# Painterly Interfaces for Audiovisual Performance

**Golan Levin**
B.S. Art and Design
Massachusetts Institute of Technology
May 1994

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 in Media Arts and Sciences** at the
**Massachusetts Institute of Technology**
September 2000



Author: **Golan Levin**
Program in Media Arts and Sciences
August 4, 2000

Certified by: **John Maeda**
Associate Professor of Design and Computation
Thesis Supervisor

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

# Painterly Interfaces for Audiovisual Performance

**Golan Levin**

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning, on August 4, 2000,
in partial fulfillment of the requirements for the degree of
Master of Science in Media Arts and Sciences

## Abstract

This thesis presents a new computer interface metaphor for the real-time and simultaneous performance of dynamic imagery and sound. This metaphor is based on the idea of an inexhaustible, infinitely variable, time-based, audiovisual "substance" which can be gesturally created, deposited, manipulated and deleted in a free-form, non-diagrammatic image space. The interface metaphor is exemplified by five interactive audiovisual synthesis systems whose visual and aural dimensions are deeply plastic, commensurately malleable, and tightly connected by perceptually-motivated mappings. The principles, patterns and challenges which structured the design of these five software systems are extracted and discussed, after which the expressive capacities of the five systems are compared and evaluated.

Thesis Supervisor: John Maeda
Title: Associate Professor of Design and Computation

This work was supported by the News in the Future Consortium, the Digital Life Consortium, and the Intel, IBM, Merril-Lynch, and Shiseido Corporations.

# Painterly Interfaces for Audiovisual Performance

**Golan Levin**

The following person served as a reader for this thesis:

**Tod Machover**
Professor of Music and Media
MIT Media Arts and Sciences

# Painterly Interfaces for Audiovisual Performance

**Golan Levin**

The following person served as a reader for this thesis:

**Marc Davis, Ph.D.**
Chairman and Chief Technology Officer
Amova.com

# Acknowledgements

I am deeply indebted to the following individuals for their support and inspiration.

Andrea Boykowycz
Charlotte Burgess
Elise Co
Peter Cho
Marc Davis
Rich DeVaul
Andre Devitt
Mary Farbood
Rich Fletcher
Ben Fry
Megan Galbraith
Scott Gibbons
Lukas Girling
Carl Goodman
Kelly Heaton
Hiroshi Ishii
Alex Jacobson
Bill Keays
Axel Kilian
Ben Lapidus
Helen Levin
Shane Levin
Tod Machover
Michael Naimark
Rob Poor
Casey Reas
Joachim Sauter
Jared Schiffman
Bernd Schoner
Greg Shakar
Scott Snibbe
Gerfried Stocker
Rich Streitmatter-Tran
Brygg Ullmer
John Underkoffler
Max Van Kleek
Connie Van Rheenen
Bill Verplank
Tom White
Shelly Wynecoop

This thesis is dedicated to my advisor, John Maeda. I am profoundly grateful to him for creating the ACG environment, and for his willingness to take a chance on me and invite me into it. The depth of John's vision, the clarity of his principles, and his indomitable and relentless pursuit of excellence are extraordinary models. Thank you, John, for the opportunity.

# Contents

# Figures

**Design Experiments**

# 1. Introduction

*"In the impossibility of replacing the essential element of color by words or other means lies the possibility of a monumental art. Here, amidst extremely rich and different combinations, there remains to be discovered one that is based upon the principle [that] the same inner sound can be rendered at the same moment by different arts. But apart from this general sound, each art will display that extra element which is essential and peculiar to itself, thereby adding to that inner sound which they have in common a richness and power that cannot be attained by one art alone."*
*—Wassily Kandinsky (1912)*

## 1.1. Motivation

A few months ago the *New York Times* reported the discovery of a 9,000 year old bone flute in China. Remarkably enough, the flute was still playable. As I listened in awe to sound files of the flute that the *Times* had posted on the World Wide Web, I was struck by an awareness that the human drive toward creative expression, as it is realized through such vehicles as musical instruments and drawing materials, must be among the oldest and most universal of human desires.

This thesis seeks to fulfill our will to creative expression, by making new expressions possible, and by advancing the state of the art in our contemporary means. My focus is the design of systems which make possible the simultaneous performance of animated image and sound. I have chosen to implement these systems by making use of the digital computer's capacity to synthesize graphics and sound in response to real-time gestural inputs.

This work is important as it represents a vision for creative activity on the computer, in which uniquely ephemeral dynamic media blossom from the expressive signature of a human user. The goal of this thesis is the design and implementation of a meta-artwork—an artwork for creating artworks—whose interface is supple and easy to learn, but which can also yield interesting, inexhaustibly variable, and personally expressive performances in both the visual and aural domains. In this thesis, I present several examples of works which come close to this goal, by bringing two things to bear on the problem space of audiovisual instruments: firstly, flexible technologies, such as real-time audio synthesis, gestural signal analysis, and expressive gestural interfaces; and secondly, a systems aesthetic, which seeks to substantiate such works with an underpinning of perceptual motivation, and infuse such works with a vibrant collaboration between the system's designer and its performer.

I am not the first person to attempt to design an audiovisual instrument. In fact, the vision of a performance medium which unifies sound and image has a long history, as Wassily Kandinsky's quote suggests. Instead, I hope to bring to this history a provocative new set of questions and answers about the power, beauty, sophistication and personality that it is possible for an audiovisual instrument to have.

### 1.2. Overview of the Thesis

The remainder of this thesis is organized into four chapters. In Chapter 2, *Background*, I present an overview of attempts to create visual and audiovisual performance systems. A large number of such systems have been developed, both prior to the advent of the digital computer, and subsequent to it. In this chapter, I discuss some of the most important historic and contemporary examples of audiovisual performance systems, and try to identify some of the basic themes, patterns and constraints which have structured the design of these systems.

Many of the prior examples discussed in the *Background* chapter cannot be considered true *audiovisual* performance instruments, in the sense that they do not permit the simultaneous authoring of both dynamic image and sound. Quite a number of these systems, such as score-based systems and control panel interfaces, place the creation of the image in a substantially subsidiary role to that of the sound. Other devices preclude the performance of sound altogether, sometimes on ideological grounds; Thomas Wilfred's landmark *Clavilux*, for example, was designed to explore the possibility of a strictly silent, visual analogue to traditional music. Some systems have indeed attempted to serve as simultaneous audiovisual performance systems, but fail in one respect or another, often because the sound and image are not commensurately malleable, or because the audiovisual output is too indirectly controlled. Nonetheless all of these systems are still enormously relevant, and form the chief historical and conceptual context within which this work is situated. As we shall see, this thesis is heavily indebted to the thinking behind Oskar Fischinger's *Lumigraph* (1950), in which continuous gestures of the hand were used to perform temporal abstractions in colored light, and Scott Snibbe's *Motion Phone* and *Dynamic Systems Series* (1995-98), which were animation performance systems in which the affordances of computation—iteration, conditional testing, simulation and data storage—were used to *augment* gesture.

I conclude the *Background* chapter by introducing a set of goals or desiderata, inspired by both the successes and shortcomings of the prior art, which I believe must be satisfied by any performance system which combines audio and visual performance. Finally, I present my own hypothesis about what sort of system would satisfy this set of goals—a painterly interface paradigm for audiovisual performance systems. This schema, which evolved as a reaction to the prior art, has formed the scaffolding of the new work I present, and is based on the idea of an inexhaustible audiovisual *substance* which is created and manipulated through gestural mark-making.

Chapter 3, *Design Experiments*, presents the new work which supports this thesis. That chapter, and the new work it represents, is divided into two main sections: a section which describes a series of software environments which were developed just prior to my matriculation in the MIT Media Laboratory's Aesthetics and Computation Group (ACG), and a second section devoted to five systems developed over the last two years in the ACG. These five systems—called *Yellowtail*, *Loom*, *Warbo*, *Aurora* and *Floo*—each enable the simultaneous creation of sound and animation, and are the core set of works that implement and support the painterly interface metaphor for audiovisual performance. As each system is discussed, the chapter considers the basic materials and methods which were used to construct it

Chapter 4, *Discussion and Analysis*, presents an analysis of my software artifacts which is designed to tease apart and taxonomize the elements of their design space; to understand the ways in which the five thesis instruments differ, succeed and fail; and to articulate principles for the design of future audiovisual instruments. The chapter is divided into four parts: a section on the design patterns which have proven indispensable to the design of the instruments; a section on the pitfalls and challenges encountered in their development; a comparative examination of the thesis instruments, in which a set of qualitative metrics are established according to which the instruments can be contrasted; and a section which discusses the greater contexts within which the thesis instruments are or can be evaluated.

Chapter 5, *Conclusion*, synopsizes the conclusions of the thesis work, and presents a section on directions for further research in the field of audiovisual performance instruments.

After Chapter 5 follow three brief appendices. *Appendix A* presents a timeline of lesser-known audiovisual performance devices, while *Appendix B* shows some of the paper sketches from which the thesis instruments developed. *Appendix C* presents pseudocode examples which explicate some of the inner mechanics of my applications.

### 1.3. Summary of Contributions

The goal of this thesis is to develop an engaging new medium for audiovisual self-expression, and to present historical, methodological, and analytical contexts for building, understanding and evaluating examples of this medium. The contributions of this thesis, with respect to this goal, include:

1. A survey and critical history of the relatively little-known history of performance instruments for abstract imagery and color-music.

2. A set of desiderata which, taken together, define the properties of an ideal audiovisual performance system.

3. A new interface metaphor for audiovisual performance instruments, intended to satisfy these desiderata, which is based on the idea of an inexhaustible, infinitely variable, dynamic "substance" whose visual and aural dimensions are deeply plastic and commensurately malleable.

4. Five new software instruments which embody this interface metaphor, including a discussion of the inner workings of each.

5. An analysis of these instruments, including a taxonomy of their successful and unsuccessful design elements; a vocabulary for evaluating the overall success of such instruments; and a comparison of the instruments' relative success, evaluated according to this vocabulary.

6. A brief outline of further directions and unexplored avenues for continued research in the domain of audiovisual performance systems.

# 2. Background

The synchrony of abstract image and sound, variably known as ocular music, visual music, color music, or music for the eyes, has a history that spans several centuries of work by dozens of gifted practitioners [Ritter 1993]. Despite the breadth and depth of this history, however, a casual Web search reveals an unfortunate ignorance of it, as numerous sites continue to advertise "an entirely novel concept, relating graphics and music" or something similar [Collopy 1999]. Adrien Bernard Klein, in his 1927 book *Color-Music: the Art of Light*, deftly characterized this myopia: "It is an odd fact that almost everyone who develops a color-organ is under the misapprehension that he, or she, is the first mortal to attempt to do so" [Klein 1927]. The absence of a ready history of this domain can be partially explained by its frequent association with the spiritual fringe, as well as the inability of the art establishment to commodify such intangible work [Snibbe and Levin 2000]. In this thesis I therefore present an extensive introduction to this little-known background, motivated as much by a desire to correct this myopia, as by a need to understand the lessons of previous work.

This chapter divides the relevant background into three sections. The first, *Visual-Music Systems in the Pre-Computational Era*, examines a few of the most influential pre-computational attempts to relate sound and image, across the domains of performance instruments, abstract film, and optical sound-synthesis. The second section, *Visual Music in the Computational Domain*, examines the most prevalent schema by which sound and image have been conventionally connected in the computer. In the third section, *A Painterly Interface for Visual Music*, I introduce a new metaphor for relating sound to image on the computer, and discuss a handful of the most directly related background examples.

## 2.1. Visual-Music Systems in the Pre-Computational Era

### 2.1.1. Color-Music Performance Instruments

#### 2.1.1.1. Castel's *Ocular Harpsichord*

The earliest known device for performing visual music was built in 1734 by a Jesuit priest and mathematician, Father Louis-Bertrand Castel (1688-1757). Influenced by the writings of the 17th Century Jesuit mystic Athanasius Kircher (1602-1680), Castel

sought "to give the colours, irrespective of their harmonic order, a kind of intensified quality of liveliness and lightness which they inevitably lack upon a canvas without life or motion" [Popper 1968]. Castel's *Clavecin Oculaire* coupled the action of a traditional harpsichord to the display of transparent paper tapes, whose colors were believed by Castel to correspond to the notes of the Western musical scale.

Castel's design consisted of a 6-foot square screen mounted above a normal harpsichord. This frame was perforated by sixty small windows, each containing a translucent colored tape, and each covered by a mechanical shutter connected by pullies to each key of the harpsichord. When a key was depressed, the shutter would open, permitting candlelight to pass through one of the transparent tapes. An improved model, built in 1754, was designed for a much larger audience and used some 500 candles with reflecting mirrors. According to William Moritz, arguably the premiere historian of color-music, Castel's second instrument must have been "hot, smelly and awkward, with considerable chance of noise and malfunction between the pulleys, curtains and candles") [Moritz 1997]. Castel described his "Ocular Harpsichord" in two essays that were subsequently translated and annotated by the contemporary German composer Georg Philipp Telemann [Peacock 1988]. "What stranger enterprise could be imagined in the whole field of art," wrote Castel, "than to make sound visible, to make available to the eyes those many pleasures which Music affords to the ears?"

Castel's dream of a visible music hardly seemed strange to the scores of artists, musicians, inventors and mystics he served to inspire over the centuries which followed. Many of these innovators adapted the basic design of Castel's ocular harpsichord, and in particular his use of a keyboard interface, as a template for their own experiments. After Castel followed a steady development of audiovisual instruments, employing a wide range of technologies and materials: Frederic Kastner's 1869 *Pyrophone*, for example, opened flaming gas jets into crystal tubes to create both sound and image [Popper 1968], while an 1877 device by Bainbridge Bishop sat atop a pipe organ and produced light with a high-voltage electric arc [Peacock 1988]. An instrument patented by William Schooling in 1895 controlled the illumination of variously-shaped vacuum tubes with a keyboard and set of foot-pedals [Peacock 1988]. Other historic examples include George Hall's *Musichrome* (1930s), Morgan Russell and Stanton Macdonald-Wright's *Kinetic Light Machine* (1931), Gordon Pask

and McKinnon Wood's *Musicolour* machines (1953), and Jordon Belson's liquid-based instruments from the late 1950's [Popper 1968, Peacock 1988, Moritz 1993]. These inventors and others are treated individually and pictorially in Appendix A, *A Timeline of Instruments for Color-Music Performance*.

The early Twentieth century was a phenomenal boom time in the development of abstract visual performance systems. Buoyed by advances in electric technology and optics, by the invention of cinema, by the birth of modern perceptual psychology, and by the rise of abstraction in Western visual art, dozens of new systems were developed in the space of just a few decades. Three Twentieth-century instruments deserve special attention for their exceptionally high degree of aesthetic and technological sophistication: Thomas Wilfred's *Clavilux*, Oskar Fischinger's *Lumigraph*, and Charles Dockum's *MobilColor Projector*. In discussing them, we shall touch on design issues—such as indirect versus gestural control, and the control of amorphous versus geometric images—which continue to bear an impact in the creation of today's computational instruments.

One more theme which has cut across more than four centuries of color-music research, from Castel's to that of the present day, is the question as to whether there are any "absolute" correspondences between sound and vision. It is one of the deepest issues in the field; some have felt that it is best answered through empirical studies in psychology, while others have denied the possibility of any essential mappings, and instead held that the matter is simply an aesthetic one, best handled by design. The truth is almost certainly somewhere in between. In the work which follows, we will see a glimpse of how some of the Twentieth century's greatest color-music innovators dealt with the issue.

#### 2.1.1.2. Thomas Wilfred's *Clavilux*

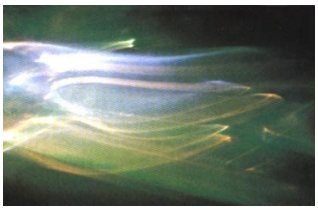


Figure 1. A photograph of a *Clavilux* projection. From [Scattergood-Moore 1998].

Danish-born Thomas Wilfred came to America as a singer of early music, and became involved with a group of Theosophists who sought to build a color organ to demonstrate spiritual principles. Initially, Wilfed sought an 'absolute' mapping between sound and color as a way of exposing these principles. Having carefully considered the work of his color-music predecessors, however, and, noting their failures and divergences, come to the conclusion that there was no absolute correspondence between color and sound, Wilfred instead turned his attention to an art of pure light in which sound and music were either completely excluded or admitted as mere accessories. He developed a color organ he

called the *Clavilux*, and named the art form of its silent animated-color projections "Lumia." These *Lumia*, which emphasized the use of slowly metamorphosing, polymorphous streams of fluid color, stand as the earliest surviving color music about which we can make fair aesthetic judgements [Popper 1968].

The first *Clavilux* was completed as early as 1919, and consisted of "a large keyboard with five rows of sliding keys and stops that could be coupled to obtain the colors; a battery of six principal projectors and a certain number of grouped auxiliary reflectors." Its design, according to Frank Popper, was very similar to an organ with its pipes [Popper 1968]. Wilfred gave his first public *Clavilux* recital on January 10, 1922, and thereafter began an extensive tour of *Clavilux* concerts in the United States, Canada, and Europe [Peacock 1988]. When he returned, Wilfred founded the "Art Institute of Light" in New York City, where he installed a 32-projector *Clavilux* in the Institute's auditorium, and gave two public *Lumia* recitals each week from November 1933 until May, 1934 [Scattergood-Moore 1998]. In addition to his large performance systems, Wilfred also constructed a variety of "Lumia boxes," self-contained units which could play for days or months without repeating the same imagery [Moritz 1997], as well as a small commercial run of "Home Clavilux" systems, which resembled televisions but were designed for performance by consumer instrumentalists.

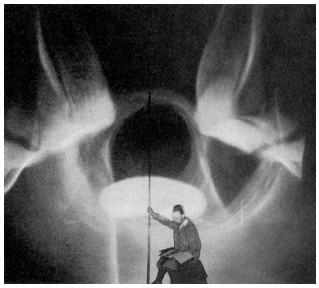


Figure 2. A *Clavilux* performance (date unknown).

Figure 3. Wilfred's *Lumia Box* and *Home Clavilux* used these hand-colored glass disks to produce a variety of light effects [Scattergood-Moore 1998]. A similar technology was independently developed by Wladimir Baranoff-Rossiné for his 1920 *Piano Optophonique* [Baranoff-Rossiné 1997], [Popper 1968].

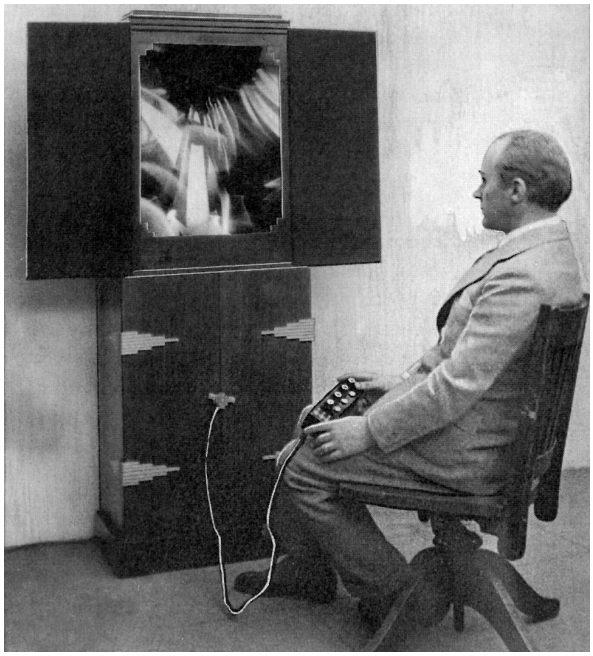


Figure 4. Thomas Wilfred using a *Home Clavilux*, c. 1930.

Wilfred's instruments and writings are important because they give voice to an aesthetics of *Lumia* as integral art form in its own right and with its own formal principles. In one article, for example, Wilfred makes a point of differentiating the composition and playing of *Lumia* from that of music. He thinks that the two arts are so different that "attempts to design *Lumia* instruments in imitation of musical ones will prove as futile as attempts to write *Lumia* compositions by following the conventional rules laid down for music." He also argued that the rules governing static composition and color harmony do not apply to form and color in motion: "If a *Lumia* composition is stopped at any point, an analysis of the static image may show both form and color out of balance from the painter's point of view." [Wilfred 1947, 1948]. These issues are no less important today, and are at play, as we shall see, in the works I have created to support this thesis; these are discussed in Chapter 3.

#### 2.1.1.3. Oskar Fischinger's *Lumigraph*

In the late 1940's the great abstract animator Oskar Fischinger invented a color organ instrument that allowed one to play light. According to William Moritz,

> "[The] *Lumigraph* hides the lighting elements in a large frame, from which only a thin slit emits light. In a darkened room (with a black background) you can not see anything except when something moves into the thin 'sheet' of light, so, by moving a finger-tip around in a circle in this light field, you can trace a colored circle (colored filters can be selected and changed by the performer). Any object can be used: a gloved hand, a drum-stick, a pot-lid (for a solid circle), a child's block (for a square), etcetera" [ Moritz 1997].

The story of the Lumigraph's genesis was recently retold by Elfriede Fischinger, Oskar Fischinger's widow:

> "...A few days later, he called me down to his studio where he had assembled what he called 'The Light Instrument.' The wooden panels had become a box-like frame about 1 foot wide and 1 foot deep. This 'frame' contained an opening that encased the latex sheet mounted on a wooden canvas-support 3 feet high by 4 feet wide. The colored gels had been fastened to glass strips that rotated on a wheel inside the wooden frame-case, and a thin slit just inside the front edge of the case only allowed the light from the (cool) neon tubes inside the case to emerge at this one point to make a thin layer of light in front of the rubber screen. This light slit was just far enough in front of the screen so that only those portions of the

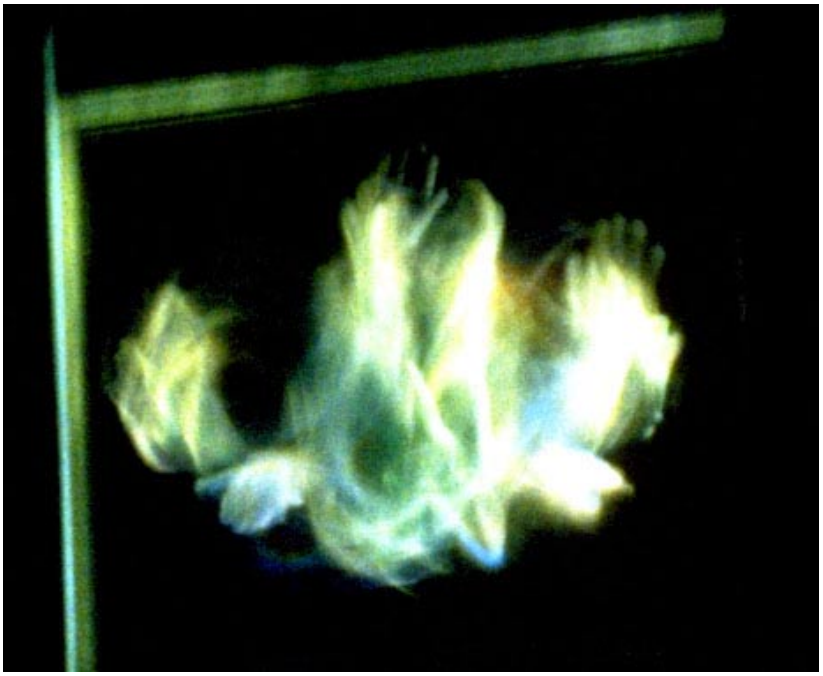


Figure 5. The *Lumigraph* in performance [Fischinger 1998].

> screen that were pushed forward would fall into the path of the light and become visible to the spectator sitting in front of the instrument—and unless something did protrude into the thin light layer, nothing would be visible at all! (The case was even painted black). Each color of gel had been mounted on a different glass strip, and these colored glasses could be rotated by pulling a canvas strip on the back side of the case.... He placed a black curtain behind the instrument, dressed entirely in black (long-sleeved turtle-neck sweater, etcetera) but wore white gloves, so that only the movements of his marvelously expressive hands would be visible, floating mysteriously in the darkness. Our daughter, Barbara, still remembers quite a bit of this, as she often worked the cords to change the colors according to Oskar's commands. For a smooth public performance, it took two people to play the instrument—one to perform the choreography of light, and one small, lithe person to pull the cords to change the colors at given cues. When Oskar played a piece like Sibelius' 'Valse Triste,' he was very particular about the colors, which had to be changed and mixed very precisely at exact moments. Although the latex screen was opaque, Oskar arranged a series of overhead mirrors so that he could see what the spectators were watching out front [Fischinger 1998].

Figure 6. Oskar Fischinger's *Lumigraph* was licensed for use in the 1960's sci-fi film, *The Time Travelers*.

Unlike Thomas Wilfred, who rejected the possibility of creating relationships between sound and image because of the questionable psychological validity of any individual mapping, we can see from this narrative that Fischinger had an altogether

different view. For Fischinger, the possibility of a relationship between sound and image, whether created with the Lumigraph or for his abstract films, represented an *opportunity for design*. However arbitrary such mappings might or might not be, Fischinger's own mappings were at the very least personal and deliberate. His attitude toward sound-image mappings is a great inspiration to my own work, described later in this thesis.

Fischinger performed the *Lumigraph* only a few times in public: at the Coronet Theater in Los Angeles, and at the San Francisco Museum of Art in 1953. The *Lumigraph* made brief appearances in an Andy Williams television special, and in the 1964 science-fiction movie *The Time Travelers*, in which it serves as a "love machine" that allows people to vent their sexual urges in a harmless sensuality [Moritz 1997]. According to Moritz, Fischinger hoped, like Castel long before, that someone would manufacture *Lumigraphs*, and that they would become common household items, used by children for play and artistic training, by adults for recreation and party games. Although that has not yet occurred, Oskar's original *Lumigraph* does survive, in the Deutsches Filmmuseum in Frankfurt, where it is played with some regularity [Moritz 1997].

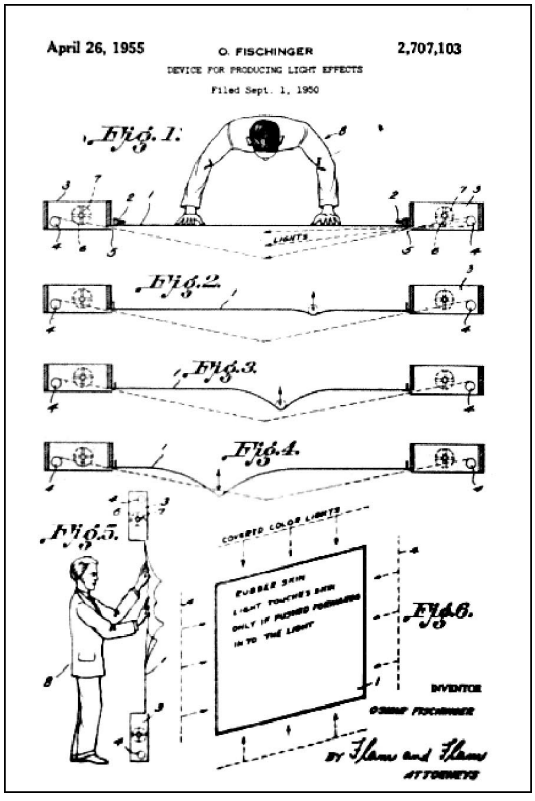



Figure 7. A schematic diagram of Fischinger's *Lumigraph*, from his 1955 patent for the device [Fischinger 1955].

The interaction design of Fischinger's *Lumigraph* represents a fundamentally important contrast to that of Wilfred's devices: Although both systems allowed a performer to perform patterns of light, Wilfred's *Claviluxes* produced visual displays according to the remotely-controlled action of motorized mechanisms, while Fischinger's simple latex screen directly and immediately conveyed the handmade and ephemeral markings of the performer's gestures. The space between "remote control" and "direct control" is a blurry one, since any medium by its nature interposes a material or process between its performer and its product. Nevertheless, the degree to which a system provides direct or indirect control is as much an issue in the design of computational systems as it is in these physical examples.

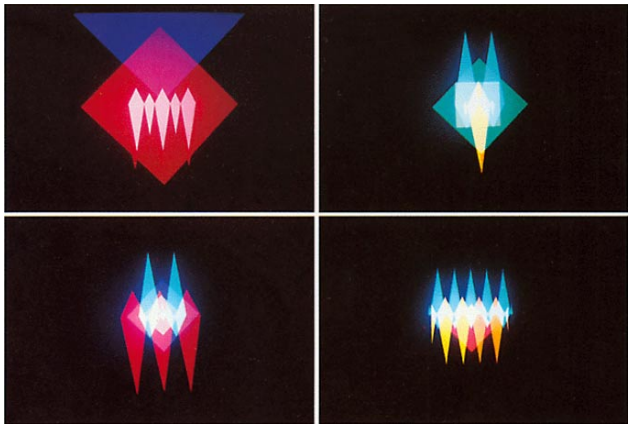
#### 2.1.1.4. Charles Dockum's *MobilColor Projector*

Charles Dockum was a California inventor who began making color organs in the late 1930's. His motivation for doing so is singularly interesting. According to William Moritz,

> "Dockum suffered respiratory problems throughout his life, and in his twenties came so close to death that he had the sort of out-of body experience in which one's spirit seems to detach itself and fly off through cosmic realms...His urge to create mobile-color projectors (console instruments for live performances of color imagery) arose from his compulsion to recreate and communicate his personal revelation." [Moritz 1993].

Dockum developed a large projection system called the *MobilColor* which allowed its performer to create temporal patterns of moving, colored shapes. The dynamics of these patterns were specified through a mechanical programming system, using differently shaped cams. Although the *MobilColor* projector could produce both hard-edged and soft-edged imagery, it did so through the use of prepared image sources. The vocabulary of its performances was therefore limited to translations, rotations, colorizations and defocusings of these constituent image-units.

Both Oskar Fischinger and Charles Dockum received fellowships from the Guggenheim Foundation through the Baroness Hilla Rebay, who specified that each spy on the other to make sure that he was really working on his grant project. Dockum's grant went into preparing a larger and more complex projector that would allow multi-layered motion in several directions—a projector destined for the Guggenheim Museum, since the rival Museum



Figures 8 (left) and 9 (right). Charles Dockum with his *MobilColor Projector*, and some examples of the instrument's projected displays.

of Modern Art had a Thomas Wilfred *Lumia* on display. According to Moritz,

> "When Dockum installed the new *MobilColor* in the Guggenheim Museum, the Baroness was shocked to learn that it required one or two operators to perform it (whereas Wilfred had developed automatic self-contained *Lumia*). The projector was consigned to storage, and a few years later dismantled, with the light units used for track-lighting in the galleries and the rest of the mechanisms trashed. This meant that all of the compositions that Dockum had created uniquely for that instrument were also effectively destroyed—about 10 years' work! The animator Mary Ellen Bute shot a reel of documentary footage that preserves about 10 minutes of short excerpts from Dockum's performance on the Guggenheim *MobilColor*, enough to show that it really did perform complex layered imagery. Dockum spent the rest of his life, into the mid-1970s, building another *MobilColor*, and composing about 15 minutes of material that can still be performed on it, at his old studio in Altadena. While these compositions are brief, they show three diverse types of imagery—geometric forms, vibrating dot patterns, and soft sensuous trails—and above all demonstrate why someone would want to go to all this trouble when film and slide projections are so simple: the light intensity from the *MobilColor* is quite simply astonishing, the vivid shapes and colors magically hang in the darkness with a 'living' glow more 'real' than any image projected through cinema" [Moritz 1993].

### 2.1.2. Abstract Film

Many other innovators designed optomechanical systems for performing visual music; an extensive chronology of these individuals and their instruments appears in Appendix A, *A Timeline of Instruments for Color-Music Performance*. Some of these

systems incorporated or produced *both* sound and light, such as Castel's *Clavecin Oculaire* or the machines of Kastner, Greenewalt, Laszlo, Cross, Land, and Spiegel. Other designers, such as Wilfred, Fischinger, Dockum, Bishop, Rimington, Baranoff-Rossiné, Pesanek, and Klein, sought to explore the forms that a visual *analogy to music* could take, and instead chose to construct machines which were strictly intended for the performance of dynamic visuals.

While these innovators developed "real-time" tools for the performance of visual music, other pioneers composed elaborate visual statements in the off-line laboratory of the animation studio. Influenced by the twin births of cinema and visual Modernism in the first two decades of the Twentieth century—and possessing deeply held beliefs in a "universal language of abstract form"—animators like Walter Ruttman, Viking Eggeling, Oskar Fischinger, Len Lye, and Norman McLaren began systematic studies of abstract temporal composition in order to uncover "the rules of a plastic counterpoint" [Russett and Starr 1988]. Landmark events in abstract cinema included the 1921 Frankfurt run of Ruttmann's short *Lichtspiel Opus I*, thought to have been the first screening ever of an abstract film for a general audience [Russett and Starr 1988], and the 1924 release of Eggeling's *Diagonal Symphony*, which was the first entirely abstract film. By the late 1930's, Oskar Fischinger had established himself as the indisputable master of the form, having invented or refined literally dozens of animation techniques. The painstakingly constructed efforts of these and other artists dramatically expanded the language and vocabulary of dynamic visual form, at a time when the language of cinematic montage itself was only beginning to be created and understood.

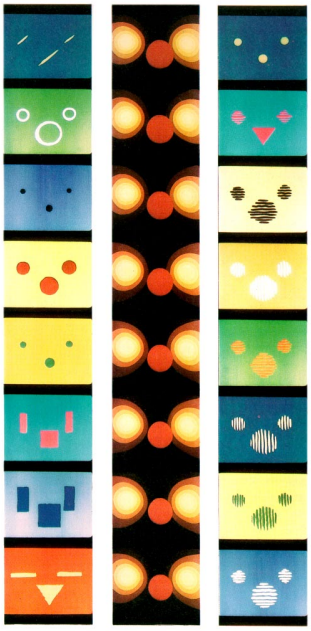


Figure 10. Frames from Oscar Fischinger's abstract film, *Radio-Dynamics* (1938).

The history of abstract cinema is too great to describe here, and has been extensively covered in, for example, [Russett and Starr 1988], [Moritz 1993] and [Moritz 1997]. Nevertheless, it is important to mention here that the visual languages developed by the abstract animators have been a tremendous source of inspiration to the work presented in this thesis. An example of such an inspiration is the cinematic vocabulary developed by the New Zealand animator Len Lye (active 1930-1960), who explored "cameraless animation" techniques such as drawing, scratching and painting directly on celluloid. Lye's work vaults the gulf between the vitality of performance and the precision of composition, for even though his movies were meticulously constructed in his animation studio, his process of improvisation

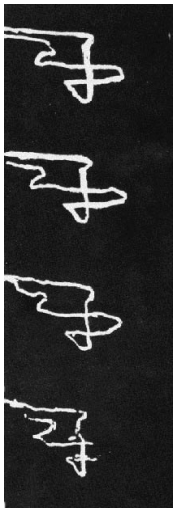


Figure 11. Frames from Len Lye's abstract film, *Free Radicals* (1957).

survives on-screen in frenetic and biomorphic works that are a direct connection to his own experience, thought and mark-making [Snibbe and Levin 2000].

### 2.1.3. Optical Soundtrack Techniques

Thus far we have focused on three important pre-computational means for the production of animated visuals and audiovisual compositions: color performance instruments, color-music systems, and abstract films. Our last stop in this section will be a brief mention of the *optical soundtrack techniques* in which certain filmmakers used entirely visual means to synthesize accompanying sounds.

Once again, Oskar Fischinger was one of the earliest and most masterful pioneers of the technique. Fischinger and his assistants painted sound waveforms on long sheets of paper he called "sound scrolls." By photographically exposing the optical soundtracks of the film to images of these scrolls, Fischinger was able to create wholly synthetic music to accompany his animation.

Figure 12. Oskar Fischinger with a "sound scroll" used in the optical soundtrack of one of his films.

While Fischinger drew individual waveforms by hand, the animator Norman McLaren, in Canada, developed a variety of template-based methods. McLaren created and catalogued dozens of index cards, each painted with a pattern of stripes whose spacings produced notes in the chromatic scale. He would then mask these stripes with cutout amplitude-envelope cards, in order to produce sounds with differing attacks and decays (Figure 13) In other experiments, McLaren dispensed with the cards and instead masked regions of a special image from which McLaren could produce any desired pitch (Figure 14).

In the early 1950's, the brothers John and James Whitney, a pair of California animators, devised an unusual technique in which "infrasonic pendulums" synthesized pure audio tones on optical soundtracks. Using mechanical components salvaged from decommissioned war machinery, the Whitneys constructed a system of pendulums which would periodically interrupt the light arriving at a film shutter. By slowly advancing the film past the shutter while the pendulums swung back and forth, the Whitneys were able to expose periodic bands of darkness and lightness onto the film's optical soundtrack. These bands would then produce audible sine tones when played back at a higher speed by the film projector. By using multiple pendulums of varying lengths, the Whitneys were able to generate chords of different tones.

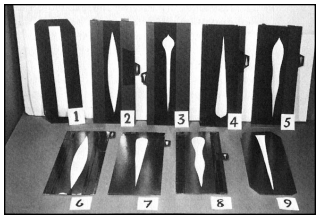



Figure 13. Norman McLaren created these template cards in order to generate sound "envelopes" in a film's optical soundtrack. Different shapes, for example, produce sounds with different length attacks and decays. From [Russet and Starr 1988].

Barry Spinello, an abstract animator active during the 1970's, followed in the footsteps of Fischinger, McLaren and the Whitneys with a related optical soundtrack technique which made use of Pres-Tone adhesive tapes. This material, also known as Ban-Day dots, consists of adhesive strips with various densities and gradations of half-tones printed on it, and was heavily used by advertising and graphic designers prior to the birth of desktop publishing. By assembling segments of Pres-Tone tapes into his optical soundtracks, Spinello was able to achieve a variety of interesting sonic textures and effects, such as gurgling, hissing and grainy noises [Russet and Starr 1988]. Spinello's technique is most notable for its contrast to those of his precursors, who seemed for the most part fixated on the synthesis of specific tones of precise frequencies. By shifting the level of granularity of his basic materials, Spinello was able to specify thousands of sound parameters with a single substance, thereby achieving sounds which would be nearly impossible to produce by hand-drawn means. His work is especially relevant to this thesis, for, as we shall see, his technique is essentially a type of "visual granular synthesis," akin to the methods used in my *Aurora* (Section 3.2.4).

