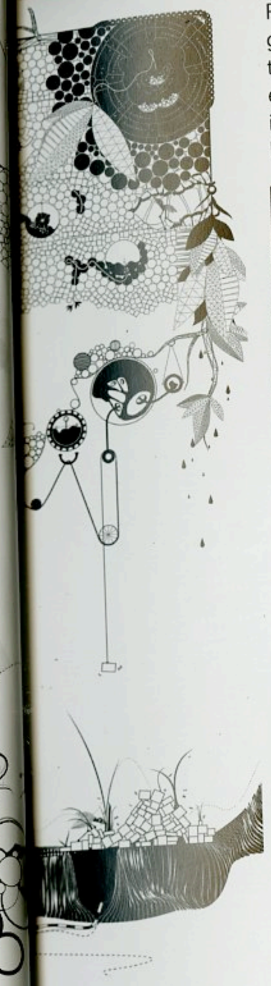
Ecorché structurel, by R&Sie(n)+D, 2008 R&Sie(n)+D imagine possible future worlds and ways of living. This SLS 3-D-printed model is described by its creators as, "The cells were no longer enclosures to protect from the outside...[but]'habitable networks, woven space,' an exfoliation of constantly reconfigured habitable organisms."



Printing technologies are an obvious example. Woodblock reproductions, etchings, and lithographs have all transformed image distribution. As a precursor to computers, the Jacquard loom (invented in 1801) used punch cards to store weaving instructions. These cards guided the machine to weave the same pattern repeatedly. Jacquard's punch cards inspired early computing devices, such as Charles Babbage's Analytical Engine. Today, digital computers are exceptional machines for creating repetition; their state can change over two billion times per second—2 GHz—to perform accurate, reliable calculations.

These letters are composed of a series of lines drawn backward in space, from interpolated points drawn along the outline of each character. The depth of each line was set by an oscillating sine wave. Although this depth increases or decreases only slightly from one point to the next, the order in which the points were originally drawn produces a unique optical effect, while accentuating the anatomy of the original letters.

Here to There,
by Emily Gobeille and Theodore Watson, 2008
This series of large-format posters combines natural and algorithmic forms that balance hand illustration with generated pattern. A suite of software tools serve as building blocks for telling visual stories. Programmed elements are mixed with hand-illustrated forms to create engaging hybrid worlds.



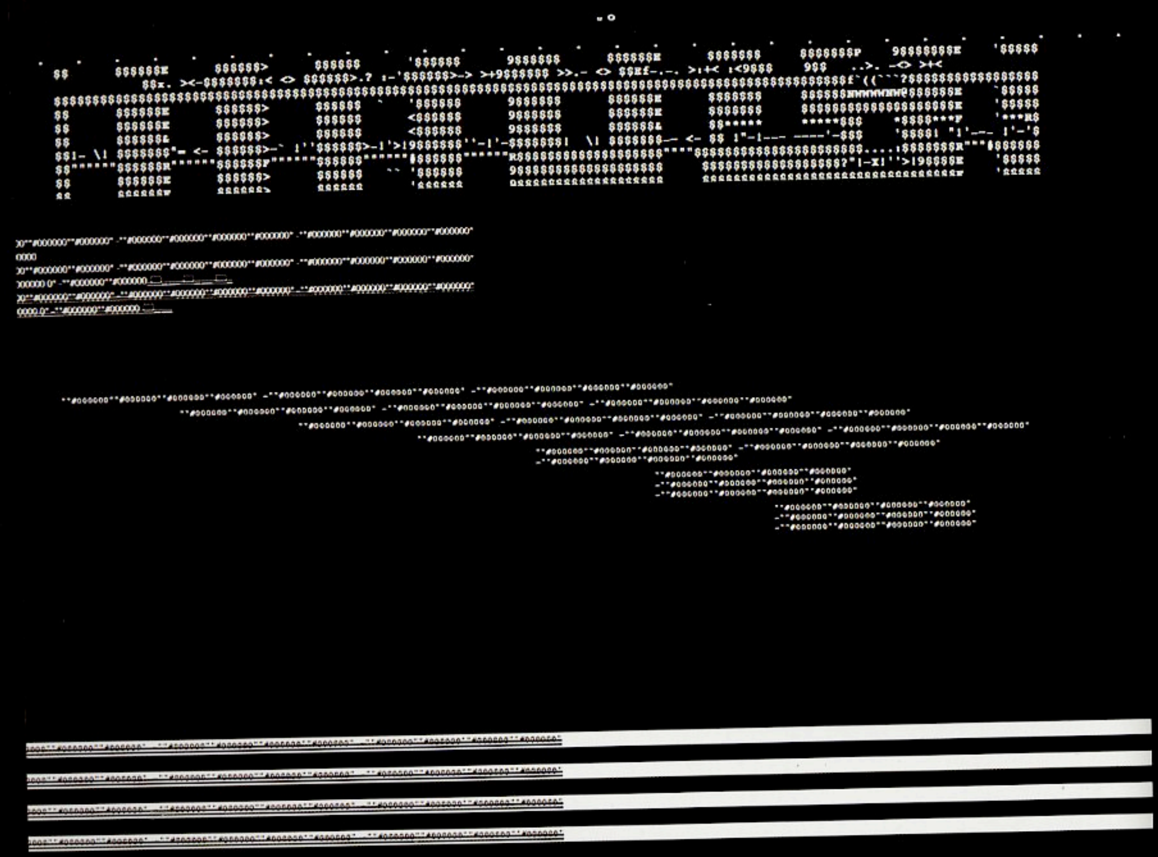
Repetition is deeply embedded into the language of computing and therefore intrinsic to the way people are taught to program. For example, a common early program for learning the BASIC language would fill the screen by reproducing the same text over and over:

```
10 PRINT "REPEAT!"
20 GOTO 10
```

