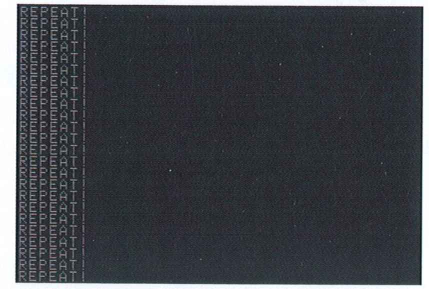
Printing technologies are an obvious example. Woodblock reproductions, etchings, and lithographs have all transformed image distribution. As a precursor to computers, the Jacquard loom (invented in 1801) used punch cards to store weaving instructions. These cards guided the machine to weave the same 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.

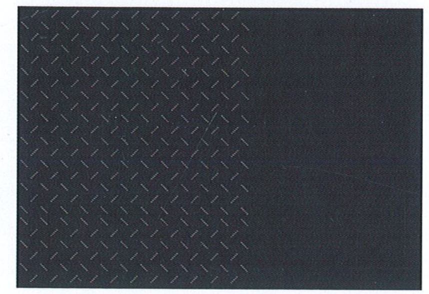
Here to There, by Emily Gobeille and Theodore Watson, 2008
This series of large-format posters combines natural and algorithmic forms that balance hand illustration with generated pattern. A suite of software tools serve as building blocks for telling visual stories. Programmed elements are mixed with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



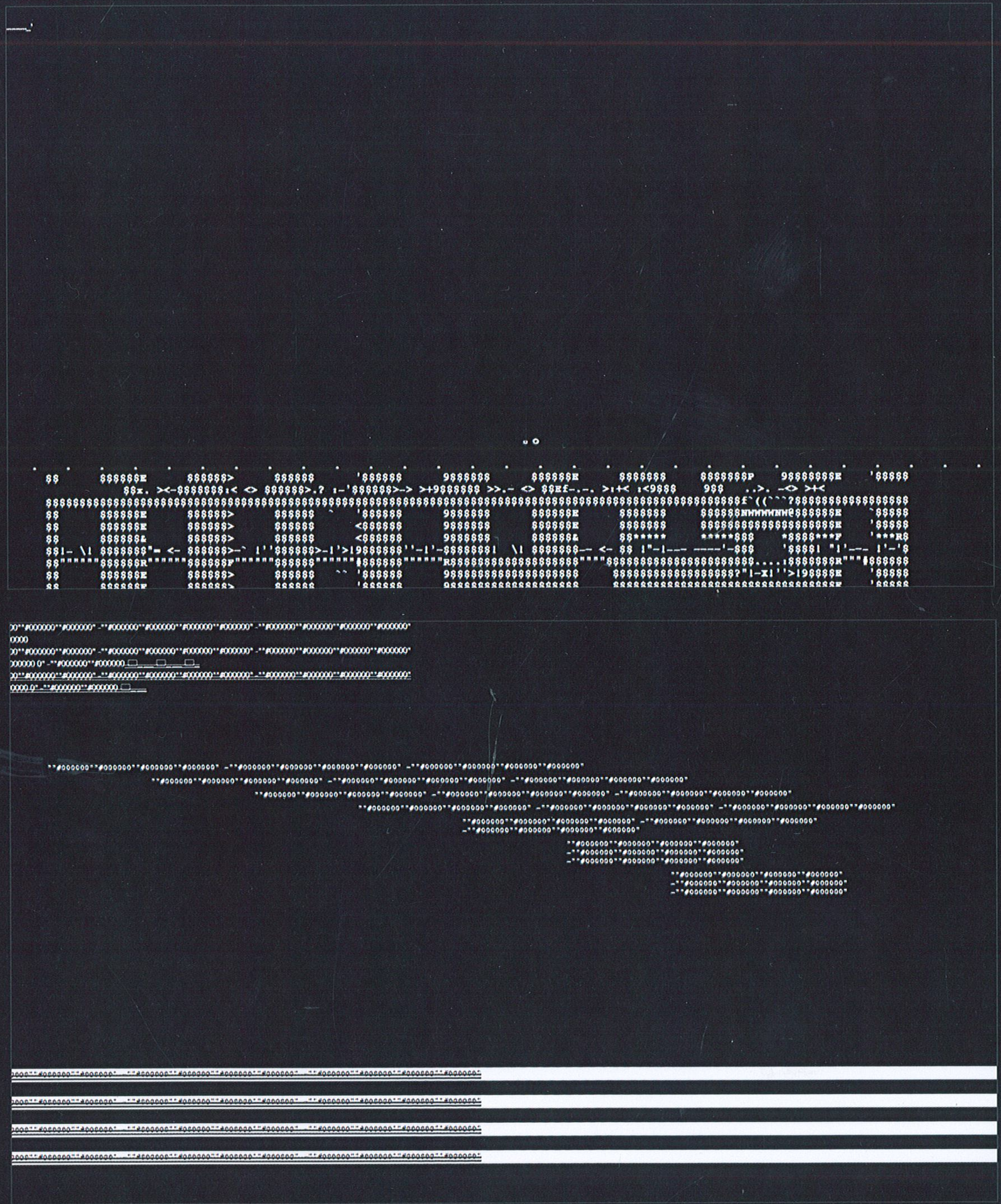
```
10 PRINT "REPEAT!"
20 GOTO 10
```

A slight modification opened new paths for exploration:

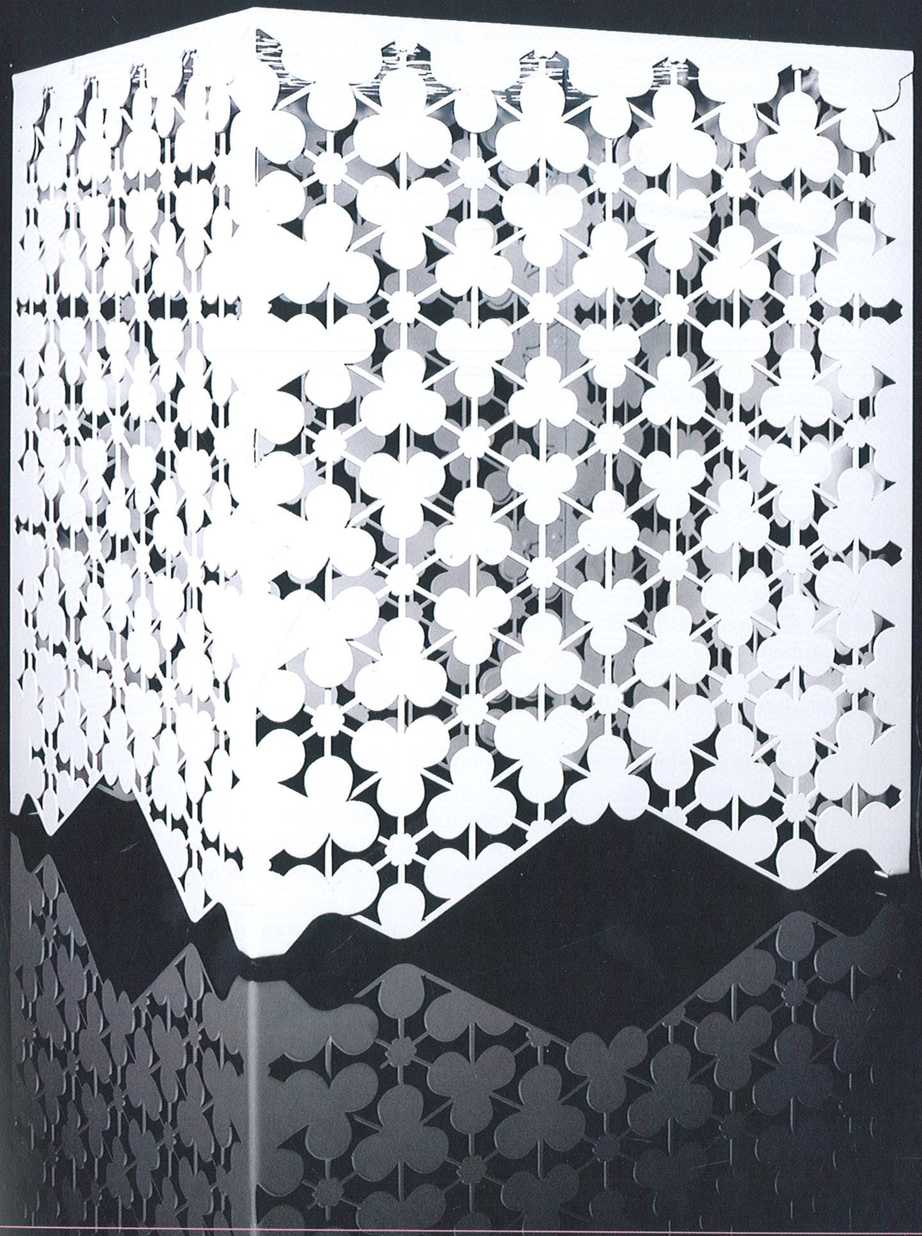


```
10 PRINT "\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \"
20 PRINT " / / / / / / / / / / / / / / / /"
30 GOTO 10
```

