PLAYING WITH TIME: INTERACTING WITH TIMED MUSICAL MACHINES

Konrad Kaczmarek

A DISSERTATION
PRESENTED TO THE FACULTY
OF PRINCETON UNIVERSITY
IN CANDIDACY FOR THE DEGREE
OF DOCTOR OF PHILOSOPHY

RECOMMENDED FOR ACCEPTANCE
BY THE DEPARTMENT OF MUSIC

Advisor: Dan Trueman

June 2015

This essay, the compositions Window’s that Won’t and Loligo, and the performance environment Metamorphic_Gestures constitute the dissertation, but are otherwise unrelated.
Abstract

Machines that capture and play back sound invariably influence the many ways that music is composed and performed. This symbiotic relationship is manifest in the turntablist virtuosically navigating the sounds engraved on a record, the guitarist generating overlapping rhythmic patterns and loops with a delay pedal, and the electronic musician manipulating and triggering samples live on a laptop. The tools and techniques used to record, edit, sequence, and play sound inevitably percolate into live music-making practices in the form of timed musical machines. These virtual machines involve both software and hardware, and incorporate varying degrees of automation, sound manipulation, and physical performance-based control.

This dissertation outlines the basic digital tools that comprise these virtual machines, providing examples of software to illustrate the underlying mechanisms, and identifying strategies for how to interact with them in musically meaningful ways.

The introduction frames the discussion with two contrasting anecdotes of interactions with recording and playback technology, and identifies the key components of timed musical machines. The first chapter investigates the ways that recorded sound is understood and manipulated as an object, first in the general case and then in applications that are unique to the digital medium. The second chapter identifies the specific building blocks and corresponding programming strategies of three classes of timed musical machines: sound file granulation, delays, and phase vocoders. Each category includes examples in the form of accompanying software. Finally, the third chapter presents
freezing time as a case study to highlight the techniques used to manipulate the sound object and to identify some of the perceptual shifts that timed musical machines create in the ways of attending to musical sound.

Since software-based timed musical machines operate on the micro-time scale, this analysis includes an investigation into the layer of automation and control that exists between lower-level processes and more direct user input. The discussion therefore includes an examination into editing and automation techniques derived from digital audio workstations and computer music programming environments, as well as an investigation into object-based compositional structures developed by the composer György Ligeti.
# Table of Contents

Abstract iii  
List of Figures vii  

0. Introduction: Broken Records  
0.1 Real 1  
0.2 Imaginary 6  

1. The Medium  
1.1 Recorded Sound As Object 10  
1.2 Expanding Outward 12  
1.3 Expanding Inward 15  
1.4 Fragmented Time 18  
1.5 Adaptive Time: Lopsided Loops, DAWs, and TMMs 21  
1.6 Sound Object and Timed Musical Machines 26  

2. The Machine  
2.1 Overview of Virtual Timed Musical Machines 28  
2.2 Time Domain Musical Machines: Sound file Granulation 29  
2.3 Delay-Based Musical Machines 44  
2.4 Frequency Domain Musical Machines: The Phase Vocoder 57  

3. Freezing Time  
3.1 Freezing Time 64  
3.2 Net Structures 67  
3.3 Performed Listening 82  
3.4 Freezing Time: Conclusions 85
Appendix A. Software Documentation

Introduction 86

A1 kk_granular 88
A2 kk_ringbuffer 92
A3 kk_livelooper 93
A4 kk_Blendronomer 94
A5 kk_treeyo 95
A6 kk_polylooper 96
A7 kk_phasevocoder 98

Bibliography 99
List of Figures

Figure 2.2.1 Granular Synthesis .................................................. 31
Figure 2.2.2 Spectrogram of a Piano Note ..................................... 33
Figure 2.2.3 Non-Linear Playback of Piano Sample Using Granular Synthesis 34
Figure 2.2.4 Non-Linear Playback of Cymbal Sample Using Granular Synthesis 34
Figure 2.2.5 Synchronous Granular Scheduler With Eight Evenly Distributed Streams ..................................................... 37
Figure 2.2.6 Sonogram Image of Asynchronous Granular Scheduler Based on Text Matching ...................................................... 38
Figure 2.2.7 Asynchronous Scheduler based on Text Matching ............. 38
Figure 2.2.8: Asynchronous Scheduler Used in Windows That Won't .......... 39
Figure 2.2.9 Examples of Amplitude Windows .................................. 41
Figure 2.2.10: Mapping Used in Real-Time Sound-File Granulation With a Tether Controller ......................................................... 44
Figure 2.3.1 Tape Delay ............................................................. 47
Figure 2.3.2 Ring Buffer ............................................................. 48
Figure 2.3.3 Delay-Based Live Looper ........................................... 49
Figure 2.3.4 Multiple Delay-Based Loops ........................................ 51
Figure 2.3.5 Secondary Delay Applied to Loop-Based Delay ................. 52
Figure 2.3.6 Delay-Based Loop with Secondary Delay ......................... 52
Figure 2.3.7 Rhythmic Patterns Generated by Delays in the Piece Tree Yo 55
Figure 2.3.8 Polyphonic Looper .................................................... 56
Figure 2.4.1 Phase Vocoder Schematic ............................................ 60
Figure 2.4.2 Temporally Quantized Random Frame Resynthesis Using the Phase Vocoder Technique

Figure 3.1.1 Temporal Dither in Sound File Granulation

Figure 3.2.1 Examples of Filling, Accretion, and Shifting from Ligeti’s Continuum

Figure 3.2.2 Piano Roll View of Continuum: mm. 1-66

Figure 3.2.3 Piano Roll View of Continuum: Full Score

Figure 3.2.4 Sound file Granulation Using Accretion Method at Various Granular Densities

Figure 3.2.5 Manipulating Granular Density with 3-Note Signal

Figure 3.2.6 Accretion, Filling, and Shifting Using Continuous Pitch Space

Figure 3.2.7 Temporal dither at 20ms and 100ms

Figure 3.2.8 Range of Temporal Dither Controlled by Amplitude Envelope

Figure 3.2.9 Granular Dyads and Chords Using Multiple Record Buffers and Feedback

Appendix A:

Figure 1: Non-Presentation View of kk_granular Patch

Figure 2: Presets Window

Figure 3: kk_granular User Interface

Figure 4: Buffer Information

Figure 5: kk_ringbuffer User Interface

Figure 6: kk_livelooper User Interface
Figure 7: kk_blendronomer User Interface 94
Figure 8: kk_treeyo User Interface 95
Figure 9: kk_polylooper User Interface 96
Figure 10: kk_phasevocoder User Interface 98
• Introduction: Broken Records •

0.1 Real

When I was two years old my family moved from California to Connecticut. My grandmother flew in from London to take care of my sister and me for two weeks while my parents drove across the country with all our belongings. Having to watch over two inconsolable children in an empty house in a foreign country proved to be a particularly traumatic experience for my grandmother. To this day, she still tells the story of how I repeated, “Mommy’s gone. Daddy’s gone. Record player’s gone,” over and over throughout the two weeks. It wasn’t our pet rabbit, my collection of stuffed animals, the books, or even the toys that I missed. It was always just my parents, and the record player. I had spent countless hours straining to reach up and touch the spinning plate. I would press the buttons that toggled between speed settings and watch the illuminated red dots do a momentary dance before they reconvened on a different row on the plate’s edge. Then there were the rare occasions when I was allowed to drop the arm down on the record, under my parent’s close watch.

Today, more than thirty years later, I still listen to music on my turntable, and vinyl is experiencing another comeback. The tidal wave of instantly available digital music has practically wiped out the retail music store, transforming the music industry in the process. In spite of this change, independent record stores somehow continue to thrive. Tape decks and even compact disc players - once ubiquitous in home stereos, boomboxes, walkmen, and car stereos - have almost completely disappeared. In their
place, sleek, minimalist speakers magically pull sound out of thin air, streaming music from our computers, phones, or directly from the cloud.

So why does the turntable - unlike the tape deck or CD player - persist in the face of such dazzling new technology? Audiophiles will tell you that the analog sound is superior to digital audio, with a warmer and more natural tone. Record collectors will cite the idea of the record itself, and the fact that each one contains a carefully crafted and complete musical statement in the form of an album. While I agree that fidelity and the nostalgia for a bygone era in popular music play important roles, I think the main reason the record player continually defies obsolescence is that it has an accessible mechanical system at its core. Music is temporal in nature; it is fleeting and abstract. The turntable, on the other hand, is a machine that is open and intuitive. It is tangible, and you use your own hands to operate and interact with it. It gives you a certain amount of control over the music.

Again, when I was a child, I tried an experiment where I taped a sewing needle to the inside of a cone I had made out of cardboard. I held the cone in one hand, carefully laying the needle on the surface of the record, and spun the turntable around with the other hand. To my amazement, music rang out! It was quieter and thinner than the sound that came out of the loudspeakers, but it was there nevertheless. Even though I now know exactly how it all works, it still feels a bit like magic every time I play a record on my turntable. It is an entirely different kind of magic than wirelessly streaming digital audio or a tiny iPod stuffed with gigabytes of music. The turntable’s magic is in the
machine itself, and in the ways that you interact with it: the physical connection it gives you to the music.

Turn on the motor, drop the needle in the groove, and watch the perfectly balanced stylus hover over the record spinning underneath. The turntable’s motor provides the constant and unwavering rotation needed to play the sounds exactly the way they were captured in the recording process. Even the slightest fluctuation in speed is difficult to ignore, producing a fluttering in the sound that self-consciously draws attention away from the music and onto the turntable itself. Watch the needle inch closer and closer to the center as the record plays on. You can see the individual tracks laid out in concentric rings on the record, the thickness of each band corresponding to the duration of the song. There is a limit to what can be engraved on a record. While dynamic range and rotational speed play important roles, a record’s duration is ultimately limited by its physical size. The motor can spin forever, but the sound will inevitably stop. Like pouring water out of a pitcher, eventually you run out of record.

Put your finger gently on the edge of the spinning plate and feel the resistance from the motor as you hear the pitch waver and drop. Put your hand directly on the record, and spin it backwards a few times against the pull of the motor. Let go, and the record begins to spin again, as time - the motor - takes over. As long as the needle is in the groove, you will never decouple the movement of the record from the sound it generates. They are physically connected and continuous, like a figure and its shadow.
Now press harder, and the record stops. Silence. The turntable I use now stops abruptly when you switch off the motor with the needle in the groove, as if ripping the sound from thin air. It is like the exaggerated sound effect used in movies. Filmmakers understand how well it communicates a sudden - often comical - break in the narrative. For a moment, time stands still. Growing up, I remember our family’s record player slowly spinning down to nothing when you turned it off. Either way you arrive, it is silence as stasis.

A broken, skipping record partially ruptures the turntable’s link between sound and movement. The repeating sound appears first as a hiccup in time, and then as an infinite echo, returning over and over again with each rotation and subsequent skip backwards in time. The record keeps on spinning, but the needle does not progress toward the center. The locked groove produces a similar effect.¹ A short segment of sound engraved there will repeat forever until you lift the needle off. Even the hiss, clicks, and pops created by an older record repeat in precisely the same pattern in an otherwise empty locked groove. The broken record is a discrete hop in time. The locked groove is a continuous loop. In either case, it is sound as stasis. Repetition or pattern frozen in time, resisting change.

The turntable allows us to intuitively experiment and interact with some of the more mysterious qualities of sound, music, and time. The motor and the record serve as contrasting metaphors for the different ways that we can experience time in music. The constant speed of the motor represents absolute time, stubbornly marching on with each

¹ A locked groove is the closed circle at the end of each side of a record that prevents the stylus from moving into the label area.
rotation. The record itself symbolizes a more malleable form of musical time, in which sounding events dictate a more subjective flow of time. The machinery of the turntable allows us to navigate musical time either continuously by spinning the record forwards or backwards against the motor, or discretely by moving the stylus. Similarly, the location of specific sounding events and exact durations can be defined by visual cues on the record’s label as it spins around, providing a convenient link between absolute and musical time. Virtuoso turntablists have mastered these aspects of the machine, manipulating both absolute and musical times. By carefully adjusting the motor’s speed, they can match one record’s beat with another. They turn their physical manipulations into highly expressive musical rhythms, and then relinquish power back to the motor at precisely the right moment so as not to skip a beat. Anyone who has ever played a record at varying speeds or backwards, however, can tell you that it doesn’t take a virtuoso’s skill to get a sense of just how malleable and plastic time can be using a turntable. The mechanism itself invites you to interact and explore.

I use the term Timed Musical Machine (TMM) to define any sound-generating device that combines automation, mechanical precision, and human input or intervention. TMMs therefore include some form of temporal sequencer along with the actual mechanism that produces sound. Acoustic examples include simple, pulse-based percussive machines like the metronome, pitch-sequencing devices like the musical box, as well as more sophisticated mechanisms like the player piano, the barrel organ, and the pneumatically driven orchestrion. Machines such as the turntable, tape recorder, and

---

echo and delay effects constitute an important group of electro-mechanical TMMs that play and record sound stored on various media. Human interaction comes in the form of interactions with the machinery itself, as well as indirect manipulations of the automation process, recorded audio, or other aspects of the sound production. Examples of physical inputs include switches, triggers, toggles, buttons, potentiometers, sliders, and foot pedals, as well as more sophisticated controllers based on sensor data or physical gestures.

With the advent of electronic sequencers and synthesizers in the second half of the twentieth century, virtually any electronic sound-generating device could be converted into a timed musical machine. Examples include step sequencers, drum machines, low-frequency oscillators, arpeggiators, as well as the myriad of synthesis and control modules associated with voltage controlled synthesizers. Communication protocols like MIDI and network-based Open Sound Control that developed towards the end of the twentieth century allowed digital devices such as synthesizers, samplers, sequencers, computers, and embedded micro-controllers to function collectively as distributed TMMs. As a result, computer software and hardware increasingly provide the mechanical precision and automation that drives timed musical machines.

### 0.2 Imaginary

I first began experimenting with software-based TMMs out of necessity while studying composition in college. I was working on a piece called *sustain*, which is a composed soundscape set in the imaginary microscopic world of a grand piano. The goal of the piece was to draw attention to the wide range of sounds that the instrument can produce
acoustically by editing together close recordings of plucking, banging, and scraping inside the piano. While perfectly well suited for mixing and arranging recorded sound, I quickly realized that the digital audio workstation’s timeline-based editing platform was not the best environment for actually generating the sounds and textures the piece demanded. To create a sustained texture or drone from a single audio sample, for example, I found myself tediously repeating the exact sequence of splice, copy, paste, and crossfade using virtual tools derived from tape editing techniques. While I was drawn to the physicality that the visual interface brought to working with sound in the digital medium, I was frustrated by my inability to fully take advantage of the procedural and algorithmic nature of composing with a computer.

