Stretchable Music: A Graphically Rich, Interactive Composition System

by

Peter W. Rice Jr.

S.B. Brain and Cognitive Sciences
Massachusetts Institute of Technology
1996

Submitted to the Program of 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 1998

Signature of Author____
Program in Media Arts and Sciences
August 7, 1998

Certified by____
Tod Machover
Associate Professor of Music and Media
Program in Media Arts and Sciences
Thesis Supervisor

Accepted by____
Stephen A. Benton
Chair
Departmental Committee on Graduate Students

Program in Media Arts and Sciences

Stretchable Music: A Graphically Rich, Interactive Composition System

by

Peter W. Rice Jr.

Submitted to the Program of Media Arts and Sciences,
School of Architecture and Planning
on August 7, 1996
in partial fulfillment of the requirements for the degree of

Master of Science in Media Arts and Sciences

at the

Massachusetts Institute of Technology

Abstract

Since the advent of modern recording technology made high quality music inexpensive and widely available, people have had less incentive to learn to play musical instruments themselves. Interactive music programs written for today's powerful personal computers represent a promising way to re-engage passive listeners in a more active musical experience. This thesis advocates a design philosophy where these systems are not thought of as tools to enable a musically deficient public, but as the expression of musical ideas as compositions in an interactive form unique to the medium of computation. To test this approach, the thesis describes the design and development of a particular system, called Stretchable Music, that combines graphics, sound, and complex, interdependent controls in a new, particularly effective way. The system invites users to shape musical layers by pulling and stretching animated graphical objects with a mouse; the objects themselves have been carefully tailored to abstractly represent the music. The thesis highlights the many design and implementation issues that were encountered during the development process of the project. Finally, it offers a broader view about how these content-based interactive music systems may fit into the culture of the next century where the dominant mode of musical experience may no longer be a spectator sport.

Stretchable Music: A Graphically Rich, Interactive Composition System

by

Peter W. Rice Jr.

Thesis Readers:

Reader____

Bruce Blumberg
Assistant Professor of Media Arts and Sciences
Program in Media Arts and Sciences

Reader____

John Maeda
Interval Assistant Professor of Design and Computation
Program in Media Arts and Sciences

Acknowledgments

First and foremost I would like to thank my advisor *Tod Machover* for encouraging me to explore a new, artistic path, and for providing patient and unwavering support through this project, and indeed duing my almost 6 years working with him at MIT.

Additional thanks go to…

My readers *Bruce Blumberg* and *John Maeda* for their many insightful comments on this document.

John Maeda for opening my mind to a new medium of expression in computation.

My *Mother* and *Father* for always standing behind me and encouraging me to strive for the very best.

Ben Denckla for editing this document, many hours of interesting discussions about artistic expression through technology, summer runs, and many hours of evening bowling.

Chad Brustin for being more on top of things than is humanly possible and patiently tolerating those who are not.

The members of the *Opera of the Future Group*:
Seum-Lin GAN for being a great office mate and providing hours of humorous entertainment
Maggie Orth for helping encourage me in me artistic pursuits and telling me how it is.
Gili Weinberg for interesting discussions on the design of interactive instruments and algorithmic music.
Ed Hammond for being a great house-mate, many late night lessons in history, and ceaselessly hazing me.
Teresa Marrin for her sharing her interesting perspectives on contemporary classical and 20th century music.

As well as *UROPs:*
Anthony Johnson for working on additional code for the Stretchables system.
Noah Schotenfeld for doing some audio work and for many interesting discussions about electronic music.

Joe Paradiso and *Josh Strickon* for working with me to combine Stretchable Music with their Laser Rangefinder interface.

Members of the *Brain Opera Team* for taking new forms of interactive electronic music to all corners of the world and having a great time doing it!

Alex Westner , co-organizer of *Cubestock* and to everyone else who helped out. It was a blast!

Fellow Members of EVFS, *Chuck Kemp* and *Dan Paluska*, for working with me to explore a new era of electronic music performance.

My *friends* for patiently putting up with me while I struggled to complete this project and this document.

My *family* for patiently enduring my absence while I do all these projects, but for supporting me all the way.

And finally, to “The *Bud* that Makes You *Wiser*” for getting me through many difficult evenings of writer’s block.

Table of Contents

Lists of Tables and Figures

List of Figures

1 Introduction

Why do musical instruments have to be hard to play? Why does it take ten years of piano lessons to qualify one to participate in the active process of making music? Why are there essentially only two kinds of musical experiences available: actively playing music or passively listening to it? Is there room for more? Can technology help bridge this gap between the musical “haves” and “have-nots,” and if so, how? These are the core issues this thesis will address through the introduction and analysis of a novel interactive music system. It will argue that the system represents a new level of musical expression on a computer for both the composer and the musically untrained public. It is the author’s hope that the system and this document will become stepping stones toward the design of new types of technologically-facilitated musical experiences that actively engage a much broader range of participants in the process of making music than is currently possible.

1.1 The Active and the Passive Musical Experience

Acoustic instruments demand considerable manual dexterity to play, mainly due to constraints imposed by their physical, sound producing mechanisms. Traditionally, however, there were few other ways to experience music: either you played an instrument or you listened (and perhaps sang along). The advent of recording technology at the beginning of the 20th century eased this situation somewhat. At least then, to listen you didn’t have to be in the same room as the person playing the instrument. On the other hand, with music even more accessible to the passive listener, there was less incentive to acquire the skill required to play an instrument. All you had to do was press “PLAY.” So common has the passive musical experience become due to recording technology, that listening to music live is considered a special occasion by most people. Despite making many more forms of music accessible to more people than ever before possible, recording technology can arguably be accused of making the musical experience more static, with many more passive listeners listening to musical recordings that come out exactly as they were recorded every time they are played.

All of this should not be taken to suggest that people don’t actually want to have more active musical experiences. There are many indications that indeed point in the opposite direction. For one thing, despite the widespread availability of recorded music by highly skilled players, many people still learn to play instruments themselves, and are content to produce music at an amateur level simply for the joy of the process. Countless amateur marching bands, college garage bands, and student jazz ensembles present throughout the world stand as a testament to the validity of this type of musical experience. Improvisational drum circles, which are often more about the process of drumming than the music they produce, are another example of people enjoying an active musical experience. In many ways, dance can be viewed as an active way to participate in a musical experience without actually making sound. Even within the medium of static recordings, a culture has recently emerged where a single DJ weaves a complex musical texture by mixing various recordings together into an endless chain of music. In this situation, passive listeners actively participate in a musical experience by dancing while listening to an active performer mix static recordings together live. This clouding of the division between active musical performers and passive listeners is a clear indication that people want more active ways to experience music.

1.2 The Problem with Interactive Music

Interactive music programs are often suggested as a way to employ computers to help unskilled users access the more active musical experience usually reserved for skilled performers. By allowing the computer program to take care of the "hard parts" of playing a musical instrument (making the sounds), amateur users would be freed to enjoy the "fun parts" of musical expression without the prerequisite ten years of piano lessons. Sadly, this technology has proved to be both frustratingly difficult to develop and disappointingly dull to use. Frankly, it's just not easy to let some guy off the street yank a couple of joysticks around to make a solo that sounds like Jimi Hendrix. The better you make the music sound when it comes out of the system, the less control you leave to the user and the less the user feels like he's playing an instrument. But the problem is really less the technology than the approach. Building special instruments that can magically take care of a user's musical deficiencies may not be the right goal to pursue.

Part of the problem is looking at the system from the perspective of an instrument at all. It's very convenient to cram a computer system that makes music into our comfortable definition of an instrument. That way, it basically acts in the same way as a traditional acoustic instrument, it just looks a little different and makes different sounds. On a more general level, its easy to think of a computer music system as a tool to produce music. After all, computers are used as tools to make hard tasks easier in many other areas, from science and math to business. In fact, much of the history of computer music has been dedicated to exploring the musical possibilities of the computer in this way. Computers often served as composer's assistants for tabulating complex serialist formulas, instruments for live performance, or sophisticated audio rendering devices. Giant efforts were (and continue to be) expended to mold the machines into aural facsimiles of traditional acoustic instruments. But in most of these cases, the computer was seen as a means to an end. The music they produced was actually quite traditional in the sense that it was meant to be listened to, much in the same fashion music has always been listened to, since the times of Palestrina or Bach.

1.3 Music In the Medium of Computation

Interactive music systems seeking to empower the passive listener with a more active role in the musical experience apply computers as tools to the medium of music and sound. This thesis proposes a different approach to interactive music, that of applying music and sound to the medium of computation. Rather than using the computer as a crutch to enable people to play music in traditional forms, this thesis suggests instead that composers write entirely new forms of music that are custom tailored to the medium of the computer. These forms should take advantage of the computer's many powerful attributes, not the least of which is interactivity. The difference between these two approaches is subtle but important. In the first, the computer is being used as an inadequate substitute; in the second it is a new vehicle for expression that conveniently bridges the gap between active performers and passive listeners.

Perhaps at some point in the future, interactive music systems will be recognized as a form of musical expression no less valid than a live concert performance or a studio produced album. Viewed as a collaboration between composer and participant, these systems might find their way into many corners of contemporary culture, appearing anywhere from clubs and coffee shops to the concert stage. Well-known artists could write for powerful, standard music platforms that might appear in homes as well as public establishments. Many different types of pieces could emerge, and, like pieces written for traditional instruments, they would require various levels of difficulty. Some would be designed for experts to showcase their skills, while others would be more suitable for novices. These systems would serve as a new musical form that is completely at home on the computer and could be played nowhere else. In a sense, they would be a natural outgrowth of musical expression in the computational medium.

Because video games are already such a successful form of interactive entertainment on the computer, they provide a useful terrain map for the strengths and weaknesses of the computer as a medium for interactive expression. These new types of interactive music systems might borrow more from video games than from their more traditional cousins, acoustic instruments. Like a video game, these systems ought to be accessible to novices, but leave plenty of room for improvement to an expert level. Similarly, they should be extremely responsive, demanding low latency and a high degree of information throughput between the computer and the user. The systems should offer a complex set of musical controls to the user in an understandable way, inviting him to explore and experiment as well as perform. Sophisticated graphics should play a major role in the control and aesthetics of the experience. In addition, multi-user participation, a staple of both music and modern video games, should be incorporated into these interactive systems. Moreover, they should make professional quality music that people would be willing to buy on CD, and not sound like toys. But most important of all, like video games, these interactive music systems should be a lot of fun to play.

1.4 The System: Stretchable Music

This thesis examines a particular interactive music system, called Stretchable Music, that was designed from the perspective described above. The system combines graphics, sound, and complex, interdependent musical controls in a new, particularly effective way. Users are invited to shape musical layers by pulling and stretching animated graphical objects with gamepad controlled pointers. The objects themselves have been carefully tailored to abstractly represent the layers of the music to which they correspond. At various times during the piece, animated icons representing tools pop up on the screen that, when grabbed, can give a user's pointer special powers to play a keyboard solo or a drum solo. The entire system follows a temporal framework or score that was predetermined by the composer. However, users can navigate between four distinct sections of the score by grabbing special advancement icons that appear at key times in the music.

The thesis will trace the development of Stretchable Music, highlighting many graphical, musical, aesthetic, and computational issues encountered along the way. It will discuss everything from the style of music in which the piece was written to technical details about the software design of the system. Similarly, it will discuss the roots of the ideas behind Stretchable Music and provide a brief survey of a body of work most closely related to the system. A critical analysis of the current system will be provided that will give not only this author's opinions on what worked and what didn't but will also include the opinions of others who have played the system. Finally, this document will discuss promising new directions that future systems developed in the style of Stretchable Music might take.

It is the hope of this author that, in addition to suggesting a new direction for interactive music, this thesis might lend insights to the design of other computer music systems. Many of the lessons learned during the development of this project may indeed carry over to the design of more traditional electronic instruments intended for skilled performers on the concert stage. With any luck, these lessons might more generally be applied to other applications in real-time computer interface design.

2 Related Work

Stretchable Music is by no means the first interactive system to ever be considered a piece by its creator. In fact, computer music has a rich and complex history etched with blurred boundaries between performers, instruments, and amateur participants. This section discusses work that is related to or had a direct influence on the development of the Stretchable Music systems. It begins with a brief survey of some of the roots of interactive music, continues through an explanation of the substantial work done at the Media Lab on the subject, and highlights other contemporary efforts. Following that survey, the discussion will turn to analyses from several experts in the field of interactive music and artwork which will provide interesting theoretical frameworks in which to characterize interactive music systems. Finally, a discussion of the projects worked on previously by the author will shed light on important lessons learned that made the development of Stretchable Music possible.

2.1 Early Interactive Music Systems

For the purposes of this document, an interactive music system can be defined as a computer system that generates music in response to a user's commands in real-time, but that generally takes care of some of the music-making process itself. Thus, simple event driven mechanisms like an organ or their computerized counterparts will not be discussed in favor of a few innovative projects that had the greatest influence on the development of the Stretchable Music systems.

2.1.1 The GROOVE System

One of the first interactive music systems to make use of a computer was Max Mathew's GROOVE system developed in 1968 at Bell Labs [Mathews 70]. Billed as a generalized system to record, edit, and manipulate human continuous control parameters, GROOVE's real purpose was to give performers new types of control over electronic music. The system's interface consisted of a series of knobs and a specialized, spatially tracked wand that allowed a human performer to input up to seven simultaneous channels of continuous control into a computer. The computer was equipped with many analog-to-digital and digital-to-analog converters that allowed it to sample and store the continuous control information in a specialized file system. The continuous control information was in turn used to control an analog synthesizer to produce music. The data could be played back and modified in real-time by a performer, or programmed ahead of time. Further, the system allowed programs to perform various mathematical operations on combinations of the control signals. Probably the most important feature of the system, however, was its capability for real-time control and editing through audible and visual feedback. Speakers provided audible feedback by playing the music that the system produced, while a CRT allowed users to actually watch all the streams of continuous control data pass through the system in real-time. Mathews described the level of musical control as "closer to that of a conductor to an orchestra than that of a player to an instrument" [Mathews 70].

2.1.2 The Sal Mar Construction

Another interactive electronic instrument, finished in 1972 and designed by Salvatore Martirano and a team of engineers at the University of Illinois, was called the Sal Mar Construction. The system consisted of a series of digital logic circuits connected to a bank of analog sound generating equipment. The sound was distributed through a multiplexer to 24 different speakers that could be arranged throughout a concert hall. Martirano designed the system primarily for real-time performance of algorithmically generated music. Its interface consisted of a panel of 291 touch sensitive switches that effected the digital logic control. The digital logic was connected to the sound generating hardware though a complex patch-bay that could be configured in many different ways. The level of control the user had over the system varied from minute details of the microstructure of a sound to the macro structure of the generating algorithms. One of the most interesting features of the system was the notion of a zoom interface that allowed the performer to control the system on many different time scales. The digital logic was organized into a hierarchical structure where different time scales of control were interconnected in a top down fashion. According to Martirano, the performer didn't really play the instrument, so much as he influenced its directions. Said Martrirano, "Control was an illusion. But I was in the loop. I was trading swaps with the logic. I enabled paths. Or better, I steered. It was like driving a bus" [Chadabe 97].

2.1.3 M

In the late 1980s interactive music became accessible to consumers through the advancement of the MIDI standard and the proliferation of personal computers. Several important and powerful real-time interactive music programs were written for the Macintosh by David Zicarelli, one of the most interesting of which was a generative music program called M. M was essentially a collection of tools for algorithmic composition that had a graphical interface which allowed users to modify parameters in real-time. The program included a fairly complex set of algorithms for interpolation and manipulation of rhythmic and harmonic motifs through a series of variations. Users interact with the program in two distinct stages: the setup stage and the performance stage. According to Christopher Yavelow, "During the setup stage, the user determines the basic musical material for four separate parts e.g. melodic patterns and/or pitch distribution ..., rhythmic patterns, accent patterns, articulation patterns, intensity ranges, orchestration ..., transposition configurations, proportions of pattern-variation methods ..., and tempo range" [Yavelow 89]. Users can then manipulate the settings of these parameters during the performance stage by manipulating a special control space in the graphic interface with the mouse.

2.1.4 Analysis

The three systems described in this section excelled in several common areas that make them particularly interesting and relevant to the Stretchable Music project. First, they all explored levels of musical control that were several steps removed from those available on traditional instruments. Mathews' GROOVE system allowed a user to modify pre-prepared musical control data (for an analog synthesizer) with continuous modification functions in real-time. Martirano's Sal Mar Construction focused on a hierarchical "zoom" architecture of control that yielded simultaneous access to minute attributes sound and broad parameters of algorithmic processes. Similarly, Zicarelli's M gave users process-level control over a variety of algorithmic composition mechanisms. Additionally, each system employed a novel interface that was particularly well suited to its level of control. For continuous control GROOVE used knobs and a joystick (and later added gesture input through an early version of Mathew's Radio Baton) while the Sal Mar Construction bared all of its algorithmic innards to the user through a complex panel of buttons. M took advantage of the graphical capabilities of the Macintosh Computer to employ a mouse-based graphical interface. Though none of these interfaces were perhaps ideal, they did reflect novel ways of thinking by their designers that were not chained to traditional instrument interface paradigms. Finally and most importantly, each of the systems placed a strong emphasis on real-time interaction and performance, effectively separating them from more common, off-line music editing tools. Thus these systems share several areas of interest with the Stretchable Music project including new levels of music control, new types of control interfaces, and real-time user interaction.

2.2 Interactive Music at the Media Lab

Interactive music research has been going on at the MIT Media Lab since its inception in 1985. Much of this research has sprung from the Opera of the Future research group (also known as the Hyperinstruments group), led by Professor Tod Machover. The group's research has evolved from the design of sophisticated computer instruments for skilled performers to the design of rich interactive experiences for amateur musical participants. Many of the projects discussed in the following section had a strong impact on the concept and design of the Stretchable Music project.

2.2.1 Machover's Hyperinstruments

The initial thrust of Machover's research at the Media Lab was to expand the sonic and expressive capabilities of traditional musical instruments through computer augmentation. Called Hyperinstruments, these digitally enhanced acoustic instruments were designed to allow virtuosic performers to control many different layers of accompanying electronic music with expressive gestures on their instruments [Machover 92]. This research culminated with Machover's composition of three pieces for stringed instruments called the Hyperstring Trilogy. Probably the most technically challenging of all the Hyperinstruments projects, the Hyperstring pieces attempted to measure with great precision subtle aspects about the way the performer manipulated his or her stringed instruments. Some of the parameters measured included finger position on the finger board, bow position in two axes, as well as bow pressure on the strings. This information was then analyzed on a computer to extract various parameters about the way the performer was playing. Machover then mapped these parameters to parts in the music. For instance, if a player played jaggedly with a lot of short irregularly placed notes, the system would mix in a jagged and irregular accompanying electronic line of accents to emphasize the effect of the players musical interpretation of the phrase [Rigopolous 94].

2.2.2 High-Level Control Systems

Later, Machover's group became interested in the design of computer instruments that would allow unskilled musicians to express themselves musically without worrying about the physical details required to master a traditional acoustic instrument. The two projects described below use high-level metaphors as a control interface to allow users to control the direction of the generated music. Each system focused on a different type of algorithm to achieve this connection between high-level parameters and actual musical output.

2.2.2.1 Matsumoto's Drum Boy

Implemented by Fumi Matsumoto in 1993, the Drum Boy system was an attempt to allow a user to traverse a complex rhythmic space using a variety of control methods [Matsumoto 93]. Based on a traditional drum machine architecture, the system would let users enter patterns of their own as well as trigger preprogrammed patterns and fills in real-time. A large rhythmic pattern database was organized in a hierarchical fashion according to perceptual parameters such as swing feel or syncopation, allowing users to call up new, similarly related patterns based on these parameters. However, probably the most interesting feature of the system was its ability to apply various transformative algorithms to existing patterns to morph them in the direction of various high-level salient parameters. By characterizing rhythmic patterns in a multidimensional perceptual feature space, the Drum Boy system allowed users to transform patterns along musically meaningful axes. In this way the system allowed real-time composing as well as rhythmic performance. The system used graphical output to display the state of the overall system as well as individual patterns. One considerable flaw, however, was the use of an ill-suited MIDI keyboard controller as a navigation device for this complex, rhythmic parameter space.

2.2.2.2 Rigopulos' Joystick Music

Later, Alex Rigopulos, with help from Eran Egozy and Damon Horowitz, developed a system to allow users to control harmonic and melodic musical material using similar high-level parameters. This time utilizing a pair of joysticks as an input device, the system used probability driven, generative algorithms to generate music in real time from abstract musical seeds [Rigopulos 94]. These seeds were essentially time varying envelopes of the continuous parameters used to control the generative algorithms. The system also included an analyzer function that translated existent MIDI sequences into seeds that could be used to reproduce the sequence. The power of the seed representation, however, was in its ability to allow the musical material to be easily modified in real-time along perceptually significant axes. Relevant parameters included rhythmic density, syncopation, harmonic color, and melodic direction. Two versions of the system were implemented, one to allow a user to control a piano comping pattern to a one bar repeating riff based on Herbie Hancock's "Chameleon," and another to allow the user to have fairly detailed control over a piano solo played over a 12 bar blues progression. Each of the systems utilized two joysticks at once to take in a total of four continuous control parameters, in addition to several binary controls provided by buttons on the joysticks.

2.2.3 Waxman's Musical Experiences for Amateurs

Composer David Waxman came to the Media Lab to continue his exploration of musical experiences designed for amateur users. Waxman made extensive use of electric field sensors, designed by Neil Gershenfeld [Zimmerman 95], as input devices that would allow users to interact with his instruments by gesturing with their hands in the air [Waxman 95]. The sensors worked by measuring the change in an electric field induced by the presence of a person between transmitting and receiving electrodes. Such non-contact sensing was deemed an attractive way to allow users to control music (seemingly magically) by moving their hands in the air. Waxman measured a variety of parameters about the users gesture ranging from absolute hand position and velocity to activity and the detection of discrete beating gestures. Further, two of the instruments he created were actually designed for two players to participate at once. Probably more important than his contributions to the analysis of gesture, however, was Waxman's approach to the design and composition of these interactive musical experiences. Directly influencing the approach eventually taken by the Stretchable Music project, Waxman thought of each of the systems he designed more as pieces or experiences than as enabling technology for crippled amateurs (as was the approach of the two projects described immediately previously). Says Waxman in the final pages of his Master's thesis,

> After building these projects, I realize that composers cannot simply translate their music to new electronic 'instruments,' but must stretch their own thinking to express themselves in a completely new medium. Over the last two years, I have improved as a composer of interactions as much as a composer of music, and though one can never ignore the latter, the two cannot be thought of independently [Waxman 95].

This sentiment was taken to heart in many ways by the author in both the development of the Stretchable Music systems as well as the thinking behind the core ideas of this thesis.

2.2.4 The *Brain Opera*

Probably one of the most ambitious interactive music projects ever attempted, the *Brain Opera* [Machover 96], [Machover 98] represented the culmination of many years of research by Machover's Opera of the Future group into interactive musical experiences for amateurs. Part futuristic musical video arcade, part avant-garde electronic music performance, and part cyber-music event on the Internet, the *Brain Opera* tried to bridge many different types of musical experiences into a coherent aesthetic whole. The opera was inspired by Professor Marvin Minsky's *Society of the Mind* [Minsky 86] which theorizes that the human mind is really a collection of many agencies working together to create consciousness. Audiences of up to 200 people at a time would begin their experience with the *Brain Opera* by interacting with six different kinds of high tech instruments in the Mind Forest. Like Waxman's Digital Theremins, these instruments were more compositions than tools, designed to let the audience explore several different facets of the music process: rhythm, melody, harmony, and the sung voice among others. A variety of different novel interfaces were developed for these different instruments including, touch-sensitive (and expressive) Rhythm Tree pads, electric field sensing Gesture Walls, and a novel handle-bar-like driving interface for Harmonic Driving [Paradiso 97]. Several of the instruments also made use of graphics to reinforce their aesthetic effect and communicate musical control to the users.

After interacting with the instruments in the Mind Forest, audiences would then go and see three skilled musicians perform the final *Brain Opera* piece using instruments similar to the ones with which they had just played. The performance was supposed to connect to the Mind Forest both conceptually and musically; samples that the audience had recorded during their previous interaction with the Mind Forest were often used as part of a musical texture in the performance. Similarly, the performance also served to connect the Internet presence of the *Brain Opera* to the rest of the experience. Besides providing extensive documentation of the project and live video and audio broadcasts of the performances, the *Brain Opera* web site also allowed users on the Internet to contribute to the performance of the piece. Users could manipulate a Java applet that sent parameters to an algorithmic music program generating music in real-time during the third movement of the piece. Despite receiving mixed reactions from its debut at Lincon Center in 1996, the *Brain Opera* proved to be a fertile testing ground for many different ideas about the way people may play and experience music in the future. It is the conviction of this author that the *Brain Opera* will no doubt be the first in a new line of interactive entertainment experiences that will rise to popularity during the next century.

2.2.4.1 The Singing Tree

The interactive musical experience that probably worked the best out of all of the instruments in the *Brain Opera*'s Mind Forest was the Singing Tree; and as such it deserves special mention because of its careful synthesis of technology and aesthetics to achieve such successful musical experience. The Singing Tree was designed by Will Oliver, Eric Metois, and John Yu as an intimate musical experience for a single user to meditate on a pure tone by singing into a microphone [Oliver 97]. The microphone and an active matrix LCD display were nestled into a white silicone hood to provide a sense of privacy for the participant using the system. The system analyzed in real-time how the user was singing into the microphone and generated music reflecting this analysis that was played back to the user as she sang. For instance, if the user sang erratically or hoarsely, the system would generate dissonant, clashing harmonies with chaotic timbres. However, if the user successfully held a pure tone without wavering, the system would play very consonant and "uplifting" music, mixing in sampled voices of chorus among other sounds. Similarly, the system used graphics displayed on the LCD screen to reinforce the user's attempts to sing purely. Imagery such as a flower blooming or an eye opening would accompany the change in state of the system as the user sung more and more purely. The overall aesthetic resulted in quite a beautiful experience both visually and musically, and was certainly one of the high points of the *Brain Opera*'s Mind Forest.

2.3 Other Contemporary Interactive Music Projects

Research in interactive music has by no means been confined to the Media Lab. In fact many different commercially available products have been developed around the idea of music that can be modified or interacted with in real-time. Unfortunately, many of these products have fared poorly (as described in the introduction) for a variety of reasons, to some extent giving interactive music a tarnished image in the public consciousness. This section highlights a few commercially available products that are interesting. None represent a completely successful manifestation of music in the medium of computation (in the opinion of the author) but each excels in a different manner of which it is worth while to take note.

2.3.1 Peter Gabriel's EVE

One of the pioneers who led the rise of progressive rock in the 1970's, Peter Gabriel has consistently pushed the boundaries of popular music into new territories during his nearly three decade career. Thus it seems only natural that he would be on the vanguard of new music technology, including interactive music. While most artists are still struggling to comprehend the medium, Gabriel has already released two critically acclaimed full length interactive titles, the second having recently received the 1996/97 Milia D'Or award for originality, creativity, and overall outstanding multimedia achievement [E 1997]. Like his earlier work Explora, Gabriel's latest effort, EVE (which stands for Evolutionary Virtual Environment), combines the massive storage capacity of a CD-ROM with carefully pre-produced audio and graphics [Gabriel 97].

The music is all Gabriel, selected from some of the best of his previous musical work, while the graphics are the commissioned works of some of Gabriel's favorite visual artists. The system invites a user to explore several different musical worlds, solving simple puzzles along the way to work towards the goal of the piece which is to reach the final "paradise" world [E 97]. The main mode of interaction is essentially point and click at static images on the screen. Users trigger sampled audio by clicking on specific parts of the graphic or by dragging static images around on the screen, adding to the background sonic texture composed by Gabriel.

Like many other titles in the multi-media, CD-ROM genre, EVE suffers from the limited degree to which it engages the user to interact with it. Unlike more traditional video games which demand fast-paced, continuous interaction from their users, EVE incorporates a much slower exploration and puzzle solving model. Though this model has proved successful in other CD-ROM titles such as Myst, one wonders if it does justice to the engaging nature of Gabriel's music. One reviewer complained of the sprawling nature of the experience he had playing the game [E 97]. He found some of the music disjointed and the puzzles superfluous to the overall interactive experience. This author echo's that criticism adding that many of the puzzles impeded his progression through the piece to such a degree that he lost interest. The question then becomes, where the puzzles necessary at all? Is a challenge-based goal structure (like those in traditional video games) necessary or even relevant to an interactive music experience? These questions will be further explored in chapter four of this thesis. Despite these shortcomings, however, Gabriel's EVE remains a good example of the state-of-the-art in commercial interactive music and will no doubt be viewed in the future as one of the important formative stepping stones in the field.

2.3.2 Toshio Iwai's Musical Insects

Probably one of the most innovative interactive music systems to combine graphics and sound in a simple, provocative way was developed by Toshio Iwai during his residency as an artist at the San Francisco Exploratorium in 1992 [Iwai 92]. Called Musical Insects (and later developed in to a commercial product called Sim Music released by Maxis [Iwai 96]), the piece allows users to literally draw musical material (notes) onto a virtual landscape by painting with small colored tiles. Animated insects then traverse the landscape playing out any musical material in their path. Every time an insect encounters one of the user's tiles, it plays out a single note. Different colors of tiles represent different pitches in a scale of the user's choice. By spatially quantizing the user's brush strokes into tiles, Iwai cleverly enforces rhythmic quantization on the music played out by individual insects as they traverse the tiles at a constant rate. Further, by arranging the tiles in various spatial patterns, users can add rhythmic patterns to their melodic material. The system allows up to four insects to traverse the landscape at once, each of which may represent a different instrument. By utilizing special tiles which cause the insects to change direction or jump to a different location, users can build up complex musical textures with spatially represented patterns that can also be visually interesting.

