computational expressionism

A STUDY OF DRAWING WITH COMPUTATION.

by **Joanna Maria Berzowska**
BA Pure Mathematics, McGill University, Canada, June 1995
BFA Design Art, Concordia University, Canada, June 1995

Submitted to the Program in Media Arts and Sciences, School of Architecture and Planning, in partial fulfillment of the requirements for the degree of **Master of Science**
at the **Massachusetts Institute of Technology** February 1999

Author **Joanna Maria Berzowska**
Program in Media Arts and Sciences
October 27, 1998

Certified by **Walter Bender**
Senior Research Scientist, Program in Media Arts and Sciences
Thesis Supervisor

Accepted by **Stephen A. Benton**
Chair, Departmental Committee on Graduate Students
Program in Media Arts and Sciences

COMPUTATIONAL EXPRESSIONISM:
A STUDY OF DRAWING WITH COMPUTATION.

by **Joanna Maria Berzowska**

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning,
on October 27, 1998
in partial fulfillment of the requirements for the degree of Master of Science

abstract

This thesis presents computational expressionism, an exploration of drawing using a computer that redefines the concepts of line and composition for the digital medium. It examines the artistic process involved in computational drawing, addressing the issues of skill, algorithmic style, authorship, re-appropriation, interactivity, dynamism, and the creative/evaluative process.

The computational line augments the traditional concept of line making as a direct deposit or a scratching on a surface. Digital representation is based on computation; appearance is procedurally determined. The computational line embodies not only an algorithmic construction, but also dynamic and interactive behavior. A computer allows us to construct drawing instruments that take advantage of the dynamism, interactivity, behavioral elements and other features of a programming environment.

Drawing becomes a two-fold process, at two distinct levels of interaction with the computer. The artist has to program the appearance and behavior of lines and subsequently draw with these lines by dragging a mouse or gesturing with some other input device. The compositions incorporate the beauty of computation with the creative impetus of the hand, whose apparent mistakes, hesitations and inspirations form a complex and critical component of visual expression.

Thesis Supervisor **Walter Bender**
Senior Research Scientist, Program in Media Arts and Sciences
Work supported by the News in the Future Consortium and an IBM fellowship.

computational expressionism

Joanna Maria Berzowska

Thesis Reader **John Maeda**
Assistant Professor of Design and Computation
MIT Media Laboratory

Thesis Reader **Scott Snibbe**
Member Research Staff
Interval Research Corporation

acknowledgments

Heartfelt thanks to Anna and Tomasz for holding my hand and pulling and pushing me through the world these past 25 years. They have taught me about language, culture, desire, despair, courage and determination. They have taught me most of what I know and feel, and I will be eternally grateful for their love. I also want to thank my little sister Justyna for being a great roaming companion.

A tearful thank you to the late Gert-Jan Zwart, who taught me great new things about life and, so very sadly, about death.

Special thanks to my advisor Walter Bender, whose insight, wit and enthusiasm shape the Media Lab in good ways. Additional special thanks to my thesis committee, John Maeda and Scott Snibbe, whose art and thought have and will continue to inspire my work.

Particular thanks to my unofficial thesis readers: Mike Best, Jack Driscoll, Nelson Minar, Arjan Schutte and Paul Yarin.

The Media Laboratory has provided the first social environment in my life where I have felt at ease. The people I interact with on a daily basis betray a lovely mixture of intellect and insanity. Among all these lovely people, there are a few whose friendship was indispensable and invaluable in the task at hand: Mike Best, Sawad Brooks, Daniel Dreilinger, Nelson Minar, Arjan Schutte, Laura Teodosio, John Underkoffler, Alex Westner and Paul Yarin. I thank them all from the very bottom of my thumping heart.

Tremendous thanks to Jimmy Wondrasek for his sincere love and belief in me. He is unquestionably one of the pivotal people in my life. Equally tremendous thanks to Gaia Marsden for being such a close and loving presence for almost ten years now. I also want to thank David Roselli for the inspiration he brought into my life for many years, Raz Schionning for showing me how to use a computer and for giving me direction and Meg Wilson, one of the most supportive and generous souls I have recently met.

contents

introduction

Computers have brought about many changes to facilitate and encourage a breadth of personal expression. Text editing, image editing, layout and publishing software allow the production of documents and other artifacts more quickly, and with an increased degree of control over the process. The ability to save multiple copies, at various levels of completion, permits a more thorough and more efficient execution. The increasingly universal access to information and to the dissemination of information through personal printers and the World Wide Web have greatly simplified the process of receiving, as well as expressing and publicizing, opinions and feelings.

Sketch made in the margin of a notebook (not the margin of the Word document) by Joanna Berzowska

Computers and computer culture continue their assiduous, penetrating and pervasive infiltration of our homes and lives. They are quickly becoming much more than tools for processing data. They have already become, in fact, for more and more of us, a primary mode of engaging the world. [SMI96] They have become a primary outlet for expression, for the building of identity and for personal, emotive interaction with others.

Computer developers are working hard to spawn products that purport to be conduits for creativity. New forms of computer art such as hyper-linked fiction, digital image manipulation, interactive graphics and interactive environments are interesting and significant artistic developments. It is more difficult, however, to find a viable computational outlet for drawing, painting, graffiti, for a true individuality of gestural, expressive style.

The very qualities that define computers as great tools for publishing and communication uses can be an impediment when we consider various implementations of drawing and painting programs. When we wish to sketch, doodle or gesture, we usually turn to different tools: pens, crayons and brushes. This is not because computational environments do not engage the user in expressive ways, do not convey the richness of emotion and aesthetic awareness that some traditional media offer. Simulating the traditional media with computational analogues is not the answer. The answer lies in understanding and developing the innate expressive potential of computation.

This thesis is a step in that direction. It investigates and develops some aspects of the natural expressive language of computers, a language that allows individual style, computational sketching and doodling, and a feeling of having achieved a level of skill and intimate understanding of the medium. The work presented moves away from pre-programmed art tools (which allow a level of expression akin to collage) and allows the artist to program a multitude of personalized solutions. In the process, it reconsiders and reevaluates the skills of composition, line making and dynamic, interactive forms, and approaches them at a rudimentary level within this new medium.

MOTIVATION

The motivation for this thesis is a desire to increase and improve the extent of possibilities for two-dimensional visual expression, in particular for gestural expression such as drawing. It is a desire to define new outlets, new forms and methods for creativity using a computer. The thesis strives to better understand computation as a raw art material and to develop a vocabulary that will allow artists to cultivate a closer relationship between expressive appetite and artistic product in the computational sphere.

The current state of visual digital art is greatly determined by the tools available for its production. Digital drawing tools are still modeled, to a great extent, on previous media and technologies. They are often inadequate in fulfilling the need for expression that traditional media can fulfill, and yet they hope to replace the use of paint, pastels and ink as our medium and material of choice.

Most commercially available software for visual arts[1] is based on one of two models: image manipulation techniques, and the replication of the visual characteristics of preceding art forms and styles. The former is an interesting beginning, but is limited by the fact that image filters provided by software packages result in a trite look, easily achieved and consistent between users. The latter revolves around direct metaphors of Van Gogh and Monet, pencils and erasers, paint buckets and spray brushes. This approach is limited by the fact that the computer cannot hope to compete with the real thing, the metaphors are restricted, and they restrict the kind of expression we can produce.

In addition, current computer interactions and interfaces are stereotyped as impersonal and lacking a certain intuitive, physical or familiar quality. The tradition or myth of engineers who design software while disconnected from anthropological concerns has spawned the extensive field of computer-human interaction design as separate from computer or software engineering.

Artists and programmers both need to better understand computation as an art medium so as to produce art pieces and tools that take full advantage of some of the characteristics of computation. Interactivity, dynamism and emergent behavior resulting from interaction among computational objects are only some of the fascinating aspects of this medium. A better understanding will give rise to a generation of digital art pieces, tools and materials that bring the same levels of emotional involvement and creative intensity in the computational realm that a dripping paintbrush, a thick chunk of charcoal or a textured piece of wood bring in the physical.

The focus should move away from duplicating the methods and materials we know from traditional media, and develop a different perspective on visual thinking. The concept of drawing with computation essentially means to program what sorts of shapes the linear motion of the hand will produce. For an artist to draw computationally means that the artist defines the patterns and colors that the brush produces, defines the behavior of the brush, provides its computational attributes, using algorithms and design principles. John Maeda[2] has said that the designer must be an engineer and the engineer

---

[1] See Appendix B for a list of currently available software for visual arts.
[2] See the sub-section about John Maeda and the ACG in the Background section

must be a designer. As more people become comfortable with programming, these distinctions will disappear, and artists will have less need of an engineer at their side in order to bring their ideas to life. Intimate knowledge and understanding of the materials and methods are an inherent part of any category of art making. The degree of understanding and command of the medium is directly proportional to the degree of skill and expressive breadth.

To achieve an individual algorithmic style, the artist must customize the software used to create visual work. The artist must write code and eventually will create an aesthetic that is unique. As an artist's algorithmic art matures it will achieve aesthetic qualities that are proper to the algorithmic style. This unique aesthetic or style does not mean that the algorithm will mimic work traced by the artist's hand in traditional media. Although it is just as individual, it is a separate form of thinking about visual representation.

FOCUS

The focus of this thesis is to introduce the concept of computational drawing, a method by which visual artists can use computation as an interactive and algorithmic tool to generate two-dimensional compositions of line, shape and color. Computation is the art medium, the iterative process of drawing is the artform, the still drawings, printed on a piece of paper or displayed on a monitor are the end product.

Visual artists historically embrace new media and technologies, and strive to experiment and express themselves within the new criteria that these impose. As a result, they have been compelled to examine and adapt existing definitions of art. Computation has opened a large world of expressive potential, which is only starting to be explored. Among other things, computation has made us realize that the idea of drawing can be expanded to signify much more than the traditional tracing of static lines on a surface.

Algorithmic appearance, dynamic change and behavior are integrated into the traditional definition of a line to define a computational line. Algorithmic appearance implies a mapping of variable complexity between the gesture that is made through the input device and the visual representation on the display. Dynamic change alludes to marks that undergo some transformation over time, whether a shift in color, position, shape, or other physical attribute. Behavior means that the lines respond to each other when present on the same canvas. They can sense the presence, proximity, intersection (topology) of other lines, which in turn affect their dynamic behavior.

The objective of the work is to draw static drawings with computational lines. The act of drawing is a dynamic, interactive process of evaluating and reassessing choices, however the focus is not on interaction but on the creation of the drawing. The work centers on the concept of line making. Drawing, doodling, graffiti, sketching, the shaping of linear forms with movements of the hands are the central point of interest. The interaction is fascinating in itself, but only insofar as it embodies the process through which a drawing is generated. The process is really one of drawing, not of interactive performance.

PRODUCT

In addition to a discussion of computational drawing, the thesis includes a series of Java applets, artifacts that represent the concepts discussed. Each applet is a drawing application that presents one or more of the computational lines created by the author. Some illustrate the progression of one algorithm; others offer an array of various computational lines with widely ranging appearance and behavior. Each one allows a viewer to examine some aspect of computational drawing and computational lines, through a process of altering them and drawing with them. The applets are visualized as still images throughout the document, illustrating stages in the process of creating a drawing. Some still images represent finished drawings by the author, the end product of the computational drawing process. The development of the work is described in the discussion section.

The Java classes used to create the work are available in an annotated package. The code is freely distributed. Other artists are invited to experiment with available lines and to create their own by changing the present algorithms. There is no custom programming tool included with the code. Maeda, in *Design by Numbers*, provides a structured environment where code is compiled and results are shown alongside each other. The editor is shown to the right of the display canvas. This allows designers to see the results of their code immediately. [MAE98] Since the present work is more concerned with drawing, it is beneficial to keep an element of distance between the code and the drawing. It is frustrating, clearly, for drawing to be such a bipartite process. One advantage is that this requires the artist to better understand the algorithms, instead of using a trial and error method of designing the computational lines.

All the code is available from http://www.media.mit.edu/~joey/demo/ or by contacting the author.

STRUCTURE

The Background section provides a motivational framework for Computational Expressionism. It describes the context of the work, introduces the history and significance of the term expressionism, the relevance of line and drawing, as well as the inspiration of John Maeda and his ideas on computational design.

The main section presents the body of work called Computational Expressionism, through an overview of projects created, and a discussion of Drawing with Computation, Algorithm, Dynamism, Artistic Process, Interface and Emergence as pivotal issues and questions. It also situates the present work in the context of interactive art, kinetic art, algorithmic art, and evolving (emergent/genetic) art.

The section entitled Drawing with Computation serves as an introduction. It provides an outline of the author's definitions for the concept of computational lines and computational drawing. The third subsection discusses the creative process involved in this work, and introduces some examples produced by the author.

Three classification schemes permeate the organization of the text.

1. The qualities of computation that inform the present definition of computational drawing are algorithm, dynamism, interactivity and

emergence. The first two make up two sections of this document, which also introduce the first two attributes of a computational line, appearance and dynamism. Interactivity is discussed in the Artistic Process section. Emergence is introduced in its own section.

2. The work produced by the author for this thesis is grouped according to the three defined attributes of the computational line: appearance, dynamism (change over time) and behavior. The three attributes are discussed and illustrated as best as possible with still images. Appearance is discussed in the Algorithm section. Another discussion of appearance is described in Manipulating the Canvas, where mappings of hand movement to graphical representation are presented in increasing levels of abstraction. Dynamic lines are discussed in the Dynamism section, and behavior, which is given a more rudimentary treatment than the other two, appears in the Emergence section.

3. Three groups of background work are used to illustrate the three attributes. The subsections on algorithmic art, kinetic art, and evolving (emergent/genetic) art can be found successively under What is Skill Now? and Algorithmic Artists in the Algorithm section, Background in the Dynamism section and Definition in the Emergence section.

The Discussion section presents some questions relating to the evolution of the work, a description of evaluation, future developments and a conclusion. The Future Work subsection explains future extensions and interests in genetic algorithms, interfaces that use drawing as a method for human-computer interaction, and developing special programming languages for artists.

Appendix A presents a selection of work created and gives some of the technical details and Java code. Appendix B is an overview of available software for digital drawing.

background

CONTEXT

This work was produced within the Electronic Publishing Group, in the News in the Future (NiF) consortium at the MIT Media Laboratory, under the direction of Walter Bender, the principal investigator of NiF.

Bender's varied interests include a desire to "build upon the interactive styles associated with existing media and extend them into domains where a computer is incorporated into the interaction." [BEND97] He professes a need for adaptive, dynamic, context-sensitive representations of media objects, illustrated by applications such as customizable news browsers that are aware of changing environments.

He helped spawn the present work by providing the inspiration of the Incredible Machine[1]. This is a game that hinges on the construction of virtual faux-physics environments out of computational elements that have specific form and behavior and interact in the workspace to produce working systems. The interactive parts provide an explicit motion, shape and functionality and act/react with one another to produce emergent worlds whose behavior is both complex and interesting. The pieces are used to experiment and to solve puzzles.

The game causes players to think of physics in a non-physical context. The physics model is stilted, simplified and embellished with elements that only the magic of computation allows. The elements each have a function associated with them and can be composed in ways that produce an engaging result. Bender proposed that a similar model would be interesting for drawing tools.

EXPRESSIONISM

The word expressionism has been applied to various art movements throughout history, all sharing a common preoccupation with the emotive, passionate qualities of material, subject and method, a desire to emphasize expression over realistic representation. [LYN97]

Drawing by Egon Schiele

The original movement, exemplified by artists such as Oskar Kokoschka, Georges Rouault and Egon Schiele, flourished in Europe in the late 19th and early 20th century. It is characterized by a desire to depict more than objective reality, and focus on subjective emotions and responses that subjects evoke. [BRI98] This is accomplished through distortion, exaggeration, primitivism and fantasy, as well as a vivid, violent, or dynamic application of formal elements.

[1] Marketed by Sierra

The German Expressionists developed a style notable for its harshness, boldness, and visual intensity. They used rough, distorted lines; crude brushwork and shrill colors to depict urban street scenes and other contemporary subjects in crowded, agitated compositions notable for their instability and their emotionally charged atmosphere. Many of their works express frustration, anxiety, disgust, discontent, violence, and generally a sort of frenetic intensity of feeling in response to the ugliness, the crude banality, and the possibilities and contradictions that they discerned in modern life. [BRI98] [LYN97]

Abstract Expressionism refers to a body of American (centered on New York) painting executed in the 1940s and 50s. Work from this period, exemplified by artists such as Jackson Pollock, Willem de Kooning and Mark Rothko, was non-representational and emphasized spontaneous, personal expression. This period of art history was also characterized by a great amount of experimentation with materials and technique. [HAR97] [BRI98]

Painting by Mark Rothko

Painting by Willem de Kooning

Artists were fascinated with the intrinsic expressive qualities of the materials, leaving behind traditional elements of composition and using the material naturally and intuitively to evoke sensual, violent, dynamic, mysterious, and lyrical element. The paint and tools were manipulated to express the force of the "creative unconscious" in art, akin to surrealist automatism. [BRI98]

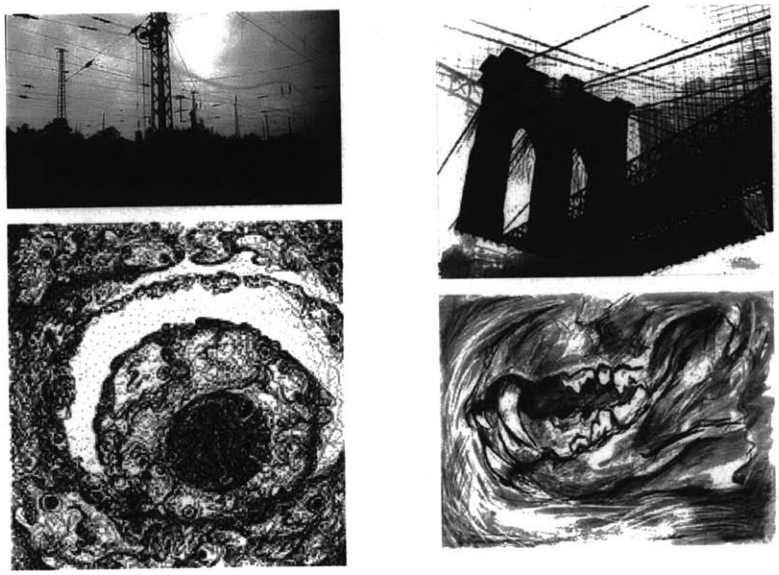
The preoccupation with and merging of material and subject is best illustrated by the work of Pollock, who used paint drippings as both style and content. Arthur Danto uses the term "the end-of-art condition" to describe "the fact that style becomes subject matter, and hence is something shown rather than used". [DAN97] The objective of Computational Expressionism sustains the spirit of these artists, by seeking out the natural expressive language of computers, to spawn an eloquent freedom, a vernacular of individual style and a level of visceral understanding of the medium.

LINE AND DRAWING

The line is crucial in this work, because computational expressionism is primarily concerned with the gestures that are made, that are reflected in the on-screen representation. Traditionally, the process of drawing involves hand gestures, dragging a soft or hard tool over a surface, to leave a deposit or an abrasion, a tracing. A drawing is the end product of a successive effort applied directly to the medium with a tool, producing linear additive or subtractive forms and silhouettes that follow and represent the hand's motion. Drawing is the art or technique of producing, with the hand, images on a surface, by making marks

with ink, graphite, chalk, charcoal, or crayon. It is a two-dimensional composition, relying aesthetically on gesture and line, an abstraction, lesser or greater, of spatial objects in the world to lines drawn on a plane. Drawing is distinguished from painting by an emphasis on form and shape, rather than mass and color. It is contrasted with other graphic arts in that a direct relationship exists between production and result, between the hand and the eye.

The line is an essential ingredient of drawing, although it can just as well be used to represent areas of tone or color. The line is an essentially abstract figure, absent in nature and appearing only as a border setting of bodies, colors, or planes. In art, it has been the vehicle of a representational, more or less illusionist, rendition of objects. Only in very recent times has the line been conceived of as an autonomous element of form, independent of an object to be represented. [BRI98]



Top left photo by Sawad Brooks.
The drawings are made with ink and chalk, ,by Joanna Berzowska

Lines, depending on the medium and tool that shapes them, can have a large range of physical and dynamic qualities. They can be thick, regular, immutable, erased. The makeup of the materials used and the temperament of the gesture (speed, direction, and pressure) determine the visual nature of the lines. A combination of line shapes, even without reference to the medium used, provides the artist with a profusion of subjective form for the expression both of general stylistic traits and of personal temperament.

JOHN MAEDA AND THE ACG

John Maeda has been concerned with issues of computational aesthetics for many years, most recently as director of the Aesthetics & Computation Group (ACG) at the MIT Media Laboratory. With his students, he is performing experiments in information navigation, models of digital performance, tangible interfaces, concrete programming, and digital expression. He leads the study of "expressive aspects of computer-human interface from the viewpoint of traditional visual communication design" [1]. He is working towards developing a

[1] See http://acg.media.mit.edu/research/research.html

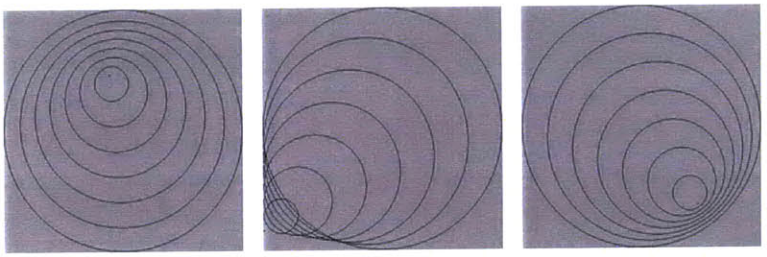
new understanding of the field of computational aesthetics by forming a new breed of worker who simultaneously performs the roles of artist and of digital engineer. [MAE97]

Maeda is striving to define and realize the distinctive potential of computation as a powerful aesthetic instrument. He claims this potential is often misused in popular work, due to the ubiquity of simplistic and unimaginative digital design tools. Current tools commodify the process of creating visual forms and attract underskilled designers. The issues associated with questions of originality as well as the pervasiveness of the medium compromise the quality of work.

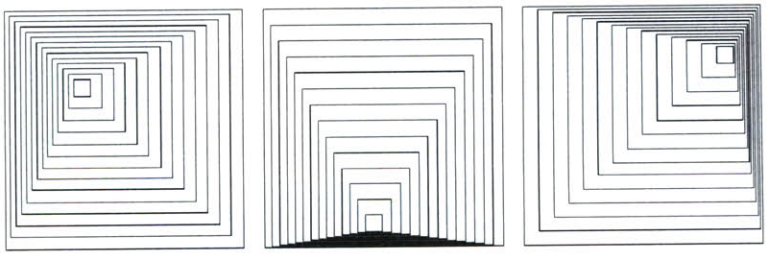
DESIGN BY COMPUTATION

This thesis is inspired and informed by a course Maeda taught at the Media Laboratory in the spring of 1997. The course encouraged the exploration of computation as an expressive medium and emphasized the unique expressive properties of the medium in order to develop a set of extensions to the designer's creative vocabulary. Problem sets for the class resulted in a body of Java applets that illustrated elements of traditional design in an interactive or reactive computational form.

Graphical elements, point, line, balance, rhythm, were programmed and could respond to user input (clicking and dragging the mouse across the canvas or drawing on a tablet) in ways that manipulated these elements of design and exposed aspects of composition to the viewer.



Stills from the Thales6 applet by Joanna Berzowska



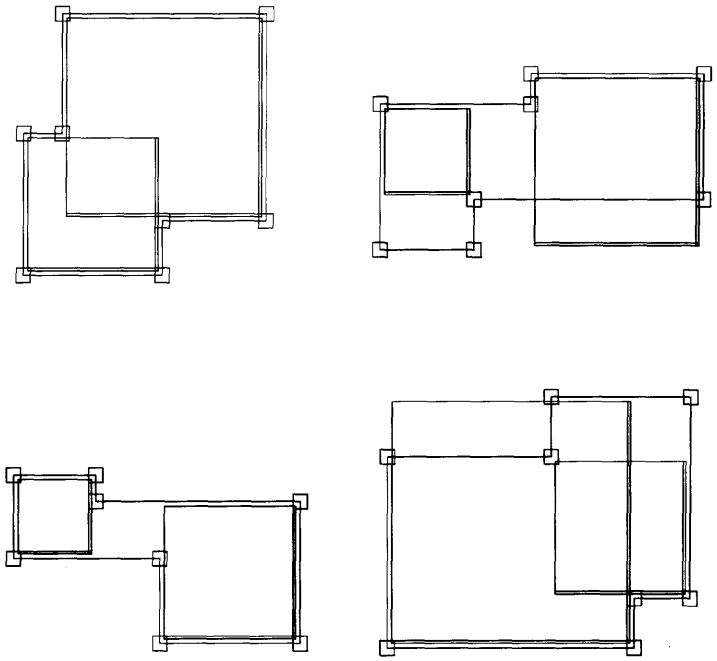
Stills from the Rumba applet by Joanna Berzowska

Thales6 was created to explore and illustrate the element of repetition in composition. The Java program draws a series of concentric circles on the canvas. The image is interactive: the center of the circles follows the movement of the cursor. The constraints of the design are defined, but also invite the viewer to alter parameters of the work in order to give both visual depth and formal complexity to the composition. As designer, the programmer can experiment with shape, distance and behavior of the composition.

In Rumba, the illusion of dimensionality is heightened by decreasing the distance between adjacent shapes, and by constraining the smallest elements within the largest ones. Such an interactive study of an aspect

of design essentially alters the process of sketching. Instead of making several drawings by hand to explore repetition, the process demands a more thorough stage of planning. Sketching is paralleled in the repetitive and iterative process of programming and allows more thorough experimentation with form once the program is completed.

Another element of design is shape, and the following stills are taken from an applet that explored space, in particular form and contour. The problem asked students to investigate union and intersection of squares. When two objects are placed on a canvas, how do they interact, how do we view them, how do we represent them as a single shape? How do their silhouettes merge, intuitively, mathematically, and visually?



Four stills from the Sqr2 applet by Joanna Berzowska

To think about shape, within the realm of computation, one must first define intersection and union. The function for determining the area that is the union of both squares, given that the squares can be moved and scaled at will, is quite complicated, much more complicated, in fact, than determining the same information by hand. Concepts of mathematics can be used, but also can be twisted and reinterpreted.

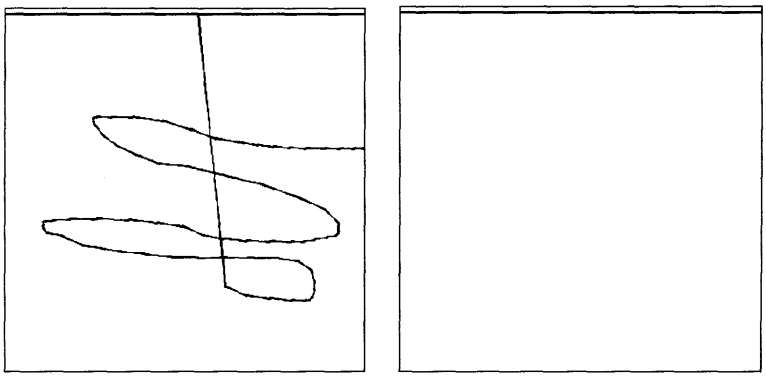
The union of squares, in the example above, is shown in an architectural aesthetic, and encompasses more that the sum of the areas covered by each square. With computation, we can represent union as an extension in space, remaining within the parameters of geometrical union and intersection, but extending the concepts, and the lines, to achieve a visual balance.