Although I had never done any coding, the visual programming environment Max/MSP seemed like the perfect tool to experiment with automating these highly repetitive audio edits. Instead of writing lines of code, I was able build a program in a modular fashion by creating boxes on the screen, each with its own basic function, and connecting them in different ways using simulated patch cords. I was immediately fascinated with the idea of visual programming, and set out to create a Rube Goldberg mechanism in the virtual world, complete with looping sound file playback, animated faders, and a screen full of toggles, sliders, and colored boxes for the user to interact with. I had created my own timed musical machine, a virtual broken record that could operate on a range of time scales not limited by a record’s size or the speed of a turntable’s motor. I added a synchronized volume automation to mask the pop that the record player’s needle made as it jumped back in time. To smooth out the resulting sound I copied and pasted several
instances of the basic program, making sure that each volume automation curve was slightly out of phase with each other. I brought my work into a composition lesson, and my instructor noted that I had just created a very idiosyncratic - and not particularly elegant - example of granular synthesis.

Unlike the turntable or tape recorder, the mechanisms of the granular TMM were not immediately open and accessible. While the visual programming environment that I adopted initially provided a useful metaphor for manipulating digital sound, the significance of the patching system quickly dissolved as the program became more elaborate. Nested iterative processes, hierarchies of encapsulated code, and global and local variables had little to do with the visual objects and patch cords I started out with. Even though I had created the patch myself and was aware of which parameters were accessible to external control, I was only able to establish musically expressive ways of interacting with the process after a great deal of experimentation and modification to the underlying programming. While some of the synthesis parameters lent themselves well to one-to-one mappings like manipulating the turntable’s motor, other parameters proved to be more problematic. Adjusting parameters that were more idiomatic to the granulation process such as pitch spread and density, for example, involved simultaneously manipulating a dizzying number of interconnected variables and synthesis settings. Finally, the laptop computer is a generalized and multi-purpose device whose strengths lie in its hyper-precision, its ability to efficiently store and access data, and a procedural nature that applies to both scale and iteration. Beyond the traditional modes
of interaction – keystrokes and mouse clicks - the virtual TMM did not suggest a natural and intuitive means of interaction as with the turntable or tape recorder.

Despite these obstacles, software-based TMM have the ability to provide a tremendous amount of control over both the automation and the sound generating processes. Given a foundational knowledge of computer music programming and an understanding of the various ways of manipulating recorded digital sound, the inner workings of the machines themselves can become the objects of experimentation. They can be hacked apart, modified, scaled, and tinkered with in ways that are utterly unimaginable with physical machinery. Similarly, while the laptop itself might be limited in terms of its physical inputs, virtually any type of external control – for example MIDI or OSC information, sensor data, video tracking, motion capture – can be flexibly routed to drive these machines. Finally, the hyper-precision afforded by the computer provides access to the microscopic timings of both recorded sound themselves and the mechanisms that drive their playback.

What follows is an investigation into software-based timed musical machines that play and record sound. The goal is to define and categorize some of the digital tools that comprise these machines, to identify strategies for how to interact with them, and to investigate some of the effects they can have on how we attend to musical sound.
1. The Medium

1.1 Recorded Sound as Object

Like generations of musicians and composers before me, I grew up experimenting with recorded sound in various media. With the amplifier turned all the way down, I would listen to the needle on our family’s record player generate its own tiny version of the song, filtered by the physical characteristics of the medium and the mechanical system at its core. Instead of using the overdub on our four-track tape recorder to produce polished, multi-instrument songs, I would create dense and chaotic textures, layering instances of the same type of sound and then playing it all back at different rates, or in reverse. In both cases the technology allowed me to intuitively focus on sound as an object, providing a physical means to both manipulate and re-contextualize the sonic world around me. Experimentation with recorded sound using digital technology has led to increased temporal flexibility,\(^4\) which I explore in the following two sections of this chapter.

In his writing about *musique concrète* in the late 1950s, Pierre Schaeffer uses the term *l’objet sonore* to describe a sound that is removed from any physical context or association.\(^5\) Experiencing sound as an object forces the listener to adopt a more reductive style of listening, he argues, focusing on its intrinsic sonic qualities instead of simply identifying the source or meaning of the sound. In categorizing the various modes

---


\(^5\) Carlos Vicente de Lima Palombini, “Pierre Schaeffer’s Typo-Morphology of Sonic Objects.” PhD. Diss., Durham University, 1993. 41
of listening behavior, Schaeffer notes that, “the better I comprehend a language, the worse I hear it.”

Recording in any medium by definition removes a sound from its original acoustic source, transforming it into a potentially disassociated sonic object. Pioneers of early experiments with tape music quickly realized that working in that medium also provides a convenient physical manifestation of the sound object. John Cage commented, “it made one aware that there was an equivalence between space and time, because the tape you could see existed in space, whereas the sounds existed in time.”

The concept of the sonic object and reductive listening are at the core of many twentieth-century innovations in both acoustic and electronic music. In addition to musique concrète, the proliferation of extended technique, graphical notation, and an ever-expanding repertoire of un-pitched percussion music are to a large degree the result of a more comprehensive view of sound, putting timbre and spectral shape on equal footing with other musical parameters like pitch, harmony, and rhythm. As both composers and audiences began to adopt a more accommodating notion of sound and noise, the tools used to define, arrange, and manipulate sonic objects became more powerful and nuanced. Composers are still challenged to come up with strategies of incorporating

---

6 Palombini, “Pierre Schaeffer,” 53.
7 Brian Kane points out that Schaeffer’s original use of the word l’objet sonore reflects his philosophical objective of “disclosing the original ground of our musical practices” through establishing a hybrid discipline based on phenomenology, ontology, and ‘realism’ (Kane, 2007). Nevertheless, theorists and composers like Drew Daniel note that “the term ‘object’ has proven surprisingly persistent, even robust, as a way of designating what we work with when we work with sound despite this methodological handicap.” (Daniel, 2014).
these new and non-conventional tools into their existing music making practices, often yielding novel approaches to form and structure in addition to timbre and texture.

An understanding of sound as an object also paves the way for process-based and algorithmic approaches to musical composition. For example, György Ligeti’s sound-mass compositions like *Apparitions* and *Atmosphères*, which prioritize timbre and sonic texture over harmony and rhythm, reveal a sculptural approach to sound and a corresponding object-based development of musical material.\(^\text{10}\) This type of formal construction, which Ligeti called ‘net-structure,’ and further developed in his *meccanico* style pieces during the 1960s, results in a “web of functions in which every strand touches others but no strand touches all.”\(^\text{11}\) Stochastic processes used to generate rich tone clusters as well as serialist approaches to musical composition also reveal an underlying belief that complex musical ideas can be assembled from simple musical elements or objects. Contemporary theories of sonic objects and germ-cell interpretations of musical development ultimately lead to a bottom-up understanding of musical form, in stark contrast to the top-down and reductionist Schenkerian approaches. From this bottom-up perspective, larger scale organization is “the result of a process of internal development provoked by interactions on lower levels of musical structure.”\(^\text{12}\)

Conceptualizing sound as an object has provided the composer with a rich and

---


multidimensional metaphor for working with musical material using this expanded sonic palette.\textsuperscript{13}

The development of experimental electronic and computer music is perhaps the most illustrative example of the twentieth century approach to sound as object. Early electronic instruments like the Hammond Novachord provided parametric control of aspects of timbre, sound, and shape. The instrument’s control interface expanded on existing models of control such as the organ drawbars, and used descriptive terms such as “brilliant,” “deep,” “full,” and “mellow”\textsuperscript{14} to describe the resulting electronic textures. Voltage controlled modular synthesizers provided control of pitch, filter, and amplitude envelopes, allowing musicians and composers to carefully craft the precise way their sounds evolve over time. Early electronic music environments like the Fairlight CMI and the UPIC allowed composers in the rapidly evolving field of computer music to manipulate specific elements of sound, providing the first gesture-based control of synthesized sonic objects.\textsuperscript{15} These systems provided a virtually unlimited range of synthesized or processed sounds, and an underlying architecture that allowed practitioners to invent, arrange, and sculpt sonic objects. As Edgar Varèse famously prophesized in 1936, “the time will come when the composer, after he has graphically realized his score, will see this score automatically put on a machine that will faithfully transmit the musical content to the listener.”\textsuperscript{16} An understanding of sound as object has

\textsuperscript{14} Holmes, \textit{Electronic and Experiential Music}, 32.
influenced countless important musical trends of the twentieth century in addition to *musique concrète* and early experimental electronic music, and continues to influence how we understand and interact with sound in the twenty-first century.

### 1.2 Expanding Outward

The landscape of technical innovation at the turn of the century brought about powerful new tools to record, edit, and process digital sound. Musicians and composers were able to manipulate recorded sound in ways that were previously unimaginable, resulting in fundamental shifts in the way we understand and interact with sound. As Paul Lansky notes in his article, “The Importance of Being Digital,” the standardization of digital audio formats, the lossless nature of copying digital audio, and the proliferation of inexpensive digital converters all contributed to a collective shift towards understanding recorded sound as just another form of stored digital data.\(^{17}\) As compact discs became the standard for consuming music, and the personal computer became an essential household item, people increasingly put these audio discs into the optical drive of their computers for storage or playback in addition to simply playing them in their stereos. This perceptual shift was further reinforced as downloads and streaming access to remote cloud-based storage inevitably replaced physical media. The ensuing technological culture not only resulted in drastic changes in the ways that we consume music, but also in how musicians, composers, and engineers interact with recorded sound in the digital medium.

---

Sound recorded on a segment of tape is intuitively understood and manipulated in the physical world, with its length directly related to its duration. Early musique concrète composers were able to interact with recorded sound in a tangible way as they spliced, taped together, arranged, measured, and looped physical pieces of tape. Accessing, viewing, and manipulating recorded sound in the digital format, however, requires a layer of software. The majority of these software applications involve either explicitly indicating the edits using a pre-made score, or interacting with graphical objects rendered on a computer screen. In either case, the interactions with the medium itself are indirect and involve keystrokes, mouse clicks, or simulated control surfaces. The lack of any physical manifestation, however, presents several important advantages to working with sound in the digital medium.

Virtual software samplers, for example, map performance- or sequence-based MIDI data directly onto recorded sound. This technique of automating recorded sound playback is a direct result of early musique concrète methodologies, as evidenced in keyboard-based analog tape instruments like Hugh Le Caine’s Special Purpose Tape Recorder and the Mellotron. In the digital medium, however, recordings of acoustic instruments performing a wide array of musical articulations, techniques, and dynamics can be carefully edited, sorted, and arranged into structured banks of audio data called sample libraries. In addition to selecting the appropriate sample and adjusting the rate of playback based on keyboard note entry, sampling software employs techniques unique to the digital medium like multi-sampling, keyswitching, and continuous controller

---

mappings. These techniques automate the process of switching between sample banks using performance data such as note velocity, gesture-based control, or programmed articulation changes. Commercial digital sample libraries like the Vienna Symphonic Library commonly contain millions of individual sampled recordings on a scale utterly unimaginably with physical media.\(^\text{20}\)

By applying digital sampling techniques to sample libraries comprised of found or recorded sounds, composers working on a *concrète* soundscape on a computer can therefore develop a profoundly intuitive grasp of the source material. They can effortlessly audition excerpts from years worth of material that has been collected and carefully organized, drawing connections and associations among the seemingly disparate recordings. Using various search and sort functions, these files can be instantly rearranged according to the date of the recording, the type of sounds they contain, the spectral or timbral shape of the sounds, tempo, harmonic and melodic content, or even a custom system of classification based on written descriptions or tags. The scalability, organization, and access to audio data provided by software samplers and sample libraries profoundly influence the ways that people interact with recorded sound. The following description from the album notes of The Books’ 2010 album, The Way Out, reveals how access to recorded sound in the digital medium influenced their compositional process.

Beeps’, ‘Breathing, Sighing’, ‘Impediments’, ‘Laughter’, and so on. Says de Jong, “In a way the subjects for a new record choose themselves by standing out through a combination of sheer mass, musical qualities, and content that resonates with both of us…” Using the library as a starting point, Zammuto then finds threads, themes, and unifying rhythms that become the seeds for musical compositions. The compositions are then built from the inside out by adding studio recordings of the duo’s guitars and cellos, one-of-a-kind homemade instruments, and occasionally sung lyrics. The result is the tightly knit, highly rhythmic sound-collage that has become the signature sound of the Books. Thanks in large part to the ever-increasing scale, scope and organization of the library, The Way Out represents a huge leap forward for the Books.21

1.3 Expanding Inward

New digital tools provide musicians, composers, and engineers with tangible tools to zoom in and magnify sounds in both the time and frequency domains, accessing formerly hidden worlds of micro-time and subtle spectral detail. Digital recording software universally uses graphical editing interfaces with the ability to zoom in and out of the visually sequenced audio events, enabling engineers and composers to edit and arrange sounds on the micro-time scale. The data used to describe digital sound can also be examined and operated on using algorithms and processes developed in the fields of computer science, human computer interaction, and digital signal processing. Theorists and musicologists employ sophisticated analysis tools to transcribe the precise micro-timings captured in recordings, establishing entirely new areas of analysis in traditionally non-notated and groove-based music.22 The same technology is also used in the analysis of expressive timing in performances of a written score.23 Programs like IRCAM’s AudioSculpt and University of London’s Sonic Visualizer allow one to visually

manipulate time-varying spectral data utilizing a set of tools similar to those found in graphics editing programs. Similarly, Michael Klingbeil’s sinusoidal analysis and resynthesis program SPEAR utilizes a vector-based graphics environment to manipulate spectral data, allowing the user to select, cut, copy, paste, drag, stretch, rotate, or remove individual or groups of harmonics. These software programs represent powerful digital tools used for both composition and audio analysis, and are entirely dependent on the digital representation of recorded sound.