Of the interactive music systems described in this section, Iwai's Musical Insects offers the most detailed control of basic musical material to the user. Rather than allowing users to explore or manipulate music composed by someone else, the system invites them to compose their own musical material through its simple, innovative graphical interface. Hence Iwai's contribution is more of an enabling interface to allow musically unskilled users to compose their own music than a specific statement of his musical ideas in interactive form. Regardless, the system's clever, direct mapping of graphics to musical output and control is worth noting for both the design of interactive music systems that are pieces (as is the system described in this thesis) as well as those that serve as enabling tools for amateur users. The system suffers, however, from both musical limitations imposed by its interface and its limited sonic pallet. Though users can "paint" in real-time while the insects are performing their piece, the system has no provision for the any continuous control, required, for instance, to articulate various musical lines in the texture. Further, the system's sonic pallet is a disappointingly narrow selection of sample-based instruments reminiscent of the General MIDI specification. As expressed previously in this thesis, General MIDI tends to produce bland, generic music. This lack of sophisticated sound control is probably the key contributing fault that unfortunately dooms Iwai's system to sounding like a toy.

2.3.3 Harmonix: The Axe

A commercial product based on technology developed at the Media Lab as part of Rigopulos's joystick music project {Rigopulos 94], The Axe by Harmonix Software allows users to manipulate high-level parameters of a lead, melodic line over background sequences [Harmonix 98]. The system essentially allows the user to act as an improvising soloist over several popular songs based in a variety of musical contexts (jazz, rock, folk, blues etc.). The musical controls are very much the same as those described in the section above regarding Rigopulos' joystick music. Users can use either a mouse or a joystick to control the melodic and rhythmic contours of a lead instrument. The mappings for both the joystick and the mouse allow the user to move rather literally within a two dimensional parameter space, adding extra trills and musical tricks with button presses. Up to two users can interact with the system at once allowing for exciting multiple solo scenarios such as dueling banjos. The system also provides several modes of graphical representation of the user's interaction with the music, the most interesting of these being real-time notation of the users solo on a traditional music score. The program allows users to save their performances to play back to their friends at a later date.

Of the interactive music systems described in this section, The Axe has the most sophisticated algorithmic music generation engine. Its ability to allow unskilled users to shape perceptually relevant parameters of a lead solo line is truly unmatched by other contemporary, commercially available systems. Further, unlike Gabriel's CD-ROMs, The Axe provides plenty of continuous, real-time control that engages the user on a moment-by-moment basis. The sounds it produces aren't great, but are more interesting and carefully chosen than those in Iwai's Musical Insects. The main problem with the program in the opinion of the author, is its apparent identity crisis. The system isn't sure whether it's a toy, an instrument, a video game, or something else. Harmonix's literature on their web page vigorously insists that the system is an instrument for amateurs, but the kind of flexibility and freedom that comes with a normal instrument seems to be missing here. And, unlike Gabriel's work, there's not enough new, interesting music to explore to make the experience worthwhile (most of the songs are well known, popular favorites). Yet, in contrast to Iwai's work, there's little facility to allow users to use the new technology to compose new songs of their own. Future improvements listed on the Harmonix web page, including collaborative, Internet jamming, will definitely help take the system beyond the initial, new technology fascination stage and hopefully make it genuinely more fun to play.

2.3.4 Interactive Music Tools for the Web

Though not examples of interactive music systems themselves, several products have emerged to facilitate the development of interactive music for web-based applications, namely web pages. Of these, two in particular championed by popular electronic music veterans Brain Eno and Thomas Dolby, deserve special mention because of their unique implementation and flexibility. Each provides an interesting example of the kinds of tools that would be necessary to develop interactive music systems commercially (e.g. a software synthesizer and parameterized control).

2.3.4.1 Brian Eno and Koan

The British company SSEYO has developed composition and music generation software called Koan, that allows composers to write compositions specified in envelopes of the program's over 100 continuously variable parameters [SSEYO 98]. The software focuses on generating music in the ambient and techno genre of electronic music, and has been used by Brian Eno to compose a recent series of 12 pieces. The software comes in two flavors, an editing suite and a freely available player that is necessary to realize the music written in the Koan format (often this player takes the form of a web browser plug-in). The generative algorithms rely heavily on stochastic decision making processes, that, as the company explains, makes every rendition of a particular piece a little bit different. This flexible, parameterized notion of a musical piece fits well with Eno's idea of ambient music permeating all corners of contemporary life. With systems like Koan, he hypothesizes, the music will no longer be static audio facsimiles of a single performance of a piece [Eno 98]. Koan also allows for limited interactivity; "koan^198" is an online collaborative piece combining the musical efforts of many different composers (of which Eno is one) [SSEYO 98]. Users can turn various layers from different composers on or off, and go to different web pages which highlight that particular composer's musical contribution. This level of interactivity seemed arbitrary and misguided given the supposedly powerful parameterized nature of the Koan system (ought the user be able to influence more?). Similarly, the author was not impressed by the sonic result of this piece, possibly because of his inferior sound card. Any expressive nuance that might have differentiated between composer's contributions was lost in a soup of primitive squawks and buzzes. SSEYO, it seems, might want to think about making their software perform more favorably on a wider variety of sound cards.

2.3.4.2 Thomas Dolby's Beatnik

Thomas Dolby's Beatnik software, produced by his interactive audio company Headspace, represents another attempt to bring parameterized audio to the web [Headspace 98]. Like Koan, it provides a free, specialized playback software that plugs in to a web browser and uses software synthesis to generate sounds. Similarly, composers must utilize proprietary authoring software to generate music in Beatnik's own Rich Music Format (RFM). RFM essentially wraps basic MIDI functionality with custom sample specification and specialized controls for the Beatnik audio engine. Unlike Koan, however, Beatnik lacks a parameterized algorithmic music generator which makes it stylistically more flexible, but less readily capable as a music engine for an interactive music system. However, its RFM sound format works with just about any sound card and produced music of impressive sound quality on the author's system. Beatnik does provide developers Java Script hooks that can be called from web pages or (presumably) Java programs themselves, encouraging others to develop interactive music content using the system's musical tools. Using these hooks, developers can start, stop, mute, solo, and transpose sequences as well as control some sonic parameters on the software synthesizer. Several small interactive music programs linked off the Beatnik webpage demonstrate these hooks with disappointingly dull mappings between simple graphics and various musical layers.

2.4 Theoretical Frameworks for Interactive Systems in Art and Music

Several authors have tackled the task of defining useful theories for designing and thinking about interactive systems for music and art. Not surprisingly these theories are in no way complete. This is partly due to the fact that the field is very new, and partly due to the fact that it is difficult to completely characterize aesthetic expression of any kind within the bounds of a theory. Nonetheless these theories do provide handy guidelines for the designer of such systems, and begin to lay the groundwork for intelligent and consistent criticism of such systems. Hence, they will no doubt come in handy when we take a closer, critical look at the Stretchable Music systems later in the analysis section of this thesis.

2.4.1 Robert Rowe's Axes of Interactive Music

Robert Rowe wrote one of the first books dedicated solely to the investigation of interactive computer music systems [Rowe 93]. The book describes in depth his design of one such system, called Cypher, which models the interactivity of a human player by providing separate software stages for the analysis of musical input and the composition of musical output. In Rowe's model, the computer listens to the music created by a player, analyzes it, and then decides what to play based on many of the same notions a trained human musician might use. Though this direct approach differs from the Stretchable Music systems in that it assumes the user is playing a musical instrument as an input device, and that the user has reasonable control over that instrument, Rowe's initial characterization of the broad types of interactive music systems based on their relationship to the user and the methods of musical generation are important to understand. The principle categories that he identifies can conveniently be laid on axes as follows:

> *Score-Driven vs. Performance-Driven Systems*: Score-driven systems progress in a linear manner, carefully adhering to a prepared score. Performance-driven systems have no pre-set track and are therefore more suitable for improvisational settings.
>
> *Transformative vs. Generative vs. Sequenced Systems*: Transformative systems take complete musical material as input and apply rules to transform it into something else. Generative systems use abstract musical data such as chordal information and scales to create new musical material, often stochastically. Sequenced systems trigger prepared musical fragments stored in memory, occasionally modifying simple parameters such as tempo.
>
> *Instrument vs. Player Paradigm*: This axis refers to the role of the music system during performance. Systems using the player paradigm often act as accompanists to a human performer and display a bit of autonomy. The instrument paradigm describes systems that respond only to direct human control, much the way traditional acoustic instruments do.

Though the Stretchable Music systems trace a varied path through these axes, certain generalizations can be made to conveniently characterize them within this framework. Generally, the systems are score-driven in that the main part of the experience is scripted out ahead of time by the designer in an object score. In fact, one of the important innovations of Stretchable Music was the introduction of large-scale time structure into a novice user's musical experience using this mechanism. Objects themselves cover most of the Transformative vs. Generative vs. Sequenced axis, though in the final implementation of the system they lean heavily towards the transformative and sequenced end as a mere matter of implementational convenience. Finally, though the objects' control interfaces indeed suggest that they are virtual instruments, the personalities of their visual design, behavior, and strongly biased aesthetic leaning of musical content and control place them closer to the player paradigm of Rowe's third axis. In fact the relationship of the user is a little more complex; the relationship of a puppeteer to a marionette seems more appropriate.

2.4.2 Todd Winkler

Winkler's book, *Composing Interactive Music: Techniques and Ideas Using Max*, provides another interesting perspective on interactive music as well as practical firsthand advice [Winkler 98]. Extending Rowe's ideas, Winkler defines the role of an interactive music system in terms of five separate stages or functions, detailed as follows:

> 1. Human Input, instruments - Human activity is translated into digital information and sent to the computer.
>
> 2. Computer listening, performance analysis - The computer receives the human input and analyzes the performance information for timing, pitch, dynamics, or other musical characteristics.
>
> 3. Interpretation - The Software interprets the computer listener information, generating data that will influence the composition.
>
> 4. Computer composition - Computer processes, responsible for all aspects of the computer generated music, are based on the result of the computer's interpretation of the performance.
>
> 5. Sound generation and output, performance - The computer plays the music, using sounds created internally, or by sending musical information to devices that generate sound [Winkler 98].

In his second chapter, "Interaction: Defining Relationships between Computers and Performers," Winkler goes on to discuss interactive composition in terms of three models: performance models, instrument models, and composition models. Within each model, he discusses several approaches a composer or designer can take, and explains pertinent issues to the design of such systems. For example, in the Performance family of interactive systems, Winkler explains the ins and outs of the Conductor model (a top down high-level control system focused on the input of a single user), the Chamber model (a more democratic dialogue where user and computer vie for control at different times), the Improvisation model, a model where the computer generates independent, improvised music in a situation similar to Rowe's player paradigm), and the Free Improvisation model (where a system is an extremely complex set of musical controls and generative algorithms that can be accessed simultaneously or at different times). Winkler also makes interesting assertions about the roles of different levels of indeterminacy in the design of interactive systems. There he discusses non-linear systems where the structure of the piece is left up to user control along with randomness at the note level in generative improvisational algorithms. In a similar fashion to Rowe, the specifics of Winkler's discussions do not relate directly to the Stretchable Music systems, but the more abstract design issues faced are many of the same. The author only regrets that the book was not published before his systems were developed.

2.4.3 Brenda Laurel

Brenda Laurel suggests three important variables are key in the design of interactive systems. They are frequency, range, and significance and are explained as follows [Laurel 89]:

> *Frequency*: Frequency represents the rate at which the user is prompted to (or able to) interact with a system. Does the user make decisions only occasionally at certain branch points, such as in the popular Choose Your Own Adventure book series, or does he have continuous access to control over the system like a video game?
>
> *Range*: Laurel describes range as the span of control the user has over the system. Is the user's primary control at a high-level like a conductor leading an orchestra, or does the system allow the user to access the intimate details of its processes, such as a professional sequence or sound design program.
>
> *Significance*: Significance is described as the degree to which the user has control over crucial elements of the system. Choosing the style and direction of a piece of music would be considered more significant than perhaps simply adjusting its volume or tempo.

Laurel goes on to suggest that these axes, however, only tell half the story. In here seminal book, *Computers as Theater*, she suggests that "there is another, more rudimentary measure of interactivity: You either feel yourself participating in the ongoing action of the representation or you don't" [Laurel 91] Laurel favors thinking about interfaces to interactive systems from a theatrical point of view. Users and the computer both work together as active agents participating in actions on the common ground interface of the stage. Literal metaphors, she asserts, are helpful but dangerous because they usually only tell part of the story. "We often fail to see that these are representations of tools and activities and to notice how that makes them different from (and often better than) the real thing" [Laurel 91]. Some of Laurels points re-enforce the abstract nature of the graphical interface of the Stretchable music system. Rather than trying to literally represent, say, the bass object as a personified bass player and thus baiting the user to expect the capabilities of the “real thing” (for instance the ability to improvise), Stretchable Music makes clear, yet abstract connections between its graphical representation and its functionality.

2.4.4 Chris Dodge’s Process Framework

Like Brenda Laurel, Chris Dodge, in his thesis “The Abstraction, Transmission, and Reconstruction of Presence: A Proposed Model for Computer Based Interactive Art” [Dodge 97] does not directly address the issues of interactive music system design, but takes aim at the related category of interactive installation and visual artwork. Unlike Laurel, however, Dodge’s treatment of his subject is somewhat more practically based in the design of the architecture for such systems. Dodge champions a design architecture not unlike the successive stages of analysis and generation advocated by Rowe and Winkler, where a user’s input to a system is analyzed for salient, perceptually important information and transformed into interesting graphical and visual output. He goes further, though, to describe many ways this data may be deduced from a variety of input devices, and gives special attention to transmission and mapping stages between the input and output of the system. Dodge’s framework revolves around a networked implementation of such a system where many different computers simultaneously share input and output with each other. To Dodge, this process of analysis, data reduction, communication, mapping, output, linked in a loop with human interaction represents the true essence of interactive art in the computational medium. Early on in his thesis he gives arguments for why this train of thought actually falls out of postmodern theory in the visual arts. Again the abstract lessons learned from this interesting investigation and characterization of interactive installation and visual art will no doubt prove valuable during closer analysis of the Stretchable Music systems.

2.4.5 John Maeda

Finally, Professor John Maeda provides an important perspective on the new relationships of artists and tools to the computational medium in his essay “A Framework for Digital Expression” [Maeda 97a]. There Maeda defines the traditional art of expression with four basic components:

> 1) media: there is some external vessel that can hold the concept outside of the expresser’s mind, such as paper, clay, etc.
> 2) tools: there is some way to shape the media in a conscious manner, such as with one’s bare hands.
> 3) skills: the expresser understands the physics and metaphysics of his media and tools, and his experience with them allows him to mold form of superior craftsmanship. Through his experience, he possesses a basic vocabulary by which he can express himself.
> 4) concept: there is something that the expresser wishes to express; most importantly, he has the will to express (this can include the will to express no concept at all). The expresser has an imagination within which the concept is nurtured and brought to reality with technique, tools, and media [Maeda 97a].

He goes on, then, to point out that in the digital domain, the first three of these collapse into an undifferentiated stew. The designer in the digital domain, he suggests must define his own medium (bounds of his expression), create his own tools relevant to these bounds, and maintain the technical skills to execute these on the computer.

> Digital craftsmanship is the ability to skillfully craft a form, combined with the ability to craft the very material of the form, together with the skills to craft the tools to manipulate the form. Creating a new material implies choosing the optimal levels of abstraction to define the overall boundaries of the form, and choosing or creating the best set of tools to iterate toward the best fitting form created with this material [Maeda 97a].

Thus, in the wide-open wilderness of digital expression, designers must do more to define their boundaries and tools than their pre-computation predecessors (analogous to Michael Jordan having to rewrite the rules of basketball as well as perform virtuosically within those rules). Maeda continues then to point out that the prevalence of commercially available design tools has tipped the balance of this process towards design within narrow, technically driven boundaries that stifle new creative thought within the community. Finally Maeda calls for a new examination of the traditional values of design applied, without obscuring lenses of technological perfectionism or application, to the computational medium to hopefully revitalize a new and creative movement in computationally-based expression.

These sentiments had a tremendous impact on the design of and motivation behind this project. In some sense, Stretchable Music attempts to set up its own medium by defining such constants as the mouse interface, graphical representation and control of musical material, and a conceptual division of the musical material into separate objects that can be controlled independently. Within these bounds, the systems choose various forms to explore several levels of musical control to express abstract musical concepts. Further, the tools with which to manipulate these musical forms were designed into the system as musical controls that were part of the piece. Finally, the systems pay special attention to the flow of time and the concepts of motion, process, and interactivity, all of which Maeda values highly in digital design: “The utmost responsibility of the post computer designer, in addition to knowing the basic elements of static form,” he states, “is to learn and create elements of dynamic form” [Maeda 97a].

2.5 Lessons Learned from Past Interactive Music Projects

During his nearly six year tenure with the Opera of the Future group, the author had the opportunity to contribute to a number of different interactive music projects. In the following section, several of these projects will be discussed from the perspective of the important issues they raised about interactive music. The lessons learned from these projects had considerable influence on the motivation, design, and development of the Stretchable Music systems described at the beginning of this thesis.

2.5.1 Public Installation: *Vox Cubed*

In the spring of 1994 Tod Machover's class "Music, Media, and Aesthetics" presented a public installation of student works created during the semester. Called *Vox Cubed*, this installation featured several interactive musical experiences designed for amateur users that explored such themes as voice, gesture, collaboration, and immersion in a lively musical environment. The author and a group of four other students contributed to the exhibit "Spatial Music," an immersive, virtual musical environment where users could navigate between several different styles of music by walking around a room. The room was outfitted with large electric field sensors on all four walls to track a single user as he traversed a 14 by 14 foot transmitting "carpet" on the floor. A computer used this position information to mix down five different pieces of original music depending on where the user stood in the room. The pieces were assigned virtual positions in the room with different tracks in the individual pieces slightly displaced (spatially) to overlap with each other. By walking around, a user could mix various parts of different pieces together to create an interesting and evolving musical texture. Each of the pieces of music was written by a member of the group and represented a different style of music including Jazz, Techno, Pop, Industrial, and traditional Indian Music. To round out the virtual sonority of the room, a few spots had sound effect tracks that gave the illusion of a waterfall or a bubbling brook. The software system, written by the author in the Max environment, did simple tracking from the sensor data in order to modulate the musical playback of the various pieces.

The entire *Vox Cubed* show was an exciting experiment as a large scale interactive music installation. The students' projects filled the Media Lab's Villers Experimental Media Facility (also known as the "Cube") and overflowed into a room next door (where the "Spatial Music" exhibit was housed), providing the audience with a variety of different experiences from which to choose. The entire experience melded into a melodious cacophony of sight and sound that may have been a testing bed for some of Machover's (then) future ideas about how the *Brain Opera* might look and sound. On a smaller scale, the "Spatial Music" installation explored issues about how to design an immersive musical experience that was flexible but coherent for uninitiated users. For example, though the music in the experience varied in style, it all shared important elements such as tempo and key, making transitions from one song to another less jarring. Such design considerations played a major role in the author's contributions to future interactive installations and had a substantial impact on the kind of thinking that was necessary for him to produce Stretchable Music.

2.5.2 Expressive Instrument: The Penn and Teller Sensor Chair

Later that summer Machover began a collaboration with magicians Penn and Teller to create new trick that fused technology, music, and magic into a lively theatrical piece to premier that fall at a Media Lab conference called *Digital Expression*. The idea was to build some new type of instrument for Penn and Teller to use in the piece that would be half musical instrument and half magical prop. The trick turned out to be a seance trick, and the instrument turned out to be a chair to which Teller would be bound, only to magically escape at a spirit's whim. The possessed Teller would then channel voices from the dead and musical out-bursts behind a thin, drawn curtain. The chair could also be played as an instrument, demonstrated by Penn at the beginning of the trick, with the sitter gesturing with his arms in the air in front of the chair. Hardware architect Joe Paradiso positioned four electric field sensors in a square directly in front of the chair [Paradiso 97]. As the user sat on the chair, a transmitter connected to the seat electrically coupled to the player allowing the four sensors in front to accurately sense the position of his outstretched arm (for mathematical reasons it was easier to sense one arm at a time). This hand position was then mapped to a variety of musical outputs, thoroughly explored in an eight section, three and a half minute piece by Machover appropriately titled “Penn’s Sensor Solo” [Machover 94]. For example, one mapping allowed the user to trigger rhythmically quantized drum samples arrayed in a 20-by-20 grid in the sensor space. Through this technique, spatial patterns were mapped to rhythmic patterns with surprising accuracy; users managed to achieve fairly repeatable results between performances. Another successful mapping allowed the user to sweep a fast paced melody through timbre and register space, while delicately adjusting its volume and accents.

In addition to being used as a sophisticated instrument for performance, the Sensor Chair also proved to be a successful instrument for amateur users. Certain modes such as the ones described above employed clear and interesting musical mappings that enabled novice users to begin to express themselves almost immediately. Users even discovered new ways to play the chair that weren’t intentionally designed into the system. Many delighted in the discovery that they could elicit a certain loud percussive sound every time they sat down on the chair (during a demonstration, comedian Robin Williams proclaimed that this should be made standard equipment on every toilet seat in America). Since the performance at *Digital Expression*, the chair has been shown in a variety of installation settings allowing young and old alike to take their turns at a drum solo or other less traditional forms of musical output.

The software for the system was developed by the author and graduate student Eran Egozy in Macintosh Common Lisp using a framework created by Joe Chung named Hyperlisp [Chung 91]. Utilizing powerful concepts from object oriented design including inheritance and a data-driven architecture, the Sensor Chair software system was far more powerful and sophisticated than the *Spatial Music* system implemented in Max. Due to a variety of tweaks and a specialized calibration layer, the system responded with considerable accuracy and nuance to the user’s gestures in the various modes of “Penn’s Sensor Solo.” Further, though a large amount of the code was dedicated to Penn’s short solo at the beginning of the piece, the software system was also responsible for controlling lights and music for the rest of the 22 minute mini-musical. The experience gained from putting together such a sophisticated system (and working with Eran) greatly enhanced the author’s ability to attack future large scale software projects including the *Brain Opera* and Stretchable Music.

2.5.3 Gesture Instrument: The Digital Baton

Following the lead of the Sensor Chair, the Digital Baton, conceived by Teresa Marrin [Marrin 96] was developed as an even more sophisticated instrument to translate the gestures of a performer into musical output. The device was designed as a digitally augmented version of a traditional conductor's baton that utilized several modes of sensing to gain input from the user. These inputs ranged from tracking the baton's tip in two dimensions with an IR camera, to detecting beats in the performer's gesture through accelerometers built into the base of the baton. In addition, five pressure sensors on the surface of the baton sensed the way in which the performer squeezed the device while gesturing. Unlike a traditional baton, however, the Digital Baton was not a slender, lightweight stick. Rather, its body was shaped more like an egg with a short stick protruding from its tip where an infra-red LED allows the tip to be tracked. Using the recently developed Rogus McBogus MIDI library [Pelletier 96], the author and Patrick Pelletier designed specialized software for the Baton (in C++) to collapse this data into meaningful events that could be mapped to musical output. The system used a mode structure similar to that found in the Sensor Chair or some of Machover-'s earlier Hyperinstrments. Both position sensing and pressure sensing proved to be quite successful on the device. On the other hand, beat detection, using the accelerometers at the base (instead of at the tip), left much to be desired. Experience gained organizing such disparate input data into some kind of musical sense helped the author to tackle an even more formidable challenge later on with the *Brain Opera* performance system.

2.5.4 Large Scale Installation and Performance: The *Brain Opera*

As described above, Machover's *Brain Opera* was the culmination of many years of research into interactive instruments and installations for amateurs. Nonetheless the *Brain Opera* did incorporate a sophisticated performance system designed for three skilled musicians to play the final piece for a more traditional, seated audience. The performance system consisted of three Hyperinstruments including the Sensor Chair, the Digital Baton, and a combination of Mind Forest instruments, Gesture Wall and Rhythm Tree, affectionately dubbed the "Gesture Cow." These three instruments were linked together by a master computer system that kept them synced to each other and to backing audio, MIDI, and video laserdisc recordings. Each of the instruments utilized a Hyperinstrument mode structure to incorporate Machover's musical material into 26 subsections of mappings (or modes) across three movements. The majority of these modes involved the real-time modification of sequenced musical material, though the mappings and modifications controlled by the performers changed with every section. In several modes, two of the instruments had to communicate on a more intimate level to line up their musical material for a type of "call and response" interaction. In such a situation, the "Gesture Cow" would trigger a sample of Professor Minsky asking a question about music, to which the Baton would respond with answers to that question that had been recorded earlier from audience members while they were in the Mind Forest. The software development effort of the performance system was led by the author and included significant contributions from John Yu, Kai-Yuh Hsiao, and Eric Metois.

In addition to developing software for the performance system, the author was able to travel with the *Brain Opera* to several destinations including New York, Linz, Tokyo, West Palm Beach, and Lisbon. Through casual observation of the diverse audiences from these locations, the author and others have noticed interesting trends in the way people respond to the *Brain Opera* experience. One surprising result was the enthusiasm of young and old people to the experience. People under 16 or over 60 seemed to have a much better time exploring the musical capability of the various interactive instruments in the *Brain Opera* lobby than did their more middle aged counter parts (another thesis is surely due to explain why this is the case). On a different note, different types of instruments were met with varying degrees of enthusiasm depending on the design of their interface. For example, installations with clear mappings between user input and sonic output, such as the Melody Easel or the Rhythm Tree, tended to fare well, especially when they incorporated graphical feedback into the loop. Finally, the diverse nature of the cultures that were observed interacting with the *Brain Opera* allowed the author and others to formulate hypotheses differentiating universally accessible features of the project from those that were more culture centric. On the whole everyone seems to enjoy the interactive part of the show, probably in no small part due to the sheer novelty of the experience. Few have had the opportunity to control music on this level before, or to explore such an foreign and interesting set of interfaces. The final performance, however, seemed to be more warmly received in Europe than in the United States, perhaps due to the former's stronger tradition in the arts and greater familiarity with the avant garde. Japanese audiences were not at all shy in the interactive Mind Forest (as was expected) and quite enjoyed the final performance.

2.5.5 MAS 964: The Principles of Visual Interface Design

Though not directly related to interactive music, Professor John Maeda's class "MAS 964: Principles of Visual Interface Design" had a strong influence on the Stretchable Music project. The class was essentially an art class where students implemented weekly assignments as interactive Java applets rather than as drawings on paper. An example of a typical assignment was:

> "1. Enter 2 points, connect the two points with some static expressive device that emphasizes the points' connectedness. Constraints are: instantaneous refresh, backgrounds limited to white, foreground is black" [Maeda 97b].

Such open-ended assignments were typical and forced students to draw heavily on their creative as well as technical facilities. Assignments were then evaluated in a class forum every Friday afternoon. Far more important than the technical, graphical chops learned about the Java language was the new perspective Maeda instilled on his students about producing interactive artwork on the computer. Maeda was (and is) a strong proponent of the computer as a medium of expression rather than a tool towards the realization of an expression in a more tradition medium (print for example). These sentiments, coupled with a rediscovered familiarity with visual design, played a large and indispensable role in the conception and development of the Stretchable Music systems.

3 The Project

3.1 Evolution of the Design

The Stretchable Music project evolved as a series of design experiments to develop a graphically based interactive music system. Contrary to many engineering design situations, there was no clear cut goal or problem that drove the development of these systems. Rather, the problems being addressed evolved right along with the designs that attempted to solve them. This process has culminated not only with a finished interactive music system, but also with a fresh perspective on interactive music in general. Hence a great part of the research described in this section involves the search for the correct problem as well as the attempt to answer that problem with a concrete software system.

3.1.1 The Basic Directions

Though the specific problems being addressed by these projects may have evolved during the research period of this thesis, several basic directions remained constant throughout the development. Each of these directions stems from what the author perceived to be a fundamental problem with electronic music. As with many aesthetic dilemmas, that these issues are reflected in each instantiation of the project from a different perspective suggests there is no single right answer to any of them but merely different approaches.

3.1.1.1 Communication to an Audience

Electronic music holds a unique place in contemporary musical culture that in many ways is defined by its lack of definition. When you go to a concert of chamber music or choral music you have a pretty good idea before you arrive of what you are going to see on stage. This is often not the case for an electronic music concert where instruments and interfaces are many times home-built or experimental. If you go to ten different electronic music concerts, the chances are high that you will see at least ten completely different ways to control music on stage, ranging anywhere from someone pressing play on a tape player to a performer with electrodes stuck to his head measuring minute changes in electrical activity in his brain.

But even more diverse than the range of interface devices you may encounter will be the multitude of sounds that emanate from the loud speakers in the hall. (The one and often frustrating constant in the “anything goes” world of electronic music is the nearly ubiquitous distribution medium of electronic amplification and loud speakers.) Thanks to considerable advancement in modern sound synthesis techniques, electronic and computer instruments have a nearly limitless palette of sounds at their convenient disposal.

Computers can synthesize strikingly faithful sonic reproductions of many acoustic instruments or completely new, other-worldly sounds that could be constructed in no other way.