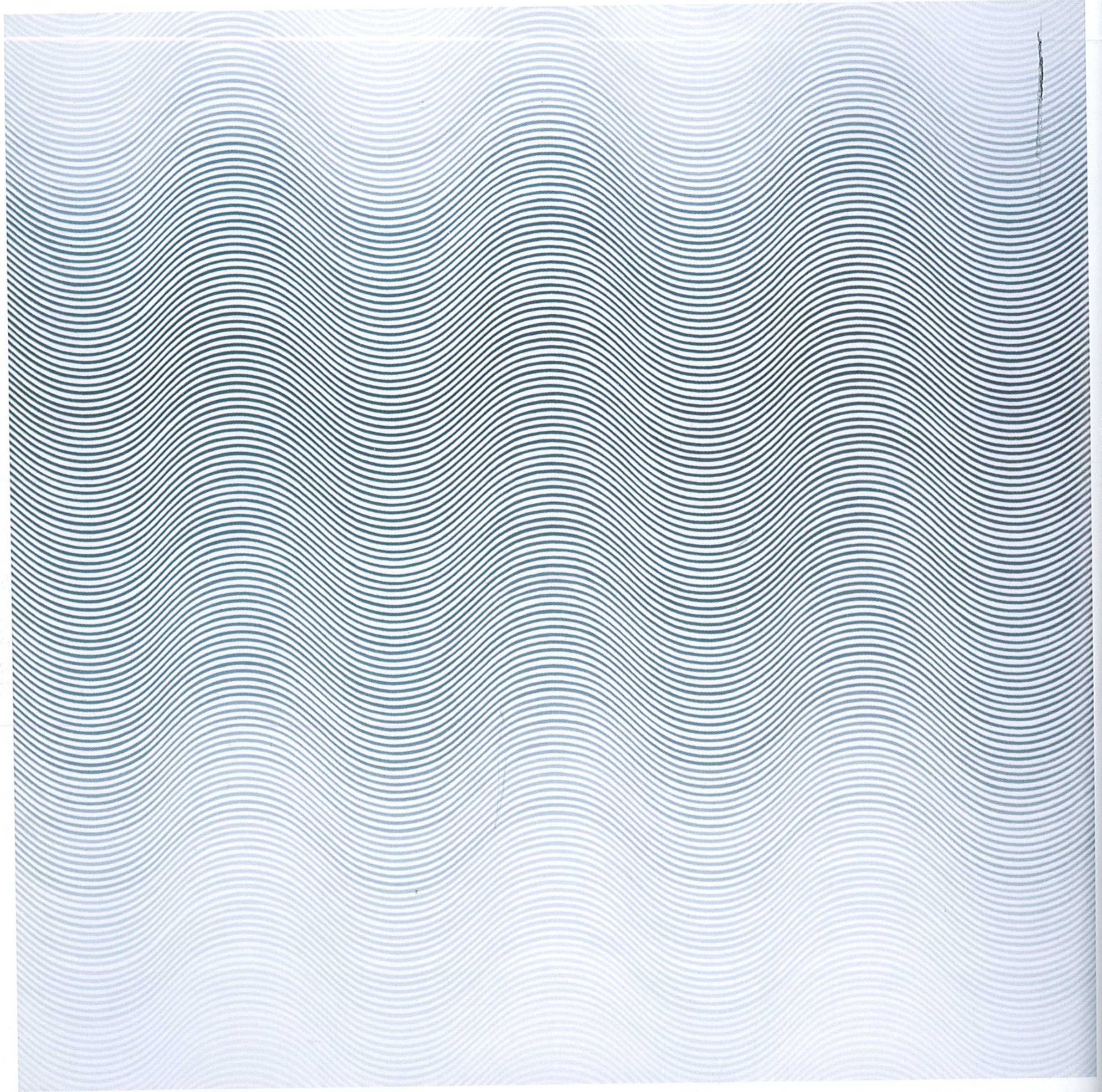
Twenty years ago, a simple moving pattern of letters often provided sufficient motivation to encourage further explorations in programming. Users were attracted to the minimal input of a two-line program and its corresponding output of symbols moving continuously down screen. Programs like these were 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.



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

QUALITIES OF REPETITION

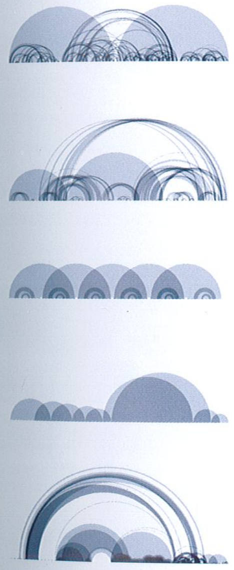
Repetition can have a powerful effect on the human body and psyche. One of the most extreme examples is the way a rapidly flashing light can trigger a seizure. A more universal example is how the beat of a good song will inspire people to dance along. In a similar way, dynamic visual patterns can appear,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

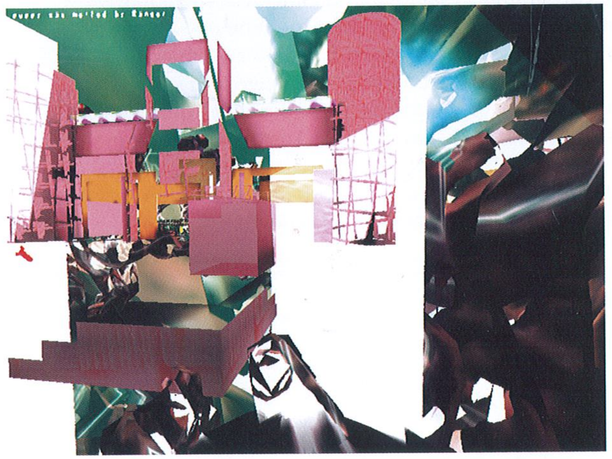
During the same period that witnessed the rise of op art, Andy Warhol used repetition in a completely different way. Instead of inducing physical affects within the human eye, he worked with repetitive images in mass media, creating portraits of iconic celebrities such as Marilyn Monroe, Jacqueline Kennedy, 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.