The sum of questions posed by Maeda during the course inspired the present inquiry into the nature of computational drawing.

computational

Sherry Turkle has pointed out that the computer differs from other media in that it is both a constructive and a projective medium. [TUR84] In the physical world, you cannot change the laws of nature to fit, for example, naive theories of motion. Computation allows abstractions, exaggerations and playful representations that illustrate intuitive conceptions of physics. It also allows representations that completely disregard the laws of physics, and allows the construction of fantasy worlds. Computation invites us to represent, but also to interpret the world around us, through experimentation and abstraction.

In this example, the line traces the movement of the cursor, which follows the gesture of the hand, holding a mouse or a pen. The tracing is displayed as a digital shadow on the monitor. The gesture is perfectly reflected in the pixels, until, several seconds later, the traced line begins to disappear from the monitor like a woven thread unraveling[1]. Visually, the black bar acts as a magnet or a spring coil that pulls and erases the drawing.



Stills from the Line1 applet by Joanna Berzowska

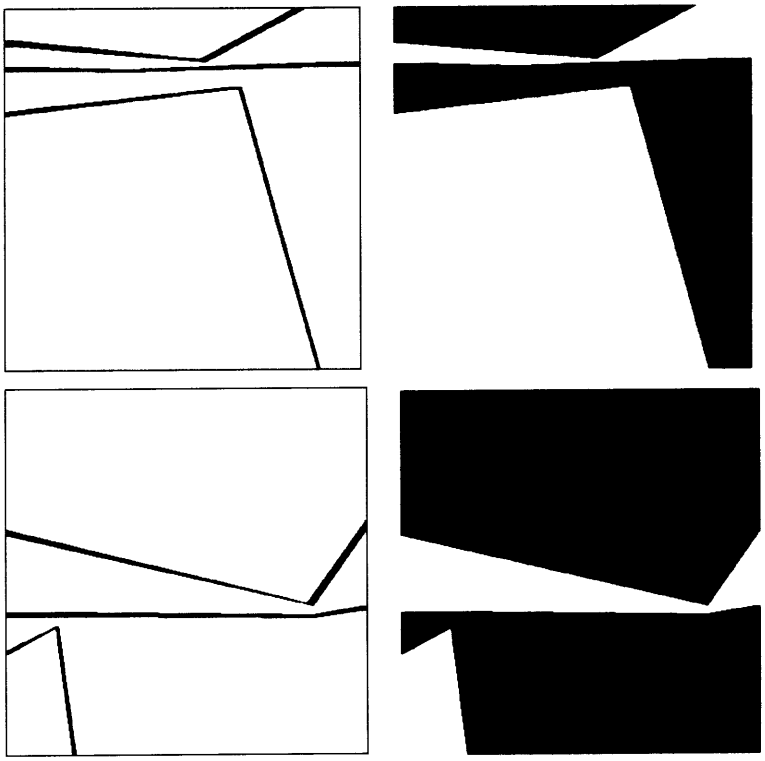
This illustrates the significance, but also the opportunities, provided by the task of drawing with a computer. In this realm, a line can be programmed to have behaviors that are unexpected, lyrical, metaphoric and even (though this serves no obvious purpose) unrelated to the motion made with the hand. The laws of physical motion, the laws of immutability of matter and other physical laws that we have learned from traditional drawing are similarly rendered invalid. We are free to pursue any interpretation of our ideas about drawing, and our ideas of the meaning of gestures.

Computational lines differ from those found in the physical world, whether on a paper surface or stretched out in space, projected onto a wall or implied, the boundary between two planes. They differ insofar as their appearance and behavior does not need to follow any laws of physics. They are not bound by the ballast of physicality, and thus can change form more easily than their physical counterparts. The very elements of composition and design become open to inquiry.

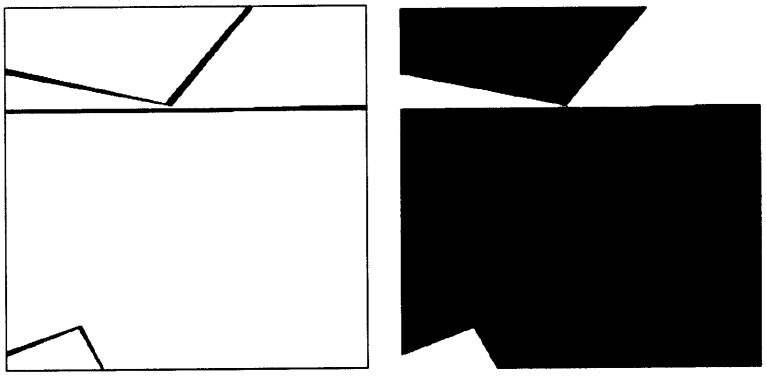
---

[1] A straight line segment is drawn joining the point $(x_n, f(x_n))$, the oldest-drawn point, and point $(x_{(n-3)}, 0)$, in the black bar at the top of the canvas, with latency. The pixel is removed and a line segment is drawn from the next point $(x_{(n+1)}, f(x_{(n+1)}))$ to $(x_{(n-2)}, 0)$.

This set of images is one solution to a problem posed by Maeda, to think of a representation of the transformation of three objects into two. This sort of magic is possible with computers, and a simple click of a mouse can launch a method that will transform a set of pixels into a form completely unlike their original state. In this case, the magic is simply that of thinking of space in terms of line and shape.



Three clicks of the mouse originally determine three points that describe the placement of three lines. Each point is represented as the vertex of a two-segment line. The y-coordinate determines the vertical position, the x-coordinate determines the angle at which the two segments meet. The subsequent click fills it the two areas delineated by the lines, as positive space polygons. The illusion of three objects becoming two is clear and simple.



Stills from the THREE applet by Joanna Berzowska

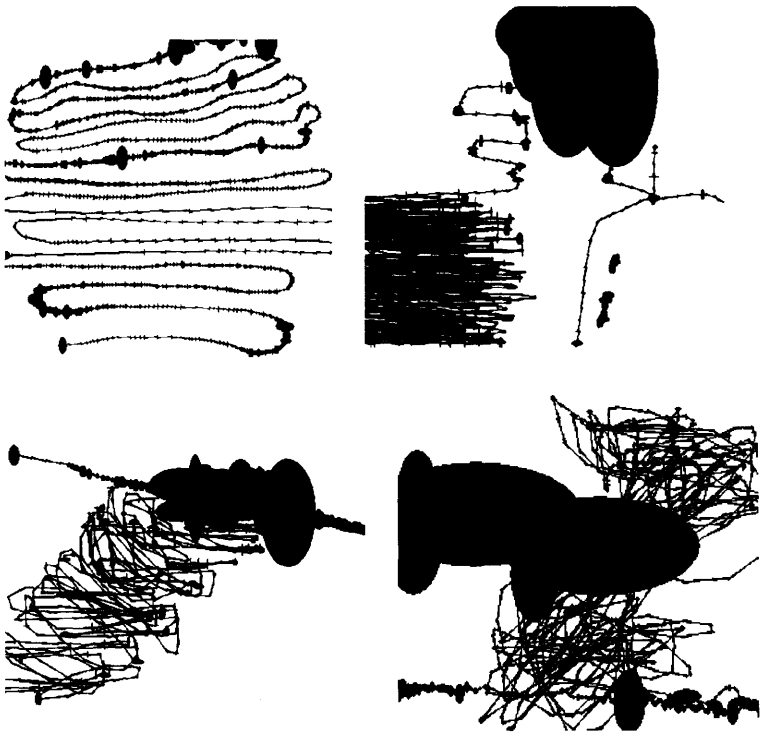
Computation allows a great fluidity of the formal elements of composition that were formerly thought to be static. It forces us to rethink very basic elements of composition and design, instead of attempting to mimic traditionally accepted ones.

DRAWING WITH COMPUTATION

COMPUTATIONAL LINE

The constrained and stylized character of current two-dimensional digital art is directly related to the fact that most artists work with already existing tools. Programmed drawing tools are much more powerful aesthetically than traditional drawing media. A single algorithm can generate whole images or patterns whereas a piece of chalk, no matter how dark, can only trace very elementary shapes. The stylistic nature of computational tools can influence the visual nature of the art to a greater extent. Algorithms are so controlling that we must carefully choose, as well as author, the tools and the methods with which we decide to create and manipulate digital media.

Calligraphy 1 is an early example of a computational line, created to suggest the sort of difficult drawing that can be done with a fountain pen on a napkin. The fibers in the napkin interfere with the smooth motion of the pen. Blotches of ink can appear if the motion is paused. The thickness of the drawn line is determined by the speed with which it is drawn. The direction of the drawing also affects the texture of the line. The line becomes thicker, and generates blotches when the speed decreases considerably. To create shapes, or filled areas of ink, one must slow down the motion of the hand.



Four stills from the Calligraphy I applet by Joanna Berzowska

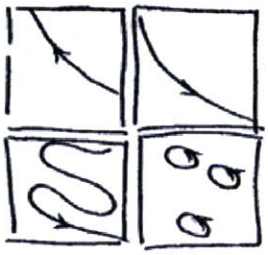
This example shows how a single line tool (from now on referred to as a "computational line") allows one to draw lines of different thickness and to create shapes of various sizes. The size of the shape is a function of the time the line is paused (with the finger on the mouse button), and the shape orientation is determined by the direction of drawing.

The *Calligraphy 1* line is very basic, the representation of the hand gesture is closely mirrored by the algorithm, with minimal embellishments. It is something that will be referred to as a "direct line". It is also a static computational line: it follows the movement of the input device and leaves static remains on the display. As will be discussed in later sections, the computational line can also be portrayed as a pattern of shapes that holds a more abstract relationship with the gesture.

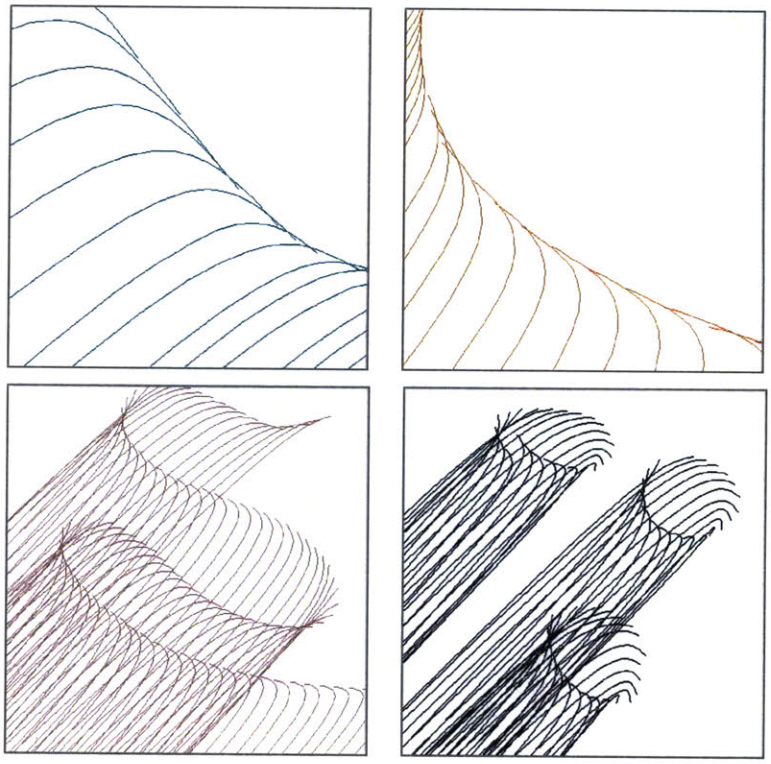
COMPUTATIONAL DRAWING

We "draw" on the computer monitor by dragging a mouse over the table. A computational line produces a visual outcome augmented by the representation of position, speed and direction of drawing. The hand gesture itself is encoded as a vector of point object. Each point object is a Cartesian coordinate pair and a time parameter, which records the exact time of creation. The points the line traverses and their temporal ordering are factors in determining the line's appearance.

The following stills show some simple gestures drawn with the WHEAT line, so named because of its appearance of wheat swaying in the wind. The first drawings illustrate single gestures, so as to highlight the versatility of this computational line. The left panel represents a movement from the bottom right corner to the top left. The right panel is a visualization of the inverse movement. The following two panels illustrate slower, more detailed movements. The spacing between consecutive strokes varies with the speed of drawing. The tilting direction preserves the direction of the gesture, the curved endpoints directly imply the direction of the hand's motion. It is an intuitive, albeit more abstract drawing tool. Complex shapes can be created by a very simple gesture.

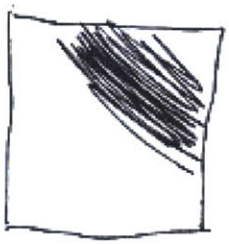


The gestures above generated the WHEAT line compositions on the left.



The next two panels illustrate a different use of the WHEAT line. Instead of creating linear shapes, it is used to create texture and areas of color. The gestures that generated these compositions are repeated movements back and forth, akin to shading with a pencil. The character of the line is evident from the shapes that are achieved.

The shading gestures above generated the WHEAT line composition on the right.



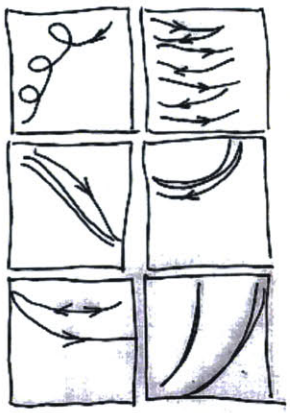
The shaded gestures above generated the WHEAT line composition on the right.

Stills from the WHEAT applet by Joanna Berzowska

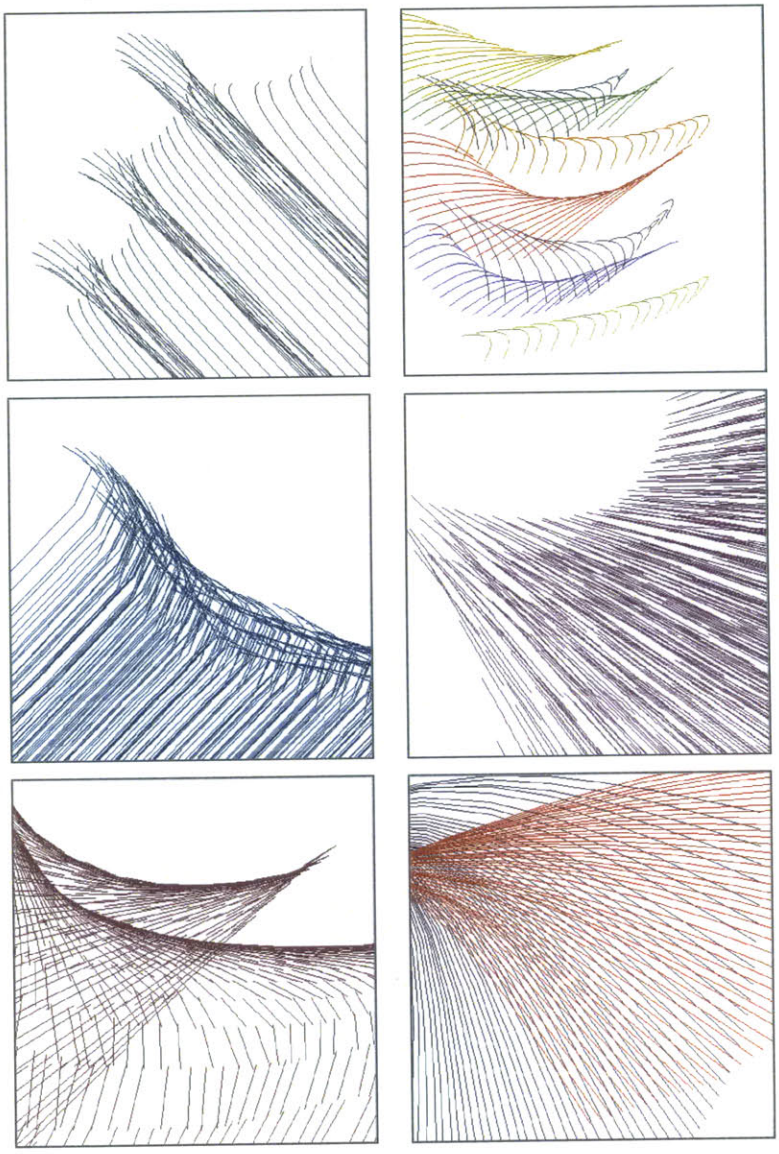
CREATIVE (EVALUATIVE) PROCESS

The following study is an exploration of the WHEAT line. Starting with the computational line illustrated above, which offers a wealth of versatile expressive possibilities, variations in the algorithm are explored to achieve a variety of line tools. Varying the length of the stroke, the amount of latency, the smoothness of the curve, direction, and other characteristics produce a wide range of computational lines.

The process of computational sketching, alluded to in the previous section, consists of making changes in the code, compiling and running the drawing program and making some sketches on the canvas. The drawings executed with the computational line instruct the next iteration of the algorithms. Changes can be saved to catalogue the progression of the computational line. The code evolves, and the algorithms are refined and varied. This iterative bipartite process of programming and drawing produces an individualized algorithmic style for this particular piece.

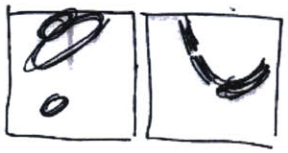


The gestures above generated the compositions on the left.

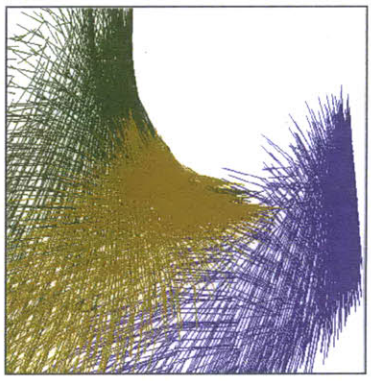


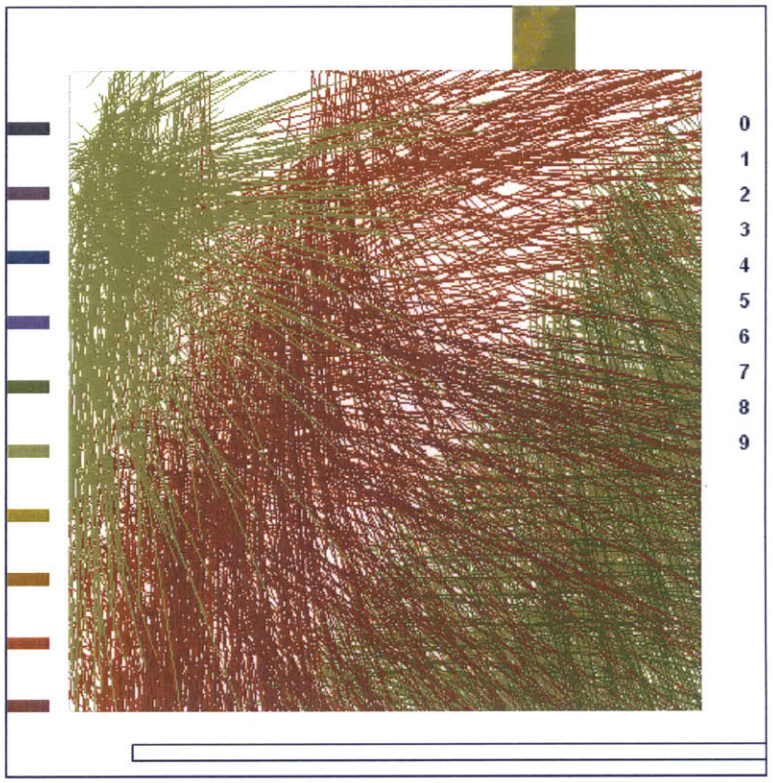
Stills from the WHEAT applet by Joanna Berzowksa

The most successful algorithms are saved and incorporated into a formal applet. A color palette is selected by the artist for this particular computational line, and the resulting drawing tool is used to create more complex compositions, as illustrated in the last two small panels in this series. The final panel shows the simple interface for this particular example, which will be explained in the Interface section.



The gestures above generated the compositions on the right.





Interface of the WHEAT applet by Joanna Berzowska

We will see in later sections that a computational line (representation of a gesture drawn with computational tools) has three attributes.

- The computational line has physical appearance, which can be a set of points joining two endpoints; it can be a shape, a pattern, a representation of a mathematical algorithm, a color.
- The computational line has individual behavior: dynamic properties such as a change of color over time, or movement across the canvas.
- Finally, the computational line has behavior in its interaction with the other lines on the canvas. It can push them away with pseudo-magnetic forces, change their color, or affect their shape.

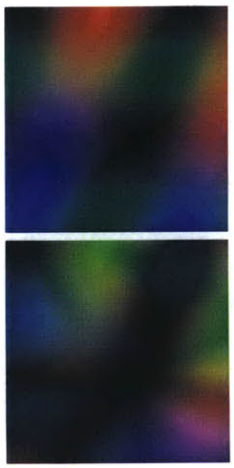
ALGORITHM

> Mathematics is the majestic structure of man to grant him comprehension of the universe. It holds both the absolute and the infinite, the understandable and the forever elusive. It has walls before which one may pace up and down without result; sometimes there is a door: one opens it—enters—one is in another realm, the realm of gods, the room which holds the key to the great systems. These doors are the doors of the miracles. Having gone through one, man is no longer the operative force, but rather it is his contact with the universe. In front of him unfolds and spreads out the fabulous fabric of numbers without end. He is in the country of numbers. He may be a modest man and yet have entered just the same. Let him remain, entranced by so much dazzling, all-pervading light. [LEC80]

DEFINITION

An algorithm is a "process, or set of rules, usually one expressed in algebraic notation, now used esp. in computing, machine translation and linguistics." [OXF98] It is a systematic, finite set of steps, processes or operations that produce some result, such as the answer to a question or the solution of an equation.

Procedural models such as fractals allow the artist to create a high degree of complexity with relatively simple input information. Similarly, using combinations of functions as fundamental as sine and cosine can easily produce engaging compositions of the plane. On the right are pictured some early experiments. The x and y coordinate values of each pixel in the plane are used to generate the color values that are then painted on that pixel. The values obtained from the position of the cursor, either through clicking, or tracing the input device over the virtual canvas, add another parameter to the methods that determine color. Through experimentation, complex and interesting compositions are created from simple primitives of algebra and calculus.



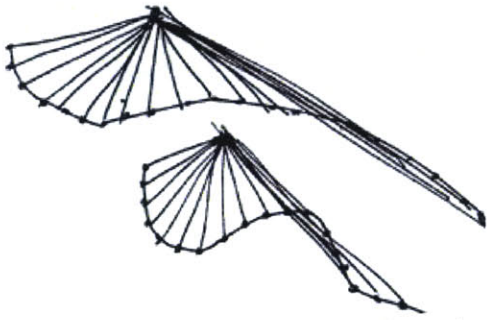
MATHART
by Joanna Berzowska

Algorithms and computer programs are invariably procedural constructs. A common reaction is that a mathematical formula and the graphs of functions cannot be considered art, as they are the products of a formal and directed process. It is true that a drawing program does not simply make art, but artists do not simply make art either, they follow a method. [VER94] What is often called talent or intuition is a complex form of reasoning, so complex, in fact, that we do not know how to describe it in words. Just as a traditional artist follows a learned and well-defined procedure in the course of creating, the algorithmic artist also uses a more rigid procedure of iterative reformulation of algorithms. The one essential element to the process, a developed

artistic procedure, is necessarily unique for each artist and for each work of art. The final composition, particular to each work, embraces more than computation, insofar as it is the product of creative approaches, reference, metaphor, memory, representation and improvisation. [VER94] It is the product of the artist's experience and level of comfort with the medium, and is inspired by a history of sketching in computation and other media.

FIRST ATTRIBUTE: APPEARANCE

Every computational line has a component that is merely its physical appearance. That is the visual mark it leaves on the canvas. If the line is dynamic, its appearance can be thought of as its initial state as it is drawn, or simply its visual representation at any given moment in time. The rudimentary building block of a computational line is the vector of coordinate points that describe the drawn gesture. The appearance that we see represented on the drawing canvas is necessarily mathematically determined and incorporates elements of computation that extend the vector. We can think of the vector of points as a spine, and the computational line representation as ribs, or tentacles that the spine controls. It is important that what we see on the canvas can be directly related to gestures that are made. No matter how complex and evolved it is, the appearance must be governed, must be drawn by the hand.

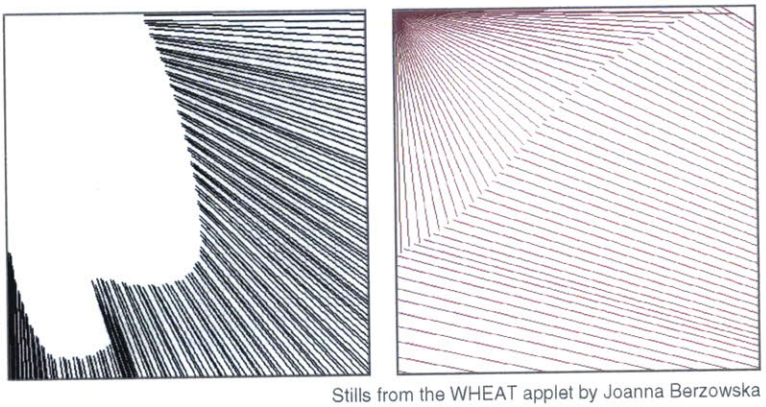


Hand sketch by Joanna Berzowska

These compositions are drawn with a traditional pencil, repeating the same motion several times to achieve a pattern of lines. A program can generate all the ribs, given the spine, and reflect the speed and direction of drawing.

This particular sketch inspired the GRIDS applet. GRIDS started out as a desire to abstract the line itself into a series of endpoints of secondary ribs, lines emanating from a common point. In this case, the line drawn in fact defines the border, the boundary of one or several shapes.

The compositions on the right were generated by the above gestures.



Stills from the WHEAT applet by Joanna Berzowska

The point of convergence of the ribs is determined either during the programming stage, or during drawing, where an initial click of the mouse marks the point of convergence, and the subsequent gesture

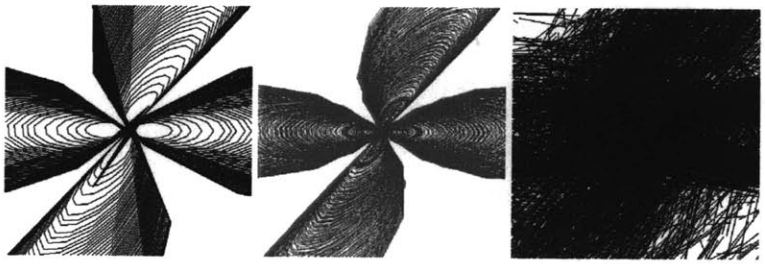
delineates the shape. Complexity can be added by defining more than one point of convergence, and moving the points onto, or far off the canvas. Examples are presented in the Development subsection.