Yet as an unfortunate side effect of the diversity of interfaces and sounds available to electronic instruments, audiences are increasingly confused, and sometimes alienated, by simply not understanding what the heck the performer is doing on stage to produce the music they are hearing. Among the many virtues of traditional acoustic instruments is the implicit understanding that they carry with them in the greater musical culture of the audience. Audiences understand what Carlos Santana is doing on stage when he plays the guitar, not just because of the direct visual connection between his actions on the instrument and the sound it makes (pluck string, hear twang), but because they've grown up watching people play the guitar and have an implicit understanding of what to look and listen for. Without this implicit understanding of electronic instruments in an audience, it's no wonder they're confused at electronic music concerts.

Thus one of the goals of this project was to figure out how to build a stronger bridge of understanding between the performer on an electronic instrument and the audience watching him. There are many approaches one can take to this problem; the path chosen for this research was to explore the potential of computer graphics to aid with this process. Hence, in all of the systems that will be described later in this section, the role of the graphics is not just to provide a control interface for a user playing the system, but to make meaningful connections between the controls of the system and the music being produced to someone who is just watching. Consequently, it was important that the graphics made an aesthetic connection to the musical control they represented as well as a functional one.

3.1.1.2 A Control Interface for Popular Electronic Music

Another driving motivation behind this thesis was the author's desire to find a suitable method to perform new forms popular electronic music live. In some ways this challenge is more difficult than designing electronic instruments for more traditional forms of music because there is no performance heritage to rely upon. Many of the styles in this new form of popular music, including techno, house music, dub, trip hop, and drum 'n bass, were conceived completely within the walls of electronic studios with no intention for live performance (aside from being mixed together by "live" DJs as static recordings). Hence, unlike with the design of new instruments for a traditional form such as free jazz improvisation, there are no convenient instrumental models to use as starting points. In fact, most of the well established, imitative electronic instrument interfaces, such as keyboards or guitar controllers, do a particularly poor job at manipulating the kinds of controls that are relevant in these new styles of music (mainly continuous control). In what little live performance there currently is of popular electronic music, artists often employ dated, unwieldy musical interfaces that kludge together vast arrays of knobs and buttons to control their performances.

Thus each of the projects described later in this chapter is also an attempt to address the issue of live performance of popular electronic music from a control stand point. Specifically, they address the diverse nature of the musical content produced in popular electronic music, and the lack of an adequate generalized control interface to deal with it. The systems test the idea that musical control mechanisms (in a computational context) should be tightly bound to the musical content they are meant to control. By providing a wide range of these content-based control mechanisms to a performer simultaneously, a music system might offer relevant and precise musical control over many different types of musical content through a consistent meta-interface (which in case of Stretchable Music is the screen and the mouse). Similarly, because the performance of popular electronic music has little heritage, the graphical design of these interfaces should owe nothing to encumbering historical precedents. The interfaces used to control this music should look as weird, strange, unworldly, and/or exploratory as the music is itself sounds. They should be functional and fun, but most of all, at home in the medium in which the music they represent was produced (on the computer).

3.1.1.3 A System for Experts and Amateurs

One of the main challenges in designing these interactive music systems was deciding for whom exactly they were to be intended. Conventional wisdom suggests that systems designed for experts will simply have too many controls for a novice to understand without becoming frustrated. Likewise, such wisdom continues that systems designed for novices will not offer experts the precise level of control they demand for performance applications. Usually these extremes are deemed mutually exclusive and visualized as the opposite ends of an axis named something like “skill level of intended user.”

Undaunted by conventional wisdom (or perhaps with reckless disregard for common sense, depending on how you look at it), this author strove to have his cake and eat it too. The systems later described in this chapter also represent an attempt to design an interface that is accessible to novice users but powerful enough to satisfy the demands of experts. Such an interface should have a linear learning curve that lets amateur users gain some satisfying results at the bottom but leaves plenty of room for improvement to an expert level at the top. This type of interface would conceivably make the process of learning to play the system an enjoyable one all the way along, ideally blurring to irrelevance the distinction between the definitions of a tool for an expert and an experience for an amateur. Though such an ideal is undeniably difficult to achieve, many of the techniques employed by these systems may shed useful light on the correct path towards this destination.

3.1.2 The Iterative Design Process

Though it may go without saying, the iterative design process played an important role in the development of the Stretchable Music systems. On one level, the individual components of a system were designed through successive iterations of "compile and test, compile and test…." In addition to fine tuning the design of these components, this process left open the possibility for "mistakes" to be discovered and exploited as aesthetic features (especially in the graphic algorithms). On a broader level, each of systems themselves represented an iteration in the evolutionary design process described at the beginning of this chapter. Hence, through both the flexibility and the refinement provided by the iterative design process, the author was able to balance the technical demands of the Stretchable Music project with creative inspiration and serendipitous discovery.

3.2 The System Basics

All three interactive music systems developed during the Stretchable Music project utilize similar elements and share the same basic structure. Each uses a graphical interface to both represent the state of the system and to act as a control interface. Each uses a mouse or some type of pointing device to allow users to interact with the system. Finally, each of the systems is essentially an elaborate MIDI controller that sends information out to a rack of synthesizers that create the sound. This section covers a few of the key concepts used throughout the project as and gives a basic over view of the setup in with which the project was developed and shown.

3.2.1 The Concept of a Music Object

Probably the most central concept throughout the entire system is that of a music object, an animated graphical object that represents a particular layer of the music being played by the system. For example, a green, globular amoeba crawling across the screen might represent a bass line playing on one of the synthesizers' voices. The animation for the music object acts as both a representation of the state of the bass part and a control interface for users to change the way the bass object plays its part. Continuing the example of the "unicellular" bass part, every time a note is played, the cytoplasm in the center of the amoeba flashes white; when extra accented notes are triggered, they appear as orange bands across the boundary of the object. Users interact with music objects by grabbing them and stretching them with a pointing device such as a mouse. These actions can cause the music object to add layers to its music, modify the timbre of its sound, or change its output all together. For instance, when a user grabs the top edge of the bass object described above, the bass line switches into an arpeggiation that can be dragged through several registers with increasing rhythmic density (quarter notes, eighth notes, and finally 16th notes).

The interface for each object is custom tailored to represent the particular musical content of that object. Likewise, the musical controls of an object are designed to be especially relevant to that object's music and musical role in the system. Even the object's behaviors and simulated physical properties are intended to reinforce the aesthetics of the musical content represented by the object. In some sense, these music objects were conceived from a holistic viewpoint where their control interface, visual representation, musical content, and other properties were all meant to be an aesthetic extension of the same abstract musical idea (or at least live in relative harmony together). This vision is reflected, as will be discussed later in this section, by the software structure of the system which packages graphics, music controls, behaviors, and music all together in compact software objects conveniently named music objects.

3.2.1.1 Roots in a Hyperinstrument

It is worth noting that in many ways the concept of a music object was inspired by ideas related to the design of Tod Machover's Hyperinstruments, discussed above. One of the most crucial parts of the software structure of one of Machover's Hyperinstruments was a section called a mode. In this part of the software, specific mappings were made between basic input controls available on the hyperinstrument, such as sensor values coming from an augmented cello, and musical outputs. For example, as the cellist bowed more aggressively, an aggressive sequence of synthesizer tones might be triggered to augment the cellist's line. Generally, these mappings changed as the piece progressed through different sections, thus updating the musical controls available to the player to relevant musical material for those particular sections. Usually these modes are triggered sequentially; the performance instruments in the original version of the *Brain Opera* had 26 modes that spanned 42 minutes of music.

This idea of combining musically relevant material and mappings into one software object was extended in the Stretchable Music project to become a music object. However, these specialized objects would no longer run only in sequence. For this project the author envisioned multiple music objects all running simultaneously that could each be controlled actively by performer, or left in some neutral state to play their musical material alone. Like the Hyperinstrument modes, these music objects would contain mappings between user input and specific musical controls relevant to their content. Unlike the original modes, however, each of these objects would employ their own custom interface for the user to control. By making all of the interfaces graphical (and thus virtual), the author was freed to explore the aesthetic aspects of the interfaces as well as their control properties. With multiple music objects all running at once, the author hoped to create a more lively environment for electronic music playback and control. The music he envisioned coming from such a system would sound more like it was produced by an improvising rock band such as the Grateful Dead than a sequencer.

3.2.2 Mouse Interaction

As mentioned previously, the primary modes of user interface to all of the systems in the project were through pointing devices such as a mouse. This level of interface provided a stable level of meta-interface between the user and the custom graphical interfaces of the individual music objects. Users may not have known how to control a given object when it first arrived on the screen, but they could be sure that by using the standard clicking and dragging controls offered by the mouse on the object, they would soon learn its secrets. In fact, it was quite important in the design of these systems to make every piece of graphics displayed on the screen active to mouse control in some way. That way the user's experience would be enriched by a more consistent exploration experience as well as by the more obvious musical benefits offered by the musical controls of the system.

3.2.2.1 Why the Mouse?

The author chose to use the mouse as the primary input device not simply out of laziness, but to focus user attention more closely on the content of the system. Through his experience developing interactive instruments with the Opera of the Future Group, the author has informally observed that musical content and mappings often "play second fiddle" to a physical input device. Many times these more aesthetic features of the system were left to an eleventh hour, all-night-before-the-demo effort to show the capabilities of the new type of interface device. On the other hand, even when substantial time could be dedicated to the music software of the system, the sheer novelty of some physical controllers often overshadowed any innovation present in the content or control of the system. (This imbalance was especially present in some of the electric field instruments described in chapter two, where users were so overwhelmed by the novelty of controlling music with their gesture, that they could easily ignore the musical particulars of the mappings). Hence the author's choice of mouse and screen for the primary mode interface for the project reflected his desire to highlight the musical content as the sole interesting part about the system. This was also a challenge to himself to develop higher quality content, for without an interesting physical controller to hide behind, the musical content and control of the system would be judged as the sole source of the success or failure of the system. More detailed analysis in chapter four will discuss how well the approach actually worked.

3.2.2.2 Stretching as a Control Interface

When designing a mouse-driven, graphical interface for an interactive music system, the stretching of abstract, graphical objects is probably not the first mode of user control that leaps to a designer's mind. However this mode of interaction was deemed appropriate for two main reasons. First, it allows a common type of user interaction, i.e. stretching, to be applied across a wide array if different specific graphical interfaces. As mentioned above, a user might not know what the controls are for a specific object but they always can count on stretching some part of it to find out. Secondly, it enables the designer to get lots of continuous control into the system in an aesthetically engaging way. Continuous control of timbral parameters is extremely important in the idioms of popular electronic music chosen for these systems. Stretching turned out to be a nice visual metaphor for these controls because it allowed graphics to smoothly transition from one state to another, closely mimicking the perceived change in sound.

3.2.3 A Word About the Music

The choice of musical style for these systems, namely techno and trip hop, was more than an arbitrary aesthetic decision on the part of the designer. One of the original criteria for the system was that it make what the author considers to be really good, professional quality music to which someone would want to listen. If swing jazz or blue-grass had been chosen as the primary style, it would have been much more difficult to make the musical output of the system match a user's expectation of professional quality music. This is mainly because swing jazz and blue-grass, along with most other styles of music, are generally performed by human beings. Designing computer systems to play music in real-time with the subtlety and emotion of a human performer is an extremely difficult problem (which has been worked on by many smart people in the field of computer music, and in the humble opinion of the author, is not anywhere near a satisfying solution). If your interactive music system is clearly trying to sound like a jazz piano player and failing, it will inevitably be viewed as a toy.

Popular electronic music, on the other hand, is generally not played in real-time by human performers, but as described above, is often pieced together carefully off-line on a computer. This should not necessarily suggest that popular electronic music lacks the degree expressiveness enjoyed by more traditional forms of music. Rather, emotion is expressed in this style of music through different channels and devices which, importantly, are easy to manipulate on a computer. Popular electronic music is completely at home in the medium of computation and thus is a natural choice of style for a computer based interactive music system. Users will be pleased to hear music coming out of their controls through the system that could pass in any club or on the radio.

3.2.4 The Setup

The hardware for the Stretchable Music systems can be divided into three major components. The fun begins on the computer where a user interacts with the graphics of sys-

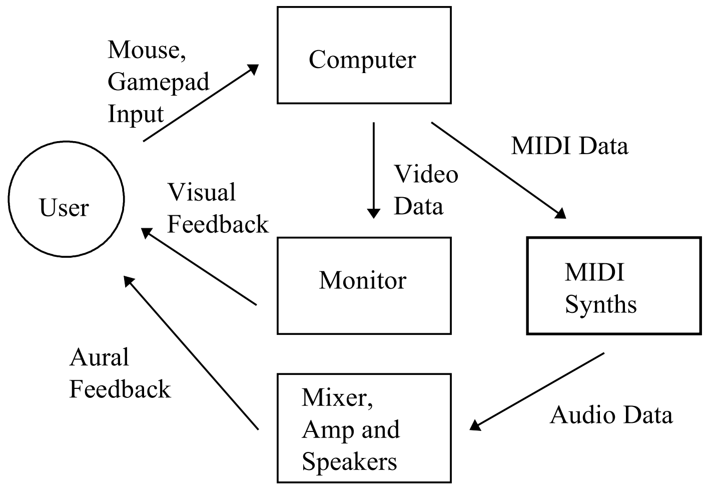

Figure 1. Data Flow Diagram

tem, using a mouse, to produce MIDI events that represent notes and various musical controls. These MIDI events are passed as messages through a MIDI cable to two racks of external synthesizers and samplers. These devices translate the MIDI messages into notes and timbral controls that are played out as audio. Audio from all of the synthesizers and samplers is collected in a mixer at the third stage and merged down to a single stereo channel that is sent to an amplifier and speakers.

The software system was developed under Windows NT Workstation 4.0 using standard tools available with the Microsoft's Visual C++ compiler (discussed in more detail below). The development platform was a 266Mhz Pentium with 128mb of ram and a hi-res video card with 8mb of on board video ram. Because of the robustness of the operating system, and the sheer power of the development machine, the author was freed, for the most part, from worrying about speed and optimization issues during the development period. This of course left more time for creative exploration, which is the true thrust of this thesis. For reasons of personal convenience, most of the musical content in the project was put together using an antiquated version of Opcode's Studio Vision MIDI sequencer on an Apple Powerbook 520c. Finally, it's worth noting that the synthesizers and samplers used in the project were specially chosen for their real-time, timbral control ability and high quality of sound. A project such as this one could simply not have been realized on a General MIDI synthesizer with a limited sonic pallet. Aggressive timbral control was paramount to the design and aesthetic of the project at every step. A more detailed list of equipment and software used is included below.

Equipment
266 MHz Pentium II Computer with 128 Mb of Ram
21" NEC MultiSync Monitor
2 Gravis GamePad Pro controllers
Yamaha Promix 01 Digital Mixer
Clavia Nord II Rack
Novation BassStation
Novation DrumStation
Yamaha VL1-m
Roland R70
Roland MC303
Digidesign SampleCell II Card (in a Mac IIci)
Ibanez Analog Delay 202

Table 1. List of Equipment

3.3 Discussion of the Design

The following discussion traces the design of three and a half versions of Stretchable Music over a period of about a year. (The half-version refers to an ambitious version of the system that was not completed with in the time frame of this thesis). Each system will be described both in overview and in terms of detailed design decisions made during the development of that system. A comprehensive technical discussion of the software for each system will be saved for the following section on general software infrastructures. However, some technical details that are particularly relevant to the aesthetic effect of a part of a system will be discussed in the context of design decisions made for that system.

3.3.1 Version 1: Spring 1997

3.3.1.1 System Overview

The first version of Stretchable Music was developed during the early part of the spring of 1997 at the Media Lab. In this system, different layers of a simple, three-part techno arrangement were represented by animated graphical objects on the screen. Each of the objects had four handles which a user could grab with a mouse and stretch to change various aspects about how the object played its musical part. The handles, which were simply green circles, would appear only when the user moved the mouse into the immediate vicinity of the object.

Each of the handles on an object controlled an abstract parameter that was measured as the distance from the grabbed handle to the geometric mean of all four handles of the object. These parameters then were mapped directly to graphical and musical parameters of the object. However, because a change in position of one of the handles affected the position of the geometric mean, a user could change all four handle distances at once by simply stretching one handle. The other three handles were usually not affected so much as to interfere with the user's understanding of the mapping of the grabbed handle; they simply added background complexity to the musical response of the object. This interdependency between musical controls was deemed one of the most interesting points of the original system.

When a user dragged a handle with the mouse cursor, the handle followed the user's cursor with a simulated force function of $f(x) = -kx$. Hence, the handle acted more like it was being dragged by a rubber band attached to the mouse cursor than by the cursor itself. Users could take advantage of this feature to fling handles around the screen causing wild gyrations in the visual and musical parameters mapped to the object. However, as soon as a user let go of a handle, the handle would stick to its current position. In this way musical parameters could be tweaked and set to desired values.

Using the right mouse button, users were also able to rotate objects about their centers to achieve more favorable orientations for the manipulation of their parameters. Further, by dragging objects by a fifth, center handle located at the geometric mean of the objects other four handles, users could move objects to other locations on the screen. Neither rotation nor translation had any graphical or musical effect on the objects; only the relative distances of the handles to the center of an object had any parametric effect.

Users also could turn objects on or off by right-clicking on the fifth central handle of the object. When an object was clicked off, it would appear as a static icon in a section of the lower right hand corner of the screen affectionately dubbed the "penalty box." The experience began with all three objects in the "penalty box," not playing any music. The user could then bring each of them on to the screen one at a time to explore their functionality.

3.3.1.2 The Objects

There were only three objects in first version of the Stretchable Music system. The objects represented the bass part, keyboard part, and drum part of a simple four bar phrase that was infinitely looped.

3.3.1.2.1 The Bass Object

The bass object was meant to be driving and "psychedelic." This aesthetic drove the choice of colors (reds and purples), the selection of a dense rhythmic arpeggiated musical motif, the design of a gyrating linear form, and the utilization of pronounced timbral controls. Graphically, the object was represented by four gyrating red strips as shown in the illustration above. The strips were actually Bèzier curves with fixed end points at each of the four the handles and invisible, rotating curvature nodes causing the strips to gyrate around the fixed ends. The animation had three graphical parameters: speed of gyration, radius of gyration and thickness of the curves. These parameters were mapped to three of the four handles of the object.

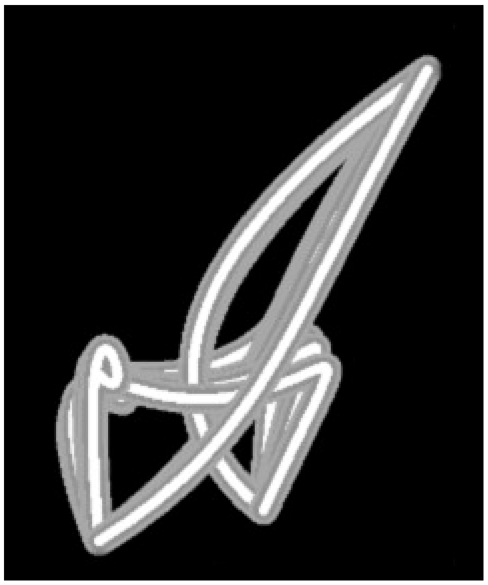

Figure 2. The Bass Object

The object played a simple four note arpeggio in straight 16th notes of a version of a C dominant-7 chord with a suspended 4th. The notes were C, F, G, and Bb in the register of C3. This simple motif was modified by three timbral parameters and a transposition parameter attached to the four handles of the bass object. The timbral parameters were filter cutoff frequency, filter resonance amount, amplitude envelope decay time. The transposition parameter allowed the chord to be transposed through eight octaves from C0 to C10.

The mappings of visual parameters to musical ones were not literal at all. The same handle that controlled the filter resonance amount on the synth, also controlled the amount of gyration of the curves. Similarly, decay time was mapped to gyration speed and filter cutoff frequency was mapped to the line thickness of the curves. But the abstract sense of the aesthetic of the object seemed to remain intact, and perhaps was even amplified with these mappings. Basic correlation, such as the more you pulled a handle out, the crazier things got both musically and visually, were consistent throughout the mappings. For example, as the filter resonance parameter increased towards its maximum value the synth sounds would begin to whine and chirp in an aggressive fashion. At the same time the gyration parameter would be increasing towards its maximum causing the object to become increasingly twisted and convoluted. Also, because, as described above, the handle parameters are interdependent on one and other, pulling this one handle towards a maximum would cause all of the other parameters to become more excited, forcing the graphics and sound to even higher psychedelic altitudes.

3.3.1.2.2 The Keyboard Object

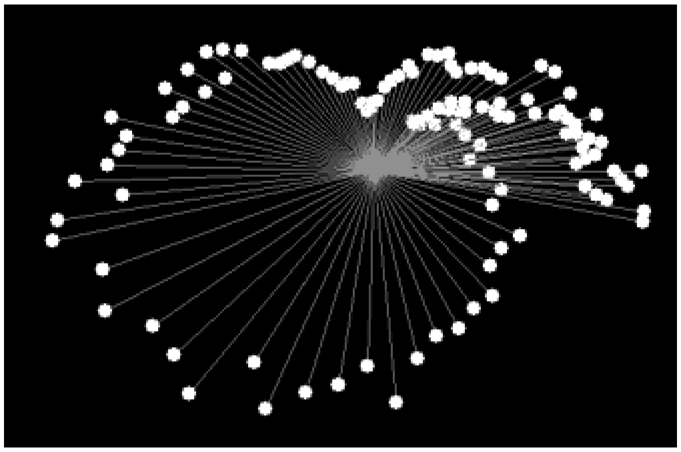
Figure 3. The Keyboard Object

The keyboard object was supposed to be cool and ethereal. Musically it represented two chord progressions that could be mixed together or played separately. The object's handles controlled filter resonance and cutoff on each of the two progressions, and the entire riff was sent through a healthy amount of delay. Graphically, the object was represented by a starburst of white nodes on the end of blue spines emanating from a central point (see illustration above). The size of the nodes, the shade of the nodes, the number of nodes, and the arrangement of the nodes along a fan shaped curve were all parameters that could be changed by stretching the object's handles. Moreover, the size of the starburst (length of the spines) was controlled simply by the extent to which all of the handles were stretched; the animation roughly filled in the space between the handles.

Again the specific mappings of graphical and musical parameters in this object were abstract. The fact that node density and filter cutoff of synth layer two were mapped to the same handle was less important in the design than giving the user two aesthetically comparable spaces, musical and graphical, to explore. In this way, one could discover "sweet spots" in the animation and musical controls where the graphics and sound line up notably well. One particularly nice spot occurred when the parameter controlling the contour of the nodes was adjusted so that the nodes all fell into line along a fan shape curve. This setting lined up with a poignant adjustment on the filter setting on one of the chord progressions to create a very pleasing synthesis of visual and audio stimuli. The discovery of such sweet spots was considered one of the strengths of this object's design. Nonetheless, each of the controls still followed the general design guidelines described above where the further a handle is stretched, the more extreme the musical and graphical results became.

3.3.1.2.3 The Drums Object

The drums object played the role of pulsating time keeper in the system. Though rhythm was clearly present with only the other two objects on the screen, the drums object really hammered the beat home with aggressive percussive sounds and a relentless kick drum on every quarter note. Musically there were four parts to this object that could be mixed together: the driving kick drum on the beat, a syncopated snare layer with hits on two and four, a hihat layer on the off beat, and an additional breakbeat. The kick drum was ever present, but could be emphasized with the addition extra bass and decay, along with subtle overdrive. Similarly, the breakbeat ran through a filter with cutoff and resonance control that could be adjusted in addition to level controls.

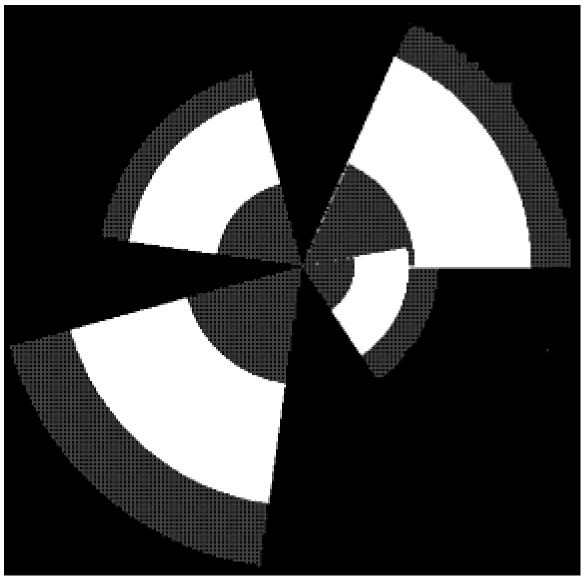
Figure 4. The Drums Object

The drums object was visually represented by a four pronged "X" reminiscent of the familiar nuclear hazard symbol (see illustration above). Each of the legs of the "X" was actually a pie shaped wedge whose width could be adjusted as a graphical parameter. A yellow band pulsed along each leg forming an expanding and contracting ring that synched to the main beat. The thickness of this band was also a parameter that was linked to one of the object's handles.

Keeping in line with the design of the other objects, the drums object’s mappings are designed to be abstract, yet correlated in terms of activity. Yet more than the others, however, the visual design of this object strongly reinforces its musical role in the system. The pulsating yellow band expanding and contracting along the legs of the object nicely reinforces the 4/4 beat of the kick drum and solidifies the rhythmic role of this object visually. Further, the pie shaped legs of the X and the connotations of nuclear hazard add expanding energy to the animation. This energy is juxtaposed with the rather sedate dark reddish brown color of the legs themselves, which provides stability which fits nicely with the object’s role of rhythmic foundation. Unfortunately, the other rhythmic layers aren’t as well represented in the graphics. Their role is to round things out musically and they might have been mentioned more prominently in the visual domain.

3.3.2 Version 2: Fall 1997

3.3.2.1 System Overview

The second version of Stretchable Music was developed in part to provide musical accompaniment to the “Wearable Computing Fashion Show” held at the Media Lab in the fall of 1997. This system differed from its predecessor in many important ways. To begin with, it was conceived as more of a complete song than the collection of four bar riffs that comprised the previous version of the system; in fact the piece was named “A Post Nuclear Lounge Adventure.” The new system had six different music objects that appeared and disappeared from the screen according to a score running in the background. In this way a larger scale time structure was imposed by the designer on the musical progression of the system. It was thought that this structure would help the whole system seem less static; music, after all, is considered a medium that flows with time rather than one that is frozen in it. Further, the addition of this feature to the system was an important step along the way to considering the system a self contained musical expression or piece rather than a tool with which to make music.

The objects in this new system had also evolved significantly from their three predecessors in the original version of Stretchable Music. Besides there being simply more of them, the individual objects were graphically far more diverse. The green circular handles that allowed user’s to stretch objects in the previous version were sacrificed for active surfaces right on the graphical animations themselves. Now just about any part of an object that was graphically represented on the screen could be grabbed and stretched. Moreover, objects generally had more personality than the abstract animations of the first version. There were fish, amoebas, worms, and ghostly hoops among others that crawled across the screen in an colorful, ethereal ballet. At times the system seemed like a window into the petri dish of a composer's brain with melodies, rhythms, and counter lines all swimming about in a colorful musical witch's brew, waiting to be explored.

The musical controls attached to objects' stretchable on-screen graphic representations were more subtle in this version of the system. Some of the objects only had one or two controls total, compared to the previous system's four controls per object. Yet these subtle musical controls were deemed more relevant in their particular musical context; each was designed to fit especially well to the musical role of the object. Melodic objects allowed timbral shaping and articulation of their melodies while rhythmic objects allowed various layers of the rhythm to be turned on and off as well as mixed together. Some of the objects utilized the simulation of physical properties to enhance the aesthetic effect of their controls. The main_bass object (described in greater detail below) would oscillate abundantly when stretched and released, causing timbral parameters attached to its perimeter to gyrate with the oscillations, and giving the entire object the feeling of jelly, both sonically and visually.

Finally, a new, higher level of user control was introduced in the second version of Stretchable Music through the mechanism of tools. By pressing and holding the right mouse button a user could sprinkle pixie dust about the screen on various objects. When sprinkled with pixie dust, an object would raise its level of musical activity, usually by mixing in extra accented notes or playing a more complex musical line. The idea originally was to have several different types of tools that could affect objects in different, high-level ways. User's would be able to add tools to the "musical soup" to increase activity, calm things down, make sounds more or less aggressive, or even adjust tempo or harmony. Because these tools would be able to affect many objects at once, they would give the user a control relationship to the system similar to that of a conductor to an orchestra, enabling him to communicate broad strokes of emotion to all of the musical elements in the system. Unfortunately, due to time constraints only one tool was implemented and only a few of the objects actually responded to it. But even with this limited implementation the importance of this level of control was clearly demonstrated.

3.3.2.2 The Objects

The main elements of the second version of Stretchable Music were six musical objects and a single tool that could be used to control higher level parameters on some of the objects. Each of these objects and the tool will be analyzed in the following sections with regards to their musical and visual design. Select comments about the implementation of these elements will be included where appropriate.

3.3.2.2.1 The main_bass Object

One of the most interesting objects in the system was the main_bass object. Visually, this object was represented by a green amoeba-like creature that looked more like an inner-tube than a unicellular organism (see illustration). As its name implies, main_bass supplied the rhythmic and harmonic foundation for the piece with mellow samba-esque bass line that oscillated between the root and the fifth of the key of C with a comfortably swung gate. Every note played by the object was visually re-enforced with a white flash of the interior of the object. When sprinkled with the angle dust tool, extra notes augmenting the main riff would appear as orange bands on the perimeter of the object. Left to its own designs, main_bass would crawl about the screen by expanding and compressing its body in the direction of motion, stopping to change its direction only when it encountered an edge of the screen. The object would, however, stop and patiently sit still while a user was interacting with it using the mouse pointer.