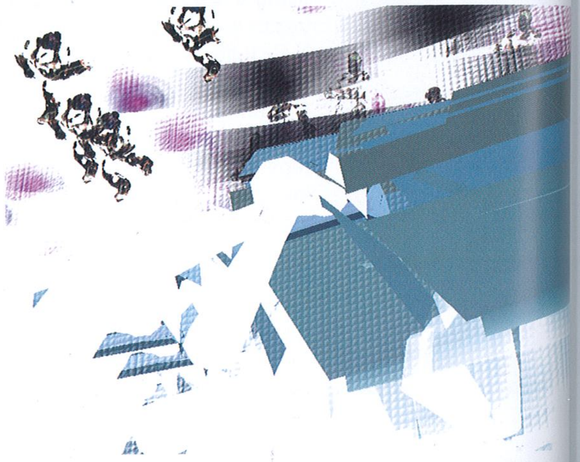
Beyond visual repetition, setting rhythms in time can have strong, palpable effects. Repetition has always been an important part of music. From classical to contemporary jazz, the repetition of musical phrases within a larger composition is integral. Martin Wattenberg's The Shape of Song software visualizes 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.



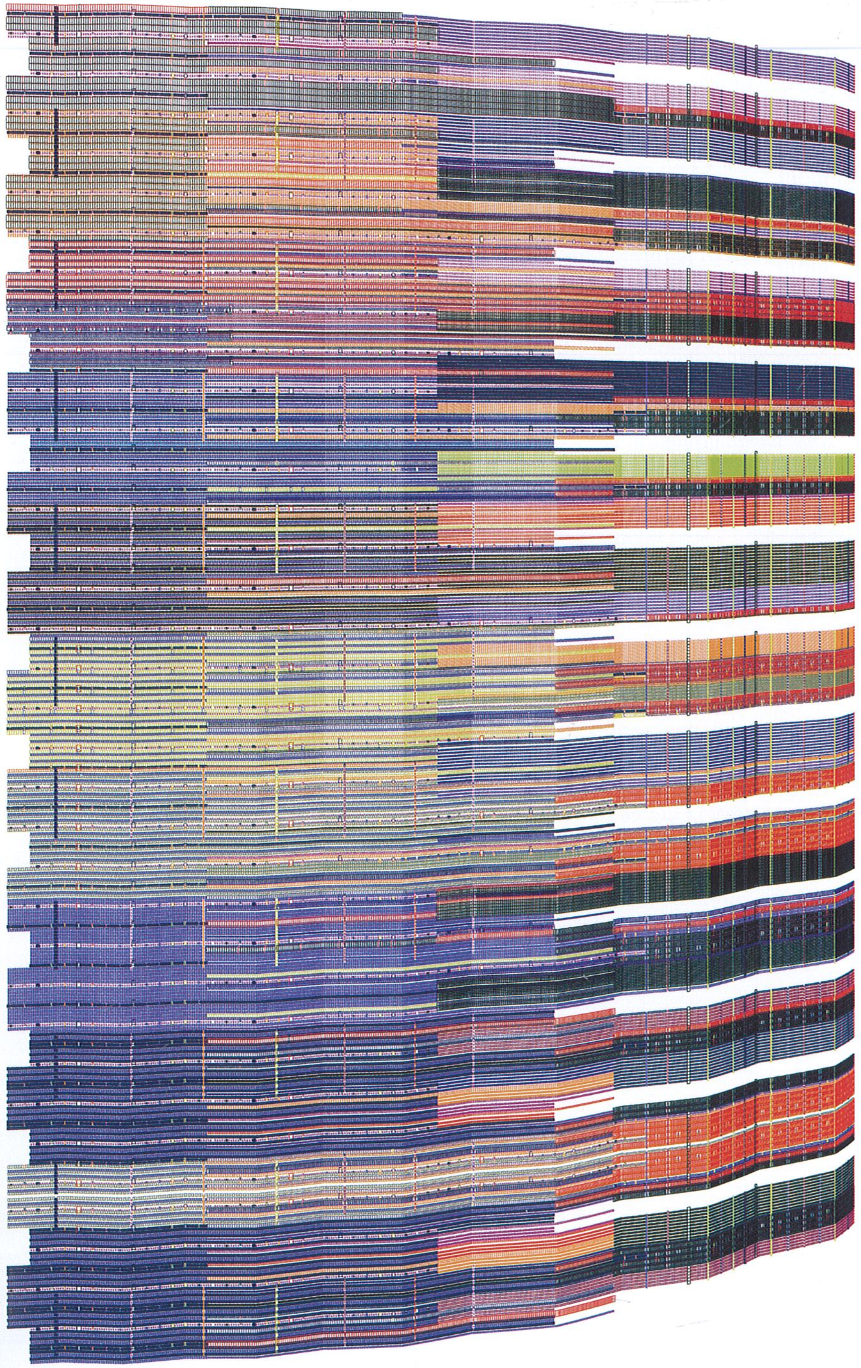
Shape of Song,
by Martin Wattenberg, 2002
This visualization depicts musical passages as arches. Each arch connects identical passages within the composition to expose the patterns that unfold in time as a single image. From top to bottom, these compositions shown are: one of the Goldberg Variations 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."