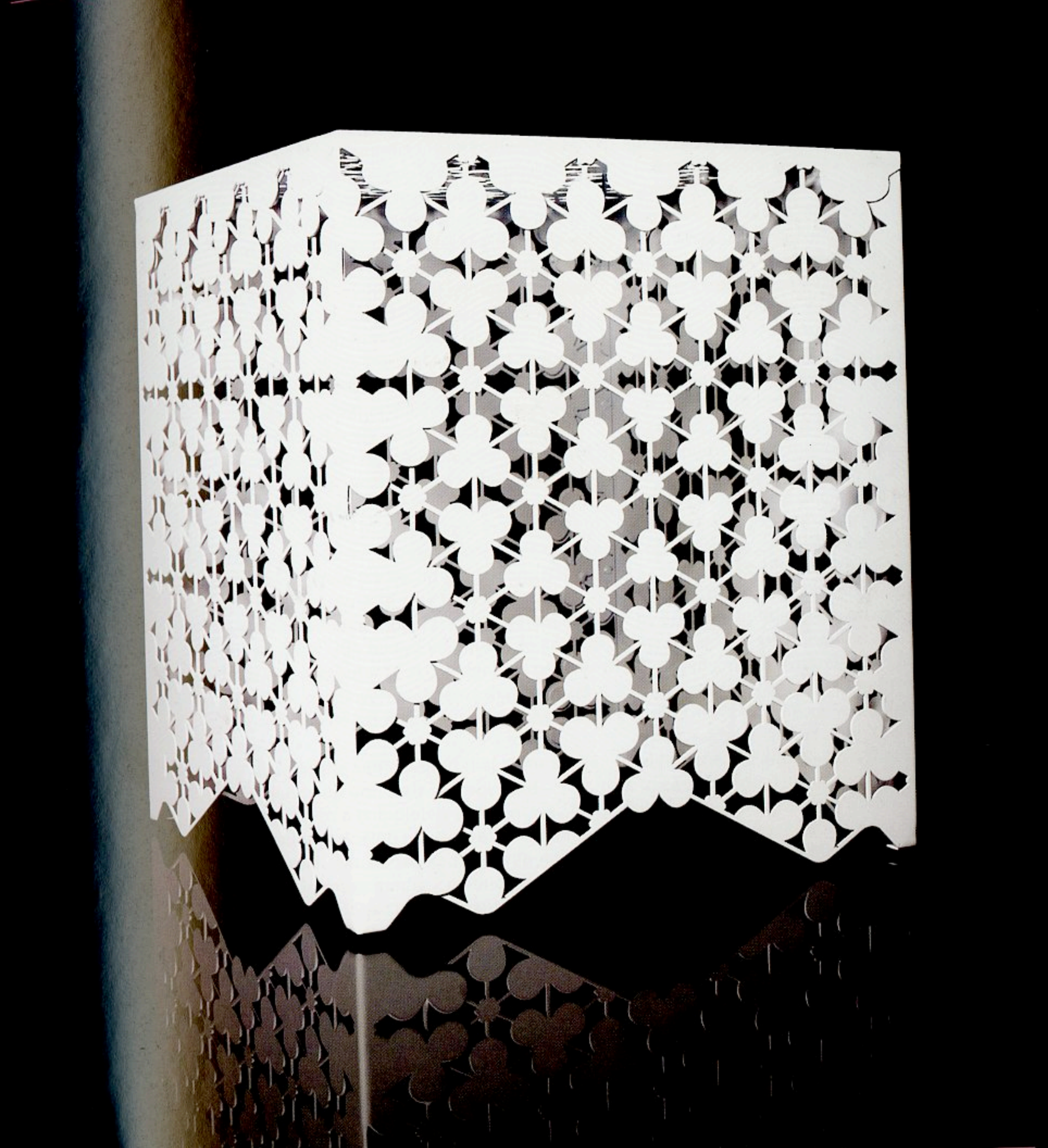
A slight modification opened new paths for exploration:

```
10 PRINT "\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \"
20 PRINT " / / / / / / / / / / / / / / / /"
30 GOTO 10
```

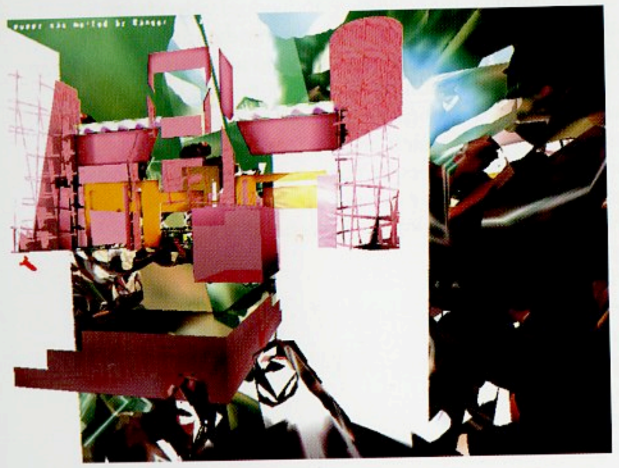
Twenty years ago, a simple moving pattern of letters often provided sufficient motivation to encourage further explorations in programming. Users were attracted to the minimal input of a two-line program and its corresponding output of symbols moving continuously down screen. Programs like these were usually written by hobbyists as well as by children that were first learning how to use computers. Today, most computer users never learn how to program and therefore never feel the thrill of directly controlling a computer. Regardless, repetition is still an inherent part of code, and it continues to be a source of motivation to learn and explore this space of limitless variation.