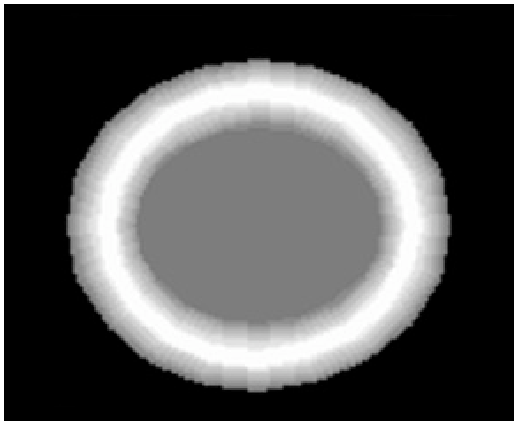
Figure 5. The main_bass Object

User's could manipulate main_bass by grabbing the rim of it's inner-tube and stretching it with the mouse. There were three separate musical parameters that were attached to this inner-tube perimeter whose values were measured as the distance from the stretched portion of the object to its resting position. The parameters were filter cutoff amount, arpeggiation control, and FM control attached to the right, top, and left most points on the perimeter respectively. The filter control allowed the user to open up a low pass, resonant filter on the samba bass line making the sound brighter; the FM parameter caused the sound to oscillate wildly in pitch creating some very weird sounding bass lines. The arpeggiation control caused the object to switch into an arpeggiation of the underlying chord progression of the bass line. The further the arpeggiation control was stretched, the more rhythmically dense the arpeggio would become (quarter notes to eight notes to 16th notes) and the more the chord would become spread through registers. At its maximum amount, the arpeggiation control achieved a nice effect of an arpeggiated chord spanning five octaves running on 16th notes with just a touch of portamento slide between the notes. Users could also affect more than one control at once by grabbing in-between the control points on the top, left and right. Stretching object there would affect the two nearest controls to an amount proportional to the distances from the grab point to the neighboring control points. In this way, some sense of interdependency was incorporated into the musical control mechanism for this object.

The most exciting thing about the main_bass object was the way it would undulate wildly when a user stretched and let go of the inner-tube perimeter. The tube was actually made up of 20 points connected by a series of shaded lines to give it depth and roundness. These points were also connected to each other and to invisible template points by a series of simulated virtual springs. When a user grabbed and stretched one of the points, the virtual springs would pull neighboring points away from their template points in the direction of the stretched point. Upon release of the grabbed point, the tension in the springs between neighboring points and the template points would cause the stretched portions of the tube to snap back into place, often overshooting their stable position and inducing wild oscillations throughout the object. Both the tension in the spring function (parameter k in the equation $F(x) = -kx$) and the drag on the entire system could be adjusted to give the system the most pleasing visual behavior. As an added bonus, because the musical parameters were attached to position of the three control points along the tube, oscillations induced in the tube would cause corresponding gyrations in the sonic controls. As a result, the wild visual undulations that a user could easily induce in the main_bass object were usually accompanied by equally pronounced filter sweeps, arpeggiatied notes, and strange FM squawks, giving the entire object a very organic feel. This unpredictability and interconnection between sonic and visual parameters was considered one of the strongest points of the object and a highlight in the piece as a whole.

3.3.2.2.2 The ether_synth Object

Visually, the ether_synth object wasn't really a single object at all, but comprised a school of six blue fish that slowly faded in and out of the black background of the system. The fish represented an ethereal four bar keyboard part that was meant to fill in around the other musical elements and to add atmosphere and headroom to the piece sonically. The riff was basically a spacey arpeggio in a high register of the keyboard with almost bell-like pitched percussive sounds with a lot of sustain and a touch of analog delay. One of the fish would subtlely flash every time a note was played making the entire school twinkle like a starry night's sky as the riff played through them. As the piece progressed, the school would wander about the screen with varying levels of interest in its new destination among its constituents.

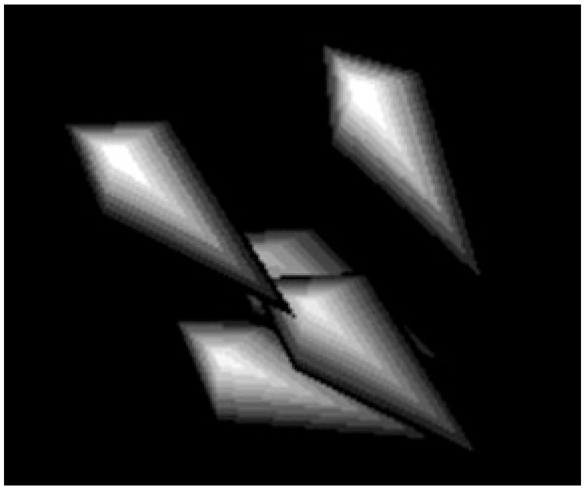
Figure 6. The ether_synth Object

Users could grab individual fish with the mouse pointer and drag them away from the rest of the school. As long as a user held on to an individual fish, an additional melody in an even higher register would play out with the same ethereal synth voice. Each of the fish had a different melody associated with it giving the user six different melodic ornaments from which to choose. Musically, these melodies were supposed to be used as fills to ornament the main keyboard riff. Because each of these additional melodies were pretty soft and sparse, this control was considered among the most subtle offered by the system and was often ignored by less musically inclined users. A more pronounced keyboard counter melody was available, however, if the users sprinkled the angle_dust tool on the school of fish.

The algorithm that caused the fish to cluster together in a school and swim towards a common destination deserves special mention because of its success in creating a life like simulation of a school of fish. Essentially all of the fish were attracted to the same destination on the screen by loose, virtual rubber bands attached to their noses. The force function used in this simulation was very similar to the one used in the tube simulation of the main_bass object discussed above. By pulling on their noses towards this destination the rubber bands not only caused the fish to stay together in a cluster but forced them all to point and swim in the same direction (an important attribute of a school of real fish). The fish were repelled from each other by an additional force that varied as the inverse square of the distance between individual fish. Hence the fish would cluster but would not swim entirely on top of one and other. The interplay between these two force functions actually provided for some surprisingly life like results. Every once in a while a fish would get squished into such an uncomfortable position by its neighbors, that it would shoot out from the school in the opposite direction of the school's destination. This "emergent behavior," was deemed pleasingly similar to observed behavior in real schools of fish and considered one of the best points about the visual appeal of this object.

3.3.2.2.3 The breakbeat Object

The breakbeat object represented the rhythmic focus of the piece. It carried the main drum riff, a chilled-out trap kit back-beat with syncopated brush snare-taps on and around the two and the four beats, as well as additional percussive material including a hihat layer, a shaker layer, a conga layer, and a snare fill layer. The musical effect was supposed to be swinging and kind of loungey, bouncing off the straighter and more predictable rhythm of the main bass object. Visually, the object was represented by a purple, hollow pentagon that kicked and jabbed to the beat of the drums (see illustration above). The purple pentagon was chosen to represent this object for several reasons. Because the rhythms represented were all pattern based, the author decided the cyclical nature of a regular polygon would be visually appropriate. However, because all of the rhythms in the object were syncopated or unsymetric in some way, the pentagon, seemed like the most appropriate regular polygon, especially with its interior slightly twisted in one direction (again see illustration above). The

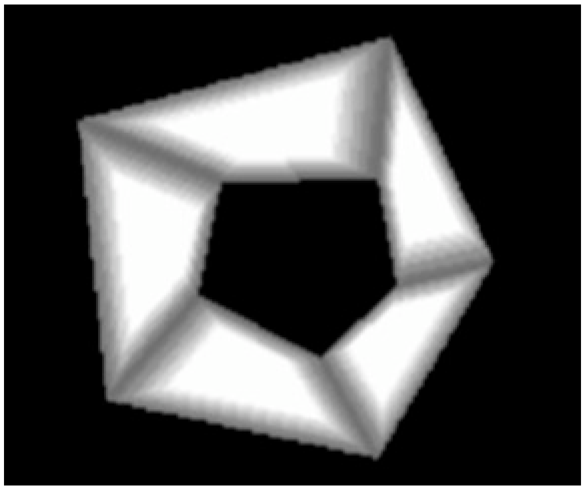
Figure 7. The breakbeat Object

purple color was deemed to be a nice complement to the strong greens, blues, and reds already represented in the piece.

Each side of the pentagon was itself a quadrilateral polygon that represented one layer in the music. Every time a note from that layer would play, the corresponding quad section of the pentagon would flash yellow. These flashes served as indicators to the user regarding which musical layers were on or off. Users could turn various layers on or off simply by double clicking on the corresponding quad sections of the pentagon. Moreover, users could mix between different layers, even if they were turned off, by grabbing the corresponding segments and stretching them away from their rest positions. This allowed the user to add a touch of congas here or a dash of shakers there with out turning on an entire layer. As soon as the user let go of the section, it would quickly return to its original position and mix. The breakbeat object also responded to the angel_dust tool by mixing in a bassy 808 kick drum, hand clap, and off beat hihat on top of its other layers. Further, every eight bars the object would wrench its self free of user control and bounce around the screen to an angular fill of spliced and shortened breakbeat samples.

3.3.2.2.4 The shakuhachi Object

One of the two important melodic roles in the piece was played by the shakuhachi object. In this object, a trail of purple hoops reminiscent of a kite tale swerved and dove in playful figure eights to the rising contour of a breathy eight bar shakuhachi melody. The hoops were chosen as a visual motif because they reminded the author of the shape one's lips take when one whistles to produce a pure, whole sound similar to that of a shakuhachi. Again the purple color of the hoops was chosen to compliment the other colored objects on the screen; the object was meant to be cool and melodic in color (thus the combination of red and blue).

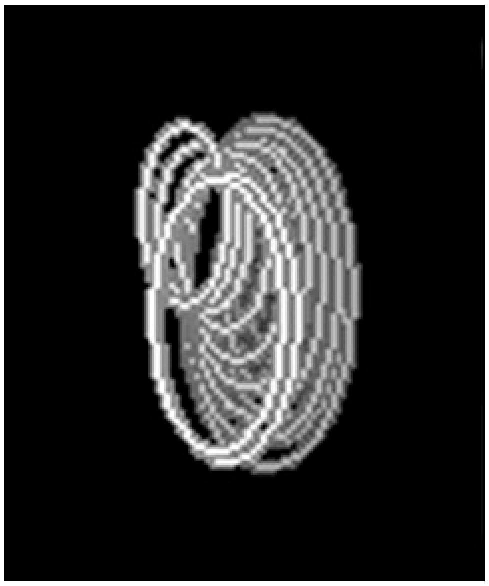
Figure 8. The shakuhachi Object

Sonically, the melody was produced by the shakuhachi patch on a Yamaha VL1-m synthesizer. This device utilizes a synthesis technique known as physical modeling where the wave behavior of air oscillating in a tube is actually simulated by the synthesizer to produce the desired sound. Users can access some of the sonic parameters afforded by this synthesis technique to shape the melody by grabbing and dragging the trail of hoops around the screen. The faster a user dragged the object, the more vibrato would be introduced into the line, culminating at a point where the entire melody was chopped up into short staccato notes that varied as much as three half-steps from the original melody. In addition, if the user dragged the object upwards from the original point where he grabbed it, it would cause the pitch of the melody to shift upwards, fading in an out of various harmonics and breath noise in much the same way as the real instrument. Motion below the original grab point would produce a similar effect by bending the pitch in the opposite direction. One of the most appealing ways to use this object to articulate the shakuhachi melody was to bend it up to specific harmonic break points and then introduce subtle amounts of vibrato to achieve a striking expressive effect.

3.3.2.2.5 The squirm_lead Object

The second lead melodic role in the piece is played by the squirm_lead object. Visualized as glowing red eel winding its way through the viscous black medium of the computer screen, the squirm_lead object represents a sing-songy four bar melody that slides between its notes with a healthy amount of portamento slide and analog delay. Inspired in part by similarly lyrical melodies in the works of Richard D. James and Orbital, this part is meant to act as a counter line to the longer sustained notes of the shakuhachi melody as well as standing on its own two feet as an independent theme. Dark red was chosen as the color for the eel because of its warmth, an intuitive connection made by the author to classic lead analog synths such as the Mini Moog. Similarly, the winding path of the eel was deemed appropriate for this line because of the generous amounts of portamento in the synth voice. In addition, every time a new note was played a bright red stripe would travel quickly down the eel's back, giving it an even more electrified look.

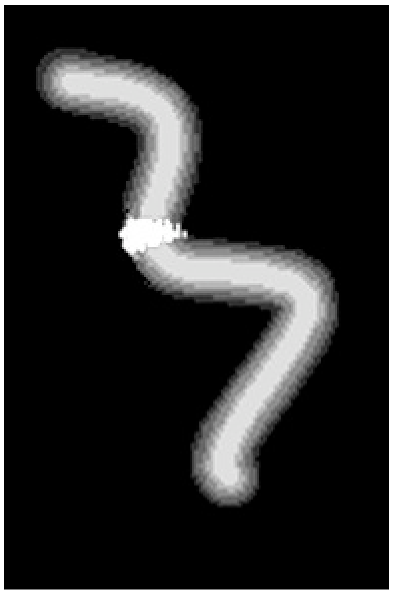
Figure 9. The squirm_lead Object

As the eel wound about the screen (turning only to avoid running off the edge of the monitor), users could grab and stretch it to manipulate filter cutoff and resonance parameters on the synth voice. The hollow square wave synth voice left plenty of upper harmonics for the user to unveil by stretching the eel's back away from the rest of its body. The same motion modulated the filter's resonance amount in proportion to the sine of the distance stretched, allowing several resonance peaks to be encountered as the filter cutoff was slowly opened. The synth used for this melody, a Novation BassStation, was notable for its particularly expressive filter. The author decided that a allowing the user to shape and articulate the melody with the Novation filter as the sole music control for the object was all that was necessary to fulfill the squirm_lead's melodic role in the piece.

Using the same technique as in the main_bass object, the squirm_lead is drawn by connecting a set of points with shaded line segments. Also in a similar manner to the main_bass object, these points are connected by virtual springs to their neighbors and to template nodes which give the object its form. The parameters of these spring loaded connections were tweaked quite differently in the squirm_lead to yield a slower, more damped spring effect. Unlike the main_bass which has a tendency to overshoot itself and induce secondary oscillations, the squirm_lead slinks smoothly back into place after being stretched. This unique physical quality was considered another important reinforcement of the squirm_lead's overall aesthetic.

3.3.2.2.6 The dist_bass Object

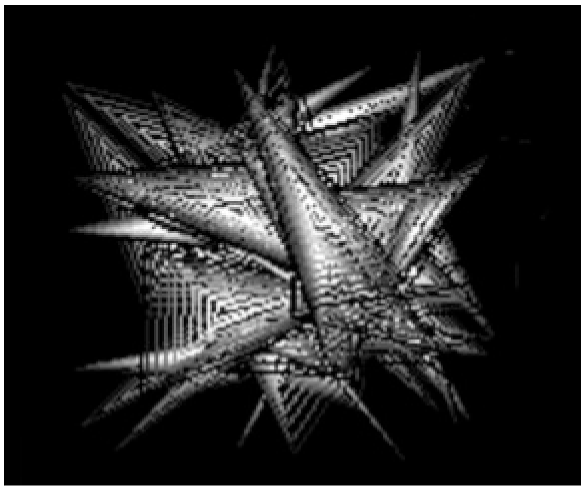
Figure 10. The dist_bass Object

Possibly one of the most visually interesting music objects in the entire piece, the distorted bass or dist_bass object (as it's called in the code) appeared as an undulating ball of mutating triangles that could be stretched out into a dazzling star shaped array of silver spikes. The object represented a thundering distorted bass riff that could be opened up into a squelched and angry rhythmic assault on the comparatively complacent remainder of the piece. Running a saw tooth wave through a highly resonant 24db low-pass filter on the Nord 2 rack module, the object gave the user only one, though admittedly very dynamic, musical control. Grabbing anywhere within the ball of triangles, the user could stretch outward causing a dramatic filter sweep over the bass voice while simultaneously morphing the object into star of spikes.

Again this single control was deemed sufficient for the musical role of this object and dramatic enough to warrant its unitary presence. Further, as the user stretched the object, the object itself would begin to move in the user's direction, picking up speed and eventually overshooting the stretched position. In this way the user could zing the dist_bass object around the screen (it would bounce off the edges of the screen elastically) like a bowling ball attached to a rubber band. The apparent weight that the object had added, in the opinion of the author, to the heaviness and authority of the distorted bass's musical role.

The visual algorithm that produced the undulating ball of triangles and could morph them into a star of spikes is particularly unique and deserves special attention. The algorithm begins with a set of points randomly placed within a specified square area. The points are then organized into an equal number of point triplets (with each point appearing in three different triplets) and connected by line segments to form triangles. Then for each triplet, successive triangles are drawn along a linear interpolation between the original triplet points and the geometric center of the triplet; meanwhile the shade of the triangles is smoothly adjusted from dark gray to white . This has the effect of drawing smaller and brighter triangles inside of the triplets of points, giving some suggestion of depth. It's important to note that none of these triangles are filled; the many layers of outlined triangles drawn one on top of another has the appealing visual aesthetic of a ball of spider webs.

The final and most important touch to make the ball undulate was to rotate all of the points through elliptical orbits of varying eccentricities and speeds around the center of the original, bounding square region. This causes the triangles to go through a variety of unpredictable transformations and gives the dist_bass object its characteristic look. To make the object morph into a star one only needs to linearly interpolate between two of the three points in each triplet and two object-wide common points near the center of the object. This essentially forces all of the triangles to share one side to an adjustable degree, which is used as a parameter to morph the object between a ball and a star. Unfortunately, this algorithm was relatively computationally expensive, mainly due to the time-consuming Windows graphics calls needed to draw the individual triangles. Several combinations of parameters including size, number of points, and number of shades per set of triangles were tried to find the optimum balance between visual complexity and speed.

3.3.2.2.7 The angel_dust Tool

As explained at the beginning of this section, the angel_dust tool was designed as a way for users to affect higher-level parameters on multiple music objects at once. This tool was supposed to make objects more musically active, each in their own way. Objects would catch individual dust particles from the tool and adjust their activity parameters according to the number of hits they had received over a given period of time. Thus the tool had a second or so lag between the time when a user would begin to apply it to an object and the time the object actually responded. Unfortunately, due to time constraints, this functionality was added to only three of the six objects in the system. Further, the system had been originally conceived with more than one type of tool. But because each additional tool required the addition of six additional custom music controls (one for each object), only a bare minimum implementation was done in the form of the angel_dust tool.

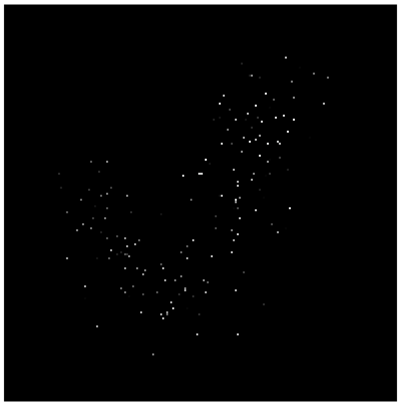
Figure 11. The angel_dust Tool

Though the tool's main function was to modulate other objects, it did nonetheless have a look and a sound of its own, and thus contributed somewhat autonomously to the piece as a whole. Visually the angel_dust was represented (somewhat literally) by twinkling points left in the wake of sweep of the pointer. Using the tool was a lot like painting with a spray can, the more you stayed in one place, the denser the collection of droplets became. However, rather than mapping this animation to rather obvious twinkling bell sounds, the author chose instead to have it represent an ethereal, electronic string voice, slathered in analog delay and sent sailng high above and to the back of the rest of the composition. To strengthen the connection between the many individual grains that visually represented the tool and the string sound, an LFO was added to modulate the volume of the sound at roughly a 16th note level. This had the effect of making the strings sound even more shimmery. The overall volume of the string line was delicately raised as the number of dust particles on the screen grew, achieving a subtle but pleasant mixing effect.

3.3.2.3 The Piece

The piece, "A Post Nuclear Lounge Adventure," begins with the green main_bass object squatting in the middle of the screen playing its samba bass line. After a few notes it begins to crawl with difficulty towards the upper right hand corner of the screen. By then however, the user has most likely grabbed the object and stretched its right side opening up a filter on the bass voice. Soon the an ethereal school fish descend from the upper right hand corner of the screen, bringing with them an atmospheric keyboard arpeggio that outlines a four bar riff. At bar number nine the purple breakbeat objects drops into the mix in the lower right of the screen, punctuating the rhythm with syncopated snare taps and an off beat hihat. By grabbing the lower right hand segment of the breakbeat object a user adds in a short snare fill, then double clicks on an adjoining segment to bring in a conga line. After the several more bars of this groove, the first melody is introduced by the shakuhachi object. The flowing eight bar ascending line is articulated with breathy bends in pitch and mild vibrato by the user who maneuvers the ghostly purple hoops across the screen like a kite. After one rendition of its melody, the shakuhachi disappears and warm sound of the red squirm_lead object ushers in the second melody. Grabbing and stretching the eel like red object, the user opens a distinctive filter on the melodic voice dragging it through several resonance peaks before arriving at a growling round tone. Soon the squirm_lead is rejoined by the shakuhachi as each darts musically between each other's notes. The situation builds with the user exerting more and more frantic control over the objects until they all dissapper at the 32nd bar to leave the lonely main_bass to reassert his rhythmic presence in the break in the piece. The piece then continues through several more such "build-ups," highlighting different combinations of objects along the way. Eventually, the distorted bass object is brought in to add its rumbling, aggressive riff to mix. Users typically react with great pleasure to this new object, dragging its undulating ball of triangles into a spinning start of spikes while releasing an angry resonant filter on the growling bass voice. The piece has no real end; all of objects eventually end up on the screen and belt out their tunes in an infinite reprise. About that time the author usually manually kills the program to give it another run from the beginning.

3.3.3 Version 3: Spring 1998

3.3.3.1 System Overview

During the first week of April, 1998 it was decided that the author should wind up the Stretchable Music project by developing a third and final version of the system to be premiered as a completed public installation at the end of May. In addition, the author was challenged to organize an event around this premier to include other works of interactive art from the lab, and to celebrate the closing of a 13 year chapter of live installations and music in the Media Lab's own Villers Experimental Media Facility ("The Cube"). The event, officially known as *(Art X Media)^{Music}* but affectionately dubbed *Cubestock* by its organizers, proved to be an effective motivator for the author as well as a useful test bed for the new system; it will be discussed in more detail in a later section of this thesis. Most importantly, however, the event forced the author to consolidate his ideas into a finished piece that he would feel comfortable presenting to the public.

Given the extremely short amount of time available for development, the author had to choose carefully the new directions to explore with this piece. The most pressing element missing from the prior two works, in the eyes of the author, was that of definition or identity. The first two systems seemed to straddle many fences while refusing commit to any clear purpose or functionality. Were they performance systems, experiences for amateurs, games, artwork, or the seeds of an interactive installation? Did they represent innovations in the design of interdependent musical controls or a new way of packaging, communicating, and experiencing electronic music? After much thought, the author arrived at the viewpoint outlined at the beginning of this thesis, defining these new interactive music systems as pieces of musical expression themselves, and not simply tools to used express unrelated musical ideas. Hence the goal of the new version of Stretchable Music became to realize this vision of interactive music in full color.

The author decided that the addition of several important features would move the system decisively in line with this new vision of interactive music. Of foremost importance among these features was construction of a rich musical experience that "went somewhere." The previous version of the system addressed this problem by incorporating an object score to allow a composer to essentially script objects' entrances and exits from the screen and the music. However, the system still seemed relatively static because objects always represented the exact same musical part and important musical parameters of the entire system such as tempo or key center never shifted throughout the piece. Thus, the new version of Stretchable Music was designed to have four distinct musical sections, complete with two different tempos and an accelerondo. Each of the sections was defined by its own object score; some even had smaller subsections such as bridges where the tonal center of the piece would modulate to a different key. Users could navigate between these sections by grabbing a special advancement icon that would appear near the end of each section. If a user chose not to advance to the next section, the current section's object score would simply loop back to its beginning again. Assuming a user went through each section only once, the whole piece would last around eight minutes and provide a much greater variety of music than the previous version of the system.

In a second major advance, the new system was designed to allow several users to interact with it at once, via multiple gamepads attached to the gameport of the computer. The software was actually built to handle up to four simultaneous players, but a last minute operating system conflict regrettably knocked that number down to two. The two users of the system were represented on screen by two colored pointers (red and yellow) with which they could grab and stretch objects. Users could grab and stretch the same object at once (sometime yielding special musical effects impossible to achieve with a single user) or stretch different objects. With twice as many musical controls usually being manipulated at any one time, the new system was capable of more variation and depth in its musical output than its predecessor.

In addition to stretching musical objects, users could add in extra layers of keyboards or percussion using specialized tools. Tools, represented by brightly colored spinning icons, would periodically pop up on the screen waiting to be grabbed by a user. By maneuvering his pointer over a tool icon, a user could capture a tool and trigger its extra musical line by depressing the "B" button on his controller. Musically, the tools were supposed to enable a player to solo over top of the rest of the texture. Originally, the user was supposed to be able to control various higher level parameters of the tool's solo (such as rhythmic density or melodic contour) by the path taken with the tool. Unfortunately, due to time constraints, these more complex algorithmic controls were forsaken for easier to implement mixing techniques. A third tool, represented in the system as a purple bomb, allowed users to solo a single music object for up to one bar. When the bomb tool was deployed by a user, all of the music objects except for the one nearest to the explosion would mute for one bar, allowing the bombed object to do a brief solo. Such solo breaks are a common technique in popular electronic music often used to mark the boundary of eight and sixteen bar phrases.

On top of the new features described above, the new version of the system also included more musical objects and a lot more music. Eight completely redesigned music objects were incorporated into the system as well as over 70 two, four, and eight bar sequences of music. The musical direction was jazzier and more aggressive than the previous piece, placing funk-derived acoustic bass lines over hard-driving electronic breakbeats. Some elements, such as the lyrical analog lead voice, were retained, but several new types of sounds including sampled vocals, made their debut in this piece. Similarly, some of the music objects shared their visual design with their predecessors, while others struck out in completely new directions. Especially novel designs were used for the breakbeat object, the vocals object, and the acidbass object. Further, many of the objects utilized bitmaps in various ways to greatly speed their graphical routines.

3.3.3.2 The Objects

3.3.3.2.1 The bubbles Object

The first object to appear in the piece, the bubbles object, helps set up the watery, "primordial soup," sonic texture that dominates the first section of the of the piece. The bubbles object consisted of a collection of green, semi-translucent bubbles of varying sizes (five in all) that seemed to grow out of the center portion of the screen. Periodically, the bubbles would pop with pleasing, analog "blaump": sound that was inspired by similar sounds on AFX Twin's Analogue Bubblebath album. Occasionally, one of the bubbles would contain a sample that would be played out when the bubble popped. Users, could also pop the bubbles by clicking on them with the "A" or grab button of their controller.

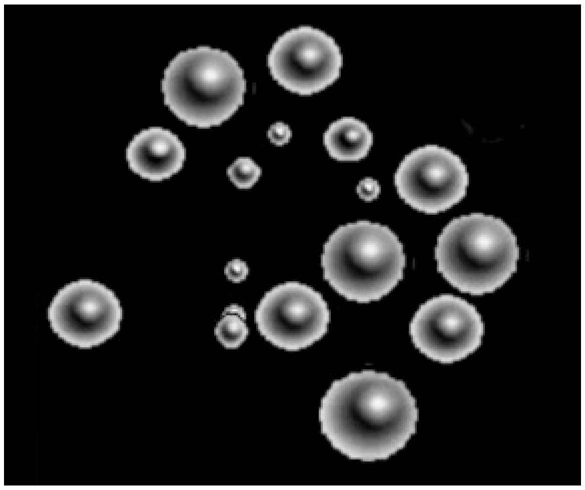
Figure 12. The bubbles Object

The bubbles were actually bitmaps that were drawn ahead of time in a two dimensional paint program. For each of five bubble sizes, a simple five frame "bounce" animation was rendered to allow the bubbles randomly squish every once in a while to provide subtle movement to the visual texture. Using anitaliasing and blend functions in the paint program, the bubbles were given an airbrushed, almost translucent look that contributed to their visual aesthetic. Some of the bubbles featured a small, animated twinkle in their upper right corner. These identified bubbles as the containers of audio samples. When that bubble would burst, a sample from an old TV commercial or other vintage TV quotations (collected from the net) would be played out with considerable analog delay. Animation sequences were also included for the bubbles immediate, pre-burst expansion as well their succeeding explosion.

As new bubbles emerged on the screen, they would push existing bubbles out of the way according to a force function that was based on their size. The bubbles began their tenure on the screen at the smallest of five sizes, but would then undergo a series of randomly determined growth stages until they finally popped. Bubbles were repelled from each other with a force inversely proportional to their distance and size. However, heavy damping was applied to the system, to make the bubble movement more realistic.