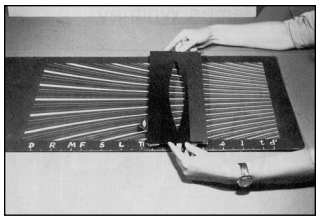


Figure 14. Here, McLaren fills an envelope template with a waveform of a specific pitch; the content of this envelope would then be photographed onto the film's optical soundtrack. By sliding the template from left to right across the converging stripes, McLaren was able to select different pitches. Note how the striped card has been marked in *solfege* (i.e. Do-Re-Mi, etc.). From [Russet and Starr 1988].

The optical soundtrack techniques developed by these innovators are important because they suggest a way in which visual patterns can be used, not to represent sound, but to directly and physically *generate* it: Although the optical drawings may be situated in a scorelike timeline, the optical soundtrack is not a score whose symbolic notations are read by a human, but an input to an optoelectric machine which automatically renders them into sound. This idea forms an important basis for the systems I present in the next chapter, many of which employ mechanized means for sonifying visual phenomena.

## 2.2. Visual Music in the Computational Domain

### 2.2.1. Advantages of the computer for visual music

Physical color organs are burdened by an inherent trade-off in their ability to yield specific versus general content [Snibbe and Levin, 2000]. The control of detailed or precise images requires a specificity of generative means, whereas the use of highly general means tends to produce amorphous and difficult-to-control results. To display the image of a triangle in the physical world, for example, requires a triangular chip of transparent material, or a triangular aperture—and that triangular element can do little else but make triangles. By projecting light through a tray of immiscible colored liquids, on the other hand, one can produce an infinity of outcomes, but its inchoate and complex results can be only vaguely directed. Computer technology has made it possible for visual music designers to transcend the limitations of physics, mechanics and optics, and overcome the specific/general conflict inherent in electromechanical and optomechanical visual instruments. One of the first artists to take advantage of these means was the California filmmaker John Whitney, who began his studies of computational dynamic form in 1960 after twenty years of producing animations optomechanically. Around the same time, Ivan Sutherland at MIT developed *SKETCHPAD*, the first software to emulate the natural process of drawing. Shortly thereafter, Myron Krueger made some of the most fundamental developments in the connection between interaction and computer graphics; his 1969 *VideoPlace*, for example, used information from motion capture to direct the animations of abstract forms [Krueger 1983]. *PAINT*, the first generic paint program, was developed in the mid-1970's by Richard Shoup and Alvy Ray Smith. Since that time, the expressive potential of real-time computer graphics have burgeoned considerably.

At the same time, electronic and computer music has burgeoned as well, spurred on by innovators eager to explore a realm of sound similarly unbound by the laws of physics. The first (and largest) synthesizer ever built was Thaddeus Cahill's massive electromechanical *Telharmonium*, built between 1897 and 1906. The advent of the transistor hastened the development of more lightweight "analog synthesis" techniques, developed by such pioneers as Karlheinz Stockhausen and Iannis Xennakis in the 1950's. The invention of the stored program electronic digital computer in the 1940's, however, truly opened the way for the

present era of sound synthesis [Roads 1996]. Since the first computational sound experiments of Max V. Matthews in 1957, dozens of sound synthesis techniques have been invented. The history of this field is vast, and is best left to other writers; it is enough to note that as many innovators have developed unique devices for composing, controlling and performing synthesized sound, as have developed the sound synthesis techniques themselves.

In the next sections, I discuss the ways in which the fields of computer graphics and electronic music have been brought together, with special attention to the paradigms of sound-image relationships that have come to populate this intersection. In the interests of space and precision, I have restricted myself to the discussion of audiovisual computer systems that are specifically intended for composition and/or performance.

### 2.2.2. Sound and the screen: strategies for sound/image relationships on the computer

The majority of computer visual interfaces for the control and representation of sound have been transpositions of conventional graphic solutions into the space of the computer screen. In particular, three principal metaphors for sound-image relationships have come to dominate the field of visually-orchestrated computer music: scores, control panels, and what I term interactive widgets. In the next sub-sections, I treat each of these strategies in turn, with special attention to the relationships between sound and image which they use, and their applicability to contexts of real-time performance.

#### 2.2.2.1. Score Displays

Alan Kay once declared music notation to be one of the ten most important innovations of the past millennium. Certainly it is one of the oldest and most common means of relating sound to a graphical representation. Originally developed by medieval monks as a method for "hinting" the pitches of chanted melodies, music notation eventually enabled a revolution in the structure of Western music itself, making it possible for complex, large-scale music to be performed, and yielding an attendant emergence of new musical roles, hierarchies, and performance instruments [Walters 1997].

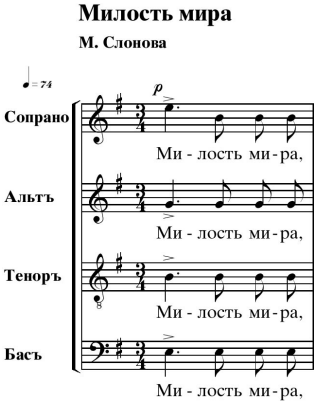



Figure 15. An example of standard music notation: the first measure of M. Slonov's "A Mercy of Peace."

Scores are generally two-dimensional *timeline* diagrams which operate by relating the dimension of time, along one axis, to some other dimension of sound, such as pitch or amplitude, on the other. In traditional music notation, there may be several parallel time axes (called staves), which make possible the synchronization of multiple simultaneous instrumentalists. In addition to Western music notation, other common examples of sound-timelines are waveform displays, player-piano scrolls, and spectrograms.

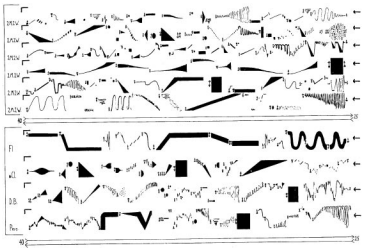


Figure 16. A page from the score of Carmine Pepe's *Plastic Containers*, illustrating his use of a personal and idiosyncratic notation system. From [Johnson 1978].

What these various sorts of timelines and diagrams share is a reliance on a coded language of graphical conventions in order to convey meaning. Once learned, this elaborate system of symbols and visual relationships, refined by generations of composers and typesetters, yields a remarkably efficient way of organizing and producing a large quantity and variety of musical events [Walters 1997]. Naturally, many composers in search of further expressive possibilities have experimented with alternative notation systems; some, like Carmine Pepe (Figure 16), have invented idiosyncratic and personal representations for various dimensions of sound. Others, like Karlheinz Stockhausen or J. Levine, have partially or wholly subverted the linear nature of the timeline itself. In Stockhausen's example below, the linear axis of time is no longer measured in absolute units of seconds or beats, but instead enumerates higher-level units ("events") of musical organization.

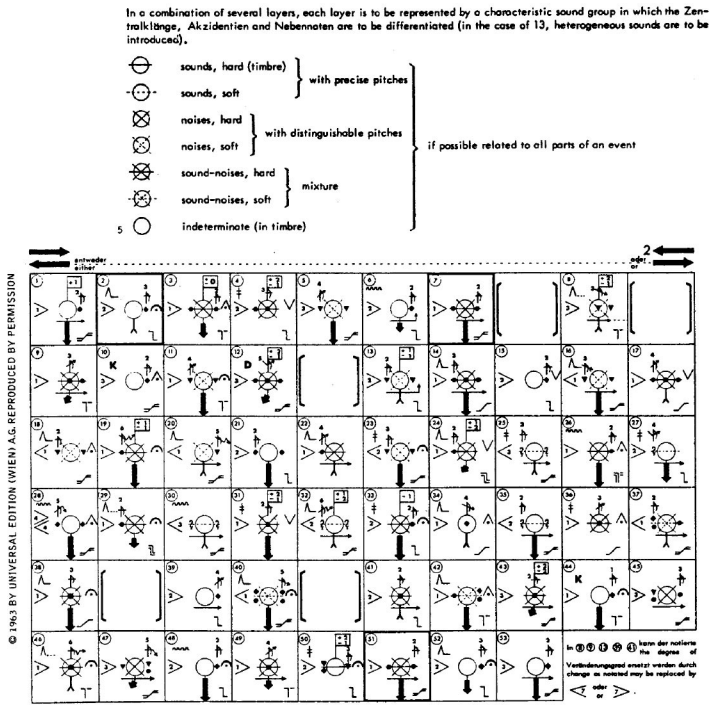



Figure 17. Part of the score from Karlheinz Stockhausen's *Plus-Minus* (1963). According to John Walters, the symbols used in the score are explained in seven pages of detailed instructions. "Each square signifies a musical event and the central open circle represents a *Zentralklang*, corresponding to one of eight chords written on a separate page" [Walters 1997].

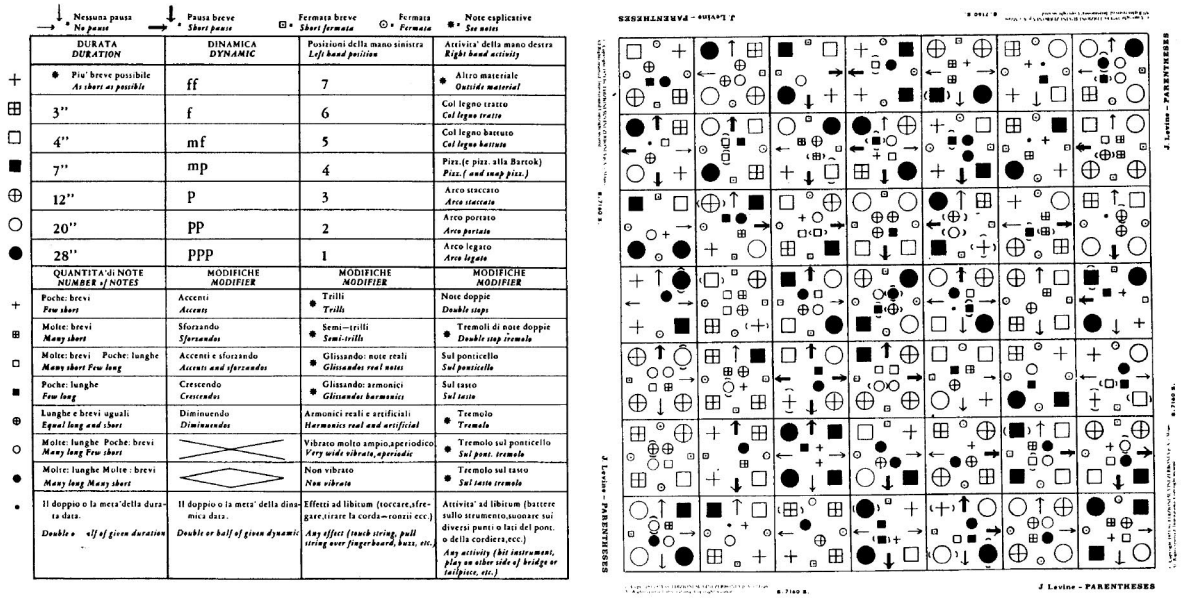
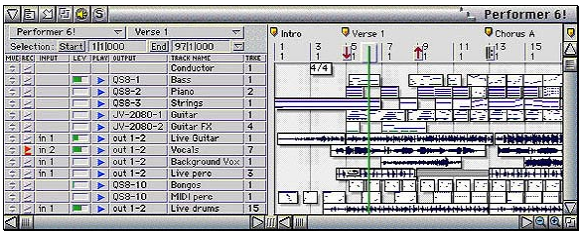



Figure 18. The score for J. Levine's *Parenthesis* [Johnson 1978]. Note the extensive instructional key on the left.

The score for J. Levine's *Parenthesis* does away with the linearity of a timeline altogether; in its place, the score substitutes a two-dimensional lattice of events, each node of which offers several subsequent event-possibilities in the squares adjacent to it. Despite their visual beauty, it is imperative to observe that neither of these examples can operate as a readable score without its accompanying, and highly detailed, chart of symbolic keys.

Despite the investment necessary to learn written or graphic languages of music notation, their use has become deeply ingrained in the daily practice of an enormous number of musicians around the world. The natural outcome of this is that score-based systems now predominate the field of visually-governed computer music. Thus the modern *sequencer*, the workhorse tool of nearly every electronic composer, wraps the functionality of a multi-track recording system around a backbone of one or more editable timeline displays. Many such systems now exist. Mark of the Unicorn's *Performer 6.0* sequencer, to take a representative example, offers three different views of musical information: standard music notation, digitized sound waveforms, and MIDI notes displayed on a so-called "piano roll" timeline [Mark of the Unicorn 2000].



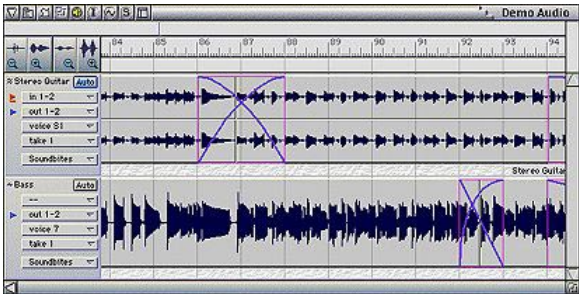
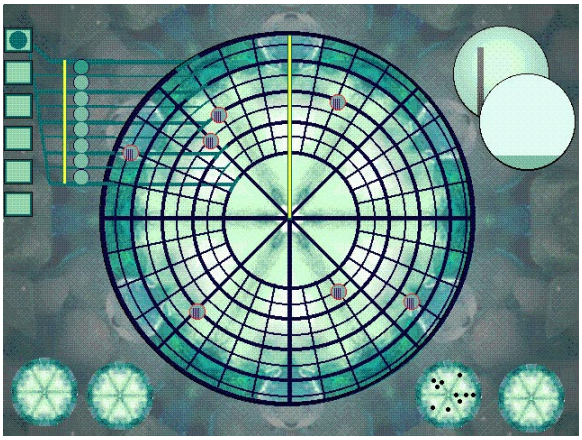
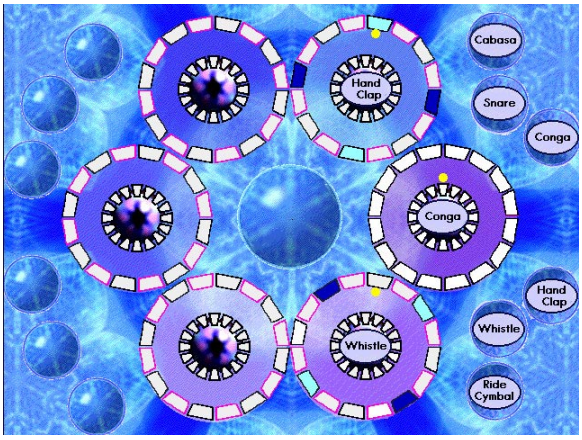




Figure 19. Two windows from Mark of the Unicorn's *Performer* sequencer, showing some of the available timeline views for musical information in a typical sequencer [Mark of the Unicorn 2000].

Some software designers have attempted to innovate within the timeline schema by permitting users to edit data *while the sequencer is playing* the information in that timeline. This solution, which dramatically tightens the iteration cycle of composing music, hybridizes the offline aspects of the sequencer's notation system with the real-time control of a performance instrument. The *Soundscapes* musical instruments, created at the Interval Research Corporation in 1995 in a project directed by Joy Mountford, embody this idea. These instruments, which wrap their timelines into one or more circles, were intended for the creation and performance of cyclical music patterns. As the user places markings around the perimeter of the circles, a current-time indicator arm sweeps around in the manner of a radar screen, triggering a MIDI event when it intersects one of the markings [Interval 1995]. Unfortunately, the visual interfaces of the Soundscapes instruments have been so encrusted with decorative eye-candy that the underlying structure of their sound-image relationship has been nearly obscured by irrelevant graphic information.

Figure 20. Two of the Interval *Soundscapes* instruments: *Web* (left) and *Shapes* (right) [Interval 1995]. A third instrument, *Orbits*, is not pictured.





Lukas Girling is a young British designer who has incorporated and extended the idea of dynamic scores in a series of elegantly spare but musically powerful interface prototypes. His *Granulator* instrument, developed at Interval Research Corporation in 1997, uses a stack of parallel looping timelines to control numerous parameters of a granular synthesizer. Each panel in the *Granulator* displays and controls the evolution of a different aspect of the synthesizer's sound, such as the strength of a lowpass filter or the pitch of the sound's constituent grains; users can draw new curves for these timelines. One interesting innovation of the *Granulator* is a panel which combines a traditional timeline with an input/output diagram, allowing the user to interactively specify the temporal evolution of a source soundfile's playback location.

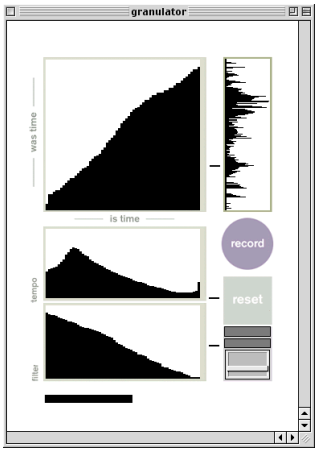



Figure 21. Lukas Girling's *Granulator* interface [Girling 1998].

When diagrammatic instruments are allowed to go unconfected, the relationship they establish between sound and image can be extremely tight. Many individuals are able to read music notation, or even speech spectrograms for that matter, as fluently as they can read English or French. Nevertheless, it is essential to remember that scores, timelines and diagrams, as forms of visual language, ultimately depend on the reader's internalization of a set of symbols, signs, or grammars whose origins are as arbitrary as any of those found in spoken language.

#### 2.2.2.2. Control-Panel Displays

A second pattern which has come to predominate the design of visual interfaces for electronic music is that of the *control panel*. Designers who use this pattern have set about to imitate or evoke the sound controls afforded by vintage analog synthesizers. These synthesizers were typically manufactured during the 1970's and are immediately recognizable by the several dozen knobs, dials, sliders and buttons which comprise their front panels. Analog synthesizers have an almost legendary appeal, not only because of their unique sound and sometimes quirky behavior, but also because their interfaces are completely laid bare, comprehensively viewable, and enjoyably manipulable.

Figure 22. A *Memorymoog* analog synthesizer, circa 1978.

With the advent of digital synthesizers in the early 1980's, interfaces for controlling musical parameters in keyboard synthesizers shifted from the use of directly-manipulable knobs, to tiny alphanumeric LCD screens with nested menu systems. While the digital synthesizers offered a wider range of sounds and greater reliability than the older synthesizers, many musicians

lamented the loss of the analog knobs they had found so ready-at-hand, expressive and responsive. When speed improvements in the mid-1990's finally made it possible for desktop computers to perform both professional-quality sound synthesis and color graphics, devotees of the old analog synthesizers responded by initiating a reactionary and nostalgic trend in synthesizer design: the on-screen imitation of knob-laden control panels. Many of the latest software synthesizers now resemble Koblo Software's *Vibra6000*, shown below.


Figure 23. The *Vibra6000* software synthesizer for the Macintosh, produced by Koblo Software. The Koblo web site advertises: "A knob for every parameter! Forget about tiny unreadable displays." [Koblo Software , 1999].

The *Vibra6000*'s use of instrumentally extraneous graphical elements—that is to say, visual elements which have no musical function—is modest by today's standards. The most baroque of the control-panel simulacra, as of this writing, is Propellerhead Software's *ReBirth RB-338*, designed specifically to imitate the sound and appearance of the *TB-303* Bass Line Synthesizer originally manufactured by Roland Corporation in 1981. The *ReBirth RB-338* puts more than two hundred little knobs at the control of the user's mouse. The human propensities for decoration and "personalization" being what they are, the

Propellerhead designers have even made it possible for users to wholly modify the graphic appearance ("skin") of the *RB-338*:

> "Here at Propellerhead we're crazy enough to let users take our precious ReBirth and redesign it any way they like. If you're skilled in graphic design and you have a bunch of cool drum samples you've always wanted to share - make a modification, mail it to us and maybe, just maybe, we will make sure it reaches every corner of the world." [Propellerhead Software, 1999].

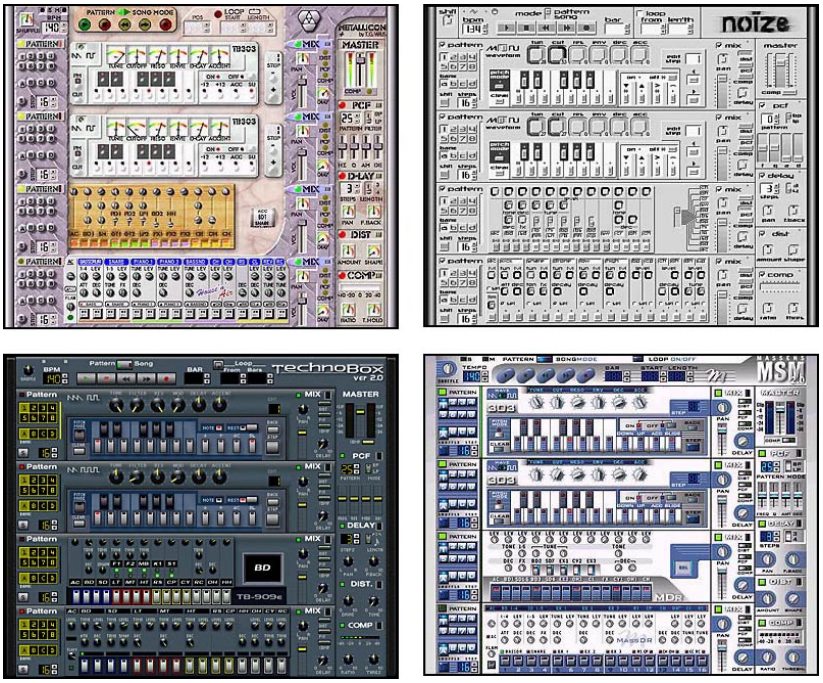



Figure 24. Four variations of the *ReBirth RB-338* by Propellerhead Software [Propellerhead Software, 1999]. Users can modify the appearance of the synthesizer by substituting their own bitmaps for the dials, buttons, etc.

Unfortunately, graphic synthesizers which use the control-panel schema replicate all of the *undesirable* aspects of multi-knob interfaces—such as their bewildering clutter, their confusing homogeneity, and their unobvious mapping from knobs to underlying sound parameters—and none of their positive aspects, such as their gratifying physical tactility, or their ability to be used by multiple hands simultaneously. Furthermore, because identical knobs are often assigned control of wholly dissimilar aspects of sound, control-panel graphics share a disadvantage with scores and diagrams: namely, that they must be "read" with the aid of a symbolic or textual key. We can conclude our discussion of control-panel displays, by observing that the ready interchangeability of the synthesizer's "skin" highlights the extreme degree to which sound and image are disconnected in the control-panel paradigm.

#### 2.2.2.3. "Interactive Widget" Displays

A third contemporary design pattern for screen-based computer music is built on the metaphor of a group of virtual objects (or "widgets") which can be manipulated, stretched, collided, etc. by a performer in order to shape or compose music. The foundation of this schema is an assumption that "a sound can be abstracted as an aural object" [Abbado 1988]. An application called *Sounder* by Jack Freudenheim—described by its author, for better or for worse, as a "musical lava lamp"—is a representative example of a software system which embodies this idea [Perpetual Music 1994]. In this software, small abstract animating sprites bounce around inside of a series of standard rectangular GUI windows. Whenever an object collides with the boundary of its window frame, it triggers a MIDI note on the computer's soundcard. Users can interact with *Sounder* by instantiating new sprites, assigning pitches and timbres to them, "throwing" them in new directions with the cursor, and modifying their periodic rhythms by adjusting the dimensions of their containing windows.

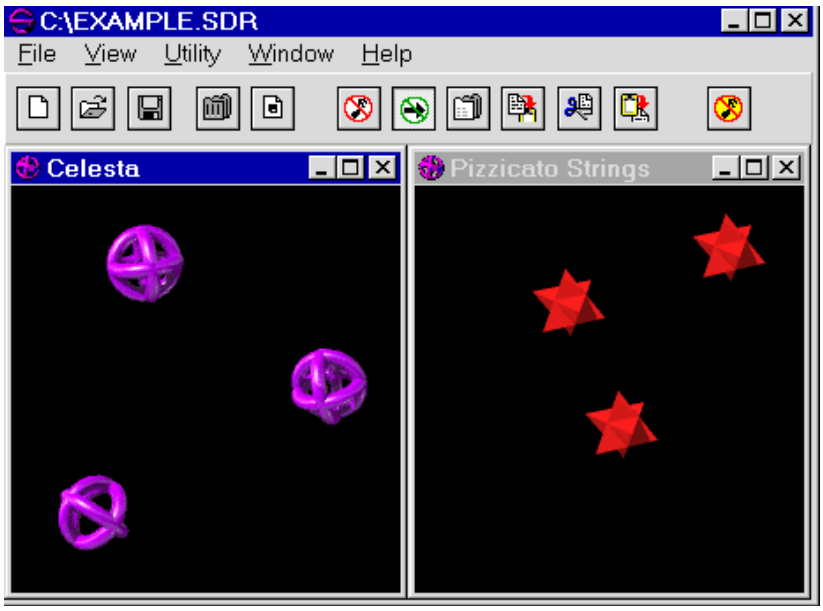



Figure 25. *Sounder* by Jack Freudenheim [Perpetual Music 1994].

*Sounder* is neither especially sophisticated in its visual design, nor terribly expressive in its musical affordances, since the results of its bouncing simulation are largely beyond the user's control. Lukas Girling's *Vector Field* instrument, developed at Interval Research Corporation in 1997, takes a step in the right direction by allowing its users to exert precise control over the *entire* trajectory of a flying widget. In Girling's work, performers use the

cursor to make modifications to the individual orientations and intensities of the elements in a two-dimensional field of vectors. These vectors then influence the flight path and spin of a small autonomous cube, whose ballistic motions across the plane are in turn mapped to certain control parameters (amplitude, filter strength, resonance, etc.) of a digitally-sampled audio loop [Girling 1998].

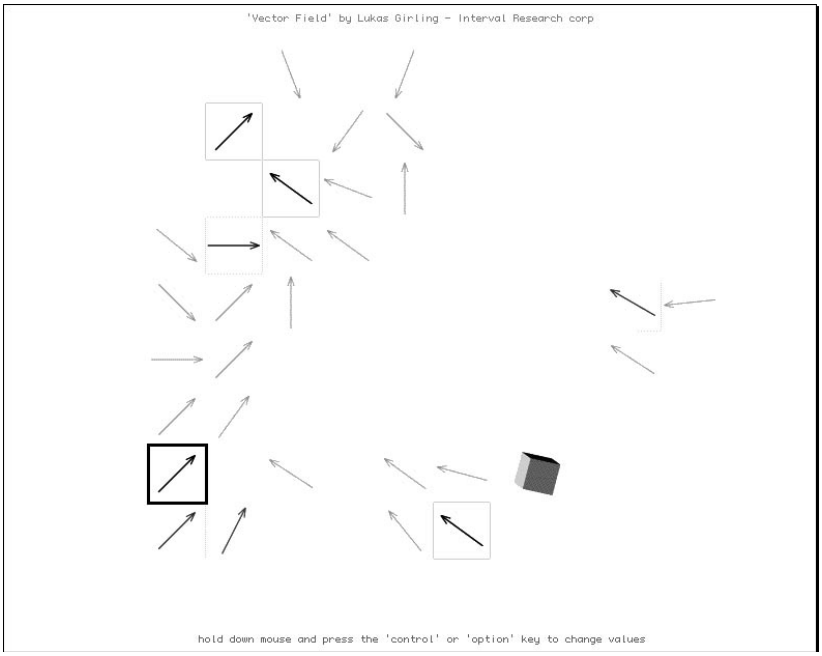



Figure 26. Lukas Girling's *Vector Field* interface [Girling 1998].

Users of Freudenheim's *Sounder* and Girling's *Vector Field* are restricted to discrete adjustment operations instead of the continuous, gestural operations which are typical of musical performance. *Sounder* and *Vector Field*, moreover, largely adopt an interaction model in which the user's discrete manipulations operate *indirectly* on the apparent agent of sound production: instead of modifying a sonic widget itself, the user instead manipulates some other property of the visual environment (such as its boundary, or its terrain), which in turn exerts forces on the sound-controlling widget.

Other designers, influenced by current work in "direct manipulation" interfaces [Baecker 1995], or perhaps taking a cue from the design of traditional, physical musical instruments, have created "interactive widget" interfaces which are both gesturally performable and directly manipulable. Reed Kram's *Transducer* software, created in 1997 in the Aesthetics and Computation

group at MIT, implements this by permitting its users to make continuous modifications directly to its cylindrical "Sound Objects." Kram describes his system thus:

> "*Transducer* is a digital system for live, audio-visual performance.... Each sound clip is visualized as a 'playable' cylinder of sound that can be manipulated both visually and aurally in real-time.... At first, the system presents a palette of cylindrical objects. As the user moves his or her mouse over each of the cylinders, he or she hears a sampled sound stream associated with that object. Each of the objects has a representative color and shape corresponding to the sound stream associated with it... In this way a single user or performer is able to build simultaneous visual and audio constructions in realtime. The user can examine interrelationships between multiple, diverse sound sources and a corresponding visual form." [Kram 1998].

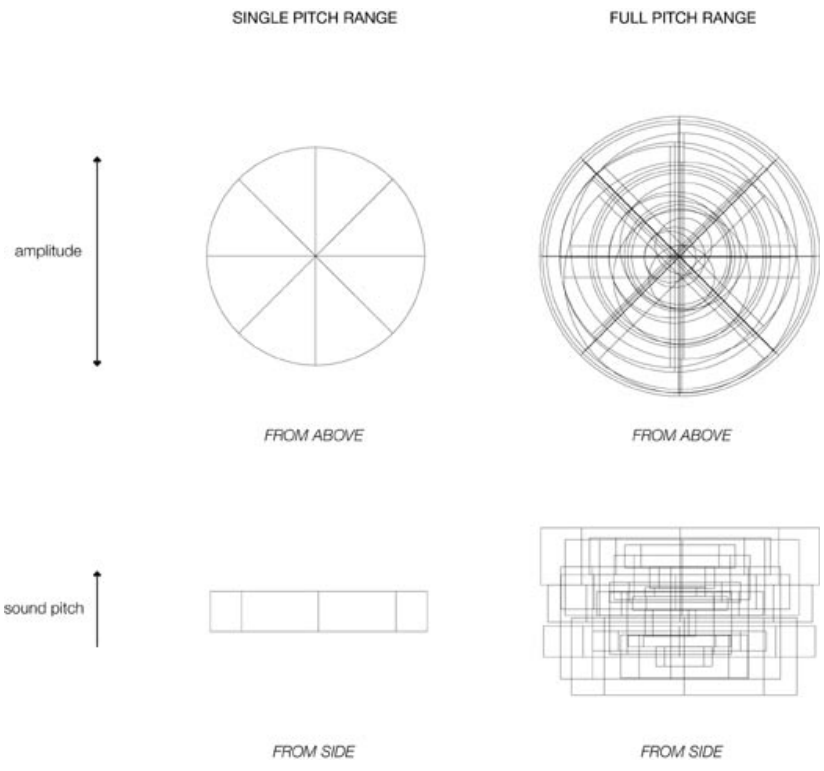



Figure 27. An explanatory diagram of the structure of the Sound Objects used in Reed Kram's *Transducer* instrument [Kram 1998].

*Transducer*'s incorporation of extremely straightforward yet arbitrary sound-image mappings, such as the relationship it establishes between a cylinder's height and a sound's pitch, give it a diagrammatic aspect not unlike the scores discussed previously. In theory, this restricted set of mappings should make the system easy to "read"; in reality, however, *Transducer*'s legibility is largely impaired by two of Kram's concomitant design choices: firstly, pitch and amplitude in *Transducer* are not represented as absolute quantities, but rather as ratios relative to a stored sound's original values. The effect of this is that samples which are heard at identical pitches may be represented by cylinders of entirely different heights, and vice versa. Secondly, Kram's strategy of representing all sounds as greenish-gray cylinders fails to generate visual analogies to sound at the right level, or at enough levels, of representation. It is impossible, for example, to distinguish the Sound Object for a spoken vocal timbre, from a Sound Object for a drum loop or a string section. The result is an instrument which substantially exchanges both musical legibility and visual interest for a dubious graphic uniformity.

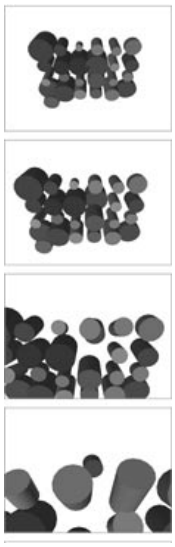


Figure 28. Frames from Reed Kram's *Transducer* instrument in use. [Kram 1998].

An interesting contrast to this can be found in the *Stretchable Music* software system developed by Pete Rice in the Hyperinstruments group of the MIT Media Laboratory [Rice 1998]. In Rice's work, each of a heterogeneous group of animated graphical objects represents a track or layer in a pre-composed, looping MIDI sequence. By gesturally pulling or stretching one of these objects, a user can create a continuous modification to some sonic property of a corresponding MIDI track, such as the filter cutoff in a "square-wave" synthesizer melody, or the amount of "breathiness" across a synthesized flute passage.

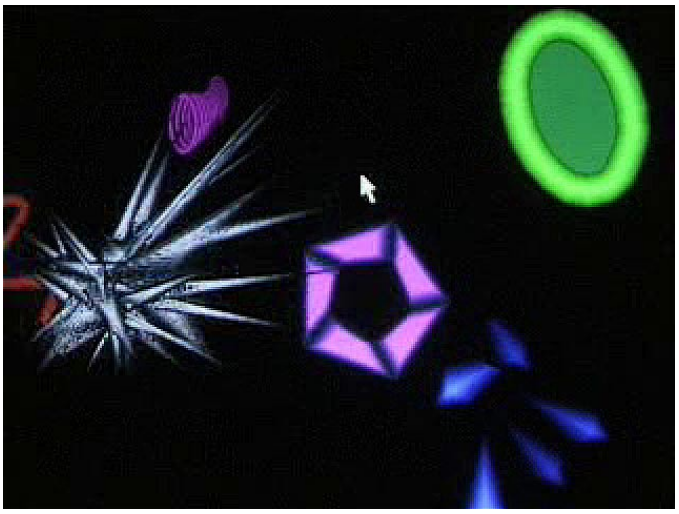


Figure 29. A screen capture from Pete Rice's *Stretchable Music* system in use [Rice 1998].

One of *Stretchable Music*'s particular strengths is its use of different widget forms to represent different audio layers. Rice establishes a personal yet consistent audiovisual context in which "rough" sounds (those with greater high-frequency audio content) correspond to "sharper" graphics (those with greater amounts of spatial high frequencies, e.g. sharp corners). Because Rice's sound/image mappings are motivated by these perceptual congruencies as well as a coherent aesthetic, they are generally successful: not only has Rice selected reasonable and imaginative mappings between specific graphic objects and musical layers, but also between any given object's dynamic visual properties and the unique axes of timbre which it controls. Rice's system, moreover, especially exemplifies the "interactive widget" schema insofar as its objects simulate the dynamic behavior of real-world physical objects. The expressive handles exposed to the user are not the mathematically and diagrammatically ideal "width" and "height" of Kram's platonic cylinders, but rather the "bounce" and "twitch" of the plausible furnishings of a physical world.

In the *Stretchable Music* system, the melodies, harmonies, rhythms, timbre assignments, and temporal structures of its music are all predetermined and pre-composed by Rice. By curtailing his users' influence to the timbral tweaking of otherwise immutable musical material, Rice is able to guarantee that his system always "sounds good": wrong notes or misplaced beats, for example, simply can't happen. Unfortunately, Rice's trade-off also substantially attenuates the depth of active engagement that a user of his system can experience: because his users have little at stake to lose, there is also little for them to gain (except, of course, an appreciation of Rice's composition). This confinement of creative options is compounded by the fact that the graphic objects in the *Stretchable Music* system are just as immutable as the musical MIDI sequences: although the visual widgets may be squashed and stretched through temporary affine transforms and other simple adjustments, their quintessential character—established by Rice, in many cases, through a set of cached bitmap images—cannot be transformed or camouflaged by the user.

Because so many aspects of the *Stretchable Music* system have been pre-composed by Rice, it is reasonable to wonder whether his system can be considered a musical instrument at all. To his credit, Rice acknowledges the limitations imposed on the users of his system and only positions *Stretchable Music* as an

"interactive composition." By inserting expressive handles into an otherwise unchanging piece of music, Rice believes that he is able to add "new levels of engagement to the continuum of musical experience previously polarized into active performers and passive listeners" [Rice 1998]. Although this is a worthwhile goal, Rice's system, and for that matter Girling's *Vector Field* and Kram's *Transducer*, fail to use pre-recorded sound materials (and visual materials) in such a way as to overcome their exhaustibility. As a result these systems, while promising infinite possibilities, become little more than mixing consoles for somebody else's tunes.

The most common disadvantage of "Interactive Widget" systems is that their canned ingredients, all too inevitably, yield canned results. The problem is fundamental and has to do with the *granularity of control* such systems afford: in general, performance systems whose interactions are predicated on the arrangement or modulation of "high-level" sonic events (e.g. entire musical passages and macrotemporal audio samples) and/or high-level graphic phenomena (e.g. predefined geometries and images), restrict users to performance experiences which are ultimately exhaustible, or shallow, or both.

This section has dealt with the use of visual interfaces for controlling sound on the computer. In the next section, I examine a parallel trend in the recent history of visual music, namely the ways in which the computer has been used to extend the tradition of systems developed by Wilfred, Fischinger and Dockum—as a performance medium for dynamic visuals.