1.4 Fragmented Time

Manipulating sound in the digital medium by definition involves working with discrete time units. Pulse-code-modulation, the standard and uncompressed time-domain digital audio format, is defined by two quantized parameters: bit-depth and sampling rate. Bit-depth defines the resolution of amplitude at each data point, and sampling rate determines the granularity of time. Whereas an analog audio signal can be thought of as a continuous function that describes a sound’s instantaneous amplitude at any given time, a digital representation of sound is essentially a table comprised of time indices and corresponding sampled amplitude values. While it is the role of the analog-to-digital and digital-to-analog converts to encode and decode data into continuous fluctuations in voltage, sound in the digital domain is temporally neutral. Any notion of time is either extracted from or applied to the data as it goes through the encoding or decoding processes. The latency involved in the encoding and decoding processes is greatly reduced as digital systems become more efficient and input/output bus architecture more
robust, resulting in the notion of synchronicity and real-time digital signal processing. Although never entirely eliminated, audio latency and delay can therefore be more flexibly compensated for in digital audio applications and TMMs.

The concept of fragmented time exists on several temporal levels when working with digital audio. On the hardware level, information about the sound is encoded or decoded according to the sampling rate, which represents the first level of time granularity. Software applications introduce an additional level of time quantization in the form of audio buffering, control rates, and the application of discrete-time audio analysis and resynthesis like the fast Fourier transform and its inverse. A real-time continuous stream of digital audio is essentially a tightly coordinated and synchronized parsing and reassembling of discrete packets of digital audio data. Digital time code represents one approach to dealing with the temporally neutral nature of audio data, and is particularly important in applications where digital audio coexists with other analog audio signals or video data.²⁴

Unlike analog media such as tape or vinyl, audio stored in the digital medium can be infinitely copied and accessed without damaging or degrading the information it contains. As a result, digital sound can be edited, processed, and layered in ways that are physically impossible in other recording mediums. Both computer hardware and software facilitate non-linear and non-sequential access to recorded sound, resulting in an increasingly fragmented approach to the sound object. The process of fragmentation lies at the core of many manipulations of digital sound, in both the time- and frequency-domains. Huge

²⁴ Manning, Electronic and Computer Music, 212.
amounts of research and experimentation in the fields of digital signal processing, telecommunication, and audio engineering have focused on the formidable challenges involved in isolating a recorded sound’s timing and pitch information. Although historically a few analog mechanisms were able to provide independent control of time and pitch (see section 2.2 on granular synthesis), the digital medium is arguably best suited to freely isolate and manipulate time as an independent variable.

Different software platforms adopt different approaches to time granularity, resulting in drastically different metaphors for working with time. Most digital recording and editing programs work on an absolute time model, which is manifest in the global clock and virtual playhead that moves across the audio and MIDI events arranged in the program’s visual sequencer. An additional layer of granularity exists in the process of quantization, which applies a virtual grid to sequenced events. The resolution of the grid is defined by the user according to musical parameters like sub-divisions of the beat, or non-musical metrics like minutes and seconds, or video frames and timecode. In addition to traditional sequencing and editing, performance-oriented software such as Ableton Live and Bitwig allows the user to structure time on a macro scale, providing an additional layer of granularity. Looped sequences and sub-arrangements called scenes, which represent form-based structures from popular music like verse, chorus, and bridge, are triggered in real-time allowing the user to perform a version of the sequenced material.
In each instance, the user or performer ultimately controls the progression of time within the prescribed temporal context.\textsuperscript{25}

Interactive programming environments like Max/MSP and Pd adopt a cause-and-effect approach to time, in which events like real-time input, triggers, or control signals and other data structures effectively control the flow of time. The computer music program SuperCollider provides a library of pattern generators, which allow the user to create and manipulate streams of control data algorithmically in addition to using real-time input. Many real-time audio programming environments use separate timing paradigms for control input, creating an additional layer of time granularity. The program ChucK offers an impressively flexible approach to time, in which time granularly can be dynamically set to many different values at the same time, allowing programmers to “flexibly and precisely control the flow of time in code.”\textsuperscript{26}

1.5 Adapting Time: Lopsided Loops, DAWs, and Timed Musical Machines

With the virtualization of the recording studio in the form of the Digital Audio Workstation (DAW) in the late 1990s, composers, producers, and musicians alike were granted new and unprecedented control over fundamental aspects of recorded sound and musical time. Alongside the development of the personal computer, the DAW gradually


\textsuperscript{26} Ge Wang, “The ChucK Audio Programming Language: A Strongly-timed and On-the-fly Environ/mentality” (PhD Diss, Princeton University, 2008).
expanded from a niche production tool to an expressive and multifaceted instrument. In many regards, the musical, cultural, technological, and aesthetic evolution of the DAW follows a similar trajectory to the turntable in the second half of the twentieth century. In both cases, the technology was co-opted and used - or misused - as an instrument, establishing its own measure of virtuosity, spawning off entirely new musical genres, and eventually securing its own robust performance practice incorporating experimentation, novel supplementary performance tools, and extended technique. One important consequence of the adoption of the DAW as an instrument is the gradual accommodation of its tools and practices, both technical and aesthetic, into the broader vocabulary of music making in the twenty-first century.

Initially developed as an alternative to multi-track analog tape recorders, DAWs are the integrated software and hardware systems used to record, edit, and produce music digitally. In addition to simply recording audio, DAWs have come to encompass a wide range of tools for processing and transforming sound, sophisticated virtual instruments for synthesis, and massive libraries of sampled sounds. The DAW has a fascinating history intertwined with the rise of the personal computer, the shifting landscape of the audio recording industry, and the technological and economic factors surrounding the rise and fall of hardware-based electronic instruments such as samplers, synthesizers, and drum machines. While there are many variations, most DAWs use a graphical timeline to visually edit and arrange sounding events, or ‘regions’, along with other time-based information such as automation data and tempo changes. DAWs also commonly include a virtual mixing console, allowing the user to dynamically control aspects of the sound
like volume level, panning, dynamics, filtering, and effects routing. Finally, comprehensive DAWs comprise a connected network of hardware-based peripherals including MIDI instrument controllers, traditional computer input devices like the mouse and keyboard, custom control surfaces based on analog mixing consoles, as well as more esoteric input devices like touchpads, joysticks, and velocity-sensitive triggers and toggles.

Although the advantages of working with digital audio had been well documented, it was not until the turn of the century that technological and economic factors aligned to produce a robust and accessible alternative to traditional recording and editing tools. In 1991, for example, the bare minimum amount of digital storage needed to work on a full length CD - approximately 700MB - would have been prohibitively expensive for the average consumer, costing thousands of dollars. By the end of the decade, however, the cost was down to less than ten dollars, and in another five years it would be less than fifty cents.27 Similarly, computer hard drives achieved faster and more efficient operational speeds, greatly expanding the number of virtual tracks used for both multi-track recording and playback. Aside from a few custom configurations, a high-end home personal computer could run the same DAW software found in commercial recording studios. Similarly, high bandwidth input and output bus architecture like firewire and USBII gave small project studios access to the same tools used to record, edit, mix, and

---

master digital music that were previously limited to trained engineers physically working in a recording studio facility.²⁸

By the middle of the 2000s, PC-based audio workstations had evolved into fully functional ‘studio-in-a-box’ software applications. Software that formerly focused exclusively on recording and editing digital audio, for example, expanded to include more flexible MIDI editing and sequencing, virtual synthesizers, and standard musical notation. Similarly, standalone software synthesizers adopted their own time-line based sequencers and eventually digital audio recording capabilities. Standardized protocols like VST, RTAS, MAS, and AudioUnits allowed external third-party processing and synthesis programs called plugins to be loaded directly into the DAW’s recording and editing environment.²⁹ These protocols not only allowed third party programs to run concurrently inside the DAW environment, but also permitted the host program’s automation data and connected control hardware to control the various synthesis and processing parameters in real-time. The result was a proliferation of software utilizing sophisticated digital signal processing to emulate both the audio effects and user experiencing of existing, studio-based hardware. The concept of a DAW in turn expanded to include all of the tools needed to create music digitally in one flexible and inter-connected environment.

DAW systems inevitably migrated onto the performance stage as laptops and other portable computers became powerful enough to handle audio processing tasks. This

²⁹ Ibid., 397.
trend was manifest in the rise of DAW-based real-time performance tools like Ableton’s Live and Apple’s Mainstage, which is the performance-oriented version of their popular recording and editing software, Logic. Many of these same software companies developed versions of their software for tablets and mobile devices, and class-compliant audio interface protocols meant that high-quality converters could be used with both PCs and portable devices. The music critic, composer, and author David Toop observed, “Live music still exists, as does housework and cooking, though all of these activities now take a certain amount of automation for granted. The computer controlled smart home exists, at least in the pages of design magazines, and many musicians now sit on stage with nothing between them and their public but a compact, portable, and unimaginably powerful laptop computer.”

1.6 Sound Object and Timed Musical Machines

The proliferation of powerful digital processing and analysis tools ensures that the craft of creating music by manipulating, layering, and arranging recorded sound continues to be an essential element of music making in the twenty-first century. Recorded sound is accessed, analyzed, edited, and manipulated in powerful ways in the digital medium. Novel digital tools couple unprecedented access to the sound object with varying degrees of automation, continually challenging and redefining the ways that humans engage with recorded sound. As a result, access to both the hidden inner worlds of recorded sound

---

30 David Toop, Haunted Weather: Music, Silence, and Memory (Serpent’s Tail, 2004), 220.
and the mechanisms that control their playback extend down to the smallest time scale. The inherently fragmented nature of digital audio similarly leads to control and manipulation on a micro-time scale. Whereas analog recording and playback technologies facilitated an understanding of sound as object - which resulted in a proliferation of compositional, philosophical, cultural, and technological innovations - the contemporary technological environment facilitates an approach to sound as instrument. The TMMs explored in the following two chapters present a compelling and illustrative metaphor for manipulations of sound as instrument.
2.1 Categories of Timed Musical Machines

The software-based timed musical machines examined in this chapter fall into two categories based on the type of audio data they process. The first category operates directly on the sampled amplitude values encoded in time-domain digital audio data. Examples of time-domain processes include sound file granulation (granular synthesis), real-time sampling, and delay-based effects like looping, reverberation and filtering. The second category generates and processes frequency domain data, and includes Phase Vocoders, Fourier transform analysis and resynthesis, sinusoidal analysis and resynthesis, and Linear Predictive Coding (LCP). These processes all utilize an analysis of the audio signal to generate discrete information about the sound’s harmonic spectrum over a certain window of time. Many of the processes in both categories involve windowing functions as a means of masking the discontinuities that result from having to parse data into discrete packets either for analysis or non-sequential resynthesis. Finally, although physical modeling and algorithmic synthesis adopt similar approaches to sound as object and instrument, for the purposes of this discussion, I will focus exclusively on virtual machines that manipulate recorded sound. Table 2.1 provides a summary of synthesis techniques.
### Table 2.1 Examples of Digital Synthesis Techniques (after Klingbeil, 2009)

<table>
<thead>
<tr>
<th>Time Domain</th>
<th>Frequency Domain</th>
<th>Physical Model</th>
<th>Algorithmic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sampling</td>
<td>Additive</td>
<td>Waveguide</td>
<td>FM</td>
</tr>
<tr>
<td>Concrète</td>
<td>Phase Vocoder</td>
<td>Karplus-Strong Modal</td>
<td>AM</td>
</tr>
<tr>
<td>Granular</td>
<td>Sinusoidal</td>
<td></td>
<td>Waveshaping</td>
</tr>
<tr>
<td>Scanned Synthesis</td>
<td>Source-Filter (LPC)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Delay lines</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 2.2 Time Domain Musical Machines I: Granular Synthesis and Sound file

**Granulation**

The term granular synthesis has come to describe “a concept of sound generation and not a single effect.” As a result, various manifestations of granular synthesis can be found in a wide range of digital audio applications, from modifications of a sample’s tempo and pitch through to more esoteric synthesis techniques such as micromontage, temporal scattering, stutter effects, and stochastically generated sound-clouds. Curtis Roads’ book, *Microsound*, provides a thorough account of both the history and the technical aspects of granular synthesis and other closely related manipulations of digital audio. On a very basic level, however, granular synthesis is the systematic parsing and sequencing of short segments of sound, called grains, on various micro and macro time scales. Sound file granulation specifically refers to the use of granular techniques on recorded sound, although the two terms are often used interchangeably.

---


32 Roads, *Microsounds*. 
The idea of an individual and elemental grain of sound dates back to Dennis Gabor’s theories of acoustic quanta developed during the middle of the twentieth century. Iannis Xenakis’ compositions *Concret PH* and *Analogique B* are among the first musical compositions to employ granular-based tape-editing techniques. As the composer states in his program notes, the technique generates “extremely rich sounds (many high overtones) that have a long duration, yet with much internal change and variety.”\(^{33}\) In an attempt to mechanically automate the granulation process, engineers in the middle of the twentieth century developed several innovative tools, such as the Phonogène and the Tempophone, to provide independent control of the pitch and tempo of analog tape playback.\(^{34}\) These tools operated in the analog medium, using a variety of techniques such as frame-based synthesis methods derived from film projection technology or custom tape apparatus involving multiple rotating playheads. Due largely in part to the fragmented nature of digital audio data, however, the digital medium has become an extremely powerful platform to experiment with granular synthesis techniques. Notable contributions to early digital granular synthesis systems include Curtis Roads’s Creatovox synthesizer\(^ {35}\) and Barry Truax’s GSX and GSAMX software programs, which are important as they were among the first applications of granular synthesis in real-time.\(^ {36}\) The basic concept of digital granulation has spawned many related synthesis and processing techniques, including concatenative sound synthesis, data-driven synthesis techniques, and FOF and wavetable synthesis.\(^ {37}\)

---

33 Roads, *Microsound*, 64.
One of the most important implications of approaching audio data in a granular way is the ability to transform a sound’s pitch and timing independently. When the granulation algorithm is controlled in part by a dynamic input, such as a keyboard or gestural controller, these transformations of time and pitch have the potential to turn recorded sound data into the malleable object of audio synthesis. Many commercial products implement varieties of granular synthesis and sound file granulation. Examples include FlexTime, the time-stretching algorithm based on the tempophone used in Apple’s music production software Logic, the popular Grain Delay and Granulator audio effects in Ableton Live, the synthesis engine Malström in Propellerhead’s program Reason, and the virtual synthesizer MetaSynth. While these examples represent black-box, or closed-source uses of granular synthesis, computer music programming environments like Pd, Max, Chuck, SuperCollider, and Csound provide powerful high-level tools that allow one to describe their own mechanisms for sound file granulation. As computer processing power steadily increases, and digital audio programming environments become more efficient, the basic concepts behind sound file granulation are becoming increasingly accessible to people with little or no programming experience. Similarly, although granular synthesis has been implemented in real-time since the late 1980s, the ability to flexibly control the underlying processes, and apply them to larger amounts of digital audio data result in categorically different approaches to real-time granulation.