3.3.3.2.2 The ether_synth Object

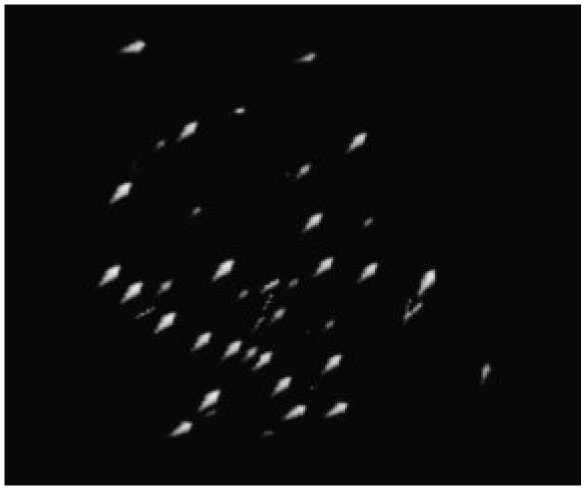
Figure 13. The ether_synth Object

The familiar school of fish from the second version of Stretchable Music returns to this piece greatly multiplied and miniaturized. Instead of consisting of only six fish, the new ether_synth object incorporates 100 fish each about one tenth the size of their predecessors. Musically the new object plays a similar role by providing an ethereal synthesizer texture to the piece, though this time the sound is much more rich and watery (reinforcing the soup idea started with the bubbles object). Users could manipulate the fish by grabbing in their general area and dragging the entire school around. Effects were applied to the synthesizer line depending on which direction the user led the school. To the left they could open up a filter, to the right increase an LFO's contribution to the filter. Dragging the fish up or down would cause the entire sound to be ring modulated, inducing watery gyrations of pitch. Moreover, multiple users could grab and stretch the fish at once activating combinations of these controls.

Rather than relying on computationally expensive Microsoft graphics calls to draw the fish individually, the fish were copied from bitmaps pre-rendered in 24 different orientations. This process proved to be an order of magnitude faster than that used to draw the fish in the second version of the system. With up to 100 fish in a single school, the author could emphasize more vividly the behavior of the body of the school as a whole, rather than focusing on individual fish. Further, the fish were pre-rendered in a green as well as blue, allowing bursts of green to occasionally overtake the school. Every once in a while, individual fish would briefly turn green giving the school a twinkling textured visual effect. The algorithm which causes the fish to cluster into a school is the same that was used in ether_synth object of previous version of the system. Thus the fish would swim about the screen in a similar, wandering fashion, until they were grabbed by a user and stretched into a new visual and sonic shape.

3.3.3.2.3 The vocals Object

One of the most visually interesting objects of the piece, the vocals object was meant to add a delicate human touch to the largely electronic orchestration of the rest of the music. Its main musical layer consisted of a haunting female vocal line pierced with a grid 16th note silences giving it a stuttering, gated edge. The vocals object was graphically represented by a collage of purple squares that faded in and out of the black background as the object moved across the screen. The squares themselves would not move but would simply get brighter as the center of the object approached. This tiled effect is achieved through a quantization algorithm that samples a more continuous, yet invisible, rotating sine wave function. The closer a square is to a point on the function, the brighter it gets. The effect was somewhat akin to moving a light source around behind the panes of a plate glass window.

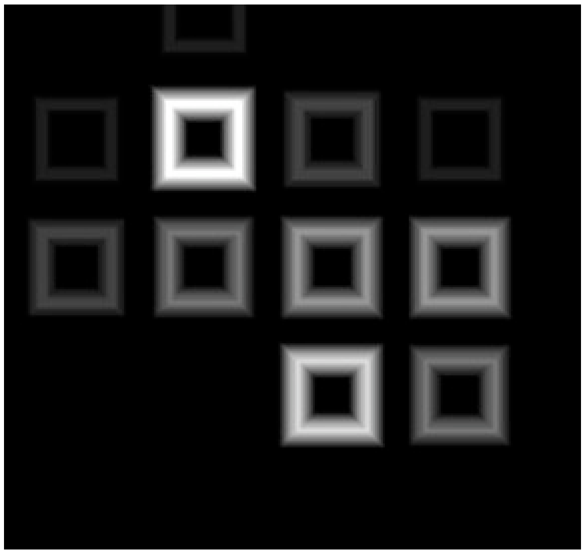
Figure 14. The vocals Object

The music controls for this object differ for different sections of the piece. In the first section, users can mix in the main vocal line as well as mix between three separate vocal lines that more or less fit into a backing rhythmic texture. These additional lines were eliminated from the controls of the object in sections two and four because they were deemed to complex for an already crowded sonic space. In these sections, users simply mix up the main vocal line (which can already be heard faintly in the background) by pulling and dragging the collage of squares around the screen. A little bit of lag was incorporated into the main volume control to allow the vocals to remain present in the mix after an initial manipulation by the user (they fade back out after about three seconds).

As with the school of fish object, several users can stretch the vocals at once, and in section one, mix in multiple layers of the backing textures at once. Also, like the distorted bass object of the previous system, the vocals object has a global sense of mass and momentum and can thus be flung across the screen like a bowling ball on a bungi cord. The object bounces off the edges of the screen so as not to desert the piece entirely. Yet because the squares themselves don’t move, the purple shading behind them takes on a eerie quality of invisible mass due to this effect.

3.3.3.2.4 The upright_bass Object

The upright_bass object plays the other half of the powerful rhythmic duo that dominates sections two and four. A versatile object, the upright_bass provides different bass riffs for sections one, two, and four along with bridge variations in sections two and four. Through all of these sections its sound is consistent: an aggressive acoustic upright bass with bright pops and snaps and a lot of low end. This unique sound is the result of a physical modeling algorithm implemented on the Yamaha VL1-m synthesizer that actually attempts to simulate the strings on a bass in order to produce a sound. This acoustic bass sound is again supposed to provide contrast to the largely electronic arrangement of the rest of the piece.

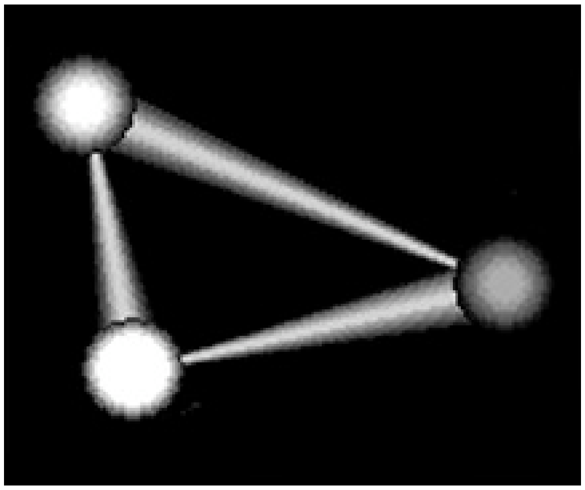
Figure 15. The upright_bass Object

The upright_bass object is visually represented by a red triangle with yellow circular nodes at the corners. It resembles the ball and stick constructions used to model molecules in chemistry classes. The triangle shape was chosen for its tightness, regularity, and angularity, all of which were attributes of the bass line that were intended to be visually embellished. The yellow nodes at the corners of the object would flash one at a time when a note was played by the object. Similarly, the highlighted node would jut out in one direction or another in sequence with the music, making the entire object appear to dance to the riff it was playing. The colors red and yellow were chosen to emphasize the hot, jazzy feel of the bass object.

Users could control the bass object by grabbing and stretching the yellow nodes at its corners. Each of the different nodes represented a different bass riff that, when grabbed, would be played out on the bass voice in lieu of the main riff. Further, the more a user stretched a node away from the object, the higher the new bass riff would be transposed. The accompanying riffs (one for each node) were written for each section of the piece including separate riffs for the bridges of sections two and four. One riff was usually a walking bass line, another a funkier variation, and the third a weird fast riff. As the user stretched the node, the object would waddle after it, allowing the user to drag the bass object around the screen like a puppy on a towel across a freshly waxed kitchen floor.

3.3.3.2.5 The breakbeat Object

Probably the most visually divergent object of the entire piece, the breakbeat object looks like Dr. Frankenstein took his home stereo system and grafted it to the back of a mechanical spider. The breakbeat object is the rhythmic center piece of the second section of the system. It is designed to look and sound like a mutant, robotic beat machine that slowly crawls across the screen to the pulse of a mechanically looped drumbeat sample. Like the other objects in the system, this object ignores the presence of other objects in its path and changes path only when it is about to walk off the screen. Visually the breakbeat object is composed of five components crawling around on four legs. The body of the object is a pulsating speaker cabinet that bounces along to the beat of the rhythm. Extending out from the body are four components, mounted on levers that can be pulled in and out by the users. The components are a reel-to-reel tape player, a ticking meter, a rotating LED level meter, and a red lever.

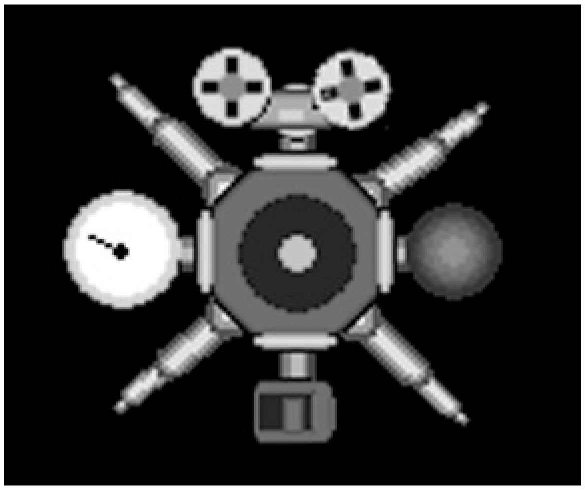

Figure 16. The breakbeat Object

Users manipulate the various rhythmic layers of the drum beat object by grabbing and stretching the components attached to the levers. Each component represents a different rhythmic layer; the level of that layer can be adjusted by setting the lever's position relative to the body. A second, more active layer can be mixed in by stretching the lever beyond its first joint. The components represent the following layers: the tape player mixes in a second breakbeat sample (over the main one) and when hyper-extended, forces the main sample to be played backwards. The ticker clock represents a ride cymbal part that extends into an active series of crash cymbals when the lever is fully stretched. The LED meter represents an electronic snare and clap layer that extends into low, round toms. Finally, the red knob lever mixes in two variations of a fast jungle snare running at twice the tempo if the main beat. The component riffs for each of these controls change between the bridge and the main sections of the second part of the piece.

3.3.3.2.6 The squirm_lead Object

The squirm_lead object was perhaps the least-changed object of those inherited from the second version of the system. The connection between is graphics, behavior, and the lyrical melody worked well in it's first appearance, thought the author, so why not use it again? This time around it plays the lead role as the sole melodic contributor to the piece. Visually, the object is based on the same algorithm as its last instantiation. Its behavior was modified, however, so that the object changes its direction to wind towards the pointer that is stretching it (thus allowing users to modify the object's direction on the fly). There were two basic control parameters on the object that were chosen depending on whether a pointer was stretching the front half or the back half of the object's body. The motivation behind this was to let two pointers at once manipulate different controls on the object. The front half of the body controlled the register transposition of the melody and amount of filter resonance while the rear half controlled an LFO attached to the filter cutoff amount and added in extra melodic notes. Though the controls stayed the same for the bridge of section two (the only section in which this object appears) the riff does change to a slightly more disjointed and repetitive variation.

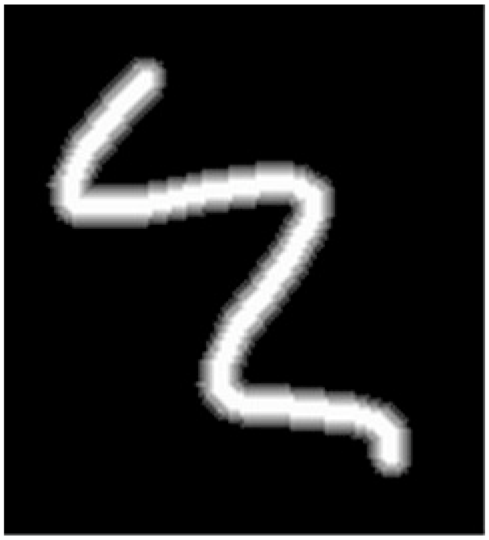
Figure 17. The squirm_lead Object

3.3.3.2.7 The acid_bass Object

The acid_bass object provides an exciting psychedelic kick to the second half of the second section of the piece. It represents a driving psychedelic synth layer, commonly known as an acid bass line, that can be articulated with dramatic filter sweeps and shifts in register. Graphically, the object is represented by a brightly colored centipede (red body, yellow legs) with eight body segments. A purple band runs down the back of the centipede like an LED counter on one of the old analog sequencers. In fact the entire object was inspired by a popular analog bass synthesizer called the Roland TB303 which is often used to produce similar psychedelic synth sounds in many contemporary techno tracks. The acid_bass object was conceived as a TB303 that had somehow sprouted legs and decided to crawl about the screen, writhing in its rhythmic intensity. The colors of red, yellow, and purple were chosen to intensify this psychedelic effect.

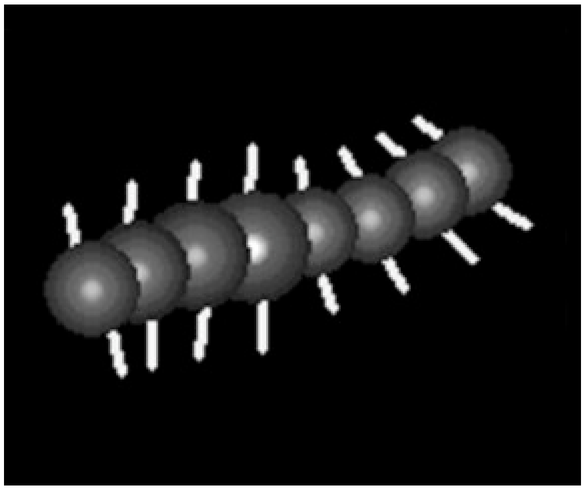
Figure 18. The acid_bass Object

The musical controls on the acid_bass object were simple. Grabbing and stretching one of the segments opens up a resonant filter on slightly distorted saw tooth wave form. The rear three segments of the creature, however, were special. Grabbing and stretching these would cause the object to switch into a new off-beat riff that could be dragged through register space as well as filtered. The way this off-beat bass pattern combined with the filter and transposition settings when the rear of the object was fully extended was considered to be particularly sweet by many of the systems users. When a segment was released it would snap back into place causing the entire centipede to jiggle a little as if it were made out of rubber.

3.3.3.2.8 The jungle_beat Object

The role of the swift, light drum ‘n bass rhythm part in section four of the piece is played by the appropriately named jungle_beat object. Posting along at 156 bpm the jungle_beat object is much lighter and faster than the more mechanical breakbeat in section two. Visually, the object is represented by a suspiciously similar algorithm to that used in for the distorted bass object in the previous version of the system. This time the undulating ball of spikes and triangles returns with a smaller, softer contour and shades of blue interspersed among its metallic edges. This object is supposed to be more nimble than its predecessor, with a higher morphing rate among the triangles and a subtle, faster rotation introduced to the contour as whole.

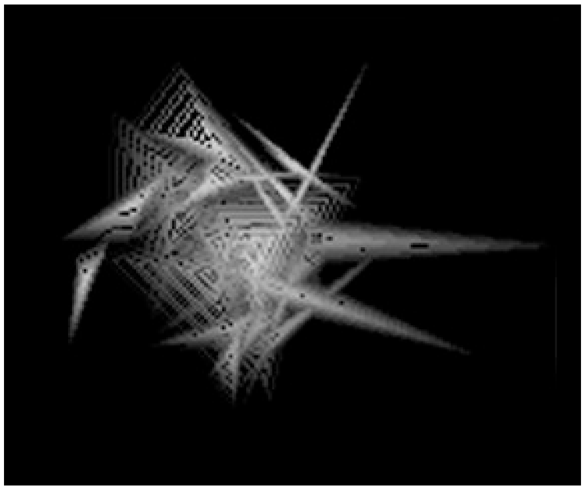

Figure 19. The jungle_beat Object

The object is actually introduced in the piece during section three as a snare roll building towards a crescendo at the end of the section. The object starts out as a quarter of its full size and grows in both size and volume throughout the second half of the section. By stretching the object into its star shaped form, users could rhythmically intensify the snare roll (moving from 16th notes to 32nd notes) and add extra accents. The object would swing around the user’s pointer as if tethered with a bungi cord in the same manner as the original distorted bass object. In section four, the object switched over to its four bar drum ‘n bass breakbeat. Users controlled a resonant filter sweep over this breakbeat by stretching the object. Opening the filter revealed successive layers of the rhythm part in order of their dominant frequency range: first the kick drum, then the snare layer, then the clap, and finally the hihat. During the bridge of section four, the drum part changes to an even lighter breakbeat sample. In this section stretching the object adds extra snare and cymbal layers instead of manipulating a filter.

3.3.3.2.9 The Keyboard and Percussion Tools

As described above, the keyboard and percussion tools play a very different role in this system than did the tools of the previous version. Rather than affecting multiple objects with higher-level parameters, these tools are supposed to allow users to solo over top of the rest of the piece. The tools would appear in random locations on the screen at specified times in the object score. The tools were scripted to appear to during what were perceived to be appropriate "solo sections." Users could fly over a tool with their pointer to acquire it and then solo with the tool by holding down the "B" button of the controller.

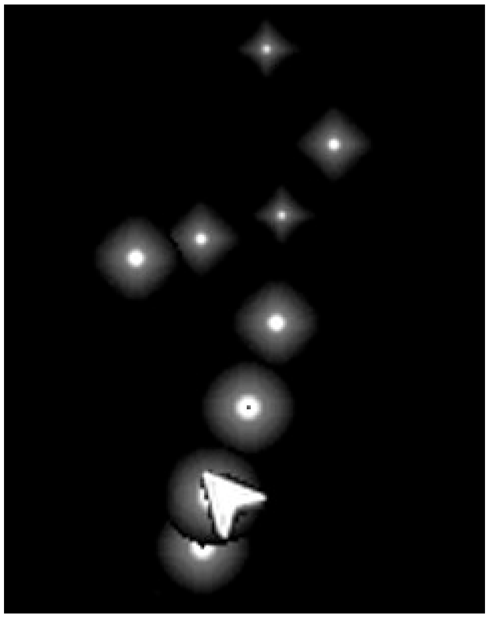
Figure 20. A pointer uses the percussion tool

These soloing tools were represented by spinning pairs of glowing stars that came in two varieties, red for a keyboard solo and blue for a percussion solo. When a user acquired a tool, the stars would spin around the user's pointer for a specified time till the tool was exhausted of its powers. As the user held down the "B" button to release notes, stars would be dropped behind the pointer as it traveled. When a user's pointer sat still, a basic comping pattern would be mixed in with either a keyboard or percussive voice (depending on which kind of tool was chosen). Upon more movement, a higher melody part would be added in to complement the comping pattern. The keyboard sounds were usually harsh, distorted organ sounds, while the percussive sounds ranged from congas and shakers to claves, toms, and hihats. However, the tools in sections one and three of the piece used more abstract electronic beeping and bell sounds to round out the ambiance of each section. Different keyboard and percussive solo tracks were written for every section of the piece, including the bridges of sections two and four, so that users would be able to solo at many different points during the piece.

3.3.3.2.10 The Bomb Tool

The purpose of the bomb tool, as described above, was to allow a user to highlight one musical object by simultaneously muting all of the others for one bar. Represented as a purple bomb icon with a short, smoldering fuse, the bomb tool would appear on the screen at times specified by the composer in the object score (usually a couple bars before eight or sixteen bar phrase boundaries). Like the other tools, users could acquire the bomb tool by flying over it with their pointer. Once acquired, a smaller version of the purple bomb icon would circle around the user's tool until deployed by the user pressing the "B" button. Once the bomb was dropped, it would immediately explode, sending a circular shock wave quickly expanding across the screen; the explosion would be accompanied by a deep, synthesized 808 kick drum hit. All of the objects except for the one closest to the explosion would then be muted for a period of one bar. Originally, the author planned to include special stunned animiations for all of the objects to visually reinforce the effect of the bomb tool on them, but due to time constraints, this feature was omitted.

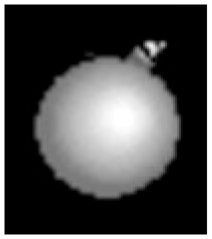
Figure 21. The Bomb Tool

3.3.3.2.11 The Advancement Tool

The advancement tool was represented by a red arrow pointing towards the right side of the screen. Users could advance to the next major section of the piece by colliding with the advancement tool icon with their pointers. Its appearance on the screen could also be scripted into the object score at any time by the composer. Generally, this tool would appear several bars before the end of the object score to give the users time to become aware of it before the score ends. As described above, if the users failed to grab the tool, purposefully or not, the object score would loop back to the beginning and start again.

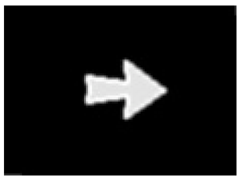
Figure 22. The Advancement Tool

3.3.3.3 The Ambient Section

The piece begins with an arhythmic ambient section that visually and musically has the feeling of a primordial soup. First to appear in the center of the screen are small green bubbles that seem to grow in spurts out the blackness. As users begin to pop the bubbles issuing organic "blurp" sounds and an occasional sample from an old TV commercial (actor Ronald Reagan plugging soap or some such), a school of one hundred tiny fish swoop down from the upper right corner of the screen. With the fish, comes an accompanying, watery, ethereal synth line that seems coat the inner sound space like a satisfying swig of blue-green Pepto-Bismal. By grabbing and leading the fish, one user complexifies the texture by shaping the synth line with filters and modulators, while another user adds eerie gurgling noises with the red solo tool she just acquired. Meanwhile, a collage of purple squares reminiscent of northern lights appear to the upper left, calmly summoning a soft female voice. With a little prodding from a user's pointer the squares erupt into a series of utterances and sung tones, while increasing the volume of their forlorn two-bar vocal melody.

After a few more bars, an ambient, wandering bass line appears to the right as a rhythmless equilateral triangle with dull yellow nodes at its corners. As it plays its line, it drags its nodes idly like a child drawing circles in the sand with a stick. As a user grabs one of the nodes and stretches it, the bass line jitters with a new sequence of fast notes sputtering out over several registers. Meanwhile the other user paints the screen with haunting gamelan rhythms of bells and gongs using the blue percussion tool. After a few more bars the bass exits, followed by the vocals, leaving the school of fish and the bubbles to their ambient noises. Soon, a red arrow appears in the upper right part of the screen. A user grabs the arrow and both bubbles and fish vanish suddenly into the darkness.

3.3.3.4 The Slow Rhythmic Section

The gentle ambiance of the first section is immediately shattered by the entrance of a hard driving mechanical rhythm machine at the beginning of the second section, slowly crawling out of the upper left corner of the screen. The machine, a frankenstinian collage of audio components and speaker parts grafted to the back of a mechanical spider, broadcasts an aggressive breakbeat sample, sodden in reverb and economically accented with a deep electronic kick drum and syncopated snare taps. A user grabs and stretches a ticking circular meter attached to the left side of the drum-bot and adds in a ride cymbal layer and a series of crash cymbals to the rhythmic texture. After four bars, the bass object returns to the screen with new intensity, completing the rhythm section with an aggressive, accented, funk-based line that fills the lower registers with thundering rhythmic intensity. Soon the shimmery vocals of the previous section reenter the picture while an eel-like red mono-synth takes the foreground with the main melodic idea. By grabbing and stretching the tail of the red eel as it winds its way across the screen, a user induces violent warbles to the melody line as well as adds in extra, ornamenting high notes. At the end of the melody, an eight bar phrase, one of the users picks up the bomb tool, and drops it on the bass line, giving it a one bar solo before diving back into the groove.

As the thirty-second bar of the second section passes, the mono lead eel and the vocals squares exit and the bass and drum objects kick into the bridge riff of the second section. The drums play another, slightly straighter but just as mechanical breakbeat, while the bass moves into a strident riff with an eighth note triplet feel. Grabbing a red tool, a user adds in a dissonant keyboard chord progression reminiscent of the work of Keith Jarret in the early 1970s with the Miles Davis band. As the user's pointer flies across the screen it mixes up an angular keyboard solo above an off-beat comping pattern. Simultaneously, the second user adds tom hits and rim shot accents with the blue percussion tool. Before too long both the vocals and the squirming melody reenter the scene, this time with much more disjoint and angular melodies. The entire musical texture reminds one user of her father’s golf outfit: clashing green and orange plaids sprouting skinny legs and topped with a yellow shirt and cockeyed hat, a spiced ensemble of textures indeed!

The 16-bar bridge is followed by a break in the rhythmic texture and the exit of all of the familiar objects introduced so far. Crawling into their place out of the bottom left corner of the screen comes a completely new creature: a brightly colored eight segment centipede. Accompanying the centipede is a growling distorted acid bass line, that opens up with a toothy filter as a user grabs and stretches one of the centipede’s segments. A stripe of purple runs down the back of the centipede in successive passes highlighting segments whenever a note is played out (which is almost always). In a few bars both the bass and the drum objects literally drop back onto the screen with a powerful down beat. Quickly, the vocals and the squirming red lead join the chorus in a powerful but dense reprise. Grabbing the centipede at the tail and stretching it across the screen, a user elicits high-pitched squeals from the acid bass line cutting through the texture like a soldering iron through warm milk. Finally, the red arrow advancement tool appears in the upper right hand corner of the screen and beckons the users into the next section. They oblige by taking the arrow and the pulsating red visual and sonic textures of the second section disappear.

3.3.3.5 The Accelerondo Section

The third section of the piece opens in stark contrast to the aggressive reds and yellows of the previous section, with the soothing blues and greens of the bubbles and the ambient school of fish. However, along for the ride to spruce things up, the acid bass line continues to crawl about the screen muttering to itself. Users trigger more samples and analog “blaumps” by popping bubbles in the now, fairly rich bubble field in the center of the screen. Mean while, an occasional bell or gurgling sound effect pops in as one of the user grabs and uses a tool. The tools have regained their more ambient sound effects from the first section of the piece. Meanwhile, the rhythm of the acid bass line is clearly getting faster. Soon a small, undulating ball of morphing triangles appears near the center of the screen in the bubble field, accompanied by a soft snare roll. Grabbing and stretching the new object, a user drags the ball into a spindling start of blue and gray spikes, intensifying the snare roll both rhythmically and dynamically. As the user pulls on the ball, it follows after him like its attached to a rubber band, bouncing off the walls like snare laden pool ball. The ball of triangles grows as the snare crescendo accelerates to a peak; the tempo of the acidbass line becomes almost too fast to pick out the original notes. Finally, without warning, the entire section lurches into the fourth section of the piece. This time there was no red arrow to give the users a chance to circle back for another try. With the accelerondo taking the tempo up from 104 bpm to 156 bpm, there was no turning back.

3.3.3.6 The Drum ‘n Bass Section

With the beginning of the fourth section, the acid bass centipede exits with the bubbles, leaving the undulating triangle ball and the ethereal school of fish. However, now the triangles have switched their tune from an accelerating snare buildup to a laid back 156 bpm jungle beat. A user grabs the object and shapes its rhythm by opening a whispy filter as she stretches it into a spindly star. Soon the bass object reappears with an ascending one bar riff that gallops in stride with the almost manic rhythms of the jungle beat object. Grabbing one of the bass nodes, a user mixes in a spirited, walking bass line whose contour arches through several registers. The spirit-like purple tiled vocals reenter the piece to augment the ethereal texture already outlined by the school of fish. Each are at their original tempos, only now the ethereal synth line fits cleanly into a 16 bar phrase, while the vocals wander out of time repeating every 12 bars. There is no melody in this section of the piece, the focus is rhythm and synthetic texture.

Moving into the bridge of the 4th section, the jungle beat object takes on an even lighter breakbeat, while the bass hops merrily along in a two chord progression. Stretching the jungle beat object, a user now adds extra snare lines and cymbals rather than opening a filter. A spirited keyboard solo meanwhile leaps from the circling red tool on the second user's pointer. After a short 16 bars of bridge, the objects switch back to their original riffs. The fish and the vocals reenter, after exiting for the particularly sparse bridge section, to add their texture to the mix. Soon the familiar red arrow appears on the screen again giving the users an opportunity to end the piece. But this time, they want to take the drum 'n bass section for another spin so they avoid the arrow and dive back into another rendition of the fourth section of the piece. Minutes later, on their second pass though the end of the final section of the piece, the users do take the exit arrow, and find themselves plopped in the middle of the bubble field again at the beginning of the ambient section that started the whole journey. Only this time, objects seem to be malfunctioning, bubbles not popping properly and the fish paying absolutely no attention to the user's repeated attempts to grab them. Before too long the system crashes and the journey is over. The piece ends as it started, in the debugger of the compiler.

3.3.4 The Unfinished Version

The fourth, unfinished version of the Stretchable Music system was actually developed in-between the second and third versions during the winter and first half of the spring of 1998. The system was an ambitious attempt to incorporate 3D graphics, networking, and a new kind of multi-level zoom interface into a collaborative, on-line interactive music experience for multiple participants. Unfortunately, this part of the project proved to be too ambitious and had to be abandoned before it was completed. However a demo system of the basic interface was created complete with pointers and stretchable objects, all rendered in 3D. The following section will discuss some of the features implemented in this system along with other features that were conceptually explored but never realized.

3.3.4.1 Hypothetical System Overview

The fourth version of the Stretchable Music system, dubbed 3D Stretchables by the author, was meant to explore the notion of a piece of interactive music taking place in an abstract, virtual space. Users would be able to navigate this space exploring different musical regions as well as affecting the music in those regions by stretching three dimensional musical objects. This view of the system differs significantly from the other three versions of the system where the computer screen provides a "gods eye view" of the musical playing field. With 3D Stretchables the playing field was supposed to be too big to fit into one window; the user's screen was to be turned into a portal to this abstract musical world. The author imagined different songs or musical jams spatially arrayed in this world so the user could fly between them, mixing their music together, and choosing in which jam to participate. Users would be able to view and contribute to this world from several levels, thanks to a new type of zoom interface described in more detail later in the section. The idea was to provide several levels of musical control to the user, ranging from high-level parameterized control (like the angel dust in the second version of the system) through direction real-time control (through stretching) to a new low level of musical control that would allow users to actually program new musical content into the musical objects. The author imagined making the first version of the system available to the public through an FM radio broadcast. Users would log onto the system through the net and hear the results of their contributions to the system in real-time over the radio. The author originally planned to initially try this scenario out with a late night broadcast from MIT's own WMBR (88.1 FM) with a couple of friends.