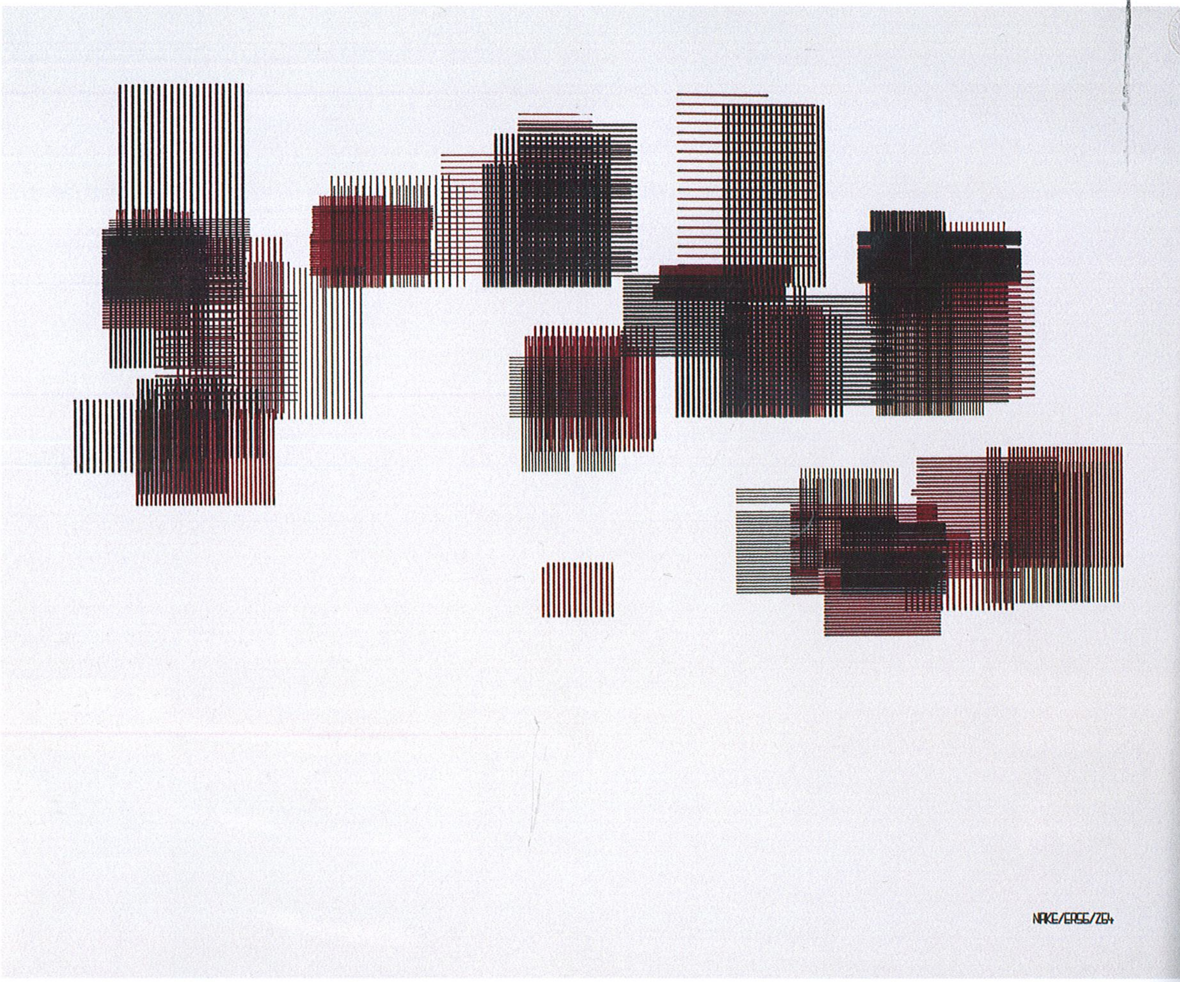


Modell 5, by Granular-Synthesis, 1994
Short clips and individual audio and video frames are recombined to create an intense performance. Four projected images of a face are sequenced to produce a hybrid machine-human choreography and choir. Editing the video and sound in parallel creates audiovisual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



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:

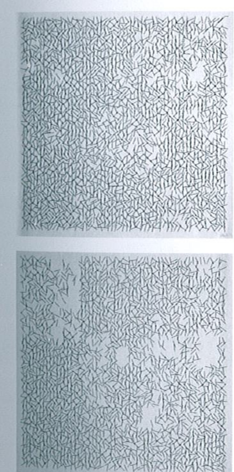
— Precision as part of aesthetical expression.
— High speed of execution and therefore multiplicity and comparativity of the works.
— The fact that hundreds of imposed orders and statistical considerations can be easily carried out by a computer instead of by the human mind, which is incapable of retaining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

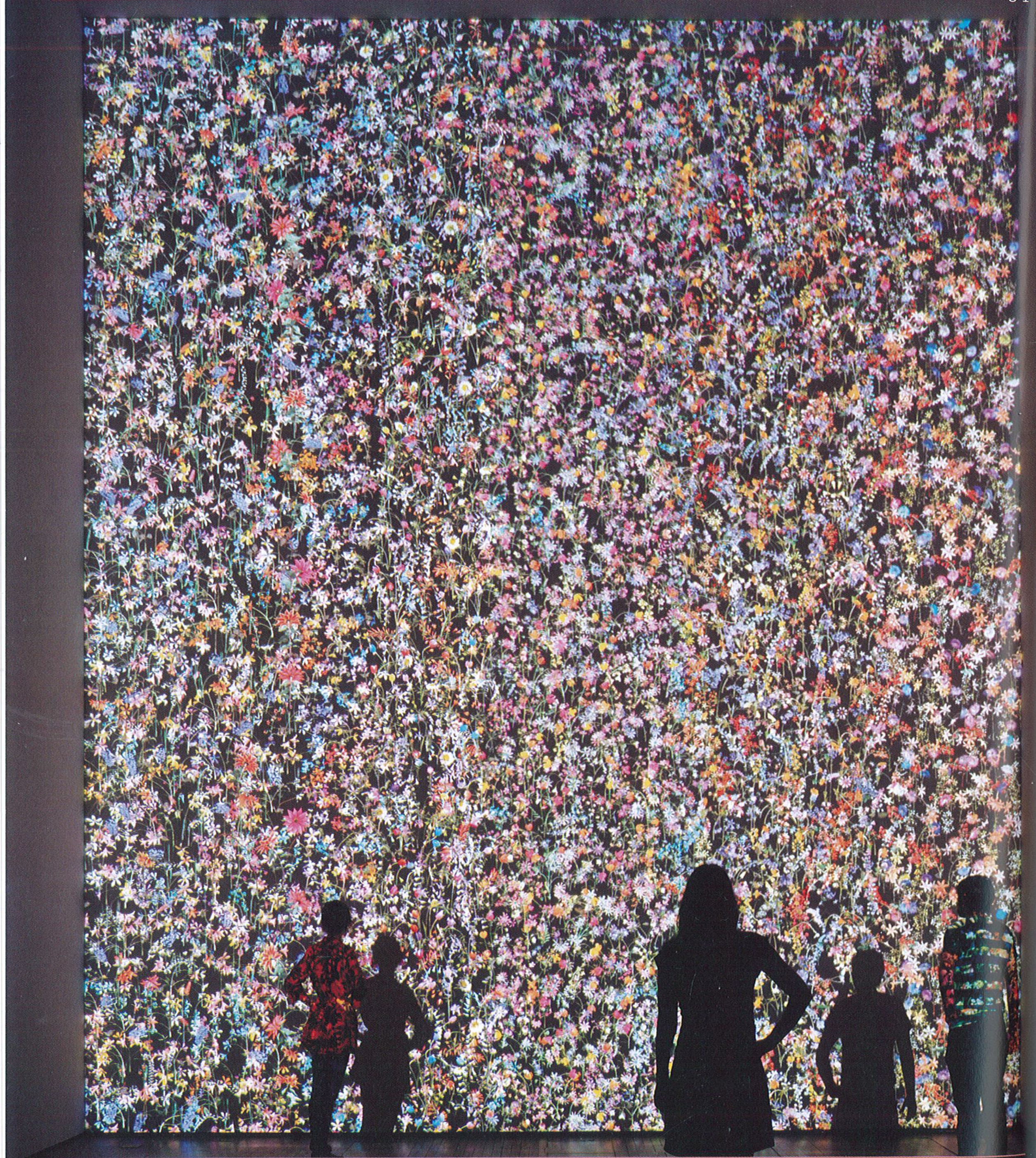
[1] Ruth Leavitt, Artist and Computer (New York: Harmony Books, 1976), 35.

[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.

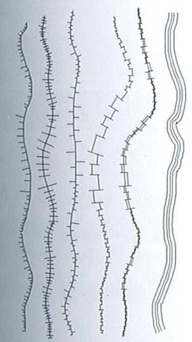
The computers used by these pioneers and their contemporaries were refrigerator-sized machines, which at the time were only available in research and government facilities. Obtaining access to the machines was difficult, and artists had to be very determined. Despite their prohibitive cost, these machines 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.

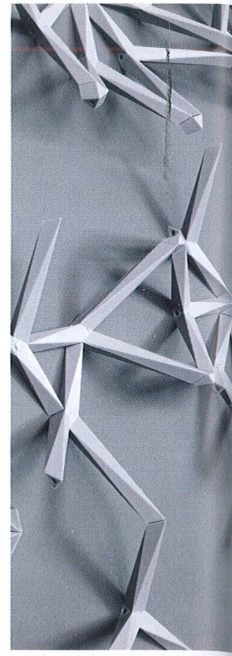
The natural talent of the computer to repeat the same calculations has followed a progression from rendering many lines to creating a population of fully realized, autonomous characters. For example, Massive is used to simulate crowd behaviors such as large-scale battles and stadium audiences, as well 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.