RANDOMNESS AND THE HAND

These three panels were created with the same computational line consisting of a parametric curve algorithm. The quality of the drawing is immensely affected by the gestures that are made with the line.

The compositions below were generated by the above gestures

The first panel illustrates a slow line drawn from the top left corner to the bottom right. The middle panel depicts a more irregular gesture that traces over the canvas several times. The right panel shows an erratic line that deviates greatly from the diagonal, and moves quickly in opposing directions. It still retains some of the quality of the shape but is expressive of the gesture.



Stills from the POPPY applet by Joanna Berzowska

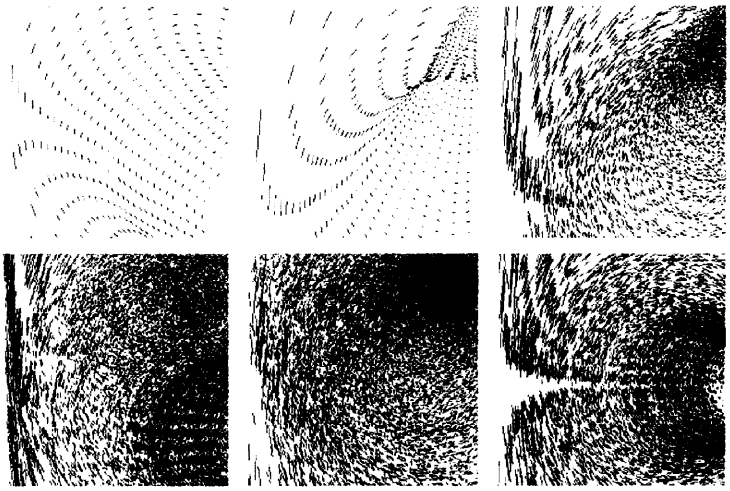
The visual form exhibits a tension between pattern repetition and variation. The expressive, agitated quality of the drawing is attributed to two factors.

First of all, the speed of the machine introduces a level of randomness into the drawing. Some aspects of the drawing are impossible to control because of the inability of the processor to keep up with the hand. Some algorithms are impossible to program because of speed issues, and thus certain design decisions are based on the abilities of the technology to do what we want it to do. If an algorithm is too slow, we must change it, or eliminate it from our repertoire. This is not necessarily different from other media, which also have their limitations. The difference becomes evident, however, in six months' time, when the things that were too slow suddenly become adequately fast. Computation is a unique medium insofar as cutting age technology, increasing speeds or storage capabilities often motivate art pieces. It is not necessary to be cutting edge or take full advantage of the technology to make meaningful computational art, although it does allow the artist to make things that are considered innovative.

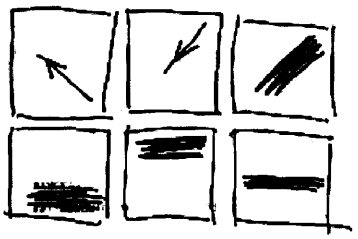
Secondly, computational drawing integrates algorithm and hand gestures (drawn lines). We are able to create visual artifacts that display mathematical uniformity together with a more "human" irregularity. The unexpected variations, the playful imperfection of the human hand are a quality of expression that simple computer programs cannot presently replicate.

In one panel in his class, Maeda addresses the question of indeterminacy as computation. He asks his students to question the nature of randomness and its aesthetic merits. He suggests that randomness is crass and overused, but can be used constructively to express surprise or the general unpredictability that is a common daily experience for most of us. Because in computation there is no such thing as true randomness, any number called random is still generated by an algorithm. If we understand the nature of this algorithm, we can use unpredictable behavior while still retaining knowledge and control of its actions. It is very important, for Maeda, to have complete control over the medium. [MAE98]

In the case of computational drawing, there are two levels of control. The first occurs in devising the algorithms, and the second consists of the hand-based gesture interaction that occurs while manipulating the algorithms. The control we exercise with the hand can be regarded as random, or certainly less deterministic than the control we exercise by asking the computer to render an equation. The use of irregular strokes can be compared to the introduction of randomness in image dithering. Because we cannot predict precisely where the stroke will be drawn, we are striving more for cloudy forms and textures.



Stills from the TUNNEL applet by Joanna Berzowska



The compositions above were generated by these gestures.

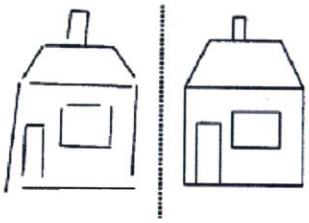
The TUNNEL line can produce all of the above compositions of line and shape. As a single stroke, it creates series of fragmented circles of increasing diameter. The diameter increases as a function of position in the plane and direction of movement. Using the single computational line in different ways creates distinct algorithmic compositions. A single gesture is represented as a pattern, determined by the movement of the gesture.

The rapid hand movement over a single area, shading areas of the canvas, creates areas of tone and shapes that emerge from these areas of tone. The shading is analogous to shading accomplished with a piece of chalk or a pencil on a piece of paper, but an algorithmic line determines a more complex pattern. The pattern is a function of position and direction of movement.

WHAT IS SKILL NOW?

The art of drawing has historically been the domain of skilled and talented artists who devoted much time and effort to the creation of images. Skills combined with creativity were prized. Artists spent considerable time in the studio, working through interminable sketches, to master these skills. Computers now provide a rapid and simpler process for creating many graphic representations that earlier demanded a high level of skill. Computers offer easy access to precise detail and high resolution, to photo-realistic representations, stylized illustrations and technical sketches.

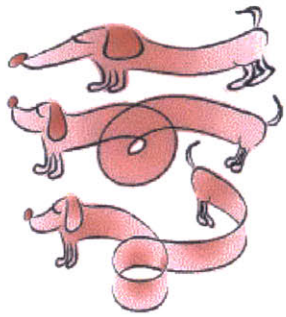
Sutherland's Sketchpad was the first program that demonstrated interactive drawing with a lightpen, directly on the CRT. It could create, manipulate, duplicate, and store engineering drawings on a computer. [SUT63] Considerable work has also been done for creating black-and-white illustrations, generally for engineering or graphical design work. Many programs are available to create design, layout and engineering drawings. The use of computation for line drawing involves creating specific sketching tools and beautifiers, which either make the drawing more geometrical, or more irregular, depending on whether the aim is to be ordered and precise or to emulate a hand-drawn aesthetic. High-resolution vector graphics also offer fun options for manipulating curves.



The automatic beautifier for drawings cleans up schematic drawings by removing hand-drawn irregularities. The system allows quick polygon-oriented sketches to be transformed into precise illustrations where lines are straight, precisely parallel and commensurate, sides of polygons are collinear, and vertices are aligned. [PAV85] [BOLZ93]

Another type of application, *Squiggle*, [DEN95] [PRE93] adds the irregularity of the hand-drawn to static, geometric CAD output to make the results appear as if they were drawn without a computer. It introduces a looser, more energetic look to computer-generated presentations by randomly tweaking the data file that specifies straight lines and curves. Pen-and-ink is an extremely limited medium, allowing only individual monochromatic strokes of the pen, yet skilled artists can create beautiful pen-and-ink illustrations incorporating a wealth of textures, tones, and styles. Salisbury's [SAL97] interactive system for creating pen-and-ink-style line drawings from grayscale images features a method where strokes of the rendered illustration follow the features of the original image.

A skeletal stroke [HSU93] [HSU94] [CRE98] is a drawing tool and image transformation instrument that changes the shape of pictures as if by bending or twisting, while conserving the aspect ratio of selected features on the picture. It is a vector graphics realization of a brush metaphor that uses arbitrary pictures as ink. The strokes are hierarchically structured so that they use a single skeleton to control an entire stroke and its features.

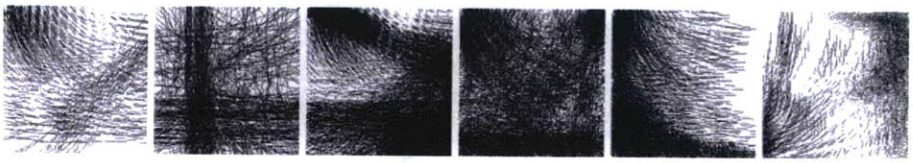


These methods are very powerful for illustration and technical drawing. Computer technology for producing more abstract and expressive work is less advanced. One option is to use one of the many available paint programs, which in their attempts to emulate traditional art tools often produce a sterile pastiche, a simulacrum, whose easily categorized style and generic aesthetic do not satisfy. Another option is to use image manipulation techniques, filters and other de-constructive methods to alter an existing image. Most of the published work on digital painting is concerned with the problem of emulating the process of traditional artists' tools, and creating both filters and interactive tools that produce the results that look similar to those produced by their predecessors.

One such method creates an image with a hand-painted appearance with a series of spline brush strokes chosen to match colors in the source photograph. Visual emphasis in the painting corresponds roughly to the spatial energy present in the source image. [HER98] This approach is noteworthy in terms of several innovations, long, curved brush strokes, a varying brush size, and changeable rendering styles.

Haeberli [HAE90] shows how scanned or rendered image information can be used as a starting point for painting by numbers. Many companies sell products that combine the stylistic expressiveness of traditional artist tools with the speed, flexibility and resolution independence of vector-based drawing. They are listed in Appendix B.

Traditionally, artists are trained in hand-to-eye skills, not the more obviously rational skills necessary to conceptualize the mathematics of drawing a line from the point (90,30) to the point (10,140). Therefore, the current tools that invite digital image expression tend to shroud the tools' computational aspects, and provide a user interface that inserts a layer of artifice between the user and the tools. This layer of artifice introduces stylistic constraints, both by pre-defining aesthetic choices, and limiting the set of possibilities. Artists carry over skills from their previous work, and continue to use the new medium in the old way. Needless to say, this is inadequate. The skills necessary to develop an expressive proficiency in the medium are those of thinking mathematically, of programming, of translating artistic vision into a concise, mathematical algorithm, and approaching composition in an explicitly procedural way.



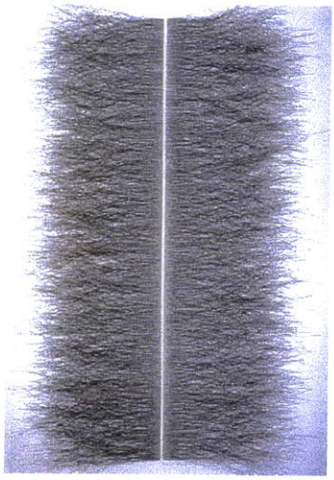
Composition created with the JoeyGraphics Java class

The importance of skill, of the mastery of a medium is partially lost in using commercial image software. The act of programming, however, provides an environment that must be learned and conquered, the long road that must be traveled and explored so as to develop a closer relationship, a conspiracy, between the artist and the medium. This sort of complex understanding is necessary to move beyond a preoccupation with the medium, and allow a greater comfort that will cause the artist to concentrate on content and meaning instead of tools.

ALGORITHMIC ARTISTS

This sort of thinking is not new, and many artists have been working with algorithms to create visual art. Algorithms are after all simply a set of steps or processes necessary to execute a task. It can be argued that the act of drawing in a more traditional medium is also an algorithmic process, [VER98] but a far more complex one. Algorithms are not a phenomenon unique to programming, but the advent of computers has given visual artists have the technology for composing art works algorithmically in ways than were unavailable before. [VER98] This subsection is an algorithmic art gallery, with sample images and quotations from a set of algorithmic artists.

Touching is a very broad concept. In these images, lines are playing a game of touching, of near-touching, of avoiding, of seeking, of crossing and intermingling: a manifestation of the purity of the line and an invitation to meditate. Algorithmically generated drawings, drawn on a pen plotter, constitute a very small segment within the area of computer artwork. It is this small segment, however, which I find most fascinating. This has to do with the archaic notion of a mechanical extension to the drawing hand, unlocking a universe of machine-generated drawings utterly different from hand drawings. [DEH98]

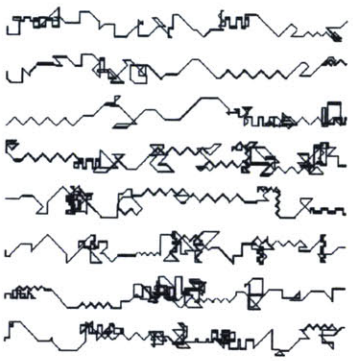


b97.9.3 by Hans E. Dehlinger

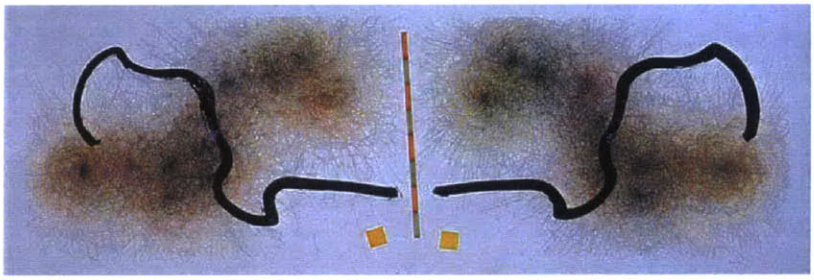
Yoshiyuki Abe

My interest in computer imaging is producing images that have never been imagined. That means I am not reproducing the images in my brain but accepting or selecting the images generated by math based processing by a preset algorithm. Computer is my collaborator and my role in the creation process is setting parameters, changing surface attributes and waiting for the results processed by the computer. [ABE97]

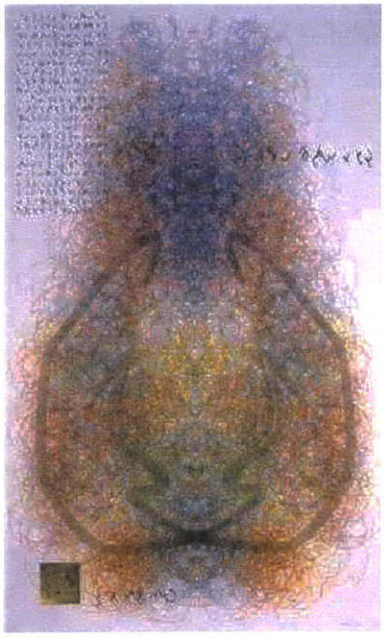
The computer became a physical and intellectual extension in the process of creating my art. I write computer algorithms i.e. rules that calculate and then generate the work which could not be realized in any other way. My artistic goal is reached when a finished work can dissociate itself from its logical content and stand convincingly as an independent abstract entity. [MOH98a] My art is not a mathematical art, but an expression of my artistic experience. [MOH98b]



Manfred Mohr



Roman Verostko



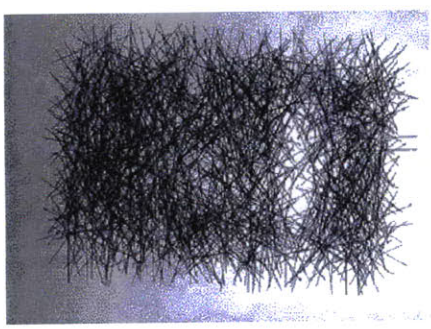
Roman Verostko

Most of my work for the past 40 years has been with pure visual form ranging from controlled constructions with highly studied color behavior to spontaneous brush strokes and inventive non-representational drawing. Such art has been labeled variously as "concrete", "abstract", "non-objective", and "non-representational". In its purest form such art holds no reference to other reality. Rather one contemplates the object for its own inherent form similar to the way one might contemplate a flower or a seashell. Procedures for creating such art have been evolving from the work of its pioneers in the first decade of this century. [VER98]

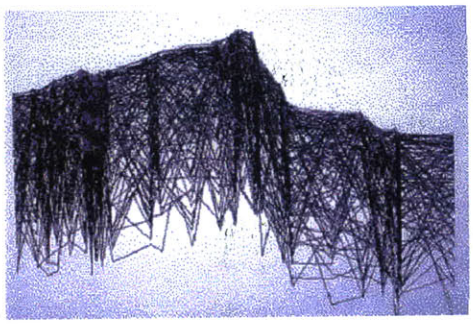
The world of forms available for artists who create such art is vast, an uncharted frontier of "unseen" worlds waiting to be "discovered" and concretized. With the advent of computers, I began composing detailed procedures for generating forms that are accessible only through extensive computing. My on-going work concentrates on developing this program of procedures for investigating and creating such forms. By joining these procedures with fine arts practice, I create objects to be contemplated much as we contemplate the forms of nature. [VER98]

Jean Pierre Hebert

I started this so long ago that many of the wonderful tools which are available now to artists did not exist. At that point I was forced to make my own and I had to program them. This is the origin of my work, which is generally personal in its goal and forms. Everything I do is based on mathematics and geometry and line arts. I chose to use a plotter because it was the most affordable piece of hardware I could put my hands on at the time. ... It was a very dynamic process for me and this process is going to end because I am using a technology which is becoming obsolete. [HEB98]



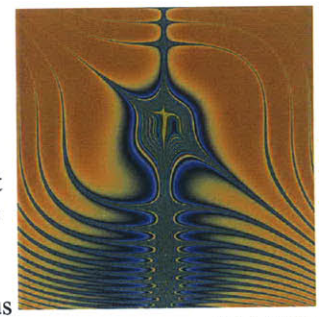
Interruptions, 1968 Vera Molnar



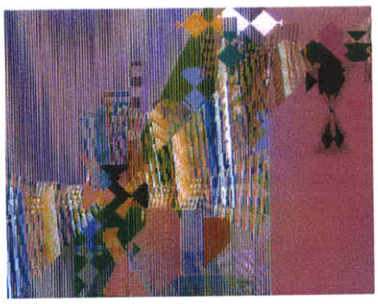
Variations Sainte-Victoire, 1996 Vera Molnar

The image obtained by a painter using a computer stops being an accumulation of unknown badly defined forms and colors. It becomes instead a pattern of thousands of distinct, intermittent, and quantified points. The position in space, the colorimetric values of these thousands of points, are perfectly defined and numerically accountable. In this way, the painter controls each one of these points. At any moment, the artist is able to modify the value of one or several points, or even the total number of them. As a result, innumerable successive approaches (many sketches, to use the accepted history-of-art term) can be shown on the screen. Proceeding by small steps, the painter is in a position to delicately pinpoint the image of dreams. Without the aid of a computer, it would not possible to materialize quite so faithfully an image that previously existed only in the artist's mind. This may sound paradoxical, but the machine, which is thought to be cold and inhuman, can help to realize what is most subjective, unattainable, and profound in a human being. [MOL98]

With technology it is possible to manifest mathematical ideas as images, sounds, sculpture and even poetry. Artists in all media have found mathematical processes of value in their creative enterprise. These processes are often described using algorithms. In describing mathematical processes with algorithms, beauty and meaning can be discovered. Numbers are mapped into light and/or sound, and perceived through the senses as objects. It is the mathematical source of these works that has aesthetic worth. Algorithms, implemented on computers, make it possible for us to see and hear the beauty of mathematical processes. [EVA98]



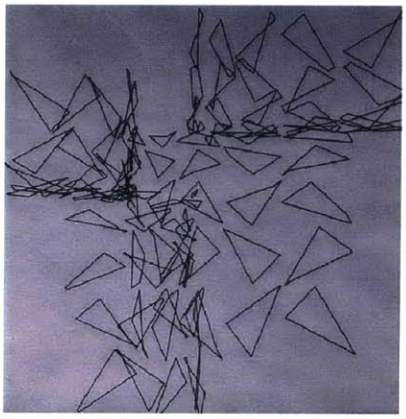
Brian Evans



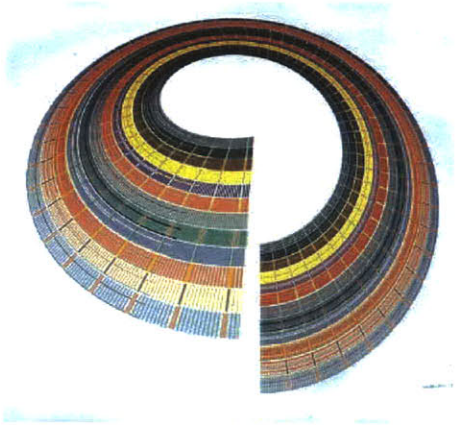
Samia Halaby

I had clear ideas about abstract form in painting and intuitively recognized the potential for expanding this new twentieth century visual language through the computer. … I seek to expand a radical new language in art. I do not feel impressed by ray tracing or by scanning photographs. I can draw well enough and render in traditional academic ways. … It is not the computer's ability to imitate optics that I find fascinating. The computer, as a digital media, has allowed me to deepen visual ideas of form and illusion not easily possible otherwise. [HAL98]

In my early work, I created a sense of presence of invisible forces in nature. For me, these forces in nature are metaphors for the interpersonal dynamics between people. I created algorithmic images, using mathematical descriptions of phenomena such as light reflecting off of irregular surfaces, that embodied these dynamic forces. In these drawings, environmental phenomena that we sense, like the wind, were visualized and given a physical presence. Algorithmic patterns were also created on fabric using heat-transfer xerography. This mapping of environmental behaviors onto cloth propelled this algorithmic representation back into the natural world. [TRU98]

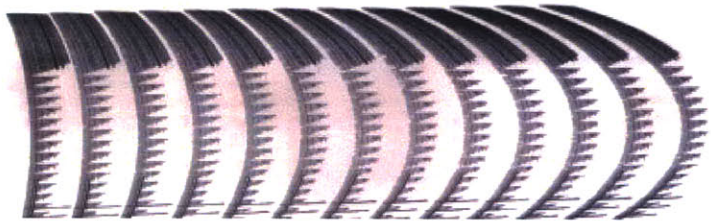


Construction E5, 1975 Joan Truckenbrod



Mark Wilson

My drawings and paintings are made in a two part process. First, images are generated on the screen of a microcomputer using a variety computer programs. Some of these programs employ simple random procedures, others utilize permutations of graphic elements such as cellular automations. Next, a rectangular section of the image is plotted, pixel by pixel The pixels can be drawn as circles, filled boxes, crosses, and so forth. They can be large or small, and can be mapped onto various geometric surfaces, such as planes, cylinders, and cones, Finally, these surfaces are projected into perspective space. I have written all the software. [WIL98]

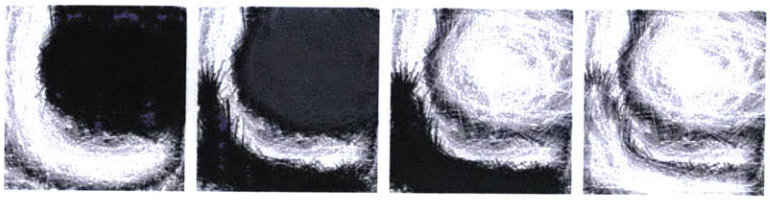


18 G 90: Mark Wilson

These working procedures offer much greater versatility and freedom from the traditionally small photographically-based computer graphic formats. The overall picture-making process is reminiscent of collage. Elements are selected and placed on the surface. Compositional decisions are made step by step. Thus, the final appearance of the image is not predetermined by the machinery, but by artistic judgment.

It would be impossible to realize my works using any other medium. I have attempted to directly use the digital nature of this medium. Rather than trying to disguise pixels, they have become the central element of my artmaking. Technology underlies all of my working procedures, but the ultimate goal is a simple one: to delight and intrigue the eye of the viewer with images that can be seen only with the aid of a computer. [WIL98]

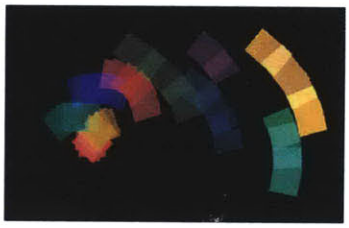
DYNAMISM



Consecutive stages in the drawing of a HAIRY drawing by Joanna Berzowska

BACKGROUND

"Dynamic form is everywhere ... As technology advances towards greater possibilities for realizing dynamic forms ... it is inevitable that most objects that we use will become increasingly dynamic." [MAE93] Interactive digital pieces are intrinsically dynamic and allow direct manipulation with hands, breath, voice or movement through space. The mappings of physical motion to screen events can be very intuitive and give a certain satisfaction of being in close physical contact with the medium. Dodge's *The Winds that wash the Seas* provides sensors for human breath and a full bathtub that can recognize the motion of hands in the water. The participants breathe and splash their hands in the water to interact with the piece. [DOD97] In Baird's *Sashay* "a participant uses a set of emotionally evocative gestures to interact with a character - the Sleeper - by constructing an animated dream." [BAI97] Seaman's The World Generator is an interactive system that allows viewers to construct and generate dynamic poetic worlds in real time based on an interactive template of potential choices. [SEA96]



A still from Motion Phone

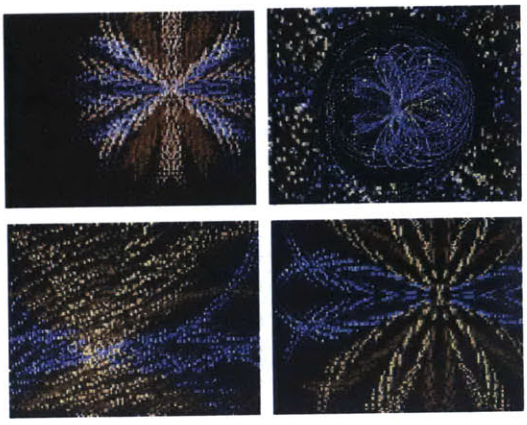
Snibbe's *Motion Phone*[1] is an "attempt to open up the language of abstract animation to a general audience by allowing spontaneous human gestures to be captured in all their subtlety." [SNI96] It moves beyond traditional animation tools, and becomes an instrument, supplying the user/artist with rich expressive mappings of gesture to shape, color and motion.

Non-interactive dynamic forms, or kinetic art, is exemplified through film, animation and musical, lyrical compositions of shapes and colors. Early 20th century avant-garde aesthetics rested on the conviction that light and movement build the poetic dimension of cinema. The lyricism of movement that can be achieved with computers is often compared to poetry or musical forms.

Sims' *Primordial Dance* is an experimental animation containing a progression of abstract textures and patterns. It is a study of emerging and transforming mathematical equations that might be considered visual music in that it attempts to provoke emotion with underlying structure and complexity without relying on specific representational entities. Whitney [WHI91] was a pioneer of composition that simultaneously encompassed music and visual form. He explored the

[1] First built in 1991. Presented at SIGGRAPH 95. Prix Ars Electronica 96 in the category Interactive Art.

relationships that exist between musical and visual design. Musical tempo and harmony determined much of the visual aspects of his work. In *Digital Harmony*, fluid, orderly form and movement generates or resolves tensions much in the manner that orderly sequences of resonant tonal harmony impact emotion and feeling.