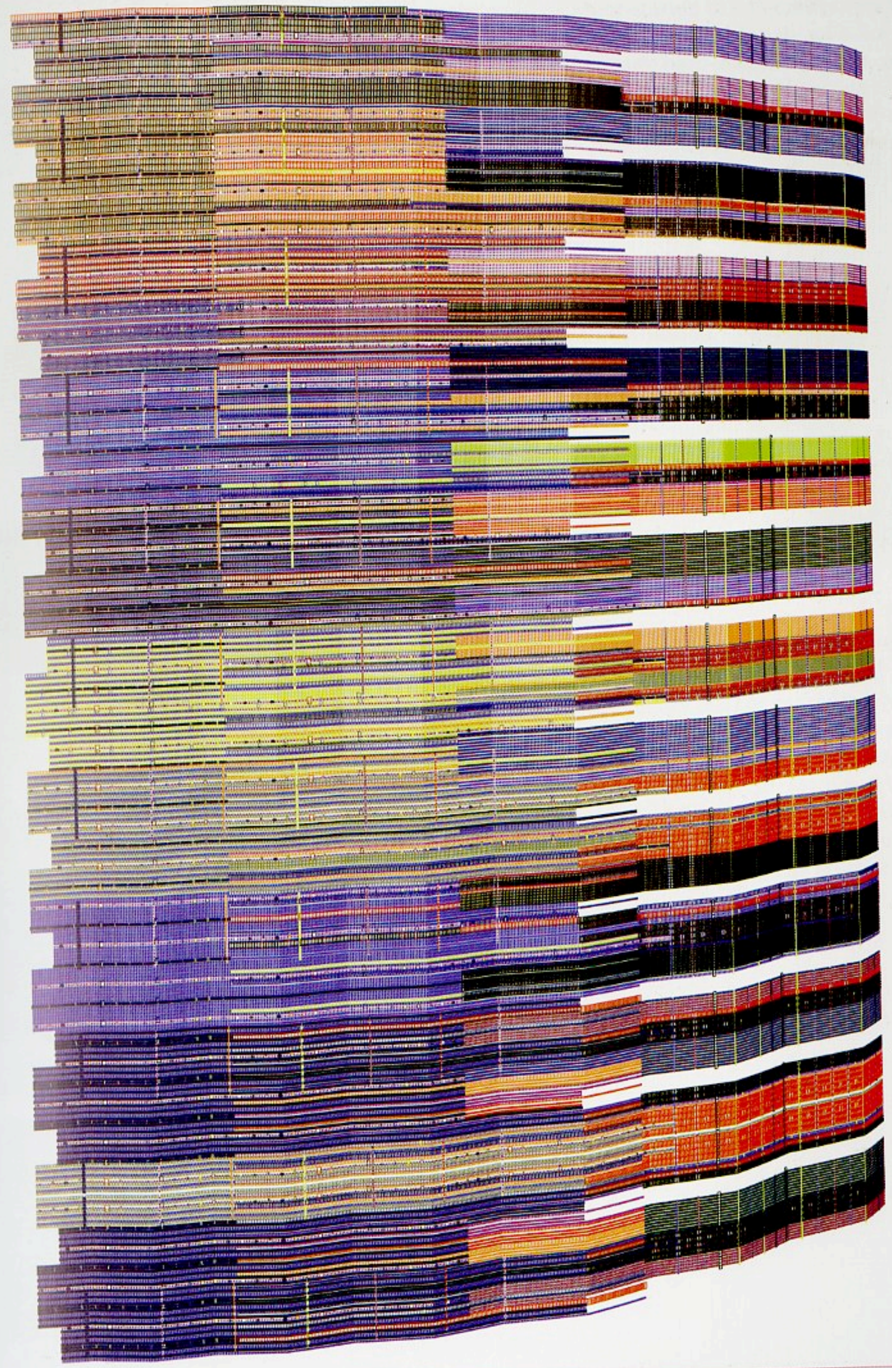


ASDFG,
jodi.org, 1997
ASDFG is a frenetic cacophony of flashing, scanning, and reloading text in a web browser. It is structured as a set of folders embedded in folders, zooming in on the naming limits of the server's operating system. The result is seen in the navigation toolbar, not long enough to display the full lengths of the paths. Looking at the growing browser history, ASDFG unfolds an ASCII path of titles that reflect the on-screen randomness.



Arktura Ricami Stool,
by Elena Manferdini, 2008
The intricate laser-cut pattern in this metal stool makes it appear delicate in contrast to its actual strength. The machine that cut the metal used software to control the position and strength of a laser, based on Manferdini's original digital pattern.





Modell 5,
by Granular-Synthesis, 1994
Short clips and individual audio and video frames are recombined to create an intense performance. Four projected images of a face are sequenced to produce a hybrid machine-human choreography and choir. Editing the video and sound in parallel creates audiovisual synchronicity. The sound of the original video recording is part of each edited frame.

QQQ,
by Tom Betts, 2002
Betts modified the computer game Quake by editing the resource files and transforming its arenas into nearly abstract spaces. By not refreshing the screen, the images accumulate and transform as the player moves through the game.

PSC 31,
by Mark Wilson, 2003
These images explore repeated geometric forms and transformations. They are an extension of Wilson's programmed works from the early 1980s.

Polarity,
by Bridget Riley, 1964
Riley's paintings use repetition and contrast to produce subtle, disorienting effects on the viewer.

QUALITIES OF REPETITION

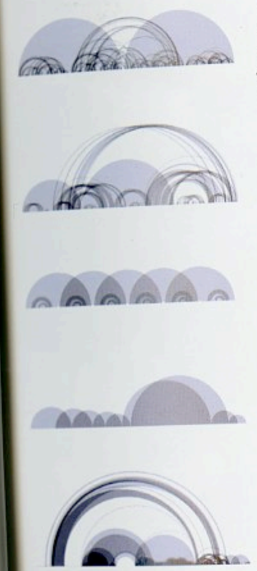
Repetition can have a powerful effect on the human body and psyche. One of the most extreme examples is the way a rapidly flashing light can trigger a seizure. A more universal example is how the beat of a good song will inspire people to dance along. In a similar way, dynamic visual patterns can appear, in subtle ways, to vibrate physically.

Within the visual realm, repetition encourages our eyes to dance. Controlling repetition is a way to choreograph human eye movement. There are many examples of artworks that modulate repetition to create strong sensations of depth and motion. Optical art (often shortened to "op art") is a term used since the early 1960s to describe artworks that induce retinal phenomena, including vibration, flashing, swelling, and warping. Pioneers within this movement include Yaacov Agam, Richard Anuszkiewicz, Bridget Riley, Jesús Rafael Soto, and Victor Vasarely. Though their works were created without the aid of computers, many of them relied on the use of algorithms. For example, Vasarely made preliminary drawings called programmations, in which he explored variations with a modular color system of six hues, each with twelve variations. Instead of using a computer to implement his programs, Vasarely employed assistants that painstakingly followed his instructions to construct the works.

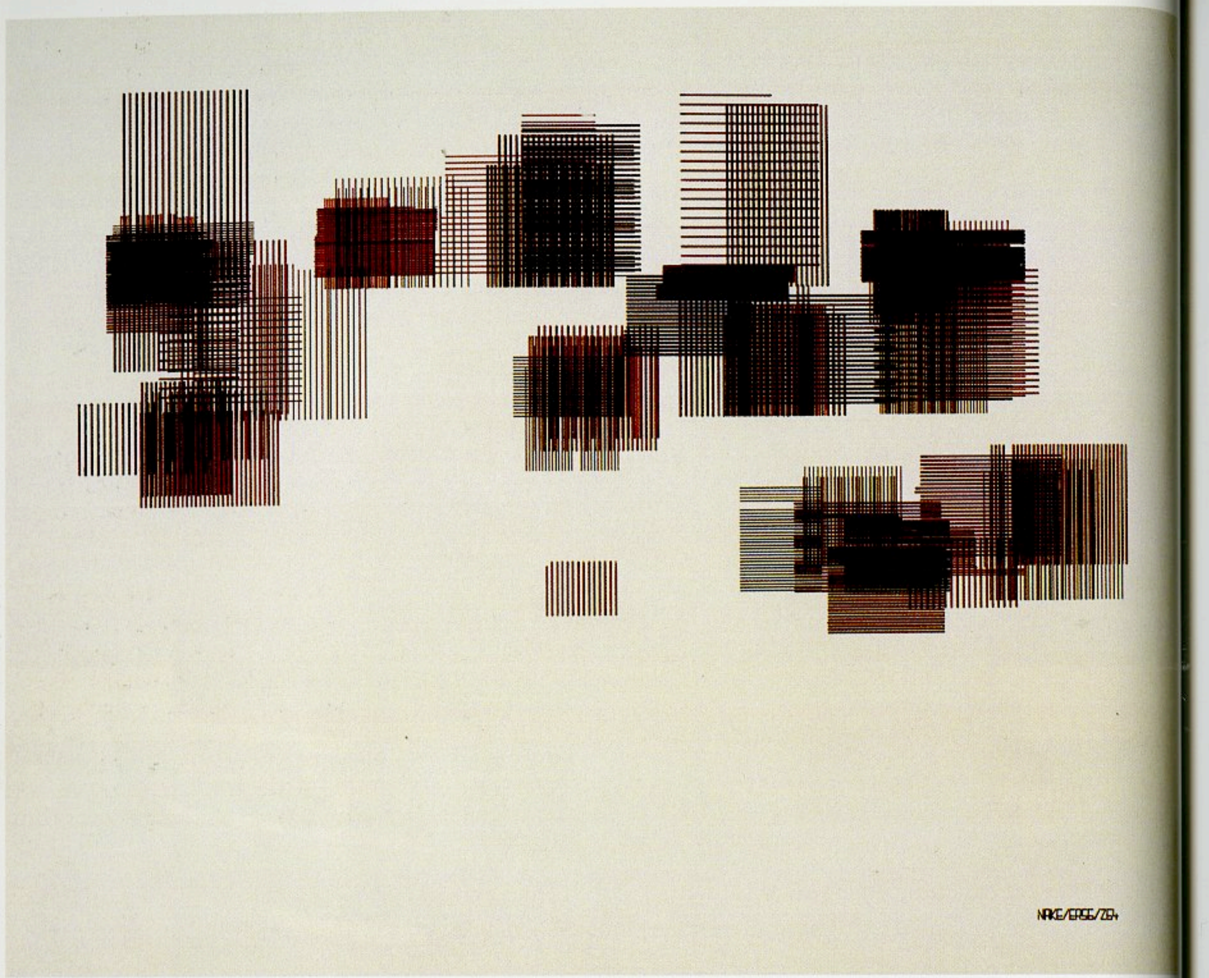
During the same period that witnessed the rise of op art, Andy Warhol used repetition in a completely different way. Instead of inducing physical affects within the human eye, he worked with repetitive images in mass media, creating portraits of iconic celebrities such as Marilyn Monroe, Jacqueline Kennedy, and Elvis Presley by silk-screening a single image many times within the same painting. Through repetition, the image lost its relation to its subject and became a product rather than a portrait.