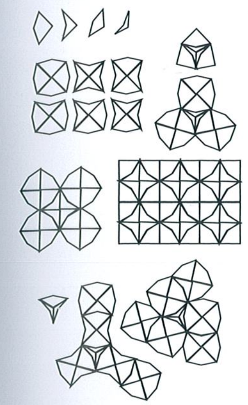
Mobility Agents: A computational sketchbook, by John F. Simon Jr, 2005
This software augments drawn lines by adding new ones in relation to the original gestures. It was inspired by Paul Klee's Pedagogical Sketchbook, and it can reinterpret a single line into many different forms



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.

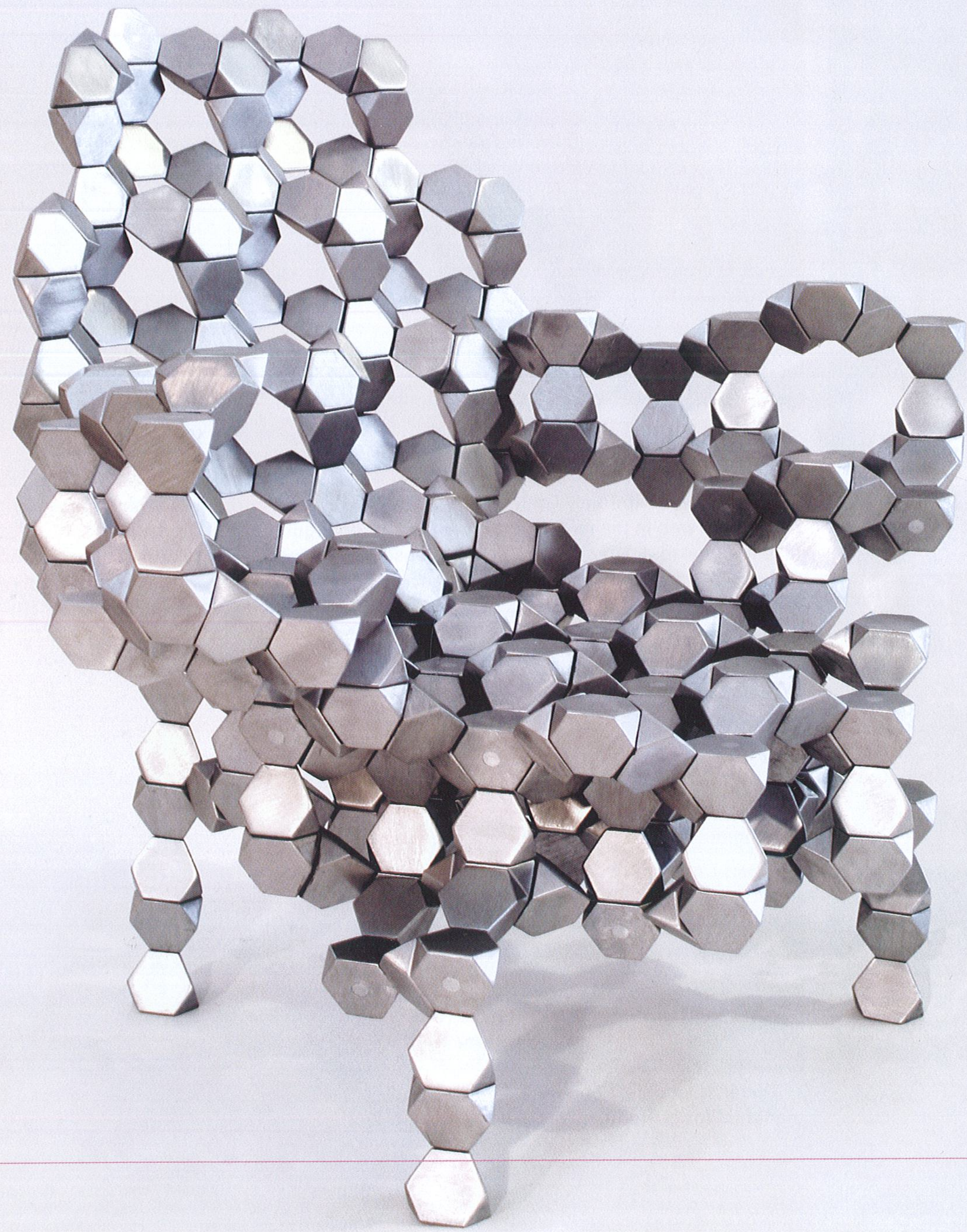
Beyond the regular repetitions demonstrated here, computational machines (i.e. computers) can produce form with endless variation. This property is discussed in depth in the Parameterize chapter.

- What are good examples of modularity? (typefaces, software, background images, construction, ikea furniture)

"Modularity involves the arrangement of one or more elements to produce a multitude of forms." Design something of fixed parts that can be recombined in multiple configurations. Digitally fabricate the forms so that they can be played with. Use any of the new tools we have learned for the design - Processing, Rhino, Grasshopper, or a combination. You can always return to Illustrator as well.

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.

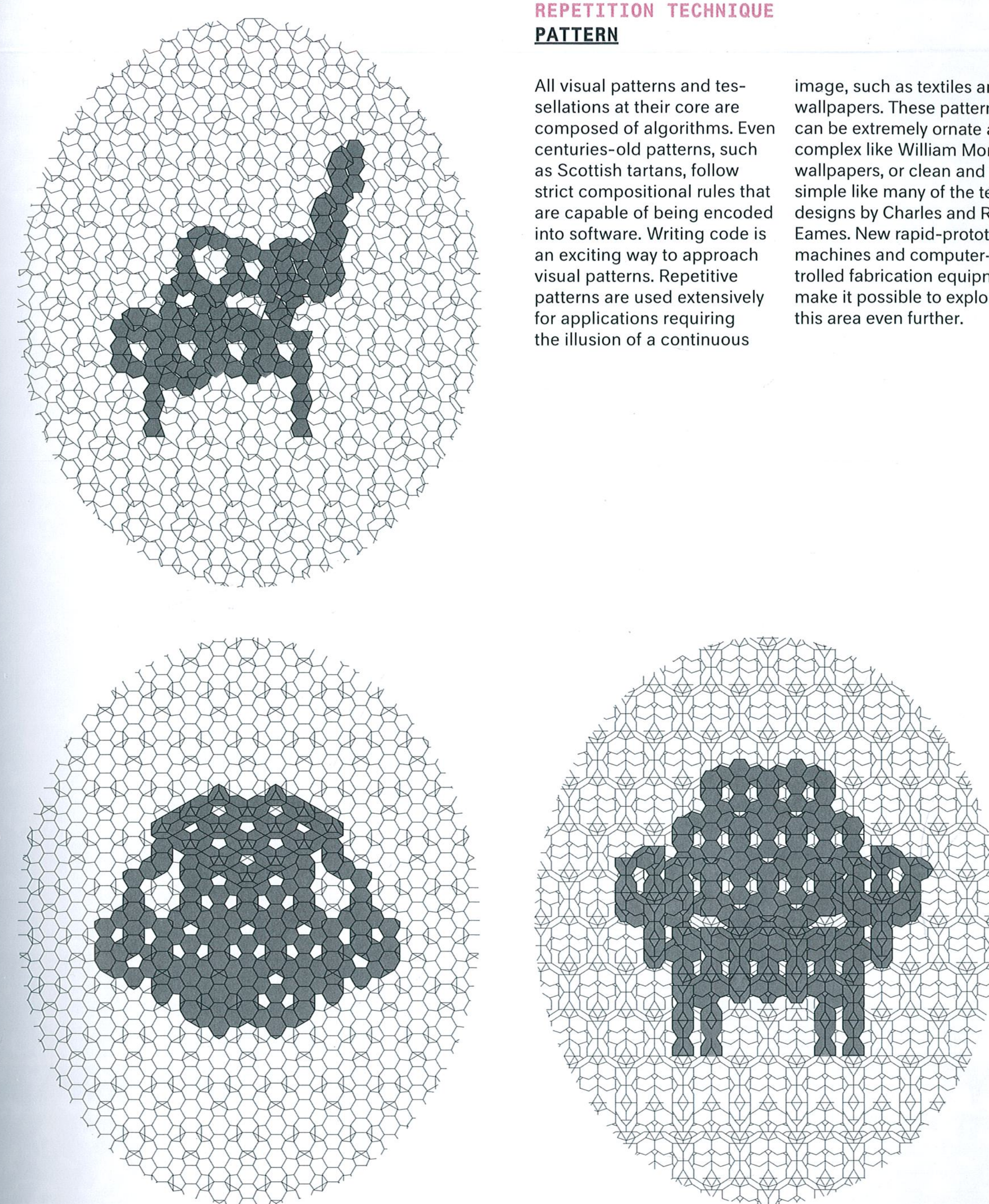
Mario Soup,
by Ben Fry, 2003
This software shows how all of the graphics used in Nintendo's 1985 Super Mario Bros. game are stored within two matrices. In this image, one matrix is shown as red and the other as blue. The colors used in the game are applied while the game is running.