### 2.2.3. Systems for Visual Performance on the Computer

The phrase "visual music" has enjoyed multiple meanings over the last few centuries. For some artists and inventors, it has referred to the products of a synæsthetic medium in which complimentary sounds and visuals are combined into a holistic unity. Examples of systems to which this understanding of the term might apply are Castel's *Ocular Clavichord*, the Interval *Soundscapes* score-systems, and Pete Rice's *Stretchable Music* widget-system. Within the umbrella of "visual music," however, lurks a second interpretation which, interestingly enough, refers to a strictly silent form. This understanding of "visual music" has stood for the possibility of a dynamic, strictly visual

medium whose temporal sophistication is equal to that of traditional music. The silent, optoelectric and electromechanical performance systems developed by Thomas Wilfred, Oskar Fischinger and Charles Dockum were all designed with this latter interpretation of "visual music" in mind. In this section of the thesis, I examine some of the ways in which this silent form of visual music has intersected with the affordances of computation.

The systems I discuss here all permit a user to gesturally create and perform, in one way or another, pure, animated abstract graphics. Of course, the space of all human gestures is much vaster than the restricted and digitized set of movements to which these systems respond. For the purposes of this discussion, and for this thesis generally, I restrict my definition of the term *gesture* to mean the combination of discrete and continuous movements, deliberately performed by the hands, in relation to or in combination with some markmaking medium or device.

Natural materials and media in the physical world excel at transforming the traces of gesture into richly textured, expressive marks. The computer's low-resolution screen, by contrast—physically displaced from the user's hand and mouse—is a poor subsitute. Nevertheless, the computer's electronic display offers unique affordances for gestural performance systems, such as temporal dynamics, state transitions and conditional testing, and models and simulations free from the traditional laws of physics. As we shall come to see, the software applications discussed in this section all use some form of *gestural augmentation*, based on these affordances, to produce considerably expressive new media.

#### 2.2.3.1. Paul Haeberli's *DynaDraw*

In 1989, the graphics researcher Paul Haeberli developed *DynaDraw*, a drawing program in which the user's gestural movements are augmented by an elastic physical simulation. According to Haeberli,

> "The program *DynaDraw* implements a dynamic drawing technique that applies a simple filter to mouse positions. Here the brush is modeled as a physical object with mass, velocity and friction. The mouse pulls on the brush with a synthetic rubber band. By changing the amount of friction and mass, various kinds of strokes can be made. This kind of dynamic filtering makes it easy to create smooth, consistent calligraphic strokes." [Haeberli 1989]

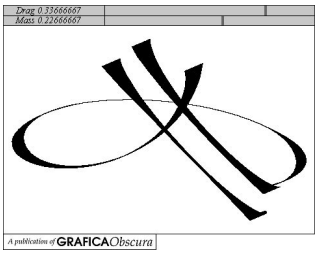



Figure 30. Paul Haeberli's *DynaDraw* [Haeberli 1989].

The chief contribution of *DynaDraw* is the idea that a user's ink can be *augmented by a physical simulation*. By interposing a virtual spring between the user's cursor and the nib of the virtual pen, Haeberli creates dynamisms which are both startlingly fresh yet comfortably familiar. In the process, he transforms a simple static paint program into a wholly new medium whose products and process are not only uniquely temporal, but are also evocative of real-world behaviors.

#### 2.2.3.2. John Maeda: *Timepaint*, *A-Paint*, and *CMYK Dance*

In the early 1990's, John Maeda developed a series of interactive software systems—*Timepaint*, *A-Paint*, and *Process Color Dance*—to study the ways in which virtual "ink" could be used to perform and display dynamic computations. Maeda's *Timepaint* is a delicate illustration of the dynamic process by which apparently static marks are made: by extending our view of a gesture's temporal record into the third dimension, Maeda's work can flip between a flat animated composition and a volumetric diagram of temporality. Maeda writes:

> "*Timepaint* ... [presents] a time-lapse display of mouse motion as a visual experience in two and a half dimensions. Multiple strokes can be programmed and colored to produce wisp-like dynamic imagery which fades into oblivion. *Timepaint* illustrates not just the lapse of a single frame of time, but the continuum of time in which the computer and user coexist" [Maeda 1995].

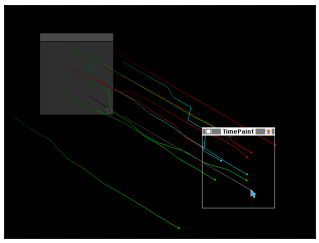



Figure 31. Maeda's *Timepaint* [Maeda 1995].

The kinds of animated compositions made possible by *Timepaint* are highly constrained; all compositions, for example, are strictly composed of moving dots and their temporal trails. Although this particular design decision places strict limits on what can be expressed in the medium, it greatly enhances the effectiveness of Maeda's temporal visualization. In his *A-Paint* and *Process Color Dance*, by contrast, Maeda instead emphasizes the affordances of an expanded visual vocabulary, and these systems offer commensurately greater expressive possibilities.

Maeda's *A-Paint* is a programming system that uses a model of "intelligent ink" to enable the designer to define inks that react

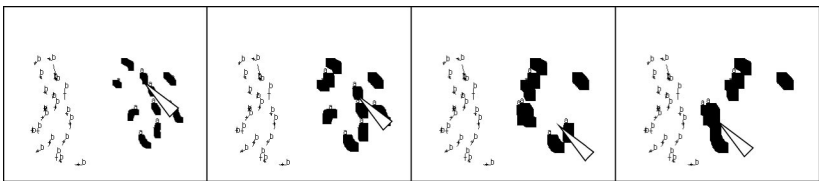


Figure 32. John Maeda's *A-Paint* [Maeda 1995].

to conditions in time, space and the user. This "intelligence" takes the form of constraints, and other functional dependencies between visual elements, that the user can specify at run-time. Strokes painted in "ink **A**" can be programmed to grow in thickness over time, move leftwards when near the cursor, and decrease in thickness when near strokes made in "ink **B**".

Maeda explored these themes once more in *Process Color Dance*, in which rectangular regions of cyan, magenta, yellow and black are used to create reactive color compositions. The most important contribution of Maeda's three systems, from the point of view of this thesis, is the idea that the ink of a user's marks can be temporal, reactive, and *augmented by an intelligence* which imbues it with interesting associated behaviors.

#### 2.2.3.3. Scott Snibbe's *Motion Phone* and *Dynamic Systems Series*

Scott Snibbe is an artist and engineer who has made particularly important developments in the field of software systems for interactive abstract animation. Snibbe has developed four works which are especially relevant to this thesis: *Motion Phone*, an application for interactively authoring dynamic animations based on recorded gesture data, and the three works in his *Dynamic Systems Series,* which explore the ways in which various software-based augmentations to user's gestures can aid in the design of animated abstraction systems. Snibbe was also instrumental in the early development of this thesis work, and collaborated on three of the interactive systems (*Streamer*, *Escargogolator*, and *Polygona Nervosa*) described in the next chapter, *Design Experiments*.

Scott Snibbe's *Motion Phone*, developed between 1991 and 1995, is an application for painting and recording abstract animations, and is an especially good example of a purely visual, expressive instrument. In this system, the user can record motion paths for a variety of simple shapes, such as trains of circles, squares, and triangles. According to Snibbe,

> The *Motion Phone* is an experiment in pure visual communication. It 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. The program draws its inspiration from abstract film and uses its language of two-dimensional animated shape and color. The quality of work created with this tool is strikingly human—in stark comparison to the work created with most computer art and animation programs today.

Figure 33. A screenshot from the *Motion Phone* in use [Snibbe 1996].

> The *Motion Phone* is a program which runs on a graphics workstation. When first approached, the program presents palettes of colors and shapes and a wide blank canvas. When a user draws upon this canvas the speed and location of his marks are entered into a digital animation loop. By pressing on the keyboard or on the graphics tablet, the shape, size and color of the marks can be simultaneously changed. As he continues to draw, his marks are added into the same animation loop, allowing him to sequentially layer multiple rhythms of form and color. [Snibbe 1996]

As with Maeda's *Timepaint*, the *Motion Phone* produces animated compositions from digitized recordings of gestures. Nevertheless, there is an important difference between the two applications. Whereas Maeda's intent in *Timepaint* is to illustrate the implicit temporality of gestural markmaking, Snibbe's goal is to provide a gestural tool for creating explicitly temporal compositions. Put another way, *Timepaint* is a deconstructivist and Rationalist visualization, while the *Motion Phone* presents a constructivist and Romantic medium. It follows that the most important contribution of the *Motion Phone* is not that it enables animated constructions to be created from captured gestures, but that it situates this endeavor in the context of a full-featured tool for doing so.

Snibbe's *Dynamic Systems Series* is a set of three applications which explore further ways in which graphic systems can computationally augment human movement. According to Snibbe, "each work in the series is embodied as a dynamic system—a model of natural, mathematical or algorithmic nature.

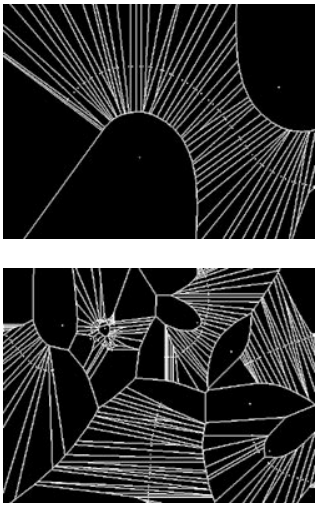
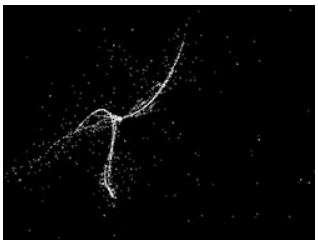


Figure 34. Stills from Scott Snibbe's *Bubbleharp* [Snibbe 1998].

Interacting with the system consists of reacting to and learning the system. The pieces are meant to provide an immediate sensation of touching an immaterial, but 'natural' world with consistent and predictable reactions, but infinite variety" [Snibbe 1998]. One system in the series, the *Bubbleharp*, constructs a Voronoi diagram from the user's movements. This kind of diagram is a common analysis tool in computer vision and computational geometry, and has many analogies in nature, such as the shape of bubbles or animal territories; it is defined as the set of regions in the plane, given a set of site-points, such that each region bounds the section of the plane which is closer to its corresponding site-point than any other region. In Snibbe's system, the user deposits and records paths for animated site-points in the plane, around each of which a "bubble" region forms. The *Bubbleharp* system has the remarkable property that, owing to the nature of Voronoi diagrams, bubbles placed in certain locations will crowd the plane, while other bubbles will free up empty space.

The remaining two applications in Snibbe's *Dynamic Systems Series* explore other varieties of computational augmentation. While the *Bubbleharp* augments gestures with a geometric construction, Snibbe's *Lazy Line* augments human movement with a generalized kernel-based filter, and his *Gravilux*, like Haeberli's *DynaDraw*, augments gesture with a physical simulation. In *Lazy Line*, the user may apply one of several simple digital filters to a line as it is drawn; if the filter is a lowpass filter, for example, the line is smoothed, while a highpass filter exaggerates wiggles. Of *Lazy Line*, Snibbe writes: "By passing a 3-element kernel filter over a line while you are drawing it, it is possible to add character to the line, while admittedly distorting the artist's form. The purpose of the tool is the process of interacting with this filter, rather than the final drawing" [Snibbe 1998].

Figure 35. Scott Snibbe's *Gravilux* [Snibbe 1998].



Snibbe's *Gravilux* places the user's cursor into a simplified physical simulation of gravity. In this system, the user applies gravity-like forces to a dense field of points; attracted to the cursor by a classic inverse-square force formula, the points can be teased and torqued in various ways, "slingshotting" the cursor when they pass too closely. In this way, the user is able to learn a little about what it might be like to "paint with stars" [Snibbe 1998].

Snibbe's works represent a wide variety of deeply engaging, expressive and playful graphic systems. These works reveal their author to be especially sensitive to the interactive opportunities and personality latent in simple yet carefully-chosen mathematical algorithms. In many ways, the experiments described in this thesis carry on the spirit of these works, by extending this kind of visual performance system into more elaborated and personalized surfaces, and—critically—into the additional domain of sound.

### 2.2.4. A Paint-Program/Sequencer: Toshio Iwai's *Music Insects*

One audiovisual software system, Toshio Iwai's *Music Insects*, is in a category by itself. Developed in 1991 for Nintendo and the San Francisco Exploratorium, and released as the commercial product *SimTunes* by Maxis Software in 1996, *Music Insects* is a hybrid of a classic, *MacPaint*-style paint program, with an interactively-modifiable sequencer. In this inventive application, a user places a variety of fat, colored "pixels" on the screen. These colored squares form a musical "score," which is sonified by several animated "insects" which crawl across the surface of the canvas. When a bug crosses a colored square, it produces a note whose pitch has been mapped to the square's color; each bug has its own instrumental timbre with which it sonifies the squares. The user can add, modify and delete pixels while the bugs are engaged in performing the score. Additional sophistication is possible through the use of certain specially-colored pixels, which have the effect of rotating or reversing the bugs which touch them. Using these special pixels, the user can cause the bugs to create looping rhythms, phasing polyrhythms, and complex passages which seem to never repeat at all. And, at the same time of course, the user of *Music Insects* is also authoring an image.


Figure 36. Toshio Iwai's *Music Insects*, installed at the San Francisco Exploratorium.

Of all the audiovisual performance systems described in this chapter, Iwai's *Music Insects* comes closest to offering a balanced solution for the simultaneous authoring of image and sound. Iwai overcomes many of the problems associated with diagrammatic scores, for example, through the use of his animated bugs, which act as self-revealing and self-explanatory "playback heads" for the sound (similar to the current-time indicators in the Interval *Soundscapes* instruments and Lukas Girling's *Granulator*). Because the system's score-elements are reductionist pixels as opposed to well-articulated symbols, moreover, the granularity of the visual substance is just right for the creation of abstract or representational images, and the visual output of the system may be read equally well as a painting or a score.

From the perspective of this thesis, the only shortcoming of Iwai's system is that its visual output is static imagery. In my opinion, it is an odd incongruence that the system's sound should be a dynamic phenomenon, while its visualization is static. In the next section, I introduce a new interface metaphor for audiovisual performance instruments, in which both the aural and visual output dimensions are assumed to be dynamic from the outset.

## 2.3. A New Interface Metaphor

### 2.3.1. Desiderata for a Color-Music Performance System

From an examination of the strengths and weaknesses of the many systems described earlier, I have derived a set of design goals which, if completely satisfied, would yield an audiovisual instrument of unparalleled expressivity. Taken together, these goals define a system with the following properties:

- The system makes possible the creation and performance of dynamic imagery and sound, simultaneously, in real-time.
- The system's results are inexhaustible and extremely variable, yet deeply plastic.
- The system's sonic and visual dimensions are commensurately malleable.
- The system eschews the incorporation, to the greatest extent possible, of the arbitrary conventions and idioms of established visual languages, and instead permits the performer to create or superimpose her own.
- The system's basic principles of operation are easy to deduce, while, at the same time, sophisticated expressions are possible and mastery is elusive.

In the next five subsections, each of these goals are treated in turn.

#### 2.3.1.1. Simultaneous Dynamic Image and Sound

"Color Music" is a broad term which has been used to refer to both the "music-like" display of silent time-based imagery, as well as the combined display of image and sound together. Quite a number of the systems and devices discussed previously fall into the former asonic category, ranging from Wilfred's

*Clavilux* and Fischinger's *Lumigraph* to Snibbe's *Motion Phone* and Maeda's *TimePaint*. The history of endeavors which mix sound and abstract image together, however, has been deeply intertwined with these former systems, and includes examples like Castel's *Clavecin Oculaire*, Kastner's *Pyrophone*, and a great deal of abstract film. Both are established approaches to "color music"; the emphasis of this thesis is on the second interpretation of color music, and takes as its foremost goal the design of systems for the real-time creation and performance of image and sound together, with the added understanding that the imagery is itself time-based and dynamic.

#### 2.3.1.2. Inexhaustible, Yet Deeply Plastic Results.

To whatever extent we might distinguish a given performance instrument from an individual composition yielded by that instrument, we acknowledge that we expect our instrument to be able to yield an even larger repertoire of compositions or performances, of which any given composition is only one. Put another way, a feature of a successful instrument is that its results are inexhaustible and extremely variable, insofar as it can afford many different kinds of compositions, and is sensitive to subtle features of a user's performance. The justification for this measure of success is straightforward: when a system's possibilities are easily or quickly exhausted by a user, the user gets bored. In the field of color-music instruments, the use of canned audiovisual materials, such as pre-prepared audio loops, cutout cardboard templates, or bitmapped sprites, is one of the likeliest indications that a system's expressivity is fundamentally limited.

Unfortunately, it is not merely enough to design a system with widely variable results, if those results are difficult or impossible to control. Consider one of the most popular color-music tools of the '60's psychedelic lightshows, a tray of colored oil blobs mixed with water: although it can yield an unlimited variety of compositions, it is difficult to understand how the details of any one of these compositions might reflect the aesthetic and expressive choices of a human performer. Thus we see that a system's inexhaustibility must be balanced by a deep plasticity, wherein the number of degrees of freedom is closely matched to the number of controls or expressive handles.

#### 2.3.1.3. Sound and Image of Commensurate Malleability.

It is a regrettable circumstance that most of the systems which use image and sound together have focused on only one of these dimensions, to the detriment of the other. This is certainly the case, for example, with the computational score-systems, control panels and interactive "widgets" described in section 2.2, which place the image in a subservient role to the sound: although the sound may be extremely malleable, as in the case of the *RB-338* control panel, the imagery is rigidly constrained by a strict visual language and a pre-determined formal design. Although the user of such a system applies simultaneous modifications to both the visual and sonic aspects, it would be too much to say that the visual aspect is a creation of the user's performance. Likewise, although some of the older keyboard-based systems, such as Castel's *Clavecin Oculaire* and its many derivatives (see *Appendix A*) afforded a reasonably fluid control of colored lights, the complexity of the resulting imagery undoubtedly paled in comparison to the highly evolved musical structures which could be played on the keyboards' strings or pipes. The problem with these systems is that the expressive capacities of their aural and visual dimensions are unequal and unbalanced.

It is straightforward to imagine even more systems which control sound from a GUI, or which have visuals which react to sound. It is more challenging, however, to propose that a successful audiovisual instrument ought to yield equally expressive performances in the image and sound domains. To do this, I put forth the goal that the audio and visual dimensions of such an instrument not only be deeply malleable, but commensurately malleable as well.

#### 2.3.1.4. Eschewing Conventionalized Visual Languages

Codified visual languages, such as scores and diagrams, often require a considerable learning phase before they can be used well. In particular, any performance system whose rules, mappings or conventions must be "looked up" or memorized from some "instructional key" is a system which burdens the expressing mind with the cognitive load of translation. In seeking to construct systems which are as immediately usable as possible, I have set the goal of eschewing such conventionalized mappings, and instead rely on the user's perceptual system, as much as possible, to intuit and interpret the system's rules.

#### 2.3.1.5. Instantly Knowable, Indefinitely Masterable Interface.

Most software systems are either easy to learn, or extremely powerful. Rarely are they both, for to be so demands that their rules of operation be simple, yet afford a boundless space of possible outcomes. This is difficult, and nearly contradictory. Nevertheless, there exist real-world exemplars of such systems, such as the piano and the pencil, which come close to meeting this goal. Although any four-year-old can discover their basic principles of operation, it is common for an adult to spend many years practicing these tools, and yet still feel that more mastery is possible or that more compositions remain to be expressed through them. Such systems, moreover, have the extraordinary property that an individual may eventually, through their use, discover or reveal a unique and personal voice in that medium. We all have our own spatio-temporal signatures, our own unique ways of moving through space; successful instruments bring the character of these traces into relief and reflect them back to us.

As a goal for an audiovisual performance instrument, the expression in the title of this subsection—"instantly knowable, indefinitely masterable"—is inherently unobtainable, hyperbolic, and contradictory. It is, notwithstanding, one of the most essential goals of this thesis. All of the work which follows is addressed to the design of audiovisual instruments that possess these qualities, of simplicity and transparency, balanced by possibility and sensitivity.

### 2.3.2. A Painterly Interface Metaphor

To meet the goals stated above, I introduce a new interface paradigm for audiovisual performance instruments. This metaphor is based on the idea of an inexhaustible, extremely variable, dynamic, audiovisual substance which can be freely "painted," manipulated and deleted in a free-form, non-diagrammatic context. According to this scheme, a user creates gestural, painterly marks in a two-dimensional input field, using an electronic drawing device such as a Wacom tablet or mouse. These marks are treated as the input to digital signal analysis algorithms, filtering algorithms, and computer simulations. The outputs of these algorithms are then visually interpreted by a graphics synthesizer, and also sonified by an audio synthesizer. Ideally, the mappings which relate the properties of the gestures to their sonifications and visualizations are perceptually

Figure 37. Jackson Pollock's studio, East Hampton, 1950. Pollock manipulates a free-form, plastic substance.

motivated, and do not rely on a codified visual or textual language for interpretation. I refer to such a system as "painterly" because I have elected to base its *process* in the act of *mark-making*, in which a gesture is made with respect to some material—as opposed to other domains of gesture, such as sign language or dance—and because part of the *product* of this mark-making is, beyond the performance of the mark-making itself, a *two-dimensional image*.

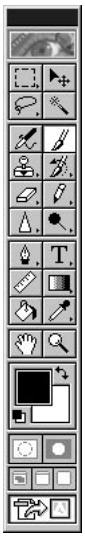


Figure 38. The *Photoshop* tool palette. *Photoshop*'s inkpot is bottomless.

The individual terms of the painted-substance scheme each bear some further explanation. The audiovisual substance can be said to "inexhaustible," for example, because the user can place as much of it on the canvas as they please, in the same way that there is no bottom to the *Photoshop* inkpot, nor theoretic limit to the number of daubs of paint which one can place on a canvas.

The "variability" of the scheme is owed to the extremely fine granularity with which the user's gestural marks are represented. By using a high-resolution, continuous representation of gesture to begin with, and mapping the qualities of the user's mark to the control parameters of extremely low-level image and sound synthesis algorithms, its expressive details can be preserved and reflected in the final output. This is, of course, stated with the understanding that *any* digitized representation of human movement is necessarily lossy, as it reduces both the fidelity and dimensionality of the represented gesture.

How is the painterly schema for audiovisual performance instruments related to its predecessors? Clearly, it inherits from many of them. It is similar to score-based systems, for example, insofar as it allows an unlimited amount of audiovisual material to be deposited on the canvas; it differs, however, because the new schema does not situate this material along a set of coordinate axes, but rather in the free-form visual structure of a dynamic abstraction. The painterly schema may also be thought of as the limiting case of a "reactive widget" system in which the number of objects approaches infinity, and their size approaches zero: in such a case, the granularity of the widget objects becomes so fine that they become a spreadable, audiovisual substance.

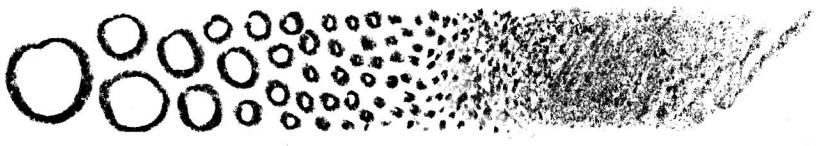


Figure 39. At the limit of granularity: objects become substance; sampling becomes synthesis.

From interactive animation systems, the painterly schema inherits the use of gesture capture, gestural augmentation, and gestural governance; it is distinguished from these prior systems, however, insofar as it uses these techniques to control sound as well as animation. Finally, the notion of an audiovisual substance is shared by Toshio Iwai's *Music Insects*. For Iwai, this substance is a static one, which only has temporal properties in the sound domain; for the goals and works which constitute this thesis, however, the audiovisual substance is dynamic in both the visual and audio domains.

The next chapter, *Design Experiments*, presents a series of interactive software artifacts which attempt to implement my painterly interface metaphor for audiovisual performance. The goals of this thesis, and the works which were fashioned to reflect these goals, evolved together organically. The next chapter therefore treats the details of this co-evolution in a narrative fashion: after an explication of a series of four older and silent experiments which introduce many aesthetic and technical issues, the chapter moves on to discuss five newer software systems which allow performers to create image and sound together.

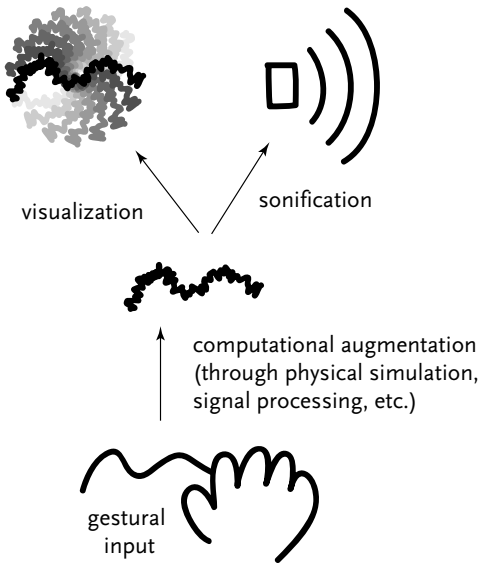



Figure 40. The structure of an "ideal" audiovisual performance instrument, according to the model proposed in this thesis. Gestural input is augmented by a computational simulation or processing algorithm, which is in turn sonified and visualized by further sub-systems.

# 3. Design Experiments

In support of this thesis research, I have developed a series of experimental software applications which explore the design space of computational instruments for visual and audiovisual performance. This chapter details these projects, beginning with a group of four pertinent silent experiments (*Streamer, Escargogolator, Polygona Nervosa*, and *Directrix*) I conducted prior to my Fall 1998 matriculation in Professor John Maeda's Aesthetics and Computation Group. After these I discuss a set of five new instruments developed over the past two years, which represents the present culmination of my research into the design of systems for the simultaneous performance of dynamic graphics and sound: *Yellowtail, Loom, Warbo, Aurora* and *Floo*. The present chapter devotes a short section to each of the nine experimental systems; each of these sections treats a given system's mechanism, its particular strengths or novel contributions, and its shortcomings and limitations.

### 3.1. Preliminary Silent Experiments, 1997-1998

This section describes the design of four silent environments—*Streamer, Escargogolator, Polygona Nervosa*, and *Directrix*—which I developed during 1997 and 1998 at Interval Research Corporation. The first three of these were created in a collaboration with Scott Snibbe, who was a colleague of mine at Interval at the time, and who had developed the *Motion Phone* animation environment (see Chapter Two) between 1991 and 1995. Although these four works predate my matriculation at MIT, I include them in this chapter for the reason that their design narratives, taken together, introduce many of the aesthetic goals, design guidelines, and technical issues which have structured the development of my newer sonified works. Apart from brief descriptions of *Streamer* and *Escargogolator* in [Snibbe and Levin 2000], none of these works have received any detailed treatment or explication in print.

*Streamer, Escargogolator, Polygona Nervosa* and *Directrix* are prototypes of "visual instruments"—quickly-executed experiments into the plausible analogy, in the visual domain, of musical instruments. Instead of allowing the creation of sound over time, these systems permit a performer to produce *dynamic imagery over*

*time*. As with conventional musical instruments, these systems were designed to offer their users the possibility of an invested, highly-present expressive engagement—a state which psychologist Mihaly Cziksentmihalyi has termed "creative flow"—with the medium of dynamic abstraction [Cziksentmihalyi 1996]. Each system attempts to do this by presenting an environment which:

- uses human gesture as a raw input of rich complexity;
- creates an animated, living environment, established by continuously changing synthetic graphics;
- has a quickly apprehensible interface that affords immediately satisfying results; yet at the same time, provides for a wide range of possible expression that one can continue to master over time; and
- can elicit joy, surprise and delight solely with abstract graphics [Snibbe and Levin 2000].

Several factors contributed to the genesis of these four silent works. One important influence was the set of advanced visual languages developed by Oskar Fischinger, Norman McLaren and other abstract animators in the early- and mid-20th century. The transposition of these dramatic and personal languages from the realm of composition (in film) to the realm of performance (in color organs) had been largely restricted by the limitations of the physical world. Computer graphics' apparent capacity to void the laws of physics, however, held the promise of making this translation possible.

A second factor was an aesthetic opportunity made possible by a technological development: when full-screen animation became a reality for desktop computers in the mid-1990s, most developers of interactive graphics overlooked the expressive potential of two-dimensional compositions, in favor of the "realism" promised by 3D, texture-mapped virtual realities. We sensed that the domain of two-dimensional imagery, so highly developed in the plastic arts of painting and cinema, had only begun to benefit from the affordances of real-time interaction and computation.

The most significant factor in the development of the four silent works, however, was undoubtedly the combined influence of Scott Snibbe's *Motion Phone*, Paul Haeberli's *DynaDraw* and John Maeda's *TimePaint*. Taken together, these systems all pointed toward the idea of an inexhaustible, exceptionally malleable, animated visual substance. This idea was a welcome contrast to

the considerable limitations of the sprite-based and ROM-based systems, such as *Director* and *mTropolis*, which dominated the field of consumer animation authoring environments during the 1990's. Having directly experienced these limitations during the development of my *Rouen Revisited* installation [Levin and Debevec, 1996] and several other Interval projects, I was eager to explore the woolly frontier of dynamic computational form to which *Motion Phone*, *DynaDraw* and *TimePaint* pointed.

At the same time that *Motion Phone*, *DynaDraw* and *TimePaint* indicated a broad and fertile territory of interactive graphics, these systems were also constrained by an interesting array of limitations. *Motion Phone*, for example, restricted all animations to looping compositions of circles, squares and triangles. Both *DynaDraw* and *Motion Phone* made heavy use of visually extraneous GUI elements, such as sliders, clearly suggesting that more research would be necessary before all aspects of an dynamic image could be performed with the same degree of direct physical control as a traditional static image. *TimePaint* was a provocative visualization of the nature of animated form, and embodied a suggestively different temporal model from either *Motion Phone* or *DynaDraw*, but it also had an extremely limited graphical repertoire. In the development of Streamer and the other silent works described in this section, these limitations were treated as opportunities to explore and invent an expanded design vocabulary for animation performance systems. Thus I set about to investigate how such systems might embody other temporal models, other graphical models, and other models and modes of interaction.

#### 3.1.1. Streamer

By December of 1996 I had spent many hours using and enjoying Scott Snibbe's *Motion Phone*, and had developed a strong desire to create a response to the many interesting ideas embodied within it. I identified two areas in which I felt there were immediate opportunities for such a response. I first noted that *Motion Phone* restricted its user to a very limited palette of Platonic forms—namely circles, squares, triangles and rectangles—and I therefore sought to develop a means by which a wider range of more malleable visual forms could be expressed. I also noted that *Motion Phone*'s animations were exclusively constructed around the exact reproduction of recorded human gesture. I sought

to explore, in reaction, the possibility of a hybrid realm of computational movement, halfway between *Motion Phone*'s literal playback of stored gestures, and the strictly procedural animation methods prevalent in the multimedia and graphics of the time.

My first exploration into these ideas was specifically prompted by an assignment given, appropriately enough, by Professor John Maeda to the students in his Fundamentals of Computational Form class at the MIT Media Laboratory. Although I was not yet enrolled at MIT at the time, I was closely following the Java applets posted online by Maeda's students as part of their classwork. One of Professor Maeda's assignments was to "design a kite guided by the cursor." After witnessing particularly successful solutions by Reed Kram, Matthew Grenby and others, I sought to try my hand at a solution of my own devising. I set to work in Macromedia *Director*, which was the only programming environment with which I was familiar at the time.

I chose to represent the long string of a kite as a graphical line which is emitted by the cursor, as in a traditional drawing program, but which is also progressively smoothed at every frame of animation, according to a naive simulation of kitestring physics. In the course of developing this simulation, I accidentally introduced a sign error, and the progressive smoothing I had intended became an exponential amplification instead. The positive feedback of this process results in a rapid exaggeration of the user's gesture, with the mark's trail overlapping itself and quickly flying in all directions away from the cursor. Even the tiniest wiggles in the user's gesture are magnified to the entire breadth of the screen within a few fractions of a second. The effect is an intoxicatingly responsive, managed chaos, similar to the experience of driving too fast. This first result was encouraging, not only for the unusual plasticity of its resulting forms, but also because it demonstrated that dynamic animation could be generated by a computational model in which human movement was the driving impulse.

In January 1997 I became frustrated by *Director*'s limited ability to generate synthetic graphics, and enjoined Scott to help build a more sophisticated version of the kitestring. The result was *Streamer*, which Scott implemented in C++ using Microsoft's DirectDraw graphics library, and in which the user's trail is represented by a Catmull-Rom spline that passes through the trail's successive pen points. In *Streamer*, a curved line emerges

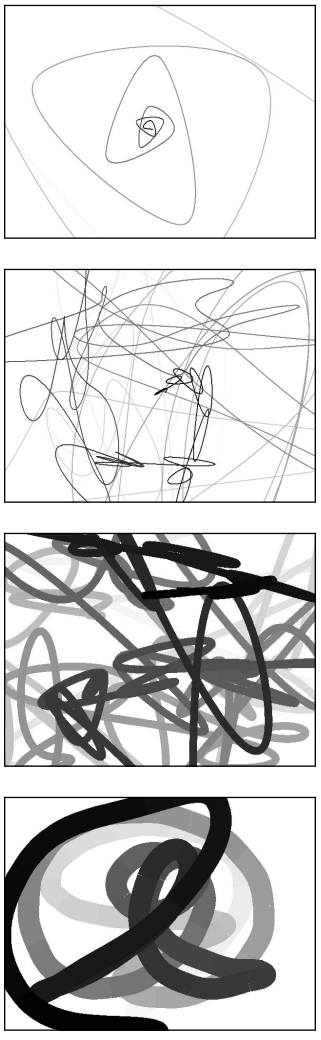


Figure 41. Stills captured from *Streamer*. The images have been inverted for better printing.

from the movement of the pen as long as the button is down. Although the behavior of this curved line is exaggerated in a manner that calls to mind the expanding smoke from a cigarette, or ripples in water, there is no computational randomness added to the system—the mechanical noise in the user's joints is sufficient to produce an interesting and organic result. As soon as the user releases the button, the curved line dissolves into blackness.

Figure 42. An image captured from *Streamer* in use. The image has been inverted for better reproduction in the printed medium; ordinarily, *Streamer* appears as a bright white line that recedes into a pitch black background.

Sometime later, Bill Verplank at Interval introduced us to Paul Klee's *Pedagogical Sketchbook*, in which Klee undertook a systematic treatment of the formal elements of visual abstraction [Klee 1923]. We found a natural affinity between our *Streamer* software and Klee's instructional drawings, which provocatively suggested how a line might be brought to life. Among Klee's first words in this book are, "A line is a walk for walk's sake." As Scott and I embarked on a set of collaborative explorations of the line's potential for dynamic expression, we took a cue from Klee and strove to design systems which emphasized their process of creation, over their products [Snibbe and Levin 2000]. *Escargogolator*, our next collaboration, extended this theme.

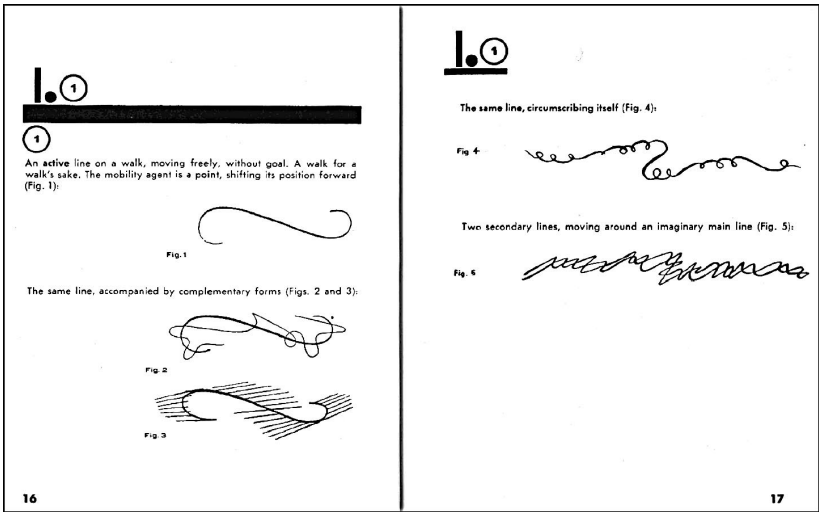



Figure 43. The first two pages from Chapter One of Paul Klee's *Pedagogical Sketchbook* (1923), in which he undertakes a systematic study of the formal elements of visual abstraction. Klee begins with a study of the Line [Klee 1923].

### 3.1.2. Escargogolator

In February 1997, Scott Snibbe and I collaborated on the design of *Escargogolator*, an abstract animation instrument in which a user's gestures are smoothly exaggerated or diminished according to their local curvatures.

As with *Streamer*, the dynamic animation in *Escargogolator* is generated by a computational model in which human movement acts as the driving impulse. Unlike *Streamer*, which only produces reactive graphics in response to the user's *real-time* input, *Escargogolator* obeys a different temporal and interaction model in which a progressive transformation is applied to a user's mark during and after its creation. Thus, while *Streamer*'s display evaporates almost instantaneously when its user ceases to feed energy into its system, *Escargogolator* allows a user to establish a configuration of gestural "initial conditions," and then observe the manner in which these conditions evolve or devolve over time.

The specific transformation applied to the user's marks in *Escargogolator* was inspired by the mathematical construction called the *evolute*, which I had read about in Eugene Shikin's splendid *Handbook and Atlas of Curves* [Shikin 1995]. In classical geometry, the evolute of a given curve is created by computing, for each position on the curve of interest, its center of curvature at that position. These centers of curvature, connected and taken together, form that particular curve's evolute. (Reciprocally, the original curve of interest is said to be the *evolvent* of its evolute curve.) The range of possible curves of evolution is essentially

infinite; circles, for example, have a single point for their evolute, while straight lines have straight evolutes located an infinite distance away. Evolute curves presented a natural opportunity for graphical exploration, since every line—even one defined by a user's idiosyncratic mark—has its own unique and oftentimes interesting evolute.