Beyond visual repetition, setting rhythms in time can have strong, palpable effects. Repetition has always been an important part of music. From classical to contemporary jazz, the repetition of musical phrases within a larger composition is integral. Martin Wattenberg's The Shape of Song software visualizes repetition in music; it's fascinating to see the difference in complexity between Madonna's "Like A Prayer" and Frédéric Chopin's Mazurka in F#.

Repetition can also be an important component within time-based works such as video, animation, and live software. In this capacity, repetition becomes a form of rhythm. The thresholds of rhythm were explored by artist Tony Conrad in the experimental film The Flicker from 1965. This work was made using only plain black and white frames; the film's structure is formed by the number of black frames shown before flipping to white, and vice versa. Conrad pushed the limits of perception by alternating between clear and colored frames—up to twenty-four frames per second (the speed at which film is pulled through a projector). The contemporary performance work Modell 5, by Granular-Synthesis (Kurt Hentschläger & Ulf Langheinrich), builds on this technique by combining image and audio elements into a striking sensorial assault. Without manipulating individual video frames, they transform the repeated image of the performer's face into a writhing posthuman machine by re-sequencing the frames alongside the audio slices that correspond to each image. These works, and many others by contemporary audio-visual artists, explore perception through subtle and violent acts of repetition.



Shape of Song,
by Martin Wattenberg, 2002
This visualization depicts musical passages as arches. Each arch connects identical passages within the composition to expose the patterns that unfold in time as a single image. From top to bottom, these compositions shown are: one of the Goldberg Variations by Johann Sebastian Bach, Frédéric Chopin's Mazurka in F#, the folk song "Clementine," Philip Glass's "Candyman II," and Madonna's "Like A Prayer."



Felder von Rechteck Schraffuren Überlagert, by Frieder Nake, 1965
For this image, Nake used seven random values to control the size, location, orientation, quantity, and pen for each set of lines.

THE COMPUTER'S TALENT

Computers are designed to accurately perform the same calculation over and over. People who write programs to control these machines often utilize this inherent talent. In fact, it is more difficult to work against the computer's electronic precision in order to produce idiosyncratic images. Early computer-generated images often featured the ease of repetition made possible through coding.

Frieder Nake's early visual works are excellent examples of programmed repetition. In the mid-1960s at the University of Stuttgart in Germany, Nake was among the first to use a pen plotter to produce drawings from code for aesthetic reasons. At the time, he wrote programs to generate drawing instructions that he then encoded onto a paper tape. The tape was fed into a Zuse Graphomat Z64 plotter to create a physical image using traditional artist papers and inks. Trained as a mathematician, Nake worked with repetition by modulating random values and applying space-division algorithms.

Vera Molnar and Manfred Mohr are two of the first artists to create custom software to realize their aesthetic concepts. In the 1960s, Molnar was making nonfigurative images composed of basic geometric shapes; she would make drawings, perform small changes, and then evaluate the differences. In 1968, she started to use computers to assist with her work. She wrote about this decision in 1975:

> This stepwise procedure has however two important disadvantages if carried out by hand. Above all it is tedious and slow. In order to make the necessary comparisons in developing series of pictures, I must make many similar ones of the same size and with the same technique and precision. Another disadvantage is that I can make only an arbitrary choice of the modifications inside a picture that I wish to make. Since time is limited, I can consider only a few of many possible modifications.[1]

Mohr started to use computers for similar reasons; he was led to software through his early hard-edge drawings, which were clearly influenced by his training as a jazz musician. For him, the motivation to write software came, in part, from his opinion that the computer was a "legitimate amplifier for our intellectual and visual experiences."[2] He outlined the new possibilities of working with software:

- Precision as part of aesthetical expression.
- High speed of execution and therefore multiplicity and comparativity of the works.
- The fact that hundreds of imposed orders and statistical considerations can be easily carried out by a computer instead of by the human mind, which is incapable of retaining them over a period of time.[3]

Both Molnar and Mohr situated their work within the context of art history and contemporary art. For example, Mohr's work has obvious similarities to conceptual artists working with systems and multiples, such as Sol LeWitt. Molnar wrote about the theme of iteration and slight variation within art, citing Claude Monet's series of haystack paintings as an example.

[1] Ruth Leavitt, Artist and Computer (New York: Harmony Books, 1976), 35.

[2] Ibid., 94.

[3] Ibid., 95.

Interruptions, by Vera Molnar, 1968–69
The prints in the Interruptions series are among Molnar's first software-generated images. She started working with computers in 1968 to produce unique ink on paper plotter drawings to realize her visual ideas.

Daisy Bell, by Jennifer Steinkamp, 2008
This massive undulating wall projection is composed of software models of poisonous flowers. Viewers are often overwhelmed by the detail and scale of repetition made possible through Steinkamp's software.

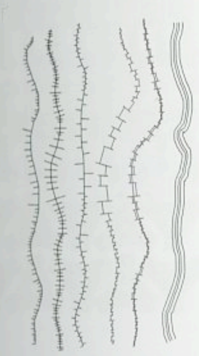
The computers used by these pioneers and their contemporaries were refrigerator-sized machines, which at the time were only available in research and government facilities. Obtaining access to the machines was difficult, and artists had to be very determined. Despite their prohibitive cost, these machines were technically primitive compared to today's computers. The Spartan quality of the early pen-plotter images attests to the visual limitations of these computers and their output devices.