3.3.4.1.1 What Was Actually Implemented

As mentioned before, this vision of the 3D Stretchables system was never fully translated into running code. Parts of it, however, were implemented. A 3D environment was developed where a user could travel using two different modes of navigation (described in more detail below in the Bi-Modal Navigation Mechanism). A Custom, 3D pointer was designed to allow the user to interact with the world. Two types of test objects were implemented that the user could grab with the pointer. One was simply a hollow box that the user could drag from one place to another. The other object was a blue, globular ball that could be grabbed and stretched by the user. Constructed of a 3D mesh of massed particles connected by a web of simulated rubber bands, the ball would gyrate in way similar to the main_bass object in version two of the system when stretched. Both of these objects lived in a test world that consisted of a green grid of squares arrayed on a plane.

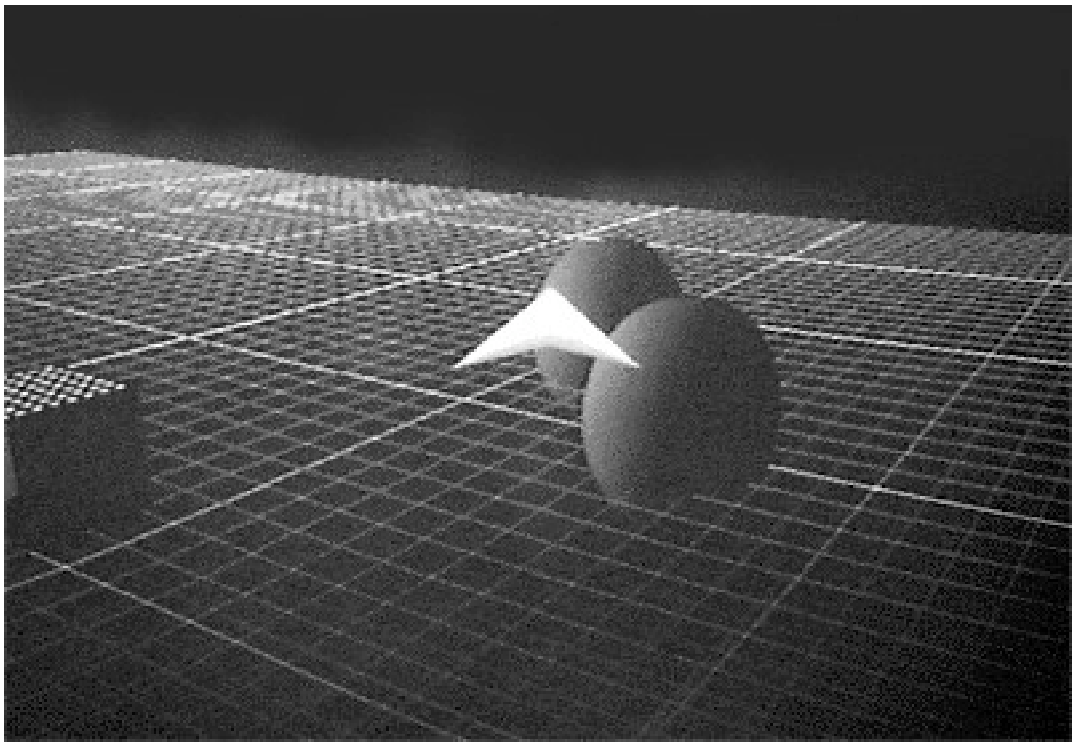

Figure 23. Screen shot from 3D Stretchables featuring the pointer (rearview) in navigation mode, two stretchable balls, and a test block to the lower left.

3.3.4.2 The Zoom Interface

One of the most important features to be implemented in the 3D version of the Stretchable Music system was the zoom interface. As described above, the purpose of the zoom interface was to allow users to interact with the music in system on several different levels. On the highest level, users would be able to affect musical objects by sprinkling "spices" over multiple objects at once to influence general types of parameters about how they played their musical parts. Like a conductor, a user would be able to marshal an ensemble of objects to play more vigorously or actively, more quietly, more together, or more independently, depending on which spice or tool they chose to apply. At the second level of the zoom interface, users would directly affect objects by stretching them, much in the same way they do in the other systems in the project. The lowest level of interface would allow the user to actually reprogram the musical content and behaviors of the musical objects themselves. One could imagine one user specially crafting (or training, depending on your metaphor) a special bass object that he could then interject into a jam happening concurrently with his friends.

One of the challenges of such an interface would be to make the transitions between the different modes of operation as seamless as possible. Using the power of a 3D graphics system, the author had planned to literally zoom the user's viewpoint in and out to provide a transition between the top two levels of controls. The third, lowest level of control proved to be more problematic. How does one smoothly shift from a real-time activity like stretching to an off-line activity like configuration and programming? One suggestion offered by Professor John Maeda was to have the objects flip over at a certain low level of zoom to expose a set of configuration controls on their bellies. Other persistent problems, however, prevented this avenue from ever being explored. Chief among them was the concern of the author over what exactly were the right controls to give up to a user of the system. How much of the system would remain the author's musical composition if you allow users to go in and change the musical material and behavior of the objects? These delicate problems remain unsolved in the current versions of the system.

3.3.4.3 The Bi-Modal Navigation Mechanism

Another vexing problem posed by the 3D Stretchables system was the issue of navigation in a 3D world. Because of their fixed view ports, the other versions of the system did not have to worry about navigation allowing users to concentrate on manipulating the musical objects on the screen. With the 3D system, part of the fun was supposed to be exploring and discovering the abstract musical landscape. However, effective 3D navigation is a tall order to stack on top of point-and-click pointer manipulation from the mouse; the author did not want to complicate the interface by involving a separate level of physical interface by incorporating the keyboard or a second joystick. The solution implemented in the test version of 3D Stretchables utilizes a bi-modal interface paradigm to allow users to focus on navigation and stretching at different times. In one mode the system provides the user with a top down view of the world. The user can zoom in and out of this view with two keys on the keyboard (a kludge) and move horizontally by moving his pointer to the edge of the screen in the direction he wants to go. However, when the user presses and holds the right mouse button the camera swings down behind the 3D pointer and allows the user to fly about the world from an almost first person view. The effect of piloting the pointer in this mode felt a bit like flying hang glider. The controls were finicky and slightly delayed in their response; it was easy to over shoot. Presumably after flying to his point of interest, the user would release the right mouse button allowing the camera to swing back up to a top down view and enabling the user to stretch objects once again. The fluidity of the transition between the two modes of interaction and control was deemed the strongest point of this bi-modal interface approach.

3.3.4.4 The Network Architecture

Designing the system to be playable by multiple users connected to a network proved to be a formidable challenge on several levels. On an architecture level, the system had to be analyzed to determine what was the most important information that could be passed in the least quantity between machines to keep them relatively synched visually and musically. In the software an abstraction barrier was drawn between this information and the actual mechanisms through which it would be broadcast. Machines were to be organized in a star network where one machine would serve as a host by re-broadcasting all of the state changes of its clients to all of the participants on the system. The implementation of the communication layer of this networked infrastructure would depend on the needs of the particular system setup. If the system were to be setup as a mutli-user interactive experience in one room, with all sound coming from a centralized sound source, then network latency would be a key issue. In this case, using a communication protocol such as MIDI might be preferable to TCP/IP. In such a configuration, the host computer would serve as the main music generation machine, acting as a gateway for all musical data to pass to a separate bank of synthesizers.

As described above, the author had planned to extend this configuration to allow geographically dislocated users to contribute to the system by broadcasting the music over FM Radio. With this setup, TCP/IP would clearly be the main choice for connectivity between the computers, though for sensitive sync issues, the author hypothesized about sending a hidden high frequency sync signal imbedded in the audio signal transmitted over the radio to keep all of the music in time with the graphics. In such a case, a user would have to connect her computer to a radio through the microphone port on her sound card as well as be hooked up to the Internet. In an ideal scenario, an external rack of synthesizers would not be necessary to synthesize the music for the system, but rather all audio processing would take place on board individual users' computers. This would allow a much wider geographic spread between users (out of the signal range of the radio) as well provide more tolerance for timing inaccuracies an sync issues. The problem would no longer be to keep all of the client computers in tight musical sync, but to re-sync all of the external client information at a single user's computer to provide a temporally coherent musical experience. Though these networking issues were explored extensively at a conceptual stage, only the networked abstraction interface was ever implemented in the demo version of 3D Stretchables.

3.3.4.5 Aesthetic Issues in 3D Graphics

One of the most difficult challenges perceived by the author in developing the 3D Stretchables system was that of how to make the visual design of the system expressive within his particular style. Driven by industrial design programs and computer games, 3D graphics hardware and software has been obsessed with realism for many years. Hence many specialized techniques have been developed to allow sophisticated calculations involving texture mapping and other common processes to be done very quickly on relatively inexpensive hardware. The result of these efforts however is an extremely noticeable 3D aesthetic that is very difficult to escape from with commercial hardware and software. Thus its easy to do 3D graphics that look basically the same as everyone else's 3D graphics, perhaps with a subtle innovations here or there. Its very difficult, however, to break out from the norm and add detailed stylistic touches that will make one's work more personalized and really stand out from the crowd. This situation is somewhat similar to the problem of writing music with General MIDI voices on a General MIDI compatible synthesizer. Sure you can play your music anywhere, but it will sound the same as everyone else's (same snare hit, same kick drum), i.e. mediocre. This constriction of the visually expressive styles of 3D graphics is one of the main reasons the author chose to go back to two dimensions for the third version of Stretchables. The author will no doubt return to take another stab at 3D graphics, but next time will try to make use of customizable tools such as Rolf Rando's cartoon rendering system [Rando 97] to achieve a more personalized visual aesthetic than is now generally available.

3.4 The Software Implementation

The following discussion examines the basic software infrastructure shared by all three and a half of the systems developed during the course of this project. Like the other features of the systems, this structure evolved with every iteration. The specific components discussed here make up the third and final version final system but are present in some evolutionary form in the previous versions of the systems. The systems were implemented in C++ because of its nice balance of object oriented organization and performance.

3.4.1 Standard Tools

Several sets of standard, publicly available, software tools were used to provide basic level functionality in both graphics and MIDI services. In some ways the choice of these tools had a major effect on the over all look and sound of the piece. The four most important software tools utilized in the project are discussed below.

3.4.1.1 The Rogus Library

The Rogus McBogus MIDI library [Pelletier 96] was developed by Ben Denckla and Patrick Pelletier during the summer of 1995 to provide a basic platform for the development of MIDI based interactive music programs for Tod Machover's *Brain Opera* [Machover 96]. Written for the Microsoft Windows platform, Rogus provides developers with essential low level MIDI services such as scheduling and input parsing while wrapping MIDI information in a convenient data structure. The library also provides certain high-level features such as a MIDI file reader and a Score Player to manage and play back musical data stored in MIDI files.

The Stretchable Music software uses the Rogus library in two primary ways: Score playback and immediate mode MIDI output. Most of the music for the systems was originally written as two, four, and eight bar sequences in a separate MIDI sequencing program. These sequences would then be loaded into the software at run time as a couple of MIDI files to have their tracks parsed up amongst the different music objects. Each of these objects would store its relevant MIDI data in the Score data structures provided by Rogus. Objects could then play the MIDI data back, looping it when appropriate, and perhaps modifying the note output on the fly. The other common technique was to use the Rogus MIDI services object to write MIDI messages directly to the output port. This way MIDI controlled synthesizer parameters, such as the filter controls described in detail above, could be modified in real-time based on the state of the system (i.e. how much a given object is being stretched).

3.4.1.2 Microsoft's GDI Drawing Tools

Most of the notorious "stretchable graphics" in the three, two-dimensional versions of the system were done (extremely inefficiently) using the standard drawing tools of Microsoft's Graphics Device Interface (GDI) of the Win32 API [Microsoft 97]. This pen based interface allows developers to create simple 2d graphics by combining graphic primitives such as lines, ovals, rectangles, and other polygons. A typical sequence of GDI calls involves selecting an object with which to draw, such as a colored pen or brush, and then applying this object to a drawable surface, usually a window, with a drawing function. Sophisticated graphical techniques such as shading or anti-aliasing are not provided in this interface.

The second two versions of the system utilized the following, rather inefficient, shading technique to give the musical objects a two and one-half dimensional look. Lines were drawn in successive layers between two points, beginning with thick dark lines and progressing smoothly to thinner lighter lines. This technique was particularly effective for drawing rounded tubes such as the boundary on the main_bass object in version two of the system. The drawback to this technique is that many pixels are drawn multiple times, up to the total number of shades used in for the object. Furthermore, the GDI calls themselves are notoriously slow and, as Microsoft admits, not intended for high performance multi-media applications [Microsoft 97].

3.4.1.3 Microsoft's Direct Draw Interface

Another graphical software tool used throughout the project, but most extensively in the third version of the system, was Microsoft's Direct Draw interface [Microsoft 97]. Part of the Microsoft Direct X library, Direct Draw is intended to provide developers with lower level access to basic functionality of standard PC graphics hardware. The primary use of Direct Draw in the Stretchable Music systems was to set up a special double buffered chunk of memory directly on the graphics board to use as the viewable surface. Double buffering is a common technique in computer animation where the computer draws the next frame of animation on a back buffer while displaying current frame on the front buffer. When the back buffer is fully drawn, the buffers are simply swapped to prevent the flickering or ghosting of images. Another place direct draw was exploited, especially in the third version of the system, was to copy bitmaps to the screen very efficiently. The Direct Draw interface allows developers to store off-screen bit maps in extra memory located on the graphics card, making it extremely efficient to copy these bit maps to the active drawing surface. Performance gains from these techniques influenced the author's decision to base the graphics of the third system heavily on pre-rendered, bitmap material. Because of these optimizations, objects such as the ether_synth fish ran up to 15 times faster than their predecessors in version two of the system.

3.4.1.4 OpenGL

OpenGL [Woo, Neider, Davis 97] is an open software standard created by Silicon Graphics for polygon based three dimensional computer graphics. Among other functions, it allows developers to perform powerful mathematical manipulations on graphical primitives such as points, lines and polygons to produce detailed, rendered 3D dimensional scenes. An unofficial (unlicensed) implementation of the OpenGL standard was used to create the graphics of the 3D Stretchables system. Created by a grass roots movement to make 3D graphics available on may different platforms, the Mesa library complies with the vast majority of the standards set forth in the OpenGL specification. Driven especially by the desire of the computer gaming community to play one particular game, Quake by Id software [Id 1997], on many different types of systems, the Mesa library is optimized to run efficiently on powerful but inexpensive PC graphics hardware, most notably the 3dfx Voodoo Graphics card. OpenGL and Mesa were chosen as the 3D platform for the 3D Stretchables system because of the relative ease with which graphics are developed using them, and because of the high performance of the Voodoo graphics hardware at the time.

3.4.2 The Object Hierarchy

The majority of Stretchable Music software consists of a hierarchy of software objects that organize data and functions around different key parts of the system. In addition, the system takes special advantage of inheritance to provide a standard interface to many types of customized software objects.

3.4.2.1 The moe Object

The most central object of the entire system is the moe object, which stands for Music Object Environment. This objects sets up an encapsulated software environment around all of the principle elements of the system, including the music objects, shielding them from as much arbitrary operating system dependent code as possible. In addition to music objects, moe is home to pointer objects, tool objects, and object_score objects (all described later in this section). Moe performs many functions to manage the operation of all of these objects. It disperses all user input to the appropriate places (generally to the pointers). It facilitates the interaction of different types of objects with each other. This includes detecting when pointers have collided with unclaimed tools, and pointers grabbing and stretching objects. Similarly, moe tells every visible object when to draw itself and provides the basic necessities required to do that (a pointer to the graphic surface on which the objects are to be drawn). Moe also manages the object_score mechanism, parsing various time-stamped messages to objects (and to moe itself) telling objects to do things like appear and disappear from the screen at certain points of the piece. Additionally, moe enables the system to switch object_scores when a user decides to advance to the next section of the piece. Finally, moe, provides basic functionality for debugging text output; because the graphics mode of the system takes up the whole screen, a separate window can not be used conveniently for text output.

3.4.2.1.1 The object_score Mechanism

One of the most important and interesting mechanisms handled in moe is the object_score mechanism. An object_score is a collection of time stamped text messages that control the large scale time structure of the piece. The messages are dispersed by moe at the appropriate times during the piece to tell objects, for instance, to disappear from the screen. However, due to the implementation of the message passing system, any type of message can be sent to an object, as long as the object is prepared ahead of time to receive it. Music objects are not the only software objects that receive these messages however. Tools can be scripted into the object_score to appear at specified times during the piece and to change their music for particular sections, say switching to a new keyboard riff for a bridge section. Music objects, too are instructed to switch musical modes (again, for example, as into a bridge section) through messages contained in the object score. Like the more musical MIDI scores in the Rogus library, object scores can be looped to provide as much time as needed for a user to explore a particular section.

The message passing algorithm for the object_score mechanism utilizes a two-pass strategy to efficiently, but flexibly represent messages as unique integer IDs which can be parsed more quickly on the fly than their text counterparts. In initial pass through the object_score at startup time, moe confirms that all of the objects mentioned in the object score are prepared to receive the messages posted there. It then assigns each message a unique integer ID for that object, that can be decoded efficiently at runtime. Moe also is allowed to receive messages and in fact uses quite a few of them to manage tools, tempo switches and other global functionality.

3.4.2.2 The input_master Object

Another globally accessible object like moe, the input_master object provides a basic layer of abstraction between operating system dependent user interface code and the rest of the system. As its name implies, the input_master object is a convenient place to put code that interprets input information from a variety of devices and packages it into a standard, understandable form for the rest of the system. Ideally, then, switching the types of input devices used in the system would only require changing code in the input_master object. The rest of the software would continue to see the familiar interface provided by the input_master. This input device mobility was used during the development of the third version of the system. For most of the development period, a mouse was used as the primary input device. At the very end, gamepads were incorporated as the final input devices. Thanks to the software abstraction barrier provided by the input_master object, this job was a smooth, relatively transparent process with very little tinkering required for rest of the code.

3.4.2.3 The synth_master and synth Objects

In much the same spirit as the input_master object, the synth and synth_master objects serve as layers of abstraction between the music_object code and actual MIDI output to the synthesizers. Synth_master is a global software object that contains instantiations of specialized synth objects for each synthesizer in the MIDI rig described above. This layer of abstraction was necessary because of the diverse capabilities and MIDI implementations of modern MIDI synths. For instance, according to the MIDI spec there are several different ways one can change a patch on a synthesizer. Some companies even go beyond the spec to introduce their own proprietary combinations of MIDI messages to achieve the same functionality. However, by simply imposing a single standard interface to switching patches in the abstract base class synth, one can cover up the proprietary messiness of individual synths by wrapping their implementation in an overridden method. The result is a much cleaner and manageable standard way to address MIDI output to the variety of synthesizers used in the project. A final advantage to this structure is the way it provides a convenient abstraction barrier for MIDI channel mappings. Certain synths in the setup receive MIDI on only a few channels; this software structure makes it very easy to change these channels in the code, and enforce these specifications on the synths themselves (by turning off unused channels).

3.4.2.4 The music_object Objects

Conceptually, the music_object is the center of the entire project. All of the other software in the system is designed to either help music objects do their job (play music and draw graphics) or let users interact with the music_objects. The music_object class is the abstract base class for all of the graphically animated musical creatures in the piece. Its functions provide an interface that enables all music objects to receive user interaction information from moe, allows them to draw themselves, and standardizes certain essential functions like starting, stopping and collision detection. The child classes of music_object are where most of the code relating to the content of the piece is placed. As mentioned in section 3.2.1, music objects and their ancestors hyperinstrument modes are essentially convenient software containers for mappings between user input and the systems musical output. Music objects take this concept several steps further by providing a customized graphical interface for each object and running several objects side by side at the time same.

Structurally, this connection between the music_object class and its derived classes is achieved through the powerful C++ technique of inheritance. All of the interface functions of the music_object class are declared as virtual, enabling them to be overridden by derived classes. Hence, when a child class wants to utilize functionality of the music_object class, it simply overrides the appropriate function and includes its specific code there. For example, all music objects have the power to draw themselves, provided by the interface of the virtual function of the music_obejct class, update_graphics(LPDIRECTDRAWSURFACE dds). When a derived class wants to draw itself in a particular way, it merely delcares its own update_graphics function, overriding its parent's version of the function, and automatically recieves the appropriate hooks, namely a pointer to the main Direct Draw surface of the system. This surface is passed in by moe calling update_graphics on a pointer to the base class music object that refers to the specific instantiation of this derived class object. The result of this software structure is an easy way to manage visually and musically diverse types of music objects through a powerful standard interface.

3.4.2.4.1 A Word About Generators

The original concept of a music_object called for all content specific graphics and music code to be contained with in one compact software object. In earlier implementations of the system, this was literally the case: each music object was implemented in one huge file that controlled both the graphics and music of the object. In the final version of the system, however, this implementation was considered too messy, and graphics and music code for each object were split into separate software structures. All of the graphics code and mapping control code were kept in the original music object structures. For the musical functionality of each object, classes called generators were created to provide an abstract set of controls to the music object that could each vary between zero and one. The controls ranged from tambral parameters like a low-pass filter cutoff amount to note level parameters like transposition of an arpeggio. The generator was usually owned as a member of the music object but would often contain a pointer back to its parent so it could communicate music related state information back the object (to change the graphics in response to notes being played for instance). The concept of generators was so successful when used with music objects that it was eventually used as a technique for encapsulating the music code of certain tools as well.

3.4.2.5 The pointer Objects

The pointer class is an abstract representation of the user's interaction with the rest of the system in moe. All of the music objects and tools are designed to respond to actions from pointer objects and not directly from user input. By funneling user input through pointers, the system can more easily handle different types of input devices and multiple simultaneous users. In fact, the pointer abstraction was one of the most important lessons learned from development of the unfinished, networked version of the system (which demanded an abstract pointer class to represent net connected users). Different types of pointers can be subclassed from the pointer class, to respond more optimally for different types of user input data. One might respond very acutely to absolute position of a controller and thus be suitable for a mouse. Another might act with a considerable amount of momentum and slip in the system, yielding the relatively loose control more suitable for the binary direction buttons on a gamepad. Pointers also enable their children to draw themselves and to acquire and use tools.

3.4.2.6 The tool Objects

The tool class is used represent tools in the same way that music_object provides a base class for the musical parts of the piece. The object provides an interface that takes care of everything a tool might want to do, such as draw itself, detect a collision, respond to a collision, have an owner (a pointer), or respond to user input (the "B" button). Note some tools, like music objects had their own music generators for convenience as mentioned above.

3.4.2.7 Main.cpp: A Haven for Windows Code

Most of the ugly interfacing between the software system and Microsoft's Windows operating system (as much as possible) was crammed into a file called main.cpp. Such calls could have been easily included in the moe object, but were banished to a separate file largely because of the aesthetic preference of the author. This was an attempt to isolate the code as much as possible from proprietary Microsoft calls. Unfortunately this barrier couldn't always be held; all of the graphics calls in all of the music objects use Microsoft specific functions. Nonetheless, all of the Windows message passing architecture is handled through this file, either talking directly to moe or going through input master first. Also, all of the basic initialization of the system is triggered from here.

3.4.2.8 Other Utility Objects

Several utility classes were used widely through out the system that ranged from mere matters of convenience to important functionality that influenced the behavior of the entire system.

3.4.2.8.1 handle Objects

Handle objects provide a convenient way for the system to keep track of which pointer is grabbing and stretching which object. They identify who they are grabbed by and provide an abstract function interface to test to see if a pointer's attempt to grab them is valid. Music objects usually derived their own specific handle classes from the handle base class to add object specific functionality to them.

3.4.2.8.2 animation Objects

Animation objects provide a convenient software wrapper around the core information necessary to create simple bitmap animations. They combine a pointer to a Direct Draw surface (which contains the actual bitmap frames) and information about how many frames the animation has, where the frames are located on that surface, and what size they are. The class will also draw the animation in with an easy to use single function call. These objects came in very handy in the third version of the system where heavy use was made of bitmap animation.

3.4.2.8.3 score_wrapper Objects

The score_wrapper class was created in response to a bug in one of the Rogus ScorePlayer objects. For some reason, the Rogus NotePairScorePlayer would not keep proper time while looping. The author hypothesized that this error was due to the score player looping back to its beginning at the last event, which in a note pair score player is a note-on event and not the final note-off message of the score. Hence, the duration of the last note in the score would be lost during every successive loop. The score_wrapper object simply kept track of time on its own and started and stopped its score object at the appropriate times to simulate looping.

3.4.2.8.4 particle Objects

Particle objects were used throughout the system to simulate physical properties such as mass, momentum, and acceleration. Based on a Java class written by Reed Kram for problem set five of John Maeda's class "MAS 964: Principles of Visual Interface Design" [Kram 97], the class includes state variables such as mass, velocity, acceleration, and position to simulate the behavior of a simple particle. Forces can be applied to the particle by calling its iterate() function to update its various state variables. This class was modified and extended in various ways, including the development of a three dimensional version that included parameters for rotational motion (used to damp the camera in 3D Stretchables). This class was essential to providing the stretchy feeling and behavior that was a characteristic of many of the animated graphical objects.

3.5 Project Wrap-Up

To conclude this rather lengthy and detailed chapter on the nuts and bolts of the Stretchable Music systems, this section will attempt to summarize the primary points of interest in preparation for analysis and criticism of the systems in the next chapter. Because the specific design goals of the project evolved with the project itself, several basic motivations played a more important role in guiding its direction than any single clear-cut engineering or artistic problem. Briefly stated, these were the desire to make electronic music performance more understandable to an audience, the need for new types of instruments to perform today's popular electronic music live, and the drive to make electronic music performance and control accessible to amateurs as well as professionals. The systems, that were later described in this section approached these problems from several different angles and eventually landed on the concept of an interactive music system as a musical piece that was explained at the beginning of this thesis.

There were three major software systems that comprised the project and led to this viewpoint of interactive music. The first system explored an abstract, iconic graphical representation of musical layers and discovered the importance of continuous timbral control in an interactive music system. The second version investigated more sophisticated graphical links between graphical control and musical content, began to be viewed as more of piece than an instrument, and made use of subtle musical controls that had particular relevance to musical context. The final system incorporated more music into a longer and more diverse interactive experience, implemented multi-player functionality, and cast the system into a more defined light slightly closer to traditional video games. The unfinished 3D, version of the system suggested interesting future areas to explore (most importantly networking) while highlighting frustrating aesthetic limitations and interface complications introduced by a 3D graphical environment. Finally, the staples of the object-based software infrastructure were discussed to show how the internal design of the systems paralleled some of the basic concepts behind the interface: musical objects that couple a unique graphic interface and musical content into a single package.

4 Results and Analysis

The goal of this section of the thesis is to shed light on questions relevant to the evaluation of the project such as “what worked, what didn’t, and why?” The section takes a three-pronged approach to this evaluation by first analyzing the observed results of users interacting with the system, then highlighting several points of criticism that the author found most significant, and finally asking whether the systems achieved what they initially set out to do. With any luck, this analysis will uncover some important points that will help with the design and development of stronger systems like Stretchable Music in the future.

4.1 The Venues

An interactive system can hardly be evaluated on the features of its software alone. As Brenda Laurel points out, it takes both a computer and a person to make an interactive system [Laurel 91]. It would be ludicrous to evaluate the success of such a system without first considering real-world scenarios where actual people (preferably those outside the system’s immediate field of research) completed the loop of interaction by using the system. Fortunately, the Stretchable Music systems have been put to this test in a variety of venues ranging from one-on-one demonstrations to performances onstage to interactive installations. The following discussion highlights the strengths and weaknesses of the systems in these venues and summarizes particular comments users have made when interacting with the system.

4.1.1 Lab Demos

By far the most common venue in which the system was shown was in the office of the author for demonstrations to a small group of people (three to five). Though public opinion is a fickle matter, the system seemed to receive very positive reactions from many of the people who played it in this situation and was considered one of the most popular attractions of the Opera of the Future group’s demonstration line-up. This demonstration situation is perhaps more like an intimate, private environment than a public installation, since one or two users actively play the system while a few others watch and take turns. Such a system might easily be adapted to another an environment such as the living room of a home (where children play with consumer video game systems), or a study where people play games on a home computer. The features of this environment that make it ideal are the small number of people there, and the opportunity for each person to comfortably explore the system and figure out its various controls.

In many ways, this is the environment for which the Stretchable Music systems were developed and hence was where all three enjoyed the most success. As described previously, the systems utilize a single graphic display and a pair of speakers for output, while limiting user participation to one or two players (one in the first two versions, two in the third). Further, the controls of the system are not immediately obvious to someone who has not had a chance to manipulate the stretchable objects and hear the results (stretching isn't the first control mechanism most people think of when they think about controlling music). Thus the system excels when users gain first hand experience using it and are left free to explore the various capabilities of its interface. In addition, a small group of participants can actively engage in the exploration of the system by making suggestions to the current users such as "go stretch that object" or "see what that icon does." This intimate, private situation is similar to the ones in which video games do best (kids in the family room, study etc.) and reinforces the connection between the Stretchable Music systems and traditional video games.