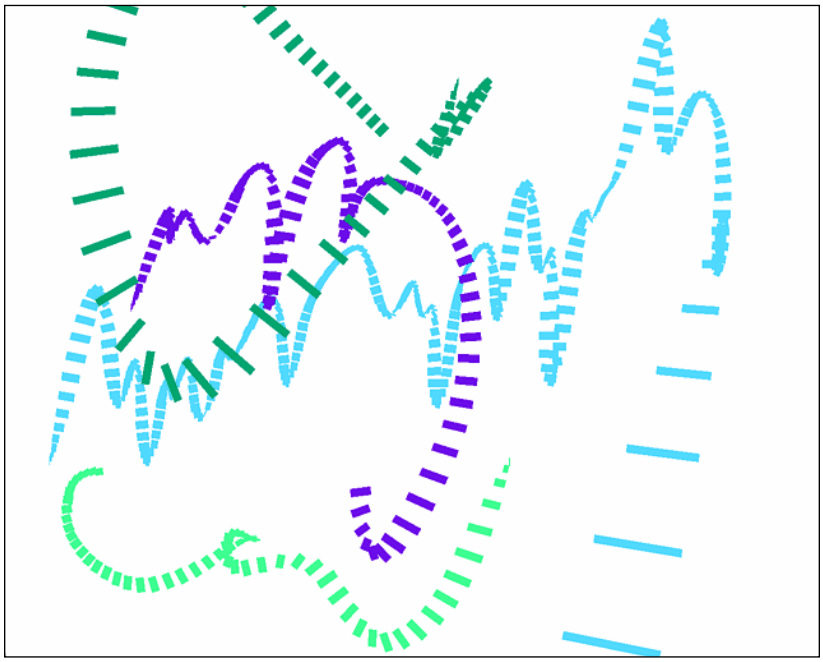


Figure 44. *Escargogolator* in use. The user has drawn four marks, which are gradually uncurling and unwinding to their initial points of origin. The image has been inverted for better reproduction in the print medium.

The development of *Escargogolator* was prompted by our desire to find out what would happen if a curve, created by the user in the form of a gestural mark, was compelled to move towards, and morph into, its own evolute. After we experimented with the mathematics, we settled on a slight simplification of this idea in which each gesture-sample moved in the direction of the dot product of its neighbors; effectively, each sample was compelled to move towards its local center of curvature, at every frame of animation. The speed of this motion was made proportional to the speed of the user's gesture at that location, resulting in interesting relationships between the line's curvature and its initial velocity profile. These relationships were further emphasized by representing the user's marks as ladderlike rungs whose width was proportional to the user's speed; though technically separate, these rungs become connected by the eye of the observer into coherent, organic forms.

The behavior which emerges from this system is peculiarly wormlike and has several interesting properties. Its most notable characteristic is that all of *Escargogolator*'s strokes gradually and inexorably unwind to their points of origin—that is, strokes eventually straighten themselves and ultimately shrink down to their first, single point (due, in part, to the system's boundary conditions). It is an interesting exercise, for example, to write in cursive and watch the letters evolve into a meaningless scribble [Snibbe and Levin 2000]. The precise manner in which the strokes uncurl themselves, though wholly deterministic, has the additional property that it is *handed*. Thus the dynamisms of the system are not only based on the amount of curvature, but also on the curvature's sign: clockwise portions of the curve gradually expand over time, while counterclockwise portions of the curve collapse. As a result, marks with certain configurations of curvature may grow substantially in size before ultimately shrinking down.

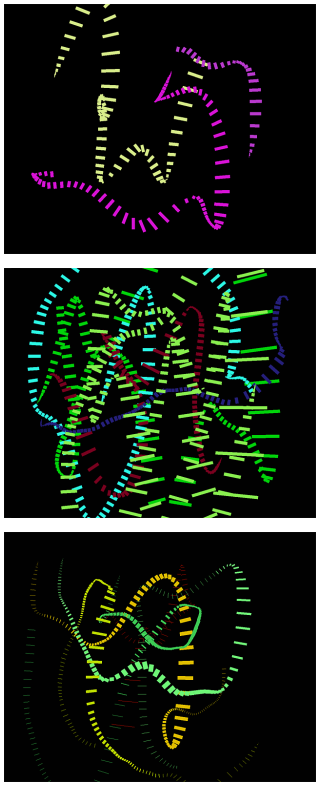


Figure 45. Additional images captured from *Escargogolator* in use.

*Escargogolator*'s chief contribution is the presentation of an alternative temporal model for interactive performance—one in which the user creates a set of initial conditions, and can then witness how those conditions evolve and disintegrate over time. Unlike Snibbe's *Motion Phone* and several of the experimental systems described in this thesis (including *Polygona Nervosa*, discussed next), the conditions established by the user in *Escargogolator* do not create a perpetually looping rhythm. If *Streamer* could be said to resemble a flute (insofar as there is only activity on the screen so long as the user infuses the system with energy), then *Escargogolator* is like a large gong, whose sound slowly fades away with an interesting timbral evolution.

### 3.1.3. Polygona Nervosa

In March 1997, I turned my attention from lines to shapes, and conceived of an interaction by which a user could simultaneously specify both the form and also the quality of movement of an animated polygon. I again developed a prototype of this interaction in *Director*, but the limitations of the Macromedia graphics environment prohibited the display of filled polygons or rounded forms. Shortly afterwards Scott ported the algorithm to his DirectDraw-based graphics environment, and these limitations were obviated. We named the new instrument *Polygona Nervosa* after its shapes' lively condition.

Figure 46. A still captured from *Polygona Nervosa*. In this example, the user has deposited ten polygons, which have been rendered by the system as filled Bezier-edged shapes. Forms create a new color in the regions where they overlap, based on the effects of an XOR ink mode applied to the shapes' colors.

In *Polygona Nervosa*, a user initiates a polygon by clicking a button on the mouse or other pointing device. With each additional click, the user adds another vertex to the polygon. When the user decides that enough vertices have been added to the shape, the user must "close" the polygon by clicking in a small hot region centered on the polygon's first vertex. At this point, the polygon begins to animate. Each vertex moves towards the subsequently-placed one, along the same motion path that the user traced out when originally depositing those vertices. The temporal duration of these motion paths are normalized, so that each vertex arrives at the destination of its path at the same time as the others. Thus the entire polygon returns to its original form—that is, the form specified by the original placements of its vertices—at periodic intervals. Many possible qualities of animation can be specified; for example, a shape can be directed to rotate in place, travel around the screen, grow and shrink, squash and stretch, extend prehensile pseudopods, or periodically intersect itself.

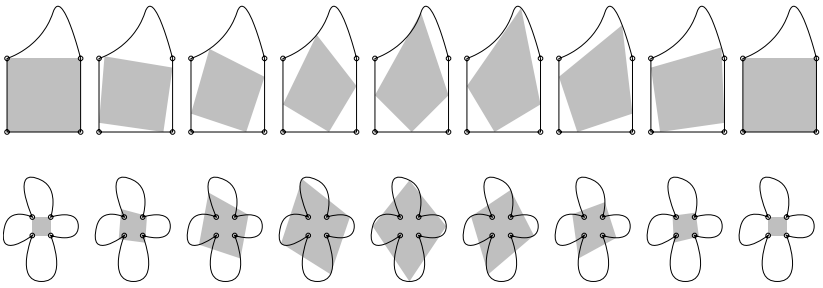


Figure 47. Two examples illustrate *Polygona Nervosa*'s underlying algorithm, by which a single gesture is used to specify both the shape and movement of a polygon. The thin black lines represent the user's gestural trace, which scaffolds the animated behavior of the final display, the light gray shapes. Small dots mark the location of the vertices created by the user's mouse-clicks. Note how each vertex in the gray quadrilateral moves along the user's gestural trace.

The user can create as many animating polygons as the computer's processing speed will allow. Because each of the polygons' motion paths have been normalized to the same period, the polygons themselves become synchronized into a looping composition (though the phase of each polygon may differ from the others).

The shapes in *Polygona Nervosa* may be either filled or hollow at the user's discretion; if they are hollow, the user can independently establish the thickness of each shape's boundary line using keys on the computer keyboard. Users can also choose whether the shapes are represented as straight-edged polygons, or have Bezier-curved edges whose control points are defined by the polygonal vertices. In this case, the system's shapes turn into smooth blobs that evoke 1950's textile patterns or the artwork of Joan Miró.

*Polygona Nervosa*'s most important contribution is the idea that a single interaction can be used to specify both the spatial and temporal aspects of an animated visual form. The system achieves this by leveraging the different affordances of the discrete and continuous aspects of a mouse-gesture: discrete mouse clicks are used to specify the shapes' spatial forms and positions, while continuous mouse movements are used to specify spatio-temporal dynamics. Both the shapes and also their animated movements have a wide expressive range, and can be imbued with a great deal of character or personality by a practiced user.

A known failure of *Polygona Nervosa* is the modality of its polygon-creation interaction. When the user initiates a new polygon, she enters into a "polygon creation mode" which cannot be exited until she closes the shape by returning to and then clicking on the shape's first vertex. The user cannot simply "let go" of the polygon to stop creating it, and as a result novice users of *Polygona Nervosa* often become confused by the unfinished polygons that are "stuck" to their cursor. This modality is wholly unnatural, especially when compared with the essentially amodal act of markmaking in traditional arts like drawing or painting: if a conventional painter wishes to stop painting a mark, she merely lifts the brush, since she is not subject to the constraints of some secret state machine.

Ironically enough, *Polygona Nervosa* uses the same modal interaction scheme for specifying closed polygon shapes as that

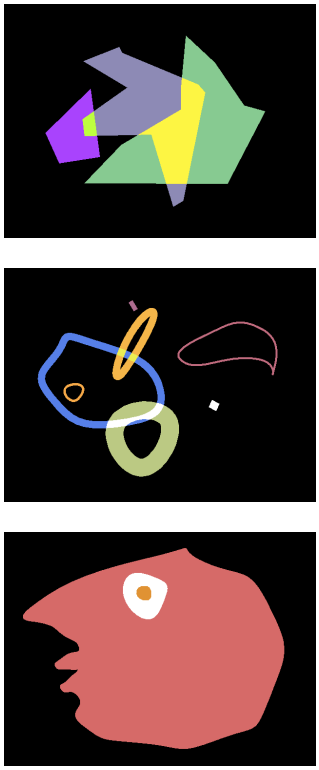


Figure 48. More stills captured from *Polygona Nervosa*. *Polygona Nervosa*'s generative scheme is flexible enough to permit the expression of animated representational forms.

of “conventional” drawing programs such as Adobe Illustrator or MacPaint. It is interesting to observe that this interaction method for polygon specification is apparently unable to survive the transposition from the static domain to the dynamic one. Even an instructive GUI “hint” (such as flashing a small circle around the first vertex, as a way of reminding the user to return there) seems unable to compete in the busy visual environment of the user’s animated composition. To outfit the system with a more elaborate mechanism of instruction or graphical guidance would miss the point: any adequate repair to *Polygona Nervosa* will require the design of a non-modal interaction for simultaneously specifying shape and animation. This is an interesting topic for further research.

### 3.1.4. Directrix

In the summer and early fall of 1998, I returned to the study of lines and developed *Directrix*, an environment in which users can quickly generate animated “pseudo-parabolas.” These complex curves are the result of an interplay between a set of dynamic and static gestures performed by the user. When several of these curves are layered together, the results can vary from sparse and delicate constructions of gently curved lines, to violently twitching, thatchy masses.

Figure 49. An animated drawing made in the *Directrix* environment.

*Directrix* creates images from a generalized model of parabolas. In classical geometry, a parabola is defined as the locus of points which are equidistant from a special point called the *focus*, and a straight line called the *directrix*. The *Directrix* environment was designed to explore the implications of two premises: firstly, that the shape of a parabola's directrix could be the personal, idiosyncratic mark of an interactant, and secondly, that its focus could be a moving point animated along the trace of a user's recorded gesture.

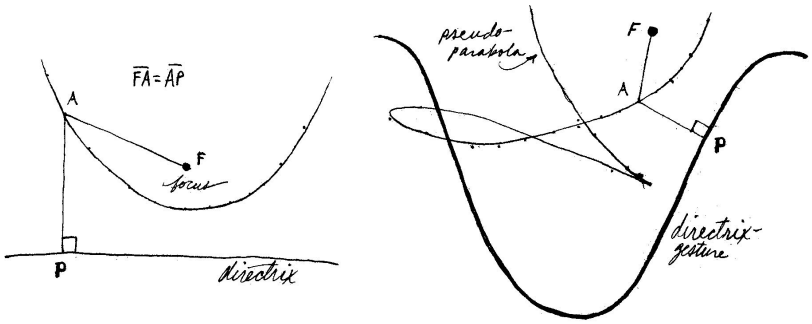



Figure 50. An explication of the parabolic constructions used in the *Directrix* environment. In the left-hand sketch, a classical parabola is generated from a focus ***F*** and an essentially straight directrix below it. Each point ***A*** along this parabola has the property that line segment ***FA*** is exactly equal in length to the line segment ***AP***, which passes through ***A*** and is perpendicular to the directrix. In the right-hand sketch, a pseudo-parabola is formed in an identical manner, between a focus ***F*** and a valley-shaped directrix drawn by the user.

A session with *Directrix* begins with a screen that is entirely black, save for a single bright point at the center, representing the current location of the focus. Users begin by drawing a linear mark, as with a traditional drawing program or paint program; this line is treated as the directrix of a (generalized) parabola. As the user inscribes their mark on the canvas of the screen, a colorful pseudo-parabola grows between it and the focus. While straight directrices generate predictably parabolic results, differently curved scribbles can produce results ranging from circles, to straight lines, to bizarrely abstract squiggles. Although the behavior of the system's generative algorithm can be peculiarly counterintuitive, *Directrix* is ultimately deterministic, resulting in an environment which is both richly variable yet quite learnable.

After the user has deposited a directrix, and thereby generated a pseudo-parabola, the user can pick up the focus (by using a different button on the mouse/pointing device) and trace out a path along which it will animate. When the focus point is released, it continues to animate along this path, looping and restarting as necessary. While the focus animates, the shape of the pseudo-parabola is recomputed at every frame, producing a curve whose shape changes dramatically and periodically. Even as the focus animates along its path, the directrix can be cleared and

replaced with an entirely different line, yielding a new parabolic curve whose form is different, but whose character of movement is the same. More than one focus can be deposited on the canvas, producing a family of related pseudo-parabolas which all share the same directrix. Because each of these foci obey independently-specified animation loops, the parabolae they generate exhibit subtly-shifting phase relationships, creating an essentially aperiodically-textured animated display.

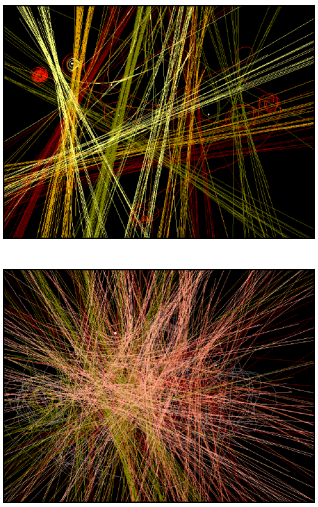


Figure 51. More stills captured from interactions with *Directrix*.

*Directrix* is interesting because of the interplay it establishes between a strictly spatial specification (the directrix) and a spatio-temporal one (the path of the focus). This interplay produces from the user's gestures an augmentation which is not only animated, but also can possess a spatial complexity that would be tedious or impossible to produce manually. *Directrix* has, nevertheless, two important shortcomings. The first, which is more of a limitation than a shortcoming, is that the temporal structure of its results generally has an extremely limited dynamic range. This is a natural outcome of the system's animation model, in which looping behaviors of different lengths phase against one another, since such systems inevitably produce a noiselike visual texture whose apparent logic, taken over time, seems non-deliberate. As a result, possible compositions in *Directrix* belong more closely to the space of animating textures, like the surface of a pond, than to the space of deliberately constructed narratives that often characterize animated cartoons.

The second shortcoming of *Directrix* is that, as with *Polygona Nervosa*, it is difficult to apprehend for the first time without instructions. In *Directrix*, this is owed to the fact that the act of drawing has been functionally overloaded: depending on the context, dragging the cursor can be used to specify the form of the parabola's directrix, or the gait and path of its focus. These two specifications, moreover, are functionally interdependent, such that it is impossible to produce animated results without authoring both specifications. Because of this functional overloading and interdependence, *Directrix* imposes a cognitive load greater than any of the other experiments described in this thesis. Of course, it is possible that a different physical interface to *Directrix*, such as a pair of functionally-distinguished pointing devices, might relieve this load substantially.

### 3.2. Experiments in the MIT ACG, 1998-2000

The remainder of this chapter is devoted to the description of five systems which permit the simultaneous performance of both abstract animation and synthetic sound: *Yellowtail, Loom, Warbo, Aurora* and *Floo*. These systems were all developed between September 1998 and May 2000 in the Aesthetics and Computation Group at the MIT Media Laboratory, under the direction of Professor John Maeda. The genesis of this work is described below.

By the summer of 1999 I had developed close to twenty systems which interpreted or augmented, in one way or another, the dynamism of two-dimensional gestures in abstract animation spaces. In addition to the four environments described above, I also developed a host of others: *Disctopia, Blebs, Molassograph, Blobby, Splat, Stripe, Ribble, Telephone, Dakadaka, Scratch, Meshy, Curly, Brillo,* and *Floccus*. These systems explored a variety of aesthetic avenues, by connecting abstract animation to (for example) photographic source materials (*Molassograph, Blobby, Brillo*), written calligraphy (*Telephone*), and typography (*Dakadaka*, developed in collaboration with ACG colleague Casey Reas, and *Ribble*). These systems also explored the aesthetic affordances of a variety of technological means, including lattice-gas cellular automata (*Ribble*), implicit curves (*Splat, Blobby*), cubic surfaces (*Meshy*), raster image convolution techniques (*Ribble, Scratch*), and finite-element physical simulations (*Blebs, Brillo, Floccus*).

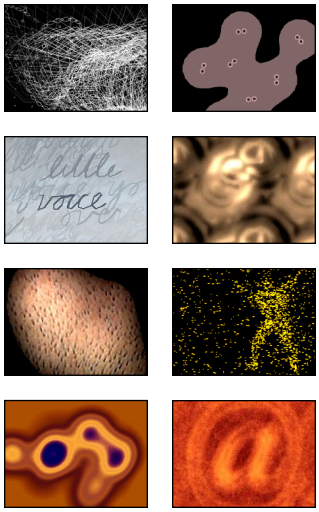



Figure 52. Stills from a variety of other works executed over the last two years: *Meshy, Splat; Telephone, Stripe; Blobby, Molassograph; Dakadaka, Ribble.*

In the course of developing these experimental systems, I also evolved a methodology which enabled such applications to be rapidly prototyped: after doodling and describing their behavior in my journal, I would render them as "interactive sketches" in the form of Java applets. Of course, such applets were subject to considerable technological and aesthetic limitations: the slow speed of the Java graphics toolkit prevented the use of the full screen resolution; the interpreted nature of the Java language imposed a serious upper bound on raw computation; and the omnipresent GUI frame of the applet's Web browser was less than desirable. Nevertheless, the development of small applets had the advantage that it allowed me to quickly evaluate whether a given sketch was worthy of further development in the more powerful but oftentimes more tedious C++ environment. Taken together, these applets also formed a cross-platform, interactive work journal which was both self-documenting and shareable.

Nearly all of these systems can be browsed online at *http://www.media.mit.edu/~golan/index.html*.

In the summer of 1999, motivated by my dual interests in both painting and music, I decided it was time to move beyond silent systems and tackle a personal "holy grail"—the design of an environment which would afford the simultaneous creation and performance of dynamic image and sound. There already existed, of course, numerous examples of computational audiovisual environments, which permitted their users various degrees of control over image and sound. Nearly all of these systems adhere to a set of basic interaction metaphors—*timelines*, *control panels*, and *reactive widgets*, discussed in Chapter Two—which place the control of image in a subsidiary role to that of sound, and substantially curtail the expressivity of their visual dimension. Two important exceptions to this, UI Software's *Metasynth* (discussed in this chapter) and Toshio Iwai's *Music Insects*, can only produce *static* imagery, and not animated imagery, among other limitations. I set myself the problem of developing a dynamic audiovisual performance system in which both the visual and sonic dimensions could be deeply, and *commensurately*, plastic.

*Yellowtail, Loom, Warbo, Aurora* and *Floo*, discussed in the next five sections, are my results. They succeed to greater or lesser degrees. Unlike most audiovisual environments on the computer—which appear to have begun life as musical systems in search of an adequate visual interface—these five systems had their origins instead in expressive gestural animation environments for whose gestural inputs I subsequently sought suitable sonifications. In doing so, I extended the strategy which had guided the development of the strictly visual systems discussed in the previous section: in the work that followed, I now strove to develop an inexhaustible, highly malleable, *audiovisual* substance.

### 3.2.1. Yellowtail: Animated, Real-Time "Pattern Playback"

#### 3.2.1.1. Origins

*Yellowtail* was my first experiment into the design of an environment for the simultaneous creation of both sound and image. It evolved out of an earlier silent piece, called *Curly*, which I developed in September of 1998. Although *Yellowtail* eventually fell short of achieving a wholly painterly

interface for real-time audiovisual performance—its final design is fundamentally scorelike—it was nevertheless an important milestone in the evolution of this thesis work. As we shall see, *Yellowtail* served as the model against which the goals of this thesis developed in contradistinction.

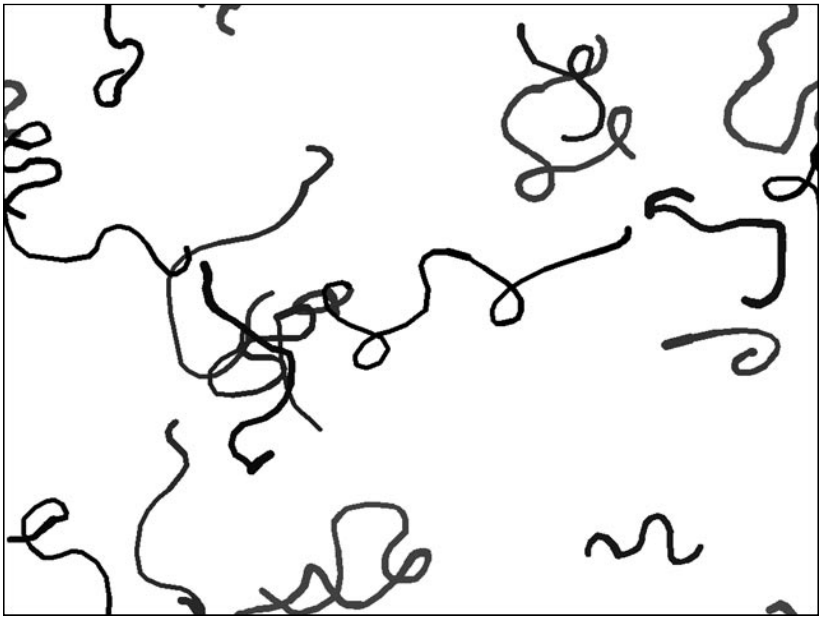


Figure 53. A screenshot from *Curly*, developed in September 1998. The image has been inverted for better reproduction in the print medium.

*Curly*, *Yellowtail*'s progenitor, was a reactive paint system in which a user's linear marks transform into an animated display of lively, worm-like lines. After the user deposited a mark, the system would then procedurally displace that mark end-over-end, making possible the simultaneous specification of both a line's shape as well as its quality of movement. Straight marks would move along the direction of their own principal axes, while circular marks would chase their own tails. Marks with more irregular shapes would move in similarly irregular, but nonetheless rhythmic patterns. *Curly*'s screen space obeyed periodic (toroidal-topology) boundary conditions, such that marks which crossed the edge of the screen would reëmerge on the screen's opposite side, rather than disappearing altogether. Two different styles of motion could be selected by the user using different buttons on the pointing device: the CURLY_TRAVELLING style, in which the marks would travel across the screen, and the CURLY_STATIONARY style, in which the marks would animate in place.

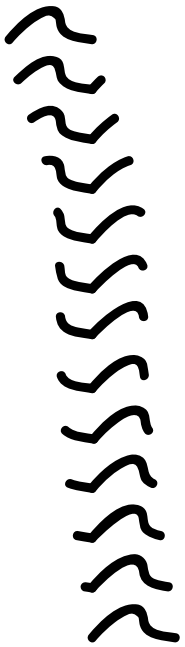


Figure 54. The evolution of a CURLY_TRAVELLING gesture as it progresses across the screen.

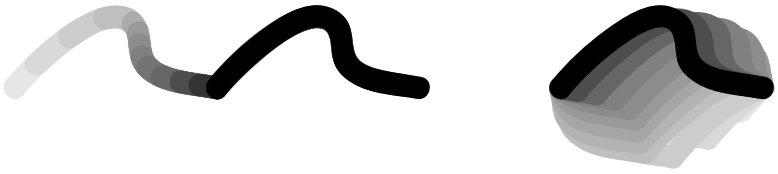


Figure 55. The marks in *Curly* can obey one of two different styles of animation. On the left is the CURLY_TRAVELLING style, in which a mark propagates along an axis of movement defined by its endpoints. On the right is the CURLY_STATIONARY style, in which a mark animates in place by cycling its shape through the stationary positions initially established by its original endpoints.

No randomness was employed in the procedural animation of the "curlies." Instead, their animated behavior is strictly determined by the shape and speed of the mark when it was drawn. Nevertheless, because each line repeats according to its own natural period, the complex phase relationships of the different marks produce the effect of an ever-changing yet coherent animated texture.

#### 3.2.1.2. Sonification

In June of 1999, I had the idea of sonifying *Curly* by treating its animating canvas as an "inverse spectrogram." Ordinarily, a spectrogram is a diagrammatic image used to visualize the frequency content of sound data. In a typical spectrogram, Short-Time Fourier Transforms (STFT) are applied to extremely small portions of a waveform, and represent the time-based information of the wave segment as components in the frequency domain. Transforms from adjacent windows of sound data are then rendered as a picture to create an image of the sound's frequency content versus time.

Spectrograms were originally developed to analyze sounds, such as speech, but took on provocative new possibilities when used in reverse, as a means of synthesizing sound. This technique, called *pattern playback*, was first developed by the speech researcher Frank Cooper in the early 1950's [Cooper 1953]. Cooper showed that it was possible to draw a pattern of paint splotches on plastic, and then use a machine of his own design to play back the sound. This made it possible for his lab to do many psychoacoustic experiments, and it also helped validate the use of a spectrogram as an analysis tool [Slaney 1995]. Cooper's machine used an array of light sources, each modulated at one of the fifty harmonics of 120Hz, to illuminate a strip of acetate tape. Patterns were painted on the film, and the light that was reflected from the pattern was transformed by photoresistors into a varying voltage and then amplified for auditory playback. The result, according to Cooper, was "highly intelligible" speech [Slaney 1995].

Since then, a number of researchers and companies have developed spectrogram-based drawing systems for the analysis and resynthesis of sound. In these systems, a digital image representing the intensity of different audio frequencies over time is used as a "score" for an additive or inverse-FFT synthesizer (a sound synthesizer in which a large number of weighted sinusoids are summed to produce complex tones). Examples of such systems include John Strawn's *eMerge* (1985), Gerhard Eckel's *SpecDraw* (1990), B. Holloway's *LemurEdit* (1993), and Malcolm Slaney's *Pattern Playback Plugins* (1995), the last of which embedded sound spectrogram technologies in an Adobe Photoshop plugin [Roads 1993, Slaney 1995]. Perhaps the most popular spectrogram resynthesizer, however, is UI Software's *Metasynth* for the Macintosh [UI Software 1998], which merges an additive sound synthesis engine with a variety of spectrogram-specific image editing tools and filters.

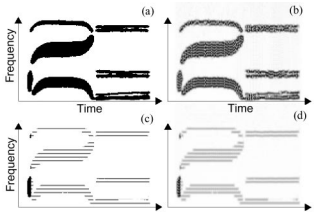



Figure 56. "Pattern Playback" of hand-painted spectrograms made by Frank Cooper in the early 1950's: (a) the original spectrogram pattern, (b) spectrogram of the inverted signal, (c) original spectrogram with pitch harmonics, (d) spectrogram of inverted signal with pitch harmonics [Slaney 1995].

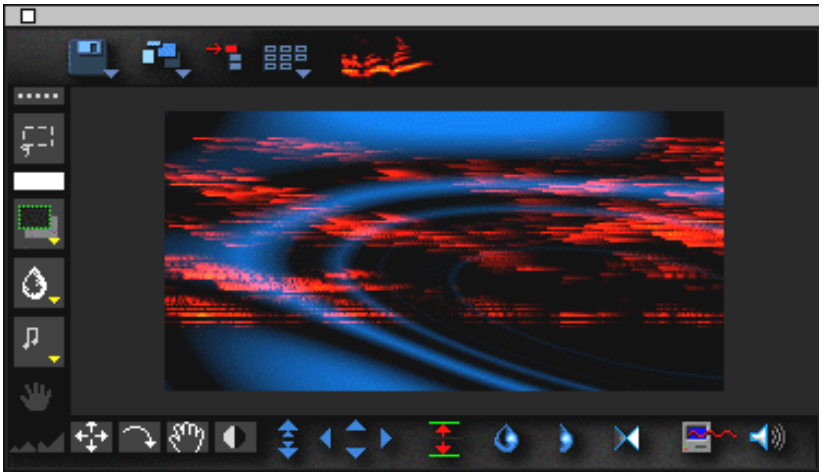


Figure 57. An interface from UI Software's *Metasynth*, a spectrogram-based synthesizer [UI Software 1998].

As powerful as such systems are, I felt that they could be improved or extended in two important ways. Firstly, none of the pattern playback systems were designed with the capacity to support real-time performance. In all cases, including *Metasynth*, the metaphor of interaction has been modeled after that of a traditional music sequencer: users paint into the spectrogram, click on the tapedeck-style "play" button, evaluate the sonic results, stop the playback, and then paint some more. This slow feedback loop of painting and audition is suitable for a meticulous style of composition, but makes improvisatory performance difficult or impossible. In sonifying *Curly* with pattern playback technology, I sought to collapse the duration of this feedback loop in order to produce an effective simultaneity of creation

and evaluation. To this end, I borrowed a technique discussed in Chapter Two, in which a looping computer score has the capacity to be modified at the same time that it plays back its contents.

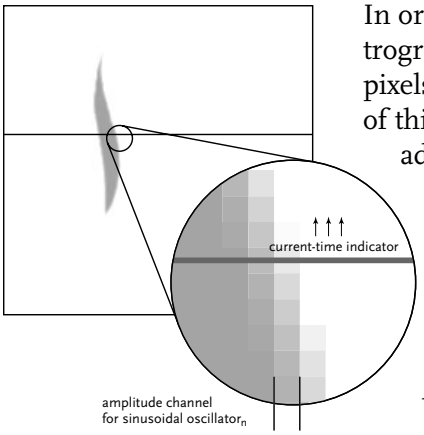



Figure 58. The spectrogram interface patch in *Yellowtail*. A horizontal line called the current time indicator sweeps the patch periodically from bottom to top. At any given moment this indicator may or may not intersect a row of pixels which belong to one of the user's animating marks. Each of the columns of pixels directs the amplitude of a given sinusoidal oscillator in an additive (Fourier) synthesizer. The greater a pixel's intensity, the more of its corresponding oscillator is heard in the final sound. The oscillators are arranged in order of exponentially increasing pitch from left to right, such that the spectrogram's width spans about six octaves.

In order to support real-time sound performance, a square spectrogram patch was added to *Curly* in the center of its canvas. The pixels of the screen's frame buffer coinciding with the location of this patch are fetched at frequent and regular intervals by an additive synthesizer; sound is then generated by mapping the brightnesses of pixel columns in the patch's frame buffer to the individual amplitudes of a bank of additive synthesis oscillators. As a result, any of the drawn marks which happen to intersect or occupy this patch immediately result in auditory events. With the addition of pattern playback sound generation and a minor visual redesign, this new version of *Curly* was renamed *Yellowtail*.

The second major extension I wished to make to pattern-playback systems was the idea of using an *animated* image instead of a static one. Even a system which permits real-time score manipulation and playback can yield tiresome results if the score's inherently static nature produces unchanging sounds when looped. An animated spectrogram image, by contrast, held the potential to create manageable variability in both sound and image. The dynamic nature of the *Curly* animation algorithm provided a ready solution. If the canvas was filled with CURLY_TRAVELLING marks, then the marks would intersect the spectrogram patch at seemingly stochastic intervals, forming a texture of controllably irregular tones and chirps. If, on the other hand, a CURLY_STATIONARY mark were placed into the spectrogram patch, the result would be a periodic event which sounded different every time it was played, yet whose variability was governed by precise bounds set by the user.

In addition to these two key innovations in animated pattern playback, three small design features of *Yellowtail* are also worthy of mention: its performance grids, its adjustable sound context, and its use of image processing techniques. The first of these, performance grids, refers to a means by which the user's gestures could optionally "snap" to specific quantization grids in the horizontal (pitch) or vertical (temporal) axes. The benefit of this feature is that users can choose to conform *Yellowtail*'s otherwise continuous sound-space into the more discretized sound-space generally characteristic of music. Marks which are conformed to

the vertical quantization grid, for example, only make sound at regular divisions of common meter, producing rhythmic noises in a manner similar to a drum machine. Marks which are conformed to the horizontal grid, on the other hand, are restricted to the nearest pitch in an equal-tempered chromatic scale.

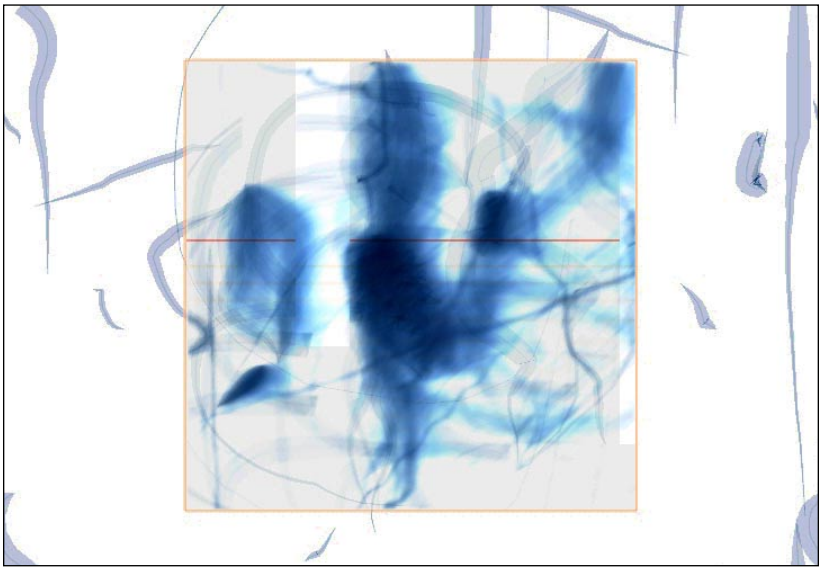


Figure 59. A screenshot from *Yellowtail*, showing its square spectrogram patch in the center, with its horizontal current time indicator. The user's marks have been blurred by *Yellowtail*'s real-time convolution filter, described below. The image has been inverted for better reproduction in the print medium.

A second interesting feature of *Yellowtail* is its adjustable sound context, in which its spectrogram patch can be picked up by the user and moved around. Originally, it was seen as a shortcoming that the spectrogram patch, owing to limitations of the computer's speed, could not occupy the entire screen space. Interestingly, however, this technological constraint eventually provided a valuable design opportunity for enhancing the system's expressivity. By grabbing the patch itself and dragging it around, the user can treat it as a mobile "sound lens" and thereby to "listen" to different regions of the visual composition. Smaller movements of the patch, such as small left-to-right adjustments, make possible the musical transposition of the marks contained within it, while large translations permit dramatic and instantaneous shifts in context.

A third special feature of *Yellowtail* is the option it provides of applying a real-time 2D convolution operation to the pixels in the spectrogram patch. Presently, only one convolution kernel is provided, namely a low-pass filter. The effects of this image processing technique are a substantial blurring of the image, combined with a frame-to-frame temporal persistence similar

to video feedback or retinal afterimages. The convolution filter produces soft and attractive visual results, but is also especially noteworthy for the corresponding changes it precipitates in the audio synthesized from the spectrogram patch. When the blurring convolution is enabled, the audio acquires an otherworldly, cavernous, deeply reverberant quality.

#### 3.2.1.3. Discussion

The two most important contributions of *Yellowtail* are that it (1) permits the real-time creation and performance of spectrographic image patterns, and furthermore that it (2) permits the use of a dynamically animated image, and not just a static image, as the raw material for pattern playback. The combination of these two ideas yields an audiovisual instrument which not only affords an unusual quality and high degree of control over the spectral content of sound, but also makes it possible for this spectral information to gradually (or abruptly) evolve over time in a manner programmed by the user's gestural movements.

It was during the course of developing and critiquing *Yellowtail* that the primary objective of this thesis—the design of a painterly interface metaphor for audiovisual performance—was crystallized for the first time. Several shortcomings of *Yellowtail*, in particular, led to the articulation of this goal. I was first struck by the manner in which the painterly visual space of *Curly* had become conceptually overridden by the addition of *Yellowtail*'s diagrammatic spectrogram patch. I quickly realized that, however its means might differ, *Yellowtail*'s basic metaphor for creating sound was no more novel than that of a traditional score or sequencer. Moreover, its spectrogram's arbitrary mapping between dimensions of sound and image, namely, {X=pitch, Y=time}, had the effect of introducing arbitrary non-isomorphisms into the pictorial plane. Thus the right half of the screen became the privileged location of high-pitched sounds, while visual artifacts situated in the left half of the screen became inextricably bound to low pitches. Such a deliberate and arbitrary non-isomorphism may be a standard device in the visual language of diagrammatic information visualizations, but was, I felt, poorly suited to the compositional language of abstract cinema which had motivated the work since the beginning, and which I wished to preserve.