In contrast, the era of raster graphics, enabled by the framebuffer allowed for a different visual quality of repetition. With this technical innovation, the world of programmed graphics transformed from skeletal outlines to worlds of vibrant colors and textures. Computer artist David Em was a pioneer in working with this new type of graphic. Like his predecessors, he worked at research labs to gain access to the high-end computers he needed to produce his work. At the NASA Jet Propulsion Laboratory (JPL) in Pasadena, California, he worked with computer graphics innovator Jim Blinn. Em wrote of the new software: "Blinn's programs, which among other things could display objects with highly textured surfaces, represented a major redefinition of the field of computer imaging."[4] Em used this capacity to work with textures in a simulated 3-D environment in order to produce a series of dense, surreal environments.

This way of working with textures was brought into the home with the Macintosh computer in 1984. The original MacPaint program made it possible to draw with the mouse and to fill these shapes with one-bit textures selected through the patterns palette. The Kid Pix software, released in 1989, built on the ideas introduced in MacPaint but added elements of play and repetition that delighted children (and, of course, many adults too). Graphic icons, ranging from a dinosaur to a strawberry, could be stamped on-screen and easily repeated. This feature enabled a dynamic collage approach to making images.

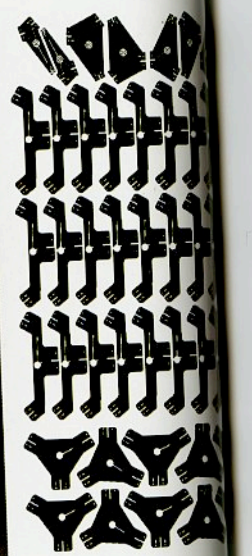
The natural talent of the computer to repeat the same calculations has followed a progression from rendering many lines to creating a population of fully realized, autonomous characters. For example, Massive is used to simulate crowd behaviors such as large-scale battles and stadium audiences, as well as for the creation of contemporary effects for films like The Lord of the Rings trilogy. Today's custom software programs have radically changed the quality of imagery that is produced and consumed.

[4] David A. Ross and David Em, The Art of David Em: 100 Computer Paintings (New York: Harry N. Abrams, 1988), 17.



Volkan, by David Em, 1982
Developed on the most sophisticated computers of the era, Em's images from the late 1970s and early 1980s combine repetitive textures and forms to create fantastical landscapes.

Mobility Agents: A computational sketchbook, by John F. Simon Jr, 2005
This software augments drawn lines by adding new ones in relation to the original gestures. It was inspired by Paul Klee's Pedagogical Sketchbook, and it can reinterpret a single line into many different forms



Ivy, by MOS Architects, 2006
This whimsical system for hanging coats, hats, and other objects uses one standard Y-shaped form and four connector types to provide the owners with the flexibility to create their own structure.

Aperiodic Vertebrae v2.0, by THEVERYMANY / Marc Fornes and Skylar Tibbits, 2008
This architectural prototype is made of 360 panels composed of 11 different types and 320 unique connections. It is held together with zip-tie fasteners.

MODULARITY

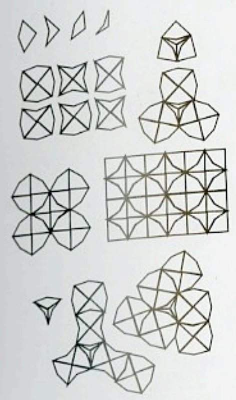
Modularity involves the arrangement of one or more elements to produce a multitude of forms. (It is related to parameters in that the elements are not transformed; they are simply repositioned.) These two themes blend together. Most typefaces are good examples of modular structures. Their range of visual forms is created through a few basic shapes. For example, the lower-case letters p, q, and b are built by arranging the same elliptical and vertical forms in different ways. Some alphabets are more modular than others. The alphabet designed by De Stijl founder Theo van Doesburg in 1919 and the New Alphabet created by Wim Crouwel in 1967 are examples of extremely modular typefaces.

In software, modularity is often used as a strategy for optimization. Because storage space and bandwidth are always limited, a small set of graphics is repeated to generate larger images. This technique is used to produce complex, vibrant images from a small group of forms. For example, when bandwidth was extremely limited in the early days of the web in the mid-1990s, it could take minutes to download graphically intense websites. To decrease the download time, many sites used small repeating images as background textures. Video games have a long history of using a small set of graphics to create large worlds. One of the most famous examples, Super Mario Bros., constructs the game environment using only a small set of 8-by-8 pixel "image tiles" that are stored directly on the game cartridge as raw data. These tiles are combined and recombined to move the characters and create all of their motions. To make this system even more complex, the game machine allows only 64 tiles to be used at a time. Ben Fry's Mario Soup software reconstructs these images as they are stored on the Nintendo cartridge. His companion software, Deconstructulator, shows how the tiles are moved in and out of the machine's memory while the game is being played.

Within the context of physical objects and manufacturing, modularity is used to reduce cost and to make complex building projects feasible. Although some high-profile design and architecture projects are built entirely with custom-manufactured parts, most budgets require working with a set of standard pieces. In fact, most buildings are constructed from standardized, prefabricated elements. The visionary structures of Buckminster Fuller pushed this idea to the extreme in the 1950s. His geodesic dome designs for homes and city-sized structures were built from uniform elements.

The modular coat hook system called Ivy, designed by MOS (an architecture firm led by Michael Meredith and Hilary Sample), is an excellent example of using software to explore a design space of fixed parts. The product comes in a small plastic bag, ready to be assembled into a wall sculpture. It includes sixteen Y-shaped elements and four types of connectors that can be assembled in myriad ways. A software simulation on the MOS website uses a layout algorithm to explore possible configurations of the system.

Beyond the regular repetitions demonstrated here, computational machines (i.e. computers) can produce form with endless variation. This property is discussed in depth in the Parameterize chapter.



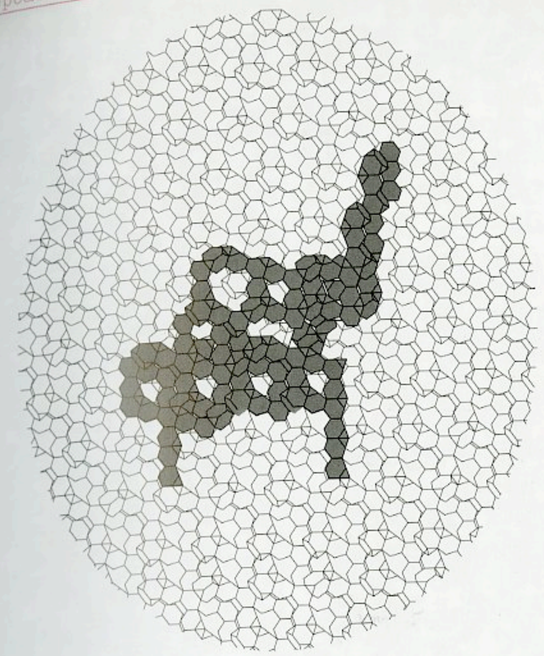
Minimum Inventory, Maximum Diversity diagram, by Peter Pearce, 1978
Pearce's book Structure in Nature Is a Strategy for Design makes a strong case for the technique of using a minimum number of elements to create a range of diverse forms. Here, four shapes are used as the basis for all of these structures.