REPETITION TECHNIQUE

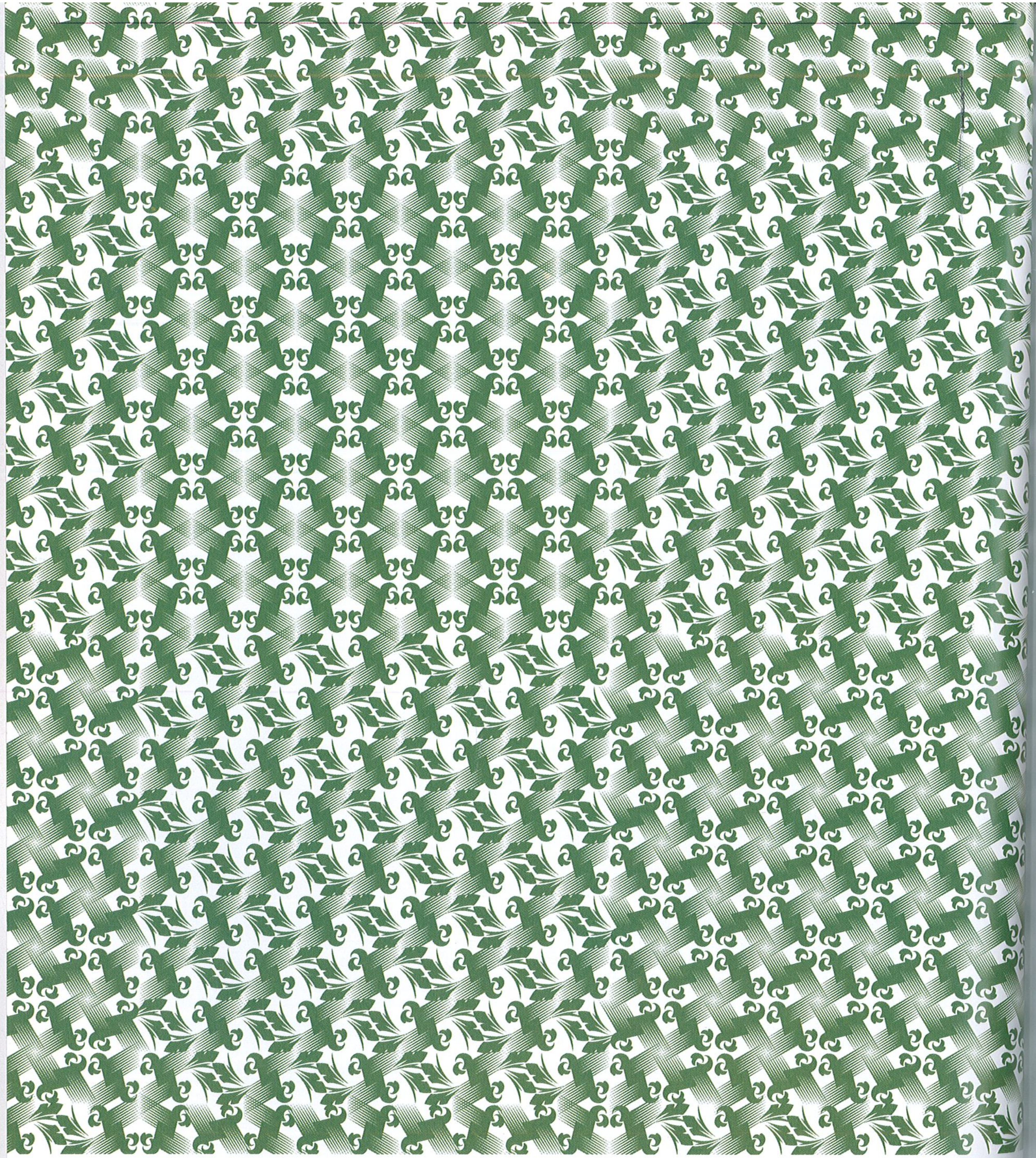
PATTERN

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.