While sound file granulation has evolved into a diffuse category of digital audio treatments, it is nevertheless helpful to identify three key components. These components are also useful in comparing and contrasting the virtual mechanisms behind other categories of timed musical machines. The three components of sound file granulation are the sequencer strategy, the scheduler, and the grain anatomy. The purpose of this classification is to help identify novel ways of manipulating the synthesis technique and to provide a more systematic approach to experimentation and exploration. Figure 2.2.1 provides a schematic of the basic elements of granular synthesis.

![Granular Synthesis Diagram]

The sequencer strategy refers to the particular means of selecting and extracting audio data in the granulation process. As the processed sound is accessed non-sequentially,

---

implementations of sound file granulation use an allocation of the computer’s RAM, called an audio buffer or wavetable, to temporarily store and access the audio data. Early versions of digital sound file granulation were limited to very short segments of recorded sound, usually lasting only a fraction of a second. The audio buffer in Barry Truax’s GSAMX program, which was developed in the late 1980s using a DMX-1000 signal processing computer, was limited to 4032 samples provided by the onboard memory. As a result, the sequencer strategy referred exclusively to the buffer index point. Despite these technical limitations, and as a testament to how powerfully expressive the synthesis technique is, early examples of granular synthesis nevertheless generated an impressively rich and complex range of sonic textures and timbres. The following example demonstrates a traditional application of sound file granulation that uses an index-based sequencer.

Unlike continuous bowing on a string instrument, or circular breathing on a wind instrument, the notes on a piano are inherently impulse-based. As a result, the sound’s spectral shape is clearly defined in time, with each impulse beginning with a harmonically rich percussive attack followed by a characteristic decay that includes changes in harmonic makeup as well as a gradual decay in amplitude, see Figure 2.2.2. The spectrogram reveals that while the higher partials generally decay faster than the lower ones, the decay is not linear in either time or frequency. For example, the amplitudes of the second and fourth harmonics pulse slightly starting at two seconds, and the fourth harmonic is louder and sustains longer than the third. This type of spectral

---

activity creates a complex and active sense of timbral motion that a performer can manipulate using a granular synthesis technique and real-time controller input.

![Figure 2.2.2 Spectrogram of a Piano Note](image)

Figure 2.2.3 demonstrates sound file granulation applied to the same piano sample with real-time input controlling the buffer index. In this example, the granular algorithm uses a 100ms window size and eight overlapping grains. A real-time continuous input, in this example provided by a tether controller or by clicking and dragging directly on the waveform image, determines the granular buffer index position. While the original sound is approximately seven seconds long, the granular excerpt is more than 40 seconds in length. In addition to a prolonged period of sustain, this excerpt also contains four timbrally contrasting sections, produced by navigating the buffer index position back towards the note’s attack. Despite the pronounced change in harmonic makeup in these

---

four sections, the entire passage has a consistently uniform sense of pitch derived from the original sample. Figure 2.3.4-A and 2.3.4-B demonstrates the same granulation technique applied to the un-pitched sound of a cymbal crash.

With the ability to efficiently access large amounts of audio data using modern computer systems, the design of the sequencer has become an increasingly important aspect of
sound file granulation (see section 1.2). In addition to the traditionally used buffer index position described in the previous example, software applications are also able to integrate high-level descriptors such as detected pitch, amplitude, or timbral shape into the sequencer. Increasing amounts of RAM and faster read speeds associated with solid state drives provide the infrastructure for the nearly instantaneous access to massive amounts of data, allowing the synthesis algorithm to access sounds from massive databases of audio files. In turn, software developers are adapting the virtual metaphors used to organize and access recorded audio data. Cycling74’s polybuffer~ object, for example, gives the user the ability to load an unspecified number of audio files into a unified and self-adapting buffer object, updating the concept of a wavetable to accommodate structured banks of sound files.\(^{43}\) In addition to the ability to load more data into memory, modern computing platforms’ 64-bit architecture virtually eliminates the floating-point error that had previously prevented flexible real-time indexing of wavetables longer than a few minutes.\(^{44}\) As opposed to early granular applications that were limited to approximately 150 milliseconds of recorded sound, the temporal scope of the sequencer can now extend well into the macro time scale, encompassing minutes, hours, days, and potentially years worth of audio material. The granular sequencer therefore provides the foundation for many related synthesis techniques, including concatenative sound synthesis, digital multi-samplers, and reconstructive phrase modeling.\(^{45}\)

\(^{43}\) “Polybuffer~: Manage multiple buffer objects” from the Cycling74 online reference pages, accessed July 20, 2014, https://www.cycling74.com/docs/max6/dynamic/c74_docs.html#polybuffer

\(^{44}\) At 44.1Khz sampling rate, the seven significant digits in 32bit single precision audio data generate floating-point errors after approximately 3.7 minutes. 64bit audio, which has 16 significant digits, would require over 6000 years to produce the same error.

The grain scheduler refers to the specific method of assembling grains. While there are many different approaches to scheduling grain events, they generally fall into two categories, defined by the metaphors that describe the types of sounds and textures they generate: streams and clouds.

The specific type of programming paradigm often informs the approach to scheduling grain events. For example, synchronous and quasisynchronous granular synthesis techniques employ periodic control functions like pulse or sawtooth waves to regulate a stream of grains.\textsuperscript{46} The frequency of the control function, which is described in terms of grains per second or inter-onset interval, determines the timing of the grains and contributes to the perceived granular density. When multiple streams are synthesized simultaneously, the phase of each stream can be manipulated to control the amount of overlap. The granular transformations in examples 2.2.3 and 2.2.4, for example, utilize a synchronous scheduler based on eight simultaneous streams whose phases are evenly distributed across each cycle, generating a smooth and continuous texture depicted in Figure 2.2.5.

\textsuperscript{46} Roads, Microsound, 93.
In the case of quasi-synchronous granular synthesis, a random deviation factor is introduced to each triggered grain event, resulting in irregular inter-onset intervals. Both types of synchronous scheduler methods use the basic building blocks of audio synthesis, called unit generators, as control functions. This type of scheduler is therefore particularly well suited to object-oriented computer music programming environments like Max, Pd, chuck, and SuperCollider, which contain extensive libraries of unit-generators for both control and audio synthesis applications.

The second type of scheduler relies on time-varying descriptions of events, generating clouds of grains. Examples include text-based scores that precisely define the triggering of grains over time, or algorithmic approaches that generate grains based on formal structures, probability, chaos, or physical models. Sometimes called asynchronous granular synthesis, this method of scheduling is driven by discrete events as opposed to periodic functions or unit generators. Figure 2.2.6 shows the sonogram image of an asynchronous granular scheduler based on matching keywords or phrases from a text. In this example, the entire text from this chapter is loaded into a single array, and a program
steps through each word at a rate of five milliseconds. Each occurrence of the word ‘grain,’ ‘granulation,’ or ‘granular’ triggers a grain event as depicted in Figure 2.2.7.

![Figure 2.2.6 Sonogram Image of Asynchronous Granular Scheduler Based on Text Matching](image1)

![Figure 2.2.7 Asynchronous Scheduler based on Text Matching](image2)

Early experiments with MIDI hardware and samplers were able to generate granular textures using an event-based scheduling technique called Rapid Event Deployment, or RED. Unlike early unit-generator based granular synthesis which relied on custom software, expensive computer hardware, and dedicated signal processors (see examples by Truax and Roads), RED-based granular synthesis used off-the-shelf components and

---

standard MIDI controllers. Given the type of input data, the event-based asynchronous scheduler is capable of a wide range of sonic textures, including tightly coordinated streams similar to synchronous GS, rhythmic sequences, or dense granular clouds. In addition to a prescribed score or distribution based on probability, an adequately robust physical input device can function as an asynchronous gesture-based granular scheduler.

The piece *Windows that Won’t*, written for the laptop ensemble Sideband, uses an asynchronous scheduler that generates collective granular textures using individual piezo triggers and wireless communication. Each performer controls a single stream of grains whose phase and frequency are determined by the intensity of the attack trigger. A conductor coordinates the ensemble via a series of messages that contain specific performance instructions, producing collective granular textures that range from sparse and pointillistic to dense and continuous. Figure 2.2.8 depicts the two modes of interacting with the asynchronous scheduler, which result in contrasting collective granular textures.

![Figure 2.2.8: Two Modes of Controlling the Asynchronous Scheduler in Windows That Won’t](image)

Each performer’s part is played through a six-channel hemispherical speaker, which provides a spatial component to the asynchronous triggering mechanism. When
individual granular densities are sufficiently low, for example, subtle deviations in phase and frequency produce rhythmic hocketing across the ensemble that clearly highlights each individual performer’s contribution to the overall texture (labeled, “Sparse Texture” in Figure 2.2.8). Individual variations in phase and frequency result in more subtle changes in the overall texture and timbre in sections where the ensemble generates a more sustained sound (labeled, “Dense Texture” in Figure 2.2.8). The software is also capable of routing individual grains to different channels of the hemisphere speaker, contributing to a collectively diffused and spatially active sonic image.

The final component of granular synthesis is the grain anatomy, which describes the parameters of each individual grain, including duration, playback speed, spatalization, and amplitude envelope. Each grain is defined by an amplitude envelope, which isolates the sound and allows the sequencer to extract it from a larger audio segment. The amplitude envelope commonly has a value of zero at its beginning, smoothly ramps up to full amplitude, and then back down to zero at the end in order to mask the transient clicks that are generated in the process of extracting and sequencing the grain. The particular shape of the envelope, however, can vary greatly and has a distinct influence on the overall timbre of the synthesized sound. (See Figure 2.2.9.) The duration of the grains range from the micro to macro times scales, encompassing chirp-like sonic events on the border of perception through to recognizable sounds and extended sequences.
In a traditional application of granular synthesis, the grains last from ten to fifty milliseconds, and are defined by a Gaussian amplitude envelope. These parameters are derived from Dennis Gabor’s initial definition of a particle of sound, which represents the basic unit of timbre and time that cannot be broken down into constituent elements.\textsuperscript{48}

Sound-file granulation, micro-montage, and concatenative sound synthesis, however, produce interesting sonic textures and rhythmic shapes using grains lasting up to several seconds with a wide array of amplitude shapes, from simple three-stage linear envelopes, to exponential attacks and sinc functions. Removing the envelope completely produces interesting sonic effects, as the rhythm and periodicity of the exposed transient clicks draw attention to the mechanics of granulation process itself.

The following TMMs illustrate how all three components of sound file granulation can be manipulated in real-time through a combination of automation and real-time input. The first three examples are limited to the types of inputs common to all computer systems: keystrokes and mouse or trackpad input. The remaining examples use a gesture-based

\textsuperscript{48} Roads, \textit{Microsound}, 86.
tether controller called the Gametrak, which is a particularly well-suited input device for manipulations of live synthesis and audio processing. These TMMs also use an input-mapping patch that allows the user to manipulate aspects of the granulation mechanism simultaneously using the controller’s six axes of input data. All of the examples use the granulation software provided with this dissertation (see Appendix A).

The first example, labeled “Preset 1 – scrubbing,” couples an automated synchronous scheduler with a user-controlled sequencer. The program allows the user to control the buffer-offset parameter of the granular sequencer by clicking and dragging on a visual representation of the recorded sound in the program’s graphical user interface, or GUI. The automation in the scheduler is provided by a phasor-driven trigger, which is set to 50 Hz. The second example - ‘text-trigger’ - uses the same GUI-based sequencer, but replaces the automated scheduler with event-based triggering from the computer keyboard. Discrete key-down and key-up messages generate single grains, allowing the program to synthesize both synchronous sustained textures as well as asynchronous cloud-like and rhythmic textures. The third example - ‘mouse trigger’ - uses mouse or track-pad information to control both the sequencer and the scheduler. The horizontal position controls the position within the buffer, and changes in either vertical or horizontal position trigger discrete grain events. Mouse position changes of twenty pixels on either axis trigger the granular scheduler, but this sensitivity can be altered to achieve different ranges in the resulting granular density. Moving the cursor in circular

---

50 Download the kk_granular program from the dissertation website: www.konradkaczmarek.com/tmm. Open the program, and click on the ‘presets’ button, select ‘preset 1 – scrubbing’ from the menu, and then press the green ‘recall preset’ button. Follow the same procedure for the remaining granular examples. See Appendix A for more information on the software.
motion generates interesting textures, as the rate of motion determines the granular
density, and the radius of the circle determines the scope of the sequencer’s buffer-offset
position.

An additional program, called kk_tether, routes the data from the Gametrak controller to
various parameters in the granulation process using three different types of mapping:
scaled, trigger, and split. Scaled mapping simply applies a linear transformation from
the input range, which is normalized to 0.0 to 1.0, to the desired output range. In the first
example, this mapping is implemented in the right-hand y-axis, which controls the grain
size over a range of 20ms to 1000ms, and the left-hand x-axis, which controls the
sequencer playback position within the buffer. Trigger mapping converts a continuous
input stream into discrete events. A ‘Trigger Resolution’ parameter controls the
sensitivity of the triggering by determining how many events are generated across the
range of input values. In this example, trigger mapping is used on all three axes of the
right hand, which control the individual grain triggering in asynchronous mode. Finally,
the split type of mapping is used to toggle on and off the pulse generator, which activates
the synchronous scheduler when the right-hand z-axis crosses the threshold defined by
the ‘split value’ parameter. Figure 2.2.10 summarizes the mappings used in the example.

---
51 Download the program “kk_tether” from the dissertation website.
52 “Preset 1 – tether trigger” in the kk_tether software, “preset 8 – tether trigger” in the kk_granular
software.
The second example uses split mapping to move to discrete points in the buffer based on the left-hand z-axis input. As opposed to the freely scaled position mapping in the previous example, this TMM allows the user to navigate to specific, predetermined sections in a recorded passage contained in the buffer. This machine also uses a synchronous scheduler with a fixed granular density of 100 grains per second. The left-hand y-axis is mapped to the position dither parameter, which provides an additional layer of interaction with the underlying granulation process.