Another important shortcoming of *Yellowtail*, from the perspective of a painterly audiovisual creation space, was that its spectrogram interface became an extraneous visual and syntactic intervention in the landscape of an otherwise vibrantly calligraphed surface. To elaborate, I consider the patch to be an extraneous visual element in the image plane, because it is not itself generated by *Yellowtail*'s user, but instead exists as an *a priori* feature of the software environment. It is, simply, an unrequested component of the visual space, whose continual presence is irrelevant to the user's visual composition, yet irrevocably a part of it; it is as if, in some hypothetical world, every fresh sheet of drawing paper arrived pre-marked with an indelible square in its center. The spectrogram patch is also a syntactic intervention because it functionally segregates the surface of the screen into "pixels which make sound" (the marks inside the patch), "pixels which don't make sound" (marks outside the patch), and, disturbingly, "pixels which signify the presence of an agent which operates on others to produce sound" (the pixels which represent the patch itself).

*Yellowtail* succeeds in producing an environment in which there is an "unlimited amount of audiovisual substance," but this substance only obeys the strict and conventional laws of the language of diagrams. In the work that followed, I sought to design audiovisual performance systems whose surfaces were situated within the expanded and free-form visual syntax of abstract painting and cinema.

### 3.2.2. Loom: Wrapping Timelines Around Gestural Spines

*Loom* was designed as a direct reaction to *Yellowtail*. In this environment, I sought to free the user's painted elements from the clutter of any extraneous visual interfaces, and also from the rigid visual language imposed by *Yellowtail*'s diagrammatic mapping between image and sound. It was my hope, by so doing, that I could create an application in which every visual element was associated with a corresponding sound-event, and vice versa. It was also my hope that by eliminating GUI components and other diagrammatic elements, I could return to a more painterly or cinematic screen space.

A user begins interacting with *Loom* by drawing a mark with the pen, mouse or other pointing device. As the user draws the mark, a musical tone is generated, whose sonic properties (such as timbre and volume) are continuously governed by the shape of

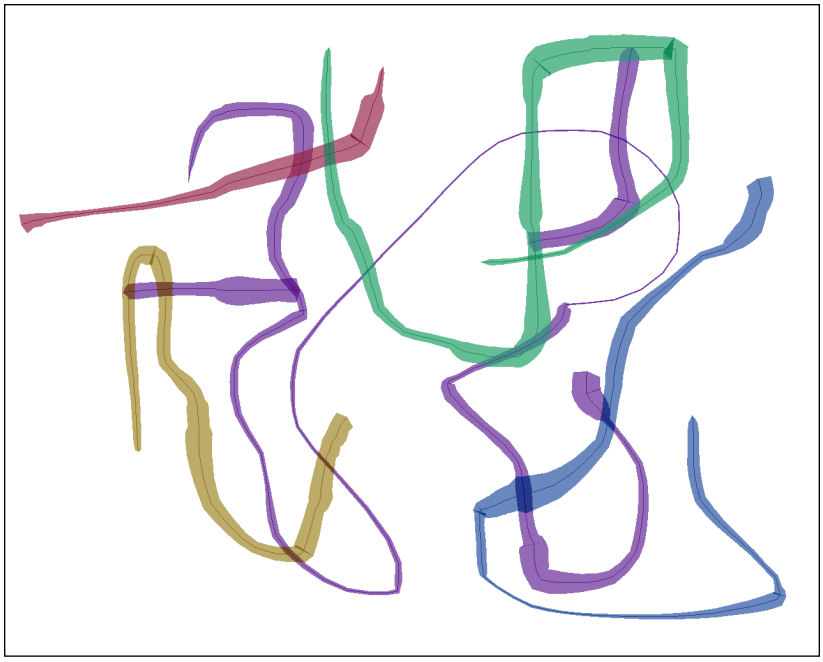


Figure 60. A screenshot of *Loom* in use. The image has been inverted for better printed reproduction.

Figure 61. The animation of a mark element in *Loom*. The mark appears to be drawn and re-drawn by an invisible hand.

the mark at its endpoint. If the user presses harder with her pen, for example, the mark is visually thickened in that location and a louder note is produced at that point in time. If the user makes an abrupt change in the mark's curvature, such as an angular bend, then the timbre of the musical tone becomes momentarily brighter. The details of this sonification will be discussed shortly.

At the same time that the user draws the mark, the temporal dynamics of the user's movements are also recorded. These dynamics are then used to play back the gesture. When the user finishes creating the mark, the mark commences a periodic behavior in which it first disappears, and then re-grows from its point of origin according to the precise dynamics with which it was created. The effect is that the line appears to be drawn and re-drawn, over and over again, by an invisible hand. As the line is redrawn, its same musical tone is heard, modulated over time in the same manner as when it was first created.

The user can place an unlimited number of such marks on the screen, each of which produces a unique musical tone. Each mark's playback synchronizes to a common clock whose period is established by the user according to one of two methods: in one method, all of the marks recur at time-intervals whose lengths are quantized to an independent (and adjustable) metronome; in the second method, all of the marks recur at time-intervals

whose lengths are integral multiples of the first mark that the user deposited. Even though it is possible (in either method) for all of the visible marks to have periods of the same duration, their phases may differ considerably according to how they were temporally placed by the user. As a result it is possible to perform marks which yield interesting rhythmic textures, interlocking polyrhythms, and call-and-response patterns.

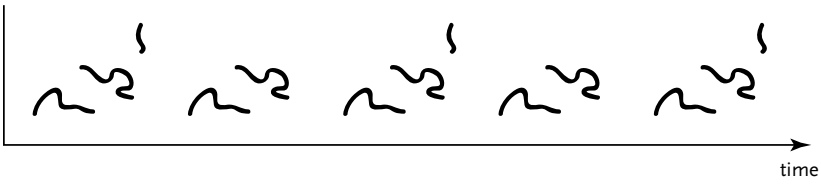



Figure 62. The user's gestures recur periodically in *Loom*. All of the marks have periods which are an integral multiple of some base length. In the example shown here, the lower two marks have the same period, while the uppermost mark has a period exactly twice as long (that is, it recurs exactly half as often). Although the marks recur in lock-step with each other, an important feature of the *Loom* environment is that each mark can have its own, independent phase in the common rhythmic cycle.

*Loom*'s sonification is based on the idea that a score or timeline can be "wrapped around" a user's mark. Unlike traditional timeline diagrams, which have straight abscissas, *Loom* treats a user's gesture as a curved spine around which the "time" axis of a score is wrapped. The data contained in a *Loom* timeline is the database of time-stamped information associated with a gestural mark, including its position, velocity, pen pressure, and local curvature, taken over its length (and duration). When a mark is "re-drawn" in an animated fashion, the *Loom* software uses these streams of information to drive the continuous control parameters of a Frequency Modulation (FM) synthesizer.

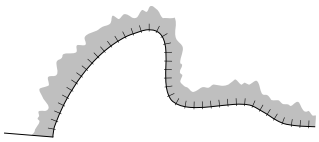


Figure 63. A fictitious example of a timeline wrapped around the spine of a gesture. In this case, a diagram representing a positive, noisy scalar value over time has been wrapped around a mark. In *Loom*, there are several such streams of continuous information defined over the length of each mark; these are used to drive the parameters of the audio synthesizer.

*Frequency Modulation* is an extremely versatile and efficient method of generating musically useful complex waveforms [Moore, 1990]. The musical possibilities of FM were first explored by John Chowning at Stanford University in the early 1970's; since then, the technique has since become very well known, due to its adoption in 1983 by the Yamaha corporation in their popular DX7 family of synthesizers. In Chowning's basic FM technique, a carrier oscillator is modulated in frequency by a modulator oscillator, according to the equation:

$$y = A \sin (Cx + I \sin Mx)$$

where ***A*** is the peak amplitude of the resulting waveform, ***C*** is the carrier frequency (in radians/sec), ***M*** is the modulator frequency (in radians/sec), and ***I*** is the index of modulation. The specific ratio of ***C:M*** defines the set of possible side-bands (harmonics and other spectral partials) that are generated by the equation. The index of modulation ***I***, on the other hand, controls the *bandwidth* of the resulting waveform; in other words, it controls the depth

of the modulation, or the amount of these sidebands added to the original waveform. Figure 64 shows the effects of increasing the index of modulation in an FM wave when the ***C:M*** ratio is held constant. Generally speaking, a higher index of modulation yields more complex waveforms, which result in brighter, harsher sounds.

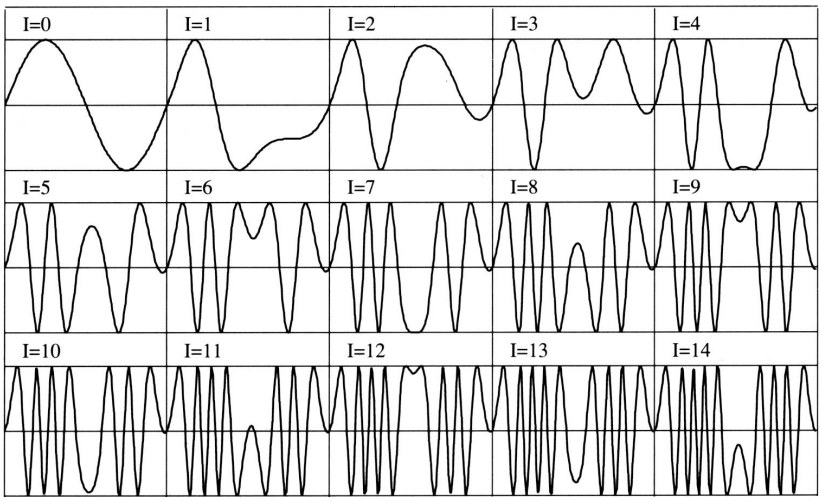



Figure 64. This diagram demonstrates the effects of increased FM modulation on a sine wave. Each cell represents the appearance of the waveform at integral increments of I, the index of modulation, in the standard FM equation

***y = A sin(Cx + IsinMx)***.

In this set of examples, ***A***, ***C*** and ***M*** are 1, and ***I*** ranges from zero to 14. Generally speaking, a higher index of modulation yields more complex waveforms, which result in brighter, harsher sounds [Adapted from Moore, 1990].

In *Loom*, several properties of a gestural mark are used to control the parameters of the FM synthesis equation. The local velocity of the mark, determined by taking the Euclidian distance between the user's mouse samples, controls the overall amplitude ***A*** of the FM voice associated with that mark, as well as the mark's visual width. The user's pressure (from a Wacom tablet or similar pen-based device, if such is available) also contributes to the control of the amplitude, in addition to affording control over a small amount of vibrato. The most interesting and noticeable mapping, however, is that established between the local curvature of the mark and the index of modulation ***I***. According to this scheme, a section of the mark which is straight will produce a smooth tone, similar to a sine wave, because its curvature is zero. A section of the mark which is curved will generate a brighter sound, proportional to the tightness of its curve. When the mark traces out an abrupt angle, however—that is, a tiny location where the curvature is "infinite"—it produces a momentary burst of brash harmonics. The result is that a user can create percussive or rhythmic modulations in the tone's timbre by drawing angles or knuckles in the gestural mark.

| Gesture Measure | Synthesizer Parameter |
|---|---|
| Local velocity | Amplitude |
| Local pressure | Depth of vibrato, and amplitude |
| Local curvature | FM index of modulation |

Figure 65. A table of the mappings between image properties and sound parameters used in the *Loom* synthesizer.

*Loom* is the first example I can present of a painterly visual interface for audiovisual performance. In it, the user can create an inexhaustible audiovisual substance, in the form of periodically animating, sonified marks. Although *Loom* flirts with the visual language of scorelike diagrams, it ultimately negates it by wrapping its timelines around these marks, thereby maintaining an abstract, painterly image space.

Unfortunately, *Loom* is not a complete success: its greatest failing is that certain important parameters of the FM synthesis equation—notably the carrier and modulator frequencies—are not assigned by properties of the gestural mark. Instead, these parameters are selected by the interactant before drawing the mark, using a momentary slider interface called forth by a secondary mouse button. Thus, although *Loom*'s eventual output resides in a painterly visual space, aspects of the *Loom* authoring experience are still dependent on axial (i.e. control-panel like) GUI elements. To remove all GUI elements and instead place these aspects under the gestural control of the user is an area for further research, and may best be solved by (for example) the use of alternative controllers and two-handed interfaces.

### 3.2.3. Warbo

*Warbo* was a quick experiment in which I combined a number of old design fragments into a new whole. In *Warbo*, the user creates a group of colored animated spots, each of which corresponds to a pure sine tone. A two-handed interface, which combines the use of a mouse and a Wacom tablet, then allows users to control how these tones are made audible.

Users begin a session with *Warbo* by selecting a color (and corresponding pitch) from a visual popup keyboard. The user can then create an animated spot, of which there are two possible styles: circular spots, or polygonal spots. The circular spots

animate according to the same technique used in John Maeda's *TimePaint* or my *Directrix*, in which they periodically retrace a recorded gesture path; the polygonal spots, on the other hand, are authored in the same manner as the animating polygons in *Polygona Nervosa*. These spots are represented as color fields which are brightest at their centroids, and fade to black at their boundaries; where spots overlap, their colors add and create a brighter region.

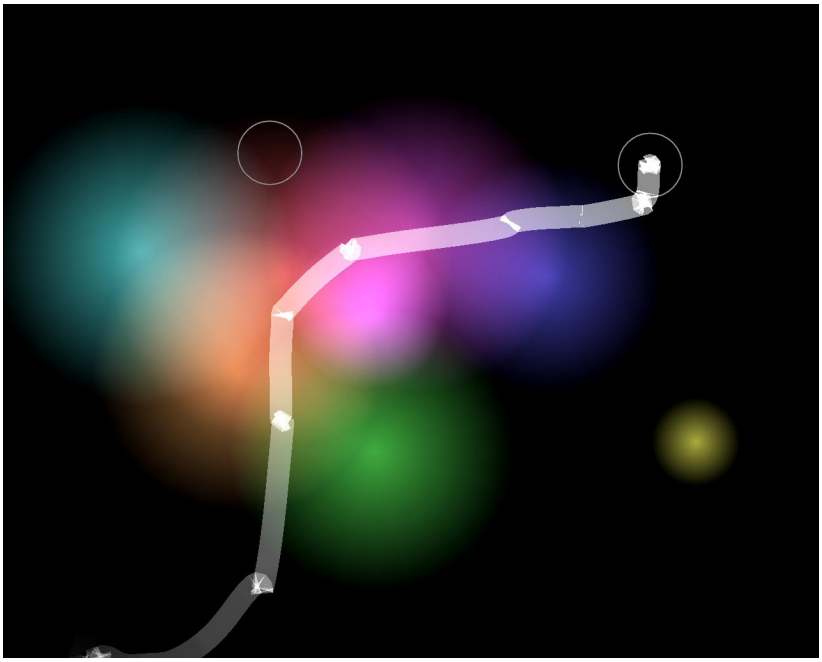


Figure 66. A screenshot from *Warbo*. The user has placed a number of colored spots on the screen, each of which corresponds to a certain sine wave. When the mouse-cursor passes over the spots, a chord is produced whose mixture is based on the position of the cursor in relation to the spots. Meanwhile, a Wacom pen in the user's other hand controls a *Streamer* line whose shape governs the timbral content of the chord.

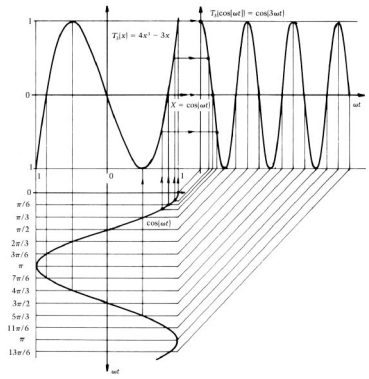
Passing the mouse-cursor over the colored spots produces a chord whose tones correspond to the spots which the cursor intersects. The relative volumes of the tones in this chord are mixed in inverse proportion to the distance from the cursor to each spot's centroid; thus, placing the cursor close to a spot's centroid will produce a louder tone, while placing the cursor at a spot's ephemeral boundary will produce a faint one. In this way it is possible to produce varied audiovisual compositions, by "playing" the cursor over the field of spots, or by permitting a collection of animating spots to sound themselves as they move underneath the cursor.

If *Warbo* is performed solely with the mouse, the chords and tones it produces are of a consistently smooth timbre—the result of summing a small handful of sine waves. It is also possible, however, to use *Warbo* with a Wacom pen in one's other hand. In

this case, the pen controls a second, independent cursor; drawing with the pen brings to life a *Streamer* trail (see Section 3.1.1) whose shape is used to govern the timbre of the chord played by the user's other hand.

The *Streamer* trail modulates the timbre of the user's chord according to an audio synthesis technique called *nonlinear waveshaping*. This technique was first developed by Jean-Claude Risset in the late 1960's, and is musically interesting because, as in FM synthesis, it provides a simple handle on the time-varying spectrum and bandwidth of a tone in a computationally efficient way [Roads 1996]. According to Curtis Roads, the fundamental idea behind waveshaping is to pass a sound signal ***x*** through a distorting function ***w*** (also called the *shaping function*), which maps any input value ***x*** in the range ***[-1, +1]*** to an output value ***w(x)*** in the same range. If the shaping function ***w*** is a straight diagonal line from ***-1*** to ***+1***, the output of ***w*** is an exact replica of its input ***x***. If it is anything else, however, then ***x*** is *distorted* by the shaping function ***w***, producing an enormous variety of musically useful results.

Figure 67. The waveshaping synthesis used in *Warbo*. A pure cosine source wave in the lower left is shaped by one of the Chebyshev shaping functions, $4x^3 - 3x$, in the upper left. The final output wave is shown in the upper right.



*Warbo* uses a special class of waveshaping functions called *Chebyshev polynomials*, which have the special property that each polynomial only emphasizes a specific harmonic of the ***x*** input wave. As a result, the use of Chebyshev shaping functions produces predictable, band-limited, well-behaved results. In *Warbo*, the curvature of each segment in the *Streamer* trail is mapped to the amount of a given Chebyshev waveshaper that is applied to the current musical chord. As the shape of the trail evolves in response to the user's gesture, shimmering harmonics brighten or dull the chord. In this way the mouse-hand controls the current pitch(es) and volume(s), while the pen-hand controls the timbre.

Figure 68: A table of the first seven Chebyshev waveshaping functions:

$T_0 = 1$
$T_1 = x$
$T_2 = 2x^2 - 1$
$T_3 = 4x^3 - 3x$
$T_4 = 8x^4 - 8x^2 + 1$
$T_5 = 16x^5 - 20x^3 + 5x$
$T_6 = 32x^6 - 48x^4 + 18x^2 - 1$

Chebyshev polynomials are defined by the recurrence relationship:

$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$.

*Warbo* is another example of an environment in which a user can create and manipulate an inexhaustible, dynamic audiovisual substance. Nevertheless, although *Warbo* produces musically interesting and expressive results, its visual dimension is something of an aesthetic hodge-podge: the visual relationship between its *Streamer* trail and its colored spots, for example, seems unmotivated. I include *Warbo* in this chapter because, although it was quickly hacked together from older components, its use of a two-handed interface points the way towards the future direction of this work. *Warbo*'s greatest contribution is precisely

that each hand can control different and complimentary aspects of an audiovisual expression.

### 3.2.4. Aurora: a "Swirling Cloud of Shimmering Gas"

#### 3.2.4.1. Origins

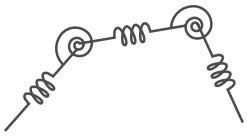


Figure 69. The inner mechanics of a finite-element model for physically simulating a hair-like filament: linear springs alternate with torsional ones.

In January of 1999 I began to study the means by which dynamic graphical lines might become able to convey a plausible sense of physicality. After some experimentation I developed a model for representing the underlying structure of "physical" lines, in which a finite-element, mass-spring-damper simulation is composed of virtual particles connected by alternating linear and torsional springs. The model I developed has the effect of simulating the tensile properties of thin physical filaments, such as hairs or twigs.

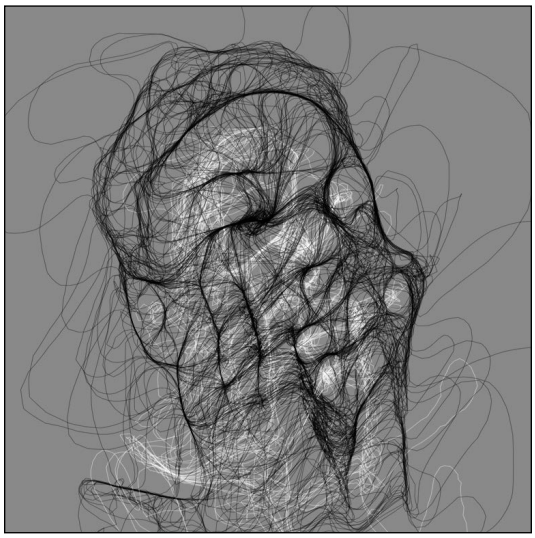


Figure 70. A portrait of my colleague Paul Yarin created with *Brillo*. In the *Brillo* system, hairlike filaments descend or ascend gradients of brightness in a grayscale photograph.

I developed two reactive drawing systems, *Brillo* and *Floccus*, which permit a user to construct images out of lines structured according to this physical simulation. In *Brillo*, lines drawn by the user are buffeted by forces derived from a hidden but underlying photograph. Light-colored filaments are attracted to bright regions of the photograph, while dark filaments are attracted to dark regions. I used these simple rules to coalesce piles of casual scribbles into several portraits of my colleagues. The results are wispy, organic and sometimes unsettling transformations: chiaroscuros in hair. Later, I learned that a similar technique, of compelling lines to perform lateral gradient descents on an image, had been developed as an edge-detection algorithm in the field of computer vision, where the method is called *active contours* or *snakes* [Kass 1987].

While *Brillo* was chiefly intended as a filter for photographic images, the software application *Floccus* was designed less for the sake of its final product—gooey balls of simulated hair—than for the enjoyable process of its interaction. In Floccus (the name is a Latin term for "hairball"), ductile filaments drawn by the user swirl around a shifting, imaginary drain centered at the user's cursor. The force mechanism which propels the filaments is based on a simplified model of gravity and is similar to that used in Scott Snibbe's *Gravilux*: an attractive force centered at the cursor attracts the simulation's particles with a strength inversely proportional to the square of the particles' distance [Snibbe and Levin 2000]. Whereas Snibbe's particles dramatically "slingshot" the cursor's location, however, filament elements in *Floccus* lines are so strongly coupled to one another by their own spring forces that they cannot overshoot the cursor without mutual interference. Instead—torn by conflicting impulses to simultaneously preserve their length, yet also move towards or away from the cursor—the filaments find an equilibrium by forming gnarly, tangled masses.

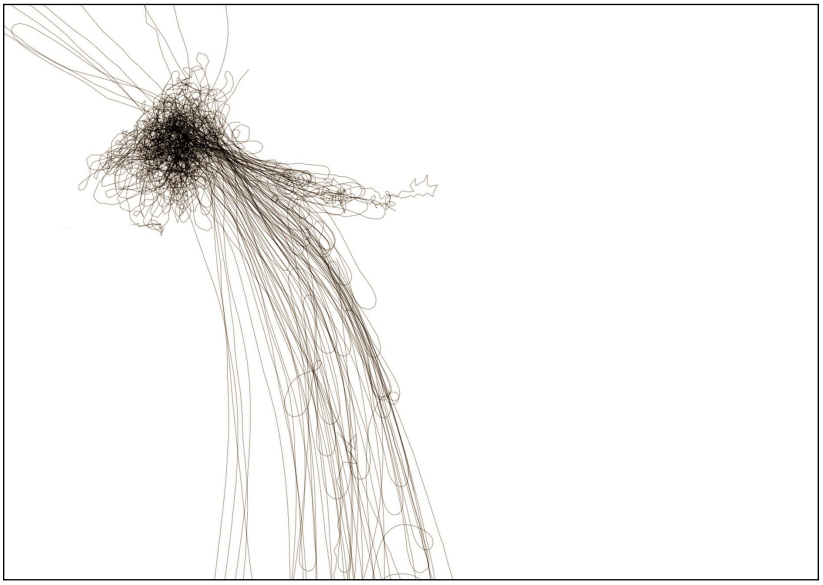
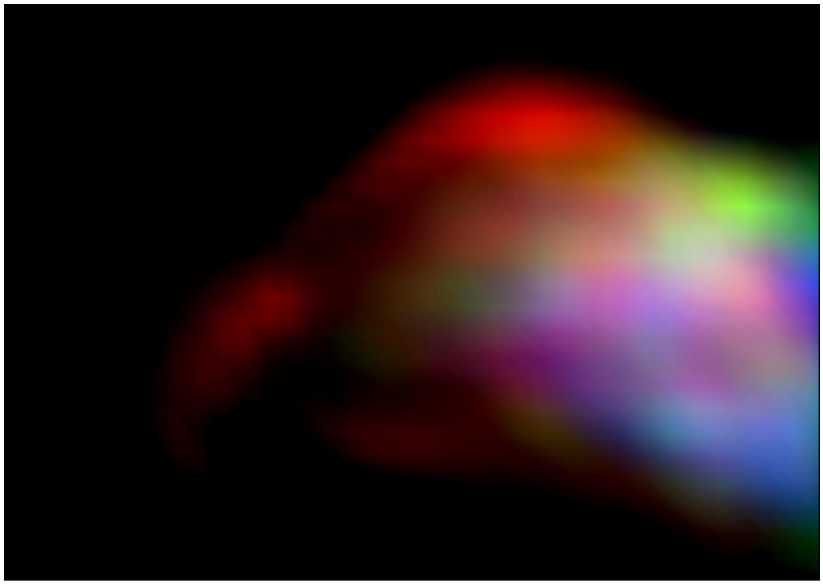


Figure 71. Simulated filaments in *Floccus* form tangled masses of curly hair.

One day in February 1999, while playing with the *Floccus* without my eyeglasses on, it occurred to me that the density map of *Floccus'* constituent particles might itself make an interesting display. Thus was born *Aurora*, a reactive system whose structural "underpainting" is a floccular simulation, but whose visual display consists instead of a blurry, shimmering, nebulous cloud. *Aurora*'s glowing formlessness rapidly evolves, dissolves and disperses as it follows and responds to the user's movements.

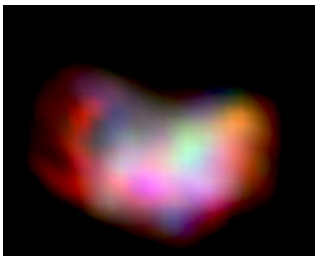
Figure 72. A line and its density field. Note how the field is darker where the line is coiled; this is a region of greater line-length per unit area.

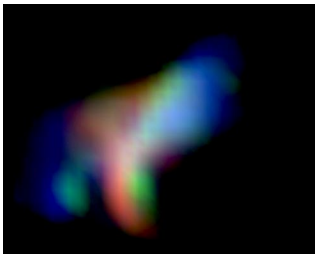
Figure 73. *Aurora* resembles a swirling cloud of shimmering gas.



The visual mechanism of *Aurora* is straightforward but structurally multilayered. At its heart is a hidden version of *Floccus* whose properties (gravitic constant, spring coefficients, damping values, etc.) have been slightly modified so that its hairs react with exaggerated vigor. Superimposed on the terrain of this simulation is a coarse grid of equally invisible square bins. Whenever the display is refreshed, each bin counts the number of floccular particles which occupy it, and assigns itself a brightness proportional to its contents. The bin cells are then visualized using a grid of smoothly-shaded quadrilaterals, which interpolate (with some added hysteresis) their neighbors' brightnesses into their own. By binning and low-pass filtering the simulation in this way, the thousands of data points in the floccular filaments are visually synopsized into an amorphous cloud. From the user's point of view, the result is an interactive painting in which the system establishes the basic constraint—that there is an animated cloud of color—but the user brings this cloud to life.

Figure 74. More stills captured from *Aurora*.





Color variations in *Aurora* are achieved by displaying each filament's density map with a different color. In one implementation, the interactant can use each of the mouse's three buttons to associate a filament (drawn with that button) with one of three available colors. These three colors have a triadic relationship (that is, separated by 120 degrees of hue on the color wheel) such that they produce pure white when added together in equal amounts. Torques experienced by the underlying floccular simulation during user interaction are then additionally put to the service

of rotating the system's color triad through the hue-space of the color wheel. It is possible to create nearly any color of cloud by carefully selecting and balancing the proportions of the colored strokes, "tuning" the colors if so desired by applying twirling forces to the simulation.

#### 3.2.4.2. Sonification

The sonification of *Aurora* was undertaken in March of 2000, and was inspired by Curtis Roads' magnificent cookbook of digital audio techniques, his *Computer Music Tutorial* [Roads 1996]. Scanning through Roads' book, I was struck by what seemed to be fertile similarities between *Aurora*'s floccular simulation and an audio technique called *granular synthesis,* in which a complex sound is built up from thousands of minute sound particles. Curtis Roads explains granular synthesis thus:

> "Granular synthesis builds up acoustic events from thousands of sound grains. A sound grain lasts a brief moment (typically 1 to 100 milliseconds), which approaches the minimum perceivable event time for duration, frequency, and amplitude discrimination. Granular representations are a useful way of viewing complex sound phenomena—as constellations of elementary units of energy, with each unit bounded in time and frequency....The grain is an apt representation for sound because it combines time-domain information (starting time, duration, envelope shape) with frequency-domain information (the period of the waveform inside the grain, spectrum of the waveform)" [Roads 1996].

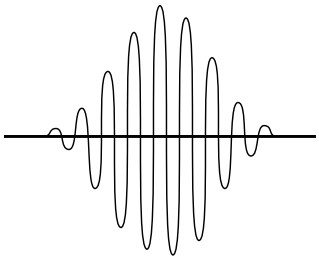


Figure 75. An example of a simple sonic grain, consisting of a sinusoidal tone-burst whose amplitude has been enveloped by a Hanning (cosine) window.

Roads' description of granular synthesis suggested a self-evident yet provocative opportunity to create a mapping between *Aurora*'s underlying floccular simulation—which can consist of as many as thirty thousand interconnected filament particles—and the thousands of sound-particles typically used in the granular synthesis technique. I chose to implement a straightforward variety of the technique called *Quasi-Synchronous Granular Synthesis* (QSGS), which generates one or more streams of grains, one grain following another, with a variable delay between the grains [Roads 1996]. I decided to associate each stream of sound grains with one of the floccular filaments produced by a user's gesture. Then I set about the problem of finding satisfactory, and hopefully well-motivated, mappings between the granular synthesizer's control parameters and the knowable features of the filaments' particles.

Granular synthesis makes a vast, malleable sonic terrain available to the instrument designer, at the cost of requiring the designer

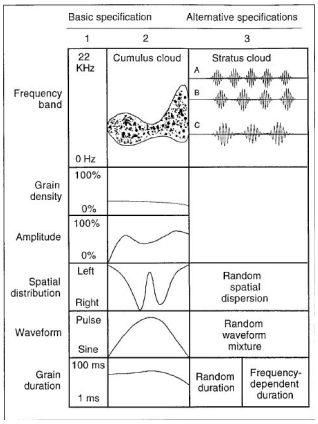



Figure 76. A pictorial representation of granular synthesis parameters. From [Roads 1996].

to stipulate a correspondingly enormous number of synthesizer control parameters. Roads observes: "If ***n*** is the number of parameters for each grain, and ***d*** is the average grain density per second of sound, it takes ***d* * *n*** parameter values to specify one second. Since ***d*** typically varies between a few dozen and several thousand, it is clear that for the purposes of compositional control, a higher-level unit of organization for the grains is needed" [Roads 1996]. Much of the granular synthesis literature discusses techniques for higher-level organization based on the idea of *stochastic or statistical control*. According to this schema, the parameters of individual grains are specified by randomization functions whose means, bounds and standard deviations are precisely and tightly controlled by the user's performance. The effect of statistical control, then, is the reduction of synthesizer control parameters from a few thousand per second to a considerably more manageable handful.

The decision to statistically control *Aurora*'s granular synthesizer was only half of the solution towards the environment's eventual sonification. A second unanswered question was the problem of how the behavior of the thousands of particles in *Aurora*'s underlying floccular simulation could be synopsized in order to control each grain's handful of "knobs." Some method of data reduction was necessary. The solution I developed—in which statistical measures drive statistical controls—became one of the chief innovations of the *Aurora* synthesizer.

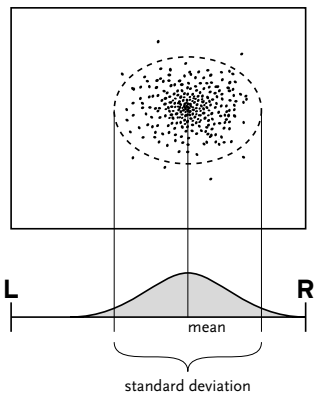



Figure 77. Mapping between the spatial distribution of particles, and the stereo placement of sound grains.

In *Aurora*, numeric measures derived from statistical analyses of the floccular simulation are mapped to the statistical properties of the randomization functions which govern the control parameters of the granular synthesis engine. A clear example of this is the manner in which the spatial distribution of the simulated particles is used to control the stereo placement of sound grains. Consider a relatively typical case in which the granular synthesizer is producing ten simultaneous streams of grains, and the duration of each grain is approximately 30 milliseconds. This means that the synthesizer will generate 300 sound grains per second, or about 10 sound grains during each frame of animation (at 30 frames per second). The question is, how shall these 10 grains be positioned in the stereo field, such that the location of the sound they produce during that frame is perceived to correspond with the visual cloud of particles? A simple answer would be to compute the horizontal centroid of the simulated particles, express that centroid as a percentage of the visual display's width,

and then situate all 10 sound grains at a position between the left and right audio channels equal to that same percentage. Unfortunately this solution fails to adequately reflect, in the audio, the difference between a narrow cloud and a broad one. *Aurora* solves this problem by randomly positioning the stereo location of each grain within a Gaussian distribution whose first and second moments are exactly equal to the mean (centroid) and standard deviation of the particles' horizontal positions in the simulation. The result is a "cloud of sound" whose spatialization not only reflects the exact location of the cloud on the screen, but also its ever-changing breadth.

There are seven unique control parameters which guide the construction of each sound grain in *Aurora*'s synthesis engine. In the table on the next page, I detail the precise mappings I established between specific statistical measures derived from *Aurora*'s floccular simulation, and the statistical control of these sonic parameters.

#### 3.2.4.3. Discussion

In *Loom*, we observed that continuously-varying properties of an animated graphical object could be used to continuously and directly control the parameters of a simple, equation-based synthesizer. In *Loom*, the relevant properties of the simulated graphical objects (such as their curvature and thickness) are *few in number*, as are the control parameters of the system's FM audio synthesizer (e.g., index of modulation, carrier/modulator ratio, etc.). It is technologically straightforward to map a property of *Loom*'s visual simulation to a specific variable in its synthesis equation, leaving the problem of selecting a suitable mapping to the domains of aesthetics and perceptual science.

The creation of a mapping between *Aurora*'s graphical simulation and an audio synthesizer is a qualitatively different problem, because the raw material for sonification—namely, the positions and velocities of a swarm of filament particles—is represented by *tens of thousands* of numbers. This condition had important implications for the selection of a sonification technique for the *Aurora* system. Since mappings from high-dimensional spaces to low-dimensional spaces necessarily require a reduction or loss of information, while mappings from low-dimensional spaces to high-dimensional spaces require the extrapolation or invention of information, I posited an information-theoretic principle for audiovisual instrument design: that domains between which

| ***Aurora* Synthesizer Control Parameter** | **Particle Simulation Property** |
|---|---|
| **grain carrier center frequency** the mean frequency of the waveforms inside each grain. | Each grain's carrier wave is assigned a pitch which falls within a specific Gaussian distribution. This distribution's mean is centered at the **grain carrier center frequency**, and has a standard deviation equal to the **grain carrier bandwidth**. The center frequency is functionally dependent on both the mean velocity of a filament's particles, and the average amount of distension experienced by the filament's linear springs. The effect is that a more vigorously moving cloud of particles produces a generally higher-pitched sound. |
| **grain carrier bandwidth** the range of possible frequencies within which the grains' carrier tones vary, centered on the carrier center frequency. | The frequency range or **grain carrier bandwidth** is proportional to the standard deviation of the particle velocities, plus a small term proportional to the mean torque experienced by the particles. The effect is such that when there is a lot of variation in how particles are moving, a greater variety of pitches are heard. |
| **grain duration floor** the shortest possible duration of any grain. | Each grain is a assigned a random duration. This duration can be as short as the **grain duration floor**, or it can be as long as the floor plus the **grain duration range**. The floor is functionally related to a measure of the particles' vigor, such that a less vigorous cloud of particles will tend to produce grains which are longer and more easily perceptible as discrete sonic events. The measure of vigor is derived from a weighted combination of the particles' mean velocity and mean distension. |
| **grain duration range** the range within which the grains' durations may vary, above the minimum grain duration. | The **grain duration range** is a function of the number of particles in each filament. Longer filaments produce a narrower range of durations. The effect is that large clouds (those made from longer filaments) sound smoother, while smaller clouds sound more irregular in a manner which corresponds with their commensurately more visible inhomogeneities. |
| **stereo placement** the left/right balance of a given grain's position in the stereo field. | Each grain's stereo location is positioned within a Gaussian distribution whose first and second moments are exactly equal to the mean and standard deviation of the particles' horizontal positions in the floccular simulation. |
| **grain carrier waveform mixture** the relative mixture between a sine wave and a square wave, that a given grain's carrier tone. | Each grain's carrier waveform can be a blend between a sine wave and a square wave. The waveforms are assigned to be more squarelike in proportion to the mean velocity of the particles in the simulation, thus producing spectrally brighter sounds when the cloud of particles moves vigorously. |
| **delay between grains** the amount of time between the completion of one grain's envelope and the onset of the subsequent grain in that stream. | The **delay between grains** is extremely short, and is slightly randomized in order to prevent artifacts of over-regularity such as extraneous formants. The delay decreases mildly as the number of points in the simulation increases, producing a smoother sonic texture for large clouds, and a more irregular texture for lumpy clouds. The effect of the irregular delay times is a controllable thickening of the sound texture through a "blurring" of the formant structure [Roads 1996]. |

Figure 78. A table of the specific mappings between the analyzed simulation measures, and the granular synthesis parameters, used in the *Aurora* synthesizer.

we create mappings should be of approximately commensurate dimensionality. With this principle in mind, I selected granular synthesis as *Aurora*'s sonification technique because it produces large-scale sonic events in an analogous way to that in which *Aurora*'s graphic system generates macroscopic visual effects: by aggregating the individual contributions of a myriad of infinitesimal "atoms."