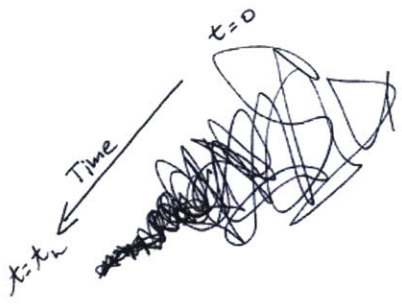


John Whitney's Digital Harmony

Maeda has introduced the concept of reactive graphics, computational designs that are programmed to directly respond to actions from the viewer of the graphics. "There are reactions that are concise, cooperate, and can smoothly flow into other reactions in a breathtaking manner appropriate to the digital medium." [MAE98b]

The computational medium allows us to create tactile, interactive environments inviting an exploratory involvement between the user and the artwork. The process becomes a new interaction model with which the user and computer can communicate. It has to be learned and can be controlled to produce expressive compositions. Both movement and physical change are very important elements of computational art, just as important as shape, color and composition are to a painter. Because a computer contains a processor that constantly and consistently assimilates data, manipulates it and can output continually metamorphosing sets of objects, computation is a most natural medium for the exploration of dynamism.

SECOND ATTRIBUTE: DYNAMIC LINES



Computation allows us to create dynamic, interactive, exploratory work. We create drawing tools that respond in unconventional ways, lines that fill the canvas with color, and animate as they are drawn.

Some of the traditional formal components and unifying principles of design are line, shape and form, space, texture, value, color, dimensionality, repetition, variety, rhythm, balance and emphasis. Computation allows us to add a new word to the traditional vocabulary

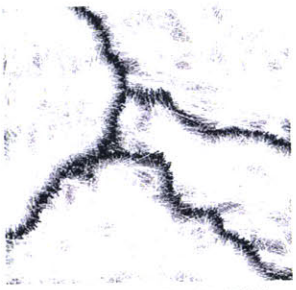
Using the conventional physical interface of a mouse for hand drawn input and a monitor for visual output, we see the result of gestures we make with our hands. With behavior, the gesture is mapped to more than process and appearance. It is mapped to a behavior that influences both the appearance of the element over time (dynamism) and the appearance and relationship of other elements around it. What is exciting is that we can define this behavior and make it abstract.

DAFFODIL composition by Joanna Berzowska

The DAFFODIL composition was created using a dynamic stroke. A gesture delineates a string of points. Each point is the center of a small circle. The pools of color are drawn as concentric circles of ever-increasing radii. The centers of the circles follow the path of the line, and create ripples of color. The distances between consecutive circles can be adjusted to create grid patterns or continuous areas of color. The distance between consecutive generator points can also be altered. The colors are also affected by time, and intricate patterns emerge from the process of lines drawing over each other until their lifespan ends. The circles draw over each other, in a dynamic, animated, interactive process. The interaction of colors is particularly captivating and entertaining. Dynamic lines create a rich visual and animated environment controlled directly by gesture, by the hand. Unfortunately, the process often becomes more pleasurable than the composition.

Designing dynamic lines is a difficult task. It is easy to think of algorithm when devising a static algorithmic line, but with added dynamism it becomes increasingly difficult to realize the connection between the individual computational line, and the eventual composition. Sketching with the lines and altering their design becomes a fun but time-consuming process.

The HAIRY line projects little hairs or tentacles from its spine. The tentacles are revealed differently as they change color over time, slowly fading to white. Their life span ends when their color fades completely. Because they are drawn in slightly different places each time they redraw, the black edge remains, and becomes positive space. Overlapping fading tentacles form texture.[1]

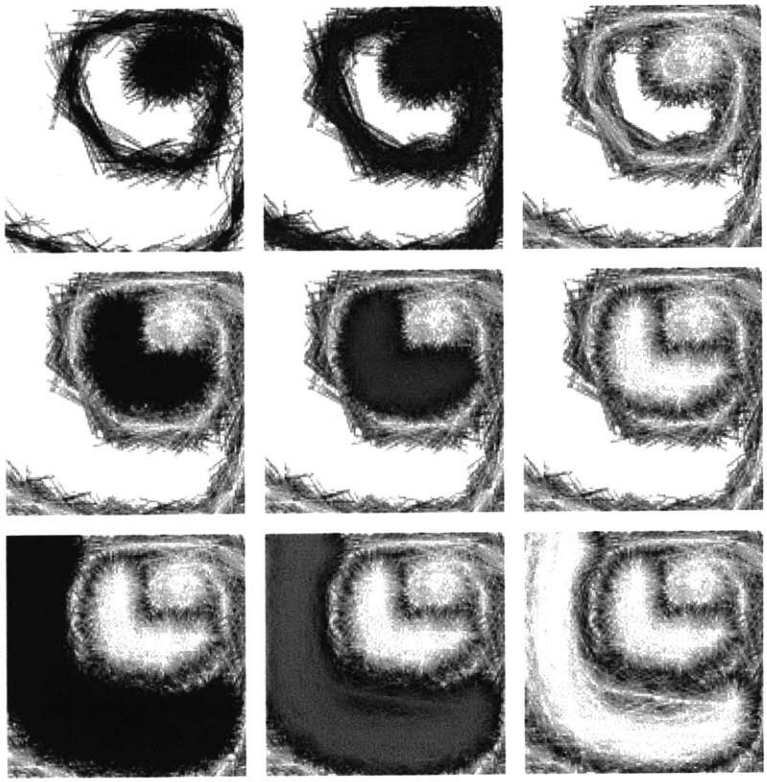


The HAIRY line

The following sequence illustrates the drawing process. The lines start out as a thick, black mark. After it has been drawn, the mark expands, grows in thickness, projecting hundreds of little hairs out of its spine.

[1] The basic component of this line is a method in the JoeyGraphics class called drawHairyLine(int x1, int x2, int y1, int y2) which fades over time. The line segments redraw in different places but shift over time, so the effect is an interesting border.

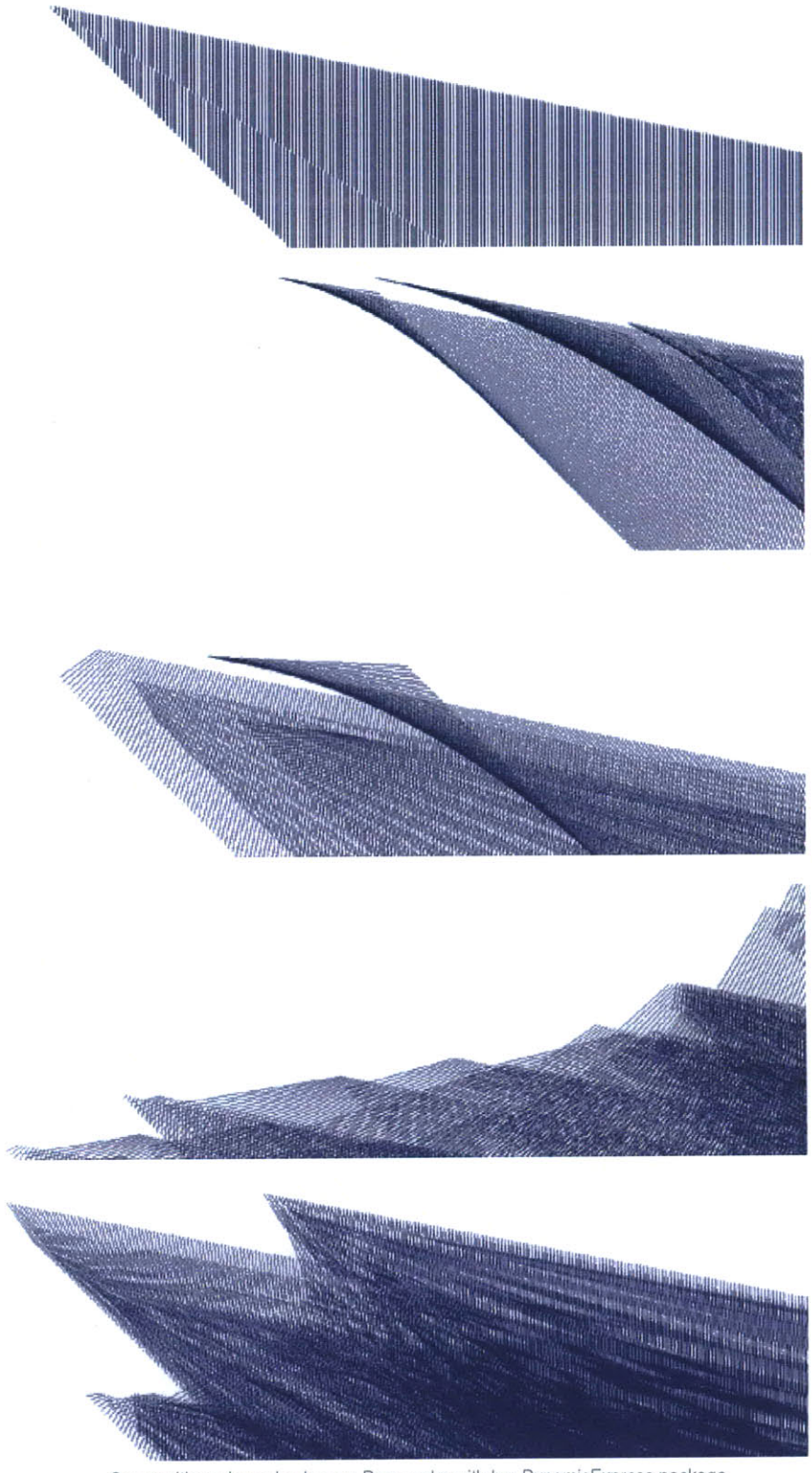
The hairs start out dark, but become lighter and lighter over time. The color of later hairs fades as they are drawn over previous ones. As a result, the overall composition fades. The space where the initial line was drawn becomes textured negative space, and only the outline remains as dark positive space. The following sequence of stills illustrates the drawing process as several lines are added over time, building up a drawing. The leftmost picture in each row depicts the state of the drawing as a new line is drawn. The rightmost picture in each row shows the drawing after all animation has come to a standstill.



Consecutive stages in the drawing of a HAIRY drawing

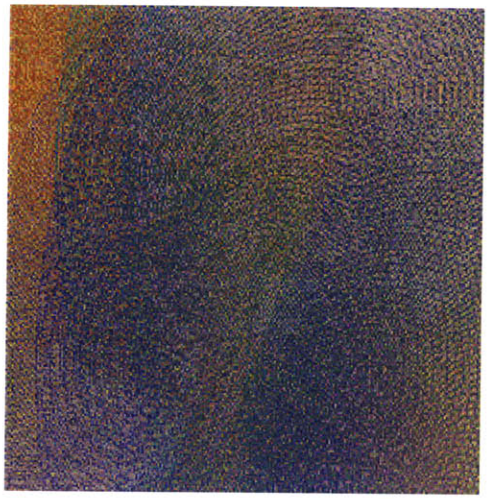
Each line, although it is dynamic, comes to rest in a predictable final state. The animation does not continue indefinitely. This can be compared to painting where the paint can drip initially, but will always dry in an expected manner.

The next series of stills are the end products of simple lines drawn with the DynamicExpress package. It is a package of dynamic lines, which do not change color, but extend spatially with time. The vertex of each triangular shape is the originating point of a gesture. The quality of the gesture, its speed, direction and position, determine the appearance and direction of the animation. The animation consists of a visual echo of repeating lines originating from the vertex. As the cursor draws over the display, the smallest lines at the vertex of the triangular shapes are traced on the display. Each small line generates a resonant visual reaction of progressively larger lines that are rendered in relation to the movement of the cursor.



Compositions drawn by Joanna Berzowska with her DynamicExpress package

The first panel shows a short, controlled gesture. The next two are representations of more vigorous, repetitive gestures. The last two show a repetitive movement back and forth.



This example draws a series of parallel lines, which redraw, line by line, for a finite time. As such, close lines redraw over each other, and the composition changes over time, to eventually settle on its final state. At any one moment in time, however, we can say that the line has a certain appearance. The dynamism is more difficult to describe.

PLEASE SAVE MY WORK

The drawing programs do not have an undo function. This can easily be programmed, but goes against the spirit of computational drawing. The nature of computation is such that the state of a composition can be saved at any moment in time, and older states can be reverted to, mistakes can be undone. Is this desirable for drawing?

Each drawing is stored in computer memory as a vector of time annotated line objects, recording the coordinate points that make up the gesture, and the parameters and algorithms used to represent the gesture on the canvas. As such, each drawing is a computer file of numbers and methods. That file can be conserved and thus can be recreated and replayed. We must ask whether we want a unique process of drawing or a performance. Do we want drawing or animation?

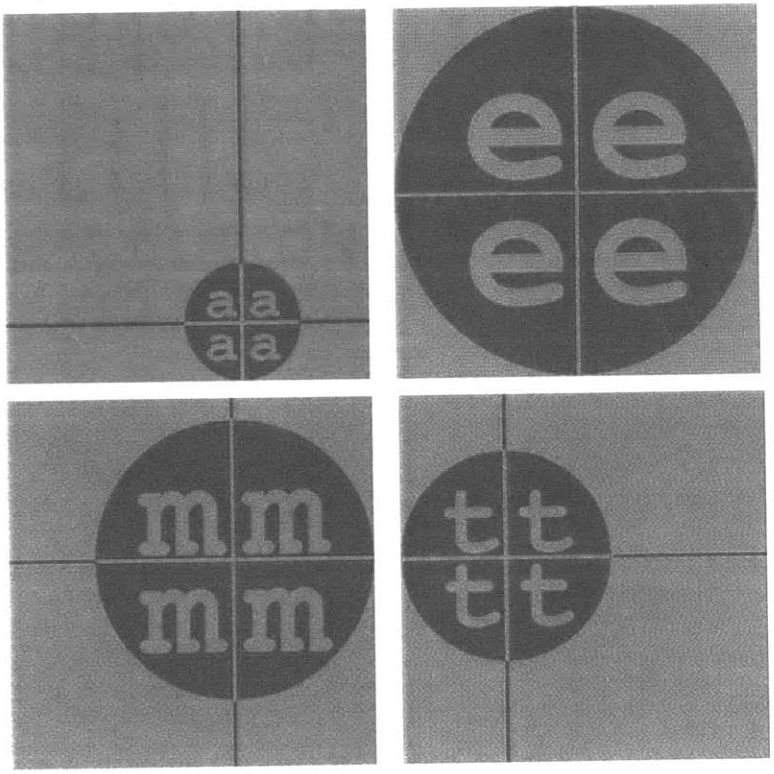
"Can an art which is concerned, as western art has always been, with appearance, with the look of things, with surface reality, have any relevance in our systems-based culture in which apparition, emergence, transformation are seminal? Can representation coexist with constructivism?" [ASC93] Then again, we should not confuse the procedure by which the artist creates algorithms with the procedures by which the algorithms execute the work. This may be the most important distinction we need to bear in mind when discussing art and algorithm.

Computational art has a potential for distribution and for the making of multiple copies. The longevity and non-degradable character of digital material (considering back-ups and multiple copies) is also problematic, as well as its fragility and dependence on hardware (so the hardware is really an inherent part of the work, as the canvas is an integral part of a painting).

> Digital media not only lends itself to being copied, but itself operates in terms of reproductive operations. On the web, novices learn to make their own web sites by copying the scripts making up other sites. But experienced programmers at times do no different when writing code. And computers are designed to copy information from one area of memory to another. [BRO98]

Works can be copied and reproduced in such a way that each copy, as a digital document, is intrinsically identical to the original. In fact, the concept of the original falls apart when we consider how a computational work is created. The recursive and continuous process of shaping a piece of computation precludes the concept altogether. The western tradition of visual art relies heavily on the concept of original. In computation, this concept is absent even in "the most perfect reproduction". [BEN92] Walter Benjamin speaks of the importance of the concept of the original in a work of art, especially of its presence in time and space. The presence implies authenticity and includes all the patina that accompanies a work's existence in the physical world. [BEN92] Disallowing undo functions and the ability to save the composition at arbitrary points during the process of drawing hopes to retain some element of authenticity for the resulting compositions.

Maeda addresses the topic of continuous interaction in computational art in one of the panels in his class, entitled "One shot performance vs. Continuum". He is referring to reactive graphics, and interaction with graphical composition. Reactive graphics can incorporate human gesture as the principal component of a real-time animated environment. This environment is not a drawing, it is a composition, a computational, dynamic composition. It can animate with a single soul, its breath cannot be retrieved, or it can provide a continuous environment of animated interaction. The viewer can return and experience a similar interaction, or never be able to replicate the original.



In the type example above, the center of the cross hair follows the cursor around. The radius of the disk is equivalent to the shortest perpendicular distance between the center of the circle and an edge of the canvas. Animated text is displayed around the crosshairs, displaying the consecutive letters of a poem. If we think of interactive art as visual

dynamic forms for expression, the poem animation is an action/reaction interactive occurrence, whereas the direct manipulation of the layout by the cursor is a fully interactive experience. Similarly, there are two sorts of temporal experiences here. The text that animates does so only once. From the time that the Java applet is launched, the poem is printed, letter by letter, until its last word. Then the letters stop. The dynamic crosshairs and disk of gray offer a longer lasting temporal interaction. The reader of the graphical poem can keep moving the cursor long after the last stanza has run out.

Computational drawing is concerned with eventually shaping a static composition. As such, it cannot consist of a continuous animated environment. The dynamic lines have a pre-determined life span, and always settle in a deliberate final state. Their dynamism is an integral part of the process of drawing form. Dynamic lines execute some tricks as they elucidate their ultimate visual form.

ARTISTIC PROCESS

> At this historical moment, does it make sense for an artist to write software and pursue an individually styled algorithmic art? [VER94]

In computational drawing, the creative process is augmented to include the creation of the drawing tools. Questions arise regarding the bipartite composition of computational drawing, issues of authorship, the artist's shift from a creative role to an evaluative role, and the problem of making tools versus making art. The computational medium offers us two features: dynamic form and interactivity. These applications of computation can be used to create pieces that are divorced from physical references and that are evocative in an unfamiliar yet intuitive way. Freed from the constraints of previous models, mappings and metaphors, a more biologically intuitive form of art can emerge.

INTERACTIVITY

Interactivity is becoming one of the most important features of contemporary civilization. [KLU95] In computation, it embodies the dialogue between machine and artist. The adoption of computation as an art medium has engendered new forms of interactive shaping of form and content.

It has been advanced that the beginnings of interactive art coincided with the advent of video tape, which allowed a previously linear film experienced to be manipulated by the viewer, who could now influence the course of the film experience. Film viewing was transformed into film reading, a linear yet adaptable process of perceiving, comprehending and selecting the narrative and stylistic experience. Umberto Eco describes a purgatory of lost art, where everyone is watching personalized films and deserting the movie theatres. "With a single pattern and an accompanying package of variants an individual could make, for example, 15,741 Antonini movies." [ECO93] These features of interactivity should not be perceived as a threat to the author, the editor of form and content. Interactivity is simply a new element of style that must be understood and applied in meaningful ways.

Interactive art demythologizes the role of the artist, assigning to it the function of the designer of contexts for receptive creation. The concept of the author is being replaced with the notion of authorship. [KLU95] Instead of creating, expressing, or transmitting content, the artist is more concerned with designing environments or contexts within which the viewer can construct experience and meaning. [ASC93] Seaman has coined the term "recombinant poetics" to talk about computation as an artistic medium which heightens the potential for an intermingling of the knowledge of the viewer with the "re-embodied intelligence" of an author. Computers can function as databases of media objects (image, sound, video, text) and the artist, or maker of interactive pieces, can author environments in which these objects interact with each other and with the viewer. These environments enable intelligent emergent poetic responses to viewer interactivity via the application of models of poetic construction. [SEA96]

Here, two levels of interaction are at work. The first level is the writing of the drawing program, the programming of the computational lines. The second level involves interfacing with the computer program, using the hand, and a hand-held input device to produce images with the computational lines. A single user, the artist, is asked to perform both interactions.

AUTHORSHIP AND RE-APPROPRIATION

The drawing procedure has been previously described as a bipartite operation of iterative programming and mark making with the hand. Let us consider a different approach to computational drawing, as a process comprised of three elements. The first is a raw material or medium, traditionally ink and paper, which is replaced by the computer, a programming language and devices for input and output. The second element embodies the gestures performed by the artist in the chosen medium. These gestures become more efficient and produce more interesting or unique results as a deeper understanding of the medium is achieved. Finally, there is an element of intended meaning attached to those gestures. The intended meaning manifests itself through memory, metaphor, and an obscure reasoning process that is often described as the gift of artistic insight.

In computational expressionism, the artist is encouraged to program individual algorithms, then use elements of gesture to deposit traces of these algorithms on the pixels of the CRT. The intended meaning, artistic desire and intuition guide the entire process.

Let us define "the second artist" as someone who decides to use computational lines programmed by someone we will call "the programmer", without altering the algorithms. The line created by the programmer has physical appearance, a shape, a pattern, a representation of an algorithm, a color. The drawing process demands a gestural method of interaction with the canvas to generate images. The line making abilities of the hand, and the interface of an input device and the screen, appeal to the second artist as a way to produce visually lyrical worlds. How can we think of the expressive experience of the second artist? Is it merely exploratory, or can it be thought of as a whole creative encounter with the medium? Is the programmer merely a tool maker, or an artist in her own right?

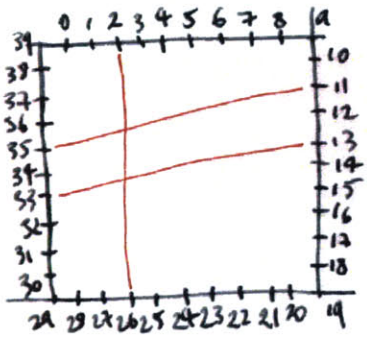
When the second artist decides to use the same computational lines, authorship is shared, and the second author navigates in a world equipped with elements of style, expression and composition, which are explored to create visual pieces. The programmer provides visual metaphors that establish stylistic guidance. The programmer is an author of visual context more so than visual content. The second artist completes the visual experiences through action and interaction with a system of form or meaning that carries expressive potential constructed of image elements within a defined system of constraints.

A majority of meaningful art necessitates great expanses of deliberate emptiness, both visual and conceptual, that excites the reflective sensibility and stimulates viewers to introspection and thought. Thus, the second artist can have an audience experience in the sense of utilizing minimalist elements of style provided by the programmer to reflect upon questions of form and style within these constraints.

The experience of the second artist can also be thought of in terms of re-appropriation, a common theme in post-modernist art where elements of art pieces created by others are used and composed in ways that is individual to the appropriator. The process of making art always requires an artist to use the whole of previous knowledge, wisdom, insight and memory, and reevaluate it so as to evoke, to create something new. Levels of authorship are ultimately irrelevant.

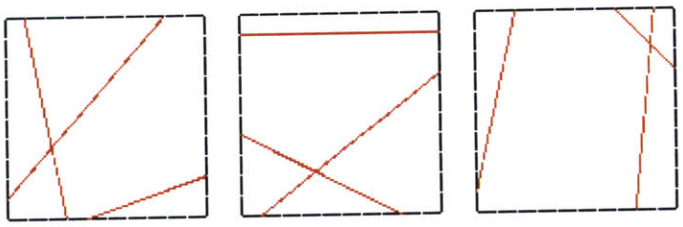
The extensive arguments regarding authorship of interactive work are not very meaningful in the definition of computational drawing, because this is a study of an artist programming individual tools, and drawing with them. The drawing is ideally a two-stage process, where creating the line algorithms constitutes one half of the work. In this vein, others can use the lines as well, and authorship of the resulting compositions is shared in a sense, but the desire is that each artist will modify at least some of the code to make it unique. As such, the Java classes are organized in a logical and object-oriented way, so that alteration can be easier. Artists are expected to work with the code and create their own dynamic, interactive lines, which are subsequently used to draw the compositions.

ROLE OF THE ARTIST



When asked to produce a composition consisting of three lines that slice through a red square, the artist can write a program. The procedure is to subdivide each edge of a square into ten segments of equal length, and label the endpoint of each segment in a clockwise manner from 0 to 39. By asking the computer to generate three pairs of random integers in the range 0-39, we can connect these as endpoints of intersecting lines.

Repeating the process one hundred times produces one hundred compositions. The artist's role then becomes to select the most interesting ones. The process is evaluative rather than creative.



Three compositions selected by the above process

A similar situation is to be found in photography, where photographers used to require lengthy poses of a single composition. Materials were costly and exposure times lengthy. In the present world of fashion photography and photo documentary, thousands of shots are taken in order to produce the award-winning photograph. This situation is exactly that of computational art. Copies and minute changes in the program itself are trivial to make and undo.

The evaluative skills of the artist are important. The algorithm creation is a repetitive, iterative process, until desired results are achieved. Similarly, the nature of computation is unique insofar as the digital has no concept of original, or concept of finished work. The bits can be saved, changes undone, and possible combinations of visual elements escalate. The artist must choose from among many possibilities, instead of creating in a single breath of creation.

Computers are procedural beasts. The evaluative process of making computational lines, the decisions the artist makes are the artistic procedures. These have to do with the individual sensibility of each one about the entire art-making process. "Each makes artistic choices as to which procedures he will articulate, what form the work will assume (size, materials), how the work will be presented-and, yes each one assembles the code to generate them-just so in terms of all their physical qualities whether in visual or sound arts forms." [VER94] The role of the artist, in response to new technologies, has been to shift towards a more evaluative role.

ARTISTS AS PROGRAMMERS

We are studying what sorts of art forms will emerge as artists start working with a programming language as a raw material. Designers of interactive digital forms who do not program must express form and interactivity with concepts derived from available software. Their image of the design space is fragmented and incomplete. It is shaped to a large extent by the tools they have been using and by the solutions they have seen. Consequently, many possible solutions are unimaginable. In addition, because most tools with which one creates digital pieces have underlying metaphoric or stylistic structure, the content generated can be heavily influenced. By working in a programming environment, we produce pieces that exist independently of the hand of other authors or programmers. Maeda, in *Design by Numbers*, states that the core skill of a digital designer is the practiced art of computer programming, or "computation". [MAE98b] Artists must design their own tools and their own interactive experience.

For a programmer, the artistic process is the act of writing the code. Different skills are involved in traditional art, which requires great hand/eye coordination and intuitive faith in the hand, and computational art, which demands concise, analytical thinking. Artists

will choose one set of skills over another, depending on their level of comfort and practice. Programming languages can be designed for artists who necessitate a more bricoleur approach to producing algorithms and procedures.

> To support the bricoleur interaction designer, it is important that the tools and materials allow for the designer to work fluently in an iterative and explorative modus operandi. It is important that the software environment enhances a "dialogue with the material" in the creative design process. [SVA97]

In Svanaes' *Painting with Interactive Pixels* [SVA97], interaction designers are asked to construct GUIs by painting with pixels that have inherent behavior. Interaction is constructed directly with the brushes that create the lines and shapes on the canvas. Svanaes defines kinaesthetic thinking as corresponding to visual thinking for design, the modus operandi involved when interactive behavior is designed directly, without making use of abstract representations of behavior.