These machines represent just a few ways in which sound file granulation can be implemented in a TMM. Even the relatively simple examples that use keyboard and mouse input nevertheless provide a compelling platform for experiencing and interacting with digital sound. While interaction with the sequencer in the form of visually navigating the buffer is a particularly intuitive approach, integration with the various automation parameters yields interesting and often unexpected results. For example, “preset 10 – density control” maps the tether left-hand z-axis input to granular density, allowing the user to expressively navigate the temporal range from discrete rhythms to sustained texture. Similarly, “preset 11 – volume_range_control,” maps the left- and right-hand y-axis data to the upper- and lower-limit of the volume range. This mapping

---

**Figure 2.2.10: Mapping Used in Real-Time Sound-File Granulation With a Tether Controller**

<table>
<thead>
<tr>
<th>Input type</th>
<th>Input monitor</th>
<th>Input mapping type</th>
<th>low cut</th>
<th>high cut</th>
<th>trigger value</th>
<th>trigger repetition value</th>
<th>split under</th>
<th>split over</th>
<th>output destination</th>
<th>output</th>
</tr>
</thead>
<tbody>
<tr>
<td>active left_x</td>
<td>0.830</td>
<td>map</td>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
<td>position_phase</td>
<td>0.1</td>
</tr>
<tr>
<td>active right_x</td>
<td>0.451</td>
<td>trig</td>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
<td>manual_trigger</td>
<td>0.1</td>
</tr>
<tr>
<td>active right_y</td>
<td>0.454</td>
<td>trig</td>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
<td>manual_trigger</td>
<td>0.1</td>
</tr>
<tr>
<td>active right_z</td>
<td>0.454</td>
<td>map</td>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
<td>gainsize</td>
<td>0.1</td>
</tr>
<tr>
<td>active right_y</td>
<td>0.451</td>
<td>map</td>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
<td>pulse_generator</td>
<td>0.1</td>
</tr>
<tr>
<td>active left_y</td>
<td>0.683</td>
<td>map</td>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
<td>position_dither</td>
<td>0.1</td>
</tr>
</tbody>
</table>
produces a range of animated rhythmic textures, as larger deviations in volume within a fixed granular density produce the impression of accented and unaccented beats.

Appendix A contains additional documentation for the purpose of developing TMMs using the granulation program and physical input devices like the tether.

### 2.3 Delay-Based Musical Machines

Delay-based musical machines emerged from experiments with analog tape in electronic music studios and experimental recording studios throughout Europe and the United States in the middle of the twentieth century. These experiments involved custom modifications to studio-based recording equipment to generate delays and feedback loops in the audio playback mechanism. The modifications allowed engineers and composers to manipulate not only the playback but also the recording capabilities of the tape machine, and transformed the conventionally fixed and archival medium into a more flexible and temporary type of storage. Whereas granular processes emerged primarily out of editing techniques like the tape splice, delay-based processes evolved out of manipulations of both the medium and machinery involved with recording.

Experiments with audio delays have had a profound and lasting impact on studio-based recording techniques, and have influenced a wide range of electronic and acoustic music. The specific hardware modifications were quickly adapted into dedicated tape-based devices, solid-state analog circuitry, and eventually simulated with digital effects units, pedals, and software emulations. As a delay by definition involves temporarily storing an
audio signal and then playing it back a short time later, the characteristics of each recording medium influence the particular sound of the delay and determine the various ways that it can be manipulated. Delays in the digital medium increasingly make use of the non-linear and fragmented nature of stored digital data. Powerful digital processing units and computer software also offer an unprecedented level of flexibility and scalability in implementing audio delays.

Initial experiments with variable-time delay using analog tape used two reel-to-reel tape machines to isolate the record and playback functionality. Engineers and inventors, however, soon created custom machines with adjustable record- and play-heads to experiment with the effect in a more precise and controlled manner. In these early tape-based systems, incoming audio is encoded in the magnetic material of the tape as it passes over the record-head. The tape then bypasses the uptake reel and is routed to the second machine, where the audio is picked up by the play-head as the tape arrives a short time later (see Figure 2.3.1). The physical distance between the record- and play-heads of the two machines and the speed of the tape determine the specific delay time. Additional play-heads, called taps, create multiple discrete delays, and movable play-heads allow the user to create precise rhythms and patterns with the delays. This type of delay is therefore often called a tap delay. Routing the output of the delay back to the input establishes a controlled feedback structure capable of repeating delays with variable decay, called an echoplex, and sustained looping.
As delays became more widely used as a production tool, experimental studio-based reel-to-reel tape delays were joined by commercial echo and delay effect units like the Echoplex EP-2 and Roland’s Space Echo. These dedicated delay mechanisms used internal tape loops, cartridges, or rotating magnetic disks for storage, and added an erase head to clear the recorded audio with each complete revolution of the loop or disk. In general, the control parameters for these delay-based devices are delay time, input and output gain, and feedback amount. Many products include discrete footswitches or continuous expression pedals to control the parameters interactively in real-time. Several early tape-based echo effects included primitive looping functionality through a ‘sound on sound’ effect, which disabled the erase head allowing the performer to add audio to the tape loop with each complete cycle. More sophisticated delays, particularly in the digital medium, allow one to dynamically change the delay time, rate of playback, and playback direction, resulting in pitch shift and reverse delay effects.

---

Delays in the digital medium implement a specific type of data structure, called a ring buffer, for temporarily writing and reading audio samples into memory. A ring buffer is an ordered array of finite length that contains a link between its two endpoints, creating a virtual structure with no effective beginning or end that is analogous to the tape loop. Ring buffers utilize sequential pointers, or references to locations in memory, which read and write sample values in sequence as they step through the data array (see Figure 2.3.2). In contrast to hard disk-based playback, recording, or editing, a digital delay’s ring buffer utilizes the computer’s RAM for dynamic access to the audio data with very low latency.

In tape-based hardware delays, the maximum delay is determined by the physical limitations of the setup, where the speed of the tape and the distance between the record- and play-head ultimately determine the delay time. In a digital delay, the maximum delay time is determined by the amount of memory allocated to the ring buffer, and playback pointers generate variable and precisely controlled output delay. Until

\[\text{delay duration}\]

\[\text{Audio Input}\]

\[\text{playback pointer}\]

\[\text{record pointer}\]

\[\text{Delay output}\]

Figure 2.3.2 Ring Buffer

\[\text{54 Download the program, “kk_ringbuffer” from the dissertation website for an example of a ring buffer-based audio delay.}\]
relatively recently, the high cost of memory placed practical limitations on the duration of a digital delay or loop. Similarly, the limited processing power available in early computer music systems restricted the ways the playback pointers could be manipulated in real time. Most current computer music programming environments, however, incorporate sample-accurate recording and playback functions, providing users with high-level tools to create their own buffer-based delays and instruments. The decreasing cost of RAM together with more powerful programming environments blur the lines between memory- and hard-disk-based playback methods. Although the traditional metaphors for creating and interacting with continuous delay-based machines differ from discrete granular machines, the underlying hardware and programming conventions are increasingly similar, resulting in a cross-pollination of techniques and corresponding modes of interaction.

Figure 2.3.3 - Delay-based Live Looper
In traditional delay-based musical machines like the looping pedal or multi-tap delay, the concepts of the sequencer, scheduler, and anatomy outlined in granular synthesis are generally defined by the characteristics of the ring-buffer and the conventions of delay processes established in the analog medium. The continuous and cyclical nature of the read and write pointers, for example, preclude the application of a more flexible granular-based scheduler. Similarly, the concept of grain anatomy does not apply to delay-based processes that utilize a continuous feedback loop. The sequencer is limited to the contents of the ring buffer and the relative position of the playback pointers. Physical interaction with simple delay-based machines in the form of switches, toggles, and expression pedals, however, allows the user to interact with the sequencer by adding new audio into the buffer or controlling what is copied back through feedback. Applying a unity value to the feedback gain creates a continuous loop equal to the current delay duration, which can extend up to the length of the ring buffer resulting in longer delays and more accurate and interactive live looping (see Figure 2.3.3).55 Furthermore, digital feedback loops are not susceptible to the degradation in audio quality that is introduced with each iteration of recording in tape systems.

Delays in the digital medium are more flexible and scalable than their analog counterparts, as they do not rely on a physical storage medium such as magnetic tape or analog circuitry. For example, more capacious hardware systems that utilize multiple buffers for live looping can provide non-destructive overlap and add functionality by assigning additional overdubs to their own dedicated and synchronized buffers. Delays

---

55 Download the program, “kk_livelooper” from the dissertation website for an example of a delay-based live looper.
that utilize multiple buffers can similarly generate complex and highly accurate rhythmic patterns based on the relative proportions of the delay or loop times. A single impulse, for example a vibraphone attack, that is fed into four looping buffers with durations of 1333.33ms, 1000ms, 800ms and 571.43ms (ratios of 4:3, 4:4, 4:5, and 4:7 of a base pulse of 1 second) produces the polyrhythm depicted in Figure 2.3.4. In the corresponding audio example, the loops are panned alternatively to the left and right channels and pitched to aide in hearing each part independently.

While this rhythm lines up vertically every four seconds, delay durations not based on integer multiples will move in and out of phase with each other over longer time periods. For example, two loops set to 1000ms and 1001ms will take over 16 minutes to re-synchronize. Similarly, playback pointers can be accurately defined on the sample level, allowing one to navigate the continuum of delay-based effects that ranges from filtering and phase cancellation through to flutter delays, echoes, and continuous looping.
An additional delay applied to the output of a looping delay will generate a secondary loop, whose phase is determined by the second delay time as demonstrated in figure 2.3.5. This type of linked looping provides a sophisticated means of manipulating repeating rhythmic patterns based on micro-time and rhythmic feel, as the secondary delay time can be manipulated freely in real-time. For example, inserting an 85ms delay in a 1000ms repeating loop produces the pattern shown in Figure 2.3.6.
The digital-delay’s playback pointers are manipulated more freely and discretely than the physical tape-head counterparts, providing a means of manipulating virtual machines in expressive and temporally interesting ways. For example, when the playback pointers are manipulated dynamically, they can generate the phase-offset effect across multiple loops without the layer of secondary delays described in example 2.3.5. One can therefore achieve a similar range of psychoacoustic effects using different sound sources or apply temporal quantization to real-time looping and delay. The playback pointers in digital delays are also capable of instantaneous and discrete change, which is not possible in physical analog delays. This allows performers to generate delays and live loops of variable duration, for example using a foot pedal, trigger, or external control function.

Dan Trueman’s digital instrument, the Blendronomer,\textsuperscript{56} relies on precisely automated changes in delay time that generate, “remarkably complex, non-random results.”\textsuperscript{57} The instrument is comprised of a sampled instrument, controlled by the performer using a MIDI keyboard and percussion trigger pad, that is routed into a delay line with variable feedback. The delay length cycles though a pre-determined sequence of durations relative to the base tempo. At the end of each delay duration, the delay line is automatically reset to the next value in the sequence. The automation process produces audible artifacts that line up precisely with the rhythmic sequence, independent of the audio contained in the delay line. When feedback is added to the delay, an additional rhythmic layer emerges as the audio from the sampled instrument is chopped up and

\footnotesize
\textsuperscript{56} Download the program, “kk_blendronomer” from the dissertation website for an example of the delay-based processed at the core of the Blendronomer instrument.
\textsuperscript{57} Dan Trueman, “Clapping Machine Music Variations.” (proceedings of the International Computer Music Conference (ICMC), New York City, June 1–5, 2010).
reassembled somewhat unpredictably according to when it enters the delay line relative to the
rhythmic sequence. The instrument also includes a variable interpolation time
parameter that controls the quality of the rhythmic artifacts that are generated with each
change in delay time. The instrument is therefore able to generate textures that repeat
with mechanistic precision yet contain all of the subtle nuances of a humanly-generated
performance. When multiple instances of the instrument are performed at the same time,
as in the piece Clapping Machine Music Variations, the expressive micro-timing of each
performer contributes to a complex combined rhythmic structure that is mechanically
precise yet lends itself to different interpretations of periodicity and phase. For example,
it is not uncommon for the performers to have completely different notions of strong and
weak beats, or where exactly the rhythmic sequence begins and ends. The composer
comments, “The behavior of this algorithm is often not intuitively predictable, but is
highly consistent and performable.”

As opposed to the automated process of the Blendronomer, my composition tree_yo uses
a timed musical machine that generates event-based changes in delay time. The
instrument uses attack detection to step through a sequence of delay settings, which can
be entered into the software either explicitly or recalled from a preset. Each step in the
sequence (labeled ‘a1’ through ‘a4’ in figure 2.3.7) contains a variable number of delays,
whose time can be set according to a base tempo or explicitly in milliseconds. Each time
an attack is detected, the delay times are reset according to the next step in the sequence,
generating the precisely notated rhythmic pattern. Performers can synchronize the

---

58 Dan Trueman, “Clapping Machine Music Variations.”
59 Download the program, “kk_tree_yo” from the dissertation website for an example of the delay-based processed used in the composition tree_yo.
rhythm of their attacks to the notated rhythmic groupings, or insert rests in between successive attacks to generate their own variations.

![Diagram of rhythm patterns generated by delays](image)

Figure 2.3.7: Rhythmic patterns generated by delays in the piece Tree Yo

The piece is written for three performers, who play the same patterns but are instructed to move in and out of phase with each other by improvising with the duration of the added rests. As with the Blendronomer, tree_yo juxtaposes precisely mechanistic delays with the expressive micro- and macro-timing generated by the input signal triggering.

The third example, the Poly_Looper, is a delay-based instrument that allows a performer to interact with the phase of a delay or loop in real-time. The software utilizes the same continuous ring buffer described in the loop-based delay, and is controlled using a footswitch mapped to record, start-loop, stop-loop, and overdub functions. In addition to the live looping functionality, however, the instrument performs an analysis of the incoming audio signal to determine the exact location and pitch of discrete note attacks within the buffer. A polyphonic implementation of the playback pointer allows the

---

60 Download the program, “kk_polylooper” from the dissertation website to experiment with the polyphonic looping technique.
instrument to generating an indeterminate number of delay taps associated with a single record buffer. Each time an attack is detected, a new playback instance is activated to begin playing at one of the indexed attack points within the buffer. Note attacks can be configured to trigger the same repeating attack index, to step through the indexes sequentially or according to a prescribed pattern, or to step through at random, as is depicted in Figure 2.3.8. Similarly, the duration of each playback instance can be a fixed amount of time or can be determined by the velocity of the detected attack, with louder attacks generating longer segments. Playback instances can also be set to play through the loop continuously, generating dense textures as multiple versions of the loop are initiated at different phases.