4.1.2 Performance: The Wearable Computing Fashion Show and *Cubestock*

The second and third versions of the Stretchable Music system were actually used in performance settings in two different venues. The second version of the system was played as a musical accompaniment to the "Wearable Computing Fashion Show" [MIT 97] in the Media Lab's own Experimental Media Facility during a large sponsor meeting in the fall of 1997. Audio from the system was mixed in between recorded songs to accompany some of the models as they walked on stage to demonstrate mostly hypothetical concepts about the collision of fashion and wearable technology in the not too distant future. The graphics for the system were displayed on large television set to the side of the stage. In this setting the author was concentrating less on demonstrating the functionality of the system to the audience, than he was on simply using it to produce good music. Hence he carefully chose which controls to manipulate at different times to accent, highlight, articulate, and shape different parts of the music.

Almost six months later in the very same Experimental Media Facility, the author co-produced a media art show called *Cubestock* (the official name, *(Art X Media)^{Music}* was largely ignored) to showcase various creative works that had been done the previous year by Media Lab graduate students. Among other pieces, the author's own Stretchable Music (the third and final version of the system) premiered there with a performance as well as installation period where the audience was invited to come up on stage and try the system out for themselves. The performance part of the situation was analogous to the fashion show in that the primary goal of the performers was to make good music with the system rather than to demonstrate its capabilities. On the other hand, without the scantily clad models on stage, the Stretchable Music graphics system, projected onto a large screen above the stage, played a much larger role in communicating to the audience some of the control parameters relevant to the system. This goal, of creating a stronger visual connection between electronic music control and performance for the sake of audience, was achieved with limited success.

Both the Wearables fashion show and the *Cubestock* performance highlight the Stretchable Music systems being used as performance systems. As performance systems, they excel because of their unique method of communicating the performer's musical actions through graphical representation. Nonetheless, in these particular situations, where the audience had never seen the systems before and had no prior knowledge of the connection between the graphics and the music, much of this goal was compromised. Most audience members made a vague connection between what they saw on the screen and the music that they heard, but many were confused as to the details. For the installation at *Cubestock*, however, this confusion worked to the advantage of the system by in many cases perking people's imaginations and encouraging them to come up and try the system afterward. The two systems also benefited in the performances from the particular musical controls that were built into the musical objects. As mentioned in the previous chapter, these controls were chosen by the author as those most relevant to the production of a lively and interesting piece of music within the idiom of popular electronic music. Hence all of the controls needed to bring the pieces to life were literally at the performer's fingertips. The major drawbacks of the systems in a performance context were the limits they imposed on improvisation and the fact that the user interface only allows a single user to manipulate one musical control at a time.

4.1.3 Installations at *Cubestock* and The Space

The third version of the Stretchable Music system was also presented in two different installation settings where members of the public were invited to try the system out for themselves. The first of these was at *Cubestock* where the system and the stage were opened up to the audience after the author's performance. The stage was set up with a large computer monitor facing a couple of single seat couches in an arrangement reminiscent of a TV in a living room. Participants used two gamepad controllers to manipulate the musical objects on the screens; audible feedback was provided by stage monitors placed around the couches. Two users at a time would receive one pass through the song before handing the controllers over to the next duo waiting. A similar setup was used in an installation at a small performance space in Worcester, MA called The Space, during an evening of experimental music performances. In this venue, however, the system was in a separate room from the main stage and was not performed prior to the installation.

In the installations at *Cubestock* and The Space, the system received many positive comments. Unlike some interactive art installations, though, which are designed for users with no prior knowledge of the interface, the system did require a quick explanation of its basic mechanics to get most users going. Fortunately, users waiting to try the system could pick up many of the basic tricks by simply watching others play or listening in on a lesson or two. The learning curve of the system's graphical interface did not seem to be too steep for uninitiated users to quickly rise to a level where they were having fun. The system's physical interface however, proved more problematic. The task of maneuvering a small pointer around the screen to grab objects with a standard video game gamepad proved frustratingly challenging to many users of the system. Users with prior experience playing video games or even using computers enjoyed a significant advantage over those who lacked such experience. On a different note, in contrast to the musical controls available in the second version of the system, the third version provided a level of depth that was difficult for users to fully explore and understand on one or two passes. The third version, hence, was deemed more appropriate, perhaps, for the first scenario described in this section, while the simpler, clearer second version of the system might have made a better installation.

4.1.4 Installation with Laser Rangefinder at SIGGRAPH '98

In a final installation at the SIGGRAPH '98 conference on computer graphics, the Stretchable Music system was combined with a laser range finder interface built by Josh Strickon [Strickon, Paradiso 98] to allow participants to manipulate graphical objects projected onto a six by eight foot screen by grasping and stretching them with their hands. For this hybrid system, the second version of the software was used, mainly due to the fact that it was the only working version in existence at the time that the marriage of the two projects was proposed (the first version of the system having been decommissioned months prior). The combination of the accessible graphics and controls of the second version of the system with the unencumbering and relatively accurate interface of the laser rangefinder proved to be a winning formula for the installation. Users enjoyed being able to stretch objects with large body motions across the screen. Moreover, the size of the graphic display made it easy for a crowd of people to watch one person interacting with the system. Due to implementation details, however, users could only use one hand at a time to stretch objects; the other hand controlled the sprinkles tool described in the last chapter. This led to some confusion when multiple users tried to manipulate objects at once; the system quickly lost track of which user was which and would randomly switch the roles of sprinkles and stretching between the users.

4.1.5 User Comments

Over the course of many demos, several installations, and a couple of performances the author collected countless comments and suggestions on the system from users of all ages, shapes and sizes (author's note: the purple ones liked the system best; triangular users had the most trouble with it). Many of the comments can be collapsed into a few basic themes which are discussed in detail below. In addition the author provides a brief reply to these sentiments, not just to defend some of his design decisions, but also to put the comments into context with deeper issues of human computer interaction and control.

4.1.5.1 The Physical Interface Needs Improvement

One of the most popular complaints about all three versions of the Stretchable Music system was aimed at their physical interfaces, either a mouse or a pair of gamepads. Reactions to these interfaces ranged from mild discontent among many users to one participant out right refusing to use the system as a matter of principle ("I don't play mouse instruments," he flatly replied to my invitation to a demo). Some users complained about an inherent "unmusicality" of these interface devices while others were concerned with more practical matters such as the fact that a mouse only allows one musical control to be adjusted at once. In general people were more put off by the gamepad interface to the third system than to the mouse input of the other two. The gamepad's lack of continuous control did make it frustratingly difficult to manipulate the pointer on the screen. And, in general, people who had less experience with computers and video games were more apt to have problems with both the mouse and the gamepad.

The message here is clear and well taken. Both mice and video game controllers are frustratingly poor physical interfaces to the rich, virtual worlds that can be simulated on a computer. Musically or otherwise, neither type of device provides the number of simultaneous controls or the degree of subtly required to provide satisfying interaction in a virtual environment. These deficiencies are probably only exacerbated in the musical context of the systems; people (especially musicians) may be more used to having a greater degree of control when dealing with music. The author can only counter with two points: a) that the design of a superior physical input device was not the focus of this project and b) that general interfaces, despite their inadequacy, do in fact make the system immediately accessible to a greater number of people. One might raise the valid argument that its impossible to divorce the physical input device from the rest of the system in the design of an interactive experience. The author would then reply that there are only so many hours in the day, and that he wanted to make sure to leave room for future master's theses by not solving all interactive music's problems at once!

4.1.5.2 More Musical Control

Many people commented that they would like to have more control over the music that the system produced, but these comments were divided into a few different types. Some users wanted the system to react to their input more like an instrument. They wanted detailed control over their own real-time expression through some unspecified extension of the graphic interface. Such responses were typically vague and offered some ambiguous solution like this, "would have been fun if we could have had more of a chance to actually 'paint' a piece of music." Another set of users was interested in gaining more access to the mechanics of the system itself. They called for an editor with which they could create their own objects, music and controls, to essentially program their own versions of the piece. The final group simply wanted more detailed, and perhaps literal control over the music objects in the piece. They wanted to be able to turn the music objects on and off at will as well as access to more real-time control of the musical content in each object.

The first two of these three groups of comments were trying to push the system back into the comfortable shoes of a tool designed to enable users to express their own ideas musically. Interestingly, these two areas of suggestions fall on opposite sides of a clearly drawn line in conventional computer music systems between performance/instrument systems vs. editing systems. The author acknowledges that it might be interesting to push the system in either of these two directions, especially in a commercial situation where consumers might receive many more hours of enjoyment by being able to compose their own music in addition to simply exploring a piece written by someone else. Nonetheless, the addition of these features would have to aid users in creating their own musical pieces specifically designed for computational medium in the style of Stretchable Music. For the system to evolve into a glorified sequencer or generalized algorithmic instrument would be a tragic misinterpretation of the most fundamental ideas expressed at the very beginning of this thesis. Besides, there already is an interface available which allows users to create their own musical objects or even to design totally new kinds of interactive music systems. It's called a compiler and should frankly receive more attention from the masses of artists and content developers ceaselessly clamoring for new and improved tools.

4.1.5.3 Graphical Representation is Good

In general, users commented favorably on the graphical representation used by the system. The direct, one-to-one connection between the actions of stretching and the resulting manipulation of some musical parameter was deemed one of the system's greatest strengths. Surprisingly, users were not terribly bothered by the abstract nature of the visual representation. Very few asked for the objects to look more like an animated drummer or bass player. Many users had their own ideas about how the representation should have been done and various opinions on how well the author's visual choices corresponded with the musical material. Of particular contention was the mechanized drum robot of the second section of the third version of the system. Some users thought the visualization of that object was innovative and creative while others felt it was out of place among the other more organic looking objects in the rest of the piece. Of almost universal appeal was the main_bass object of the second version of the system. Something about the way users could induce wild gyrations in the object accompanied by similar gyrations in the music no doubt contributed to the popularity of the object.

4.1.5.4 Other Comments

Several other comments that popped up during informal conversations about the system with different users don't fit into a clear category but are listed here for completeness. Many people seemed to like the music in all three versions of the system. The most popular was probably the second version of the system, possibly because of its strong, catchy melodic content. Users also liked the accessibility of the music, citing as pluses a regular beat that you could tap your foot to and a more traditional song form. However, users often desired more, and more diverse, music in the system. After a couple of listens, one discovers most of the main musical ideas, so it would be nice to include lots more material for users to explore. On a different note, users were united in their appreciation of the multi-player facility incorporated into the third version of the system. Aside from the occasional confusion about who was controlling which pointer, users enjoyed being able to collaborate together on the same piece, often trading instructions like “you go over there and grab that while I get this line over here.”

4.2 Significant Points of Analysis

This section highlights several points which the author felt were most significant to analyze in evaluating the system. These assertions are the result of informal observations on the part of the author of other users playing the systems as well his own opinions on what worked and what didn’t.

4.2.1 Who the Systems Worked for and Why

Because the systems were designed with the likeness of a video game in mind, it is no surprise that that they enjoyed the most popularity among video game savvy males in their teens and twenties. Moreover, the strong stylistic leanings of the music in the systems made them much more accessible to fans of popular electronic music, many of whom tend to be under thirty. As mentioned above, many people had trouble with the physical controllers of the systems, especially the game controllers used in the third version of the system. Again, younger people with more video game and computer experience tended to have a much easier time overcoming the hurdles of the basic interface to enjoy exploring the rest of the system. Anecdotally, the author’s parents, unable to manipulate the input controls at the premier of the third system at *Cubestock*, stared in frustrated bewilderment as young children piloted the system with seemingly effortless elegance.

On a different note, though boys were more likely to dive right into the system to begin figuring out the controls, given the chance, many girls found the system enjoyable to play. This trans-gender appeal (appeal to girls and boys, not necessarily just to transgenders) might be attributed to the universal appeal of music among the genders, but may also stem from the exploratory, non-confrontational nature of the systems. Users are more apt to cooperate with each other to create interesting musical textures and to discover new features of the system than they are to compete with each other for control of the various musical objects. In fact, in the third version of the system, some of the objects incorporated musical controls that could only be fully realized with two users cooperating on the same object simultaneously.

4.2.1.1 Amateur vs. Skilled Users

The previous chapter mentioned that one of the design goals of the Stretchable Music systems was to create a musical control interface that was accessible to amateurs but responsive to more skilled users. Such an interface would have a steady learning curve to gradually allow users to become better while having fun at the same time. Nonetheless it was deemed important for there to be a noticeable difference between the performance of a beginner and an expert on the system. These goals were realized with limited success in the final version of the system. The system was very accessible to novice users, who, because most had never seen anything like it before, made up the bulk of the user pool. After a short explanation of the controls, beginners were off exploring the functions of the different objects and messing around with the tools. Some music controls had more obvious effects than others; occasionally the more subtle controls were lost in a dense audio mix. But there were enough clear mappings that beginning users could get some satisfaction immediately. Users skill level at the system also improved over time as they discovered what all of the musical controls could do. Performances by more experienced users sounded markedly different from those of the beginners. Users learned the functions of the tools and began to apply them at musically relevant times, for instance dropping the bomb tool on the bass object to give it a one bar solo at the end of an eight bar phrase. Users also learned to use their musical controls more sparingly, accenting passages at musically relevant times rather than pulling controls to their fullest extant to discover what they did. Though users could continue to improve their skills with the system, they would approach a ceiling limiting the amount of control they could exert over the music. As expressed earlier in this chapter, the system was not ideal for performance by skilled users due to this limited amount of control. By incorporating even greater depth and variety in the control of the system, future designs may better be able to balance the demands of amateurs and experts with increased room for improvement.

4.2.2 What Kinds of Musical Controls Worked Best

The design of the Stretchable Music system definitely emphasized to the author what kinds of musical controls work the best when left to whims of amateur participants, and what kinds of musical controls didn't. The first major distinction of control comes between the levels of control at which users are allowed to affect the system. Should the users have direct control over the arrangement and execution of musical material (such as that provided by more traditional instruments such as a piano)? Or should the user's primary task be to shape and/or select musical material provided for them by the designer of the system? What kind of influence should the user wield over the greater structure of the piece, such as choosing when themes come in and out and deciding in what order major sections are played? Can high-level parametric controls on an algorithmic generation system give users an interesting musical space to explore, and more importantly, provide a fun and coherent musical experience?

The Stretchable Music systems pretty much took the middle ground among these levels of control by providing users with a strictly enforced musical direction (discussed in a later section on musical structure describing the object_score mechanism) and well defined musical material to manipulate and shape. Much of the reason these levels of control were chosen was to define the system as more of piece or a carefully designed interactive experience, than an instrument or a tool for the user to invent his own music. Stretchable Music employs a model not unlike a six lane highway, where the composer decides the basic route, decides when on and off ramps come, and provides the basic musical material, but allows the user to do the actual driving within the six lanes provided. Such a design was thought to provide an aesthetic coherence to the experience both on the structural level and on the level of interacting individual musical motifs: most of the musical controls were carefully composed so they would sound good together no matter what the user did. Hence the user's control in the system was not so much as making their own music, as articulating the music of someone else.

Given this level of user interaction, several types of musical controls proved to be very effective. Some of the most successful controls on many of the objects were timbral controls that allowed users to shape the way a sequenced musical line was articulated. All three versions of the system made extensive use of resonant filters to both shape the sound output of particular lines and to mix them in and out. Not by coincidence, it turns out that timbral parameters, especially resonant filters, play a major role in articulation and control of most contemporary popular electronic music. Other timbral parameters employed included FM modulation, ring modulation, filter resonance and cutoff frequency, envelope shape, LFO modulation, and equalization. Another successful method of control was allowing the user to mix between various layers, or even enabling them to turn these layers on or off. This method proved to be very useful for allowing users to control percussive layers, where timbral controls on the sampled instruments were not always readily available. Some basic algorithmic controls were used such as the arpeggiation control on the main_bass object in version two of the system, but algorithmic controls in general were avoided because of their difficulty to fine tune to particular musical contexts. On a final note, due to the stretching mechanisms implemented in versions two and three of the system, all of the musical controls would return to a steady state after a user released the object from his or her grasp. This had an effect of keeping the piece on the composer's defined track while allowing users to add flourishes and articulation to the lines as they chose, effectively acting as a pliant guard rail on the composer's six lane highway.

4.2.3 A Word About the Graphical Representation

As mentioned previously in this chapter, the graphical representation and aesthetic of the system was deemed to be one of its strongest points. Users didn't seem to mind the abstract nature of the graphical controls they manipulated to affect various parts of the music. Two key factors, in the mind of the author, contributed to the success of these graphics in the system. The first was one to one mappings. As often as possible, musical events, whether triggered by the user or otherwise, were represented by graphical animations on the objects. Every time a note would play out of the amoeba like main_bass object of the second version of the system, the center cytoplasm would flash white. Similarly, the fish representing the ethereal synth line of the same system would flash briefly for every note they played. The first version of the system took this correlation further by directly mapping the controls a user could manipulate to parameterized graphical output of the music objects. Even in later versions of the system, literal direct manipulation of objects by stretching was almost always mapped to clear, obvious musical controls. This connection was direct but due to the abstract representation not literal in most of the objects, thus achieving a pleasing balance of clarity and sophistication.

The second successful quality of the graphics was indeed their abstract nature. The visual design of the system was intended to spark the user's imagination and heighten their curiosity as much as to enhance the aesthetics of the music. To the author music seems to be a very abstract and fuzzy medium to begin with so it seemed only natural to represent it with somewhat psychedelic shapes and objects. One criticism leveled at this design was that many of the objects suggested a certain level of personality or character quality that could not be backed up by suitably sophisticated and convincing behaviors. In response to this, it is important to point out that the objects were only supposed to be caricatures of real creatures such a centipedes, amoebas, and fish, leveraging off the ideas they bring to mind to achieve their aesthetic effect.

4.2.3.1 Relative vs. Absolute Control Spaces

Worth pointing out is the fact that all of the graphical controls of the system were measured relative to objects' positions rather than in absolute screen coordinates. In other words, it didn't matter if you were stretching an object in the lower right hand corner of the screen or the middle, only the distance from the point being stretched to its original position on the object was measured. This avoidance of spatially absolute controls made the system more visually flexible. Many different control spaces could run on the screen simultaneously, constantly changing their spatial relationships to each other while creating a pleasantly evolving and dynamic visual texture. Clear problems arise from the application of the other extreme. Imagine that certain properties of objects, such as how loud or aggressively they play, were controlled by their horizontal position on the screen. In order have all of the objects play at the same intensity, users would have to arrange the objects in a vertical line, which would severely limit the visual dynamic of the rest of the screen. Higher-level correspondence between the arrangement of objects on the and such musical parameters could provide interesting territory for new mappings, but the literal correlation of absolute position to musical control seemed too heavy handed and cumbersome to be used in these systems. Mobile, object-relative graphical control spaces were deemed a more effective way to parse up a complex musical control into a dynamic visual interface.

4.2.4 Individualism Among Objects

One of the most important contributions of Stretchable Music was undoubtedly its specialized, and diversified, approach to designing a control interface for an interactive music program. All of the musical functionality of Stretchable Music could have been controlled through a bank of faders, knobs and buttons on the screen. This approach is popular among many mixing, sequencing, and sound production software tools available today. Yet, by breaking these controls up along conceptual boundaries of musical role, and by reinforcing these groups of controls aesthetically with specialized graphical interfaces, the system takes on a whole new musical feel that is less mechanical and more inviting to exploration. Hence the individuality of the control mechanisms of the different objects do not just contribute to the aesthetic diversity of the system, but in fact may make the control space easier to parse, especially for an uninitiated, novice user. In addition, object specialization provides a convenient way for a composer to limit the control of the different parts of the system without frustrating users. The standardized interface of a fader bank would probably highlight an incomplete or limited control set, while the unique interfaces of stretchable graphical objects turn this control set into an interesting space for the user to explore. Composers can use these limited control spaces of each of the different objects as a convenient way to carefully control the musical space in which the user is allowed to play. This is a usefully way for composer to apply an overall aesthetic direction and control to the sound and feel of the piece. Composing the Stretchable Music system was as much about defining the musical capabilities and feel of its instruments as it was about writing music.

4.2.5 What About Structure?

Another extremely important innovation in the system was the use of the object score mechanism to impose large scale time structure on the musical experience. One of the toughest things to do in an interactive music experience [Waxman 95] is to create this flow of time and diversity of music. You may be able to count on a skilled improviser to take basic musical controls and shape them into a coherent musical whole, but an installation designed for the general public must provide some structure of its own. Furthermore, large scale time structure of an interactive piece is arguably one of the most important contributions of the composer to the interactive experience. The composer should provide the basic materials for the user to manipulate/explore, but should also impose the progression of these materials through time. With out this attention to progression and structure the piece will seem static and lacking of that essential musical quality of time flow. One of John Cage's favorite ways to think about musical composition was as slots or bins of time, into which indeterminate material could be placed [Griffiths 81]. This view of composition fits very well with the time scale architecture implemented in the second two versions of the Stretchable Music systems.

The very relevant question then becomes, how successful was the time structure imposed by the object score mechanism and how might this have been improved? Users did not seem to mind too much that objects appeared and disappeared according to an invisible background score, even if they did so while a user was trying to manipulate them. Occasionally users would get confused by these transitions, thinking that they somehow mistakenly triggered the objects to disappear or appear. The biggest problem with these transitions, in the mind of the author, was their abruptness. Some type of visual cue to alert the user that an object was about to appear or disappear might have been a good idea. The original specifacation for the system actually provided for entrance and exit animations for each of the objects to make their transitions from on and off screen seem smoother. For instance, instead of suddenly disappearing when it receives and "EXIT" message, an object might spin rapidly and fly of the screen or fade into the background, to make the visual transition smoother. Of course these transitions should vary from object to object to remain consistent with the theme of individualism among objects.

Another feature that was deemed successful by the author was the crude navigation mechanism employed by the third version of the system to allow users to advance to the next major section of the piece or stay in the same section for another loop. Structural control on a macro level seemed to be an important constraint to impose by the composer on the piece, especially in ordering of sections. But giving the user the ability to modulate the rate of his or her progression through these sections seemed like a worthwhile option of control to leave into the hands of the user. Another model that might have been interesting would have been to let the user choose which sections to visit, effectively remixing the ordering of the major sections of the piece. This might be thought of as analogous to letting listeners listen to songs on a recorded album in any order they choose. One could imagine a high-level menu world that would allow users to fly to any of many different pieces. Perhaps like in many video games, the ordering of the user's progression through this world would be modulated by the completion of certain pieces: some songs become available only after others have been successfully completed.

4.2.6 Goal Structure?

With all the talk of video games it can hardly be appropriate to conduct a discussion of an interactive music system based on their model with out considering the goal structure that plays so prominently in their design. Most video games motivate users with two major devices: accumulation of points and the threat of death. Users are rewarded for accomplishing various tasks with points that accumulate as their score (which then provides a crude measure of their success in the game). Likewise, users are often punished for not achieving a basic level of competency in the game by having their virtual player eliminated in some typically gruesome way (the more gruesome the better). Many games use the latter mechanism to retard the progress of the user through various levels, the theory being that part of the challenge is for the user to see how far he or she can get before death. Other games forgo both the death and the points models in favor of various puzzles and tests of skill that allow users to pass on to successive levels. In these games, the point is to explore the various carefully designed (and often very aesthetically interesting) levels, seeing how far a user can progress using wit and skill alone. Of course video games exist with all combinations of these goal structures and constraints. The most important fact to note is that some kind of goal structure is present in just about every video game you will find.

Yet if Stretchable Music is a video game then here it decisively breaks ranks with the rest of the crowd. One might argue that the goal of Stretchable Music is to enjoy oneself and produce an interesting rendition of the piece, but unlike most other video games, there is no explicit measurement of the user's success towards this goal incorporated into the system. Users are only limited in their progression through the piece by time; in the third version of the system, special icons enabling the user to advance to the next section appear only at the end of a section. The real question is whether this kind of goal structure is applicable to interactive music systems at all or whether it is an artifact of more traditional video games and should be completely left out of the picture. One useful function the goal structure does provide is to limit the progress of the user and thus extend the interest period of a particular piece of interactive software. Further, by making a user work to solve a puzzle or acquire skill through a particular interface, a goal structure also gives the user a feeling of accomplishment at the end. In such a system, the piece is not just like a tour through an art museum but a challenging experience that required some serious effort to get through.

From this perspective, adding a goal structure to Stretchable Music or other interactive music programs doesn't sound like all that bad of an idea. The real question is how to impose this structure so it doesn't trample the user's musical experience and how to measure quantitative musical results in a relevant and meaningful way. Potentially, users could be graded along crude axis such as their activity or accuracy of timing, or whether or not they make use of certain features at particular points. Such mechanisms of merit could then serve as keys to allow users to progress on to various songs in a large musical structure such as that described in the previous section. The draw back of such an implementations that it would impede creative musical use of the system by imposing perhaps arbitrary musical goals on the aesthetic outcome of the piece. The quality of a user's creative use of a system for musical ends would be almost impossible to effectively measure. Hence the problem of incorporating a goal structure into a piece of interactive music remains a sticky one, and probably explains why it was avoided altogether in three versions of the system described in this thesis.

4.2.7 Commentary On Production: This Should Be a Team Sport

On a final note, it goes with out saying that the development of an interactive music system like Stretchable Music draws on several different kinds of skills including visual design, music composition, and computer programming. The author simply wants to point out that this is an awful lot for one person to do and that future development of such systems might benefit from the video game model where an entire team of technical and artistic specialists work together to produce a final outcome (another example of this sort of collaboration comes from the movie business). The challenge in an interactive music setting, almost more so than in conventional video games, is to marshal such a team to create a tightly woven and aesthetically coherent piece in the end by fusing the input of everyone on the team. This would no doubt be an arduous task, but one well worth the extra organization and personnel-power if truly full fledged, “feature length” interactive music systems are to be produced in the future.

4.3 Summary

This thesis set out to define a new type of interactive music system that is less of a tool than a composition. The previous chapter discussed three systems designed from this standpoint covering in great detail the aesthetic and technical design decisions that were made during their creation. The present section has analyzed these systems from the point of view of users’ comments as well as the author’s own criticism while suggesting in what venues the systems are most at home. To conclude this analysis, this section will briefly summarize the most important points discussed above in an attempt to draw broader conclusions that may be useful in the design of future interactive music systems of this ilk.

Though the Stretchable Music systems were shown in a variety of venues, the most appropriate seemed to involve a small number of people in a semi-private setting. The depth and complexity of the system coupled with its novel control interface required the users’ concentration to figure it out. People felt more comfortable taking their time to discover these controls in the office-demo/living room setting than in a public installation. As a control interface for a performance system, Stretchable Music proved to be too limited (especially with respect to improvisation) for the author’s taste. But some of its ideas regarding the direct correspondence between a performer’s musical gestures and a graphical display may prove useful in the design of future performance systems.

Users, on the whole, responded positively to the system, though they had two major complaints. First, they disliked the awkward, general physical interfaces of the systems for musical control. Secondly, they were divided over the types and levels of musical control that should be provided by the system. To some extent, these complaints represent users trying to fit the systems into more comfortable, traditional concepts of music making devices (namely instruments and tools). However, users’ enthusiastic acceptance of the abstract graphical representation and control interface used by the systems indicates that they are willing to explore this new form, even if they did not think of it as such at the time.

Finally, the author highlighted a series of issues that he felt had an important impact on the successes of the systems. He pointed out first of all that the system had a broad general appeal among young people of both genders, especially among those who were fans of electronic music (hence the system made a connection that was clearly musically based). Musically, the author found that a mid-level range of control worked best, where user's shaped pre-composed musical material rather than being relied upon to create their own. For this level of control, timbral controls proved to be some of the most interesting. Like the users, the author believed that the abstract graphical representation in the system was a good way to parse up the control space. It made the interface interesting to explore, aesthetically engaging, and fun. Further, the individualism and diversity among objects' graphics, music controls, and musical content was thought to be a big win. Finally, the author pointed out that the large scale time structure imposed on the system by the object-score mechanism (utilized to its fullest extent in the final version of the system with 4 major musical sections) was quite an important development to make the interactive experience flow with time.

All of these points helped make the systems active and engaging musical experiences that drew constant participation from users in real-time. Thus they can serve as the guide posts for a design methodology that will influence future interactive music systems of the same style. Designers must be aware of what the user is doing, the breadth of choice available, the level and kinds of musical control, and most importantly, where the experience is going and how it changes over time. Further, he must go through great lengths to ensure that design decisions are rendered with contrast and definition, reinforcing the musical and graphical aesthetics of the system. Many of these lessons take on greater relevance in the next section when future applications of the Stretchable Music systems and the author's future artistic goals are discussed.

5 Future Work and Conclusion

As they stand now, the Stretchable Music systems are still only a hint of what is possible in the rich, and largely unexplored new area of expression in interactive musical experiences. This section will attempt to peer into the future to see what some of these experiences might be like, both as a direct extension of the model provided by Stretchable Music, and based on more general principles the author has identified as fertile areas for exploration in this medium. Finally, the section will try to put the work described in this thesis into perspective in relation to its contribution to the field, the experience and lessons it imparted to the author, and its potential ramifications for the way people think about music in the future.

5.1 Future Versions of the System

Through the three versions of the system described in this thesis, the ideas behind Stretchable Music have come a long way. As mentioned at the beginning of Chapter Three, the concept of what the system was supposed to be evolved right along with the technical features and musical capability of the software. Hopefully, this iterative design process will not stop with this thesis. The following discussion outlines some promising directions for the next iteration of the system and offers a view of how this type of system might slide its way into mainstream society.

5.1.1 Improvements to the System

The following list of improvements is by no means exhaustive, and would probably be difficult to incorporate into the next version of the software all at once (barring the addition of a team of experts to the author's development effort). Nonetheless, they do represent a decent sampling of some of the more important and promising directions for the current version of the system.