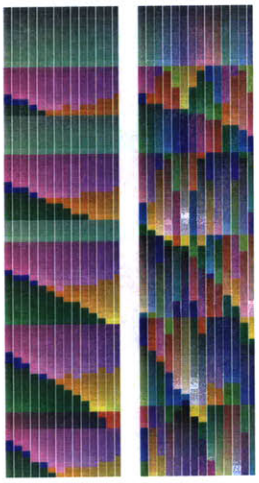
For a visual artist the study of drawing primitives, the line, the circle, can be very instructive, especially those that deal with transformations such as translation, rotation, and scaling. To design a procedure for generating a new form one must think in very primitive and analytical ways about the very nature of the drawing process.

MANIPULATING THE CANVAS

> Drawing a stroke with a pen is no different from drawing a stroke with a mouse. The real challenge is to discover the intrinsic properties of the new medium, and to find out how the stroke you "draw" with computation, is perhaps a stroke you could never imagine without the computational medium. [MAE98b]

Algorithmic and dynamic strokes have been discussed in previous sections. An even greater abstraction of the mapping between gesture and computational line representation can produce a stroke that creates and alters the patterns and colors displayed on the canvas. A design is programmed to reflect certain parameters. The artist determines its color range, patterns, and rules of manipulation. Certain characteristics, however, remain dependent on later input. The input takes the form of a gesture, a stroke. The whole design, therefore, can be manipulated with gesture. Rubbing and manipulating the canvas with the input device creates interesting variations in the design. These variations would be much harder to achieve were the artist to try to program each one separately. The design on the right can be altered to produce a wide variety of compositions, simply by drawing strokes over its surface. The hue, saturation and value of each colored square are direct functions of x and y coordinates of the canvas and the gesture.

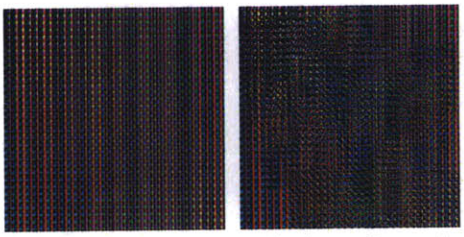
m11 by Joanna Berzowska



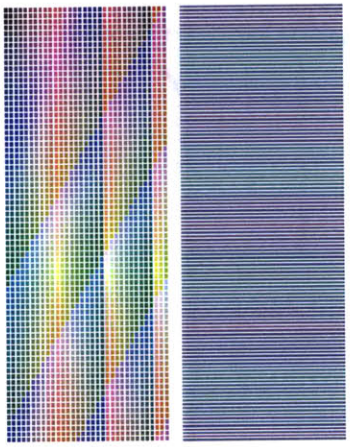
FIELDS by Joanna Berzowska

The computational line defines shape, color, pattern and motion. The canvas can also have properties; it can assign color to the composition depending on the coordinates of the pen, or affect the shape of the line, as a function of coordinates. In FIELDS, the image begins with a series of pictures of an abstracted landscape, like the still frames of a film, each differing slightly from the next. As the cursor starts moving over the images, an amount of disorder is introduced. The scenes fall apart, the colors become brighter, more contrasted. Over time, the scene reverts to its tranquil state. This is not a drawing, but uses gestural input to affect composition.

In this example, a pattern is generated. The user can upset its regularity with the repeated rubbing of the cursor over its surface. Some elements of the original pattern remain.



m4 by Joanna Berzowska

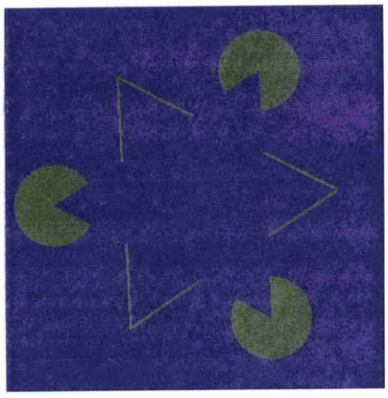


m7 and m9 by Joanna Berzowska

The images on the left are two further examples of manipulating the canvas color space. The size of the pattern as well as the colors are determined by (x,y) coordinates of a mouse click. The artistic process is one of design, not drawing, but design heavily mediated by computation. A general visual theme or set of constraints is programmed and then explored along several degrees of freedom, functions of cursor position. This process of design allows a greater amount of experimentation and a different approach to sketching.

The canvas can also be manipulated to reveal elements that were not visible earlier because of optical illusions, or because of a rearrangement of geometrical objects. A Kaniza triangle consists of illusory contours. We perceive a blue triangular surface in front of yellow circles and a black triangle. Equiluminance inhibits motion perception, direction sensitivity, stereo perception, perspective, shading, and certain form perception, such as the illusion associated with the Kaniza triangle.

In the KANIZA applet, the motion of the cursor causes changes in the scale and rotation of the yellow graphical elements on the purple background. The position of the cursor also alters the color of the background. As the two colors become closer in value, the Kaniza illusion becomes less obvious. Shapes are brought in and out of focus through the manipulation of the graphical elements.



KANIZA applet by Joanna Berzowska

This illustrates an even greater abstraction of gestural manipulation as drawing. Shapes fade in and out of focus as a function of position, which determines color and thus influences the optical illusion.

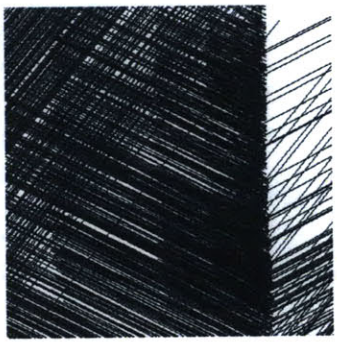
In the piece called MAN, a line does not link two points visually, but metaphorically. The line becomes a storytelling or a semantic tool. The canvas shows a minimalist silhouette of a man. As a line is drawn, its origin is labeled with a large black dot, superimposed on the man's silhouette. The endpoint is labeled with a word that names the body part on which the starting point lies. Drawing becomes a labeling game: the line is signified by origin and identification. This sort of metaphoric line is outside of the scope of computational drawing, and approximates an interactive graphical piece, a reactive design.

ARTISTIC EXPERIENCE

How to qualify the drawings that are produced? Are they representative? Are they evocative? Are they expressive?

The present work with the code consists of non-representational, abstract compositions, relying heavily on the line and texture created with distinct computational lines as its main building blocks. The computational lines themselves rely on a close mapping of gesture to the vector of points that is drawn on the canvas.

This piece is created with a single line, a repeated up and down vertical movement, at varying speeds and horizontal positions. The line uses directional textures—collections of strokes arranged in different patterns—to generate shape and value. The movement of the pen achieves a desired tone, and the computer draws all the individual strokes.



In conducting experiments in what sort of creative, aesthetic, expressionistic experience is possible when using computation as art medium, we have to question whether it is possible to find the

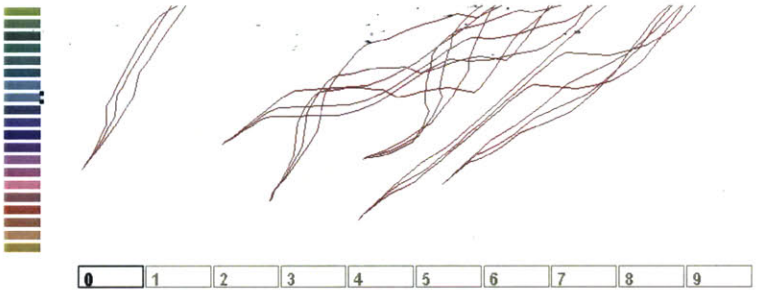
necessary "soul" in computation, considering that it is so incorporeal and immaterial. Can we ever achieve the same richness of expression using the computer as we do using the hands and the whole body. Every gesture is an extension of the movement through space, the gestural expression of the muscles and bones and tendons moving and reacting with the canvas.

Walter Bender has remarked that drawing is a closed-loop process: there is feedback between the hand and the eye. [BEND98b] Drawing is algorithmic in a very complex cognitive way. It can be loosely described as the process of the hand attempting to obey the eye, but not necessarily succeeding. Innovation is often a result of mistake, and mistake is perhaps a result of inspiration, a cognitive process so complex that it is often referred to as feeling and not reason. I want to underline the aspect of computational drawing that consists of drawing. Drawing is a process that cannot be planned out prior to execution and as such, the programming sets up an element of style, but only that. The hand and eye execute the rest.

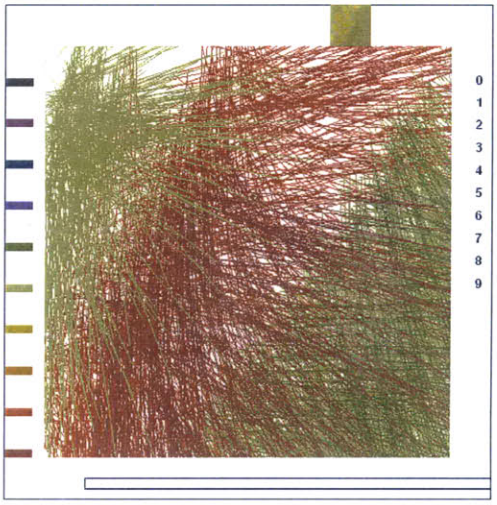
The algorithms are necessarily methodical, deterministic. The gestures can be slow and determined, to replicate a complete version of the algorithm slowly and faithfully, but can also convey a vast world of expressive possibilities. Repeated versions of the algorithm and the layering lines produce beautiful textures and shapes. A representational effect can be achieved, drawing with computational lines that allow a greater control over every pixel drawn. Since most of the lines involve abstract algorithms, it is difficult to draw representational shapes. The algorithm determines the style, and content, of the drawing. The hand provides character, expressiveness, and emotive content.

INTERFACE

Interaction between humans and computers hinges on the development and mutual understanding of new languages and communication models. These are referred to as interface. They involve the mapping of concepts and user actions onto elements of cultural and technological iconography, which are themselves representations of particular digital events. Interface metaphors must partake in and reflect the concept, the physical action and the digital event, as well as bridge the gap between the user and the machine. As such, interfaces often become complex and clumsy. Innovation in the area strives to reduce them to artifacts that are transparent and intuitive, in both a cultural and biological sense.



The interface to the Stream applet



Interface of the WHEAT applet

The interface to the drawing programs is very simple. One menu allows color choice, another menu offers different line choices, and the bottom offers an erase bar. The interface is minimalist and perhaps unclear, because it is created by the artist, to be used by the artist. The behavior of the lines is well known, and thus no further explanation is needed. The color palette is also chosen specifically for each set of lines. If the focus were more on the drawing applet, a more thought out study of interface issues would be necessary.

MODULAR TOOLS

Drawing with computation becomes a bipartite process. The artist programs her own tools and uses them. Two aspects are experimented with, the algorithm, and the subsequent gestures.

Two drawings with the JoeyGraphics class

Computational expressionism deals with the representation of a hand gesture on a two-dimensional canvas. I have noted before that the computational line possessed the added attributes of dynamism and interactive behavior, in addition to the expected attribute of appearance. The computational line is a Java composite of three properties, or components, one from each of the three categories, and these can be recombined to create an individual algorithmic style.

In the present definitions of lines, the artist has created combinations of attributes that she finds aesthetically interesting. The computational

artists who are reading this, and hope to utilize computational lines to gain a greater understanding of the computational medium, should change the parameters and manipulate the code to develop tools that best suit their style of expression. In order to make this easy, or at least in order to provide a plausible starting point for this endeavor, we have developed class libraries for appearance, dynamism and interactive behavior.

PHYSICAL INTERFACE

The physical interface to this system can initially be regarded as irrelevant. Computational drawing concerns itself with the gestures, the algorithms, and the relationship between them as displayed on the monitor. The drawing can be done with the hand, with the mouse, a pen or any other input device, traditional such as the keyboard or untraditional such as Tom White's bladders or Josh Smith's Fish sensors. [WHIT98] [SMI96] The results are displayed on a monitor. There exists an interesting issue, however, that results from the imposed physical distance between hand and eye focus. One of the reasons that drawing on a computer is a substantially different qualitative experience is the distance imposed by physical interfaces between cause and result.

The input device and the method of viewing the work (monitor, high resolution, print) are irrelevant in a philosophical sense. The issues are to become fluent in the language of computation as an expressive tool. One should talk about the methods, the environment, the procedures and the technology used, in order to situate the work. One can achieve a very different experience drawing directly on a display than drawing away from the display. To an inexperienced user, it is stunning to experience the computational line blooming from under the pen, as opposed to being controlled from a remote mouse movement or drawing palette.

Even though physical input devices will approximate more and more the feel of traditional creative tools, the exact feel, behavior and aesthetic of a certain pencil on a certain piece of paper cannot be duplicated. That this is precisely not the point of the present work. Computer applications are often concerned with replicating and improving something physical. Computational art should be about understanding the material and using it, in all its flaws and qualities.

What are the implications of the fact that the physical separation of tool and canvas that exists in traditional media does not exist here, that medium and tools are one? What are the temporal/qualitative differences of not being able to use more than one tool at once? Different applications will evolve from different working environments. If artists had at their disposal an advanced vision system so that they could waive their hands and record their movement precisely, a different sort of art would evolve.

COLOR THROUGH COMPOSITION

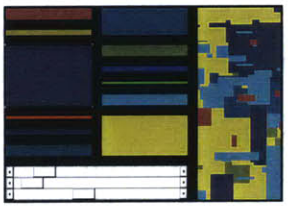
Computational expressionism is also concerned with color. In particular, an important question is how to integrate color into the design process. How to integrate color in a way that goes beyond picking a specific color to use for the drawing of an object.

The computational line has dynamism and behavior that may affect its color over time. This is one way of working outside the boundaries of traditional color selection. The program decides for you.

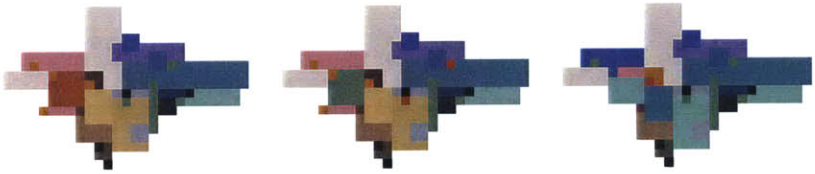
Another possibility is to give color properties to the canvas itself. In this example, the stroke of the line is chosen, but the color of the line depends on the pen's location on the canvas. The formula for color is a function of (x,y) coordinates. Shape and form become a secondary consideration, dependant upon the color that one wishes to use. This demands a careful examination of the dependencies inherent in the relationships between shape, line and color choices.

In an effort to ask designers and artists to better consider their choice of color, another set of tools was created to promote the use of color as a design tool, as opposed to using primarily line and shape. The first program was a drawing application where the toolbox is composed of color panels and scrollbars to help select the particular colors and color combinations.

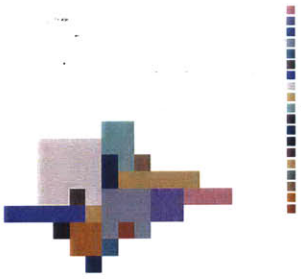
As you select and create colors, the composition is created. You have no control over the shapes or the placement of the color in the composition, but obtain an overall effect of the proportional color combinations and the color composition.



The color panels in the color toolbox are of different sizes. A color that we anticipate will be more often used in the eventual artwork should be selected into a larger panel. Already, an idea of the color composition is given by the arrangement of the color boxes. Additionally, there is a small canvas to the right of the color toolbox. When a color is selected, it appears randomly scattered on the canvas automatically without having to use a line or shape tool. When another color is selected, it also appears on the canvas, pushing some of the first color aside. The size of these areas of color is directly proportional to the size of the color panels from which the color originated.

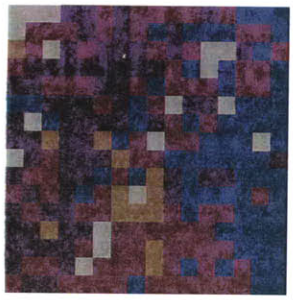


A set of color compositions that illustrate variations in palette selection.



A color composition applet by Joanna Berzowska

In this example, a color composition made up of rectangles of different sizes is provided. A palette of colors is selected in the right upper hand corner. As you select a color, you must use it in the composition. One drags colors from the palette and drops them on elements of the composition. They immediately assume that color. Each rectangle can store up to three different colors and cycle through them at the click of a mouse. This tool demands the consideration of the palette choices through making them fit into a fixed composition.



This example provides a canvas of squares, very large pixels whose color is fixed by rubbing them with a cursor that is dragging one of the special color squares from the upper right corner. It is a model of color pollination, of determining the color of a square through its spatial association with an action square. The low resolution of the display forces the consideration of color composition over shape, line, and realistic representation of objects.

The tool forces users to select color combinations before selecting the shapes that will be represented. Many designers do that already. They start out with palettes. But how do we help them choose better palettes?

How do the relationships between colors affect perception? Why are some color combinations differently evocative than others? And can this experience of the relationships among colors be translated across a color space without disturbing the evocative quality of the relationships? Inversely, can color be organized such that a mapping between a color combination and an evocative reaction will remain invariant under translation (along perpendicular axes) in the color space? [JAC96]

This application allows designers to choose combinations of colors that fit into one of four categories. Diagrams (icons) of a little person sleeping, waking up, falling down and leaping illustrate the categories. They consecutively describe monochrome, analogous, "beyond" analogous and complementary relationships. [JAC96]. The user selects one icon, and one color. A set of colors that includes the chosen color is chosen by the application reflecting the color combination described by the icon.

> Relationships between colors are, in many respects, universal, and thus relatively free from individual and cultural influences. The "experience of color" can be described objectively, so that predictable visual sensations can be elicited by adjusting the relationships among colors. A model of color experience is described that is based on the types of interactions among colors. [JAC96]

INTERFACE METAPHORS

We can distinguish between two kinds of intuitive interfaces. We can say that an interface is intuitive when it offers us a set of symbols lifted from cultural semiotics and the physical world. An example of this is the desktop metaphor in the Macintosh operating system. This sort of intuitive artifact can be described as culturally intuitive, because it refers to a set of mediated, learned iconography and behaviors.

We can also say that an interface is intuitive when it can be perceived or directly manipulated in an expressive, tactile, gestural way. This is how we use conventional physical tools. For instance, using gesture to delineate a selected area in PhotoShop, or using color to mark a progression in time or space. These sorts of intuitive interface elements I refer to as biologically intuitive, because they either demand to be experienced and manipulated directly, and exhibit a pattern easy to learn and assimilate, or they offer a clear mapping to physical gesture, metaphor-less. Basic concepts like front/back, movement, stillness, color, speed, whether they are instinctive or learned, have very little to do with the mediated society we function in. These intuitive interfaces should be able to span across different cultures and educational backgrounds. [LAK87]

It is important to note that the use of metaphor has different significance whether it is used in a tool or in an art piece. Metaphors usually enrich the perception of a work of art. Interaction between systems of images and models of representation demands a new perspective on a work, can engender meaning which reflects the multitude and the superposition of interpretations and adds a whole world of contextual meaning. The tool application attempts to make metaphors useful and transparent the art piece tries to make them revealing.

An interesting research direction is not to reduce or simplify the interface metaphors, but to formalize them into new languages and explore their effect on creative expression and cognitive perception. Explicit formal mappings between distinct vocabularies and grammars facilitate the creation of expressive works. Instead of attempting to hide the interface elements, and assuming that the particular mapping used is irrelevant to the resulting work, we should place the emphasis on the mapping and exploit its role in producing content. [BER97]

Velcro Dreams is a project that uses the vocabulary of a paint program in order to generate poetry. [BER97] It is a Java applet with a custom painting and color selection interface. Salient features (color, shape, position and texture) are extracted from an image. A loose grammar of poetry is used to generate text based on the visual variables. *Velcro Dreams* uses an overt mapping of color and shape to words and textual structure. [BER97]

A screenshot from the Velcro Dreams project

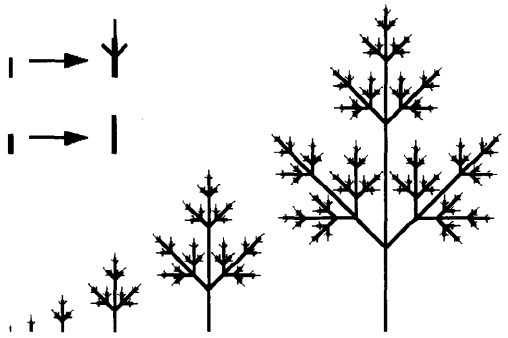
The system is responsive and dynamic, so the text evolves in conjunction with the evolution of the image. The mapping is intuitive to a certain point, applying concepts of color theory, but also allows a degree of non-determinism and variation between several instances of processing identical images. The process becomes a new language with which the user and computer can communicate, emphasizing the collaboration between an artist and a set of digital tools in the creation of poetic expression. [BER97]

EMERGENCE

Connectivity, interaction and emergence are becoming the watchwords of artistic culture. [ASC93] Art no longer is a window onto the world but a doorway through which media and processes enter a world of interaction and transformation.

DEFINITION

For our hierarchical mentality, complex objects intuitively imply a creator's hand. Explanations attempted by folk science for intricate or perplexing phenomena often revolve around supernatural causality or other hegemony. In this thread of thought, religions have postulated the existence of god, and modern authority figures, mass media, speculate extraterrestrial life and paranormal phenomena. The mythical has been replaced by the supernatural, but these objects and events that we struggle to explain are often caused by something much simpler, by the evolution or collective actions of simple rules of biology, physics or chemistry.



Developmental model of a compound leaf, modeled as a configuration of apices and internodes.

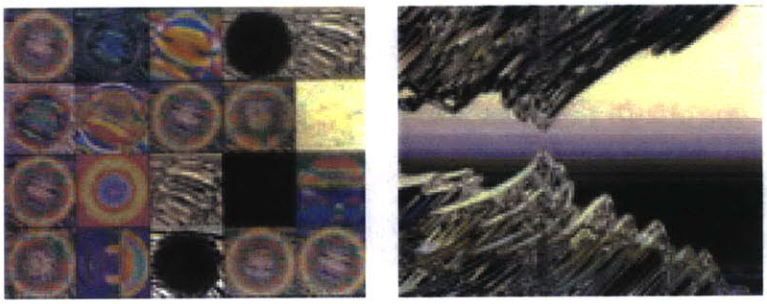
> The figure above illustrates the development of a stylized compound leaf including two module types, the apices (represented by thin lines) and the internodes (thick lines). An apex yields a structure that consists of two internodes, two lateral apices, and a replica of the main apex. An internode elongates by a constant scaling factor. In spite of the simplicity of these rules, an intricate branching structure develops from a single apex over a number of derivation steps. The fractal geometry of this structure can be viewed as an emergent property of the rewriting rules. [PRU98]

The leaf of a fern, the rotationally symmetric petals of a daisy, the choreographed flight of a flock of birds, or the food gathering practices of a colony of ants are some examples of the beautiful and surprisingly complex patterns and behaviors that can emerge from simple rules or interactions. The human being, product of many generations of evolution, is another example of emergence. "Although evolution didn't create the watch, it did create the watchmaker, a much more complex and beautiful result." [RES97]

Some of the essential ideas developed by the Epistemology and Learning group at the Media Laboratory deal with emergent behavior as a way of explaining complex phenomena. Emergence is an important aspect of understanding decentralized thinking and how intricate behaviors and patterns can result from a simple set of rules. This is used to teach children about geometry, arithmetic, scientific models, programming, and much more. [RES95]

Emergence is a process in which a collection of interacting units, simple rules or behaviors, generates or acquires qualitatively new properties that cannot be reduced to a simple superposition of individual contributions. Studies of emergence are among the central themes of Artificial Life, "the study of man-made systems that exhibit behaviors characteristic of natural living systems". [LAN87]

Karl Sims uses evolutionary techniques of variation and selection to create complex simulated structures, textures, and motions for use in computer graphics and animation. [SIM91] In the example illustrated below, he employs symbolic Lisp expressions to create two-dimensional compositions. An image's genotype, the symbolic expression, is used to spawn a set of offspring, using some degree of random mutation in the genotype that changes each child's appearance. He uses concepts of natural selection based on visual perception (a user manually selects the most engaging image) to choose among the set of procedurally generated offspring. This process allows the user of his system to artificially evolve complex compositions, with a minimum of effort and without the burden of having to understand the underlying expressions. [SIM91]



A set of 19 random mutations and an artificially evolved image from Karl Sims' system.

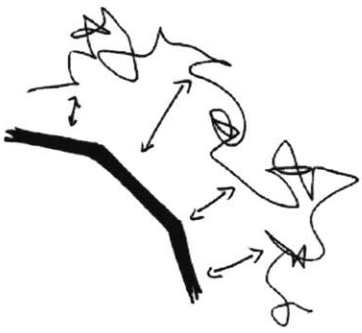
This iterative process of repeated variation and user selection has similarities to natural biological evolution of organisms. An alternative interpretation might be that the computer attempts to learn from the user about how to produce pleasing images. The user provides a decision for preferred images, and the computer reacts by generating a set of similar, related images. [SIM91]

The concept of emergence has several applications to the making and the consumption of computational and interactive art. In his interactive installation, *What Will Remain of These*, Dodge [DOD97] creates animations of particles whose behavior emulates the emergent motion patterns of the large collective audience that has walked through the installation surveillance space. Ascott [ASC93] writes that interaction and emergence are now a deeply relevant part of artistic culture. The viewers or consumers of art are moving towards the center of the creative process, from the periphery that they used to occupy. Similarly, Bill Seaman believes that what he calls recombinant poetic networks embody "the metaphor of recombinant DNA via the

computer-mediated recombination of operative poetic elements." [SEA97b] The aesthetic experience emerges from interactive construction mechanisms and the navigation and exploration of content elements by the viewer.

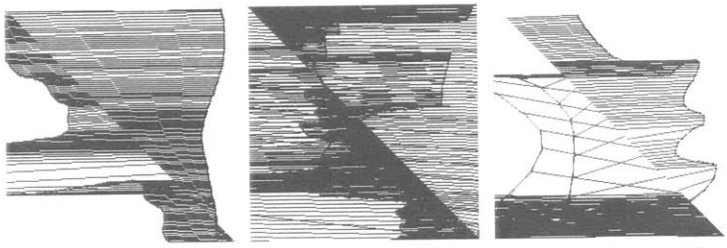
THIRD ATTRIBUTE: BEHAVIOR

The concept of emergence and emergent behavior for computational drawing is introduced through simple examples. Even more so than with dynamism, it becomes increasingly difficult to think of a computational line as a composition element for the creation of static images. The interaction and animation becomes engrossing and more aesthetically engaging than the final piece. The focus of the creative process moves away from the desire to create a drawing and becomes more of an engaging interactive process, which has been described by Scott Snibbe as an instrument. [SNI98] We play with the lines not to achieve a final image, but to experience the pleasure of affecting a dynamic, responsive environment. Just as it is pleasurable to elicit sound out of a musical instrument, we take great pleasure in extort movement and reactions from the computational lines. The process, the experience, is of interest here.