![Figure 2.3.8 PolyphonicLooper](image-url)
While both delay-based musical machines and sound file granulation process time-domain audio data, the traditional metaphors that describe how they operate in software differ noticeably. Whereas the skip of a broken record symbolizes the basic process at the core of sound file granulation, the record’s continuous locked groove epitomizes the process at the core of delay-based virtual machines. Similarly, while the fragmentation of audio data at the core of the granulation process resembles the discrete razorblade tape splice, delay-based processes more closely resemble the uninterrupted tape loop. While these contrasting metaphors help to illustrate some of the underlying concepts, the two techniques are increasingly realized using an overlapping set of digital tools and programming strategies. Each playback instance in the polyphonic looper, for example, can be thought of as an individual grain in an event-based asynchronous sound file granulation process. Similarly, ring buffer-based stutter effects in which playback taps rhythmically jump to random index points within the buffer are identical to synchronous sound file granulation that lacks an amplitude envelope. In real-time granular applications, when the recording buffer is large enough to capture an entire performance and the grain size is set to encompass an entire sounding event or phrase, the resulting texture can resemble a series of delays rather than a traditional granular texture.

As highlighted in the introductory chapter of this dissertation, the turntable provides tangible access to the playback of pre-recorded sound. The tape recorder and tape-based delays provide equally compelling access to both the recording and playback of sound, bringing the two formerly isolated activities into the same physical location and relative timescales. The automation associated in analog delays, which involves the tape medium
physically moving over record- and playback-sensors, couples the two processes in ways that ultimately govern how the user can engage with the mechanism. Delays in the digital medium, however, are not subject to the same physical conditions. Additional playback pointers can be instantiated, instantaneously moved, and flexibly automated in real-time. The delay processes involved in the blendronomer and tree_yo instruments, for example, are unique to the digital medium. The automation integrated into virtual delay-based TMMs adds real-time editing capabilities to the recording and playback functionality. The poly_looper program, for example, automates the activation of playback pointers at indexed attack points within an ongoing delay line. The actual sounding event is therefore governed by the performer’s triggers, and not absolute time as in a traditional delay. When taken out of real-time, this process resembles the editing techniques involved in creating a concrète piece of music – isolating sounding events, splicing them out of the recording, and then reassembling them in a new temporal sequence. By imbedding elements of the compositional process into the automation, these delay-based machines blur the lines between composition, performance, and instrument.

**2.4 Frequency Domain Musical Machines: Phase Vocoder**

As opposed to the time-domain manipulations of audio data in granular synthesis and sound file granulation, the phase vocoder is an example of a timed musical machine that operates on frequency domain audio data. Phase vocoder-based machines involve analyzing digital audio for information about the sound’s frequency content, or spectrum,
as it varies over time. The data rendered by this type of analysis can be visualized, processed for further analysis and transformation, and ultimately resynthesized back into a stream of time-domain audio data. This class of digital audio tools are also characterized as analysis-resynthesis.\(^6\)

The phase vocoder specifically implements a short-term Fourier transform, which uses a windowing function to parse a continuous stream of audio into discrete packets, which are then analyzed for their frequency content. See Figure 2.4.1 for a description of the short-term Fourier transform. The phase vocoder uses an implementation of the Fourier transform called a *fast Fourier transform*, or *fft*, that is optimized to work efficiently with digital audio data. The result is a sequence of snapshots of a sound’s frequency spectrum as it varies over time. These snapshots are called frames, and contain two spectra that correspond to the amplitudes and phases of a bank of harmonically related sinusoids. When summed together, these sinusoids accurately reproduce the original time-domain audio signal carved out by the fft’s windowing function. The window size, in samples, establishes the number of harmonics in the spectral frame, and thus determines the frequency resolution of the analysis. As in all windowed spectrum analysis, there is a tradeoff between the time and frequency resolutions. A larger window size provides greater detail in the frequency analysis at the expense of exact timing and vice versa. Phase vocoders commonly incorporate manipulations of the spectral data in real-time to alter the pitch and timbre of the processed sound. Various frame resynthesis techniques also provide control of the playback speed and position within a sequence of frames.

allowing independent control of time and pitch as well as non-linear playback.\textsuperscript{62} Both the timbral and temporal control provided by the phase vocoder make it particularly useful in manipulations of temporal index.

![Phase Vocoder Schematic](image)

\textbf{Figure 2.4.1 Phase Vocoder Schematic}

While both sound file granulation and the phase vocoder provide independent control of pitch and time, there are several key differences in how the processes are implemented. The concepts of the sequencer, scheduler, and anatomy first introduced with granular synthesis highlight these differences. For example, the anatomy and scheduler components of the phase vocoder are determined by the specific settings of the Fourier transform, which can be determined by the user in most digital audio programming environments, but are less accessible to real-time modifications. Thus, window size, shape, and overlap are generally not dynamically controlled variables as they can be in time-domain granulation processes. Similarly, the window size, in samples, must be a

power of two in order to take advantage of the more efficient \textit{fft} algorithm, further limiting the potential range of window sizes. Finally, the resynthesis process, or inverse \textit{fft}, must use the same parameters of the analysis, and the reciprocal of the windowing function that occurs in the inverse \textit{fft} results in a scheduler that can only generate synchronous streams with even phase distribution.

While the data contained in a sequence of spectral frames is different than an indexed array of amplitude samples, it can nevertheless be stored into memory and accessed using a similar set of time-domain tools and programming approaches. The phase vocoder can effectively read and write frequency-domain data into a time-domain audio buffer or wavetable by using an appropriate bit-depth to accommodate the range of values produced by the \textit{fft} analysis, and indexing the data in a systematic way that corresponds to the size of the spectral frames. The sequencer component of the phase vocoder can therefore be expressively manipulated in many of the same ways as sound file granulation, including non-linear playback, feature-based sequencing, and the concatenation of different sound sources. Several strategies exist for compensating for the synthesis of non-adjacent analysis frames, such as the tracking phase vocoder and phase accumulation techniques.\footnote{Thorsten Karrer, Eric Lee, and Jan Borchers, “PhaVoRIT: A Phase Vocoder for Real-Time Interactive Time-Stretching,” (paper presented at ICMC, Tulane University, November 6-11, 2006).}

Additional advantages of the phase vocoder include more sophisticated processing of the spectral data such as formant-based pitch shifting, spectral delays, spectral smearing, cross-synthesis and morphing between different sounds. Similarly, researchers have
developed algorithms that can adapt to the types of sounds being analyzed, resulting in more accurate analysis and life-like transformations. While a great deal of research and development has gone into creating more accurate and transparent time- and pitch-shifting applications, the basic techniques used in the phase vocoder can also be used to create more esoteric sounds and textures that highlight the mechanisms of the analysis and resynthesis processes at its core.

The following TMMs demonstrate two different applications of the phase vocoder’s frame resynthesis technique, both using the same piano sample from the sound file granulation Example 2.2.3. The first TMM maps the mouse’s horizontal screen position to the buffer index position within the sequence of spectral frames. Fractional frame values are achieved by interpolating between the amplitude and phase information of adjacent frames, resulting in a spectral smearing effect unique to frequency-domain manipulations of audio data. The second example, “preset 2 – phase_vocoder_rhythm,” illustrates an automated non-sequential frame resynthesis technique. Instead of continuous input from the mouse position as in the previous example, this TMM automates the changes in spectral frame according to a randomized offset signal sampled at regular intervals. The expressive parameters are the offset range, in frames, and the sampling interval, in milliseconds. Figure 2.4.2 demonstrates the rhythmic texture generated by 500-millisecond sampling interval and 20-frame offset, which is characterized by distinct changes in both volume and timbre as the resynthesis frame jumps non-linearly to new random positions. TMMs labeled, “preset 3 - vertical_map_discrete,” and “preset 4 - vertical_map_continuous,” utilize the vertical

64 Open the kk_phasevocoder application, and select “preset 1 - horizontal scrub” from the preset menu.
mouse position to modulate the sampling interval, first in quantized rhythmic values and then in a continuous mapping. While these applications of the phase vocoder do not simulate a life-like sustain of the piano sample as in the granular examples, they draw attention to the underlying processes and automation, eliciting a form of performed listening, which is described in the following chapter. Specifically, rhythmic shifts in timbre function as a form of cognitive mismatch negativity (MMN),\textsuperscript{65} shifting attending focus alternatively between the surface of the sonic texture and a hypothetical higher-level rhythmic pattern generated by the implied strong and weak accented subdivisions.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{temporal_quantized_random_frame_resynthesis.png}
\caption{Temporally Quantized Random Frame Resynthesis Using the Phase Vocoder Technique}
\end{figure}

• Freezing Time •

3.1 Freezing Time

“The focal change from the note proper to the internal behaviour of components has furnished a new musical potential, since the electroacoustic medium makes viable the composition, decomposition, and development of spectral interiors.” 66

“Musical sounds are inextricably bound up with this experiencing of time passing as interpreted through changes in spectral shape.” 67

- Dennis Smalley, from Spectro-Morphology and Structuring Processes

Software-based timed musical machines have the flexibility and precision to operate on wide range of time scales. Exploiting the fragmented nature of digital audio, these virtual machines are able to provide independent control of a sound object’s time and pitch by decoupling the temporal index from the particular method of playback. The resulting manipulations of recorded sound, which I call Freezing Time (FT), ultimately transform the sonic object into an expressive instrument. The technical processes used to achieve FT encompass a diverse set of virtual tools, and are capable of generating a wide array of sounds from fast rhythmic textures to slowly shifting drones and soundscapes. These virtual machines operate on the temporal extreme defined by processor cycle speeds and sample-based durations, utilizing increasingly efficient signal processing algorithms and more powerful computer hardware. As the digital tools migrate from the studio to the stage, FT enables the real-time control of time and pitch on the micro-time scale, allowing both parameters to be performed as expressive musical parameters.

67 Ibid., 68.
As the name implies, FT employs various techniques that intervene in the way sound naturally unfolds in time, providing dynamic and real-time control over spectral interiors and changes in spectral shape described in the quote at the beginning of this chapter. The virtual playhead, a graphical line that displays the advancing of time as it moves across a segment of recorded sound, provides an illustrative metaphor for FT. When controlled by the DAW’s global clock, for example, the playhead functions solely as a visual feedback device that displays the current position in absolute time within the project. Manipulations of FT, on the other hand, allow the performer to take control of the playhead, effectively hijacking absolute time by scrubbing through sound in real-time. The practice of audio scrubbing originated with reel-to-reel analog tape systems as a means of navigating recorded audio and video media quickly and efficiently. Speeding up or slowing down the tape in these analog systems changes the sounding pitch, as the movement of the tape over the playhead is necessary to generate the sound or image.\(^ {68}\) The various methods of digital scrubbing utilized in FT allow independent control of time and pitch and can be exploited for a wide range effects including simulating infinite sustain and sonifying temporal stasis.

Granular synthesis is able to generate “expressive turbulence, intermittency, and singularity,”\(^ {69}\) and is therefore particularly well suited for FT transformations of complex real-world sounds. Adding a small percentage of random noise to parameters such as

---

\(^{68}\) Eric Lee, Thorsten Karrer, and Jan Borchers, “Improving Interfaces for Navigating Audio Timelines,” (paper presented at CHI: Workshop on Supporting Non-Professional Users in the New Media Landscape, San Jose, USA, April 2007).

\(^{69}\) Roads, *Microsound*, 86.
grain size, playback rate, or playback position animates the synthesis algorithm on the micro timescale, and is particularly important in digital scrubbing when the temporal shape is controlled externally. These deviations function as a temporal dither, and are essential in effectively processing sound with complex harmonic makeup. Figure 3.1.1 demonstrates the sound file granulation of a piano sample first with static variables (A), and then with a temporal dither of 20ms applied to the buffer index position (B). Both examples use an 80ms grain size. The dither masks the audible artifacts such as phasing and flanging that the overlapping windowing process generates, and provides subtle spectral variation that gives the sound a life-like quality.

![Figure 3.1.1 Temporal Dither in Sound File Granulation](image)

The mechanics of FT involve two components. The first is the specific playback implementation that allows the audio data to be synthesized with independent control of temporal index and pitch (see sections 2.2, 2.3, and 2.4 for technical descriptions). There are many different approaches to achieve FT, each with its own set of control parameters
and varying degrees of transparency on the processed sound. The second component of FT involves the various ways of controlling the many parameters of the synthesis algorithm, and incorporates aspects of human computer interaction, custom hardware controllers, instrument building, mapping, and the real-time use of automation data (see sections 2.2, 2.3, and 2.4 and Appendix A for examples). In addition to mapping input and automation data directly to synthesis parameters, a secondary layer of interaction and control that functions on the macro-time scale provides an important bridge between higher-level musical descriptors and the various mechanisms that function on the micro-time scale. Object-based compositional structures associated with acoustic music with a primary focus on texture, timbre, and process provide a compelling metaphor for this type of secondary control. The following section therefore explores György Ligeti’s compositional process called net-structures as a means of simulating different methods of controlling and interacting with FT-based TMMs.

3.2 Net-Structures

“I favor musical forms that are less process-like and more object-like. Music as frozen time, as an object in an imaginary space that is evoked in our imagination through music itself”

- György Ligeti

During the 1960s, while working with Gottfried Michael Koenig at the electronic studio in Cologne, Ligeti began exploring the psychoacoustic effect of generating continuous...
musical textures from discrete sounds or tones.\footnote{Richard Steinitz, György Ligeti: Music of the Imagination (Northeastern University Press, 2003), 165.} In one particular experiment, he was able to create a sustained tone by splicing together very short segments of recorded tape in rapid succession. If the source material had the same pitch but contrasting timbre, then the process coalesced into single complex tone. If the source material contained different pitches, then a chord emerged. The threshold of this coalescence, he found, was approximately 18 to 20 events per second, which corresponds to the perceptual rate at which a succession of impulses transforms from a rhythm to a pitch.\footnote{Steinitz, György Ligeti, 165.} Although several composers used tape splicing techniques to create proto-granular compositions, the technical apparatus to implement this type of process in a robust compositional manner with recorded media was still years away (see section 2.2 on granular synthesis). Ligeti nevertheless incorporated the underlying granular concept in his scored acoustic work. In the program notes for his 1968 piece for solo harpsichord, Continuum, he writes:

I thought to myself, what about composing a piece that would be a paradoxically continuous sound, something like Atmosphères, but that would have to consist of innumerable thin slices of salami? A harpsichord has an easy touch; it can be played very fast, almost fast enough to reach the level of continuum, but not quite (it takes about 18 separate sounds per second to reach the threshold where you can no longer make out individual notes and the limit set by the mechanism of the harpsichord is about 15 to 16 notes a second). As the string is plucked by the plectrum, apart from the tone, you also hear quite a loud noise. The entire process is a series of sound impulses in rapid succession, which create the impression of continuous sound.\footnote{Jane Piper Clendinning, “The Pattern-Mecchanico Compositions of Gyorgy Ligeti,” Perspectives of New Music, Vol. 31, No. 1, (Winter 1993), 194.}

In addition to colorfully describing the defining sound of the piece, the program note reveals Ligeti’s focus on timbre and texture, which are informed by his experiments with electronic and tape-based music. If the goal were simply to create a sustained musical texture, Ligeti could have chosen a more suitable instrument, perhaps a string instrument with continuous bowing or a wind or brass instrument with a circular breathing...
technique. Instead, Ligeti chose the harpsichord, whose individual attacks are discrete events, like segments of tape or “slices of salami,” with both a pitched tone and a “loud noise.” The basic compositional concern, therefore, became how to translate his experiments with splicing segments of tape together in the studio into a notated score that could be performed acoustically by a person on the stage.