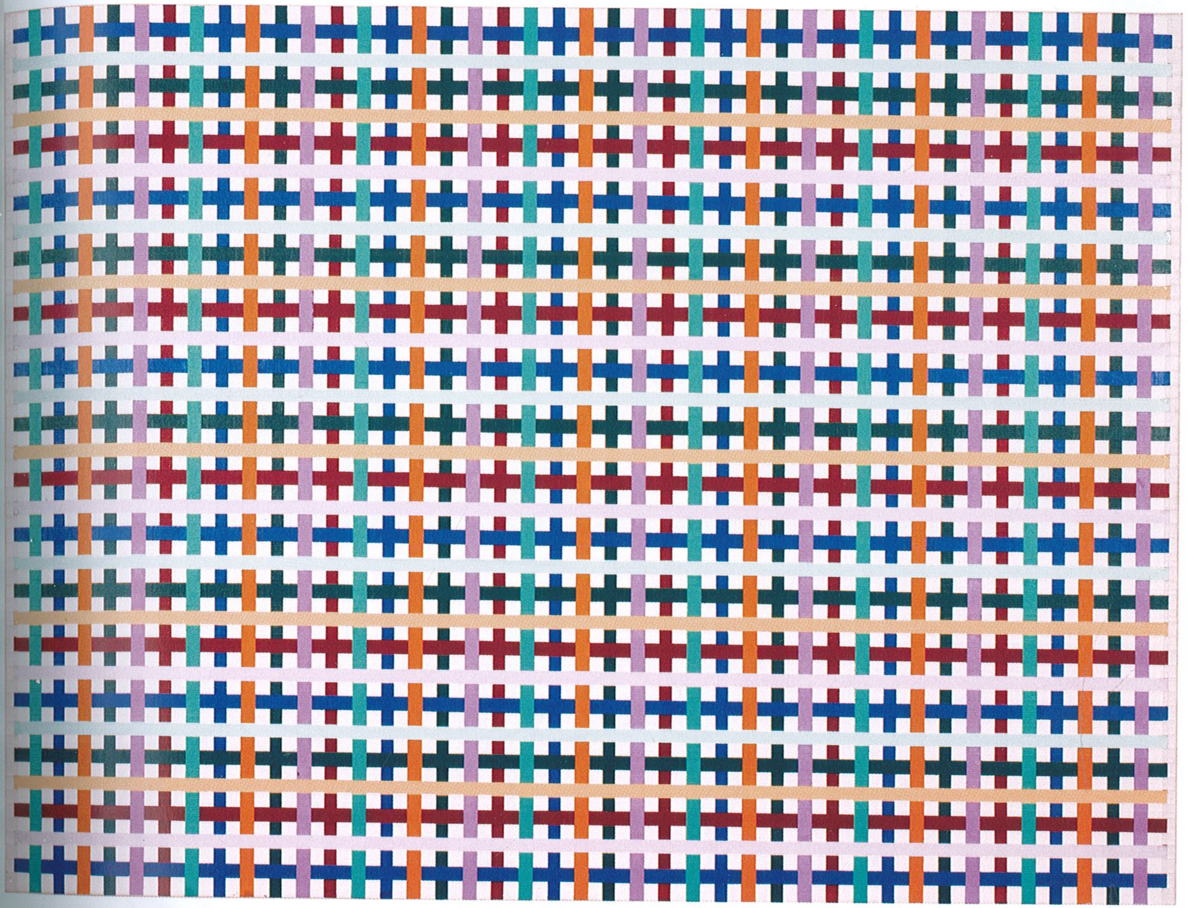


REPEAT

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.

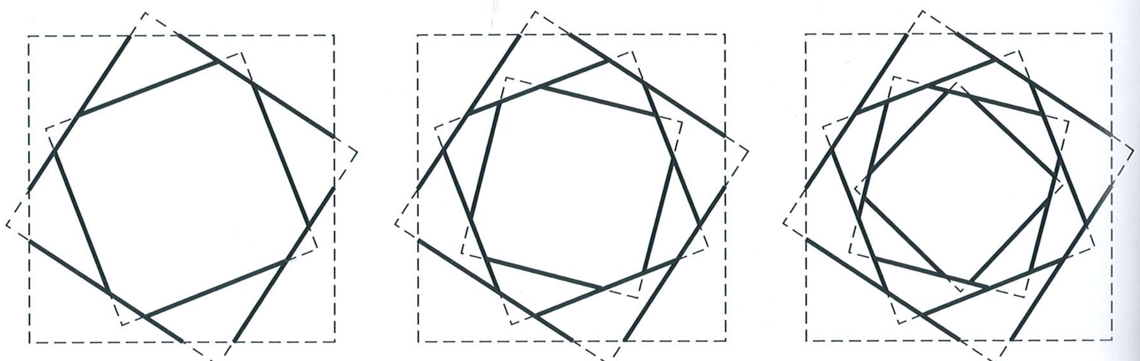


REPEAT

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:

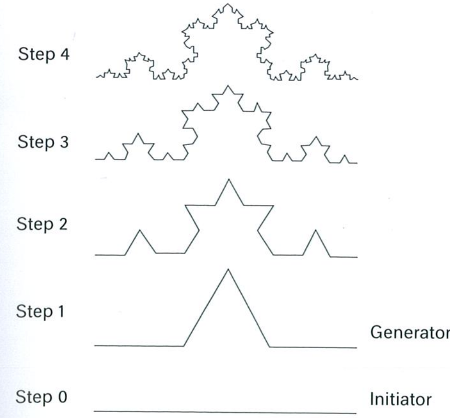
NINE COLORS WERE DIVIDED INTO THREE VALUE GROUPS:
BLUE/GREEN/RED, VIOLET/ORANGE/TURQUOISE, AND LIGHT ORANGE/LIGHT VIOLET/LIGHT BLUE. RED WAS FIRST. BLUE WAS SECOND. GREEN WAS THIRD.
VIOLET, ORANGE, AND TURQOISE WERE ARRANGED VERTICALLY.
LIGHT ORANGE, LIGHT VIOLET, AND LIGHT BLUE WERE ARRANGED HORIZONTALLY.



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



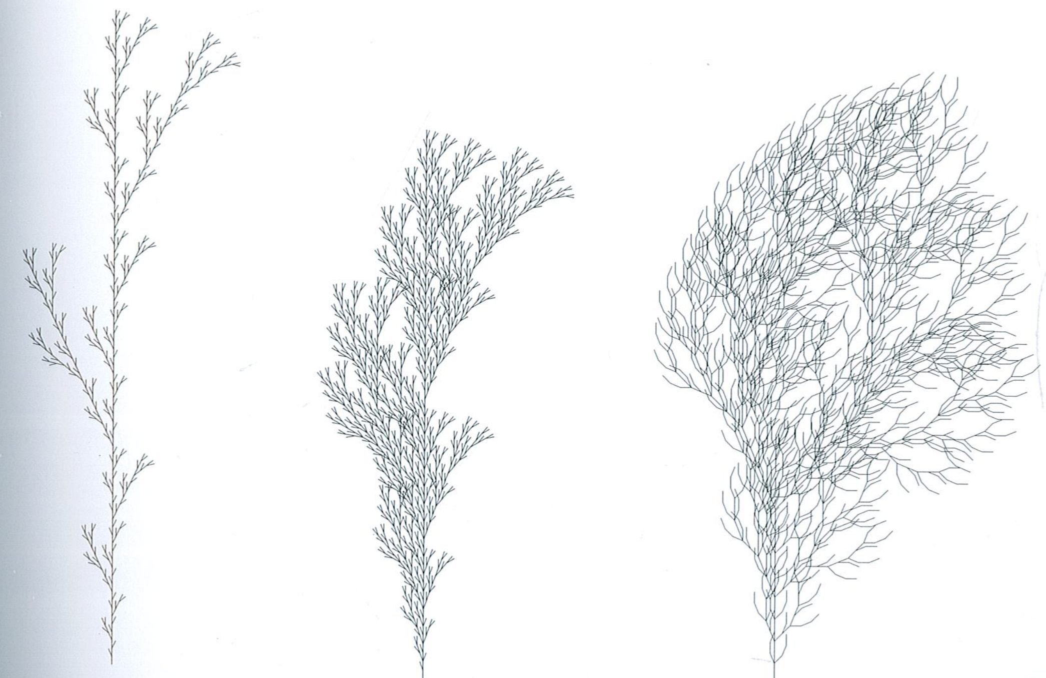
The technique of recursion is an extremely powerful tool for generating form. Using a broad definition, recursion is a process of repeating objects in a self-similar way. A fern leaf is an example of a recursive form; each leaf is composed of a series of smaller and smaller leaves. A joke about the definition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



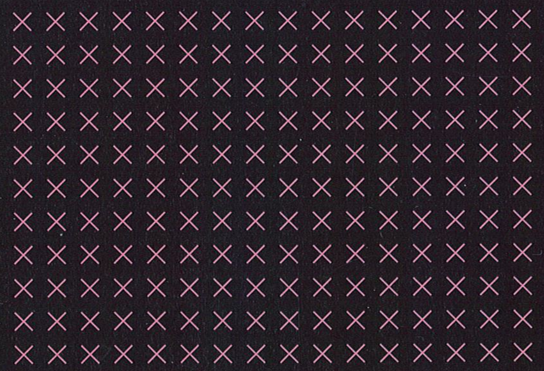
F → F[+F]F[-F]F

F → F[+F]F[-F][F]