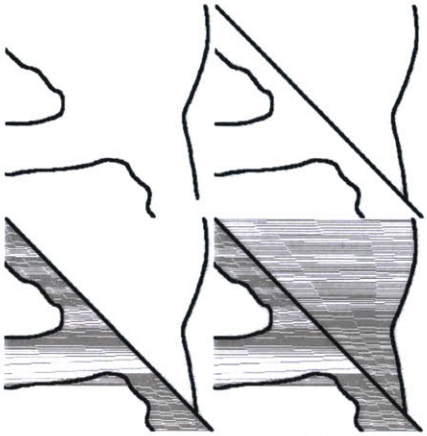


In this work, emergence is evident in two capacities: in the creation of each computational line and in the interactive generation of the visual pieces with the computational lines. The first process involves a series of steps akin to the artificial evolution of Karl Sims' work. [SIM 91] The second process relies on each computational line affecting its neighbors on the canvas. Simple for animation and behavior of each line produce a more complex visual result.

The final algorithms for computational lines emerge (although a better word here might be result) from the iterative, evaluative process of writing code and testing lines. This is an evolving process, with the artist's eye serving as fitness function. The creative process has already been described as being more evaluative than generative. The programming of a computational line consists of making small changes to its algorithms, compiling and running the drawing program, and testing the results visually. The changes that are made are often an arbitrary tweaking of numbers and mathematical functions. One does not need to fully understand the algorithm to produce it. Sketching with computation means many repetitive attempts to modify existing code, without fully anticipating the outcome. Thus, the process of creating a computational line can be described as an example of emergence.



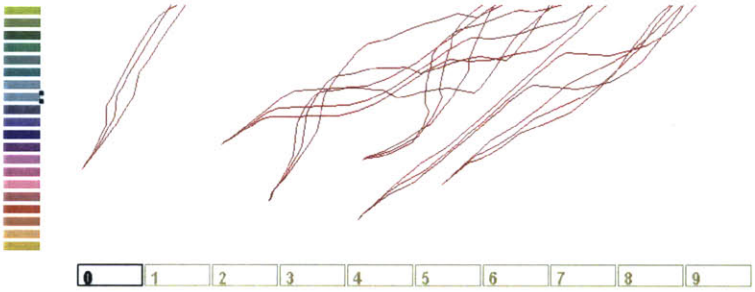
Stills drawn with the symmetric line



Symmetric Line process

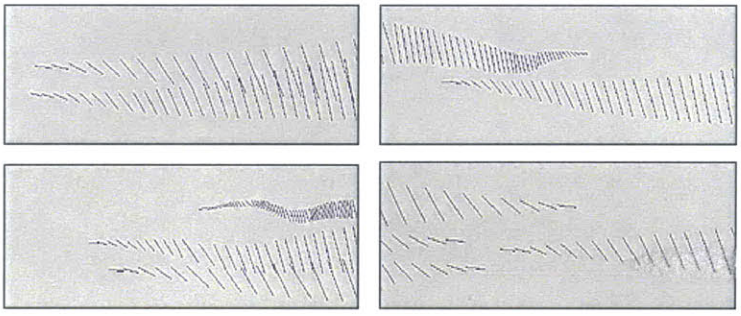
The symmetric line (examples shown above) is an example of what has been called a computational sketching mistake. It was a mistake, a typo, made during the programming stage of the process. The simple transposition of the letter x for y produced this interesting visual form. The function draws lines between the point (x_n,y_n) to the point (x_{n-1},x_{n-1}) instead of (x_{n-1},y_{n-1}). The resulting drawings allow the creation of symmetric shapes through the drawing of the outline. The procedure is illustrated on the left. The upper left panel shows the gesture that is actually drawn with the hand..

The computational sketching mistake is an example of the first category of emergence within this work. The second is the emergence that governs composition, as computational lines interact with one another to produce visuals that are more complex. In the next example, the Stream applet, the vertex of each set of flowing lines follows the movement of the cursor. The lines respond to each other's presence with a simplified physics model, they sense proximity and push each other away. The compositions emerge from the interaction among the different dynamic behaviors of individual lines.



In the Stream applet, a dynamic set of interactive lines responds to the direction of nearby lines. After some time, all the lines move "downstream".

The next example shows a different line, called REED, within the Stream applet. This applet is one of the few drawing programs created for this thesis where the dynamic behavior of the lines has no temporal limit. Their motion does slow with time, but there is no such thing as a final static state of the composition. The REED lines are pulled away from the drawing gesture and swich direction over time. The gesture is followed by the apex of each REED line, but their ondulating, repetitive body follows and amplifies the gesture in constant motion.



The REED line in the Streams applet by Joanna Berzowska

discussion

FUTURE WORK

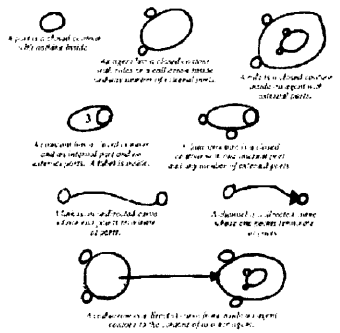
PROGRAMMING LANGUAGES FOR ARTISTS

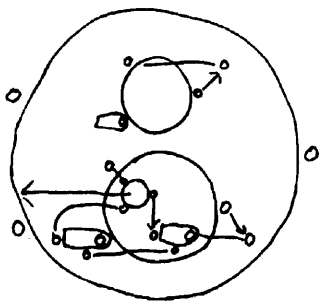
The code is a set of computational drawing tools and environments that enable me to make the kinds of drawings that I want to make. In my definition of computational lines, I have created combinations of attributes that I find aesthetically interesting. The computational artist should alter these, and manipulate the code at will, to develop tools that best suit her style of expression. To facilitate this task, or at least to provide a possible starting point for this endeavor, I have developed annotated class libraries for appearance, dynamism and behavior.

One focus of the project is to make a library of classes that will encourage other artists to graphically express themselves using computation. Another focus is to construct a learning environment for programming, where graphical objects, drawings, can be written quickly and easily in Java. Both of these will have to be developed further.

John Maeda's forthcoming book, Design by Numbers, introduces a Java based programming system freely available on the web. "The emphasis throughout the book is to build an understanding of the motivation behind computer programming, as well as the many wonders that emerge from effectively written programs." [MAE98] The language is designed for artists and designers, it has few commands and is quite simple in comparison with other languages such as C, Java or Lisp. The aim is to help the artist or designer overcome the initial transition of thinking in a procedural way, and be prepared to begin learning other programming languages. [MAE98]

Painting with Interactive Pixels is a tool that enables interaction designers to construct graphical user interfaces by painting with pixels that possess inherent behavior. Each pixel is an interactive agent communicating with its neighboring agents. The designer constructs interactive behavior directly at the pixel level with tools resembling those in paint programs like MacPaint. [SVA97]





Pictorial Janus

Pictorial Janus is a language in which the syntax is defined in terms of the topological relations between picture elements. Relations like "inside," "touching," and "connected" are used in defining the syntax of the language. Shape, color, size, and texture are left for programmers to use as they see fit. Programs can be drawn on paper, scanned in, and parsed or constructed using an illustration program. [KAH96]

The primary role of syntax in programming languages is to aid in the communication of a program from a human to a computer. [KAH96] Much of the debate about whether visual programming is better than text-only programming and the debates about which visual programming language is better are really debates about human psychology. [KAH96] The greater issue of the role of visual thinking in math, logic, and science has been studied by psychologists for more than a century. [KAH96]

GENETIC ALGORITHMS

Karl Sims has demonstrated the creative potential of genetic algorithms and artificial evolution techniques. By using various selection criteria, he has programmed engines that evolved abstract two-dimensional compositions and three-dimensional creatures. [SIM91] [SIM94] Genetic algorithms are a specific method for generating a set of offspring from a parent population, by introducing little variations in their genetic code. [HOL98]

Lines with behavior that are aware of each other on a canvas can employ selection criteria and use evolution theory to spawn offspring. Slight variations in algorithms can be explored as lines replicate themselves and grow and change as compositions are created.

DRAWING INTERFACES

Drawing interfaces refer to gestural, expressive approaches to information design, interfaces that are choreographed and metaphor-full or tactile and physically intuitive.

Gesture has been successfully employed as a communication tool, both as formal sign language, and in interpretive dance, theatre, mime and other performance art forms. Drawing, as an abstraction or application of gesture, is also a very powerful expressive language. As such it has much potential as a control mechanism or physical interface to computation. Also, it is easier for present computer systems to recognize and analyze gestures when they are drawn upon a surface rather than performed in space. This has been illustrated by writing recognition systems such as the Palm Pilot and the Apple Newton.

Animating images with drawings "extends the power of 2D animation with a form of texture mapping conveniently controlled by line drawings. By tracing points, line segments, spline curves, or filled regions on an image, the animator defines features which can be used to animate the image. Animations of the control features deform the image smoothly." [LIT94] Peter Cho's Datapaint combines freehand drawing with gestural navigation. It involves using freehand sketching as a method for human-computer interaction, as a navigational and expressive means of interacting with the computer. [CHO98]

Movement and, more specifically, movement of the hand to make lines on a canvas can replace pressing on buttons and invoking discrete choices. Gestural drawing tools can be very useful in performing tasks such as authoring or browsing of content. The direct manipulative powers of tracing on a surface are already employed when organizing files on a desktop, for instance. Gesture is used to drag icons, and so allows the user to organize files in directories. The drag-and-drop mechanism is very much a drawing interface.

A close relationship with content can be achieved using elements of traditional drawing and painting, so that users perceive and express with line, shape, color, dynamism and other vocabulary from the art world. Searches on the web, for example, can be refined using gestural underlining or crossing out of search terms, as opposed to the conventional check-boxes.

CONCLUSION

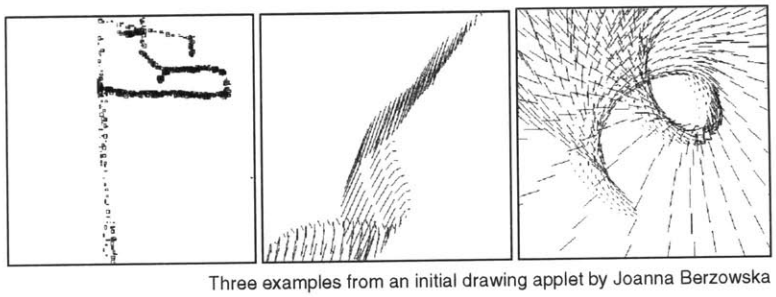
CLASSIFICATIONS

The computational line is an evolved form of what is traditionally thought of as a line, through the incorporation of some elements of computation. The qualities of computation that inform the present definition of computational drawing are algorithm, dynamism, interactivity and emergent behavior. The line can:

1. Display self – Algorithmic Appearance
2. Animate self – Dynamism
3. React to actions – Interaction
4. React to objects – Behavior

The following axes are useful when thinking of classifications of possible computational lines. Each pair of terms offers a method of situating a particular computational line.

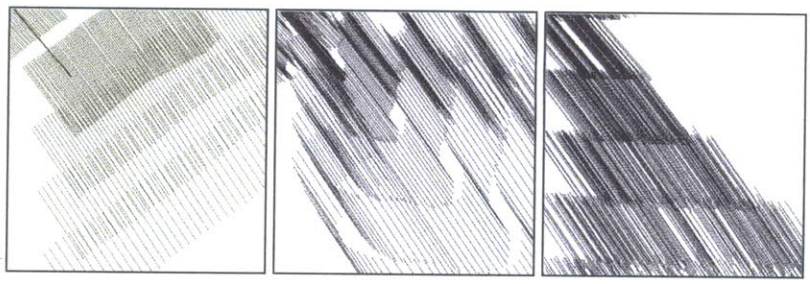
DIRECT – ABSTRACTED



Three examples from an initial drawing applet by Joanna Berzowska

Mappings of hand movement to graphical representation are presented in increasing levels of abstraction, from a direct line to a more abstract (or spatially complex) line. The most direct representation is simply the single thin line that accurately follows the hand's gesture. Abstraction or complexity increases as mappings that are more elaborate are devised. These axes can also be described as ranging from a more literal to a metaphorical representation, where an example of a metaphorical line is one that manipulates the entire color composition of the canvas through gestural input.

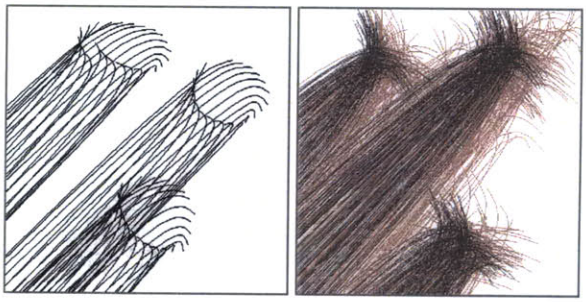
LOCALIZED – GENERAL



A set of sketches produced with a single computational line.

One gesture can produce markings close to where the cursor has traveled, but also in other areas of the canvas. The first panel shows a straight line gesture that generates both a tracing of the straight line (the outermost boundary of the shape in the lower right corner) and a pattern, dependant on the gesture, yet filling the whole canvas. The second panel shows a fluid gesture. It keeps the feeling of fluidity, but augments it with a regular pattern of parallel lines. The last one shows the sorts of shapes that can be drawn when the gesture is restricted horizontally or vertically. The single computational line can produce lines and shapes, locally, or covering the whole canvas.

ORDERED – VARIABLE



The WHEAT line.

A computational line can be represented as a very regular geometric pattern, or used as a textural tool to produce areas of color. To produce the first quality, the gesture must be slow and deliberate, for the second, the gesture must be faster so as to create texture, tone and shape.

STATIC – DYNAMIC

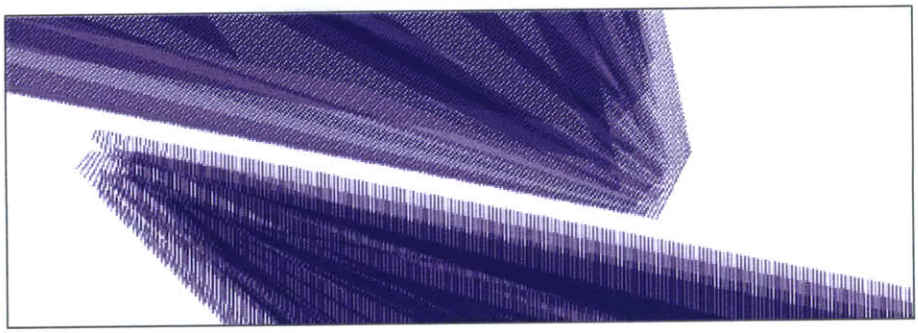
The GRIDS line, the Hairy line and a behavioral line.

The first panel illustrates an algorithmic line that, as it is drawn, has a more complex (or abstract) appearance. Once it is drawn, it does not move, change color, or animate in any other way. The middle panel shows the Hairy Line, which embodies a dynamic drawing process. Its animation plays itself out and then stops in an expected final state. In

the third panel, parallel line segments originate from the drawn line and over time redraw over one another, changing the colors of nearby lines and generating shapes made up of color areas. The lines appear to be moving upwards, adding an interesting element of three-dimensionality. This animation is longer, and more involved. The various lines of different colors interact with one another, it is an appealing process, even though the final outcome is less interesting.

Computational lines are used to design dynamic, interactive experiences whose end product is a static drawing. It is very difficult to separate the beauty of dynamic interaction from that of a finished piece.

PERSISTENT – FLEETING and IMMEDIATE – LATENT



Drawn by Joanna Berzowska with the DynamicExpress package

The example above illustrates a line that, once it is drawn, continues to animate. The marks that it has made on the canvas remain drawn, and the finished look is the composite of all its positions in time. Another model I have tested, which approximates animation, erases the canvas between each consecutive redrawing of the line's new position. Both of these models are interesting, albeit the first is closer to a drawing process, whereas the second is closer to an interactive animation tool. Should the dynamism be immediate, or can the line have some hidden attributes that only become apparent after it has been drawn? Can the expressive quality of a dynamic line be re-experienced through history?

EVALUATION

This work has been evaluated in the art critique tradition, through a collaborative consideration of and inquiry into the tools and images created. At each step of the programming and drawing process, the reviewers asked questions about components of the work, both technical and aesthetic. Debate, predicated and informed by diverse human experience, is a good method of evaluation for this type of creative work as opposed to a quantitative approach. Individual critiques consisted of the examination of drawn compositions and an explanation of the concepts they illustrated as well as the underlying mechanisms. We all brought different ways of modeling aesthetic representation, different skills, and different spirit and humor to the discussion. I was asked to concentrate less of the actual drawing tools, and focus on the development of abstract forms, illustrations of my vision of computational drawing. I was asked to develop a personal algorithmic style, and to rigorously examine what sort of artwork I was striving to produce, and how my expectations were challenged.

My computational lines and my drawings are only one example of what could have happened in the exploration of computation as an art

medium. How can they be described as my personal expression? It is very interesting that aesthetically, this work is very different from my previous artwork in traditional media.

Andy and David, 1989. by Joanna Berzowska

The paintings presented here were painted between 1989 and 1998. The earlier work is oils on canvas and the later is acrylic on canvas. The self-portrait is drawn with charcoal on paper. The work draws heavily upon the tradition of portraiture, and relies on the expressionism of human forms as content.

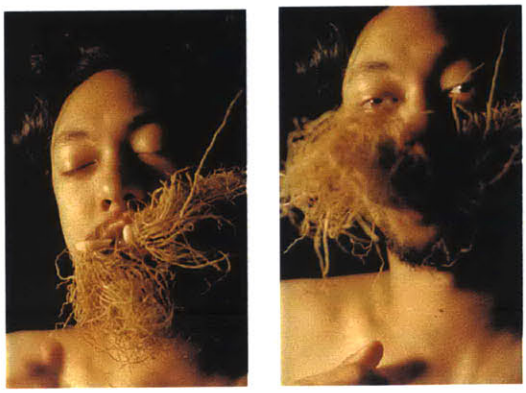
Self-portrait, 1991

Kathrine, 1990. by Joanna Berzowska

This previous work shows a clear preoccupation with expression as embodied in human faces and the composition of, and relationships between, different human bodies. Meaning is derived less from stylistic constraints than from the arrangement of human features and bodies. The gaze, the formal composition of space and an aspect of discomfort created through implied meanings and readings are the crucial aspects of the expression. Style is an important, but by no means exclusive, component of the visual experience.

Regret, 1998. by Joanna Berzowska

My work with photography deals exclusively with portraiture, the introspective qualities of human representation. The five-panel painting, Regret, owes its striking appearance to the multitude of story interpretations possibilities presented to the audience, as they are faced with this array of human forms.



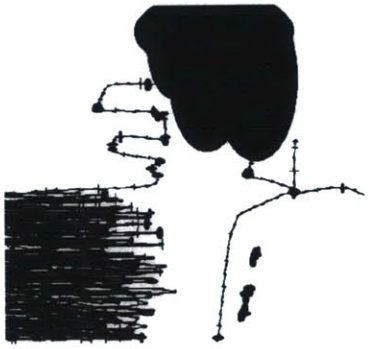
Portraits of Paul, 1998. by Joanna Berzowska

The computational medium for me is vastly different. Algorithmic lines do not lend themselves well to representational images. In particular, the human form is difficult to reproduce and interpret.

Girl drawn with String Art line

The drawing on the left is a quick sketch of a woman done with a computational line reminiscent of string art from the 1970's. This line is what I call a very direct line, as it accurately traces the movement of the hand with only minimal embellishment. Overall, this drawing is neither evocative of human expression, nor computationally interesting.

On the other hand, representational art need not be realistic. A high level of stylistic abstraction is possible in the replication of human form and expression. The panel on the right shows a highly abstracted figure drawn with the Calligraphy1 line, which produces line and shape out of the temporal characteristics of a gesture. A quickly drawn line is represented as a ragged linear form. A pause in the drawing process or a slower gesture produce shape artifacts whose size is proportional to sluggishness.



Calligraphy1 woman

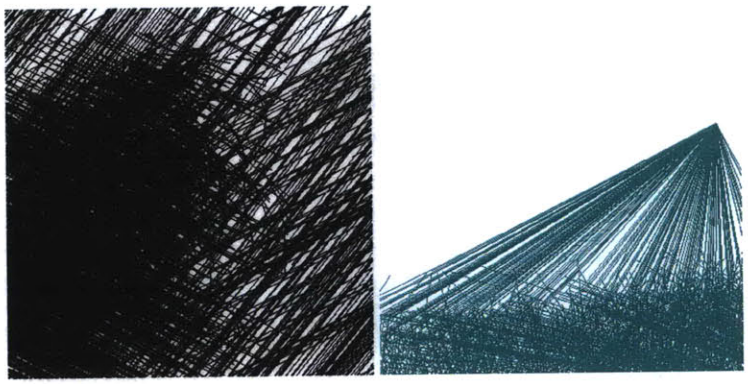
In the second image, the apposition of anthropomorphic cues and computational aesthetic produces a more intriguing image. The composition is interesting because it blends human form with a more

involved algorithmic drawing process. The algorithm, however, is very specifically suited for the task. It generates both direct linear forms that can define precise boundaries, and solid shapes of various sizes and orientations. The two can easily be combined to form compositions that appeal to our need to anthropomorphize our surroundings.

In general, the aesthetic qualities of an algorithmic drawing or a dynamic interaction far outweigh representational potential. Thus, my work in computational drawing has been almost exclusively non-representational. The quality of lines and the abstract composition of areas of varying tones and textures in a constrained space are the main aesthetic themes.

The most exciting aesthetic component for me is the apposition of regular, mathematical and clearly algorithmic patterns with more gestural shapes and areas of tone and texture. My favorite compositions are those that use regular algorithms and impose qualities of variation and irregularity to their representation. The combination of mathematical order and gestural disorder is fundamentally enticing to me, perhaps because of my dual training in pure mathematics and fine arts. What is most exciting is that both aesthetic qualities can be produced with a single computational line. They are merely a function of the manner in which the line is used. I am showing off the unique qualities of computation to produce such visual constructions.

I am interested in computational drawing primarily because of the gestural input, which I regard as a very powerful expressive device. Drawing is different from painting insofar as it is based on continuous line paths as opposed to continuous surfaces. Textures and tones are achieved through a layering of lines. The sharp contrasts of linear forms and the expressive potential of gesture are very appealing to me.



Two images that illustrate the combination of mathematical order and gestural disorder.

The drawings are mostly abstract and non-representational, in particular as the computational line becomes a more complex mapping of gesture to illustration. The algorithm as an element of style overpowers the representation. Style becomes content. This illustrates, according to Danto, an end-of-art condition characterized by the fact that style becomes subject matter, and hence is something shown rather than used. [DAN97] This raises questions of how and whether this work can be considered art.

How can we discuss interactive digital artwork, when we consider its overwhelming accessibility, longevity, and its potential for digital re-appropriation and endless duplication (which are not qualities that have historically been associated with high art)?

> Our relationship with mass-produced goods has changed and also with the products of 'high' art. Differences have been reduced, or erased; but along with the differences, temporal differences have been distorted, the lines of reproduction, the befores and the afters. [It is] the development of the arts itself, today, that tries to obliterate this distinction. [ECO90]

An artwork can be defined as an "expression" because it is caused by a feeling or an emotion on the part of its maker, and which it in fact expresses. [DAN81] So art is differentiated from a mere thing by the order of its mental cause, intention, feeling expressed. However, what about other emotive products like tears? Are these artworks? [DAN81] Art has at other times been described as mimesis, as a mirror held up to nature in the tradition of representational art. [DAN81] This definition is similarly inadequate in the case of contemporary art. The artwork has also been described as design without constraints, or something that lacks explicit function but possesses form. The postmodern concept of intentionality has been advanced as a possible definition. If the artist intends to make art, it is art.

All these are at best problematic and it is not clear to me whether I should be concerned with the question of whether my work is regarded as art. A better question is how to disseminate it, and of what value it can be to others, both audience and other artists.

A definition of whether and how digital work is perceived as art suggested by Scott Snibbe raises the question of venue. He writes, "How and where a work is presented radically alters its perception. This is why I control the presentation of my work - engaging in a gallery or a talk creates a type of attention and thought impossible when something is in the low attention arena of the web or freeware." [SNI98b]

Where is my work to be displayed, in what format, medium or size? Is it to be shown in a gallery, or experienced momentarily on the monitor and then deleted? How is the quality affected by the resolution of the display device? How would stylistic content be affected by a change of venue? It is clear that a purposeful showing is preferable to an unstructured distribution, but what do I show? One immediate answer is that a gallery showing of printed drawings would be essentially different from an installation that allows users to interact with the computational lines that I have created, and draw images with them. The question is whether I should show this work in a printed format, or on a computer display. Should I exhibit the finished drawings or the models of interaction?

I maintain that the salient aspect of computational drawing is the finished piece. If I want to stick to Scott Snibbe's advice to experience work in its proper form, I would have to insist on showing finished drawings. The drawback is that it becomes impossible to judge the quality of the dynamic pieces. Snibbe writes, "Often the good dynamic pieces look terrible in a still image and vice versa." [SNI98b]

The real question here is what exactly is the art form. Is it the computational lines that I build, is it the body of drawings that I make with my tools, or is it the process of combining computation and drawing? Since I believe that the artist needs to program her own computational lines and then draw with them in order to access the sort of creativity I am speaking about, it is very important that each artist

programs her own computational lines. I am weary of using the terms "user" and "tools" because I view the drawing process as composed of two parts, the programming and the drawing. As such, the computational lines I have written are not tools, simply one portion of my own drawings.

How can I best apply what I have produced to other artists? Since I do not view my computational lines as tools but as an intrinsic part of the drawing process, I would feel uncomfortable displaying my own drawing applets for use by others, and calling this process the showing of art. I have no objection, however, in showing them, and distributing the code, so that other artists can understand the process and begin to imitate me.

How important is it for the artist to do all of the programming? What is sacrificed if the artist becomes too involved in the technology? Scott Snibbe writes, "As artists become programmers, the problems of distraction come about. One can become obsessed with the technical details and the mechanical goal of 'making it work' rather than the aesthetic potential. I find that it takes a strong mind shift to go from tool builder to user. In fact, it's better to separate these phases by large amounts of time to come at the tool freshly as a user." [SNI98b]

What is the distinction between the drawing and the programming? How is the intentionality in the creative process divided? How much of the aesthetic quality of a final composition is a product of the algorithms, and how much depends on the gesture? The answers to these questions depend strongly on the particular example. A generalization I will venture to slip by is that the algorithm injects more of the stylistic elements, whereas the gesture controls expressive elements to a greater degree. That is why I have chosen to concentrate on computational drawing as opposed to rendered algorithmic art, or purely interactive art. It is vital to me that static compositions are created by interacting with the algorithms through gestural input. The process is really one of drawing.

As a new medium, computation offers enormous potential, but its exploration is progressing slowly due to a lack of understanding of the technology by many artists and to the aesthetic limitations of many engineers and technologists. My battle cry echoes that of John Maeda: computational artists must learn to program, in order to gain a more meaningful understanding of the medium.

reference

[ABE97] Abe, Yoshiyuki. WRO97 Media Art Biennale Catalogue. Wroclaw, Poland. 1997

[ASC93] Ascott, Roy. From Appearance to Apparition: Communications and Consciousness in the Cybersphere, In *Leonardo Electronic Almanac*. Cambridge, Massachusetts: The MIT Press, 1993.