Although the sheer numbers of floccular-simulation data points and granular synthesis knobs were roughly commensurable, it was nevertheless not possible, nor even sensible, to directly map each number in the simulation to a specific numeric control in the synthesizer: the fact remained that each body of variables described a different domain—one of space, the other of time. To bridge the two domains, I adopted the *statistical distribution* as an intermediate representation. Although this representation had the unfortunate effect of collapsing the dimensionality of both the simulation's information and the synthesizer's control parameters, it worked remarkably well at translating the subtle dynamics of one domain into the behavior of the other. *Aurora* achieves an extremely tight connection between sound and image because the aggregate behaviors of both have been carefully matched.

In sum, *Aurora* presents an audiovisual environment in which a user is able to create and manipulate an inexhaustible, dynamic,

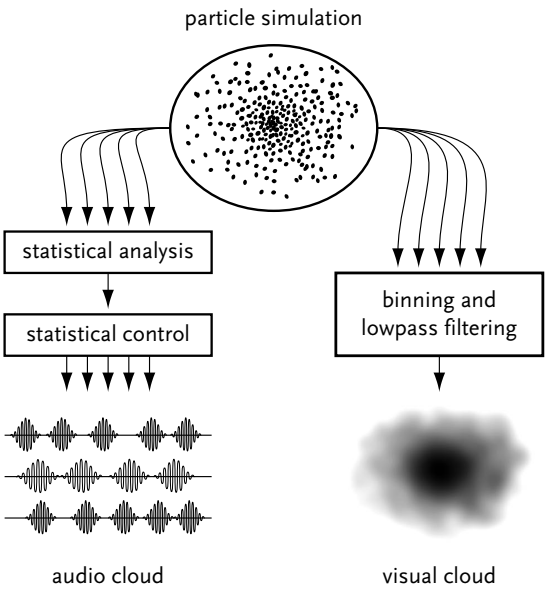



Figure 79. The relationship of *Aurora*'s image and audio generators to its underlying particle simulation. The functional couplet of [*statistical analysis*] and [*statistical control*] is used to conform the dynamic behavior of the audio to that of the visual simulation.

audiovisual substance. This substance is arranged in time and space according to an abstract language of plastic form. Although this substance is amorphous, the characteristic problems of controlling analogously amorphous physical media have been solved or circumvented through the use of computational simulation methods. *Aurora* extends our technical vocabulary by offering the *statistical distribution* as a powerful intermediate representation we can use when creating mappings between high-dimensional domains. Although domains that are constrained to share statistical distributions will, broadly speaking, share many dynamic behaviors, it is nevertheless important to remember that this technique is subject to information loss and will therefore interpose this loss between the user's input and the final audiovisual display.

### 3.2.5. Floo

*Floo* is an interactive audiovisual environment constructed around a Navier-Stokes simulation of fluid flow. Users create synthetic sound and image by depositing a series of fluid singularities (sources and vortices) across the terrain of the screen, and then steering a large quantity of particles through the flow field established by these singularities. An image is gradually built up from the luminescent trails left by the particles; at the same time, sound is generated by a granular synthesizer whose parameters are governed by the dynamic properties of these particles.

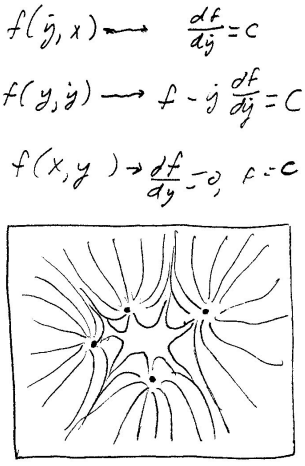


Figure 80. An early sketch of *Floo*'s underlying structure.

Users begin a session with *Floo* by clicking with the mouse or pen somewhere on the screen. At the moment the user clicks, a large number of particles emerge from the click location. These particles spread outward from this spot, propelled by the combined action of two forces: firstly, a force that pushes the particles away from the user's cursor; and secondly, a force that pushes the particles away from the click location, which is treated as a "fluid source." If there is more than one such fluid source, their corresponding forces on the particles are summed. All of the forces, including the force away from the cursor, obey an inverse-square dependency on distance. It is possible to create "vortices" (singularities which apply tangential forces instead of radial ones) by clicking with a different button. When particles reach the edge of the screen, they restart from their original click location.

Users can "paint" an image with the luminescent trails left by

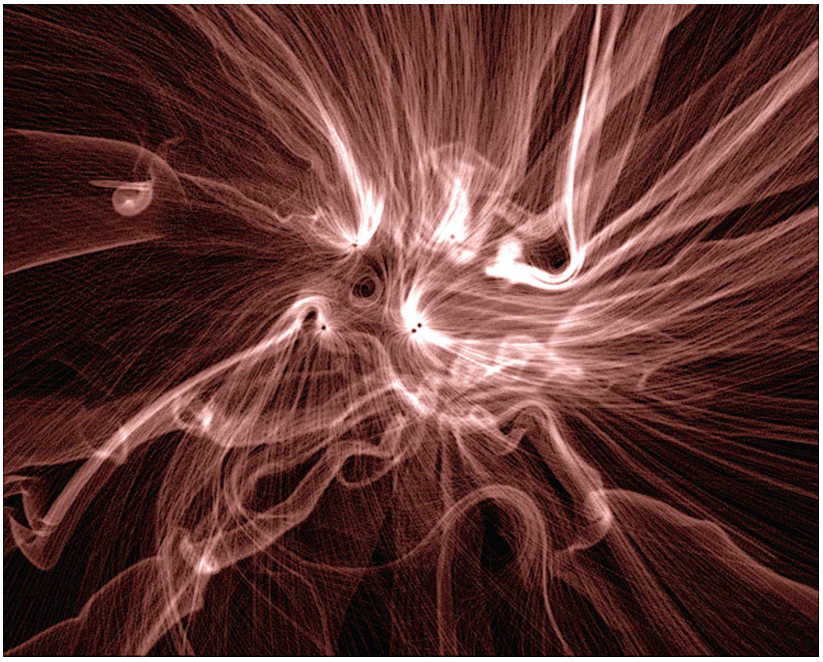


Figure 81. A screenshot of *Floo* in use.

the particles. The shapes of these trails are entirely a result of the radial and tangential forces originating from the user's cursor and singularities. As the particles tread again and again over a given location, that spot becomes brighter and brighter.

The sound in *Floo* is generated by a granular synthesizer. Certain aspects of this synthesizer are similar to the granular synthesizer used in *Aurora*; for example, all of the grains use a Hanning window (a raised inverted cosine shape) for their amplitude envelope. One difference, however, is the number of simultaneous streams of sound grains: while *Aurora* allocates one stream per filament (on the order of ten), *Floo* allocates one per particle (on the order of hundreds).

The most important difference between the *Floo* and *Aurora* granular synthesizers is the shape of the waveform with which *Floo* fills its grains. While *Aurora* fills its grain envelopes with simple waves like sine tones and square tones, *Floo* uses a unique and complex kind of sound called a *Shepard tone*. Shepard tones are a form of "audio illusion" discovered in the 1960s by Roger Shepard, a research psychologist at the AT&T Bell Laboratories. According to Richard Moore, "these tones exhibit the peculiar property of "circular pitch," in the sense that if one constructs a musical scale out of them, the notes eventually repeat themselves when the notes are played consecutively upward or downward. [...]

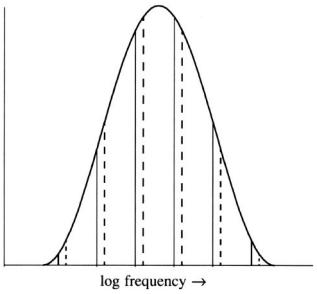



Figure 82. The spectrum amplitude envelope of Shepard tones. As spectral component amplitudes fall to zero (moving either upward or downward) at one end of the spectrum, they are replaced by new ones entering at the other end. From [Moore 1990].

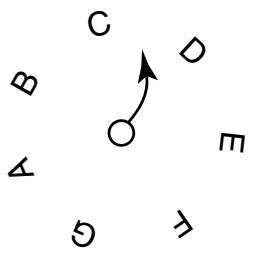



Figure 83. Floo's granular synthesizer maps a particle's orientation to the pitch of a Shepard tone in a grain. In this way, the particles can move in a seamlessly circular pitch space.

A visual analogy of Shepard tones might be the optical barber pole illusion, in which helical stripes painted on a rotating cylinder appear to move up or down as the cylinder rotates without ever moving off the ends of the pole" [Moore 1990]. Shepard tones can thus be used to create the illusion of a continually increasing or continually decreasing pitch.

*Floo*'s granular synthesizer maps the *orientation* of a particle's velocity to the pitch of a Shepard tone used in a stream of grains. Thus, the sonified particles move in a seamlessly circular pitch space. Particles which move in opposite or different directions will create chords, while particles which move in similar directions will create thick, chorused drones. This mapping and others are discussed further in the table below.

Figure 84. A table of the specific mappings between the particle simulation and the granular synthesis parameters used in the *Floo* synthesizer.

| ***Floo* Synthesizer Control Parameter** | **Particle Simulation Property** |
|---|---|
| **grain carrier pitch**<br>the apparent pitch of the Shepard tone inside each grain. | Each grain's carrier wave is assigned a Shepard tone whose pitch in the tone wheel is directly derived from the orientation of the particle's bearing. The effect is that the user can control the pitch of a particle by steering it in one direction or another. |
| **grain duration**<br>the duration of the grains. | The grain duration is a constant. The result of this is that the overall sound has a dronelike quality, whose fundamental is a function of the grain duration, and whose formants and harmonics are a function of the grain's carrier wave. |
| **stereo placement**<br>the left/right balance of a given grain's position in the stereo field. | Each grain's stereo location is positioned at its corresponding particles' horizontal position in the display. |
| **delay between grains**<br>the amount of time between the completion of one grain's envelope and the onset of the subsequent grain in that stream. | The **delay between grains** is extremely short, and is slightly randomized in order to prevent artifacts of over-regularity such as extraneous formants. |
| **grain volume**<br>the amplitude of a given grain | Each grain's amplitude is functionally related to the speed of its corresponding particle, as well as the distance from its particle to the cursor. |

*Floo* presents an interaction which brings together the complementary affordances of discrete and continuous gestures: discrete clicking creates new sets of particles and spatial configurations of fluid flow, while continuous cursor movements guide particles in real-time. There are many ways to play *Floo*; rapid clicking, for example, creates bursts of sound associated with visual pock-marks, while slow cursor movements create subtly changing drones that are associated with long pseudopods of light. If the system is left alone for some time, the particles eventually slow down, and the sound and image fade away.

### 3.3. Summary

This chapter presented five new interactive systems, developed over the last two years at MIT, which make possible the simultaneous creation and performance of animated imagery and sound: *Yellowtail*, *Loom*, *Warbo*, *Aurora*, and *Floo*. These five systems make available to the user an inexhaustible and dynamic audiovisual substance which can be freely deposited and expressively controlled. Importantly, the latter four of these systems permit this substance to be situated in contexts which employ the non-diagrammatic visual languages of abstract painting and film.

In the next chapter, the failures and successful contributions of these applications will be teased apart in greater detail, with the intent of extrapolating their design principles, and discerning the axes of the design space they inhabit.

# 4. Discussion and Analysis

In Chapter Three, I presented four software systems for visual performance, and five systems for the performance of animation and sound together. In this chapter, I attempt to address the questions: How do these systems work? In what ways do they fail? What are the metrics by which we can compare the success of these systems, and how do they measure up? To approach these questions, I have divided this chapter into four sections: *Design Patterns and Opportunities*, where I discuss some of the design patterns that structure the software environments; *Challenges and Pitfalls*, in which I describe some of the design patterns I have sought to avoid; *Comparative Examination*, in which the different applications are situated along a set of analytic axes, and thereby contrasted with one another, and *Evaluation*, in which the works are measured individually and collectively against a set of appropriate standards.

## 4.1. Design Patterns and Opportunities

In this section, I highlight a set of design patterns which have implicitly or explicitly structured my thesis software applications. Certain patterns, such as *Gesture Augmentation Through Physical Simulations* or *Alternative Visual Representations*, have been used extensively in the thesis software. Other patterns, such as *Augmentation and Sonification From Intermediate Representations* or *Gestural Inputs with Greater Degrees of Freedom,* have been only touched upon in this thesis, and represent promising opportunities for further research.

### 4.1.1. Interaction Schema: Capture/Augmentation/Governance.

An examination of my thesis applications reveals that each of the systems uses one of three basic schema for structuring audiovisual performance: *gesture capture, gesture augmentation, and gestural influence (or governance)*. These schema describe different technologies for relating a system's input to its output, and have important implications for the degree to which a user feels like they are directly controlling, versus indirectly guiding, a display.

#### 4.1.1.1. Gesture Capture and Playback

The ability to record a user's gesture can be an extremely useful element in the design of a computational performance instrument. Of course, almost any paint program is in some sense a gesture-capture system, since it "records" a user's movement

as a two-dimensional stroke or mark on a virtual canvas. In this case, I refer to a kind of gesture capture in which the *temporal* component of a user's mark is recorded and played back, in addition to its spatial properties. The advantages of this gesture capture technique are its tremendous capacity to produce lively, organically-animated results, and the exceptionally tight relationship it establishes between the user's input and the system's output.

Important examples of interactive creation environments which have used gesture capture are Myron Krueger's *Videoplace*, Scott Snibbe's *Motion Phone*, and John Maeda's *Timepaint*. Although gesture capture is used in several of the applications I present in this thesis, it is important to point out that it is not always used in the same way—as a discrete conceptual design element, gesture capture can be neatly separated from the use to which it is put. Thus, in the works which support this thesis, it is used in *Warbo* and *Directrix* to control a point, *Curly/Yellowtail* and *Loom* to control a line, and *Polygona Nervosa* to control a shape.

#### 4.1.1.2. Gesture Augmentation

*Gesture augmentation* refers to a large class of techniques which apply some form of transformation to the spatio-temporal properties of a user's gesture. These techniques can produce a wide range of interesting results: the user's movements may be amplified, shrunk, sharpened, dulled, embellished, simplified, reversed, echoed, repeated, accelerated, fragmented, joined, etc. etc. Taken as a group, gesture augmentation techniques can be used to present the user with feedback systems whose rules are not always immediately apparent. The experience of accustomizing oneself to these rules, if these rules are regular enough, can be quite enjoyable. In this thesis, I have identified and used at least three varieties of gesture augmentation techniques: methods based on geometric transformations, methods based on signal filters, and methods based on physical simulations. In many cases, these techniques can be applied to their own results, producing recursive feedback systems with interesting behaviors. These kinds of techniques can also be easily combined with each other to produce further flavors of augmentation.

Augmentations based on geometric transformations include rotation, translation, and scaling. My piece *Curly/Yellowtail* is a simple example of a system which augments a gestural mark through the geometric transformation of translation; see, for

example, Figure 55. Of course, geometric augmentations may be based on more sophisticated mathematics as well. My piece *Directrix* is an example of this; in addition to employing a gesture capture technique for the control of its parabolas' foci, it also uses the geometric augmentation technique of parabola construction in order to produce its graphics. *Directrix* converts straight marks into parabolas, and other marks into stranger marks—a classic and modestly complex geometric augmentation.

Augmentation techniques which produce results by contrasting previous information with current information can be broadly characterized as "signal filtering" augmentations. These include lowpass filters, highpass filters, bandpass filters, and notch filters, and can be applied to a mark's spatial information, its temporal information, or both. These augmentations produce the best results when used recursively; that is, when continually applied to their own previous results. In this thesis, two systems which use such recursive filtering methods are *Streamer*, which progressively magnifies the mark with a gently exponential feedback, and *Escargogolator*, which progressively "unwinds" a mark. A convolution kernel need not be explicitly specified in order for an augmentation to have filter-like properties.

The third variety of augmentation techniques I have identified and used in this thesis are those based on a *physical simulation*. It is something of a red herring to distinguish augmentations based on "physical simulations" from those based on "filters", since all digital filters can be said to represent a simulation of some (abstract) physical system, and all computational physical simulations will have some properties of filtering systems, such as phase lag, susceptibility to resonance, frequency cutoff and response, etc. Paul Haeberli's *DynaDraw* is an especially good example of a drawing environment which treads the boundary between a "filtering" augmentation and a "physical simulation" one, since its augmentation can be equally viewed as a resonant lowpass filter or a bouncy spring. Nevertheless, gestural augmentations based on physical simulations are uniquely capable of producing the feeling that one is creating and manipulating a plausibly real material. An example in this thesis is *Floccus*, which is based on a finite-element particle system composed of simulated springs, masses, and dampers. Many other simulation systems are possible; one can imagine, for example, a hypothetical system which allows its user, through selective acts of shattering, to construct images out of the physically-simulated "cracks" in a brittle surface.

#### 4.1.1.4. Gestural Governance

A third kind of interaction scheme explored in this thesis is a "gestural governance" model, in which the behavior of an animated display is governed or influenced by the user's movements. This scheme is closely related to the augmentation scheme, but differs in the extent to which the user perceives herself to be modifying a material which is *independent* from her own marks. In this scheme, an independent substance responds to the user's gesture according to its own internal logic. Two of the experimental systems described in this thesis follow this schema: *Aurora*, in which the user can influence the movements of a boisterously swirling cloud, and *Floo*, in which the user can "guide" the behavior of a glowing, spreading fluid.

The risk of the gestural governance technique is that the audiovisual material appears able to take on a life of its own. If the display seems equally content to produce color-music with or without the user's intervention, the user herself may begin to sense that her input is irrelevant or unnecessary. This can present a very real disincentive for further interaction. A partial solution to this is to design the system such that the audiovisual material is itself brought into being by the user. Another partial solution is to conserve the energy imparted to the system by the user—so that when the user ceases interacting with the system, it gradually loses energy to simulated "friction" and comes to a halt. These two techniques are used in the underlying simulations of *Aurora* and *Floo*.

### 4.1.2. Augmentation and Sonification Based on Analyzed Representations

A user's gesture is an information-bearing signal. When a mark is digitized, however, its "content" may not be immediately evident from the raw numbers which represent its **(x, y, t)** coordinates. Is the mark especially wiggly? Or jagged? Or straight? Does the mark resemble a letter of the alphabet? Or a previous mark made by the user? The answers to these questions can be used as the inputs to a variety of interesting augmentations, such as the filter-based or simulation-based augmentations described above. To find the answers, we can turn to signal analysis techniques.

Signal analysis is a broad term for techniques which extract information from raw temporal or spatial data. Some of these analysis techniques may calculate simple geometric properties of a mark (e.g. curvature, orientation), while others may

calculate statistical properties (e.g. centroids, moments of inertia, eigenfeatures) or create representations of the mark in the frequency domain (e.g. Fourier analysis). Pattern-matching techniques can be further applied to the results of these analyses, yielding additional representations of discrete information (e.g. the categorization of a mark as a specific shape or letter) or continuous information (e.g. the judgement that a given mark is 85% similar to another).

A simple example of the use of an intermediate representation can be found in *Escargogolator*, which uses a mark's curvature as the input to a filter-based augmentation (in this case, an IIR high-pass filter). In other of the applications presented in this thesis, intermediate representations from signal-analysis techniques were used to drive the control parameters of audio synthesizers, such as in *Loom* (which uses the curvature of a mark) and *Aurora* (which uses a variety of statistical measures). Nevertheless, this thesis has barely touched on the possibilities afforded by such intermediate representations, and the use of these representations in the design of audiovisual performance systems represents an important area for further research.

#### 4.1.3. Functionally Overloaded Gestural Inputs

One useful component in designing an audiovisual performance system is the idea that a single gesture can be used as both a spatial specification as well as a temporal specification. *Curly/ Yellowtail, Loom, Escargogolator* and *Streamer*, for example, all provide various means for simultaneously specifying the shape of a line, as well as its quality of movement. *Polygona Nervosa*, similarly, makes it possible to simultaneously specify the shape and dynamics of polygonal or bloblike forms. This kind of overloading presents unique opportunities for the design of audiovisual environments, which (for example) can make ready use of the spatial specification for the visuals, and of the temporal specification for the evolution of the audio and animation.

#### 4.1.4. Functionally Interrelated Gestural Inputs

Related to this is the idea of *functionally interrelated gestures or actions*, in which different kinds of actions serve specialized roles in creating the final expression. In *Directrix*, for example, one kind of action creates a spatial specification (the shape of the directrix), while a different kind of action creates a spatio-temporal one (the trace of the focus). Although the system can yield visible results if only one of these specifications is made, the system

produces the most sophisticated expressions when both are executed. In *Floo*, clicking creates a spatial configuration of force-generating singularities; the manner in which the system's particles subsequently navigate this terrain is as much a function of the user's rollover movements, however, as it is a function of the location of these singularities. In *Warbo*, dragged mouse movements create a path for a shape, while mouse rollovers permit the canvas' shapes to be continuously "played"; a Wacom pen in the other hand performs an entirely different function, namely modulating the timbre of the sounds produced by the other hand.

These kinds of functional specializations of gestural input are common in the design and performance of traditional musical instruments; most stringed instruments, for example, require the use of one hand for strumming, plucking or bowing, while the other hand is used to clamp the strings on the neck or fretboard. Functionally specialized interoperating actions seem to be easiest to learn if they are made perceptually distinct. The use of a two-handed interface, such as that of *Warbo*, is a simple way of enforcing this kind of perceptual distinction; users rarely confuse one hand for the other, particularly if each hand is holding a differently shaped device. Another way to enforce such a distinction is to leverage the perceptual contrast between continuous and discrete movements, e.g. dragging and clicking, as in *Floo* or *Polygona Nervosa*.

#### 4.1.5. Gestural Inputs with Greater Degrees of Freedom

This thesis is specifically devoted to the use of gestural inputs which belong to the space of two-dimensional mark-making. It would be a misconception to assume that the information conveyed in such marks is necessarily limited to two-dimensional Cartesian coordinates. Although standard computer mice can only convey two continuous dimensions, for example, a variety of other devices for digital mark-making can convey five or more. The Wacom pen, for example, transmits five continuous dimensions of information in addition to the states of its buttons: X location, Y location, pressure, orientation (azimuth), and tilt (elevation). A pressure-sensitive pad controller from Tactex corporation allows for the continuous transmission of a dense mesh of pressure information—effectively conveying hundreds of simultaneous degrees of freedom—with the further feature that the raw information can be pre-filtered and segmented in order to track multiple input centroids simultaneously. The Haptek

corporation's *PenCat*, and Sensable Technology's *Phantom*, are force-feedback actuated devices for the input of 2D and 3D gestures; these devices track the additional dimensions of force that a user may apply to a stylus. The Ascension corporation's *Flock of Birds* is a six degree of freedom device, which transmits accurate information about its three-dimensional position and rotation. Some researchers, such as Pete Rice at the MIT Media Laboratory, have even used one-of-a-kind 2D inputs, such as the novel "Laser Wall" interface developed by Josh Strickon and Joe Paradiso; this unusual device can track the positions of multiple unencumbered hands across the surface of a large screen. No doubt other devices already exist or remain to be designed, which will convey even more information about a user's gesture, such as the strength of the user's grip on a stylus, the posture in which the stylus is held, or the size of a stylus' footprint on a flat drawing surface.

Each degree of freedom offered by these devices represents an opportunity, for the designer of an audiovisual performance system, to establish an expressive mapping between a system's input and output. Each additional such mapping, moreover, has the potential to increase the variety of possible expressions in a system—and therefore the depth of engagement such a system can offer. With the exception of the *Warbo* environment, which uses the Wacom tablet's location and pressure data in tandem with the mouse, this thesis has restricted itself to the two continuous dimensions and one discrete dimension afforded by the most ubiquitous of interfaces, the mouse. Although most of the applications built in support of this thesis might be adequately serviced by this device, it is certain that nearly all could have been vastly improved by the use of some other gesture-capture technology. It cannot be overemphasized that some of the greatest opportunities in the future development of audiovisual performance systems lie in the use of more sophisticated input devices.

Do interfaces for audiovisual performance systems need to be grounded in the physical act of drawing? As mentioned above, this thesis has restricted itself to the consideration of physical interfaces which have a clear, designed relationship to a two-dimensional drawing surface; the mouse and Wacom pen are the most familiar examples of the many kinds that are readily available. But entirely different families of interfaces are possible. Noting that the applications in this thesis are intended for

*audio*-visual performance, a more musically-inclined reader might suggest the possibility of using an interface whose design drew from the physical language of traditional musical instruments. There are, in fact, a number of such devices on the market, which are typically marketed to electronic musicians as alternative MIDI controllers: guitar controllers, electronic wind controllers, MIDI violins, etc. Oftentimes these devices produce no sound of their own, and instead produce continuous streams of gestural expression data. Although such devices could be used to control images as well as sound, I offer without proof the hypothesis that a "musical instrument" interface, if connected to an audiovisual performance system, would present considerable challenges for the creation and control of graphics. Although this is conjectural, I base this claim on my belief that the interface to any audiovisual performance system must provide some means by which the audiovisual substance can be deposited at a precise *location* on the canvas. Without an interface for precise spatial specifications, such a system will seem more like a musical instrument accompanied by a sound-responsive graphic, than a true audiovisual construction system in which the visual dimensions are as malleable as the sonic ones. Naturally, the reader is encouraged to inform me of any examples to the contrary.

#### 4.1.6. Alternative Visual Representations

In the computer, the digital information which represents a line, and the way in which that information is represented on the screen, are wholly disjoint. It is, of course, easy to forget this, and designers must frequently remind themselves of the plethora of graphic options available to them. The simple fact is that not every "line" must necessarily be represented as a line, and there are often interesting and valuable aesthetic gains to be had by exploring the alternatives.

This notion has been illustrated in some of the works which support this thesis, such as *Escargogolator* and *Aurora*. These two systems both present unique and indirect ways of visualizing a database of line segments: in *Escargogolator*, the trace of the user's mark is represented as a disconnected sequence of barlike elements, while *Aurora* represents the user's marks by the means of their density map. These design decisions don't just provide interesting variations in style or decoration; they also provide the opportunity to communicate other kinds of information. In *Escargogolator*, for example, the width of the barlike elements is

used to represent the *speed* of the user's trace, while *Aurora* brings into relief those regions of the canvas to which the user has added more material, while de-emphasizing the exact positions of the user's marks.

### 4.2. Challenges and Pitfalls.

In the course of surveying the current state of audiovisual performance systems, and in developing the set of software applications which support this thesis, I have come to identify a set of challenges and pitfalls which seem to crop up again and again in the design of such systems. Of course, it is not strictly the case that my thesis applications succeed in avoiding all of these pitfalls. In this section, I describe these snares, discuss the examples in which my own applications succumb to them, and make an effort, where possible, to suggest ways around them.

#### 4.2.1. rANdOmNeSs

Of all of the pitfalls facing a designer of audiovisual performance systems, the use of computationally-generated randomness is one of the surest. John Maeda puts the matter plainly in *Design by Numbers*, his primer for novice programmers:

> "The amateur may be tempted by the cheap thrills of randomness. Random numbers, noise, stochastics, or whatever you want to call the complete lack of control that serves as the root of techno-styled graphics, is a form of profanity that you should generally avoid. But in many ways, resistance may prove futile because complete control of a complex computational process is still something of a faraway goal and the allure of randomness can be overpowering. My personal philosophy has been that if you are going to use randomness, you should at least know where it comes from." [Maeda 1999]

The appeal of randomness, in theory, is that it can be used to introduce new "information" in order to keep an interaction fresh. Unfortunately, the problem with randomness is that it contains no information at all—and can therefore be misinterpreted as actual information by an unsuspecting human user. The user suffering from a random system may ask: "Did I just do ***x***, or did it just happen by itself?" "Am I controlling this, or is this behavior a coincidence?" In such cases, randomness is a confusing distraction which makes it more difficult for a user to understand and master an interactive system. This is especially problematic

for audiovisual performance systems, whose fundamental appeal is predicated on a tight cybernetic coupling between human and machine. Any arbitrary intervention in the feedback loop of performance detracts from the system's immersivity, and perforates the user's experience of creative flow.

In many cases the use of human gesture itself obviates the need for randomness. Human gesture, in particular, is already such a rich source of input, with its own stochastic and irregular properties, that additional randomness is hardly needed. This observation contributes, for example, to the design of *Streamer*, which magnifies the user's marks exponentially; it is not necessary to add any randomness to the system, since the behavior of the streamer is already so sensitive to the noise in the user's joints, and to the dust particles which clog the computer's mouse. Randomness can also be avoided through a deeper examination of the gesture: the use of signal analysis techniques, for example, can tease apart additional expressive dimensions latent in the user's gesture, such as the color of its noise or the formants in the gesture's frequency spectrum.

It is somewhat embarrassing to admit to the use of randomness, after making these exhortations. Nevertheless, I have chosen to use it in two specific instances. The first instance uses randomness as an alternative to a color-picker in *Polygona Nervosa*. When the user wants to select the color of the subsequent polygon, they press the spacebar on the computer keyboard. Each time they do, a small color chip on the screen displays a randomly-generated color. The advantage of this interaction is that it circumvents the design of an elaborate and possibly distracting color-picking interface. The randomness in this case is managed entirely by the user at their own volition, and its results can be easily overridden if the user so desires by pressing the key again.

Randomness is also used in *Aurora* as a way of generating certain parameters of sound grains in its granular synthesizer. Once again, however, this randomness is carefully and volitionally managed by the user. In particular, the means, bounds and statistical distributions of these randomized properties are precisely and deliberately matched to the means, bounds and statistical distributions of a particle simulation governed by the user's gestures. The result is that the user has precise control over a sonic texture, even though the microscopic details of this texture are the product of a stochastic process. Without statistical noise, the *Aurora* system would seem lifeless, overly-regular, and dull.

### 4.2.2. The Taste of Mathematics

Mathematical relationships are all around us, structuring both natural phenomena and human artifacts alike. Recently, some designers have come to believe that it is enough to invoke the mathematical symmetries of nature in order to produce meaningful artwork. These designers extol the "intrinsic beauty of mathematics," or something similar, and are responsible for the proliferation of "fractal art" and other spiral forms. Regrettably such works are only rarely more interesting, personal, or provocative than the equations which generated them.

At the same time, nearly all computational artwork inescapably involves, at some level, the programming or specification of mathematical relationships and equations! The medium itself is so deeply structured by these relationships that the entire field of computer artwork is often broadly regarded as cold or impersonal. Clearly, it is rarely enough to implement a mathematical expression and call it a work of art. Instead, the challenge in designing an interactive visual system is to *overcome* the mathematical materials which one must necessarily use, and surpass them in the service of some greater expression. To do any less is to risk the construction of an artwork which—as my colleagues in the Aesthetics and Computation Group occasionally say—"tastes like math."

Do the works described in this thesis overcome the taste of math? Some do; others do not. The least successful system in this regard, perhaps unsurprisingly, is *Directrix*, whose core idea is the construction of gestural parabolas. Although this system is able to produce organically thatchy compositions in the hands of an experienced user, it otherwise too often resembles a mathematics demonstration. The use of fluid dynamics equations in *Floo* sometimes produces shapes with the similarly mathematical and pedagogical taste of magnetic field lines. By and large, however, the applications which support this thesis have been designed to point toward a new aesthetic of organic computer artwork. It has been my observation that users of *Floccus* and *Curly*, for example, respond more frequently to the living quality of their graphics, than to the underlying mathematics which scaffolds them.

### 4.2.3. Cartesian and Diagrammatic Mappings

Often in an audiovisual system we will see that the designer has assigned the ***x*** axis to time, and the ***y*** axis to pitch, etc. Couldn't these axes be swapped? Or flipped? Or rotated diagonally, or made

non-orthogonal? The answer is of course that they can, because these assignments are the completely *arbitrary* products of the visual language of diagrams. The commonplace decision to map ***x*** or ***y*** input coordinates to simulation parameters is one which relies on an abstract and artificial convention, rather than on geometric intuitions that are more analogous to the natural world [Snibbe and Levin 2000]. Although Cartesian coordinates may be computationally and electromechanically convenient for input and output on raster-based systems, the reactions and processes of nature obey an organic logic which is often poorly described by relationships to a Cartesian grid. In designing the systems which support this thesis, I therefore adopted the more perceptually-oriented primitives of pre-Cartesian geometry, such as direction, velocity, orientation, and curvature, as the building blocks of these environments. These primitives are "read" directly by the eye, requiring no recourse to the labels of some axis for interpretation. By basing the visual dynamisms and aural sonifications on these *intrinsic* properties of gestural marks—as opposed to the *extrinsic* properties of some axis or grid—an extremely tight relationship between gesture, animation and sound can be the result.

Pitch is one of the most fundamental perceptual dimensions of sound. Because the spectrum of audible pitch is continuous and scalar, it is often convenient to map pitch to a linear axis in physical and graphic interfaces. Such a design strategy, motivated by basic physical principles (i.e., shorter strings and pipes produce higher pitches) has become a *de facto* pattern for physical musical instruments, and we therefore witness linearly-spatialized pitch interfaces in almost all classes of musical devices, such as keyboards, guitar necks, and the bores of wind instruments. Because these traditions are so deeply ingrained, the search for alternative interface organizations for pitch control is a difficult one. One alternative that this thesis work points to, however, is the substitution of spatialized interfaces with temporalized ones. Thus *Aurora*, for example, controls pitch with *force* and *velocity*—both time-based quantities. And, in fact, there are precedents for this in the real world, such as the holeless *shakuhachi* flute, on which different pitches are achieved by more forceful blowing. In mapping parameters to temporal controls instead of spatial ones, lies the possibility of freeing up the meaning-making affordances of the visual space itself.

Of the five audiovisual systems presented in Chapter Three, only one, *Yellowtail*, stoops to the use of an explicit, diagrammatic,

Cartesian grid. This is surely the weakest aspect of the piece, since this grid seems so irrelevant and so arbitrary when contrasted with the organic graphics which swim around and through it. All of the rest of the systems base their sonifications and animations on the *perceptual* properties of their marks and shapes.

### 4.2.4. Modal Interactions

It is difficult to design a software system which can be characterized by continuous interactions in a single state. For this reason, many GUI designers have become fond of modal interactions, which present a big payoff in terms of the number of possible operations and states that a system can offer. Unfortunately, although modal interactions are learnable, they are not necessarily easily intuitable, and they therefore frequently necessitate some form of user instruction. As a result, modal interactions are a potential pitfall which can make systems especially difficult for novices to use.

As a matter of principle I strongly recommend that instruments be designed with non-modal interfaces, in order to prevent the user from getting stuck or requiring recourse to instructions. If it is absolutely necessary for an instrument's interface to have different modes, it should be as easy and obvious how to switch between those modes as it is for a musician to switch from bowed to pizzicato violin, or for an artist to turn a pencil upside-down to use its eraser.

Nearly all of the systems which support this thesis operate without modal interactions anywhere in their interfaces. *Polygona Nervosa* is the marked exception, because it places users into a shape-creation mode which can only be exited by a special "shape-terminating" action. Unsurprisingly, *Polygona Nervosa* is substantially more difficult for novices to learn than the other instruments.

### 4.2.5. ROM-Based Solutions

The use of ROM-based media in the design of audiovisual performance systems instruments is both a challenge and a pitfall. It is in part a challenge, because the theoretical advantages of using pre-composed media (such as audio samples and bitmap images) are great, especially for the textural quality of a system's output: ROM-based sounds, for example, can sound incomparatively more rich, organic or familiar when contrasted with conventionally synthesized sounds.

Unfortunately there are substantial disadvantages as well, many of which have been covered as part of the background material in Chapter Two. Both the designer and user of ROM-based instruments suffer: from the designer's perspective, the use of pre-composed materials adds the burden of preparing and pre-composing those assets—it is much more tedious work to produce a large bank of high-quality samples than to declare a few functional relationships in code. A second disadvantage is that designing expressive handles for ROM-based media is a difficult task, since the contents of most assets are usually less than computationally intractable. The final disadvantage is for the user, who may lose interest in the system when the collection of canned materials becomes exhausted.

In this thesis I have emphatically stressed a design standpoint in which the designer ultimately specifies, through code, the origin of every pixel and every audio sample produced by the instrument. Of course, this standpoint shifts the designer's burden from the careful construction of assets to the careful construction of mathematical relationships. Are other methods possible? It turns out that the answer is yes. The boundary between "musical instruments" and "record players" is a fuzzy one, especially as the size of the pre-recorded materials approaches zero—hardly anyone, for example, would call a piano a "record player for piano sounds"! In the same way, new signal analysis and resynthesis techniques (such as wavelet resynthesis, granular resynthesis and phase vocoding) are emerging from research laboratories which are very good at expressively manipulating and combining small audiovisual fragments into larger wholes. Delving into these methods is beyond the scope of this thesis, but represents an important way that the texture of the real world can be made as malleable as synthetic materials.

### 4.3. Comparative Examination

In this section, I order the different audiovisual applications along a set of subjective analytic axes, in order to indicate some of the ways in which the systems contrast with one another. Throughout this section, a consistent set of icons (see Figure 85) have been used to represent the five audiovisual performance systems. I have chosen to compare the different systems along the axes of *learnability*, *predictability*, *expressive range*, *granularity*, and *perceived source of determination*. Each axis, moreover, has been split into its visual and aural aspects.


Figure 85. A key to the icons used throughout this section to represent the five audiovisual performance systems.

### 4.3.1. Learnability

How easy or difficult is a system to learn? Each of the five audiovisual performance environments has its own learning curve. *Aurora* seems to be the easiest to pick up; one creates the cloud of color, and pulls it around. *Floo*, on the other hand, invites its user to try to compose an image, but presents a subtle and sensitive method for doing so; it must be finessed, and learning how to handle the rules of its simulation takes some time. An interesting feature of the other applications, such as *Loom* or *Warbo*, is that it seems easier to learn how to perform their graphical dimension than their sonic dimension. I attribute this to the fact that it is easier to make something look inoffensive than sound inoffensive.