F → FF-[-F+F+F]+[+F-F-F]

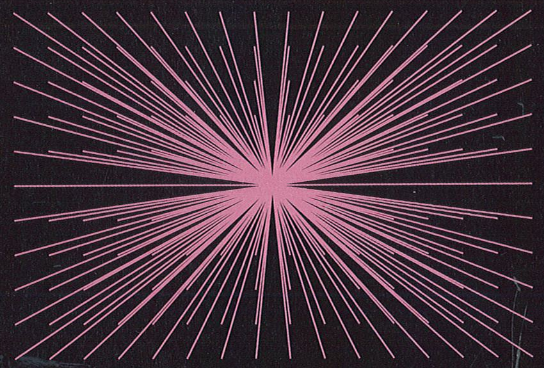
L-Systems, first introduced by Aristid Lindenmayer, 1968 Lindenmayer systems (L-Systems) are an elegant way to simulate plant forms. A starting pattern is replaced according to a set of rules, and it is then transformed again and again.

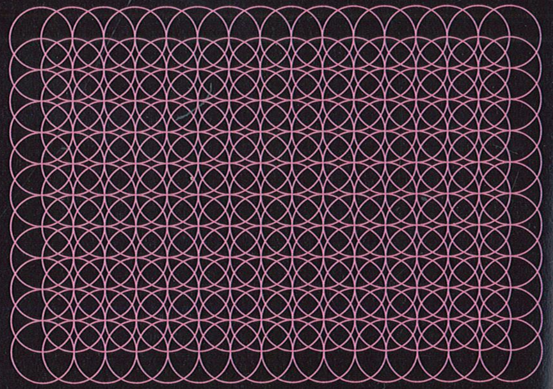
CODE EXAMPLES
EMBEDDED ITERATION

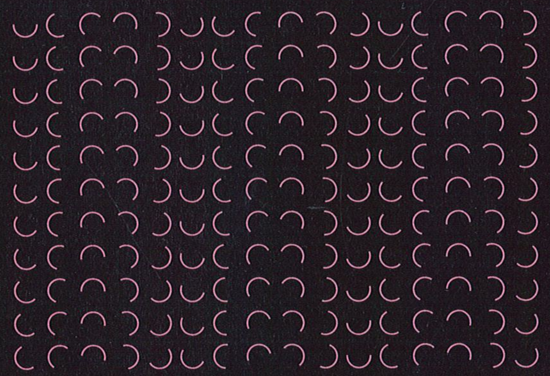


All programming languages can repeat an action, such as drawing the same shape over and over. When one repetition sequence is embedded within another, the effect multiplies. For example, if the action of drawing five lines is repeated ten times, fifty lines are drawn. This simple technique can be used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

Each of these images was generated from the same grid 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 differences between theses images.

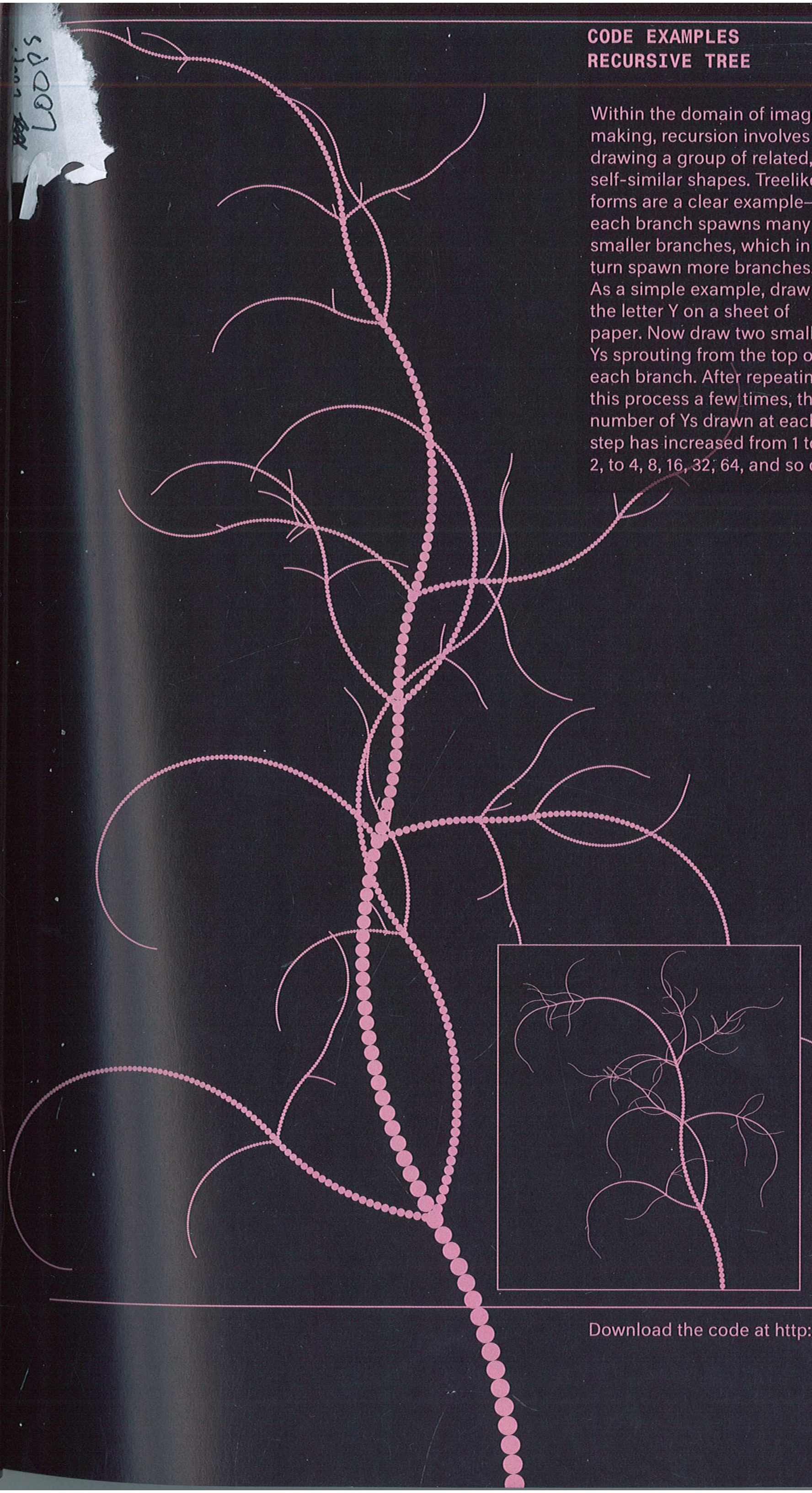






Loops

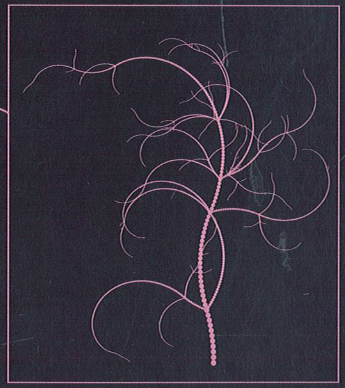
CODE EXAMPLES
RECURSIVE TREE



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