[BAI97] Tilting at a Dreamer's Windmills: Gesture-Based Constructivist Interaction with Character. In *Consciousness Reframed: Art and Consciousness in the Post-Biological Era*. First International CAIIA Conference. Newport, Wales. 1997.

[BAR97] Barrett, Cyril. Kinetic Art. In *Concepts of Modern Art*, ed. By Nikos Stangos. London: Thames and Hudson. 1997.

[BEND97] Bender, Walter. Personal web site. <http://nif.media.mit.edu/people/walter/> [accessed 02 October 1998]

[BEND98] Bender, Walter. Web site for the Electronic Publishing Group at the MIT Media Lab. <http://nif.www.media.mit.edu/ep/> [accessed 09 September 1998]

[BEND98b] Email exchange with Walter Bender in September 1998.

[BEN92] Benjamin, Walter. The Work of Art in the Age of Mechanical Reproduction. In *Film Theory and Criticism,* ed. G. Mast, M. Cohen and L. Braudy, 665-681. New York: Oxford university Press, 1992.

[BER97] Berzowska, Joanna. Velcro Dreams: a Paint Interface to Poetry. In *Consciousness Reframed: Art and Consciousness in the Post-Biological Era*. First International CAIIA Conference. Newport, Wales. 1997.

[BIR97] Birkerts, Sven and Murray, Janet H. Digital Storytelling: Is it Art? *HotWired: Synapse - Brain Tennis.* <http://www.hotwired.com/synapse/braintennis/97/31/index0a.html> [accessed 09 July 1997]

[BOLZ93] Bolz, Dieter. Some aspects of the user interface of a knowledge based beautifier for drawings. In *Proceedings of The International Workshop On Intelligent User Interfaces*, 45-52. 1993.

[BRI98] *Encyclopedia Britannica Online* <http://www.eb.com> [accessed 10 October 1998]

[BRO98] Brooks, Sawad. *Lapses and Erasures.* <http://www.thing.net/~sawad/erase/trait/text.html> [accessed 10 July 1998]

[BRO97] Brooks, Sawad. *Technology in the 1990s.* <http://www.tech90s.net/sb/> [accessed 10 July 1998]

[COH95] Cohen, Harold. The Further Exploits Of AARON, Painter. In *Stanford Humanities Review. Constructions of the Mind: Artificial Intelligence and the Humanities*. ed. Stefano Franchi and Güven Güzeldere. Vol 4, issue 2, Spring 1995

[COY95] Coyne, Richard. Designing Information Technology in the Postmodern Age: From *Method to Metaphor*. Cambridge, Massachusetts: The MIT Press, 1995.

[CRE98] Creature House Computer Graphics Research and Development. <http://www.creaturehouse.com/> [accessed 27 September 1998]

[DAN81] Danto, Arthur C. The Transfiguration of the Commonplace: A Philosophy of Art. Cambridge, Massachusetts: Harvard University Press, 1981.

[DAN97] Danto, Arthur C. Criticism, Advocacy, and the End-of-Art Condition: A Working Paper. In *Artnet Magazine.* <http://www.artnet.com/magazine/features/danto/danto3-6-97.html> [accessed 09 December 1997]

[DEH98] Dehlinger, Hans E. *25th Anniversary Celebration of Pioneering Computer Artists.* ACM SIGGRAPH '98 Conference Proceedings. 1998.

[DEN95] Denning, James. Palsy for Your Printer. In *Wired* Issue 3.01, January 1995.

[DEN98] Dennett, Daniel C. *Brainchildren.* Cambridge, Massachusetts: The MIT Press, 1998

[DOD97] Dodge, Christopher. The Abstracted Process: Providing for Consistent Metaphors between Content and Computation in *Interactive Media Art.* WRO 97 Media Art Biennale paper. <http://liquid-sky.media.mit.edu/cdodge/papers/abstractedprocess.html> [accessed 19 September 1998]

[DRU96] Druckery, Timothy, ed. *Electronic Culture: Technology and Visual Representation.* New York: Aperture Foundations Inc, 1996.

[ECO93] Eco, Umberto. Make Your Own Movie. In *Misreadings*, 145-155. London: Jonathan Cape, 1993.

[ECO90] Eco, Umberto. The Multiplication of the Media. In *Travels in Hyper Reality*, 145-150. New York: Harcourt Brace & Company, 1990.

[EVA98] Evans, Brian. *Artist's Statement.* <http://www.vanderbilt.edu/VUCC/Misc/Art1/statement.html> [accessed 1 October 1998]

[FOU91] Foucault, Michel. What is an Author? In *Rethinking Popular Culture*, ed. C. Mukerji and M. Schudson, 446-464. Berkeley: University of California Press, 1991.

[HAE90] Haeberli, Paul. Paint by Numbers: Abstract Image Representations, In *Computer Graphics* (ACM SIGGRAPH '90 Conference Proceedings), 207-214. 1990.

[HAL98] Halaby, Samia. *Artist's Statement.* <http://www.911gallery.org/samia/kinetic.html> [accessed 1 October 1998]

[HAR97] Harrison, Charles. Abstract Expressionism. In *Concepts of Modern Art*, ed. By Nikos Stangos. London: Thames and Hudson. 1997.

[HEB97] Hebert, Jean Pierre. *Artist Presentation Session #2.* SIGGRAPH '97 Art Exhibition: Ongoings. Leonardo On-Line:

Siggraph 97 <http://mitpress.mit.edu/e-journals/Leonardo/isast/articles/SIGGRAPH97panel/Panel_Two_B/Panel_Twonew.html> [accessed 1 October 1998]

[HER98] Hertzmann, Aaron. Painterly Rendering with Curved Brush Strokes of Multiple Sizes. In *Computer Graphics* (ACM SIGGRAPH '98 Conference Proceedings), 453-460. 1998.

[HOL95] Holtzman, Steven R. *Digital Mantras*. Cambridge, Massachusetts: The MIT Press, 1995.

[HOL98] Holland, John. Emergence. Reading Massachusetts: Addison-Wesley Publishing Company, 1998.

[HSU93] Hsu, S. C. Lee, I. H. H. and Wiseman, N. E. Skeletal Strokes. In *Proceedings of the 6th Annual Symposium on User Interface Software and Technology* (UIST'93), 197-206. 1993.

[HSU94] Hsu, Siu Chi and Lee, Irene H. H. Drawing and Animation using skeletal strokes. In *Computer Graphics* (ACM SIGGRAPH '94 Conference Proceedings), 109-118. 1994.

[JAC96] Jacobson, Nat, and Bender, Walter. Color as a Determined Communication. In *IBM Systems Journal*, Vol. 35, Issue 3&4, 1996.

[KAH96] Kahn, Ken. Drawings on Napkins, Video-Game Animation, And Other Ways to Program Computers. In *Communications of the ACM*. Vol. 39, No. 8, 49-59. Aug. 1996.

[KLU95] Kluszczynski, Ryszard. Audiovisual Culture in the Face of the Interactive Challenge. In WRO95 Media Art Festival, 24-40. Wroclaw: Open Studio, 1995.

[LAK87] Lakoff, George. Women, Fire, and Dangerous Things: What Categories reveal about the Mind. Chicago: The University of Chicago Press, 1987.

[LAN87] Langton, Christopher G. Artificial Life. In *Artificial Life* (Proceedings of the First International Conference), ed. by Chris Langton, 1-47. Addison-Wesley, 1987.

[LEC80] Le Corbusier. The Modulor I and II. Cambridge, Massachusetts: Harvard University Press, 1980.

[LIT94] Litwinowicz, Peter and Williams, Lance. Animating Images with Drawings. In *Computer Graphics* (ACM SIGGRAPH '94 Conference Proceedings), 409-412. 1994.

[LYN97] Lynton, Norbert. Expressionism. In *Concepts of Modern Art*, ed. By Nikos Stangos. London: Thames and Hudson. 1997.

[MAE] Maeda, John. Deconstructing Cyberspace.

[MAE93] Maeda, John and McGee, Kevin. *Dynamic Form*. Tokyo: International Media Research Foundation, 1993.

[MAE97a] Maeda, John. A Framework for Digital Expression. In *Digital Communication Design Forum at Tokyo Design Center*, 25-32. Tokyo: International Media Research Foundation, 1997.

[MAE97b] Maeda, John. *Reactive Books*. Tokyo: Digitalogue, 1997.

[MAE98a] Maeda, John. *Aesthetics and Computation Research.* <http://acg.media.mit.edu/people/maeda/research.html> [accessed 10 January 1998]

[MAE98b] Maeda, John. *Design by Numbers.* To be published by the MIT Press, 1998.

[MIT92] Mitchell, WJT. Word and Image. In Critical Terms for Art History. Ed. Robert S. Nelson and Richard Shiff. Chicago: The University of Chicago Press, 1992.

[MOH98a] Mohr, Manfred. *Manfred Mohr/Monograph: Works from 1960 – 1998.* <http://sciweb.nyu.edu/~mohr/ini_ri.html> [accessed 25 October 1998]

[MOH98b] Mohr, Manfred. *The Digital Artist : Art, Abstraction and Algorithms.* <http://www.wmgallery.com/news/apr98.html> [accessed 25 October 1998]

[MOL98] Molnar, Vera. *25th Anniversary Celebration of Pioneering Computer Artists.* ACM SIGGRAPH '98 Conference Proceedings. 1998.

[OXF98] *The Oxford English Dictionary* <http://bion.mit.edu:8000/> [accessed 10 October 1998]

[PAV85] Pavlidis, Theo. An Automatic Beautifier for Drawings and Illustrations, In *Computer Graphics* (ACM SIGGRAPH '85 Conference Proceedings), 225-230. 1985.

[PEN94] Penny, Simon. The Darwin Machine: Artificial Life and Art. In *ISEA'94 Proceedings.* 1994. <http://www.uiah.fi/bookshop/isea_proc/nextgen/penny.html> [accessed 19 September 1998]

[PRE93] The Premisys Corporation, Chicago. Squiggle, 1993.

[PRU98] Prusinkiewicz, P. Hammel, M. Mech, R. and Hanan, J. The Artificial Life of Plants. In *Artificial Life for Graphics, Animation, Multimedia, and Virtual Reality*, Course 22 of SIGGRAPH '98 Course Notes. ACM SIGGRAPH, 1998.

[RES95] Resnick, Mitchel. Turtles, Termites and Traffic Jams: Explorations in Massively Parallel Microworlds. Cambridge, Massachusetts: The MIT Press, 1995.

[RES97] Resnick, Mitchel, and Silverman, Brian. *Exploring Emergence. An "active essay" on the Web.* <http://el.www.media.mit.edu/groups/el/projects/emergence/> [accessed 16 January 1998]

[SAC98] Sack, Warren. Artificial Intelligence and Aesthetics. <http://wsack.www.media.mit.edu/people/wsack/ai-aesthetics.html> To appear in Michael Kelly (editor-in-chief). *The Encyclopedia of Aesthetics.* New York: Oxford University Press, 1998.

[SAL94] Salisbury, M, Anderson, S, Barzel, R, and Salesin, D, *Interactive Pen-and-Ink Illustration*, In Computer Graphics (ACM SIGGRAPH '94 Conference Proceedings), 101-109. 1994.

[SEA96] Seaman, Bill. Passage Sets: One Pulls Pivots at the Tip of the Tongue, In Mediascape. Guggenheim Museum Soho, New York, 1996.

[SEA97a] Seaman, Bill. Models of Poetic Construction and their Potential use in Recombinant Poetic Networks. CADE97, April 1997

[SEA97b] Seaman, Bill. Emergent Constructions: Re-embodied Intelligence within Recombinant Poetic Networks. In *Proceedings of Consciousness Reframed: art and consciousness in the post-biological era*. Wales, 1997.

[SIM91] Sims, K. Artificial Evolution for Computer Graphics, In *Computer Graphics* (ACM SIGGRAPH '91 Conference Proceedings), 319-328. 1991.

[SIM94] Sims, K. Evolving 3D Morphology and Behavior by Competition, In *Artificial Life IV Proceedings*, ed. R. Brooks and P. Maes, 28-39. Cambridge, Massachusetts: The MIT Press, 1994.

[SMI96] Crampton Smith, Gillian and Tabor, Philip. The Role of the Artist-Designer. In *Bringing Design to Software*, ed. Terry Winograd, 37-57. Addison-Wesley, 1996.

[SMI96] Smith, Joshua R. Field Mice: Extracting Hand Geometry from Electric Field Measurements. In *IBM Systems Journal*, Volume 35, No. 3&4.

[SNI98] Snibbe, Scott. *Motion Phone.* <http://www.snibbe.com/scott/mphone/index.htm> [accessed 16 January 1998]

[SNI98b] Email exchange with Scott Snibbe in Ocotber 1998.

[SUT63] Sutherland, Ivan E. Sketchpad: *The First Interactive Computer Graphics.* <http://wwwwswest2.sun.com/960710/feature3/sketchpad.html> [accessed 28 September 1998]

[SVA97] Svanæs, Dag. Kinaesthetic Thinking: The Tacit Dimension of Interaction Design, In *Computers in Human Behavior*, Vol 13, No. 4, 443-463. Elsevier, 1997.

[THO90] Thompson, James M. ed. *20th Century Theories of Art.* Ottawa, Canada: Carleton University Press, 1990.

[TUF90] Tufte, Edward *Envisioning Information*. Cheshire, Connecticut: Graphics Press, 1990.

[TUR84] Turkle, Sherry. *The Second Self: Computers and the Human Spirit*. New York: Simon & Schuster, 1984.

[TUR95] Turkle, Sherry. *Life on the Screen: Identity in the Age of the Internet*. New York: Simon & Schuster, 1995.

[TRU98] Truckenbrod, Joan. *25th Anniversary Celebration of Pioneering Computer Artists*. ACM SIGGRAPH '98 Conference Proceedings. 1998.

[VER98] Verostko, Roman. *ALGORITHMIC ART: Composing the Score for Fine Art.* <http://design.mcad.edu/home/faculty/verostko/algorithm.html> [accessed 25 October 1998]

[VER94] Verostko, Roman. Algorithms and the Artist. In *ISEA'94 Proceedings*. 1994

[WHI91] John Whitney, Fifty Years of Composing Computer Music and Graphics: How Time's New Solid-State Tactability Has Challenged Audio Visual Perspectives. In *Leonardo*, 597-599. Vol. 24, November 5, 1991.

[WHIT98] White, Tom. *Introducing Liquid Haptics in High Bandwidth Human Computer Interfaces*. MS Thesis, MIT Media Lab, May 1998.

[WIL95] Wilson, Stephen. Artificial Intelligence Research as Art. In *Stanford Humanities Review. Constructions of the Mind: Artificial Intelligence and the Humanities*. ed. Stefano Franchi and Güven Güzeldere. Vol. 4, Issue 2, Spring 1995

[WIL98] Wilson, Mark. Distinction Prix Ars Electronica 92 in the category Computer Graphics <http://www.aec.at/prix/1992/E92azG-18.html> [accessed 25 October 1998]

[WON98] Wong, Michael T., Zongker, Douglas E., and Salesin, David. Computer-Generated Floral Ornament. In *Computer Graphics* (ACM SIGGRAPH '98 Conference Proceedings). 1998.

[ZEL96] Zeleznik, R. Sketch: An Interface for Sketching 3D Scenes, In *Computer Graphics* (ACM SIGGRAPH '96 Conference Proceedings), 163-170. 1996.

appendix A

VISUAL ARTS SOFTWARE

MetaCreations

Kai's SuperGOO
Real-time liquid image distortion tools.

Art Dabbler
Digital imaging tools that let you trace photos and turn them into paintings or drawings. Teach you to draw, paint and animate.

Painter 5
Software that simulates traditional tools and techniques.

KPT 3
Creates texture backgrounds, color and 3D text effects.

Expression
Combines the stylistic expressiveness of traditional artist tools with the speed, flexibility and resolution independence of vector-based drawing.

Imaja

Bliss Paint
Real-time painting and animated color synthesis. Generate images by drawing from a large library of animated shapes and patterns.

Geometric Bliss
Dynamic tiling tools and paintings for Bliss Paint and Bliss Saver.

Adobe Systems

Adobe After Effects
The professional tool for broadcast design

Adobe Illustrator
The industry-standard illustration software

Adobe Photoshop
Create, paint, correct, and retouch with the "camera for your mind"

Xaos Tools

Paint Alchemy
Offers 75 built-in brushstroke effects, 36 brush styles and the ability to modify the look or create individual styles or brushes.

Terrazzo 2
Creates symmetrical tiled backgrounds from any image by selecting a tiling symmetry and then selecting an image area.

Corel

CorelDRAW

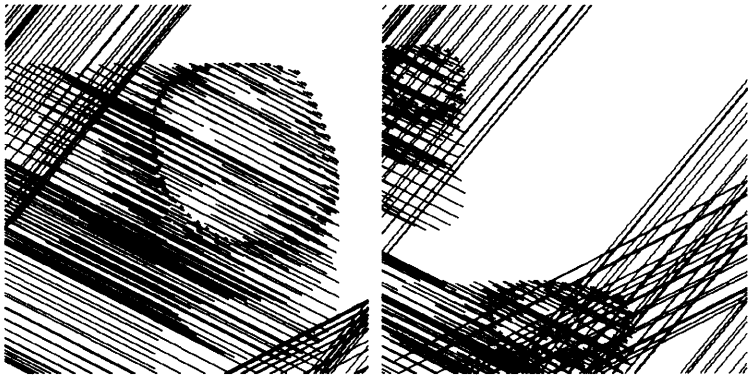
PHOTO-PAINT

Graphics Pack II

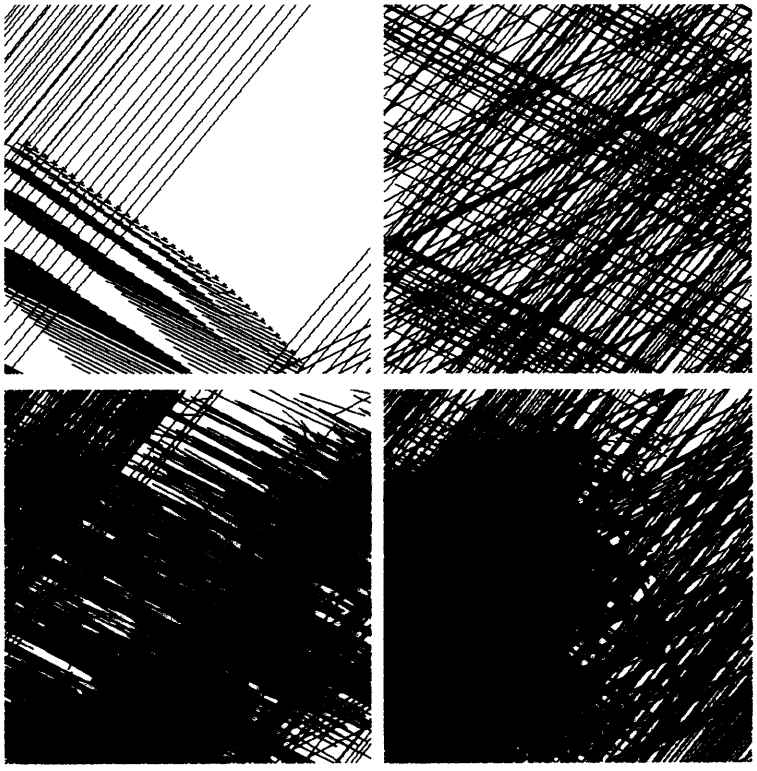
appendix B

COMPUTATIONAL DRAWINGS

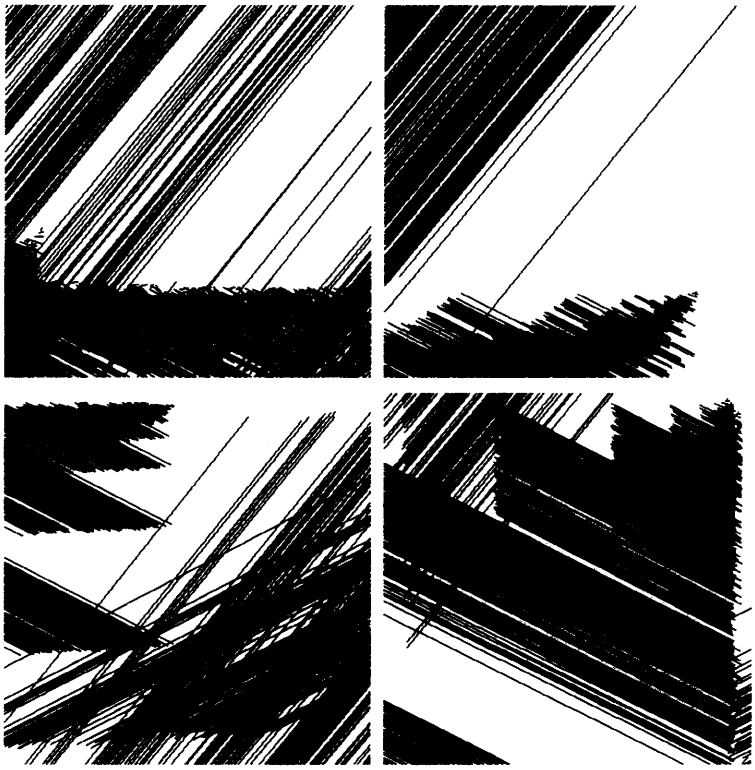
The TABLE line behaves very differently depending on which section of the canvas it is drawing upon. The first two panels show two single gestures: one large circle on the left panel and two smaller circles on the right. The circles are represented by a series of small line segments on the display. The computational line has additional components that are also drawn. The parallel, intersecting lines, which occur farther away from the gesture, but still, follow its directionality.



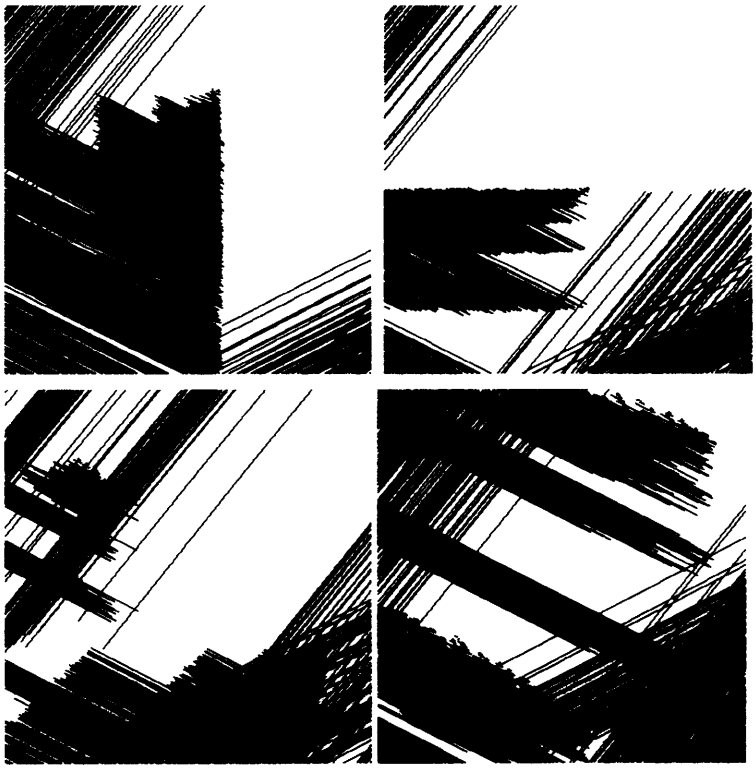
It is possible to create compositions of parallel lines alone, by keeping the gesture outside of the boundary of the canvas.



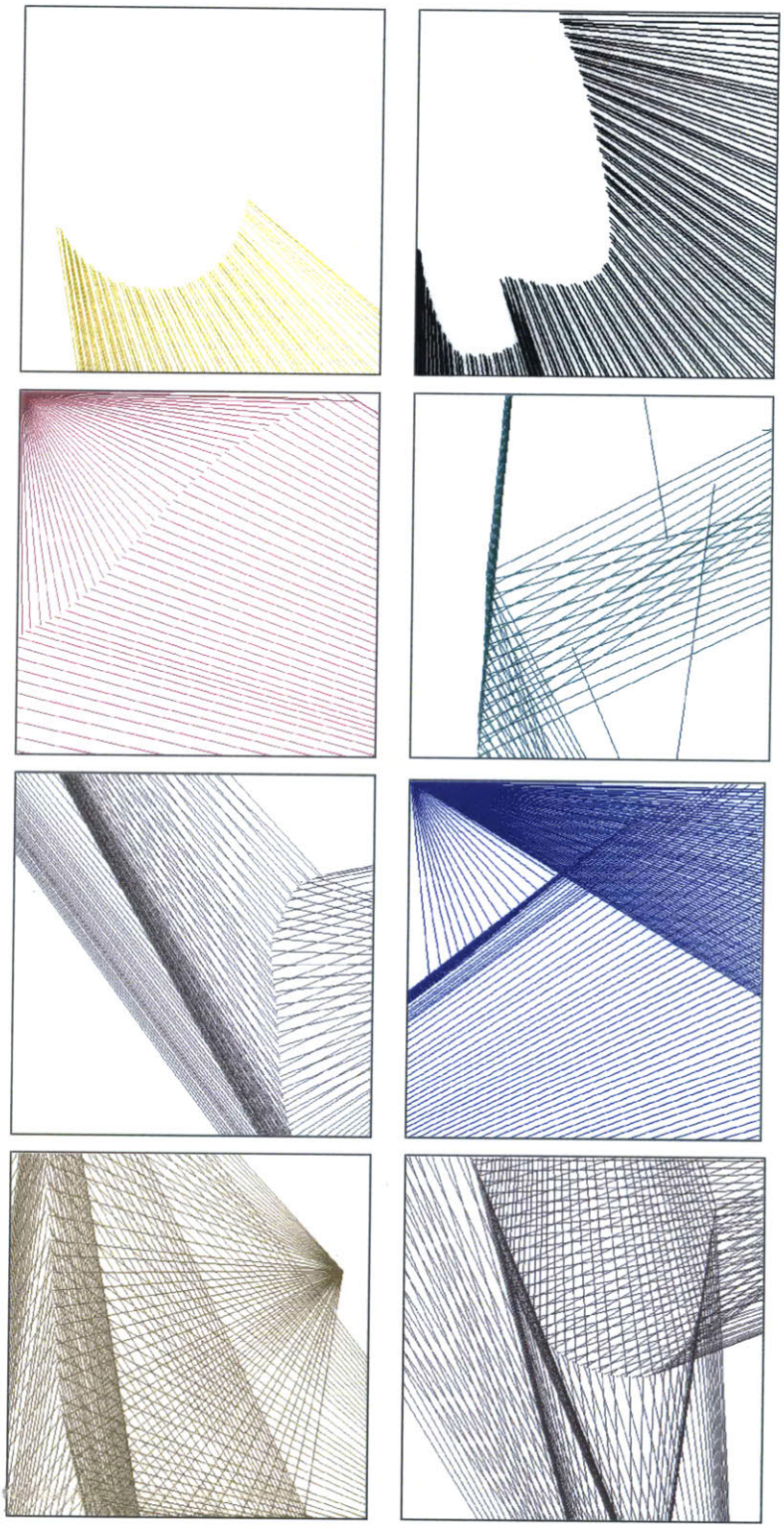
Two types of linear representation are feasible: a set of parallel stripes that shade the canvas, and the smaller segments that actually follow the cursor. The smaller segments have an interesting erratic quality that contrasts nicely with the parallel lines.