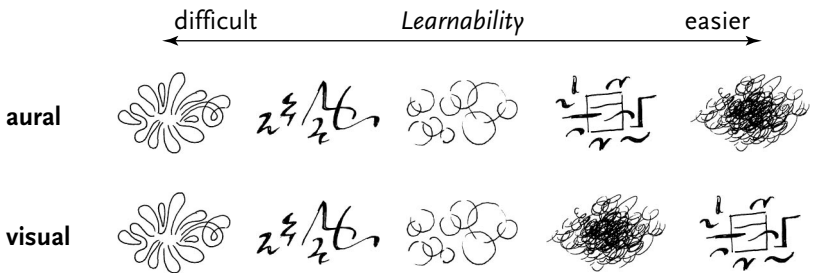



Figure 86. The systems' learnability.

### 4.3.2. Predictability

Closely related to the learnability of a system is its predictability, or the degree to which its output is a sensible consequence of its input. Here we see that applications which call for the precise spatial placement of graphic material, such as *Loom* and *Warbo*, are among the most visually predictable, while *Floo* trails in this regard because of the subtle nuances effected by its fluid dynamics. In this case *Floo* makes an especially interesting contrast with

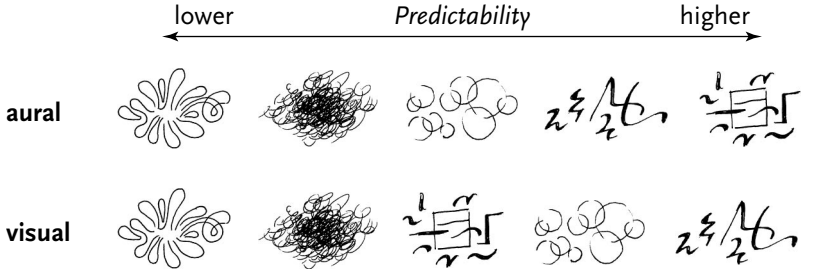



Figure 87. Predictability.

*Aurora*, as the dynamics of *Floo* are wholly deterministic, while the texture of *Aurora*, which seems to be much more predictable, is at its root stochastic.

### 4.3.3. Expressive Range

An important metric by which we may compare and evaluate the thesis systems is their expressive range—the breadth of possible results that they can produce. A system with a larger expressive range will be able to support longer interactions for a given user, while systems with truly broad expressive potential will support repeat interactions, and may even reflect the different expressive "voices" of different experienced users. In Figure 88, we see that *Yellowtail* has the broadest range of the set. Some applications, such as *Warbo* and *Loom* provide a wider range of possibilities in their sonic dimension than in their graphical aspect, while for *Floo* the opposite is true. *Aurora* is the most constrained system overall; although its audiovisual display is extremely fluid, its variety of perceptually distinct results is narrow—the cost of its amorphousness.

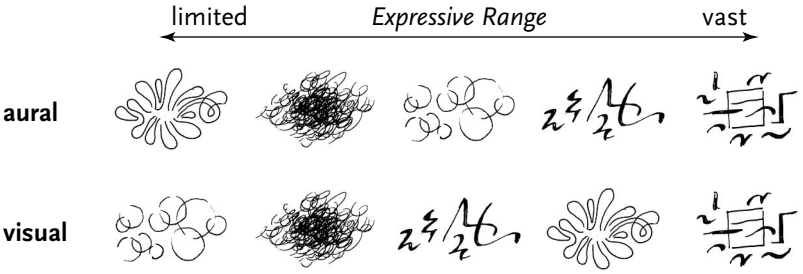



Figure 88. Expressive Range.

### 4.3.4. Granularity

The audiovisual systems which support this thesis present the user with a malleable audiovisual "substance." The *granularity* of this substance can have a substantial impact on how easy it is to produce expressions in that medium: a material's whose "grain" is too fine obliges its user to make an enormous number of specifications (or be satisfied with vague control), while a coarsely-

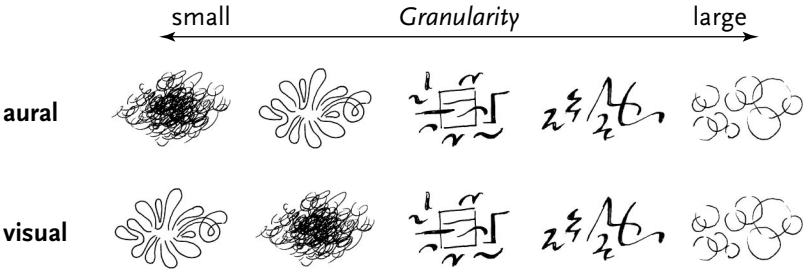



Figure 89. Granularity.

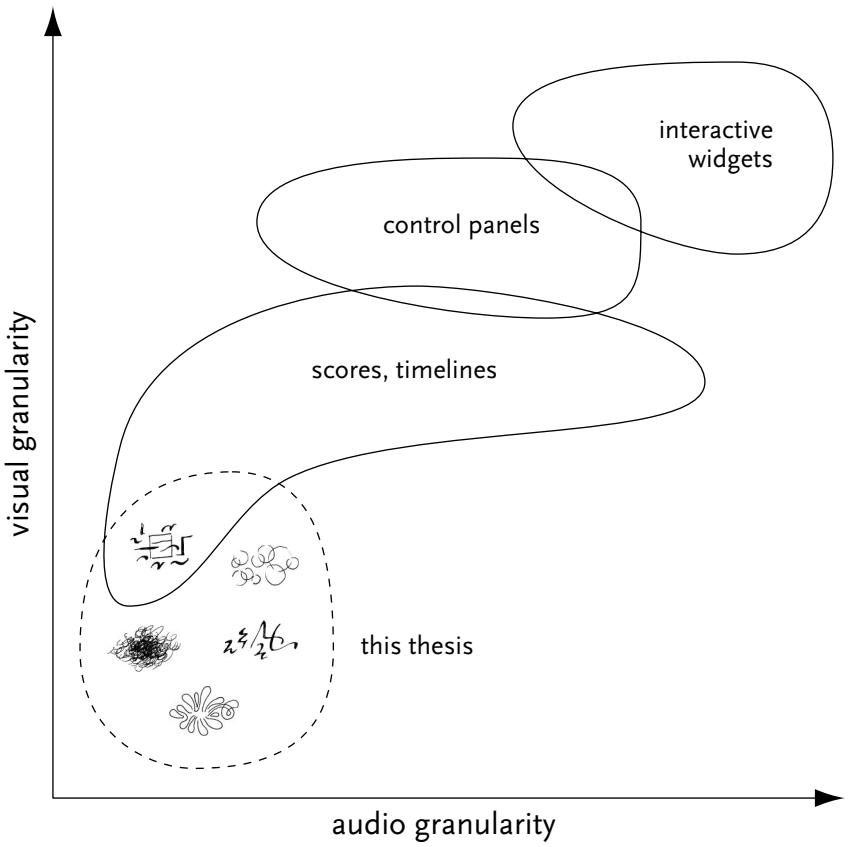



Figure 90. Granularity of the thesis instruments contrasted with the granularity of other classes of audiovisual performance systems: diagrammatic timelines, control panels, and reactive "widgets."

grained material is difficult to personalize. In Figure 89, we can see that the *Warbo* has the coarsest granularity, as its substance is composed of largish spots, while *Aurora*'s cloud of infinitesimal particles has the finest granularity. A happy middle ground can be found in *Loom* and *Yellowtail*, which use marks that can span a wide range of sizes.

The aural and visual axes of granularity critically define the expressive capacity of an audiovisual performance system. In Figure 90, I contrast the five thesis instruments with the broad classes of computational audiovisual tools which I discussed in Chapter Two: *scores and timelines*, *control panels*, and *interactive widgets*. We can see that the thesis systems compare favorably with these other media for audiovisual expression.

### 4.3.5. Perceived Origination.

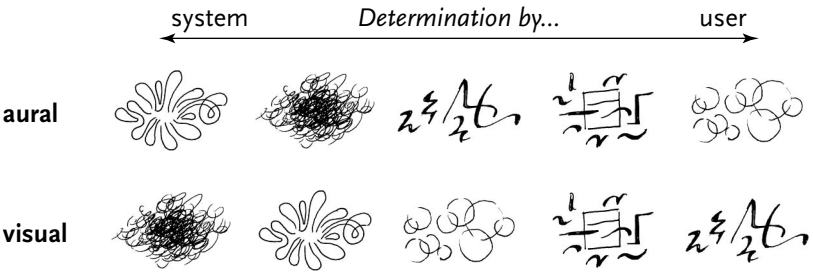



Figure 91. Perceived origination. Here we evaluate the extent to which the possible expressions in a given system's visual dimension, or aural dimension, seem determined by the user or by the system.

When we use a performance system we learn to sense the set of possible expressions dictated by the system. Any given composition made in the system thus represents a path the user has navigated within the dictated constraints. How forceful is the system's dictation—how strong are its constraints? Here we evaluate the extent to which the act and products of expression seem determined by the user, or determined by the system. The five audiovisual performance systems are represented in such a way that we can additionally compare the perceived origination of the systems' visual dimensions, with the perceived origination of their aural dimensions. We observe, for example, that *Aurora*'s visual dimension feels largely determined by the system; put another way, this is to say that the system seems to strongly determine that its visual output will look like a colored cloud, no matter how the user interacts with the system. This metric is closely related to expressive range; nevertheless, it differs insofar as one tightly-constrained system might produce a wide range of possible expressions, while another might only be able to produce a single possible expression.

In the adjacent figure, we can see how an "ideal" audiovisual performance system ought to fare, according to the above five metrics, and according to the goals laid out for such a system in Section 2.3.

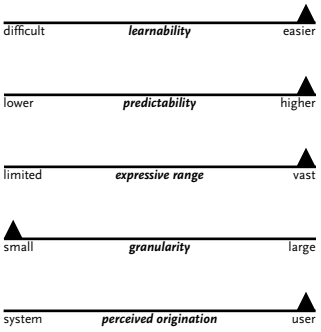



Figure 92. An "ideal" audiovisual performance system, evaluated by the metrics used in this chapter.

### 4.4. Evaluation

In order to evaluate the success or failure of the work in this thesis, it is helpful to establish the context in which the work is positioned and according to whose standards it should be measured. As with many MIT Media Laboratory theses, this is made difficult by the interdisciplinary nature of the work; the software systems that support this thesis inhabit a domain at the juncture of art, design, and the engineering of tools and instruments. As artworks, they fit within and extend an established Twentieth Century tradition in which artworks are themselves generative systems for other media; in Marshall McLuhan's terms, such systems are characterized by an "outer medium" (in my case, gestural performance and interaction) whose forms make possible the articulation of yet other expressions in an "inner medium" (for this work, synthetic animation and sound). Distinguishing such meta-artworks from the kinds of artifacts we conventionally call "tools" or "instruments" is largely a question of semantics and context; certainly the works I propose fit well within the usual definitions of these categories. I take exception to the "tool" label insofar as it carries with it the implication that a given tool is successful only if it is held to be useful and desirable by a broad base of consumers. I did not develop these systems with an audience of consumers in mind, but rather as vehicles through which I could explore and present a strictly personal vocabulary of design practice, and suggest new technological solutions for human-machine interaction. In this sense this thesis work has had more in common with a "Hyperinstruments model" of artistic activity and technological craft (e.g., in which an artist originates specialized tools for himself or herself), than to a commercial, "Adobe model" of populist software development (e.g., in which market-driven usability specialists refine plug-and-play solutions for efficiency-seeking consumers). Thus, although my software may coincidently have some potential marketability—an opinion drawn merely from my own observation that numerous people have enjoyed its use—I leave its evaluation by such metrics to those who are customarily concerned with maximizing this sort of value.

Instead of the marketplace, there are numerous other external contexts within which we may conceivably evaluate this work, such as peer-reviewed competitions, the music hall, and the art gallery, to name a few. In fact, this work has met with some

success in the first domain: during the Spring of 2000, the set of five audiovisual performance environments received a Bronze award in the ID Magazine Interactive Design annual competition; it was designated a Winner in the 6th Annual Communication Arts Magazine Interactive Design Competition, and also a Winner in the Digital2000 Annual Competition; and it has earned an Award of Distinction (2nd prize) in the Prix Ars Electronica 2000. Unfortunately, the degree to which these accolades reflect the success of the work is difficult to assess. The contemporary climate of interactive design is often alarmingly myopic, as much unaware of its own history as it is of the inner workings of its chosen medium, computation. Because so few interaction designers write their own software, designers are still too easily "wowed" by any artwork which exceeds the considerable technological constraints of their dominant tools, the *Shockwave* and *Flash* technologies by Macromedia. As a result, there is a scarcity of bases for comparison, and the critical dialogue remains at a generally low level. Although John Maeda's Aesthetics and Computation Group at the MIT Media Laboratory has done important work in elevating this dialogue and its standards, through the cultivation of a new breed of computationally sophisticated designer, it seems that it will nevertheless be some time before the works which compete in these venues reflect the greater promise of computation unfettered by the limitations of commercial authoring systems.

I submit that the software artifacts which support this thesis should minimally be able to support (A) a public performance by expert users, and (B) an engaging experience for interested gallerygoers. The success of the thesis systems in these real-world contexts will become clear over the coming months. In June of 2000, the systems will be displayed in the Emerging Technologies hall of the American Museum of the Moving Image in New York City; in July, the works will also be tested at the Sega Joypolis theme park in Tokyo. The audiovisual environments will receive their most strenuous test, however, in September 2000 at the Ars Electronica festival in Linz, Austria. There I have been invited to produce a half-hour concert of live color-music, performed on the thesis instruments by a quartet of artist-musicians, for the conference's headlining event. For this concert, the technical issues involved in transforming my current "demos" into robustly performable tools will be the least of my worries. Instead of tool design, I expect I shall be much more concerned with the aesthetic issues of composing a half-hour of passionate and compelling color-music. One might say it is about time.

For the present, the most important question is, do the systems succeed on their own terms? That is, do the systems present solutions to the challenge posed in this thesis: to develop an inexhaustible, dynamic, audiovisual "substance" which can be freely deposited and expressively controlled, in a context which is non-diagrammatic, readily apprehensible and deeply masterable? The answer, I believe, is yes. Obviously, the idea that any instrument could be "instantly" knowable or "infinitely" masterable is a worthwhile but unattainable goal; in reality, the instruments differ in the degree to which they are easy to learn, or capable of an seemingly unlimited variety of expression. One of the instruments, *Yellowtail*, fails to meet all of the stipulated criteria: it relies on a score-based solution for sonification. Broadly speaking, however, the audiovisual performance systems developed to support this thesis succeed in implementing a new, painterly interface metaphor for audiovisual expression, and therefore represent an important step towards a new type of creative medium.

# 5. Conclusion

This thesis represents an investigation into the history, design, and technology of instruments for the simultaneous performance of image and sound. In this last chapter, I briefly summarize the conclusions of the thesis work, and offer some of the directions for further research which it suggests.

## 5.1. Conclusions

The quest for an audiovisual "unity of the senses" is an ancient one, which extends at least as far back as the Classical Greek philosophers who developed tables of correspondence between colors and musical pitches. Although the recorded history of actual audiovisual instruments is only four centuries long, we may surmise that the roots of the idea are as old as music, shadow puppetry and painting themselves. A proliferation of audiovisual expression systems designed over the last hundred years—made possible by the technological affordances precipitated by the scientific, industrial and information revolutions—has dramatically expanded the set of expressive languages available to humankind. Many of the artists who developed these systems and languages, such as Oskar Fischinger and Norman McLaren, have also created moving and passionate expressions in them, in exemplary models of simultaneous tool-development and tool-use. The work in this thesis has aimed to follow in the footsteps of these innovators, exploring the creative potential of the latest and perhaps greatest tool for audiovisual expression yet, the digital computer.

The computer is a natural choice for such an exploration, as its fundamental material is pure information, essentially unfettered by the constraints of the physical world. A survey of current techniques for the visual control of sound on the computer, conducted at the beginning of this thesis work, revealed the existence of three popular interface metaphors: *timelines and diagrams*, *control panels*, and *reactive widgets*. Because each of these schema imposes fundamental and substantial constraints, whether visual or aural or both, on a system's expressive potential, I set for myself the goal of developing a system in which both the image and sound dimensions could be deeply, and commensurately, plastic. Eventually, a series of experiments to this end led to the articulation of a new metaphor for audiovisual creation and performance, based on the free-form visual language and gestural kinesics of abstract

painting. The kernel of this painterly metaphor for audiovisual performance interfaces—which merges the design intuitions of animation-performance systems with those of traditional musical instruments—is the idea of an inexhaustible, extremely variable, dynamic, audiovisual substance which can be freely created, deposited, manipulated, and deleted in a free-form, non-diagrammatic image space.

In the physical world, nearly any real object or material is an "audiovisual substance," possessing both an appearance and a set of associated or potentially associated sounds. These associations are intrinsic to our objects and materials and are essentially immutable: a drinking-glass basically looks like a glass, and the sound it makes when it drops and shatters is unmistakable. In the computer's world of pure information, however, the designer of a synthetic "audiovisual substance" must establish these associations entirely by himself or herself. Unfortunately, any such mapping which is created between an image and a sound will nearly always be an arbitrary or personal association: there is no "objective" mapping from sounds to image or vice versa. This is even the case, we learn from psychologist Richard Cytowic, for the rare individuals who are true synæsthetes; while these unusual persons experience strong mappings between pitches and colors, for example, these mappings are almost never the same across individuals. Although there are some general principles which we may extract from the work of the Gestalt psychologists, such as the idea that high-frequency information often occurs crossmodally, the designer of synthetic sound-image mappings must operate largely from intuition.

Can we do any better than merely guessing? The answer, it seems, is yes, because some mappings are less arbitrary than others. One of the most important contributions of this thesis' painterly metaphor for audiovisual performance is the idea that we may eschew mappings based on the arbitrary conventions of visual language, or the arbitrary affordances of computational technologies, in favor of mappings which are more directly based on more perceptually meaningful properties of animated marks, such as their velocity, orientation, and curvature. In this thesis, I have made extensive use of two particular mappings that are grounded in basic perceptual primitives. One mapping is quite specific: where possible, I have tried to map the left/right spatial position of a mark to its location in the stereo field. The other mapping is more abstract: where possible, I have attempted to match high-

frequency content in the gesture domains, to high-frequency content in the visual and aural domains.

A perceptually plausible mapping between image and sound is of little use to an interactive audiovisual performance system unless it is *generative*—that is, the mapping can be used to create a seemingly infinite variety of audiovisual results. The use of perceptually plausible, generative mappings is central to this thesis, for such mappings make it possible for a system to simultaneously achieve two seemingly contradictory goals: ready apprehensibility, because the mappings can be quickly intuited, and infinite masterability, because the mappings provide an enormous space of possible results.

An important conclusion of this thesis is that successful technologies for creating tight, deeply interactive audiovisual relationships will be best served by the tandem use of synthesized graphics and synthesized sound. The infinite plasticity of a synthetic canvas demands that any sonic counterpart to it be equally malleable and infinite in its possibilities. This can only occur if the system's model of sound generation ultimately affords the expressive control, however abstractly or indirectly, of every single sound sample. To provide any less—by resorting to a model based on the mixing or filtering of canned sound loops, for example—merely creates a toy instrument whose expressive depth is drastically attenuated and explicitly curtailed from the outset. In this thesis, I have settled on a methodology in which I create software synthesizers from scratch, exposing expressive software hooks into their inner mechanisms along the way.

This thesis presents five interactive software systems which implement this painterly interface metaphor for audiovisual performance: *Yellowtail*, *Loom*, *Warbo*, *Aurora*, and *Floo*. They succeed, for the most part, in satisfying the conditions and goals of the thesis: they permit the simultaneous creation of animation and sound in an abstract visual space; they are easy to understand and perform; and they have an effectively unlimited range of expressive results. Naturally, some of these systems are more successful than others: *Aurora*, for example, has a comparatively narrow expressive range, while *Yellowtail* makes use of a diagrammatic mapping between image and sound, and therefore requires a greater degree of explanation or previous familiarity.

In the analysis and evaluation of the thesis instruments, I have identified a number of design patterns which I have found useful or desirable in an audiovisual performance system. Described in Chapter Four, these patterns include gesture capture, gesture augmentation, gestural governance, functionally overloaded gestures, functionally interrelated gestures, high degree-of-freedom gestural inputs, and the potential for alternative graphic representations. At the same time, I have also identified a variety of challenges and pitfalls which plague the design of such systems, both my own and those of other designers: the use of computational randomness, over-mathematization, Cartesian mappings, modal interactions, and ROM-based playback. Finally, I have also identified a number of relative metrics by which audiovisual performance systems may be compared and evaluated; these include their learnability, their predictability, the breadth of their expressive range, their granularity, and the extent to which their possible expressions seem to be determined by the user versus determined *a priori* by the system.

In conclusion, this thesis has presented a new computer interface metaphor for the real-time and simultaneous performance of animation and sound. This metaphor, which developed as an organic reaction to the successes and challenges which comprise the long history of visual music, is based on the idea of an inexhaustible, infinitely variable, dynamic, audiovisual substance that is "painted" into a non-diagrammatic image space. This metaphor is instantiated in five gesture-based interactive software systems whose visual and aural dimensions are deeply plastic, commensurately malleable, and tightly connected by perceptually-motivated mappings. The design principles and challenges which structure these five systems are extracted and discussed, after which the expressive potentials of the five systems are compared and evaluated.

Where this thesis will sit in the unwritten future history of computational color-music instruments remains to be seen. For the time being, the work described here can be plainly identified as a descendant of Fischinger's *Lumigraph*, insofar as it directly uses human gesture to permit the performance of complex images, and Snibbe's *Dynamic Systems Series*, insofar as it makes use of computation to augment these gestures with dynamic, reactive simulations. At the same time, this thesis succeeds, perhaps for the first time, in applying the principles of such systems

to the additional control of sound. By forging a unity between gesture-graphics, graphical-music, and gesture-music, the audio-visual environments described herein perch themselves in a territory foreign to any previous "color-organs" of which I am aware.

### 5.2. Future Directions

The systems presented in support of this thesis are still extraordinarily primitive, particularly when compared with tools like the violin, flute, or paintbrush, which have had the benefit of hundreds of years of refinement, or the human voice, which has evolved over several million years. The digital computer, by comparison, has existed for only about fifty years, and has only had the capacity to produce real-time animation and sound for about the last ten or twelve. It is therefore an optimistic and easy project to enumerate some of the most important and outstanding ways in which computational audiovisual instruments can be improved. Perhaps the most important development will be the use of more sophisticated input devices. Joy Mountford once observed that to design an interaction for a computer's mouse interface is to treat an entire person as if they were a single finger. Future interfaces will not only have more degrees of freedom, but will be ergonomically integrated with the entire body, and will capture expressive nuances from the mouth and its breath, electromagnetic brain waves, eye-tracking, posture, and four-limbed interactions. A second important domain for further research is the use of advanced signal-analysis techniques, which hold the promise of extracting useful information about expressive content from complex input devices or a user's gestural marks. Further analytic "intelligence" applied to the matter may also begin to tease apart the highest-level patterns of a performer's expressions; in this way, a system could eventually develop a sense of history, and adjust itself to its unique user in order to better accompany or support the creation of long-format compositions. Finally, entirely new contexts for audiovisual performance—such as extremely large displays, very small screens, or networked creation spaces involving dozens or thousands of passive or active participants—will change what it means to make expressions in color-music. Given the right cultural climate and a convenient format, such as a keychain computer, color-music creation might even attain a popularity on par with portable video games.

Just as I am not the first person to attempt to create an audiovisual performance system, I am also certain that I will not be the last. It is my honest belief that, as the field is developed, audiovisual performance instruments will come to feel yet more coextensive with ourselves, and allow people to engage and communicate more deeply with themselves and others, through a medium that addresses the heart and mind through the eyes and the ears. It is my sincere hope that this thesis will be of some use to those who continue this long project of color-music performance.

# Appendix A.
# A Timeline of Instruments for Color-Music Performance

| Date | Inventor | Device | Description |
|---|---|---|---|
| 1734 | Louis-Bertrand Castel (1688-1757) | *Clavecin Oculaire* | Father Louis-Bertrand Castel, a Jesuit priest and mathematician, designed and constructed the first known "color organ", his *Clavecin Oculaire* (Ocular Harpsichord). Motivated by both natural philosophy and a spiritual mysticism, Castel augmented a traditional harpsichord with mechanically-exposed colored tapes backlit by a series of candles. [Peacock 1988][Popper 1968]. A more complete description of Castel's device is given in the body of the thesis text. |
| 1789 | Erasmus Darwin (1731-1802) | oil-lamp device | The physician and inventor Erasmus Darwin (a grandfather of Charles Darwin) suggested that visual music could be produced by projecting light from oil lamps through pieces of colored glass [Peacock 1988]. |
| 1791 | Karl von Eckartshausen | colored-liquid clavichord | Karl von Eckartshausen, in his *Disclosures of Magic from Tested Experiences of Occult Philosophic Sciences and Veiled Secrets of Nature,* acknowledged the influence of Castel in his design of a modified clavichord. Eckartshausen wrote that he "had cylindrical glasses, about half an inch in diameter, made of equal size, and filled them with diluted chemical colors. Behind these glasses I placed little lobes of brass, which covered the glasses so that no color could be seen. These lobes were connected by wires with the keyboard of the clavichord, so that the lobe was lifted when a key was struck, rendering the color visible....The clavichord is illuminated from behind by wax candles. The beauty of the colors is indescribable, surpassing the most splendid of jewels. Nor can one express the visual impression awakened by the various color chords" [von Eckartshausen 1791]. |
| 1844 | D. D. Jameson | colored liquid device | Possibly inspired by Darwin or Eckartshausen, Jameson's instrument also filtered light through "glass receptacles containing liquids |

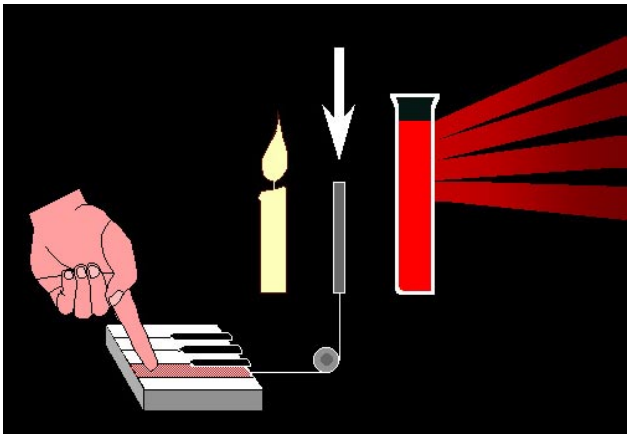


Figure 93. A schematic diagram of von Eckartshausen's colored-liquid clavichord.

of various colors," and projected the resulting colored light onto a wall covered with reflective metal plates. In addition to developing the device, Jameson also published a pamphlet, "Colour-Music," in which he described a system of notation for the new art form [Popper 1968].

1869-73 Frederic Kastner *Pyrophone; Singing Lamp*

Frederic Kastner's 1869 *Pyrophone* produced both sound and image by opening flaming gas jets into crystal tubes. Essentially a gas-activated pipe organ, the device made sounds comparable to "the human voice, the piano, and even the full orchestra." His larger *Singing Lamp*, made in 1874, added an electrical actuation mechanism to the *Pyrophone*, permitting the instrument to be played from a considerable distance away [Popper 1968].

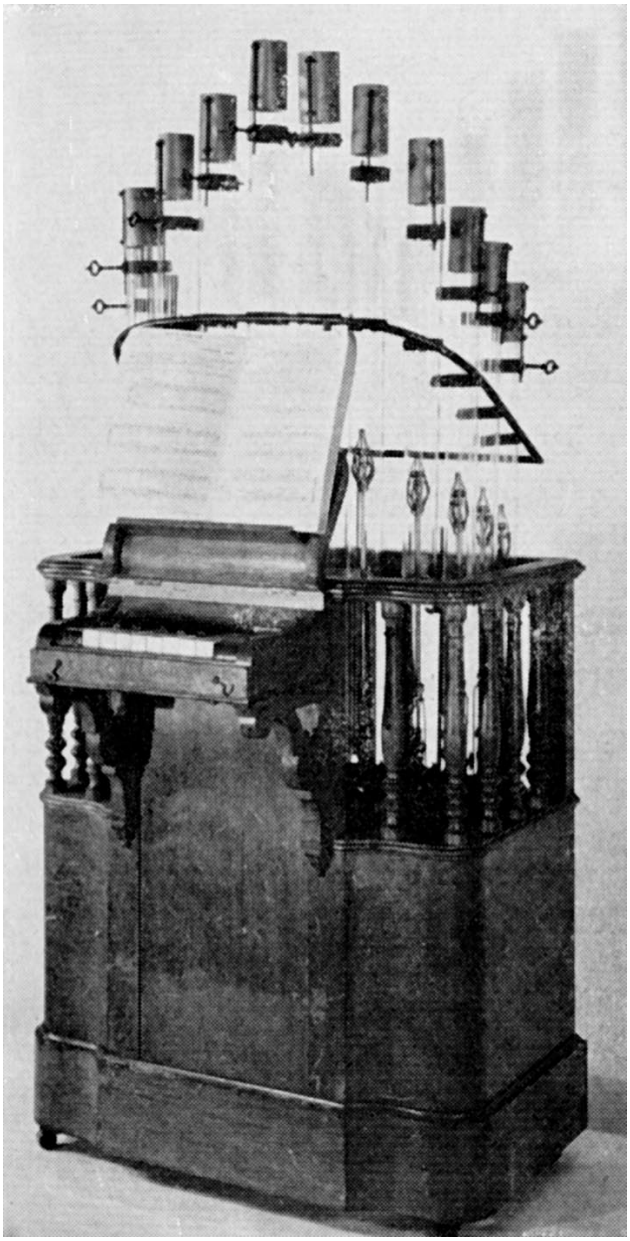
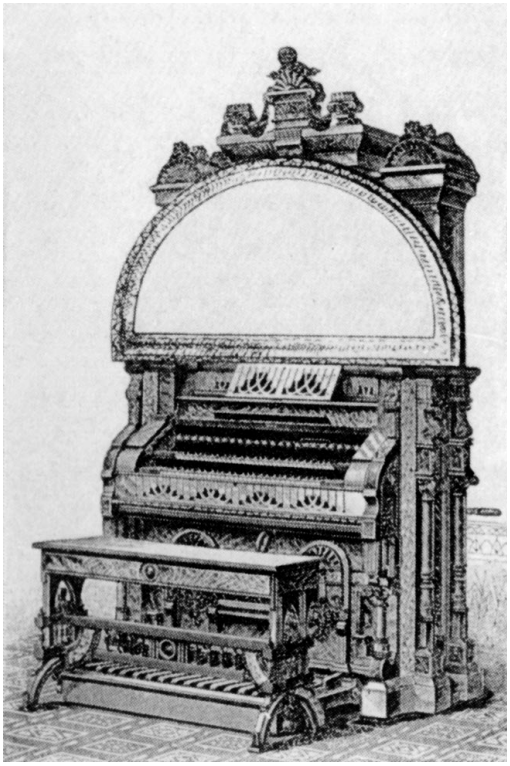


Figure 94. Kastner's *Pyrophone*. From [Popper 1968].

1877 Bainbridge Bishop color organ



Bainbridge Bishop was an American interested in the concept of "painting music." He modified a small pipe organ, such that combinations of colored light were rear-projected onto a small screen when music was performed. His projections used daylight at first, and later an electric arc. [Peacock 1988] [Popper 1968].

Figure 95. Bishop's color organ. From [Popper 1968].

1893 Alexander Wallace Rimington (1854-1918) *Color Organ*

It is to the British painter Alexander Wallace Rimington that we owe the term "color organ", which he patented along with his device in 1893. He later described its mechanism (essentially a bellows organ coupled to an electric colored light projection system) in his 1911 book, *Colour-Music: The Art of Mobile Colour*. Rimington had considerable success in concert halls with his color-music performances of compositions by Wagner, Chopin, Bach and Dvorak [Popper 1968]. In the design of his instruments, according to [Peacock 1988], "Rimington was keenly aware of the 'executant', and wished to create instruments that could be played in performance as well as composed for. Form played little role in Rimington's work; he recognized it as a factor that might be explored, but felt that colour by itself could be satisfying for an immense number and variety of compositions."

| | | | |
|---|---|---|---|
| 1895 | William Schooling | vacuum-tube instrument | William Schooling constructed a device in which the illumination of variously-shaped vacuum tubes was controlled with a keyboard and set of foot-pedals [Peacock 1988]. |
| 1911 | Alexander Scriabin (1872-1915) | *Tastiera per Luce* | For his symphony *Poem of Fire (Prometheus)*, the Russian composer Alexander Scriabin devised an accompaniment of changing colored lights. Scriabin's score called for a keyboard-based color organ he named the *Tastiera per Luce*, which was probably based on the design of Rimington's instrument. Scriabin wanted everyone in the audience to wear white clothes so that the projected colors would be reflected on their bodies and thus possess the whole room [Moritz 1997]. Unfortunately, Scriabin was disappointed by the performances of the *Poem of Fire*, owing to the deplorable state of the light projectors available at the time [Gerstner 1968], [Peacock 1988], [Popper 1968]. |
| 1912 | Alexander Hector | sound-color instrument | Referenced in [Peacock 1988]. |
| 1915 | Modest Altschuler and Preston S. Millar | *Chromola* | Referenced in [Peacock 1988]. |
| 1919 | Thomas Wilfred (1889-1968) | *Clavilux* | Thomas Wilfred completed his first instrument for the production of visual compositions in 1919. He called his instrument the *Clavilux*, and chose the term "lumia" to describe the new art form of silent color-music projections. More information about Wilfred and his instruments is provided in the thesis text [Scattergood-Moore 1998]. |

Figure 96. Images produced by a Wilfred *Home Clavilux*. From [Scattergood-Moore 1998].

1919 Mary Hallock-Greenewalt *Sarabet; Visual-Music Phonograph*

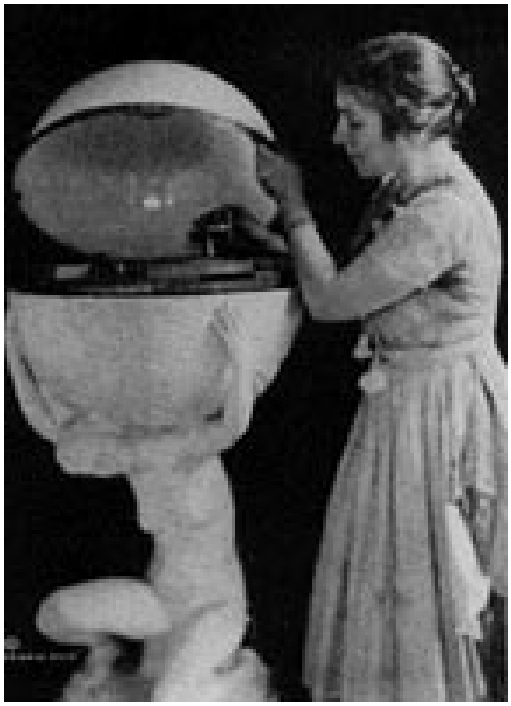


Figure 97. Mary Hallock Greenewalt with her *Visual-Music Phonograph* (1919). Photo by Shewell Ellis.

Mary Hallock Greenewalt was a classically-trained pianist with an illustrious concert career, before her desire to control the ambience in a concert hall led her to experiment with light modulation. Greenewalt was chiefly interested in how variations in luminosity could parallel nuances in music, and so she designed a large color-organ, the *Sarabet*, to put 1500 watts of power at the service of up to 267 shades of color [Popper 1968]. According to historian William Moritz, Greenewalt "invented the rheostat in order to make smooth fade-ups and fade-outs of light, and the liquid-mercury switch, both of which have become standard electric tools. When other people (including Thomas Wilfred) began infringing on her patents by using adaptations of the rheostat and mercury switch, she tried to sue, but a judge ruled that these electric mechanisms were too complex to have been invented by a woman, and denied her case." [Moritz 1997] Although she was unquestionably a victim of foul play, Greenewalt's claims show that she may have also been unfamiliar with her history, for on the general matter of an instrument for color-music, she wrote in her 1946 book, "It is I who conceived it, originated it, exploited it, developed it, and patented it." [Greenewalt 1946]. Greenewalt continued to accompany orchestras on her *Sarabet* for many years, during which time she designed a special notation system that recorded the intensity and deployment of various colors during any given musical composition.

1920 Wladimir Baranoff-Rossiné (1888-1944) *Piano Optophonique*

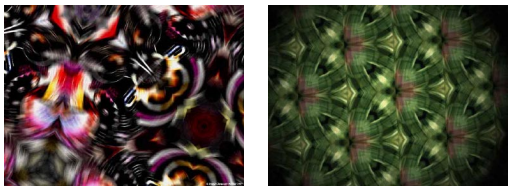


Figure 98. Images produced by the *Piano Optophonique*.