Around the same time in the middle of the 1960s, Ligeti began to establish his *meccanico* style, which he described as “the ticking of malfunctioning machinery.”74 This compositional style first emerged with his metronome experiments like *Poème Symphonique*, and the *Satanic Clocks* movement from *Nouvelles Aventures*. *Meccanico* style figured predominantly in later works like Study No. 2 for Organ, *Coulee*, the third movement of his Second String Quartet, and the third movement of his Chamber Concerto.75 In the score to *Continuum*, for harpsichord, he indicates that in order to achieve the desired continuous texture, rhythm and tempo should serve a functional rather than musically expressive role.

“Prestissimo = extremely fast, so that the individual tones can hardly be perceived, but rather merge into a continuum. Play very evenly, without articulation of any sort. The correct tempo has been reached when the piece lasts less than 4 minutes…The vertical broken lines are not bar lines - there is neither beat nor meter in this piece - but serve merely as a means of orientation.”76

The tempo is defined by the time it takes to complete the piece, setting the baseline for a successful performance and challenging performers to achieve ever faster, more continuous realizations. While the piece contains no notated beat or meter, the *meccanico* technique nevertheless produces perceptible rhythms and patterns. Ligeti

---

74 Steinitz, György Ligeti, 98.
75 Ibid.
comments that, “the actual rhythm of the piece is a pulsation that emerges from the
distribution of the notes, from the frequency of their repetitions.” Ligeti thus
establishes several simultaneous time scales. The individual note level, which mimics the
mechanism ticking at the perceptual border between pulse and tone, is intended to be
functional and structural. The perceptible rhythmic time scale is provided by the
psychoacoustic effects generated by transformations in the specific patterns of notes.

A focus on textural density and a rhythmic language grounded in mechanical systems
allowed Ligeti to experiment with formal, texture-based compositional structures, which
he calls ‘net-structures’. This compositional framework reflects Ligeti’s belief that,
“composition consists principally of injecting a system of links into naïve musical
ideas.” In his work throughout the 1960s, these naïve musical ideas consist of ‘interval
signals’, which are simple dyads or trichords representing stable points of rest between
areas of flux or transition, or ‘links’. Ligeti’s net-structures therefore provide a
systematic means of generating transitions and contrasting areas of stability based on
texture and timbre without having to rely on, “pairs of opposition in traditional tonal
music,” such as tension and release through functional harmony or the juxtaposition of
consonance and dissonance. Ligeti uses the evocative term ‘mist’ to describe the effect
that these links produce when transitioning between stable signals.

“Signals are divided by blurred areas, so that you hear an interval [or signal] that gets
gradually blurred and in the ensuing mist another interval appears, it becomes clearer and
clearer until the surrounding mist completely clears, and you hear the new interval [or signal]
all by itself.”

---

78 Hicks, “Interval and Form,” 173.
79 Ibid., 174.
80 Ibid., 174.
Ligeti uses several techniques to generate the characteristic mistiness between stable signals. In the piece *Continuum*, for example, he uses filling, adding new pitches inside an interval, accretion, adding new pitches outside of an interval, and shifting, in which one or more notes of the interval more up or down. See Figure 3.2.1 for examples of each type of interval manipulation extracted from the score.

![Figure 3.2.1 Examples of Accretion, Filling, and Shifting from Ligeti’s Continuum](image)

Both the harmonic and rhythmic shape of the signal change when a pitch is added or removed through either filling or accretion. As the piece employs a continuous texture at the level of the individual note, the rhythmic shape of a signal, defined by the number of notes it contains, provides an important functional component. As the piece is structured with separate left- and right-hand signals, transformations of filling or accretion can provide an additional layer of ‘mistiness’ by offsetting when the changes in rhythmic shape occur in each hand.

Standard notation can obscure the underlying object-based transformations at work in net-structures. A ‘piano-roll’ style of viewing the score, in which notes are arranged
vertically according to pitch and horizontally on a timeline, can provide a more illustrative picture of the systems of signals and transitional links.\textsuperscript{81}

In the first line of Figure 3.2.2, mm 9 to 27, the initial signal of a minor third (G, Bb) is gradually blurred first through accretion with the addition of the F in the right hand, and then filled with an Ab in the left hand. The process of filling and accretion continues through mm. 28, yielding two 5-note clusters that contain all the chromatic pitches spanning F# to Cb. The left- and right-hand figures then gradually collapse through a process of shifting and the corresponding reduction of duplicate notes until both hands arrive at the second stable signal of the piece (F#, G#) in measure 50.

\textsuperscript{81} Hicks, “Interval and Form,” 189.
The piano roll visualization is similarly useful in illustrating aspects of the overall structure and form of the piece. The transitional links clearly contrast the stable signals in the image, and the bifurcation that begins at the middle of the piece as well as the ultimate thinning out of texture at the end are particularly evident in the visual medium (see Figure 3.2.3).

![Figure 3.2.3 Piano Roll View of Continuum: Full Score](image)

Figure 3.2.4 and the corresponding audio example illustrate several examples of a granular-based timed musical machine replicating the process of accretion from Figure 3.2.1A. Instead of using a singular rate of playback value for the grain anatomy, this machine cycles through an array of pitch intervals, which correspond to the pitches in the net-structure’s signal. The grain anatomy therefore adopts a form of automation based on the arpeggiator functionality of synthesizers and sequencers. The pitch array can be updated in real-time as the machine is running, effectively blurring and re-focusing the signal. The following six examples illustrate the pitch array changing from (0, 3) to (-2, 0 3) using six different granular densities. As with the acoustic performance on the harpsichord in Continuum, in the 10, 15, and 20 grains-per-second (gps) examples, one
hears the secondary rhythmic pattern emerge as the signal expands from 2 to 3 pitches. Notions of stability and mistiness, of creating a continuous texture from discrete impulses, and of rhythm emerging out of texture that are derived from net-structures therefore aptly apply to this granular musical machine.

![Figure 3.2.4 Sound file granulation using accretion method at various granular densities](image)

The scheduler mechanism of the virtual granular machine can achieve rates that exceed the physical limitations of acoustic instruments and human performance. The 40, 80 and 120 gps examples, for example, produce changes in timbre and harmony, as the grains fuse together to form a sustained chord. Granular density can therefore contribute to the perceived mistiness or stability of a signal, resulting in changes in timbre as well as vertical harmony. Granular density can also be manipulated in real-time (see section 2.4, ‘preset 10 – density_control’). Figure 3.2.5 demonstrates a continuous controller manipulating granular density in the three-note signal from Figure 3.2.4, seamlessly
crossing over the perceptual threshold from rhythmic pattern to sustained chord and back again.

![Figure 3.2.5 Manipulating Granular Density with 3-Note Signal](image)

The pitch content of the signal, and how individual notes behave in processes of accretion, filling, and shifting are also freely manipulated using sound file granulation. For example, individual elements - or voices - contained in the signal can move continuously across the pitch space in processes of shifting by interpolating changes in the granular rate of playback. Individual voices can also bifurcate or coalesce in a similarly continuous manner in processes of accretion or filling. This type of granular functionality is well documented in the ramp-based control structures in Barry Truax’s POD, PODX, and GSAMX real-time granular applications. Figure 3.2.6 and the accompanying audio excerpt demonstrate granular-based shifting, accretion, and filling using a continuous pitch space and a five-second interpolation time.

---

82 Barry Truax, “Composing with Real-Time Granular Sound” Perspectives of New Music, Vol. 28, No. 2 (Summer, 1990), 120-134.
Figure 3.2.6 Accretion, Filling, and Shifting Using Continuous Pitch Space

The granular-based musical machines used to generate the audio in Figures 3.2.5 and 3.2.6 incorporate a mixture of automation data and real-time input or intervention. In the first example, the granular sequencer and anatomy are fully automated, while aspects of the scheduler such as granular density and overlap are manipulated expressively and in real-time. In the second example, parameters such as density and grain size are fixed, the rate of playback is automated using a linear interpolation function, and the specific collection of pitches is manipulated in real-time. The automation mechanisms involved in these granular machines therefore link lower-level synthesis parameters to intermediary and higher level descriptions of sound and compositional structure.

Net-structures can also be applied to the sequencer component of granular-based musical machines, generating a continuous spectrum from stable to increasingly ‘misty’ textures by automating how the source material is accessed. One approach is to use a control signal - either automated or manipulated in real-time - to modulate the amount of temporal dither applied to the sequencer’s buffer index location. The amount of temporal dither, in milliseconds, determines the range of random values that can be added to the
buffer index position. It therefore describes the potential area from which the sequencer may extract a grain. A small amount of dither - approximately 20ms - generates a life-like and relatively steady timbre as described in Figure 3.1.1. As the amount of dither is increased, however, the range extends into other areas of the sample, producing increasingly active and dynamic textures. The effect is particularly pronounced when applied to sounds with a clear contrast between the attack and steady state, or when the granulated sound contains more than one pitch in sequence. Figure 3.2.7A and the corresponding sound file depict 20ms and 100ms dithers applied to a sample containing several different pitches struck on the piano. While the 20ms range is limited to the sustained and steady area of the central note, the larger range clearly encompasses the attack of the central note as well as the attack and sustained areas of several of the adjacent notes. The result is a texture that embodies Ligeti’s concept of ‘mistiness’ in both the timbre and pitch content. Part B of Figure 3.2.7 and the corresponding sound file demonstrate a smooth interpolation between the two dither ranges over the span of sixteen seconds.
Figure 3.2.7: Temporal Dither at 20ms and 100ms

Figure 3.2.8 and the corresponding sound file illustrate a more sophisticated and dynamic control of temporal dither using a granular machine that uses a live audio input as a control signal. More specifically, the machine employs an envelope follower to map the amplitude of the live input signal - in this case a contact microphone on the soundboard of an acoustic piano - onto the range parameter of the temporal dither. As in Figure 3.2.7, the buffer index is set to a sustained area of the note recorded in the buffer. Higher amplitudes in the input signal generate more active textures in the granulation as the dither range extends into the note attack area and adjacent note areas. The synthesized texture returns to a steady state as the amplitude of the input signal naturally decays after each attack. Lengthening the release time of the envelope follower produces granular reverberation and echo-like effects, as the dither takes more time to return to the resting state than the input control signal.
The second approach to manipulating the granular instrument’s sequencer in real-time in the context of Ligeti’s *net-structures* involves controlling precisely how and when sound is copied into the granulation buffer. In addition to modulating the amount of temporal dither, the envelope follower in the granular machine from Figure 3.2.8 can be used to regulate when audio is recorded into the buffer. When the amplitude of the input signal crosses a set threshold - corresponding to a note onset - audio is recorded alternately into one of two buffers. The audio is also slightly delayed - approximately 100 ms - to ensure that the attack of the note is not cut off due to the latency of the envelope follower’s attack detection. Both buffers are granulated at the same time using the same set of control parameters described in the preceding paragraphs, resulting in sustained dyads that contain the previous two notes. When a new attack is detected, audio already stored in the buffer is either overwritten or copied back into the buffer according to a feedback variable. The performer can therefore generate harmonic stability or ‘mistiness’ by choosing a particular sequence of pitches, and controlling how much of the previously
recorded notes are retained each time a new attack is detected. This processes is illustrated in Figure 3.2.9 and the accompanying sound file.

As the feedback level increases, the resulting granular textures become dense, harmonically blurred, and chaotic. Whereas the granular textures in Figure 3.2.9 part A contain only the two most recent pitches, the chords in part B cumulatively contain all of the preceding pitches. A feedback value greater than zero but less than one produces a temporally non-linear fade-out that is dependent on the timing of subsequent triggers as opposed to absolute time. The texture can also coalesce into a stable signal by eliminating the feedback and recording new pitches, generating a steady-state granular texture that contains only the new pitches. Examples 3.2.8 and 3.2.9 use two buffers to generate dyads and build up chords, but the instrument can easily be scaled to accommodate additional granulation buffers resulting in more complex harmonies and modes of interaction. For example, the velocity of the note attack can be used to determine which buffer to record into, and what type of granulation process to apply.
This process is sometimes called “selective granulation,”\textsuperscript{83} and is closely related to multi-sampling techniques described in Section 1.2.

In discussing different approaches to working with complex synthesis algorithms, Michael Klingbeil, the author of the spectral editing and analysis program called SPEAR, makes the following observation.

Because interesting musical sounds are rarely static, nor entirely stochastic, the precise control parameters must be specified at every instant. If this control information is not a natural result of the synthesis procedure in use, the result is an unwieldy data explosion with separate detailed control streams for each sonic parameter of interest.\textsuperscript{84}

Net-structures therefore provide an additional layer of control and organization to software-based musical machines by acting as an interface between low-level synthesis routines and high-level interaction. The examples presented in this section demonstrate how control processes based on net-structures can apply to granular-based musical machines manipulated in real-time. The combined use of automation and real-time input therefore prevents the “unwieldy data explosion,” and provides a tangible metaphor for how to engage with the virtual mechanisms.

\textsuperscript{83} Roads, \textit{Microtime}, 191.
\textsuperscript{84} Michael Klingbeil, “Spectral Analysis, Editing, and Resynthesis: Methods and Application” (PhD Diss., Columbia University, 2009), 4.
3.3 Freezing Time as Performed Listening

“Already his subtle direction of time gives the impression that it is our listening attention, not the composer’s manipulation, that creates the horizontal dynamic.”  