Shapes are drawn by carefully shading certain areas of the canvas.

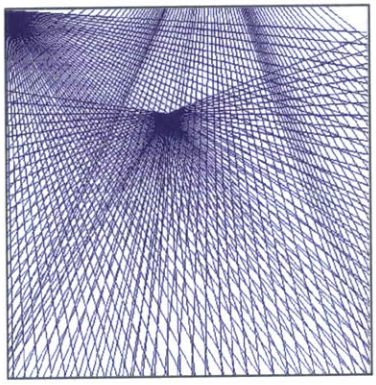


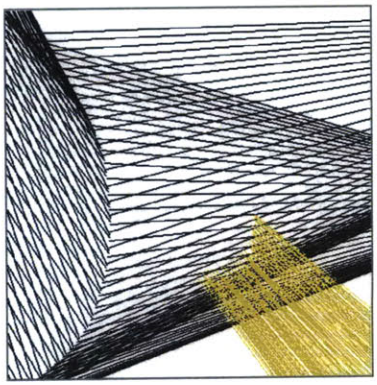
In GRIDS, each computational line is an algorithm that generates a grid-like pattern of lines, a function of the coordinates of the pointer on the canvas. The emphasis is on a set of lines that are parallel or all tend towards the same vanishing points.



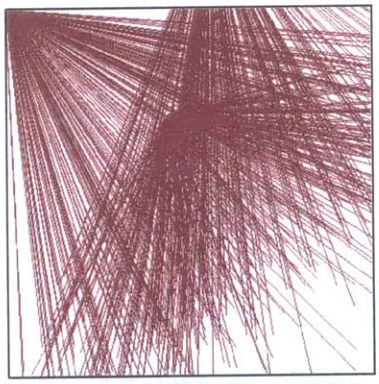
These were inspired by a fascination with lines, and patterns formed by the intersections of many parallel lines as well as a desire to create

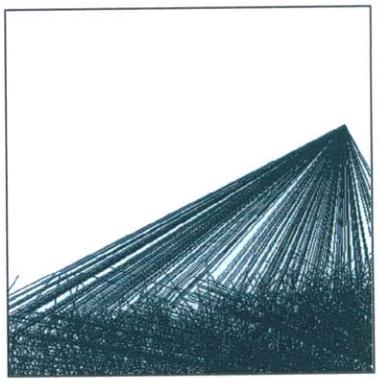
complex patterns with one single gesture, and to be able to shape that pattern of lines through properties of the gesture. The speed and direction of motion has a strong impact on the character of the composition. The examples follow the development of complexity of the GRIDS lines.

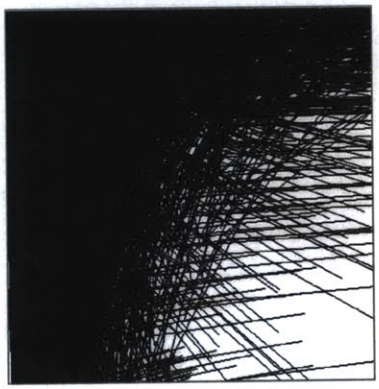


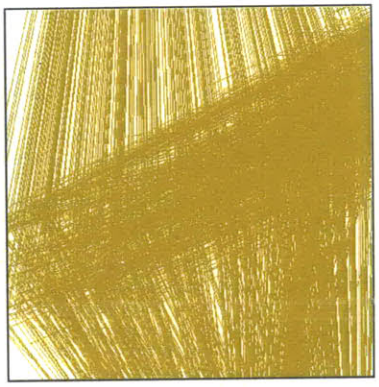


GRIDS lines can be used to create a single geometrical composition. Texture can be built up from several lines, and shapes can be constructed through the repeated definition of tone and texture. A more rigid gesture produces a more geometrical composition, whereas frantic movements deconstruct the geometry, and allow a more textural approach to building up shape. It is a transition from drafting to painting tools, the more expressive gestures exacting a more expressive, less deterministic approach. It is interesting how this is a natural result of computation. A faster input puts more strain on the processor to perform calculations, and more disarray is introduced into the algorithms and into the plotting procedures. The lines are no longer exactly parallel, the pattern becomes muddled.

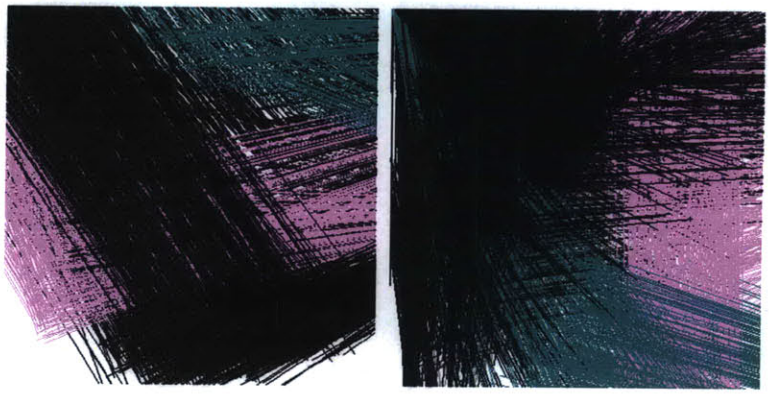




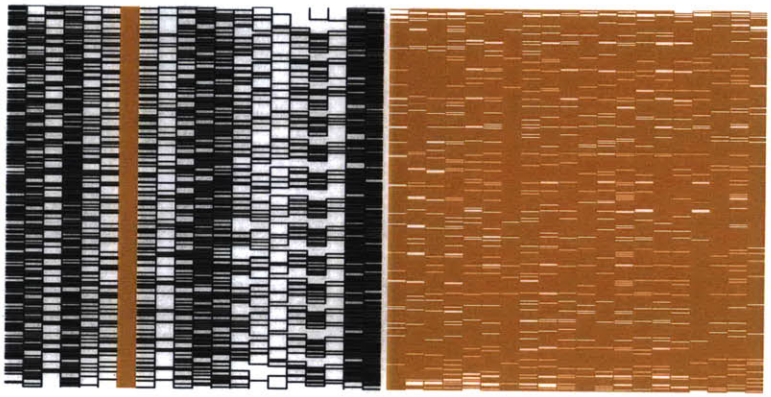




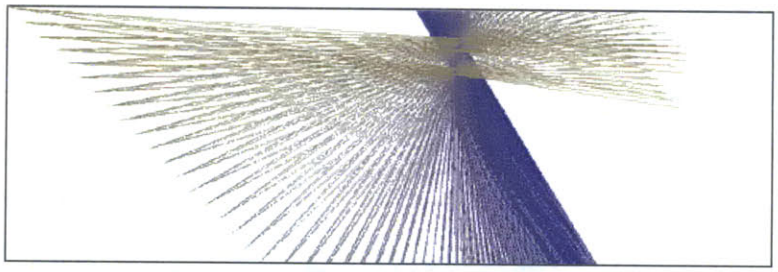
Finally, the two last panels show use of multiple colors.



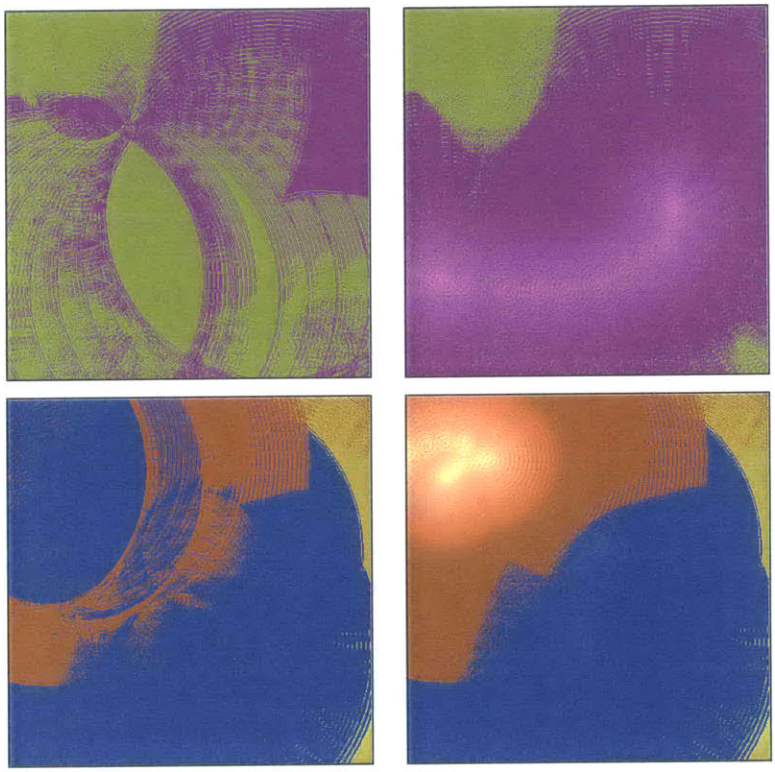
The following example called BOXES illustrates a computational line that restricts one degree of freedom of movement. Gestures along the horizontal axis define where columns of boxes are drawn. Movement along the vertical only determines frequency and sizes of boxes within each column. Colors can be selected to apply strictly to particular columns. The line can be drawn both as positive and negative space, depending on the intensity of rubbing.



This drawing, called Angel3, was one of the first examples of a dynamic line. The line originates from the central blue area, and rotates while changing color to create the shape shown below.

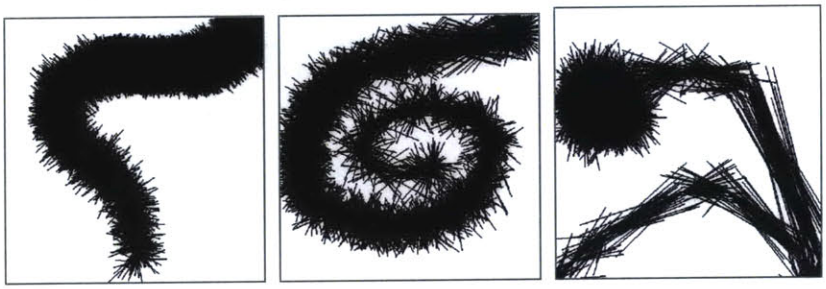


The next set of stills comes from a dynamic animated line called OSCOPE. This was one of the first dynamic tools, and revealed a very appealing process of drawing but a pedestrian outcome. The drawing tools are essentially in constant movement, the lines draw over each other, overlaying color and line in circular patterns, the center of the circles follows the movement of the cursor. A faster input speeds up the animation. The colors of the lines fade as the radius decreases. The lines take turns drawing over each other, and the artist can spend long periods of time making marks without a clear composition in mind. This is certainly an animation tool, a way of exploring dynamic shape, or drowning in the magic of hypnotic, multicolored, responsive motion.



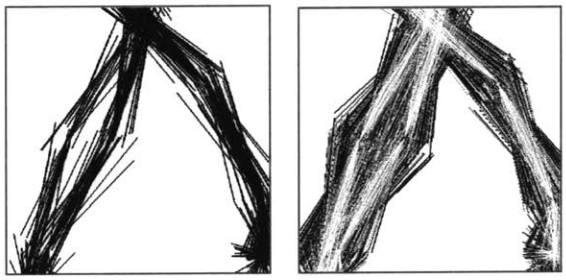
The next line is one that has already been introduced in the body of the thesis, called the Hairy Line. The lines start out as a thick, black mark. After it has been drawn, the mark expands, grows in thickness, projecting hundreds of little hairs out of its spine.

A range of effects can be achieved through differences in the quality of the stroke. A quick stroke causes shorter hairs, close together and perpendicular to the direction of drawing. A quick stroke generates longer hairs, farther apart and more parallel to direction of the stroke.



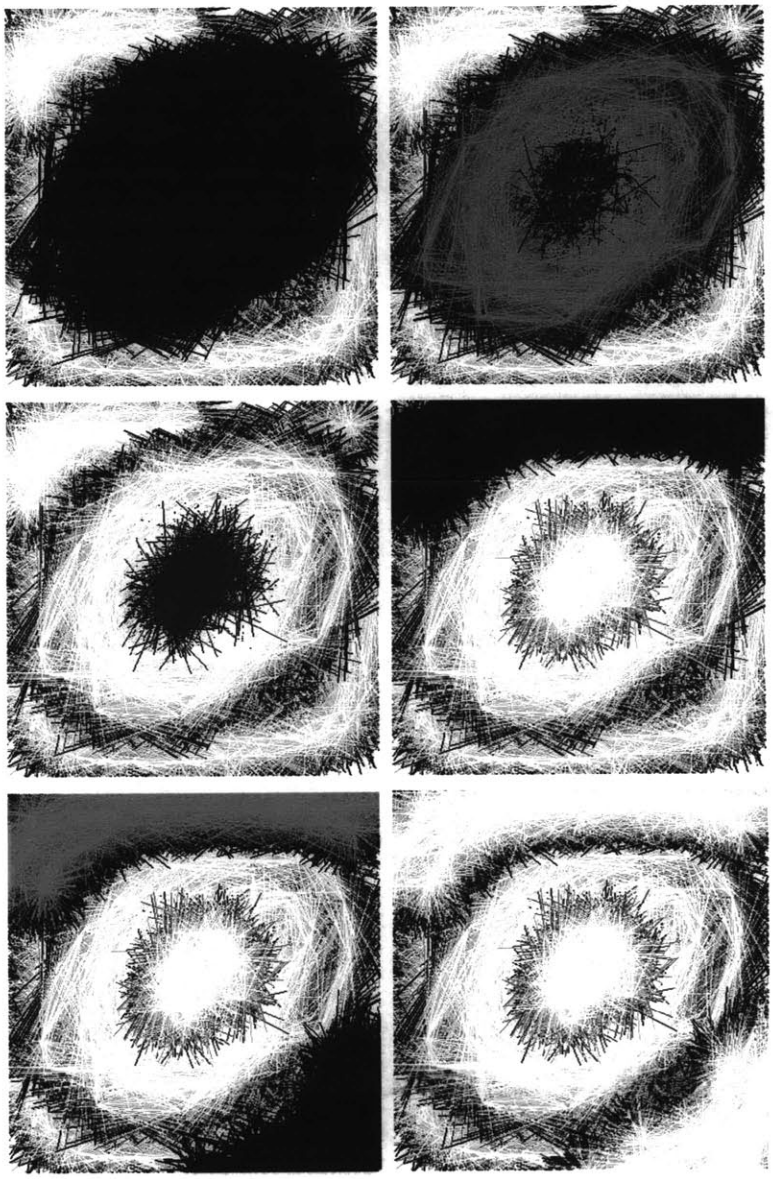
Varying effects generated by the Hairy Line

The hairs start out dark, but become lighter and lighter over time. The color of later hairs fades as they are drawn over previous ones. As a result, the overall composition fades. The space where the initial line was drawn becomes textured negative space, and only the outline remains as dark positive space.



The Hairy Line fades over time.

The following sequence of stills illustrates the drawing process as several lines are added over time, building up a drawing.

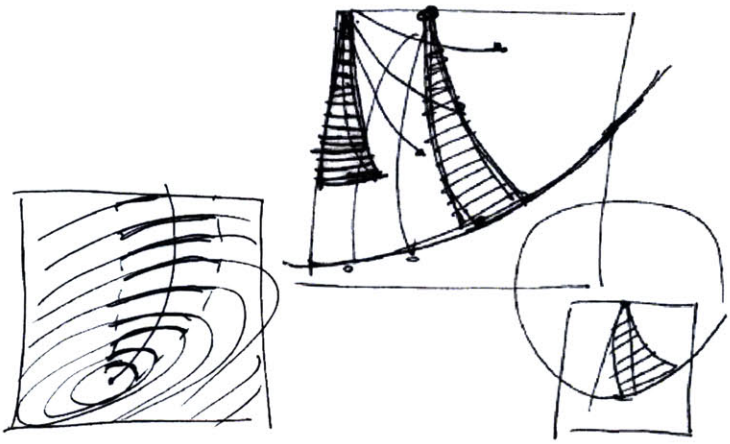


THE CREATIVE PROCESS

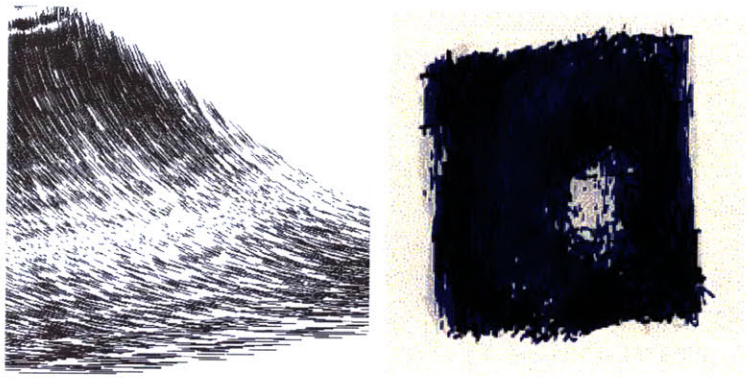
Computational line ideas:

line (cartesian/polar)
shape and form
space (positive/negative)
texture
value
color
third dimension
repetition (patterns)
variety
rhythm
balance
emphasis
push (xyz)
pull (xyz)
in/out of focus
agitate
stabilize
rub
act/react
time
place
speed up/down
your movements/speed
push the color
fade
erase
melt together
join paths
interact in abstract animation
grow/shrink
along a path
geometric transformation
translation
reflection
rotation
scaling
background to foreground
in and out of focus
change of appearance
change of color
change of state:
from line to shape
from photo to color
pixilated pictures
squeezed into shape
disappearing line
invisible line
force fields

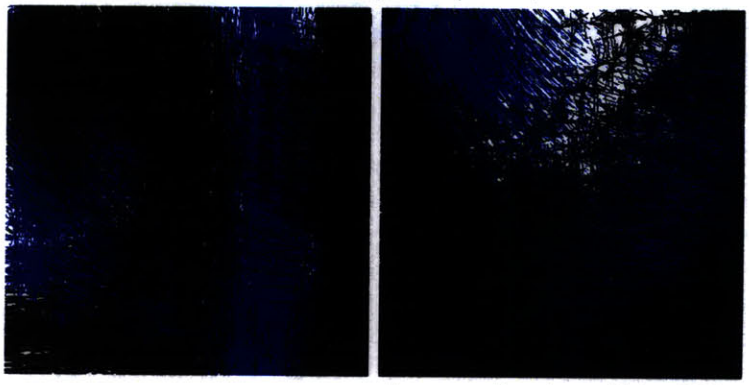
The role of sketching is important both in the idea planning stage and in the execution stage where various algorithms are evolved through experimentation. On the left, I have listed a set of computational line variables and ideas that involve appearance, dynamism and behavior. Hand drawn sketches were also used to generate computational lines. The drawings below inspired the ARC computational line.



These panels illustrate a progression from a sketch on paper to a set of computational drawings , using the ARC computational line shown in the upper left corner.



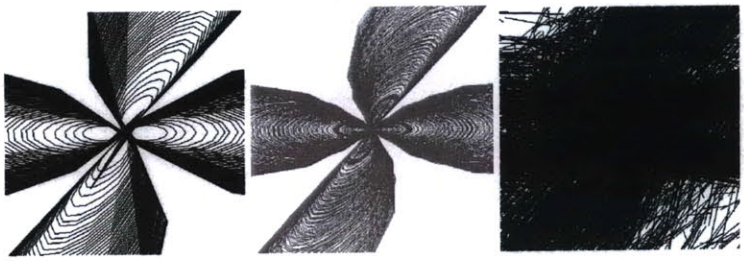
The ARC computational line and the paper color drawing that inspired the following two compositions, drawn with the ARC line.



Two compositions drawn with the ARC line

Next, I show one simple example of code, and I explain the progression of code development. All the code is available from http://www.media.mit.edu/~joey/demo/, or by contacting the author.

THE POPPY LINE

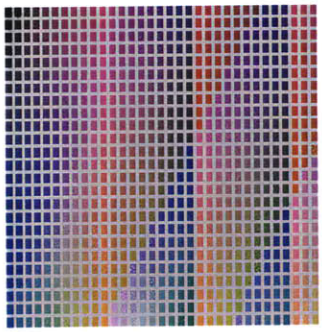


```
Starting from a general parametric curve for a folium:
x = (cos(3t))(cos(t)) y = (cos(3t))(sin(t)) for 0<t<Π
We evolved the following form:
public void drawLoopyLine(int x1, int y1, int x2, int y2) {
int xStep = (int) ( (float)(x2-x1)/(float)32 );
int yStep = (int) ( (float)(y2-y1)/(float)32 );
int c = (x1+(3*y1))/4;
for (int t=-6; t<37; t++) {
float s1 = (float)(t)/10;
float s2 =  (float)(t+1)/10;
float X1 = (float)(c*Math.cos(3*s1)*Math.cos(2*s1));
float Y1 = (float)(c*Math.cos(3*s1)*Math.sin(s1));
float X2 = (float)(c*Math.cos(3*s2)*Math.cos(2*s2));
float Y2 = (float)(c*Math.cos(3*s2)*Math.sin(s2));
g.drawLine(200+(int)X1+t*xStep, 200+(int)Y1+t*yStep,
200+(int)X2+(t)*xStep, 200+(int)Y2+(t)*yStep);
        }
}
```

CODE: FIRST ITERATION

Computational expressionism deals with the representation of a hand gesture on a two dimensional canvas. The computational line possesses attributes of dynamism and interactive behavior, in addition to the primary attribute of appearance.

The first iteration of code consisted of individual applets that illustrated single ideas or components of computational drawing. The Hairy line is one such self-contained applet. It is made up of four classes:
DrawingApplet.java,
FadingLineObject.java,
PointObject.java
JoeyGraphics.java.



MATHART example

Another set of applets from the first generation were the MATHART applets which often consist of a single class (that extends applet) and describe the even handling of the gesture and the colors of all the pixels represented on the canvas.

The color method in the following code example takes as parameters the x and y coordinates of each pixel, the width and height of the canvas, and the size of little squares to be drawn. The posx and posy variables represent mouse location.

```
public void color(int x, int y, int rW, int rH, int sW, int
sH) {
  for (int i=x-rW/2; i<x+rW/2; i=i+sW) {
    for (int j=y-rH/2; j<y+rH/2; j=j+sH) {
      double iq = (double)((double)i/(double)30);
      double jq = (double)((double)j/(double)(80+posx));
      double ijq = (double)( (2*(double)i+(double)j)
/(double)(25+posy/3));
      int red = (int) ( ( iq) / (double)2*(double)255) ;
      int green = (int) ( ( jq) / (double)2*(double)255) ;
      int blue = (int) ( (ijq) / (double)2*(double)255) ;
      if ((red < 0) || (red > 255)) red=255-(red%255);
      if ((green < 0) || (green > 255)) green = 255-
(green%255);
      if ((blue < 0) || (blue > 255)) blue = 255-(blue%255);
      c = new Color(red,green,blue);
      offGraphics.setColor(c);
      offGraphics.fillRect(i,j,sW-2,sH-2);
     }
   }
 }
```

All the code is available from http://www.media.mit.edu/~joey/demo/, or by contacting the author.

CODE: SECOND ITERATION

Each gesture that is drawn is abstracted into a vector of points in two-dimensional Cartesian space. The vector is composed of point objects that record the (x,y) coordinates and the processor time at which each point was created. This is used for animating dynamic drawings as well as to reserve the possibility of recreating the drawing in the future.

The vector also knows its starting point, the time at which the line itself was drawn. Finally, the line is given an ID number, or simply sub-classes an existing line object, to determine its appearance and behavior. The computational line is a Java composite of three properties, one from each of the categories, that can be recombined at will.

The second iteration of the drawing tools organized the classes and methods into three categories:

express.appearance
express.dynamism
express.behavior

The first category can be described as a set of methods that take two coordinate pairs and an integer ID as arguments. The ID is used to determine how the space between the two points is to be represented graphically. Some representations are very simple, such as a straight line connecting two points, of varying thickness and color, or a series of little squares that overlap in the direction of the gesture that generated them. Others are more complex, such as parametric patterns of line or color, curves that are influenced by the direction of the stroke, its speed, and the acceleration and deceleration. Finally, the concept of line is abstracted to the point where a line can be graphically represented as a color field, or a grid pattern, or text, or images from a film.

The second category gives movement to the graphical objects (lines). This movement can be a translation, rotation or other geometrical transformation on the vector of points that make up the line. It can also be a change of color, or an animation, or a fading into the background,

change of scale. It occurs once, and is a very intrinsic quality of the line making tool.

The third category takes movement one step further, into what I call behavior. Behavior usually involves movement, but it is not a quality of the line making tool, but a quality of the drawn line. To make an analogy to painting, a movement is the paint that drips from the brush onto the canvas, and the color that drips immediately after being painted. A behavior is the way the paint cracks after drying, and the way it reacts to other brushstrokes that are layered on top of it. If the paint is relatively fresh, it will blend a little, otherwise it is masked. Applying a faster drying paint on top of a slower drying one will produce an interesting pattern of cracking paint. These are behaviors. They center around the interaction between different strokes, and time.

An integral part was the JoeyGraphics class. JoeyGraphics has a method called drawIdLine(int id, int x1, int y1, int x2, int y2). Its arguments are two coordinate pairs and and an integer ID, which is used to determine how the space between the two points is to be represented graphically. An ID of 1, for example, simply invokes the Graphics.drawLine method. Other ID numbers invoke one or several joeyGraphics methods. Some of their names are drawThickLine, drawHairyLine, drawPensiveLine or drawIrritableLine. There is also a set of methods called drawParamXLine, where X is an integer. These methods paint parametric curves.

CODE: THIRD ITERATION

The third iteration aims to decompose the computational lines into very basic components. The package, called express, has the following structure:

```
express
    DrawingProgram.class
    awt
        ChoiceBox.class
        ColumnOfBoxes.class
        ColumnOfNumberBoxes.class
        GBconstraints.class
        NumberBox.class
        RedrawingCanvas.class
        TimedPoint.class
    color
        ColorSelector.class
        HSVcolor.class
    lines
        Redrawable.class
            RedrawableLine.class
                ChangingColorLine.class
                        Line011.class
                        Line012.class
                MovingLine.class
                        Line001.class
                        Line002.class
```

The awt package is used to build the drawing applications. The color package helps with color conversions and provides a framework for building a color selector. The lines package contains the Redrawable and RedrawableLine interfaces which are implemented in the abstract classes ChangingColorLine and movingLine. These two abstract classes describe basic color changes and movement for computational lines. Individual line classes extend the abstract classes and provide specific information for appearance, dynamism and behavior.

All the code is available from http://www.media.mit.edu/~joey/demo/, or by contacting the author.