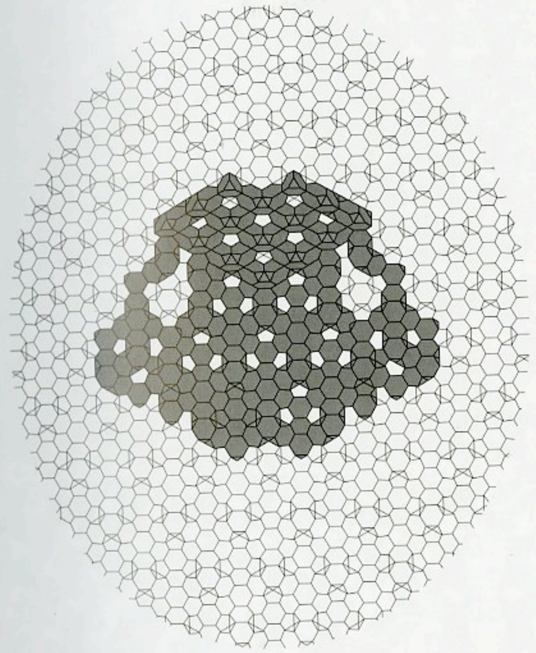
Mario Soup, by Ben Fry, 2003
This software shows how all of the graphics used in Nintendo's 1985 Super Mario Bros. game are stored within two matrices. In this image, one matrix is shown as red and the other as blue. The colors used in the game are applied while the game is running.

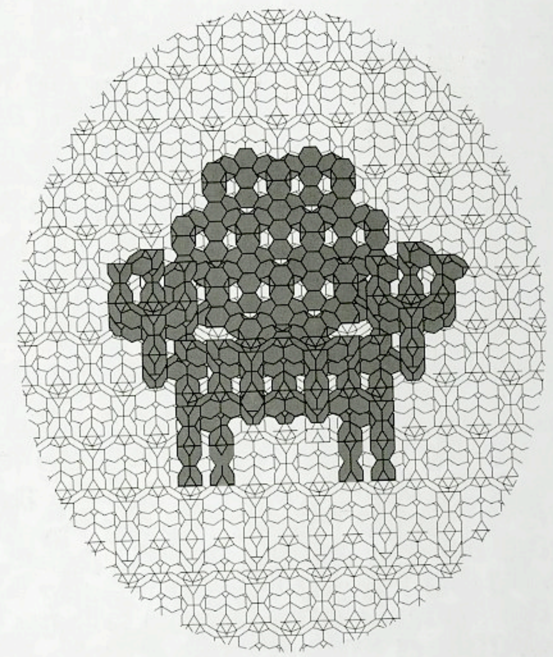
REPETITION TECHNIQUE

PATTERN

All visual patterns and tessellations at their core are composed of algorithms. Even centuries-old patterns, such as Scottish tartans, follow strict compositional rules that are capable of being encoded into software. Writing code is an exciting way to approach visual patterns. Repetitive patterns are used extensively for applications requiring the illusion of a continuous image, such as textiles and wallpapers. These patterns can be extremely ornate and complex like William Morris' wallpapers, or clean and simple like many of the textile designs by Charles and Ray Eames. New rapid-prototyping machines and computer-controlled fabrication equipment make it possible to explore this area even further.

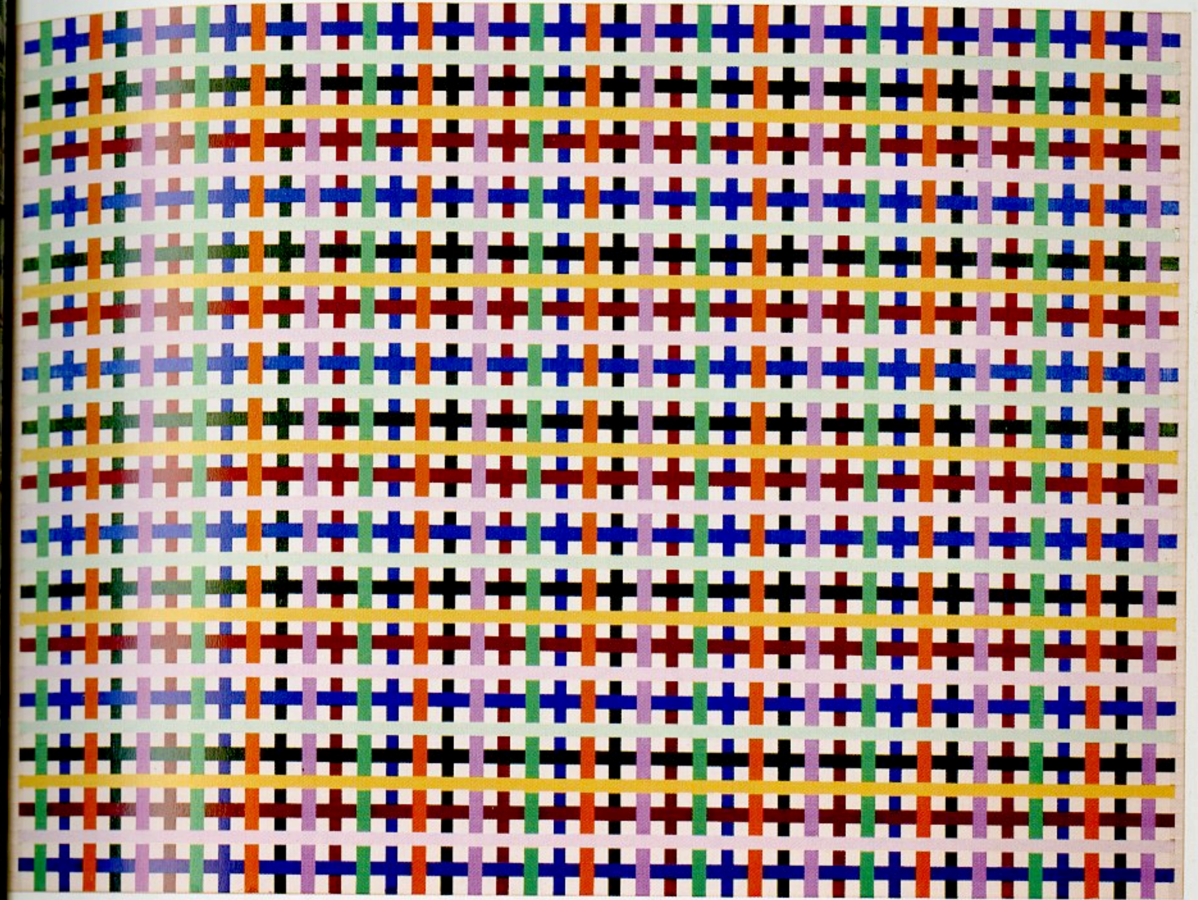






1774 Series Fauteuil, by Aranda\Lasch, 2007 The form of this aluminum chair was "found" within the repeating pattern of an enlarged model of a manganese oxide lattice. The shape of the chair is based on a Louis XV-style armchair.