- Katharine Norman on Paul Lansky’s *Night Traffic*

Processes of FT allow a performer to influence the way a listener attends to sound when experiencing a piece of live music. In her article, “Real-World Music as Composed Listening” the composer and theorist Katharine Norman describes an approach to structuring and analyzing music in which, “listening is as much a ‘material’ for the composer as the sounds themselves.” In this compositional framework, the physical, cultural, and contextual significance of recorded real-world sounds are considered along with their pure sonic or timbral characteristics. Expanding on Schaeffer’s description of reduced listening, Norman therefore adds referential, reflective, and contextual modes of listening. These added modes emphasize the active and participatory experience of listening as opposed to focusing exclusively on the sound’s intrinsic sonic qualities.

Real-world tape music still, and primarily, celebrates that internal fusion of listening and imagination. In fact it depends on our listening participation and invites us - through our active, imaginative engagement with ‘ordinary’ sounds - to contribute, creatively, to the music. In listening to a piece of real-world music we employ, and develop, the ‘non-musical’ strategies that we ordinarily use, in addition to our more rarefied musical sensibilities.

FT has the potential to elicit the same perceptual shifts in attending to sound during a performance, resulting in what I call *Performed Listening*.

---

86 Ibid., 5.
87 Ibid., 5-19.
88 Ibid., 4.
Sound emitted from a loudspeaker during a performance can be interpreted in several ways, each with its own implied listening strategy. On the most basic level, the unprocessed reinforcement of an acoustic sound functions to help the listener ‘hear’ the performance better. Even this type of seemingly transparent acoustic reinforcement, however, has the potential to elicit reduced and reflective modes of listening. Non-trivial spatalization resulting from dramatically separating the speaker from the acoustic source, or exaggerated amplification, dynamics processing, or equalization can result in a more active type of listening, attending to both the “more rarefied” musical object and the more basic sound object. The various modes of listening can be further manipulated by processing the acoustic sound (reductive), introducing prerecorded sound not generated in real-time (reductive and reflective), and revisiting sounds sampled from earlier in a performance (contextual). TMMs that incorporate delays or the granulation of audio captured in real-time similarly possess the ability to initiate these different modes of listening.

A sound with a strongly defined temporal envelope, like the staccato note played on the piano from the temporal dither examples from Section 3.2, provides an illustration of how manipulations of FT can elicit changes in the different types of listening behavior. By avoiding the brighter areas associated with the attack of the note and navigating the temporal index of resynthesis towards more sustained and resonant areas, FT generates a continuous and sustained texture from a sound with a naturally fixed spectral envelope. Furthermore, granular machines with multiple record buffers or record-based feedback can create sustained chords from individual notes sampled in real-time. A single note
attack is thus translated from the linear to the vertical pitch space (see Figure 3.2.9.B). This in turn suggests the listener adopt a more reductive mode of listening, focusing their attention on the surface texture of the sound and not on how any one note might fit into a temporally linear structure like a melody or phrase. The function of melody in the performance therefore becomes a process of populating vertical harmonies. This in turn influences both how the audience experiences the sequence of pitches in time, and how the performer executes them.

The approach to FT from the phase vocoder Figure 2.4.2 superimposes rhythms and pulses onto an otherwise sustained texture by modulating the re-synthesis frame randomly at regular intervals. A similar effect is achieved in sound file granulation by utilizing an amplitude envelope with an exaggerated attack or decay, or by removing the envelope altogether revealing the transient clicks produced in the sequencing process. Inserting time between successive sounding events when using a pulse-based sequencer produces a similar result. The tempo and timbral variation produced by this effect can be controlled and manipulated by the performer in real-time using a technique similar to the temporal dither example in Figure 3.2.7. This type of FT manipulation draws attention to the mechanics of the synthesis technique, eliciting changes in attending to sound that alternate between a focus on timbre, vertical harmony, and aspects of rhythm and pulse.

By shifting the focus towards subtle and nuanced aspects of timbre and micro-timing, manipulations of FT in real-time elicit the type of heightened attention to sonic detail associated with contextual listening that is usually achieved through familiarity with the
recording from repeated listening. FT therefore allows the user to assert a more active role in the listening experience, challenging the ways that we attend to both musical and real-world sounds.

3.4 Freezing Time: Conclusions

FT involves three important compositional and technological threads based on the evolving ways that sound has been understood as an object over the course of the twentieth and into the twenty-first centuries. First, FT is the real-time articulation and animation of object-based compositional structures. Second, FT produces perceptual shifts in the different ways of attending to acousmatic sound, such as reduced, referential, reflective, and contextual modes of listening. Finally, realizations of FT in the form of performance-based TMMs represent a compelling example of how digital recording and playback technology continues to exist in a symbiotic relationship with broader music-making practices.
Appendix A: Dissertation Software

The following appendix provides documentation of the software used to generate the examples in this dissertation. Collectively they illustrate many key concepts of software-based TMMs, and therefore constitute an important body of supplemental material. All of the programs are stand-alone applications created with the visual programming environment Max, version 6.1.9, and can be downloaded at www.konradkaczmarek.com/tmm. I have attempted to make the software as intuitive as possible, labeling each synthesis and control parameter directly in the user interface. In addition to interaction with the graphical interface as described in the following sections, the synthesis parameters can be accessed via networked messages, allowing the software to be manipulated by outside controller data. The individual patches that make up the applications are included in the download, and I have attempted to structure them so that they can be easily loaded into existing patches as abstractions or bpatchers. Finally, in addition to the information in this appendix, the patches themselves contain extensive documentation commented directly into the code that is visible in non-presentation mode.
All of the following software contains a ‘presets’ button, which opens a graphical interface for recalling and saving different configurations of the software. Whenever applicable, I had included the presets that were used to generate the examples in the dissertation. These presets are labeled according to the chapter and section they are found, or by a description of the TMM for which they were used.
A1: kk_granular

The kk_granular program provides a single platform for the different types of sound file granulation described in chapters 2 and 3. Individual grain playback is performed by an abstraction made with the gen~ library that is loaded into a poly~ object for handling an indeterminate number of grain instances. This implementation provides the flexibility required for asynchronous control as well as sample-accurate triggering for synchronous applications. The amplitude envelopes of individual grains are animated with green dots where they occur on the waveform, providing an important visual representation of the granulation process. Clicking the ‘view sonogram’ button opens a separate window with a corresponding image of the sound’s frequency spectrum as it varies over time. All of the synthesis parameters are labeled directly in the graphical interface.

![kk_granular User Interface](image)

The buffer-offset parameter of the sequencer, represented by the red line in the waveform window, can be set in a number of different ways. Clicking directly on the waveform
automatically sets the ‘position – phase’ parameter. Values can also be explicitly entered into the number box. A ‘position – rate’ parameter automates the buffer-offset position, and the specific value determines the rate that the sequencer moves through the buffer. The ‘position – dither’ parameter represents the temporal dither applied to the sequencer position.

The user can navigate the audio buffer visually by holding down the command-key and dragging on the waveform. The up- and down-direction magnifies, and the left- and right-direction scrolls through time. These settings can also be set explicitly using the ‘selection start’ and ‘selection duration’ number boxes (these values are also saved with each preset settings). The ‘select all’ button resets the view back to the entire buffer.

Grains are triggered individually by clicking on the ‘Manual Trigger’ button or automated as a stream by activating on the ‘Pulse Generator’ button. The pulse Generator’s rate is determined by the ‘density (g/s)’ number box, and a corresponding ‘dither’ value provides quasi-synchronous deviation. Audible clicks are generated each time a grain is triggered in either synchronous or asynchronous modes, which can be auditioned by adjusting the ‘trigger’ volume slider.

Grainsize and grainsize dither are controlled using the corresponding labeled number boxes. Each grain has a volume and pan setting determined by the range sliders, which set the minimum and maximum boundaries within which a random value will be selected at the onset of the grain event. Clicking and dragging establishes a range, and holding
down the shift-key and dragging allows you to modify the existing range. The amplitude envelope applied to each grain can be changed by pressing the ‘grain window’ button, and then choosing from the menu of window shapes or drawing directly into the waveform editor window.

The particular way that the software interprets pitch information is determined by the ‘Pitch Map Behavior’ menu. The ‘spread’ setting uses the ‘pitch center’ and ‘spread’ parameters to define a singular rate of playback with a noise-based modulation range, similar to a dither. The ‘step’ and ‘random’ modes adjust rate-of-playback based on semitone sequences entered manually into the ‘pitch list’ field or performed using a MIDI keyboard (middle C, MIDI note 60, corresponds to rate of playback 1.0). Finally, the ‘pitch interpolation time’ determines the time it takes to ramp to new rate-of-playback values in either ‘spread’, ‘step’, or ‘random modes’.

The ‘buffer controls’ button opens a window with information about buffer length, recording input signal, and recording feedback level as well as buttons to trigger recording, load in new samples, and clear the buffer.
All of the following parameters can be controlled externally by sending the corresponding UDP messages to the port designated at the top right of the user interface (the default port is 8080).

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>granular_density</td>
<td>position_rate</td>
</tr>
<tr>
<td>density_dither</td>
<td>position_dither</td>
</tr>
<tr>
<td>manual_trigger</td>
<td>record_feedback</td>
</tr>
<tr>
<td>pulse_generator</td>
<td>record</td>
</tr>
<tr>
<td>pulse_reset</td>
<td>buffer_clear</td>
</tr>
<tr>
<td>grainsize</td>
<td>buffer_length</td>
</tr>
<tr>
<td>grainsize_dither</td>
<td>trigger_volume</td>
</tr>
<tr>
<td>pitch_mode</td>
<td>granular_volume</td>
</tr>
<tr>
<td>pitch_center</td>
<td>pan_range</td>
</tr>
<tr>
<td>pitch_spread</td>
<td>volume_range</td>
</tr>
<tr>
<td>pitch_interptime</td>
<td>waveform_start</td>
</tr>
<tr>
<td>pitch_list</td>
<td>waveform_length</td>
</tr>
<tr>
<td>position_phase</td>
<td></td>
</tr>
</tbody>
</table>

The patch ‘send_data.maxpat’ demonstrates how to properly format messages, and ‘key_trigger.maxpat’ provides a simple example of how to map keystroke triggering to the asynchronous granular scheduler via the ‘manual_trigger’ message. Finally, the kk_tether_control application from Chapter 2 provides is a more comprehensive example of routing input data into the kk_granular program, incorporating several different mapping strategies and preset configurations.
A2: kk_ringbuffer

The kk_ringbuffer program uses a combination of buffer-based recording and playback patches created with the gen~ library. As in the granular program, the playback patch is loaded into a poly~ object that allows the user to dynamically add and remove instances from the signal processing chain in real-time. The multislider objects labeled ‘Delay Times’, ‘Delay Levels’, and ‘Delay Panning’ and the corresponding text-edit fields allow the user to set various parameters of the playback pointers. The buttons at the bottom of the interface allow the user to create precise echoplexes, distributing the playback pointers evenly across the entire buffer, or a set duration.
The kk_livelooper program uses the same underlying buffer-based delay patch as kk_ringbuffer, but adds feedback and automation functionality for live looping applications. The sliders directly below the record and feedback buttons allow the user to enter non-binary values for the two parameters, and the ‘start loop’ and ‘stop loop’ buttons provide automated control over the delay time, record level, and feedback amount.
The kk_blendronomer software uses the same underlying buffer-based delay patch to illustrate the types of automation used in Dan Trueman’s Blendronomer instrument. The start/reset and stop buttons initiate the automated rhythmic changes in delay time described in section 2.3 of this dissertation. The pulse time, specific pattern, and ramp time parameters are edited using the labeled user interface fields.
The kk_treeyo program provides a simple demonstration of the event-based changes in delay described in chapter 2. The patch steps through a sequence of delay states according to attack-detection performed on the ‘trigger input’ signal. The sequence of delay states and the corresponding changes in pitch are entered manually into the patch using the labeled text fields. The user has control over pulse rate and input/output level. Finally, the preset section provides several illustrative examples of the instrument that should serve as a starting point for experimentation.
**A6: kk_polylooper**

The kk_polylooper program represents the most intricate application of ring-buffer based delays in a TMM. Please refer to example 2.3.8 from Chapter 2 for a thorough explanation of the automation process. The ‘trigger mode’ menu allows you to select how the detected attacks trigger the delay-based playback. The options are ‘random’, which jumps to random attack index points as illustrated in example 2.3.8, ‘repeat’, which triggers the same attack index indicated in the ‘repeat trigger number’ field, and ‘step’, which sequentially steps through all of the detected attack indices.

The ‘signal trigger’ toggle turns on and off the automatic attack detection on the indicated input signal. Triggers can be manually generated using the ‘t’ key on the computer keyboard, or triggered in a pulse by pressing the ‘automate trigger’ button and setting the pulse speed.
An ‘attack offset time’ parameter allows you to adjust the timing of the playback pointers relative to the attack indices, and is particularly important when, for example, when playback back the delays in reverse. The ‘delay envelope’ toggle button initiates the enveloped playback instances, and when the button is turned off, all voices will sound continuously. Finally, the number of playback instances is set using the ‘number of voices’ field.
Finally, the kk_phasevocoder program plays audio based on buffered spectral analysis data and phase-accumulation frame resynthesis techniques. The position within the sequence of frames can be automated according to a rate of playback parameter, or manually entered into the position field or by clicking and dragging directly on the waveform image. The ‘random position rate’ and ‘random position range’ parameters allow the software to generate rhythmic patterns based on discrete changes in frames. Finally, the pitch of the playback can be manipulated with the ‘pitch’ parameter.

Figure 10: kk_phasevocoder User Interface

A7: kk_phasevocoder

Finally, the kk_phasevocoder program plays audio based on buffered spectral analysis data and phase-accumulation frame resynthesis techniques. The position within the sequence of frames can be automated according to a rate of playback parameter, or manually entered into the position field or by clicking and dragging directly on the waveform image. The ‘random position rate’ and ‘random position range’ parameters allow the software to generate rhythmic patterns based on discrete changes in frames. Finally, the pitch of the playback can be manipulated with the ‘pitch’ parameter.
Bibliography


Lee, Eric, Thorsten Karrer, and Jan Borchers. “Improving Interfaces for Navigating


Truax, Barry. “Composing with Real-Time Granular Sound” Perspectives of New Music, Vol. 28, No. 2 (Summer, 1990), 120-134.