5.1.1.1 Networking

As the multi-player facility incorporated into version three of the system showed, the interactive music experience can become a lot more fun when there is more than one person involved. Allowing different copies of the program to connect to each other over the Internet would be an ideal way to bring users together who may be separated by geographical or cultural boundaries. One model for this kind of connection would be a sort of a musical chat room based the client server model of many of today's networked computer games. Users could set up a music server that would coordinate the actions and music of many client programs running the same Stretchable Music software simultaneously. Servers could be based around particular musical genres or simply represent a place for a few friends to get together and jam. Individual servers might have very different feels both musically and visually depending on the kinds of musical objects that were on them. Perhaps users could even upload their own objects and musical content to the server, allowing the jam to become a hotbed for new musical ideas and improvisation. The technical challenges to putting such a system together are by no means trivial (many of them are discussed later in this thesis). However, building in the ability to connect people musically over great distances would be well worth the effort required.

5.1.1.2 User Customization and Programming

The potential of allowing users to modify or even create their own musical objects and material has been mentioned several times during this thesis. Though this idea of incorporating a tool-like interface and functionality goes against the author's original conception of the system as a piece of musical expression itself, it is nonetheless worth considering due to the potential value it could add to future users. Also, these two visions of the system need not be mutually exclusive; though some users would probably take extensive advantage of the ability to create their own content, common sense tells us that the vast majority of users will be satisfied to explore and manipulate someone else's content. This means that the artistic vision and content side of the program would probably have the greatest impact on its success with the general public. Nonetheless, it seems that the two modes of interacting with the system, namely editing and performing, should be separated into two distinct functional interfaces of the program. The author wonders how conceptually relevant editing and reprogramming would be in the midst of a real-time performance situation, and how such controls would be integrated seamlessly into the overall interface. Further, the question remains of what kind of controls would be accessible at this level and how much users would truly be able to customize the musical objects. Such questions are a delicate matter that must be considered with the balance of the overall user experience in mind. Clearly there can be several takes on this matter and it is a ripe area for future investigation.

5.1.1.3 More Music

The musical experiences provided by the systems have been getting progressively longer, more complex, and more diverse as the systems have evolved; the author believes this to be a good thing. It would be nice to make future versions of the system album length, in that they would incorporate at least 50 to 60 minutes of continuously interactive music played straight through. This amount of time, however, pales in comparison to the average of 25 to 30 hours it takes to make it through one of today's computer games. Part of the reason games take this long, though, is due to their pronounced goal structure, which was discussed in chapter four. These games usually don't incorporate 25 hours of content, it just takes the player that long to solve all of the puzzles or to acquire the skill required to finish them. Part of the appeal of a larger Stretchable Music would hopefully be that it could be played again and again much like normal music albums are. Artistically, the composer would want to carefully weave many subtleties into the piece so that users continue to discover new features or relationships with every pass. As suggested in the last chapter, a system that organized major musical structures or songs into different worlds to which a user could depart from some kind of index vantage point would be an interesting next step. Each of the worlds could have a different musical theme so that the entire system could cover a large variety of styles within the electronic music genre. Further, different worlds could allow users different amounts of musical control, making some easier to play while others would become harder, almost virtuosic show-off pieces. Such greater diversity in the music, visual aesthetic and degree of control would undoubtedly increase the life span of user interest in the system.

5.1.1.4 More Coherent Game Play

Rules that govern how objects interact, how users can effect objects, how users can use tools to effect objects, and how users can navigate between sections of the music all fall under the umbrella of game play. Game play can be distinguished from the direct action of stretching objects or the particular controls on an object, by the way it governs the flow of the system and the things that a user can do within that framework. Right now there is precious little game play incorporated into the system. Good examples of elements of game play currently present in the system revolve around the tools, which the user can choose to use or not to use. From the point of view of game play, the most interesting of these tools is the bomb tool, which actually modifies the behavior of the objects when it is used. The user experience could be made much more interesting and volatile with the introduction of more of these meta-tools that alter the behavior of other musical elements in the system but don't necessarily represent a direct musical control themselves. One could imagine other tools that would cause objects to play backwards for a bar, or advance the entire time in song to a bridge section or perhaps the end. One could also imagine a warp tool that would whisk the user away to a totally different section of the music only to return after a specified time. By using interesting devices such as these, the game play of the system could be enhanced to an interesting and exciting experience where many different levels of control are cleverly made available to the user.

5.1.1.5 Better Music Controls

One fertile area for future investigation is the incorporation of new types and levels of musical control of the musical content in the objects. The existing three versions of the system rely heavily on mixing and timbral controls for their interactivity, mainly because these allow the basic content of the system to be carefully structured. More sophisticated algorithmic controls that would allow users to influence the music on more of a note level than a sound level would add a whole new dimension to the user experience of the system. One major area such algorithmic controls would open up would be high-level musical controls where users influence the objects' musical output through broadly defined musical metaphors. This conductor-like level of control is briefly examined in the second version of the system using the sprinkles tool, but warrants much closer investigation, especially with algorithmic music generation techniques. Another area where new types of musical controls, especially algorithmic ones, would benefit the system would be by providing the capability for users to discover new musical output that was not explicitly included by the author of the system. Such facility that allowed for the discovery of new ways of using controls designed for something else would truly mark the design of a great system. It would also allow the systems' designers to get more bang for their buck, by not forcing them to meticulously preprogram every possible scenario into the system. Such a system would also have much more natural depth for the user to explore and contribute to the life span of interest of the system as described above. Unfortunately these types of controls are notoriously difficult to get right, so they constitute a promising but challenging area of future research for this system.

5.1.1.6 More Complex Object Behavior

Most of the musical objects in the current version of the system exhibit some kind of simple behavior that causes them to move about the screen in a remotely lifelike fashion. The reason this behavior was implemented in the first place was not so much to simulate the actions of living creatures (centipedes, fish amoebas), as to provide an interesting and evolving visual texture that reinforced the idea of motion in the music. Another interesting direction for future investigation would be to invent ways in which these simple behaviors could be more tightly linked to the musical output of the system. Such an investigation is certainly not easy because it's difficult to constrain literal mappings of object behavior and music to be meaningful in both domains. For instance, say objects are programmed to pause and play an alternate melody every time they run into each other. Because of the way objects crawl around on the screen, it would be impossible to regulate these collisions so they would come at musically meaningful points (or even intervals); the alternative of regulating the paths of the different objects would make the visual texture of the system hopelessly structured and inorganic. Hence, behavioral artifacts such as collisions have to be handled very carefully when applied to meaningful musical mappings in the system. The collisions, to continue the example, would better be mapped to random ambient noises that will add to the sonic texture but not detract from the music by their irregular rhythm. Other such clever mappings between object behavior and music undoubtedly exist and should be investigated in future versions of the system.

5.1.1.7 Software Synthesis

One obvious, yet challenging, improvement that could be made to the system would be to incorporate all of the music synthesis, which now takes place on external MIDI synthesizers, into the program itself. This modification would have clear ramifications on the portability of the program to different hardware platforms. For example, to implement the network strategy described above with the current system, each of the different participants in the networked experience would have to have an identical set of MIDI hardware, each copy of which, for the current version of the system, costs thousands of dollars. Any attempt to distribute the program widely on the Net would be unsuccessful due to this drawback. The sounds used by the system could be adapted to use some standard instrument mapping like General MIDI, but as the author has emphasized again and again, such a compromise would only make the music sound mediocre. A much better solution would be to implement a fast software synthesizer that could emulate the sounds and timbral controls offered by the MIDI synths in the author's setup. Such a synth would definitely eat up extra CPU cycles on the user's computer, but high many-end Pentium based machines that users now purchase for performance-oriented games could handle the added load.

5.1.2 Music Video Games of the Future

Clearly with a little more work, Stretchable Music or another similar interactive music system could be ported to the comfortable turf of present day video games in dedicated game consoles and home computers. However, the future of such systems need not necessarily be confined to the living room TV or the parlor PC. Interesting graphical interfaces combined with content-based interactive music systems might find their way into many kinds of public spaces and could provide a convenient and interesting way to play music across the Net. The most important attribute these systems would share would be a tight coupling of a graphical, aesthetically interesting interface to their musical output and controls. The mechanisms through which users might interact with such systems could range from interesting physical input devices to non-contact sensing technology including vision tracking and electric field sensing. Perhaps a user could stand on a simulated surfboard and navigate through a virtual musical world projected in front of him merely by shifting his weight. Or, imagine an interface where a user dances in front of a large projection screen where the motion of her limbs are transformed into streaks of color that interact with various musical/visual textures represented on the screen. Consider a video-game-like interface used to control a jukebox-like device, where users could download and select interactive music tunes by their favorite artists, mixing one into the next in a continuous stream of live, generated electronic music. One could definitely imagine such systems popping up in arcades and dance clubs of the future. Perhaps more subdued versions of these systems would appear in coffee shops, lobby spaces, and even art galleries as interactive sound art. Such pieces would have their own pleasing self-evolving behaviors but could be perturbed in one direction or another by input from a passerby.

5.1.2.1 Application: Online Club

One of the most ripe places for animated virtual interfaces to interactive music systems to be explored would be in an online environment such as a chat room or a club. Systems such as Stretchable Music would be ideal to provide real-time, expressive musical interaction between people in a way that is simply not possible with today’s text-based chatrooms. Further, an aesthetically rich graphical interface based on animated musical objects would be an ideal way to represent music and musical interaction in an online virtual environment. Rather that focusing on outdated visual metaphors based on acoustic instruments, such an interface to an online musical experience would focus on visually challenging and exciting aesthetics that would be far more comfortable in the online, computational domain. Such an online gathering place facilitated by one of the interactive music servers described earlier in this chapter could become a gathering place for people who are interested in the same kinds of music. Musical and visual content could be specifically tailored by the proprieties of the site so that each online venue would have its own particular style and feel that would be found nowhere else. Also as mentioned before, people could visit the site, not just to interact with the musical material there, but also just to listen to what’s playing and to chat with others who are there. Tightly knit online communities already exist that draw people together around political discussion, similar musical interests, or even popular video games. Adding real-time, live musical interaction to this mix could only enhance the online experience these communities already share. Who knows: maybe Neal Stephenson’s “Black Sun” is not too far off after all [Stephenson 93]!

5.2 My Future Work in Electronic Music

The author’s future work in electronic music is by no means limited to the further development of the Stretchable Music system. Rather, there are several key problems that he has identified that he would like to explore of the course of the next several years. This section characterizes the problem space which the author sees as most relevant to his future work and suggests several specific project ideas that he has in mind to explore this space in both the long and the short terms.

5.2.1 Problem Space

The problem space of interest to the author’s future work can be conveniently characterized by two axes (see figure below). The first spans the continuum between interactive installation and live electronic performance. This axis basically measures the degree to which most people in an audience are actively influencing the musical outcome of the experience. The two extremes of this axis are effectively demonstrated by the *Brain Opera* Mind Forest and Performance Space. In the former, the majority of the audience is actively engaged in exploring the variety of interactive instruments that make up the first half of the *Brain Opera* experience. In the latter, the audience sits passively and watches performers create music on stage in a more traditional format. The second axis, graphics/communication vs. music generation/control, differentiates systems mainly built to emphasize musical gestures through other stimuli (graphics, gesture, theatrics) from systems designed in a purely functional way to provide new levels of musical control. An example of the first extreme would be a multi-media performance given by northeastern electronic music band Emergency Broadcast Network (EBN). EBN utilizes video sampling and other video playback to sync large, projected images to their soundtrack during a performance. Most of the music, however, is prerecorded so the main interest in the system is the visual reinforcement of the musical ideas. On the other extreme end of the axis would be performances given on modular analog synthesizers during the 60's and 70's where performers would crouch in front of walls of rack-mount electronics improvising wildly creative new sounds on the spot but making no effort whatsoever to let the audience understand what they were doing.

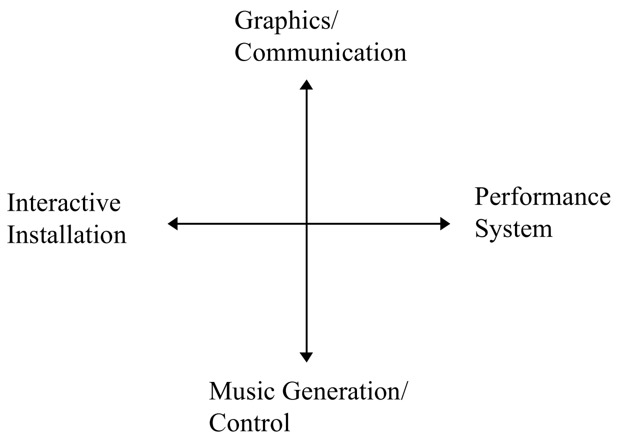

Figure 24. Future work problem space

Though these four areas of investigation are placed at opposite ends of two axes for ease of explanation, they are by no means mutually exclusive. Most systems overlap in several areas on these axes in a topology that is too complex to represent. The Stretchable Music systems, for example, might be represented in the upper left hand corner of the graph in their most common form of installation, but could easily be reflected across the y axis into performance as the last chapter demonstrated. The author does not intend to attack all of these areas at once in his future work, but will hopefully cover a few of the more important corners though the next couple of projects described in this section.

5.2.2 The Sonic 5: A Performance System

The author's first major project after finishing this thesis will be to design a performance system for his band, Earth vs. the Flying Saucers (EVFS). This system will be affectionately dubbed the Sonic 5 and will be developed over the next six to eight months (the initial version anyway). On the axis described above, the Sonic 5 will sit squarely to the middle right. The system will incorporate some of the graphical ideas in Stretchable Music to hopefully facilitate a new level of communication between an audience and the performer controlling the system. Musically, the system, also like stretchable music, will be riff-and-pattern-based, allowing the performer orchestrate various layers of the music as he winds his way through the piece. Unlike Stretchable Music, though, much more provision will be included to allow the performer to improvise. Without the constraint of having to make the interface understandable to a novice user, the system will try to make many diverse controls available to the performer. The controls will most likely be organized among the patterns available in the system. Again drawing from Stretchable Music's design, individual patterns and pieces of musical content will have their own musical controls that are custom tailored to their musical content. Hence, algorithmic devices such as melodic generators and arpeggiators will sit alongside sample based drum loops and sequenced patterns to provide a diverse palette of musical material for an improvising performer to work with.

Visually the Sonic 5 will use a variety of output devices including LED arrays, old CRT's, oscilloscopes, and Christmas tree lights. Its main mode of visual output, however, will be four or five 14" monitors mounted on speaker stands around the stage. These output devices are deemed superior to the more traditional medium of LCD projection because of their greater brightness and lower price tags. The Sonic 5 will simultaneously route live and prerecorded video to these monitors as well as generate graphics in response to musical control in real-time. The performer on the Sonic 5 will be responsible for choosing and routing these various graphics to the monitors during the performance. Individual musical objects will probably have their own visual output that can be mapped to any display at any time. Facilities will be included in the Sonic 5 to allow real-time input from a variety of sources, including MIDI, audio, and video information. Initially, this information will be used to mainly to allow other members of the band, who are playing more traditional instruments, to add their input into the Sonic 5's mix. Eventually, however, these facilities will be used to allow real-time audience interaction to help control the musical output during a performance as described below.

5.2.3 Ideas for Future Interactive Installations

Ideas for future interactive work center around two main types of situations. The first is a performance situation for EVFS shows, where the band will go through great lengths to get the audience involved in the music-making experience. The second is in the more tradition area of interactive sound sculpture with a heavy emphasis on computer graphics. One of the goals of EVFS (and one of its most difficult challenges) will be to figure out how to involve an audience in a live electronic music performance in a meaningful and fun way. Clearly you don't want to just throw control over to random members of the audience by, say, handing them a musical interface and telling them to do a solo. Further, you want to stay away from trying to give decision making power to the crowd such as allowing them to choose the next direction in the music (such types of audience interaction have been tried in interactive movies with disastrous results). The key is to make the crowd feel like they're involved while carefully modulating their input to make musical sense.

The author has dreamt up two possible ideas that may make large scale crowd interaction possible. The first is an interactive beach ball or some other inflatable device that can be tossed out into the crowd to bounce around. Parameters such as when the ball is hit by somebody, or how it is spinning, could be sent back to the computer to control various musical elements during a particular part of a song. This type of interface is nice, because it allows an entire audience to feel like they're involved with out having them all contribute at once. By owning the beach ball, the entire audience will be in charge of that part of the music; however, only one member of the audience will be able to contribute at once. And that contributing member will change frequently as the ball bounces all over the crowd. The second interface would incorporate an IR camera to detect the heat emanating from the crowd in front of the stage and display the image on a large projection screen behind the band. Video input is a particularly good method to collect lots of data about a crowd, and IR is even better, because it separates meaningful information about warm, moving bodies out from irrelevant background information such as the floor. Simple video analysis could reveal interesting data such as activity of motion in the crowd. Further, by displaying this video information back to the crowd, and adding, say musical objects into the video stream that simply respond to the level of intensity around them, members of the audience could actively participate in a virtual musical environment during the show.

On the more traditional end of things, EVFS and the author intend to investigate the possibilities of future interactive sound installations that would be at home in art galleries and raves alike. One of the first ideas they intend to explore is an installation that gains its input from a wall of knobs, levers and other common mechanical input devices for continuous control. The idea is that the wall would be as much a physical sculpture as an input device. Many different types of knobs would be incorporated, ranging from large wheels the size of a metro bus steering wheel to small knobs found on a stereo. The knobs would be chosen for their aesthetic appeal as well as their feel when turning them (some would have momentum and feel heavy while others would be light to the touch). Users would interact with the system by exploring all of the knobs and trying to determine their functionality. The musical mapping of each knob would have to be carefully tailored for a maximum amount of audible contrast, so that users would be more able to figure out what they were controlling. The musical mappings of the knobs should also correspond somewhat to their visual aesthetic. Large, heavy knobs should will mix in and manipulate substantial bassy sounds, while smaller delicate knobs will control high, light sequences. The system may also be hooked up to a multifaceted graphical display, projected for instance on the wall opposite to the knob installation. The key to making such an installation work, as always, will be extremely demonstrative mappings between controller input and musical and visual output.

5.3 Conclusion

This thesis almost began with the following analogy: "Interactive music is a lot like anti-gravity: it sounds like a great idea at first but turns out to be a bitch to implement." The idea of building interactive instruments for amateurs is not a new one, yet researchers have gone round and round trying to figure out the right way to approach this problem. The fact that fifty years into the digital revolution, the dominant electronic musical interfaces still look strikingly similar to their acoustic forefathers shows clearly that this problem is probably not a technological one, and has yet to be solved. The system described in this thesis represents but one more attempt to bridge this gap between active players and passive listeners with a new kind of interface. This time, however, the interface represents not just a set of controls for an interactive music program, but a fundamentally different way of thinking about the relationship of the composer, instrument, participant, and medium involved. The following section will wrap up this thesis, by reiterating what this new approach is, and why the author thinks it's different and important, as well as offering some broader ramifications for this way of thinking both to the author's view of himself as an artist and to others who seek to express themselves creatively in the computational medium.

5.3.1 The Significance of This Work

The work described in this thesis represents a different way of looking at an interactive music system. Rather than looking at the system as an enabling device such as an instrument or a mixer, this document advocates viewing it as a form of musical expression itself in the computational medium. The distinction between these two viewpoints is subtle but important. One goes about designing an instrument in a very different way than one goes about composing a piece. The instrument designer focuses on flexibility, ease of use, and most importantly, raw expressive power. The composer, on the other hand, is more concerned with the particular musical experience conveyed by his music. The designer of an interactive music system is both a composer and an instrument builder, but this thesis argues crucially that he should be a composer first. Systems designed from this standpoint, in the opinion of the author, will reach a broader audience than those designed as more traditional instruments, and will yield a more exciting and rewarding active musical experience.

The Stretchable Music project attempts to explore this idea through the design and implementation of several systems that were tested in real-world situations in a variety of venues. Through informal observation and questioning, the author determined that the vast majority of participants who tried the system enjoyed themselves and would be open to trying interactive music systems of a similar ilk in the future. In hindsight the author admits that a more formal survey, perhaps using video tape and standardized questionnaires, may have helped quantify audience response to the system. On the other hand, like any aesthetic investigation, the success or failure of the piece and the ideas behind it can really only be judged on an individual basis. Either it works for you or it doesn't. It is the hope of the author that the true measure of success is whether future designers recognize the strength in these ideas and incorporate them into their own work. Hopefully this will manifest itself in new types of interactive music experiences in years to come. With any luck the next dominant mode of musical experience in the next century may not be a spectator sport!

5.3.2 What the Author Learned

Without a doubt the author has learned an immense amount by developing this series of systems and has matured tremendously as a programmer. However, perhaps more importantly over the course of this project his view of himself as an artist has evolved fundamentally. An MIT undergraduate education prepared the author with a thorough set of skills and a mindset to approach the world as an engineer, solving problems from a logical and methodical standpoint. Somewhere midway through the project described in this thesis, the author realized that he could apply these skills formerly restricted to the technical domains of engineering and science, to artistic expression in music and the visual arts. You don't have to play the guitar or (the author's evil nemesis) the piano to compose and perform music; you can also program your musical ideas into a computer and have them realized in just about anyway you can imagine. The realization that the computer was as valid a medium for musical expression as any other fundamentally reshaped the author's viewpoint of himself as an artist and the validity of his skills as a programmer. All those programming classes you took in college won't just help pay the bills so you can make music in your spare time, they actually will allow you to express yourself artistically in a medium that is more powerful and versatile than just about any to come before it.

5.3.3 Expression Through Programming

Thus the author asserts that programming is a valid form of expression and is arguably the most valid or the only true form of expression in the computational medium. There is no reason to argue that it is any less elegant or some how aesthetically inferior to other modes of expression. These sentiments have probably arisen because most programming languages are not designed for artistic expression. They are designed by companies like Microsoft to write business software. Hence to express one's self creatively within many of them, an artist must first wade through tons of irrelevant technical documentation designed for business and other applications, to distill the meaningful functionality that will enable him to realize his piece. Artists' fear of this process and of the complex internals of most software today probably explains their overly heavy reliance on commercially available tools to produce their creative work. Many of these tools contain scripting languages that are nearly as complex as the raw code they are intended to hide, but along the way allow artists at least some access to the fundamental ideas of computation. The answer is not to make watered down, baby languages for artists nor to make every artist who intends to work in the computational medium get a degree in computer science. New interfaces must be developed that allows artists more streamlined access to the functionality that is relevant to them in the computational medium. Similarly, artists must stop whining about getting their fingers dirty (and sitting around and waiting for the next Photoshop plugin) and embrace programming as yet another tool in their toolbox for expression.

5.3.4 The Take-Home Message

People have been making music with what ever they've had around them since the beginning of time. Instruments and forms of musical expression have fairly closely traced technological development for most of history. The computer poses the most formidable challenge yet by being the most powerful and versatile tool man has ever invented. This thesis encourages others to look at the computer as a medium rather than as a tool. Hopefully, Stretchable Music will inspire others to do this and will help forward the development and exploration of musical expression in the computational medium. Such exploration, in the author's opinion is essential to keeping our increasingly technologically dominated world habitable and human well into the next century.

Bibliography

[Chadabe 97]	Joel Chadabe, *Electric Sound*, Prentice Hall, Upper Saddle River, NJ, 1997, p. 291.
[Chung 91]	Joseph Chung, *Hyperlisp Reference Manual*, MIT Media Laboratory, 1991.
[Dodge 97]	Christopher Dodge, *The Abstraction, Transmission, and Reconstruction, of Presence: A Proposed Model for Computer Based Interactive Art*, S.M. Thesis for MIT Media Laboratory, Cambridge, MA, 1997.
[E 97]	Jonathan E., Not Quite All About Eve, MicroTimes web site, 1997, http://www.microtimes.com/163/mm101.html.
[Eno 98]	Brian Eno, "Generative Music 1," SSEYO website, 1996, http://www.sseyo.com/.
[Gabriel 97]	Peter Gabriel, "Peter Gabriel's Eve," Radio Real World web site, 1997, http://realworld.on.net/eve/.
[Griffiths 81]	Paul Griffiths, *Modern Music: The avant garde since 1945*, George Braziller, New York, NY, 1981.
[Harmonix 98]	"The Axe Technical Overview," Harmonix website, 1998, http://www.harmonixmusic.com.
[Headspace 98]	"Beatnik DocPack 2.0," Headspace website, 1998, http://www.headspace.com/beatnik/developers/doc-2.0/index.html.
[Id 97]	"id Software," Id Software website, 1998, http://www.idsoftware.com/.
[Iwai 92]	Toshio Iwai, "Music Insects," Toshio Iwai Works web site, 1992, http://www.iamas.ac.jp/~iwai/artworks/music_insects.html.
[Iwai 96]	Toshio Iwai, "SimTunes," Toshio Iwai Works web site, 1997, http://www.iamas.ac.jp/~iwai/simtunes/index.html.
[Kram 97]	Reed Kram, "kite," Solution to MAS 964 Problem Set 5, April, 1997, http://acg.media.mit.edu/courses/spring97/mas964/users/kram/ps5/kite/particle.java.
[Laurel 89]	Brenda Laurel, "A Taxonomy of Interactive Movies," *New Media News*, Vol. 3 No. 1; Boston Computer Society, 1988.
[Laurel 91]	Brenda Laurel, *Computers as Theater*, Addison Wesley Publishing Company, Reading, MA, 1991.
[Machover 92]	Tod Machover, *Hyperinstruments: A Progress Report, MIT*, Media Laboratory, 1992.
[Machover 94]	Tod Machover, *Media/Medium*, Musical Score, Milan/Paris: Ricordi. 1994
[Machover 96]	Tod Machover et al., *The Brain Opera*, interactive, multimedia opera,1996 http://brainop.media.mit.edu.

[Machover 98] Tod Machover, *The Brain Opera*, CD recording, Erato Disques, Paris, France, 1998.

[Maeda 97a] John Maeda, “A Frame Work for Digital Expression*,” Digital Communication Design Forum*, International Media Research Foundation, Tokyo, Japan, 1997.

[Maeda 97b] John Maeda, “MAS 964 Problem Set #4,” Aesthetics and Computation Group web site, 1997, http://acg.medai.mit.edu/courses/spring97/mas964/ps4/index.html.

[Marrin 96] Teresa Marrin, *Toward an Understanding of Musical Gesture: Mapping Expressive Intention with the Digital Baton*, S.M. Thesis for MIT Media Laboratory, Cambridge, MA, 1996.

[Mathews 70] Max Mathews, “Groove - A Program to Compose, Store, and Edit Functions of Time,” *Communications of the ACM*, Vol 13, No. 12, December 1970.

[Matsumoto 93] Fumiaki Matsumoto, *Using Simple Controls to Manipulate Complex Objects: Application to the Drum-Boy Interactive Percussion System*, S.M. Thesis for MIT Media Laboratory, Cambridge, MA, 1993.

[Microsoft 97] “Direct Draw,” *Platform, SDK, DDK, Documentation*, Microsoft, Redmond, WA, 1997.

[Minsky 86] Marvin Minsky, *The Society of Mind*, Simon & Schuster, New York, NY, 1986.

[MIT 97] “Wearables ,” *Frames* Number 71, MIT Media Laboratory, 1997.

[Oliver 97] William Oliver, “The Singing Tree: A Novel Musical Experience,” S.M. Thesis for MIT Media Laboratory, Cambridge, MA, 1997.

[Paradiso 97] Joseph Paradiso, “Electronic Music Interfaces: New Ways to Play,” *IEEE Spectrum Magazine*, Vol. 34, No. 12, pp. 18-30, Dec.,1997.

[Pelletier 96] Patrick Pelletier, “Rogus McBogus Documentation,” Rogus web page, 1996, http://theremin.media.mit.edu/Rogus/.

[Rando 97] Rolf Rando, “LiveStyles Guide 1.0,” Thinkfish website, 1997, http://www.thinkfish.com/tsupport.html.

[Rigopulos 94] Alexander Rigopulus, *Growing Music from Seeds: Parametric Generation and Control of Seed-Based Music for Interactive Composition and Performance*, S.M. Thesis for MIT Media Laboratory, Cambridge, MA, 1994.

[Rowe 93] Robert Rowe, *Interactive Music Systems*, MIT Press, Cambridge, MA, 1993.

[Woo, Neider, Davis 97] Mason Woo, Jackie Neider, Tom Davis, *OpenGL Programming Guide: Second Edition, Addison-Wesley Developers Press*, Reading MA, 1997.

[SSEYO 98] “Koan Generative Music System from SSEYO,” SSEYO website, 1998, http://www.sseyo.com/.

[Stephenson 93] Neal Stephenson, *Snow Crash*, Bantam Spectra, New York, NY 1993.

[Strickon, Paradiso 98] “Tracking Hands Above Large Interactive Surfaces with a Low-Cost Scanning Laser Rangefinder,” Appears in *CHI98; Extended Abstracts*, April 1998

[Waxman 95]	David Waxman, *Digital Theremins*, S.M. Thesis for MIT Media Laboratory, Cambridge, MA, 1995.
[Winkler 98]	Todd Winkler, *Composing Interactive Music Techniques and Ideas Using Max*, MIT Press, Cambridge, MA 1998.
[Yavelow 89]	Christopher Yavelow, "Music and Microprocessors: MIDI and the State of the Art," *The Music Machine*, MIT Press, Cambridge, MA, 1989.
[Zimmerman 95]	Thomas Zimmerman, *Personal Area Networks (PAN): Near-Filed Intra-Body Communication*, S.M. Thesis for MIT Media Laboraroty, Cambridge, MA, 1995.