Whirligig,
by Zuzana Licko, 1994
Licko composed the 152 Whirligig characters as building blocks for infinite pattern variations. Because it is packaged as a typeface, composing a Whirligig pattern is as simple as typing. The repetition works on both the micro and macro scales. To create each element, a simple form is repeated and the elements are combined to form second-order patterns, as the positive and negative shapes of the elements connect.

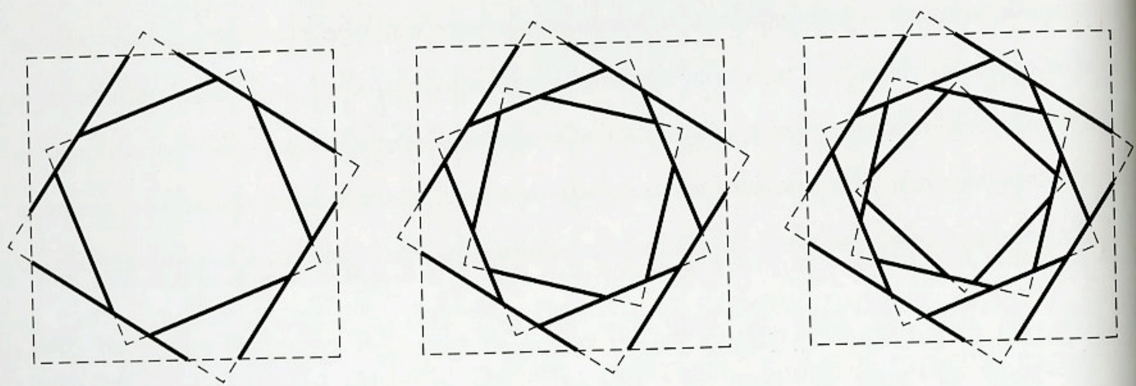


REPEAT

Painting #207 - N,
by Vasa Mihich, 2004
Mihich is a sculptor and painter, but he started sketching with computers in 1998.

He works with fixed algorithms that sometimes introduce the element of chance. This painting was composed with the following rules:

NINE COLORS WERE DIVIDED INTO THREE VALUE GROUPS:
BLUE/GREEN/RED, VIOLET/ORANGE/TURQUOISE, AND LIGHT ORANGE/LIGHT VIOLET/
LIGHT BLUE. RED WAS FIRST. BLUE WAS SECOND. GREEN WAS THIRD.
VIOLET, ORANGE, AND TURQOISE WERE ARRANGED VERTICALLY.
LIGHT ORANGE, LIGHT VIOLET, AND LIGHT BLUE WERE ARRANGED HORIZONTALLY.

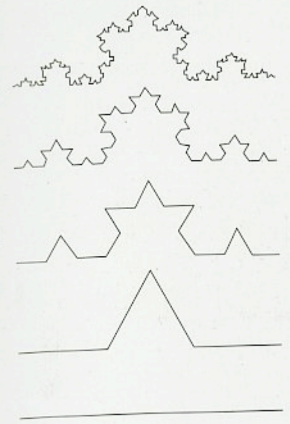


Serpentine Gallery Pavilion, by Toyo Ito & Associates, Architects, and Arup, 2002

The rhythmic lines of Ito's pavilion resulted from a recursive system of rotated concentric squares. Arup helped to create a pattern of beams that was structurally sound and preserved the chaotic look of the building.

REPETITION TECHNIQUE

RECURSION



The technique of recursion is an extremely powerful tool for generating form. Using a broad definition, recursion is a process of repeating objects in a self-similar way. A fern leaf is an example of a recursive form; each leaf is composed of a series of smaller and smaller leaves. A joke about the definition of recursion gets the point across:

Recursion
See "Recursion"

A more technical definition within the context of code defines it as a function that includes a reference to itself as a part of the function. This is a potent technique, but it can be difficult to control. The definition points out the potential problem: it can cause an infinite loop, unless there is a condition to break out of the cycle.

The Koch Snowflake example clearly shows how the idea of recursion is used to create a complex form from a simple base element. At each level of the recursion, a straight line is replaced by a four-segment triangular bump. This powerful process clearly emulates nature and can be applied to many other situations.

REPEAT

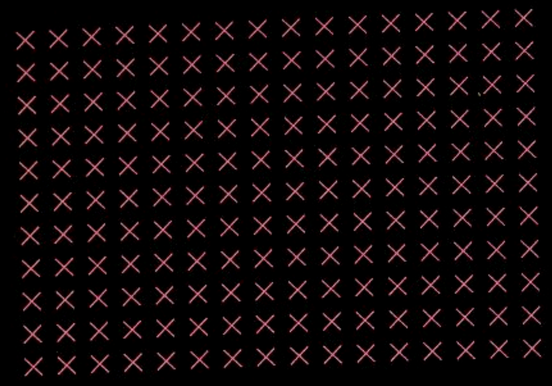
F → F[+F]F[-F]F   F → F[+F]F[-F][F]   F → FF-[-F+F+F]+[+F-F-F]

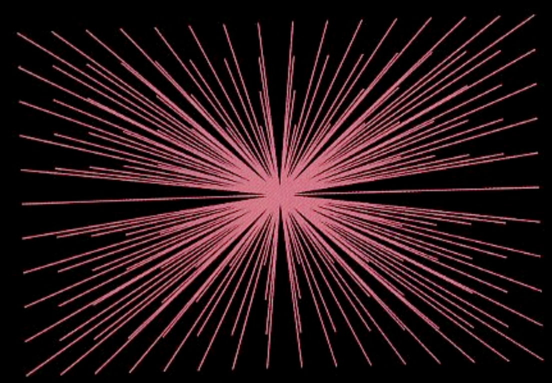
L-Systems, first introduced by Aristid Lindenmayer, 1968 Lindenmayer systems (L-Systems) are an elegant way to simulate plant forms. A starting pattern is replaced according to a set of rules, and it is then transformed again and again.