The Ukrainian artist Wladimir Baranoff-Rossiné appears to have been largely overlooked in the English-language literature on color organs. Around 1916, Baranoff-Rossiné began development of his *Piano Optophonique*, a light-projection performance machine. Baranoff-Rossiné's machine operated by passing a bright white light through a variety of keyboard-controlled "luminous filters: simply coloured ones; optical elements such as prisms, lenses or mirrors; filters containing graphic elements and, finally, filters with coloured shapes

Figure 99. The *Piano Optophonique*. From [Baranoff-Rossiné 1997].

and defined outlines." At the heart of the instrument were a series of "Optophonic Disks": circular glass filters covered by translucent paintings, and made to spin inside the projector by the action of electric motors. The machine's resulting projections are lush and kaleidoscopic. Baranoff-Rossiné gave notable Optophonic performances at the Meyerhold (Berlin) and Bolshoi (Moscow) theaters in 1924, and at the Parisian Studio des Ursulines theater in 1928. The *Piano Optophonique* stands out as important because of the interesting relationship it establishes between the freedom afforded by its improvisational controls, and the immutable playback of its pre-recorded optical media. [Baranoff-Rossiné 1997], [Gerstner, 1986], [Popper 1968].

| | | |
|---|---|---|
| 1922 | Ludwig Hirschfeld-Mack | colored shadow shows |

The Weimar Bauhaus of the early 1920's was a hotbed of experimentation in kinetic sculpture. In the summer of 1922, Ludwig Hirschfeld-Mack, Joseph Hartwig and Kurt Schwerdtfeger developed a new mode of expression involving mobile light bulbs of different colors. Hirschfeld-Mack and his group composed a number of scored shadow show performances, such as his *Lichtsonate* and *Farbensonatine*, which were accompanied by piano music.[Peacock 1988], [Popper 1968].

| | | |
|---|---|---|
| 1922 | Kurt Schwerdtfeger | *Reflektorische Farblichtspiele* |

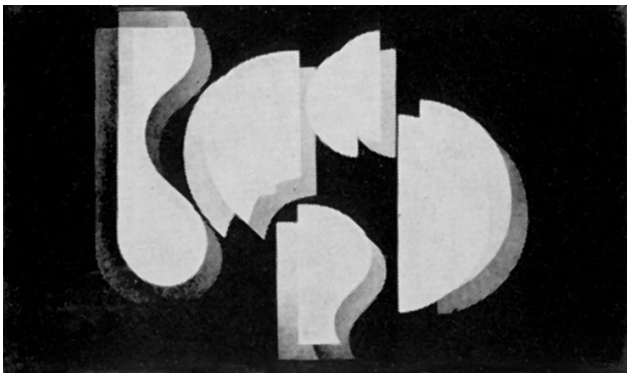


Figure 100. An image produced by Schwerdtfeger's instrument. From [Popper 1968].

Kurt Schwerdtfeger created his *Reflektorische Farblichtspiele* at the Weimar Bauhaus. In this system, "colored rays from mobile light sources shone through forms cut in cardboard, thus producing staggered projections on the screen." Schwerdtfeger evidently adapted his instrument to both abstract and figurative forms. [Popper 1968].

1925 Sandor (Alexander) Laszlo (1895-1970) *Sonchromatoscope*

In 1925, the Hungarian composer Alexander Laszlo wrote a theoretical text on the relationship of color to music, *Die Farblichtmusik*, which became an influential text among many European color-organ designers [Popper 1968]. Laszlo also constructed his own instrument, the *Sonchromatoscope,* which contained switches for colored spotlights and slide projections on the stage above his piano. According to William Moritz, "When the first reviews complained that the visual spectacle was much tamer than the Chopin-like dazzle of Laszlo's virtuoso piano compositions, he contacted Oskar Fischinger to prepare some filmed abstract images of greater complexity and vibrancy. Fischinger prepared a dazzling spectacle with three side-by-side movie projections that were augmented by two more overlapping projectors to add extra colors to the finale, and some complementary changing slide-projections around the borders of the film projection. Much to Laszlo's chagrin, the reviews flip-flopped: the astonishing visual imagery was much livelier and more modern that the old-fashioned Chopin-style piano music." [Moritz 1997] In later decades, Laszlo became a well-known composer of film and TV music, including themes for *Charlie Chan* and *Attack of the Giant Leeches.*

1927 Raoul Hausmann (1886-1971) *Optophone*

Raoul Hausmann, also known for his Dada poetry and writings, and husband for a time of photomonteur Hannah Höch, developed a keyboard-based *Optophone* in the early 1920's. [Popper 1968].

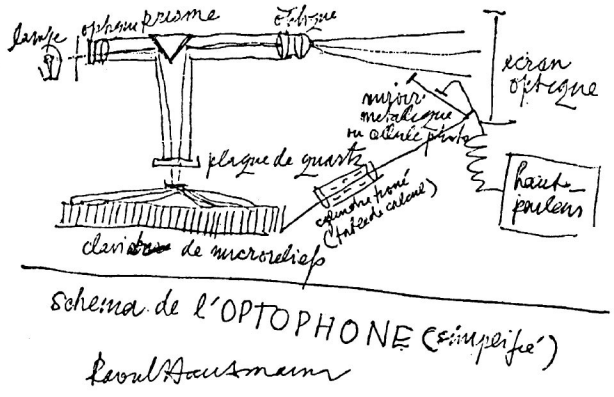



Figure 101. Hausmann's drawings for his *Optophone.*

| | | | |
|---|---|---|---|
| 1928 | Zdenek Pesanek (1896-1965) | colour keyboard | Zdenek Pesanek, a Czech master of glass sculpture, was also interested in light-play and worked on a color keyboard during the 1920's [Popper 1968]. |
| 1930 | Baron Anatol Vietinghoff-Scheel | *Chromatophon* | Referenced in [Moritz 1997]. |
| 1930 | Laszlo Moholy-Nagy (1895-1946) | *Lichtrequisit (Light-Space Modulator)* | Moholy-Nagy's *Light-Space Modulator* is an important work of performable kinetic art, which is still performed regularly at the Harvard art museum where it resides. The sculpture is a twofold installation piece: on the one hand, it could function as a self-contained installation, "to be enclosed in a box with an opening through which spectators could look at the lighting effects within. On the other hand, it functioned as an interactive, site-modifying work, transforming its surroundings with revolving rays of projected and filtered light." [Malina 1974]. |
| 1930s | George Hall | *Musichrome* | According to [Peacock 1988], Hall created a *Musichrome* device which used eight keys to control two sets of four colors each. |
| 1931 | Morgan Russell (1886-1953) and Stanton Macdonald-Wright (1890-1972) | *Kinetic Light Machine* | Morgan Russell and Stanton Macdonald-Wright, abstract painters, experimented with their *Kinetic Light Machine* as a way of animating "synchromies," their own term for color harmonies [Kushner 1990]. |

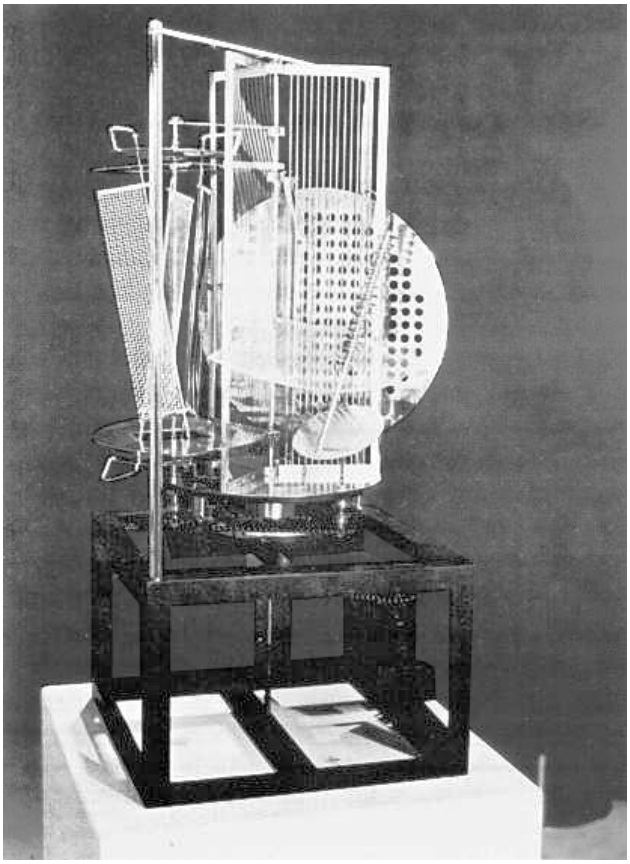


Figure 102. Moholy-Nagy's *Light-Space Modulator.*

| | | | |
|---|---|---|---|
| 1933 | Adrien-Bernard Klein | color projector | Klein's book *Colour-Music: The Art of Light*, written in 1927, is a classic text on the subject [Klein 1927]. Klein felt strongly that the frequencies of colored light "provide a scale of wave lengths whose ratios may be compared directly to the twelve equally spaced intervals of the chromatic scale." According to [Popper 1968], Klein put his theories into practice a few years later, with the design of a "colour projector for the theater...Klein gave showings of his projections in 1993 at the Astoria Cinema, London." |
| 1940's | Cecil Stokes (1910-1956) | *Auroratone* films | Stokes designed special time-lapse devices to capture the color effects of crystal formations as they grew under polarized light. He accompanied his films with "slow sedative and mildly sad music," and hoped that his films would help mentally ill souls "ventilate their pent-up tensions resulting from conflicts and frustrations" [Collopy 1999]. |
| 1940's | Charles Dockum | *MobilColor Projector* | Charles Dockum began to build color-organs in the late 1930s. His *MobilColor Projector* was capable of producing both hard-edged or soft imagery, since it used prepared image sources; the movements and colors of these elements, moreover, could be "programmed" in advance to produce complex temporal patterns [Russet & Starr 1988]. A later, larger version of the *MobilColor* permitted multi-layered motion in several directions, but met with practical difficulties because its use demanded two operators [Moritz 1997]. More information on Dockum's work is included in Chapter Two. |
| 1950 | Oskar Fischinger (1900-1967) | *Lumigraph* | Oskar Fischinger's career in abstract moving images spanned several decades. Although he was chiefly known for his animation, Fischinger's ingeniously simple *Lumigraph* stands as a monumental statement about the degree of subtlety and expressivity attainable with a visual performance device. A more complete description can be found in Chapter Two of the thesis text. |

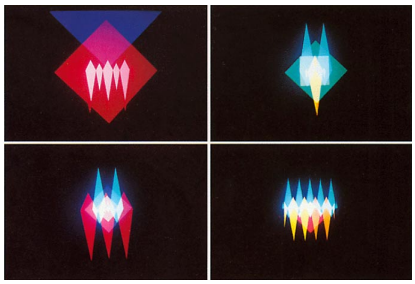


Figure 103. An image produced by Dockum's *MobilColor Projector.*

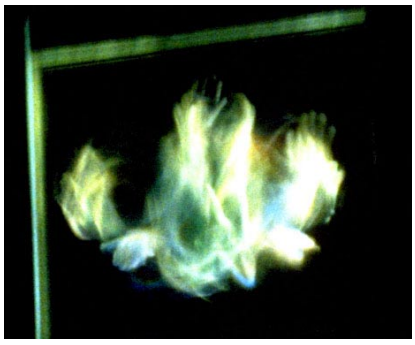


Figure 104. An image produced by Fischinger's *Lumigraph.*

| | | | |
|---|---|---|---|
| 1953 | Gordon Pask and McKinnon Wood | *Musicolour Machine* | Gordon Pask was a cybernetician concerned with the psychology of aesthetic pleasure. Between 1953 and 1957, Pask and his collaborator McKinnon Wood designed a *Musicolour Machine* that used musical input to create visual output. By incorporating simple "learning" strategies (automatic parameter adjustment) into their machine, Pask and Wood were able to investigate human performers, and audiences, in cybernetic feedback loops [Pask]. |
| 1956 | Nicholas Schöffer (1912-) | *Musiscope* | Schöffer's *Musiscope* was an instrument for performing visual music, operated by an electronic keyboard. Schöffer described his machine as having "a complete range of buttons...[which] enable the performer to obtain on the screen -- simultaneously or successively -- a considerable number of families of images, colours and light effects, and to combine them or vary their degree of definition or intensity. Besides this, there is a rheostat which works upon the respective speeds of rotation of the coloured filters and the sculptural element, allowing the enactment of the images in time to accelerate, slow down, or stop completely." [Popper 1968]. |
| 1957-61 | Jordan Belson (1926-) and Henry Jacobs | *Vortex Concerts* | According to William Moritz, "the composer Henry Jacobs convinced the Morrison Planetarium in San Francisco to let him use their newly renovated four-track surround-sound system for a series of concerts. Jacobs commissioned new pieces of electronic music from international composers and asked Jordan Belson to prepare visual imagery that could be projected on the dome during each number. Having access to the planetarium "starscape" projectors, as well as conventional film and slide projectors, opened for Belson the possibility of rivaling Thomas Wilfred's light projections, which had impressed him years earlier in New York." [Moritz 1996]. |

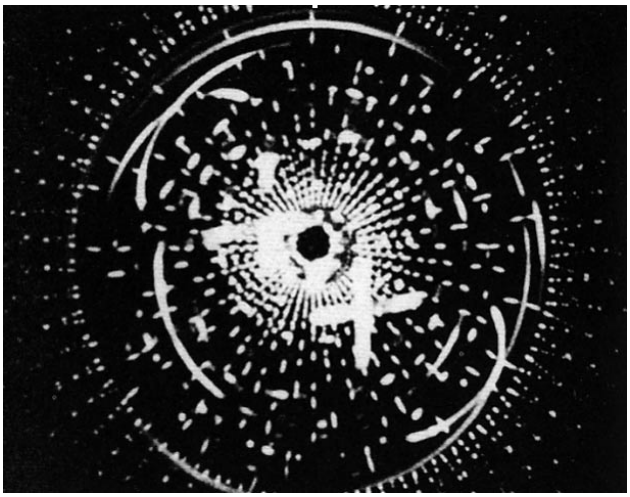


Figure 105. A still from one of Belson's *Vortex Concerts.*

| | | | |
|---|---|---|---|
| 1960's | Lloyd G. Cross | *Sonovision* | The holographer Lloyd Cross was quick to see the visual-music potential of the newly-invented laser in the late 1960's. His *Sonovision* system for the visual display of sound directly coupled the mechanical deflection of a laser beam to audio pressure variations. In this device, a thin flexible membrane was stretched over the front of an ordinary audio speaker cone; a small Mylar mirror was then cemented in the center of the membrane. When a laser beam was reflected off the mirror, the result was a complex set of highly responsive Lissajous-type figures, in sync with the music, cast on the walls and ceiling. [Malina 1974][Fairstein 1997]. |
| 1963 | Frank Malina (1912-1981) | *Lumidyne, Reflectodyne* | Frank Malina developed a number of systems in the 1960's to explore dynamic lightforms. The *Lumidyne* and *Reflectodyne* were manipulable kinetic sculptures which modulated light with a set of rotating painted acrylic disks. Frank Malina was also notable as an historian of kinetic art and as the founder of the *Leonardo* journal of arts and sciences. [Popper 1968],[Malina 1974]. |
| 1966 | Richard I. Land | *Chromara* | Land's easy-to-reproduce *Chromara* color organ became one of the "standard" designs popularized in 'Sixties psychedelic culture. His device applied a bank of differently-tuned passband filters to incoming audio, and used the responses of the filters to determine the intensity of a corresponding set of colored lights. Land's writings further proposed the introduction of manual controls and rhythm-sensing circuits. [Malina 1974]. |
| 1974-79 | Laurie Spiegel | *VAMPIRE* | Laurie Spiegel at Bell Labs created *VAMPIRE* (Video and Music Program for Interactive Realtime Exploration), a performance system for the simultaneous synthesis of video images and electronic sound. Spiegel's system built on the *GROOVE* computer music system created by Max Matthews, by combining it with paint-program-like graphic algorithms by Kenneth Knowlton. The *VAMPIRE* system offered real-time, gestural control of both sound and video image, through a variety of continuous input devices [Spiegel 1998]. |

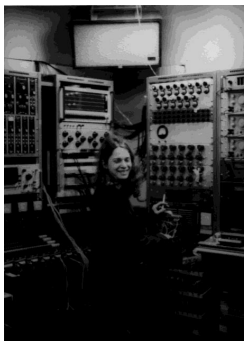
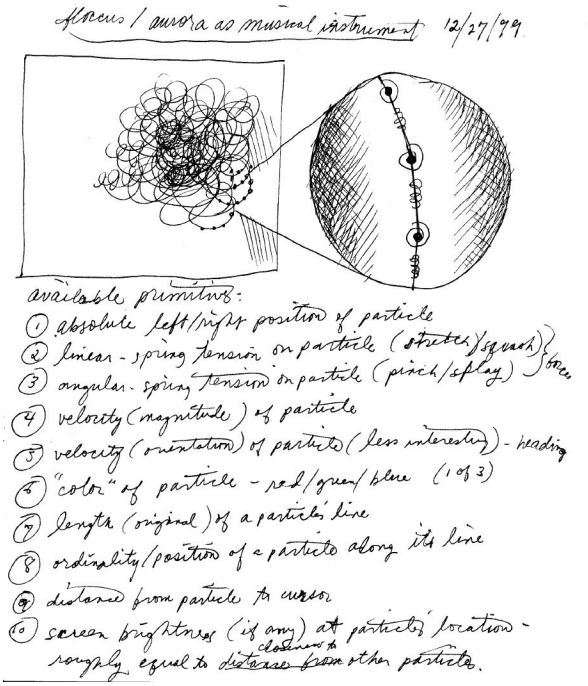
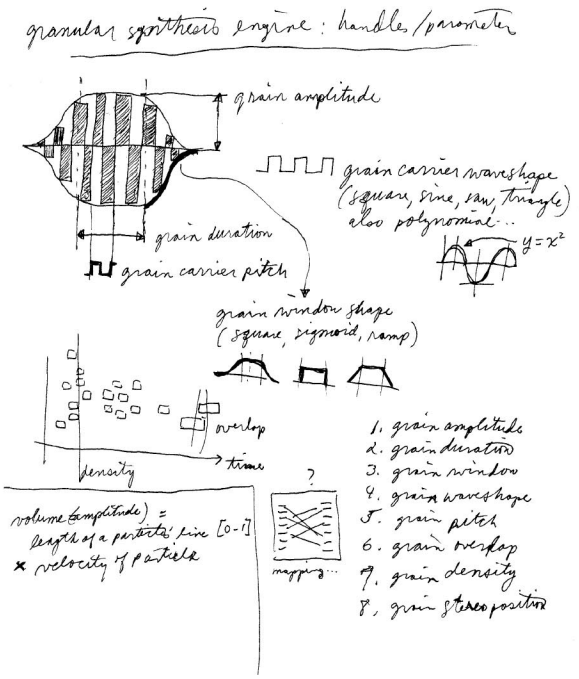


Figure 106. Laurie Spiegel with the *VAMPIRE* system.

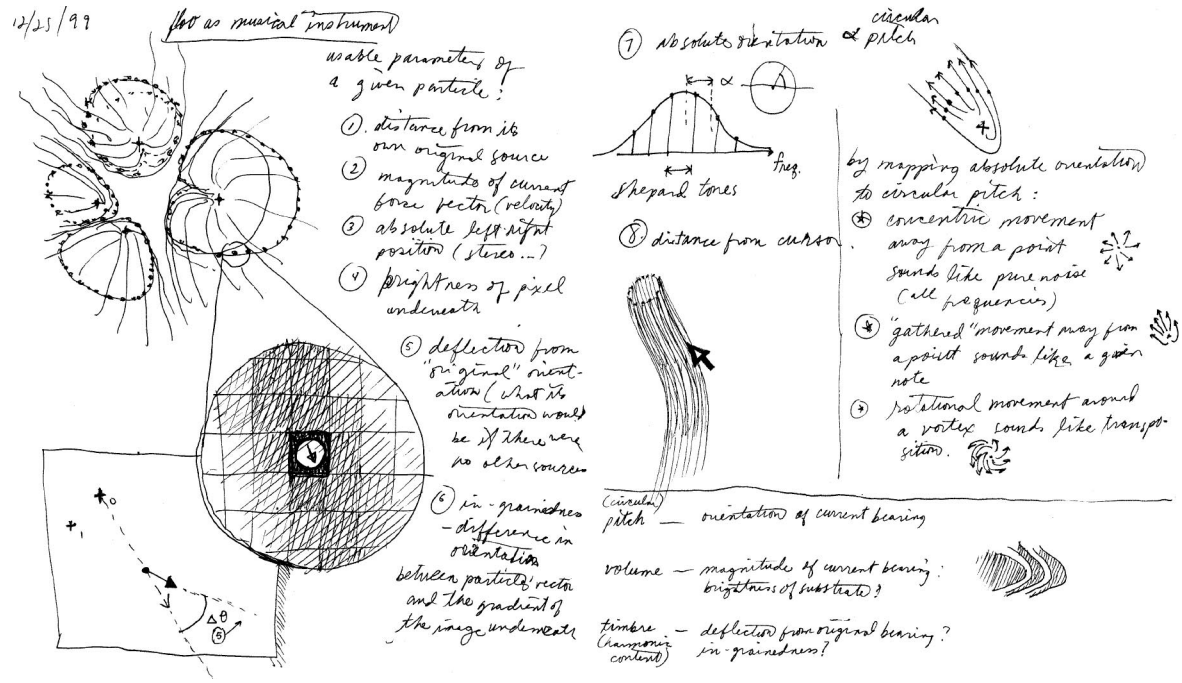
# Appendix B. Supplementary Sketches




▲ Figure 107 (above). A preliminary sketch for the *Aurora* synthesizer.

▼ Figure 108 (below). A preliminary sketch for the *Floo* synthesizer.



# Appendix C. Application Pseudocodes

In this appendix, I provide several implementation details, written in a C++-like pseudocode, which explicate some of the internal mechanisms of the software applications which support this thesis. Many of the software experiments described in Chapter 3 were originally developed in more than one programming environment; for example, the silent applications created at Interval Research Corporation were sketched in Macromedia Director and then rendered in C++ using Microsoft *DirectDraw* libraries, while the more recent audiovisual applications developed at the Media Lab were, more often than not, sketched as Java applets and then rendered in C++ using the *OpenGL* graphics libraries. For those applications in which sound was directly synthesized, I used the *DMedia* sound libraries (on the SGI) and the *MMmedia* and *DirectSound* audio libraries (on Windows NT/2000 computers).

All five of the audiovisual applications which support this thesis—*Yellowtail*, *Loom*, *Warbo*, *Aurora*, and *Floo*—make important use of a multithreaded operating system. Specifically, each application has two threads: a graphics/interaction thread, which is clocked at a visual refresh rate of approximately 30Hz, and an audio-handling thread, which is clocked at a higher rate, generally around 200Hz. This audio thread is awakened whenever the sound card's output buffer drains below some critical threshold; at that point, a method queries the state of the graphics engine for any relevant features, and computes new audio samples which take those features into account.

Optimization efforts inside of the innermost sound synthesis loops yield substantial rewards in performance improvements. I assume the existence of certain helper functions, such as `determineIfSoundBufferNeedsFilling()`, which will be provided as part of the system's sound API and will therefore be platform-dependent.

## C.1. Additive Synthesis in *Yellowtail*

```
int     nsam = numberOfSamplesInOutputBuffer;
bool    soundThreadActive; // whether we're making sound
float   sampleRate;        // samples per second, e.g. 44100
float   curTime;           // current time, in seconds

int     nosc = number of additive synthesis oscillators
float   noct = number of octaves of oscillators
float   lowf = lowest oscillator frequency
float   base = noct/nosc;

do {
    if (determineIfSoundBufferNeedsFilling()){
        for (int s=0; s<nsam; s++){
            float output = 0;
            for (int i=0; i<nosc; i++){
                float amp = brightness of pixel[i]; // 0...1
                float freq = lowf * pow (base, i);
                output += amp * sin(twoPi * freq * curTime);
            }
            soundOutputBuffer[s] = output;
            curTime += (1.0/sampleRate);
        }
    }
} while (soundThreadActive);
```

## C.2. FM Synthesis in *Loom*

```
int     nsam = numberOfSamplesInOutputBuffer;
bool    soundThreadActive; // whether we're making sound
float   sampleRate;        // samples per second, e.g. 44100
float   curTime;           // current time, in seconds

int     nstr = numberOfActiveLoomStrokes;
float   val;

// each Loom stroke has a carrier frequency and a
// modulator frequency associated with it by the user

do {
    if (determineIfSoundBufferNeedsFilling()){
        for (int s=0; s<nsam; s++){
            float output = 0;
            for (int i=0; i<nstr; i++){
                float I = stroke[i]->getCurrentCurvature();
                float A = stroke[i]->getCurrentPressure();
                float C = stroke[i]->carrierFrequency;
                float M = stroke[i]->modulatorFrequency;
                // the FM equation is A*sin(Ct + I*sin(Mx))
                val = A* sin ( C*twoPi*curTime +
                               I*sin(M*twoPi*curTime));
                output += val;
            }
            soundOutputBuffer[s] = output;
            curTime += (1.0/sampleRate);
        }
    }
} while (soundThreadActive);
```

## C.3. Granular Synthesis in *Aurora* and *Floo*

```
int     nsam = numberOfSamplesInOutputBuffer;
bool    soundThreadActive; // whether we're making sound
float   sampleRate;        // samples per second, e.g. 44100
float   curTime;           // current time, in seconds

int     ngrn = numberOfActiveGrains;

do {
    // update the status of each grains
    for (int g=0; g<ngrn; g++){
        bool grainFinished = ((curTime - grain[g]->startTime) >
                              grain[g]->duration);
        if (grainFinished) {
            grain[g]->setDurationBasedOnGraphics();
            grain[g]->setFrequencyBasedOnGraphics();
            grain[g]->setPanBasedOnGraphics();
            grain[g]->setStartTime(curTime);
        }
    }

    // generate audio samples from grains
    if (determineIfSoundBufferNeedsFilling()){
        for (int s=0; s<nsam; s++){
            float output = 0;
            for (int g=0; g<ngrn; g++){
                float envelopeLoc = (curTime - grain[g]->startTime)
                                    /grain[g]->duration;
                float amp = hanningWindow (envelopeLoc);
                float freq = grain[g]->frequency;
                float grainOutput = amp * sin(twoPi*freq*curTime);
                output += grainOutput;
            }
            soundOutputBuffer[s] = output;
            curTime += (1.0/sampleRate);
        }
    }
} while (soundThreadActive);
```

## C.4. Chebyshev Waveshaping Synthesis in *Warbo*

```
int     nsam = numberOfSamplesInOutputBuffer;
bool    soundThreadActive; // whether we're making sound
float   sampleRate;        // samples per second, e.g. 44100
float   curTime;           // current time, in seconds

int     nspt = numberOfActiveWarboSpots;
int     nseg = numberOfStreamerSegments;
```

```
do {
    if (determineIfSoundBufferNeedsFilling()){
        for (int s=0; s<nsam; s++){
            float output = 0;
            for (int i=0; i<nspt; i++){
                float F = spotArray[i]->getPitch();
                float A = spotArray[i]->getCurrentAmplitude();
                float val = sin (F*twoPi*curTime);
                for (int j=1; j<nseg; j++){
                    float amp = streamerSegs[j]->getCurvature();
                    // amplitudes are scaled 0...1
                    val = Chebyshev(j, amp*val);
                    // the j'th Chebyshev waveshaping function,
                    // see Figure 68
                }
                output += A * val;
            }
            soundOutputBuffer[s] = output;
            curTime += (1.0/sampleRate);
        }
    }
} while (soundThreadActive);
```

## C.5. A Simple Spring

```
bool    simulationIsActive;
float   initialPosition;
float   position = initialPosition;
float   velocity = initialVelocity;
float   damping = 0.975; // for example. must be < 1.0
float   mass = 1.0; // particle mass
float   K = 1.0; // spring constant

do {
    // use Euler integration to move a springy particle.
    float distension = position - initialPosition;
    float force = K*distension; // Hooke's Law: f=-kx
    float acceleration = force/mass;

    velocity += acceleration; // integrate once
    velocity *= damping;      // apply friction
    position += velocity;     // integrate again

    object->render(position);

} while (simulationIsActive);
```

# Bibliography


Adriano Abbado. *Perceptual Correspondences of Abstract Animation and Synthetic Sound*. M.S. Thesis, MIT Media Laboratory, June 1988.

Jacques Bertin, *Semiology of Graphics: Diagrams, Networks, Maps*. University of Wisconsin Press, 1983.

Ronald M. Baecker, William Buxton, and Jonathan Grudin (Editors), *Readings in Human-Computer Interaction : Toward the Year 2000*. Morgan Kaufmann, 1995.

Dmitri Baranoff-Rossiné, *The Piano Optophonique of Wladimir Baranoff-Rossiné*, <http://perso.club-internet.fr/dbr/piano.htm>, 1997.

William S. Cleveland, *Visualizing Data*. Hobart Press, 1993.

Fred Collopy, *Imagers and Lumia*. <http://imagers.cwru.edu/index.html>, 1999.

Perry R. Cook, *Music, Cognition, and Computerized Sound: An Introduction to Psychoacoustics*. MIT Press, 1999.

Frank. S. Cooper, "Some Instrumental Aids to Research on Speech," *Report on the Fourth Annual Round Table Meeting on Linguistics and Language Teaching*, Georgetown University Press, pp. 46-53, 1953.

The Corcoran Gallery of Art. *Thomas Wilfred: A Retrospective*. Exhibition catalog, Washington D.C., 1972.

Mihaly Csikszentmihalyi, *Creativity : Flow and the Psychology of Discovery and Invention*. Harper Collins, 1997.

Mihaly Csikszentmihalyi, *Flow : The Psychology of Optimal Experience*. Harper Collins, 1991.

Marc Davis, *Media Streams: Representing Video for Retrieval and Repurposing*. Ph. D. Thesis, MIT Media Laboratory, March 1995.

Tom DeWitt, "Visual music: Searching for an aesthetic," *Leonardo*, 20 (1987), 115-122.

Sarah Douglas and Ted Kirkpatrick, "Do Color Models Really Make a Difference?" *Proceedings of CHI'96*, 1996.

Karl von Eckartshausen, *Disclosures of Magic from Tested Experiences of Occult Philosophic Sciences and Veiled Secrets of Nature*. In Sergei Eisenstein, *The Film Sense*, 1942.

Sergei Eisenstein. *The Film Sense*. New York: Harcourt, Brace & Jovanovitch, 1942.

John Fairstein, *The San Francisco School of Holography*. <http://user.icx.net/~jfairstein/SOH.html>, 1997.

Elfriede Fischinger, *Writing Light*. <http://www.iotacenter.org/Elfriede/WritingLight.htm>, 1998.

Oskar Fischinger, *Device for Producing Light Effects*. United States Patent #2,707,103. United States Patent Office, 1955.

Karl Gerstner, *The Forms of Color: The Interaction of Visual Elements*, Cambridge, MA: MIT Press, 1986.

Lukas Girling, *Granulator* and *Vector Field*. Software prototypes developed at Interval Research Corporation, Palo Alto, 1997-1998.

Mary Hallock Greenewalt, *Nourathar: The Fine Art of Light Color Playing*. Philadelphia: Westbrook, 1946.

Paul Haeberli, *DynaDraw*. Silicon Graphics Corporation, Mountain View, California, 1989. <http://www.sgi.com/grafica/dyna/index.html>.

Interval Research Corporation, Palo Alto. *Soundscapes* musical software instruments. S.Joy Mountford, project director. <http://web.interval.com/projects/soundscapes/>, 1995.

Toshio Iwai, *Music Insects*. Interactive exhibit at the San Francisco Exploratorium. Toshio Iwai Works web site, <http://www.iamas.ac.jp/~iwai/artworks/music_insects.html>, 1992.

Toshio Iwai, *SimTunes*. Software released by Maxis Software. Toshio Iwai Works web site, <http://www.iamas.ac.jp/~iwai/simtunes/index.html>, 1996.

Roger Johnson, *Scores: An Anthology of New Music*. Schirmer Books, 1978.

M. Kass, A. Witkin, and D. Terzopolous. "Snakes: Active Contour Models," *Proc. International Conference of Computer Vision*, pp. 259–268, London, England, 1987.

Adrien Bernard Klein, *Colour-Music: The Art of Light*. Crosby, Lockwood & Son, London, 1927.

Koblo Software Corporation, Denmark. *Vibra6000* software synthesizer. <http://www.koblo.com>, 1999.

Reed Kram, *System Models for Digital Performance*. M.S. Thesis, MIT Media Laboratory, June 1998.

Myron W. Krueger, *Artificial Reality II*. Reading, Mass: Addison-Wesley, 1991.

Marilyn S. Kushner, *Morgan Russell*. New York: Hudson Hills Press, 1990.

Golan Levin and Paul Debevec, *Rouen Revisited*. Interactive installation presented at the Bridge: SIGGRAPH '96 Art Show. New Orleans, August 1996. <http://www.media.mit.edu/~golan/rouen/index.html>.

Golan Levin and Paul Yarin. "Bringing Sketching Tools to Keychain Computers with an Acceleration-Based Interface." *Proceedings of ACM SIGCHI '99,* May 1999. <http://www.media.mit.edu/~golan/shakepad/index.html>.

Karon Maclean, Scott Snibbe and Golan Levin. "Tagged Handles: Merging Discrete and Continuous Control." *Proceedings of ACM SIGCHI 2000,* April 2000. <http://www.media.mit.edu/~golan/handles/index.html>.

Jock Mackinlay, "Automating the Design of Graphical Presentations of Relational Information." *ACM Transactions on Graphics,* Vol. 5, No. 2, April 1986, pp 110-141.

John Maeda, *Digital Expression: A Framework for Artistic Creation of Forms On and Using the Computer,* Ph.D. Thesis, University of Tsukuba, Japan, 1995.

John Maeda, *Design by Numbers*. Cambridge, Massachusetts: MIT Press, 1999.

Frank J. Malina, *Kinetic Art: Theory and Practice,* Dover Publications, New York, 1974.

Mark of the Unicorn Software, *Performer 6.0* sequencing software, 2000. <http://www.motu.com/>.

Barton McLean, "Composition with Sound and Light," *Leonardo Music Journal,* 2 (1992), 13-18.

F. Richard Moore, *Elements of Computer Music*. Englewood Cliffs, New Jersey: Prentice-Hall, 1990.

William Moritz, "The Dream of Color Music," in *The Spiritual in Art: Abstract Painting 1890-1985,* ed. Maurice Tuchman. Abbeville, 1993.

William Moritz, "A Survey of Experimental Animation."
In the *Absolut Online Experimental Animation Festival,* <http://
panushka.absolutvodka.com/panushka/map/koji/index.html>, 1996.

William Moritz, "The Dream of Color Music, And Machines That
Made it Possible", *Animation World Magazine,* Issue 2.1, April 1997.
<http://www.awn.com/mag/issue2.1/articles/moritz2.1.html>.

Gordon Pask, "A comment, a case history and a plan," in *Cybernetics, Art and Ideas,* ed. Jasia Reichardt. London: Studio Vista, 1971.

Kenneth Peacock, "Instruments to perform color-music: Two centuries of technological experimentation," *Leonardo,* 21 (1988), 397-406.

Frank Popper, *Art of the Electronic Age*. Thames and Hudson, 1993.

Frank Popper, *Origins and Development of Kinetic Art.* New York Graphic Society, 1968, pp. 156-172.

Patric Prince and Robert Holzman, "John Whitney: The Problem: How Shall Motion Pattern Time?", *SIGGRAPH Stuff* (ACM SIGGRAPH Education Committee Newsletter), Vol 1, Issue 7, Fall 1996.
<http://www.education.siggraph.org/stuff/spacef95/john.html>.

Perpetual Music, Inc. *Sounder*. Interactive music software for Windows.
<http://www.sounder.com/>, 1994.

Propellerhead Software Inc., Sweden. *ReBirth RB-338* software synthesizer, Sweden. <http://www.propellerheads.se>, 1999.

Peter W. Rice, *Stretchable Music: A Graphically Rich, Interactive Composition System*. M.S. Thesis, MIT Media Laboratory, September 1998.

Alexander Wallace Rimington, *Colour-Music: The Art of Mobile Colour,* New York: Frederick A. Stokes Company, 1911.

Don Ritter, "Interactive Video as a Way of Life," *Musicworks 56,* Fall 1993, 48-54.

Curtis Roads, editor. *The Music Machine: Selected Readings from the Computer Music Journal*. Cambridge, MA: MIT Press, 1989.

Curtis Roads, *The Computer Music Tutorial*. Cambridge, MA: MIT Press, 1996.

Robert Russett and Cecile Starr, *Experimental Animation: Origins of a New Art* (2nd edition), New York: Da Capo Press, 1988.

Gene Arthur Scattergood-Moore, *Thomas Wilfred (Richard Edgar Løvstrøm)*. <http://www.gis.nest/~scatt/wilfred.html>, 1998.

Malcolm Slaney, *Pattern Playback from 1950 to 1995*. Proceedings of the 1995 IEEE Systems, Man and Cybernetics Conference, October 22-25, 1995, Vancouver, Canada. <http://web.interval.com/papers/1994-036/>.

Wayne Slawson, *Sound Color*. University of California Press, Berkeley, 1985.

Scott Snibbe and Golan Levin, *Towards Dynamic Abstraction*. Interval Research Corporation, Palo Alto. Internal document, October 1997.

Scott Snibbe and Golan Levin, "Interactive Dynamic Abstraction." *Proceedings of the Symposium on Nonphotorealistic Animation and Rendering,* Annecy, France, June 2000.

Scott Snibbe, "The Motion Phone." *Proceedings of Ars Electronica '96*. Christine Schopf, ed. <http://kultur.aec.at/lab/futureweb/english/prix/prix/1996/E96azI-motion.html>.

Scott Snibbe, personal web site, 1998-2000. <http://www.snibbe.com>.

Laurie Spiegel, *Graphical Groove: Memorium for a Visual Music System*. <http://retiary.org/ls/btl/vampire.html>, August 1998.

John L. Walters. "Sound, Code, Image." *Eye* 26, 1997, pp.24-35.

John Whitney, *Digital harmony: On the Complementarity of Music and Visual Art,* Peterborough, NH: McGraw-Hill, 1980.

Thomas Wilfred, "Composing in the art of lumia," *Journal of Aesthetics and Art Criticism,* (VII) December 1948, 79-93.

Thomas Wilfred, "Light and the artist," *Journal of Aesthetics and Art Criticism,* (V) June 1947, 247-255.

Gene Youngblood, *Expanded Cinema,* New York: E.P. Dutton, 1970.

**Colophon**

This document was prepared with Adobe *InDesign* and Macromedia *Dreamweaver*. The text of this thesis was set in eleven-point *Scala* (1990) and *Scala Sans* (1993), both designed by Martin Majoor. The chapter titles were set in *Interstate* (1994), designed by Tobias Frere-Jones.