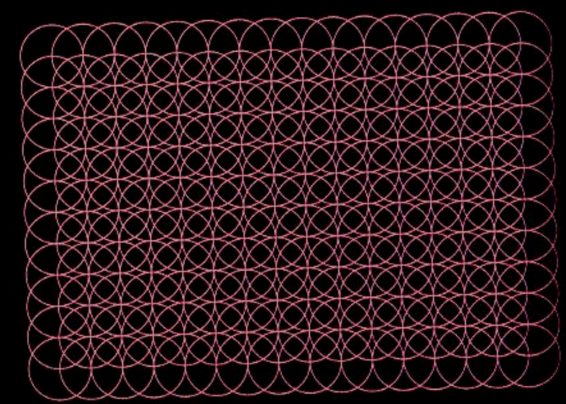
CODE EXAMPLES
EMBEDDED ITERATION

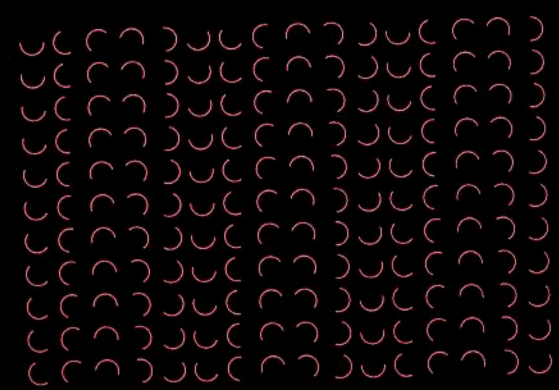
All programming languages can repeat an action, such as drawing the same shape over and over. When one repetition sequence is embedded within another, the effect multiplies. For example, if the action of drawing five lines is repeated ten times, fifty lines are drawn. This simple technique can be used to explore many kinds of patterns.

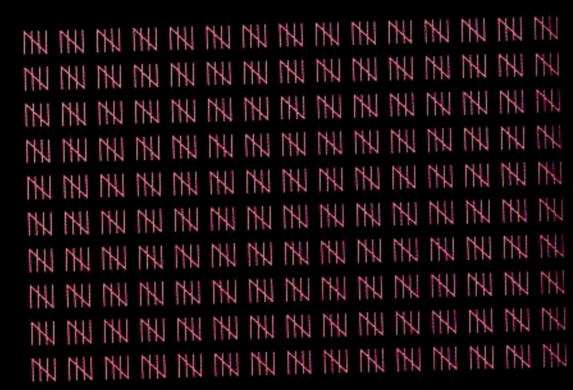
Each of these images was generated from the same gr[illegible] of points. Sixteen elements along the x-axis and eleven along the y-axis combine to form 176 coordinates. Changing just one line of code produces the differenc[illegible] between theses images.

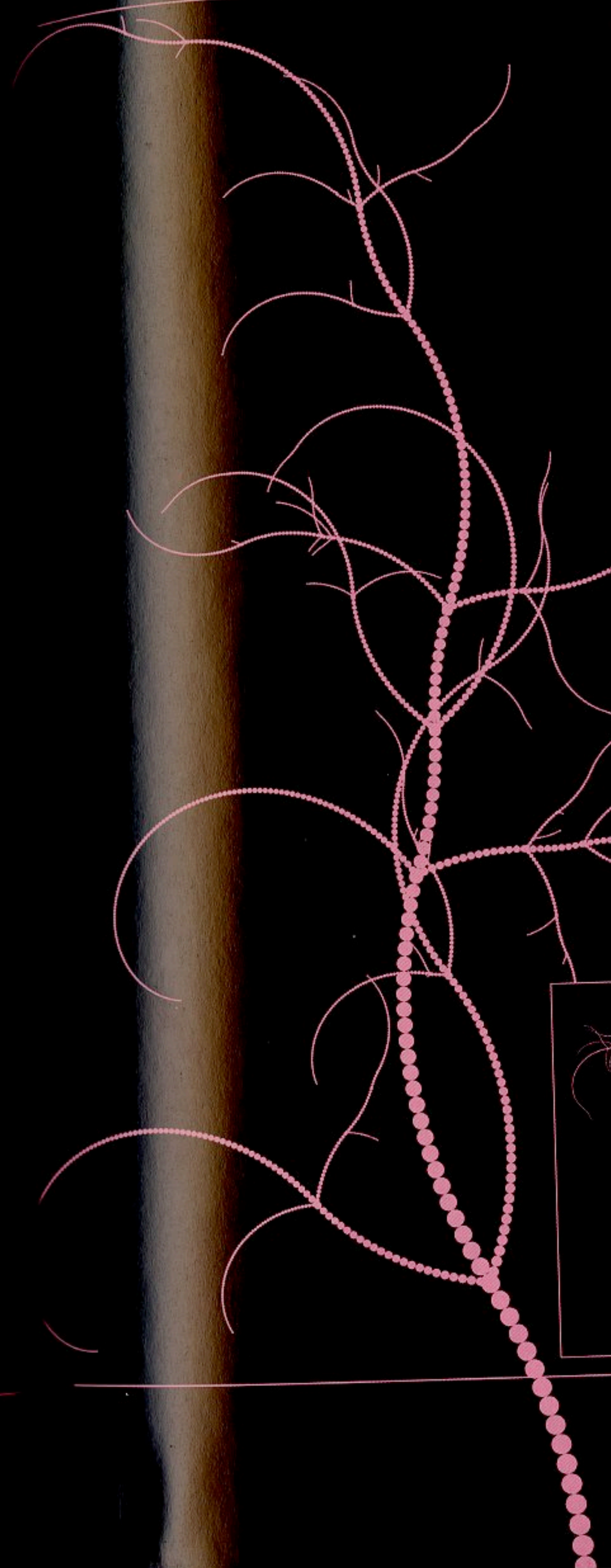










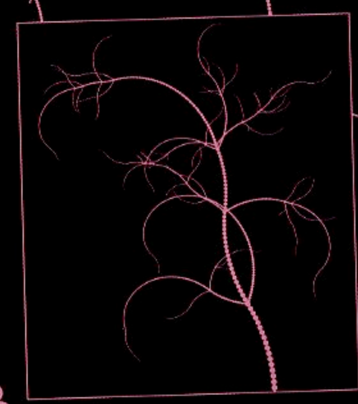


CODE EXAMPLES
RECURSIVE TREE

Within the domain of image making, recursion involves drawing a group of related, self-similar shapes. Treelike forms are a clear example—each branch spawns many smaller branches, which in turn spawn more branches. As a simple example, draw the letter Y on a sheet of paper. Now draw two smaller Ys sprouting from the top of each branch. After repeating this process a few times, the number of Ys drawn at each step has increased from 1 to 2, to 4, 8, 16, 32, 64, and so on.

While this type of tree makes a predictable shape, adding a small amount of randomness to the line lengths and number of branches can yield more organic forms.

These treelike forms were created by drawing one circle at a time. Starting at the base, each circle was slightly rotated and scaled relative to the one before it. At random intervals during the growth, two smaller branches sprout to form new growths. This continues until the circles reach the minimum size set by the user.



Download the code at http://formandcode.com